To the Graduate Council:

I am submitting herewith a thesis written by Moinak Chatterjee entitled "Providing Customized Real Time Traffic Information Through the Internet: Implementation Using GIS." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Geography.

Shih-Lung Shaw

Major Professor

We have read this thesis
and recommend its acceptance

Bruce Ralston

Cheng Liu

Accepted for the Council:

Anne Mayhew

Vice Provost and Dean of Graduate Studies
(Original signatures are on file with official student records)

PROVIDING CUSTOMIZED REAL TIME TRAFFIC INFORMATION THROUGH THE INTERNET: IMPLEMENTATION USING GIS

A Thesis
Presented for the
Master of Science
Degree in Geography
The University of Tennessee, Knoxville

Moinak Chatterjee
May, 2003

ABSTRACT

For my Masters thesis I implement a web enabled GIS application for presenting personalized real-time traffic condition information. Due to the dynamic nature of traffic condition reporting, often large amounts of data have to be reported. The process of introducing personalization to traffic condition reporting hopes to reduce the amount of such data transmitted to users.

The personalization of the presented traffic condition information is achieved by storing geographic definitions of routes and travel zones frequently traveled by the client. Since traffic update areas frequently requested for daily travel routes are often geographically identical, stored routes or zones can be used within a Geographic Information System (GIS) environment to retrieve traffic volume information and visualize the intended route before the start of the client's routine daily trip. This saves both browsing time and data uploading for the client. The research implements such tools in a server-side (most of the processing done on the server) environment. The research concludes that existing GIS tools can be enhanced to implement the concept of using customized traffic profiles to transmit user specific traffic data.

TABLE OF CONTENTS

Chapter		Page
1.	**Introduction and Problem Statement** ……	1
2.	**Literature Review** ……	9
	Client /Server Model and Web Software Architectures ……	15
	Client-Side and Server-Side Web GIS Applications …….	23
3.	**Project Data Description and Research Plan** …….	28
	Project Data …….	29
	Formatting and Storing the Project Data ……	32
	Establishing a Traffic Network in ArcInfo 8.1 ……	33
	Proposed Web Service Functionality ……	35
4.	**Creating and Using Customized User Profiles** ……	37
	Custom Areas of Interests and Potential Travel Paths …….	38
	Implementation Basics – Setting Up the Necessary Internet GIS Components…….	42
	Using ArcIMS to Provide Updated Traffic Information to the Client with Areas of Interests……	49
	Using ArcIMS to Provide Updated Traffic Information to the Client with Potential Paths ……	57
5.	**Results and Further Research** ……	64
	Creating and Using Rectangular Travel Profiles (AOIs) ……..	64
	Creating and Using Potential Preferred Paths ……..	67
	Suggested Trip Paths Using Shortest Path Potential Profiles ……..	69
	Implementation Challenges …….	71
	Challenges Down the Road …….	73

References……..... 75

Appendix ……. 78

Vita ……. 176

LIST OF TABLES

Table		Page
2.1	Transit planning website taxonomy (Source: Peng and Huang 2000)	11
2.2	Advantages and disadvantages of client-side and server-side GIS architectures (Modified from Gifford, 1999) ……...	25
4.1	Major websites that use user profiles (both client- and server-side)……...	40
4.2	Major scripting languages used for research ……..	48

LIST OF FIGURES

Figure		Page
1.1	WSDOT traffic website …….	4
1.2	Research overview …….	7
2.1	File server architecture ……...	16
2.2	Two-tiered architecture ……..	17
2.3	Three-tiered architecture with TP monitor …….	19
2.4	Three-tiered with application server (Modified from Schussel, 1995) …..	20
2.5	Outlined research methodologies using a server-side implementation……	27
3.1	Different ways traffic volume data can be stored ……...	28
3.2	Snapshots of the PDOT dataset with the Adams County street data ….....	31
3.3	Basic model for storing user profile information ……...	34
3.4	Overall web application design …….....	36
4.1	AA Road watch traffic path updates by mobile/text messaging ……	39
4.2	Selection of a potential path based on map selection of start and end trip locations …….....	40
4.3	Selection of a preferred potential path based on selection of a set of links from the start to the end location ……...	42
4.4	Defining an AOI using a rectangle limiting update areas ……..	43
4.5	GeoMedia WebMap architecture featuring the client-side ASP page and server-side processes ……..	45
4.6	Overall ArcIMS architecture showing the flow of data from the client's browser and the data server/sources though the application server ……...	46
4.7	The Adams County network for Southeast Pennsylvania ……..	49

4.8	ArcIMS locks shapefile storing AOI shapes …….	50
4.9	Process for storing a new AOI for future use ……	51
4.10	Flow diagram for the selection and extraction of traffic volume data within AOI rectangles ……	53
4.11	Transmitting traffic volume data from the server back to the client using XML ……..	55
4.12	XML file storing updated traffic links ……...	56
4.13	Methodology for creating preferred paths …….	58
4.14	Methodology for using preferred paths for traffic updates ……	60
4.15	Creating and storing shortest path profiles ……	62
4.16	Methodology for using shortest paths ……	63
5.1	AOI profile rectangles drawn and stored for future use ……	65
5.2	Traffic volumes rendered within the AOI identified as ‘1’ in figure 5.1 ……..	66
5.3	Traffic update information using potential preferred paths ……...	68
5.4	Shortest path route guidance ……..	70

CHAPTER 1

INTRODUCTION AND PROBLEM STATEMENT

Integrating information management with geographic visualization has enhanced the power of Geographic Information Systems (GIS). This is evident in the world of transportation. With the advent of increased traffic monitoring programs and user information services, updated traffic information is conveyed to the public using location based referencing and visualization tools provided by GIS. Even before various national and statewide initiatives on real time traffic condition reporting were introduced, spatial and network analysis through GIS contributed to various transportation studies and planning initiatives. For example, various travel demand analysis projects utilized the information management and visualization tools provided by GIS to aid in analysis, organization and presentation of results.

By the 1980's, traffic congestion problems started plaguing various metropolitan areas like Washington DC and New York. Not only was it important to plan better and effective roadways, but existing traffic had to be better managed. Central to this idea were endeavors involving the dissemination of updated traffic information to the public. Often, traffic congestion patterns would quickly increase and decrease by choices made by drivers before and en route to travel. This prompted various nationwide and statewide initiatives such as Intelligent Transportation Systems (ITS) and Advanced Traveler Information Systems (ATIS). ITS attempted to increase the efficiency of the nation's highway systems through the use of advanced computing, real-time data sensor and

communication technologies (Miller and Shaw, 2001). Programs like ATIS (operating within the ITS framework occasionally) helped in *collection, consolidation* and *communication* of traffic information (Gilroy, Puentes and Schuman, 1998). Communication of collected traffic condition data to the public is an integral part of the ATIS. More recently, GIS has played a significant part in this ATIS infrastructure. Presenting, mapping and organizing rapidly changing traffic information for the general public are a few such functionalities where GIS can continue to contribute to ATIS.

In tandem with ITS and ATIS progress, traffic data has also become more readily available. This has been made possible with the advent of traffic monitoring stations, video data, induction loops and fast data acquisition/reporting procedures. Real time information about roadway accidents, potential delay locations and congestion spots can be reported to the public (the public will be referred to as the client or user from now on) almost as soon as they become available. With this increase in traffic data availability, the client has been presented with different methods of real time traffic data reporting. Traditionally, clients had to rely on radio or television updates. Trouble areas or incidents were verbally or visually conveyed to the client either before or during travel. The main problem with this method of information dissemination was the lack of mapping and detail. Trouble spots could only be described without mapping these locations to exact geographic entities such as interstates and street intersections. In addition to this, detailed information like traffic volumes on streets (not just major interstates and highways) could not be conveyed. This makes the information transmitted by radio-television traffic broadcasts somewhat incomplete. For example, a verbal report indicating an overturned trailer on the outer loop of the beltway at exit 31 told people to

avoid the beltway around that area. Information on backlogs along connected streets and the region was generally ignored. While event notification is helpful, more detailed information about traffic volumes on streets would be welcome information critical to avoiding lengthy waits. With the increase in popularity of GIS and location based services, traffic conditions or any other type of attribute information can be associated to geographic coordinates and transmitted to the client (What are Location Services?, 2000). Also, real time descriptive traffic information along streets and highways could be tied to relevant locations and stored for further analysis or presentation.

With the growth of the Internet during the 1990's, traffic condition reporting reached the PC desktop world. Public services such as the 'Trafficstation' web service (www.trafficstation.com) or 'Smartraveler' (www.smartraveler.com), to name a few, provided updated maps every minute or even every thirty seconds indicating traffic congestion areas and traffic volumes (usually classified into intervals). Public sector Department of Transportation (DOT) initiatives like the Washington State Department of Transportation (WSDOT) site provided up to the minute traffic volumes along major highways (Figure 1.1). It was up to the user to pan or zoom into locations of interest. A majority of web services also provided rated (specifying color coded traffic volumes from heavy to light) traffic volumes along interstates and major highways. These color codes were meant to designate areas that should be avoided by the user during an intended travel trip. Reasons to avoid travel in these zones could be heavy traffic, construction or accidents.

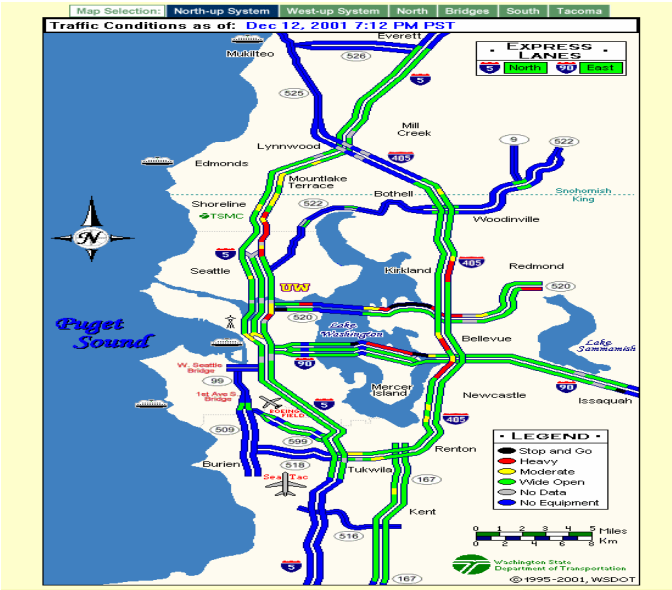


Figure 1.1 WSDOT traffic website

Usually entire metropolitan areas were mapped and the user had to navigate to intended areas of travel through a series of time consuming clicks and substantial reloading time. Seldom were traffic conditions on smaller highways and streets shown. To add to this, daily trip planning tools were not available. For example, the client might have wanted to know about the pros and cons of choosing an alternate route to work. The client might also have wanted the shortest path from home to the nearest convenient store mapped out before leaving home. This guidance would be even more useful if this shortest path included streets where traffic volumes were the lowest. Many of the limitations mentioned above, such as long browsing times and lack of pre trip planning tools (such as route guidance) could be overcome by the use of customized or personalized traffic services. In addition, availability of data permitting, real time traffic conditions along

streets and not just major highways can be visualized with the use of such personalized traffic services.

Customized real time traffic information services work on the premise that every client has individual traffic information needs and requests. These services provide real time traffic condition information, which is pertinent only to the user's intended or frequently observed travel paths. For example, a user from the suburbs frequently visits downtown areas on the weekend. When he or she pre-defines the downtown city destination areas and asks the web service to store this region for future use, a customized travel profile is created. Before future downtown trips, the client can then initiate an information retrieval from the web service, which gathers current traffic conditions (usually traffic volumes) within the pre-defined region. This eliminates web browsing that was needed to zoom into locations. Also, since the geographic scope of the request for traffic information is reduced to a single region within the larger network, more detailed traffic conditions and traffic volumes can be displayed and mapped.

Customized web services (not limited to real time traffic sites), have been a recent addition to the ensemble of web services being provided by various Internet services. Companies from Yahoo to QVC allow clients to customize the content of their web pages. Clients can define the appearance of their web pages or complete online transactions using shopping carts to store items for purchase. 'My Yahoo' (www.yahoo.com), for example, allows clients to define yellow page locations and other information that affect the appearance of their 'Yahoo' home pages. Each client receives a unique page tailored to his or her specific needs. The Weather Channel (TWC available

at www.weather.com) and Monster (www.monster.com), for example, have followed suit.

With customization, the concept of a 'shopping cart' for online transactions has also become popular. Using shopping carts, the user can now interactively select several items in one or more online sessions. Information about these items (like inventory indices, prices, etc) is stored so that the final transaction can include these items. However, these services have remained scarce in the world of real time traffic information. While some Internet services, like Chicago's transit service (www.transitchicago.com) and ARITIS (http://www.airtis.com/airtislite.asp) do provide the client with some online transaction capability and route storage options, the scope of customization is limited. These sites merely provide basic trip planning or text based traffic condition reporting. This research introduces the concept of increased customized services to real time traffic reporting.

More specifically, the research develops a running application with a set of Internet tools that allow the user to log in and create his or her travel profiles. These could be general areas such as rectangles that represent frequently traveled network zone or simply a definition of a frequently traveled path. Profiles can be drawn on the client's browser screen and stored for future use. Once stored, they can be used to quickly access traffic volumes within streets and highways that constitute the client's profile. Along with implementing this process using the framework provided by some commercial GIS packages (like Environmental Systems Research Institute, Inc), software development challenges and future directions of research for similar applications are discussed. The major objectives of the research are summarized in Figure 1.2.



Research application providing the use of user specific travel profiles

Recommended travel path

Focused access to traffic network allows more detailed and reporting of real time traffic conditions



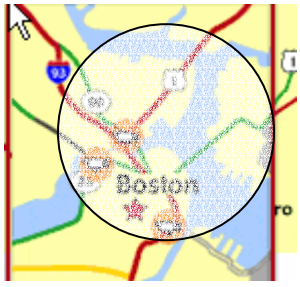


Figure 1.2 Research overview

As illustrated in the figure, this research hopes to develop Internet tools with the following functions:

1) Assuming that real time traffic data is available in a GIS, expand the scope of current real time traffic notification services by developing techniques that allow the user to indicate frequently traveled paths on a traffic network. These frequently traveled paths (referred to from now on as travel profiles) include a portion of the traffic network or specific paths within the network that represent routine trips. Profiles can also be a pair of start and end locations indicating an intended travel origin-destination pair.

2) Store these profiles so that they can be used to provide real time traffic volume information along the traffic network portions that fall within these intended travel locations or paths. This way, only necessary portions of the traffic network have to be visualized to each client.

3) Provide some route guidance for the client by indicating a shortest path for profiles that only contain start and stop locations (instead of a specific travel path or region). This shortest path minimizes the travel cost along network links using current traffic volumes as the cost parameter.

CHAPTER 2

LITERATURE REVIEW

The following research builds an application that presents real time traffic information to the public via the Internet. Peng and Huang (2000) conducted research in the area of customized services for transportation applications. They discuss the evolution of traffic information services with transit agencies. Before the 1990's, transit users were presented with schedules and had to decide their own paths and transfer points. Often, customer service agents would suggest itinerary plans while keeping the transit user updated. Peng and Huang argued that this form of transit planning was tedious, time consuming, redundant and often erroneous.

By the mid and late 1990s, transit users had access to computer aided trip-planning systems. These systems (for transit passengers) were facilitated even more with the popularization of the Internet. Since then, Peng and Huang indicate that most traditional customer-based services like schedules, routing and trip planning have been enhanced or even replaced by web-based information. These services can dynamically tie information together and create more comprehensible routing information compared to what traditional brochure-based planning had provided. Along with these advances, the visualization provided by web-based presentations captured the geography of the proposed route. This contributed to the effectiveness of the traveler information being provided.

There have been many public web sites dedicated to pre-trip transit and automobile traveler information and planning. Peng and Huang (2000) have developed a taxonomy based on the quality of service provided by the web based transit information application shown in Table 2.1. They use the columns in the table to define the content of information that has to be provided by the service. This refers to the utility level as far as general web services are concerned. For example, a simple text search (level 1) is a common and basic utility, while online transactions (level 4) constitute a level of utility that is more sophisticated. Each row, on the other hand, defines the level of analysis function each site provides. They show such things as how interactive a site is or whether or not it provides customizations that allow transactions. For example, real-time information for bus locations and delays provide rather advanced data content that the user can access. For this reason, this type of functionality is classified as a D status. On the other hand, general information like bus timings and static bus routes provide a simpler level of information. These have been assigned an A status.

Peng and Huang (2000) develop and discuss the basic architecture involved in their web based transit and traffic information systems. They use the following four categories:

(1) User interface design: The typical user interface required that the user input origin, destination and travel time data. They suggest that the designer could choose to provide the user with an interface that provided text only interfaces or GIS based systems that provided the user with interactive route guidance services, spatial query and search functions.

Table 2.1 Transit planning website taxonomy (Source: Peng and Huang 2000)

	Content level	Web browsing (HTML and pdf)	Text search, static graphic links	Interactive map-based search, query and analysis (Internet GIS)	Customization and information delivery	Online transaction
Function Level		0	1	2	3	4
General Information	A	A0	A1	A3	A4	A5
Static information (routes, schedule and fare)	B	B0 ▪ www.city.toronto.on.ca/ttc/schedules/index.htm ▪ www.itsmarta.com ▪ www.suntran.com	B1 ▪ www.wmata.com without the itinerary planning system	B2 ▪ www.transitinfo.com	B3	B4 ▪ www.transitchicago.com
Trip itinerary planning	C	C0	C1 ▪ www.wmata.com ▪ www.romanse.com ▪ www.mta.net ▪ www.theride.org	C2	C3	C4
Real time information (bus locations and delays)	D	D0	D1	D2	D3	D4

(2) Map server functions: The map server provided map rendering and address matching capabilities.

(3) Data and DBMS: They used a relational database in Microsoft Access and the Open Database Connectivity (ODBC).

(4) Network analysis component: They indicated that the network analysis component was a key component to providing itinerary trip planning. Peng and Huang (2000) used a path finding algorithm for transit usage that is different from the usual algorithms for highway usage.

Using this architecture, Peng and Huang (2000) also suggest developing personalized traveler information systems. The concept of personalized traffic information services involved having the system store user profiles of frequently traveled routes. However, even now, very few if any websites on the Internet provide the capability to store frequently traveled origins and destinations. These stored locations can be useful so that when the user logs in, the trip origin and destination information is recalled immediately. This profile storage capability can then be used to make a traffic update procedure that covers a smaller geographic area, thereby decreasing the information retrieval time. According to Peng and Huang's (2000) typology, this would be a D3 (both real time and customized travel updates) website. To investigate current technology in the area of personalized services, I conducted an inspection of current real-time traffic update sites available on the Internet. This inspection is summarized below through a list of some of the websites reviewed:

- http://www.smartraveler.com/scripts/phlmap.asp?city=phl&cityname=Philadelphia

Smartraveler provides users with route codes that identify major interstates and highways. Using these route codes, clients can dial in and get quicker updates on those coded highway segments. Since the user can use route codes that limit the area of updates, Smartraveler does provide minimal customization. However, updated traffic information along streets is not presented visually for the client. This fact, coupled with the lack of interactive capability for the client, makes the Smartraveler site a minimal D3 service.

- http://www.airtis.com/airtislite.asp

AIRTIS provides a partial D3 service that allows the user to store a customer profile defining a frequently traveled route. This updated information is emailed to the registered customer at regular intervals. This service, however, does not allow for visual updates in real time and route selection is not interactive.

- http://www.wsdot.wa.gov/PugetSoundTraffic/

The Washington State Department of Transportation (WSDOT) site gives updated traffic conditions for the Puget Sound area in Washington State. This web based traffic information service provides color-coded maps indicating traffic volumes on major interstates such as the I-495 and I-5. Categories ranging from "stop and go" (heavy congestion), to "wide open" (low congestion), are rendered on maps which are updated frequently. This data is collected from WSDOT Traffic Management Systems (TMS) and provides the updated information using data collected from loops (remote sensors placed roadside). This website can be categorized as a D2 (real time information with simple graphical browsing).

- http://frida.transport.civil.ntua.gr/map/route.html

Travel time information is the focus of this website providing traffic condition updates for the Athens, Greece. This University of Athens web application helps users decide on where they can go in 15 minutes from any major street intersection. While this site does not provide tools for customized travel information and detailed traffic updates, it does allow the user to plan a possible course of action during a trip. Once again, this site does not strictly meet the D3 functionalities as defined by Peng and Huang (2000). At best, it can be classified as a mixture of C2 and D2 (providing trip planning and real time information in a non-interactive setting).

All of these websites provide useful information in different ways. Some give the user detailed current traffic conditions that include accident and congestion information but limit the data to only major highways. Other websites provide customized routes with only a text delivery mechanism (for example, AIRTIS). Thus, the need of providing content unique to each client's needs, combined with detailed road traffic conditions, opens the door for customized real time traffic applications. However, customization relies heavily on the ability to store frequently traveled paths or regions and effectively use this information to provide traffic updates in real time. This means that large quantities of spatial and related attribute information have to be transmitted from the client's browser to the website hosting the data. This is where the overall design and architecture of the resulting web application becomes critical.

The client/server architecture of web applications is very popular. For Internet applications, any computer accessing the service through the Internet is referred to as the client, while the host providing the services (i.e. updated traffic maps and volumes) is the server. This generally constitutes a traditional client/server model (Webopedia, 2001).

The client/server model has undergone many transitions and upgrades in recent years. The following discusses the evolution of this model.

Client /Server Model and Web Software Architectures

The client/server model was initially developed in the late 1980's (Schussel, 1995). It subsequently went through many transformations. The initial push to develop the client/server model resulted from the gain in popularity of PC network computing. Data had to be distributed from file servers to client computers. The first emergent architecture was the file server architecture that was popular in the 1980s. With the file server architecture, the client computers would simply download necessary files from the server when needed. All applications would reside on each client machine and files would be moved from network servers through a local area network connection. File server systems were slow because large files had to be transferred from the server to multiple clients on a regular basis. This could cause network congestion problems (Schussel, 1995) due to overloading client resources. In time, software companies like Xbase, DBase, and FoxPro came up with different file sharing systems. Figure 2.1 illustrates such a file sharing system. Necessary files were transferred upon request from the file server to each client using a stackable hub for transmitting data. All the software resided on the client (for example DBase+ etc), where all the processing was done. This was the initial version of the client/server model.

As demands for faster data and file transfers increased, client/server architectures evolved, and two-tiered client/server architectures emerged. Two-tiered client/server architectures were different in that the network file server in file sharing systems was

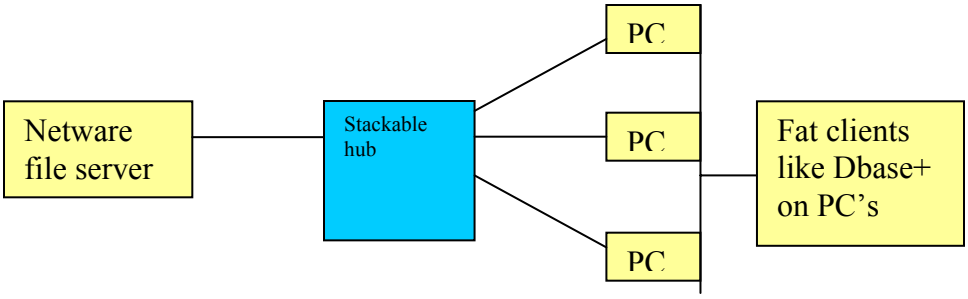


Figure 2.1 File server architecture (Modified from Schussel, 1995)

upgraded into a true database server (Schussel, 1995). This meant that the clients only had to run interfaces that would launch SQL queries to the Database Management Systems (DBMS) tier. Figure 2.2 depicts this process. Some of the processing was actually done on the server, which housed the DBMS system. Once again, the stackable hub shown in the figure connected the various clients to the DBMS system. Clients only had to run queries that were used to fetch data from the DBMS using Structured Query Language (SQL). Consequently, network traffic would also be significantly reduced since data did not have to be downloaded on to the client in its entirety. Only queries and their results were transmitted through the network. Simply put, two tiered architectures allowed the client to download and store less data by simply fetching the required data from the server. In time however, as the number of clients increased, the DBMS server would retain threads for each client connected to the server even when no work was being done (Schussel, 1995). This would cause overloading of the server when the number of clients reached a critical capacity.

Schussel (1995) also mentions another drawback regarding two-tiered application architectures. Moving two-tiered applications could be very cumbersome. Migrating

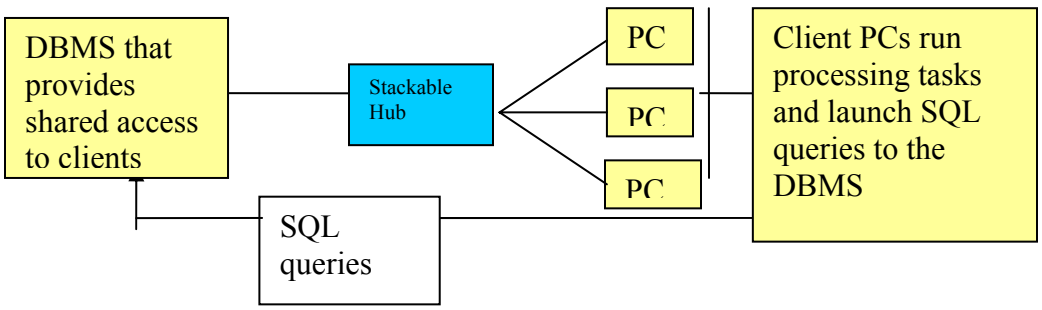


Figure 2.2 Two-tiered architecture (Modified from Schussel, 1995)

such applications from one server to another would require a regeneration of procedural code that would make such a design disastrous when portability was critical. Dickman (1995) also emphasizes this drawback about two-tiered approaches when he describes them as "fat-client" (high software overheads and required bandwidth on the client) systems and therefore a burden to the clients. However, Dickman (1995) also points out that remote management programs like Lotus Notes can manage the database tier to improve its behavior and thus overall application speed from server to client. Dickman (1995) indicates that this still "is often an afterthought" (Pg. 2).

To alleviate some of the problems about two-tiered architectures, a third middle tier was added. This third tier would lie between the client and the DBMS server. Three-tiered architectures moved all the business processing (executables and programs that run client applications) from the client to the middle tier. The client sends its request to the middle tier and then disengages. The request is scheduled and prioritized by the middle tier, which then sends the request to the database server. Variations arose as to how the

middle tier was set up. How much of the application was kept in the middle tier and splitting the middle tier into further sub tiers were issues that had to be considered. According to Schussel (1995), there are four major types of three-tiered architectures:

(1) Three-tiered with Transaction Processing monitor:

The transaction-processing (TP) monitor was one of the oldest methods of transaction processing that dated back to the mainframe era in the 1970's (Schussel, 1995). The client connected to the TP monitor instead of directly connecting to the DBMS server as in two-tiered systems. The TP monitor then disengaged the client and assumed the responsibility of completing the request from then on. This architecture used scheduling algorithms like round robin to allocate computing resources in a priority driven environment. Figure 2.3 shows how the DBMS on the server handled queries to completion thus disengaging each client after the query was completed. However, this practice in a three-tiered architecture did not last for long because the clients were still running substantial applications and heavily taxing the network (Schussel, 1995). According to Schussel (1995), TP monitor driven three-tiered architectures can still handle more clients than two-tiered systems (approximately by about 100 to 200 users).

(2) Three-tiered with messaging server:

The three-tiered architecture with a messaging server was similar to the TP monitor architecture with one major difference. The messaging server handled only intelligent messages. Each packet of data contained information about both logical and physical network addresses. The messaging server was reduced to only performing communication related functions like encryption and transportation of data packets (Schussel, 1995). This change added more flexibility to the system, something that was

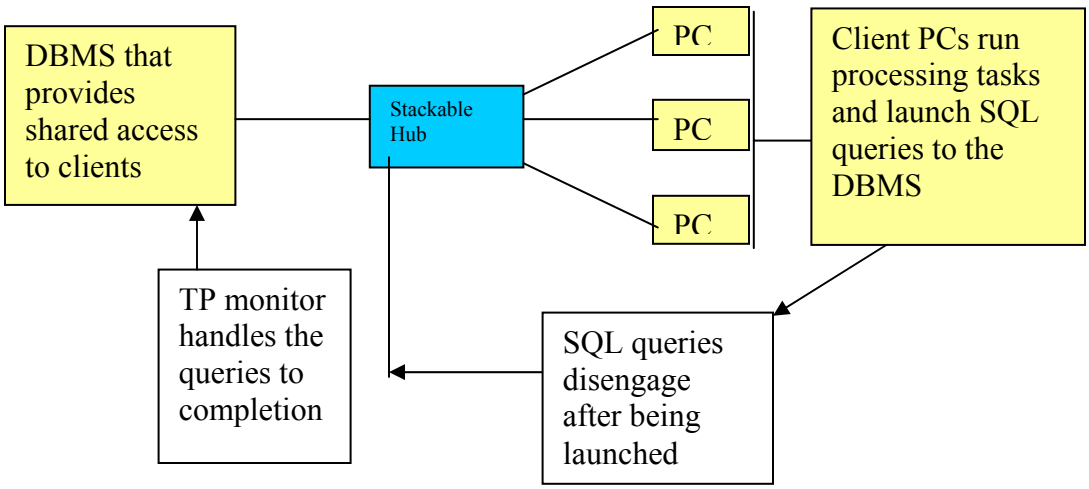


Figure 2.3 Three-tiered architecture with TP monitor (Modified from Schussel, 1995)

missing in the earlier TP monitoring technology. Even with this flexibility, the need for a thinner client eventually resulted in messaging server architectures giving way to application server systems.

(3) Three-tiered with application server:

The three-tiered architecture with application server operated on the premise of a truly thin client. This meant that the PC or client computer was used merely for presentation services, almost like the terminal of a mainframe computer. In concordance with the X architecture developed in the 1980's by the Massachusetts Institute of Technology (MIT), the entire business logic tier was placed on the application server. The application server was responsible for executing the application, tasking and monitoring the client requests along with interacting with the DBMS server. The three-tiered application architecture described in Figure 2.4 shows the intermediate application server tier that contains all the executables that process and transmit SQL queries to the DBMS tier. In other words, a bulk of the processing now resided in the intermediate

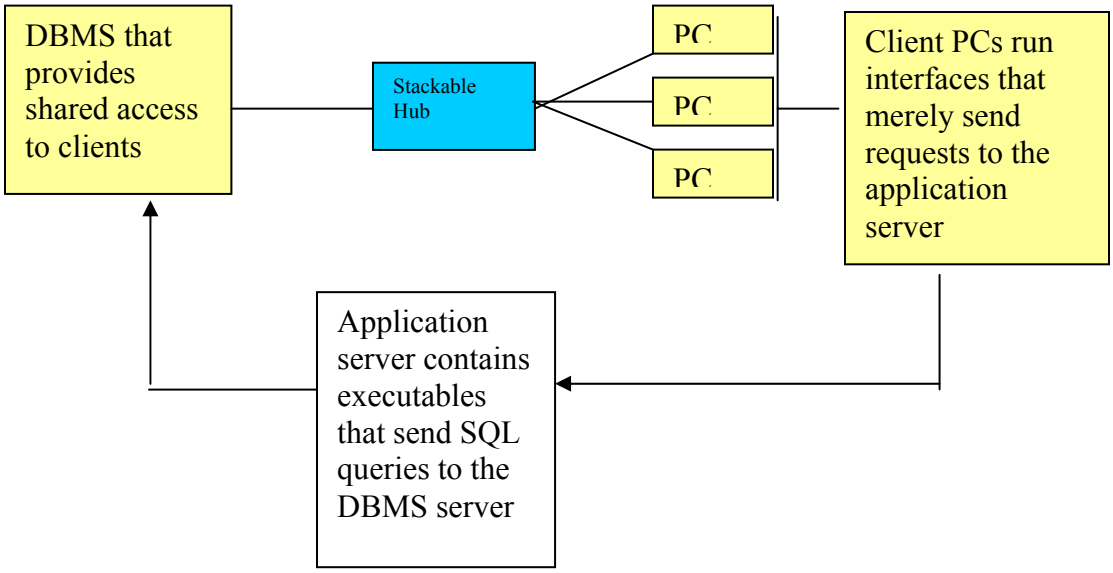


Figure 2.4 Three-tiered with application server (Modified from Schussel, 1995)

application server tier. There were many advantages to this form of three-tiering. These include (Schussel, 1995):

- Less software on the client and alleviated worries about security
- The application was more scalable because the server could be a single processor, a multiprocessing sequent environment, or even a massively parallel system
- One server was easier to maintain as opposed to hundreds of PC's
- Added flexibility because application portioning was easier. Code could be reallocated to new servers after the system had been built

However, with application servers, queries had to be processed for each request. The query and database objects could not last the lifetime of the application. This need for

persistence spawned the design of Object Database Management Systems for the middle tier.

(4) Three-tiered with Object Database Management System (ODBMS):

In this architecture, the middle layer was replaced with an ODBMS. The ODBMS retrieved the data from the DBMS and created persistent objects that lasted the lifetime of the application. The ODBMS architecture led to the formulation of distributed components within a three-tiered architecture. With the advent of Microsoft Object Linking and Embedding (OLE) or Object Management Group's (OMG's) Common Object Request Broker Architecture (CORBA), primary application objects and database objects could be copied and distributed on to various servers. This added distribution led to an increase in fault tolerant computing since the failure of certain objects could be compensated by using its copy in a different server (Schussel, 1995).

With the design and implementation of two-tiered and three-tiered software architectures, the client/server model had evolved into a popular choice for the deployment of various software solutions. Currently, however, Internet applications are special cases (two-tiered or three-tiered) of the aforementioned client/server model. While the origins of the Internet can be traced back to the 1960's, using the Internet for distributing and interacting with Geographical Information Systems (GIS) did not become widespread until the mid 1990's. This coincided with the privatization of the Internet and vast user growth. Ever since, it has appeared that the advancement of Web GIS products has occurred in tandem with advances in Internet technology.

One of the first Web GIS endeavors was developed by Steve Putz at the Xerox's Palo Alto Research Center and put online in 1993 (Plewe, 1997). The site

(http://map.web.parc.xerox.com/map) generated maps from public domain data using non-commercial GIS programs on the server-side. This idea quickly caught on with other web sites like Virtual Tourist (http://www.vtourist.com/web), NSDI (National Spatial Data Infrastructure at http://www.fgdc.gov) and the online digital library of spatially referenced information funded by the Alexandria project (http://alexandria.sdc.ucsb.edu). All of these sites provided the user with basic interactive abilities like panning and zooming, along with data downloads and pre-designed raster maps.

A change was initiated in 1995 with the development of live mapping engines such as the Topologically Integrated Geographic Encoding and Referencing (TIGER) mapping service at http://tiger.census.gov. "Live" referred to the introduction of interactive ability supplied to the user. The user could not only pan and zoom as with the earlier Xerox applications, but he or she could turn layers on and off. They could alter symbology by changing map rendering and download maps as images. In other words, the user could now change the web content interactively.

According to Plewe (2000, p.13), 1996 is the year "everyone joins the party". Coinciding with the coined "year of the Internet", major GIS vendors such as Intergraph, ESRI (Environmental Systems Research Institute) and Bently started publishing their software and data online. Internet architectures matured with increased bandwidth connections while heftier clients and servers developed from strides in the microprocessor and related industries. It was getting easier for the public to access and interact with spatial data. By the late 1990's, ESRI and Intergraph along with a few others came up with commercial packages that allowed GIS applications running on the server-side to be accessible interactively by client computers. This significantly

minimized the amount of development for third party developers. Tasks such as load balancing, maintenance of client-side states and web publishing were available through commercial packages such as the ArcIMS by ESRI and the GeoMedia WebMap Internet map control by Intergraph.

With this increased ease of web application deployment, development choices started to become important. Applications could provide increased processing capability to the client by loading the server with data processing capability. On the other hand, GIS applications could also allow the client's computer to do a chunk of the required processing and analysis. This would free up the server for increased load handling. Either one of these development paradigms could be chosen.

Client-Side and Server-Side Web GIS Applications

Gifford (1999) states that even though the Internet adheres to the client/server model, it is based on a network that is usually slow and constrained by a large network size and the need for widespread administration. Fortunately, he says, communication protocols are written so that in case a path is "down" or congested, many other different paths may be taken. Notwithstanding this flexibility, the quality and integrity of data transmission and connection bandwidths remain the lynchpins in the success of any client/server application. The amount of data and other information that has to be transmitted between the client and server is critical.

Addressing these issues, Gifford (1999) groups GIS applications into server-side and client-side applications. Server-side applications require the web browser to initiate server requests while client-side applications usually have the user (or client) enhanced to

perform and support GIS operations. Server-side applications load the server machine with a majority of the application processing. For example, if the web service in question returns all hospitals within a ten-mile radius for a given address, the client would simply send the address to the server. The server then performs a query or computes a buffer to extract the necessary information. After processing is complete, the hospital's information is sent back to the client for display. Most of the processing, such as data query and analysis are done on the server, thus making this a server-side web application. On the other hand, if the client were made to download necessary hospital data and perform a buffer on the client machine, the application would take a decided turn towards a client-side service. Table 2.2 lists some advantages and disadvantages of client-side and server-side systems as outlined by Gifford (1999).

Both client-side and server-side applications have their advantages and disadvantages for serving GIS functionality over the Internet. As expected, the requirements of the application dictate the development mode.

For this research, the client/server model is important because the effective usage of customized user profiles hinges on the ability to process concurrent web requests and download updated traffic information to the client. This downloading can be kept at a minimum by developing a client-side application that would entail downloading and storing traffic data on the client's computer (client-side approach). Another option is to store profile information on the server and develop efficient processing methods that would take advantage of the server's computing recourses (server-side approach).

The literature review above established there is a need to introduce personalized services into real-time traffic reporting through the Internet. It also revealed that there

Table 2.2 Advantages and disadvantages of client-side and server-side GIS architectures (Modified from Gifford, 1999).

	Client-side	**Server-side**
Advantages	▪ Excellent on operations that occur locally ▪ Less Internet traffic required ▪ Vector data can be more readily used ▪ Modern interfaces available	▪ Adheres to all web and Internet standards ▪ Can be accessed with a standard web browser ▪ Platform issues are more or less eliminated (with some exceptions) ▪ Low bandwidth requirements ▪ Performance is predictable ▪ Centralizes ownership of data ▪ User support is minimal
Disadvantages	▪ Requires users to obtain additional software ▪ Incompatibilities with client browsers and platforms ▪ Initial download times can be long. ▪ Low performance with large databases	▪ Vector formats are less readily supported ▪ Low graphics quality ▪ Primitive user interface ▪ Creates many requests ▪ Information re-transferred for each request.

are two major development choices (namely client-side or server-side methods). The review also pointed out that the choice of appropriate development technology could influence the amount of information that the client has to process on his or her machine. Quite often, client computer capacities, like available memory or processing speeds can be unpredictable. In addition to this, security reasons can prevent clients from running certain plug-ins, like Java applets or DirectX by Microsoft. As described before, the server-side development method minimizes the software that the client has to run or download onto his or her machine. Hence, this development methodology becomes a requirement to the major research objectives introduced in Chapter One.

Keeping this in mind, the research attempts to create server side tools for customized real time traffic reporting. This requires the implementation of a communication channel for frequent spatial data transfer between the client and the server. This research will develop such a communication channel while keeping a majority of processing tasks on the server. Figure 2.5 provides a summary of the research deliverables based on the above discussion about personalized traffic information procedures and server-side web deployment.

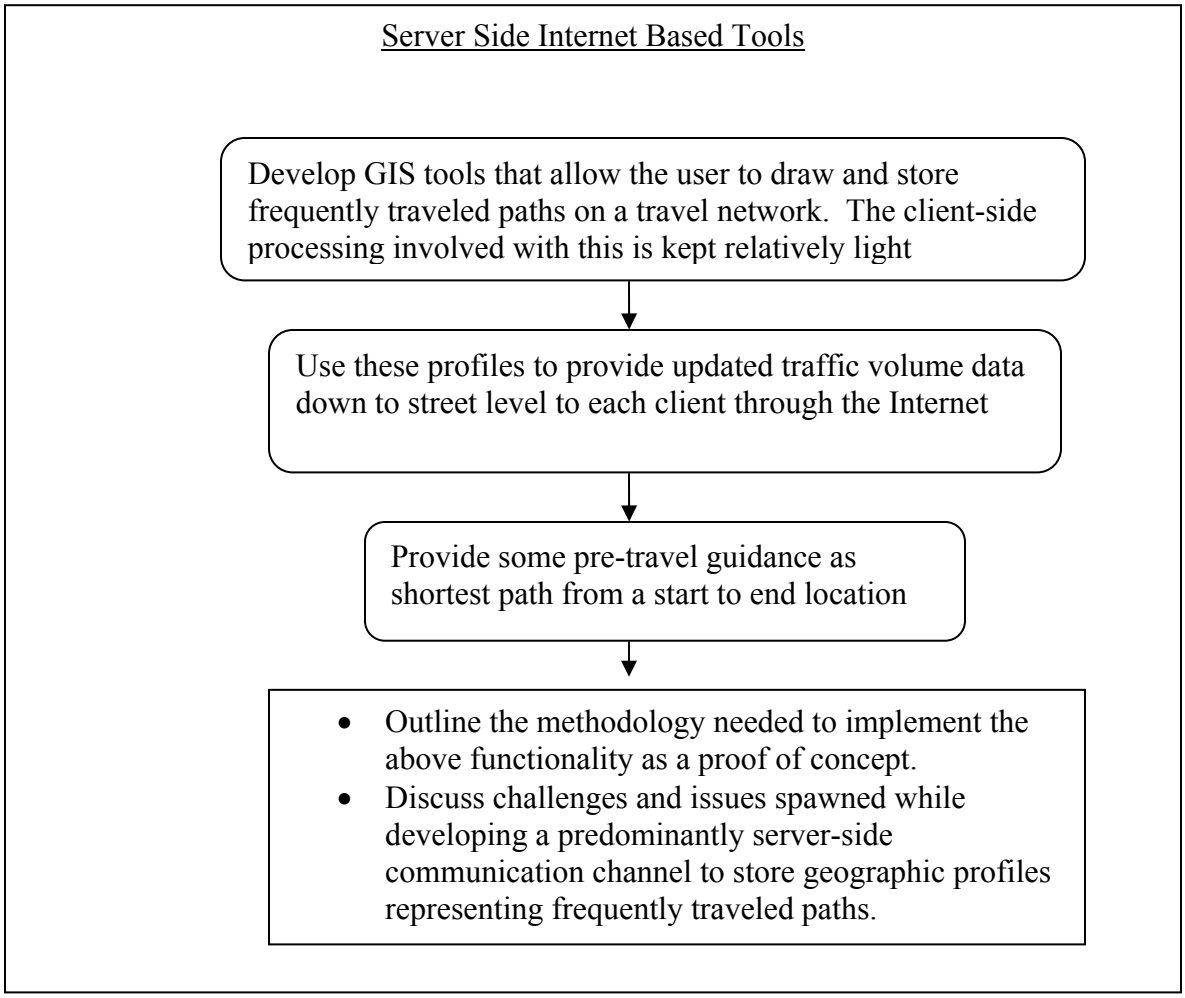
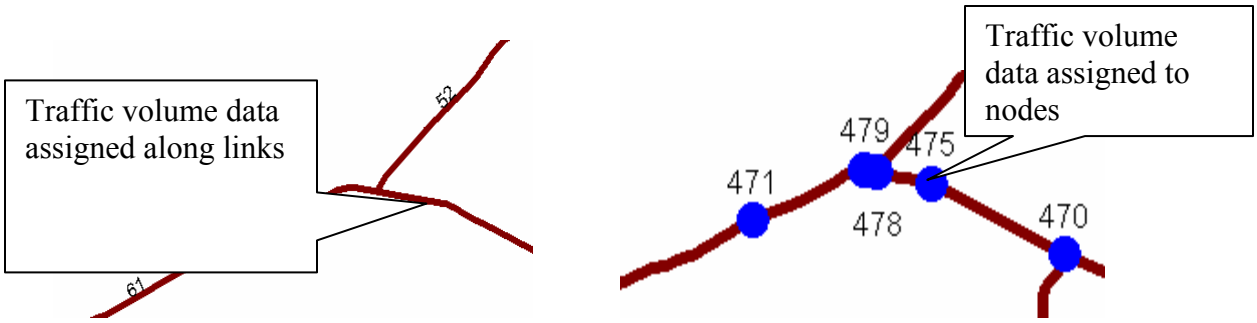


Figure 2.5 Outlined research methodologies using a server-side implementation

CHAPTER 3

PROJECT DATA DESCRIPTION AND RESEARCH PLAN

This research presents traffic information in real-time through the utilization of the Internet. A multitude of traffic data sources, such as traffic accident locations, locations of special events or simply traffic volume data, can be displayed on real-time traffic information sites, often referenced along traffic networks. Data for this type of presentation would commonly be stored as point locations on a geographical database. However, traffic congestion measures such as current traffic volumes can also be represented along highway segments. The illustration in Figure 3.1 shows that the labeled traffic volume numbers can be attributes of either links (Figure 3.1a) or assigned to points (Figure 3.1b).



(a) (b)

Figure 3.1 Different ways traffic volume data can be stored

Project Data

Whether data is recorded along points or lines, it has to be collected from highway networks. The following examples are the types of traffic data that are available from various traffic monitoring programs:

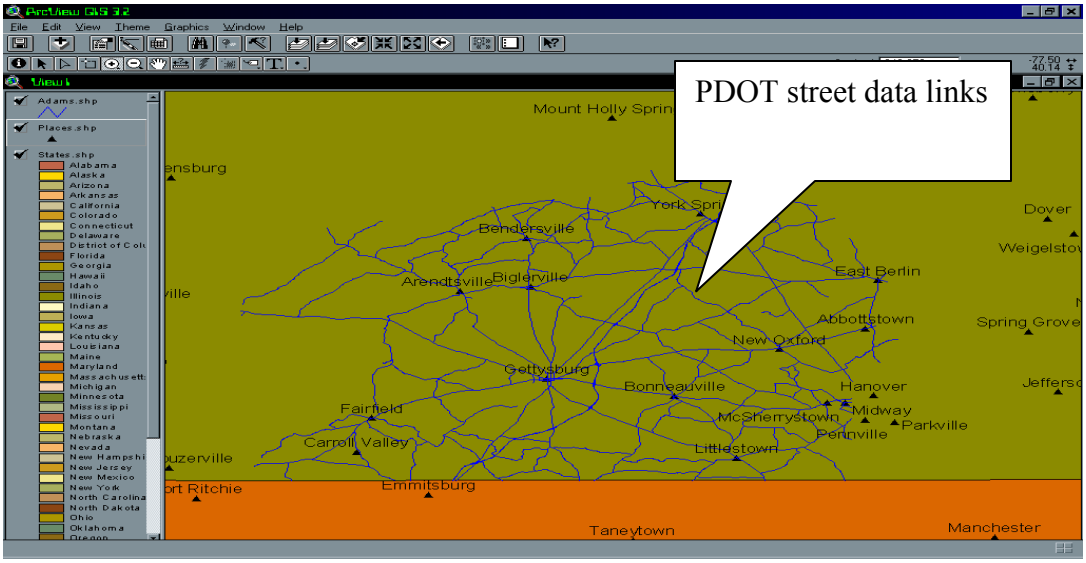
(1) Real time traffic speeds and volumes: Often traffic speeds are recorded using loop detectors. Loop detectors are digital devices controlled by microprocessors that are usually placed across streets. These detectors send traffic volume, speed and density data to a Traffic Management Center (TMC) to which the loops are wired. Loop detectors are often used alongside ramp meters to regulate the flow of traffic on to interstates. Each loop detector has a unique ID that is used to identify it on a GIS coverage representing state highways. Another way of collecting traffic count data is to implement rubber tubes. These tubes are placed alongside roads. The pressure exerted on them by passing cars causes the mechanical counter to record the passage of each vehicle across the road (Johnson City MPO, 2001). More recently, video surveillance has been used for traffic count estimation.

(2) Accident and incident reports: Many state Departments of Transportation (DOTs) offer updated accident and incident/construction reports. Traffic accidents can be reported to emergency centers when a car equipped with an emergency notification system transmits its location. In addition to this, Intelligent Traffic Systems (ITS) programs offer sensor signals and video feed that can be used for acquiring real time accident or construction information. Acquired accident information can be identified on a geographic dataset such as a street network using a unique ID. Then, attribute data such as delay times and construction site status reports can be reported in real time. The

Tennessee Department of Transportation (TNDOT at http://www.tdot.state.tn.us/information-office/const.htm) site reporting updated interstate construction conditions provides a good example.

(3) Average Annual Daily Traffic (AADT): These data are compiled and reported each year to provide estimations on the average number of vehicles using a roadway during a year. Quite often, these various traffic-monitoring programs are used to collect and store the traffic count data. The data are then adjusted for various errors and compiled to provide traffic counts.

Mainly due to its availability, this research used the AADT traffic volume data assigned to links on a street network as traffic data. This dataset was acquired from Pennsylvania Department of Transportation (PDOT). The data was obtained by downloading through the Internet from the following address: ftp://www.pasda.psu.edu/pub/pasda/padot2001/state/padot. The PDOT CD data contained road surface conditions, truck miles, and truck percentages along with AADT data. PDOT traffic volumes were stored in a street shapefile with each link having a unique ID along with other attributes. Figure 3.2 is a snapshot of the Adams County (southeastern Pennsylvania) traffic count data from the PDOT CD available on the web. The metadata for this dataset is included in Appendix B. The field 'cur_adt' holds the average annual daily traffic counts. The field 'Objectid' uniquely identifies the link that represents a road segment. Other fields hold attributes that describe various condition descriptors for road segments.



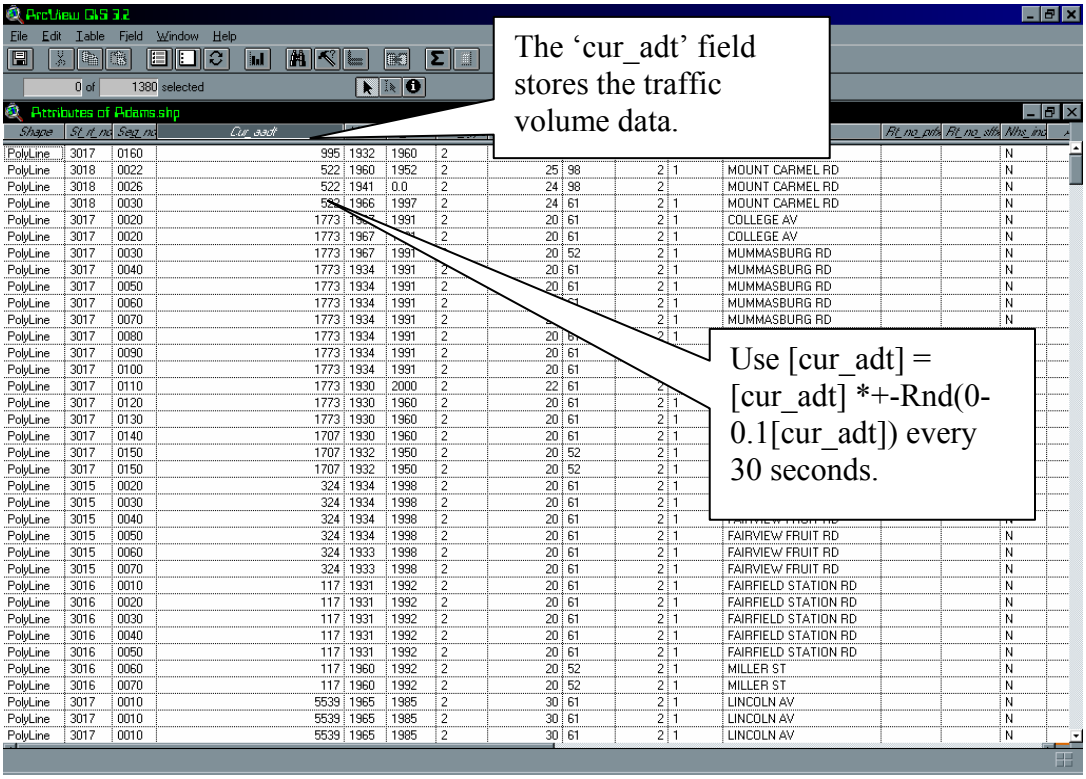


Figure 3.2: Snapshots of the PDOT dataset with the Adams County street data. (Data-source PDOT available at: ftp://www.pasda.psu.edu/pub/pasda/padot2001/state/padot-stroads-adams_2001.zip)

For this research, obtaining real time traffic count data was not feasible. Therefore, for the purpose of this research, it was assumed that the data had already been recorded into a GIS. Since the AADT value in the field 'cur_adt' represented the average annual daily traffic count for a particular link or intersection, this daily count was randomly varied to produce a traffic count value that changes every minute. Since most real time traffic update applications offer traffic updates every minute or thirty seconds, the formatting of the data simulated the actual traffic volume numbers along a street.

Formatting and Storing the Project Data

In an effort to simulate real time data capture, the 'cur_adt' field in the PDOT database was subjected to random fluctuations. The value stored in the 'cur_adt' field was multiplied every minute by a random number that either increased or decreased the existing traffic volume by 0 to 5%. This meant that for the 'cur_adt' value of 1773, the automated program (set to operate as a batch process) on the server would change this value to a number between 1861.3 and 1684.7. The fractions would be rounded off to the nearest integer. These fluctuations were sufficient to test the major issues addressed in the thesis.

For representation purposes, traffic volumes were put into three major classes ranging from light, moderate to heavy. Since interstates and roads can often be two-way roads with multiple lanes, volume to capacity ratios might have been more useful. However since this capacity data was not available, traffic count values serve for an adequate demonstration of the research methodology.

Establishing a Traffic Network in ArcInfo 8.1

Once the PDOT data were obtained and the attribute values formatted to resemble real time traffic counts, the next step was to organize the network data so that it could be used in the proposed application. Reiterating the major goals of the research, traffic volumes had to be displayed to individual clients within their specified geographic region or intended travel path.

To make sure that analysis functions of this nature can be performed on the dataset, the Adams County traffic count database was converted into an ArcInfo 8.1 traffic network. A traffic network allows for topology to be established within the traffic count database. This topology allows for routing solutions (like shortest path between two travel locations) to be computed. Along with the traffic network, a copy of the base Adams County shapefile is also used. This shapefile is used for referencing intended travel regions within the county network, along with visualizing the streets and highways for the client. Even though both the traffic network dataset and shapefile have similar geometry and attributes, the traffic network database allows shortest path computation along with building appropriate topology.

Serving personalized traffic updates to a multitude of clients also requires the establishment of user based data management tables. These tables allow for the attribute data for each user to be associated as a series of network links that represent the user's daily intended travel regions or paths. Figure 3.3 shows a simple data model

Unique-id	Usernum	Profile coordinates
1	1	
2	1	
.....		
10	2	

Attributes define regions within the PDOT network

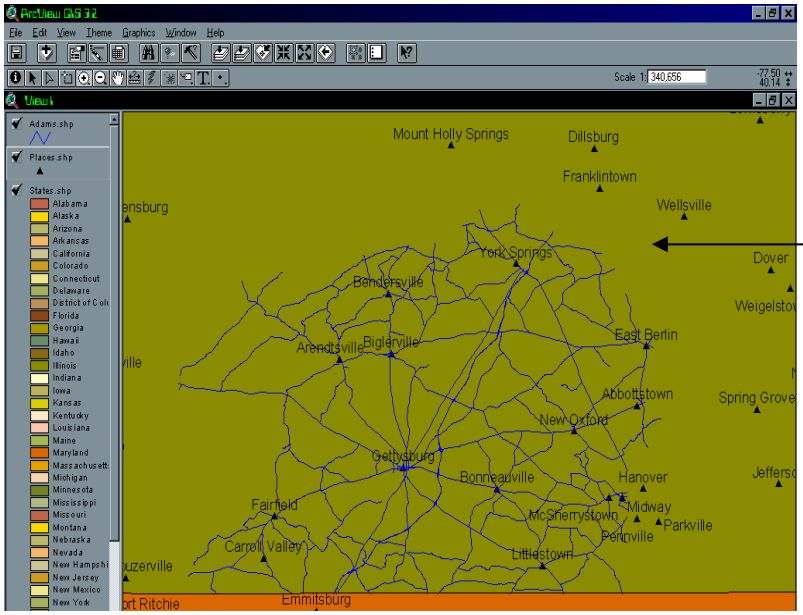


Figure 3.3 Basic model for storing user profile information

for the major user account management tables used in the research. This master table was named ‘tuserpoly’. Each record in this table stored basic information about travel profiles and associated properties, like coordinates of travel regions, and descriptive notes. This table was also indexed by a unique identifier (‘Unique-ID’ field) that could be used to reference each record. Along with this, the user-number field ‘Usernum’ identified the user associated with each profile.

Proposed Web Service Functionality

With the database structure in place above, the basic framework for the storage of individual profile information linked to the traffic network was established. The next step was to design the overall functionality of the web service so that the profiles stored in the master table ‘tuserpoly’ could be used by the client to receive updated traffic information and route guidance.

In order to keep client-side processing to a minimum, the web service was designed to operate in two distinct modes. The first mode (Mode1 in Figure 3.4) allows the client to interactively construct travel profiles on a browser. These profiles are constructed by indicating start and stop locations on the network, drawing rectangles or tracing out a frequently traveled path. These profiles are stored on the server using the master table ‘tuserpoly’ along with some additional reference tables.

The second mode (Mode2 in Figure 3.4) presents updated traffic information to the client. Mode 2 operations use the profile information stored on the server to extract either recommended travel paths between two locations or updated traffic volume

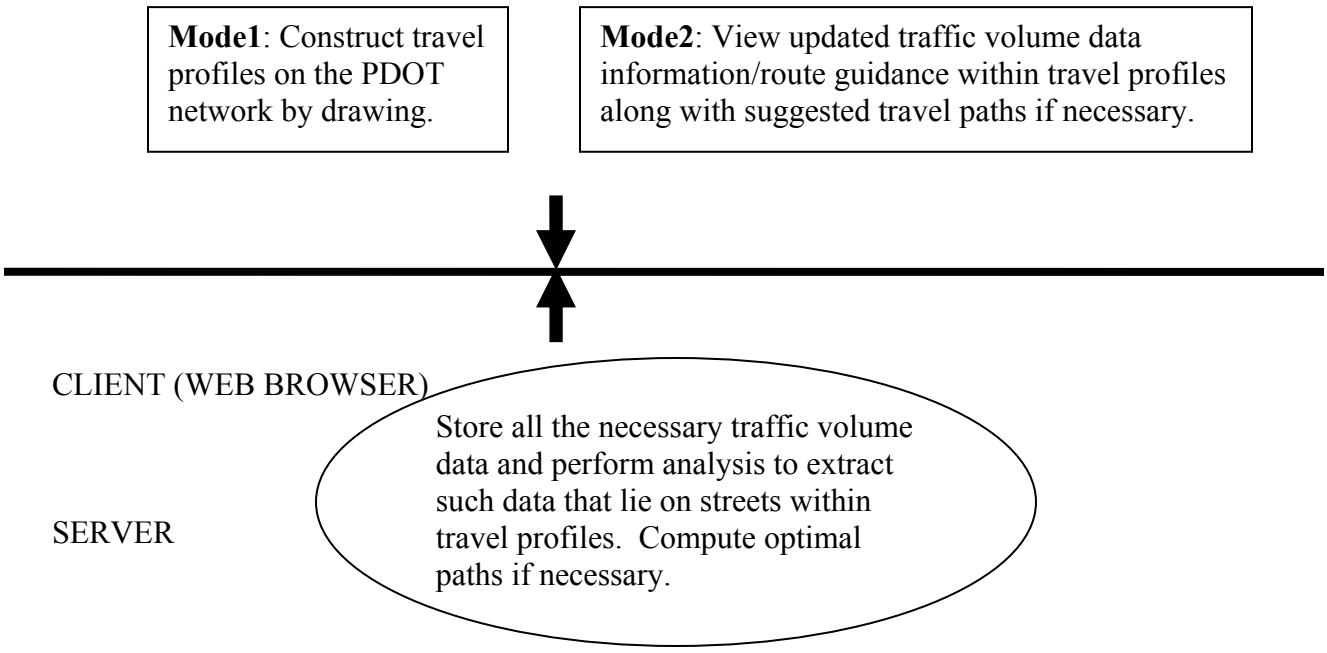


Figure 3.4 Overall web application design

information for regions and paths. For mode 2 to function effectively, updated traffic volume information on each network link within a profile region has to be uploaded from the server and has to be rendered on the client's browser. If this step is implemented using the profile region information entered in mode 1, concise real time traffic information can be delivered to the client using a predominantly server-side processing. The details of implementing both modes are discussed in the next chapter.

CHAPTER 4

CREATING AND USING CUSTOMIZED USER PROFILES

Providing custom content to clients is not new. Various websites and Internet Service Providers (ISPs) like America Online (AOL) create custom profiles that store commonly used web page settings and images the client might use. For example, the Weather Channel (at www.weather.com) can personalize the content of each client's web page by showing the temperatures and related weather for locations that are of interest to the client. Applying a similar concept to real time traffic reporting using the Internet, this research proposes the use of traffic profiles as user specific regions of interest within the street networks (referred to as AOIs from now on).

These AOI shapes or routes are created on the client's computer and transmitted as shapes or text from the client to the server databases that are indexed by unique user numbers. In this way, the personal profiles can be retrieved every time the user logs onto the Internet application. Also provided is network analysis, such as providing suggested trip paths based on the shortest trip from an origin to a destination based on the current traffic volumes. Creating the necessary software so that the client can create these aforementioned travel profiles is the first step in this research.

Custom Areas of Interests and Potential Travel Paths

There are many ways to define a profile representing a frequently traveled route. Statistical methods that use actual archived travel data are ways to automatically determine a frequently traveled route for the client. If sufficiently archived travel activities within travel areas are available, statistical methods can be used to roughly estimate the origins, destinations and expected routes of travelers. Trip paths can be obtained by having the user fill out an Internet survey form indicating a ranked list of frequently traveled origins and destinations, preferred interstates or highways and time windows for travel. The data could then be stored with the locations geocoded into a trip database with each user having a set of locations and other attributes.

In reality however, only a few real-time traffic websites allow the users to do this. For example, the Automobile Association (AA) Roadwatch service (http://www.theaa.com/travelwatch/) transmits personalized routes to registered users in the UK. The overall layout of this service is illustrated in Figure 4.1. The web service first asks the user to input a start and an end location, both of which are then geocoded and stored on the server. The shortest path is then constructed (just like Mapquest trip planner available at www.mapquest.com) and traffic conditions like accidents or traffic jams on that route are indicated to the user using his mobile service/text messaging services. This is mostly a server-side service since no plug-ins or executables have to be run on the client other than the current web browser. The thin client (low software or bandwidth overheads on the client) service allows for the creation and storage (on the server) of user profile information. However, the traffic updates provided to the user are

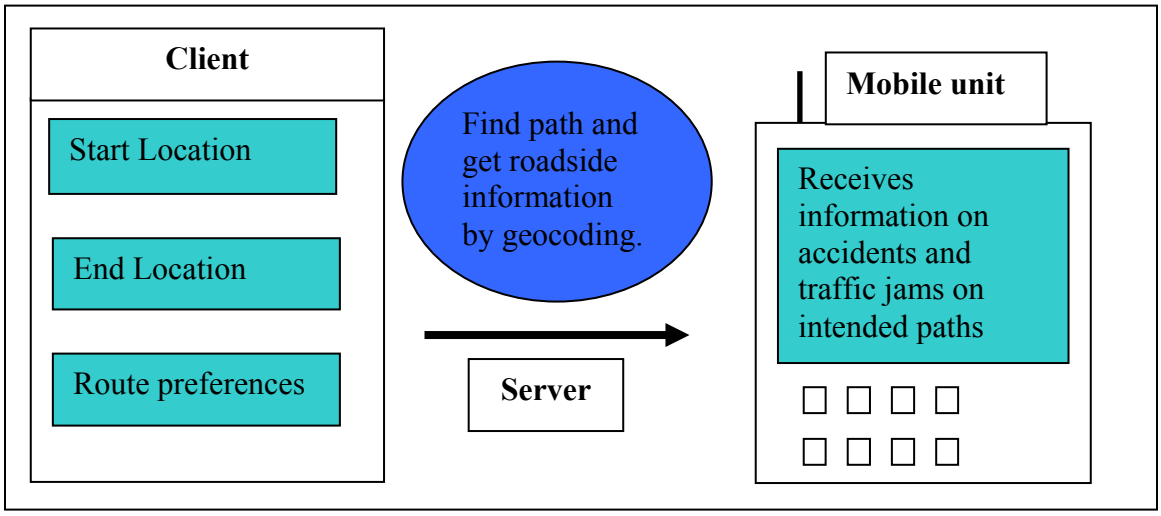


Figure 4.1 AA Road watch traffic path updates by mobile/text messaging.

not based on current traffic volumes nor are they presented back to the user as geographic data that can be put into spatial context. Thus, this provides a minimal 'D3' (customization of content and real time traffic data) website as per Peng and Huang's (2000) taxonomy, as GIS and visualization capability are absent even though some customization in trip planning is provided.

Table 4.1 summarizes some of the existing applications that use custom profiles (not necessarily traffic update applications) and the type of technology used in terms of client-side or server-side development. Trip paths can also be created by selecting a start location and an end location geographically. Simply allowing the user to select the start and the end location on a street map can define a potential trip. In Figure 4.2 for example, the user selects a start location and an end location by clicking on points that are near to identifiable landmarks such as a street intersections or places of interest.

Table 4.1 Major websites that use user profiles (both client and server-side).

	Generic sites using profiles: (Weather channel/AOI profiles)	**Mapquest/Trafficstation**	**AA Roadwatch**	**Puget Sound traffic site**
Method of creation of profiles	Survey form filled out by client	No profiles used. Updates are for entire cities and only major interstates/highways only.	Survey forms for start and end trip locations.	No profiles.
Storage and usage of profiles	Heavy client-side executables and dynamic link libraries (dll)	Updated maps are provided using considerable server-side processing. Simple browser suffices.	Server-side processing used.	Client-side executable does the processing.

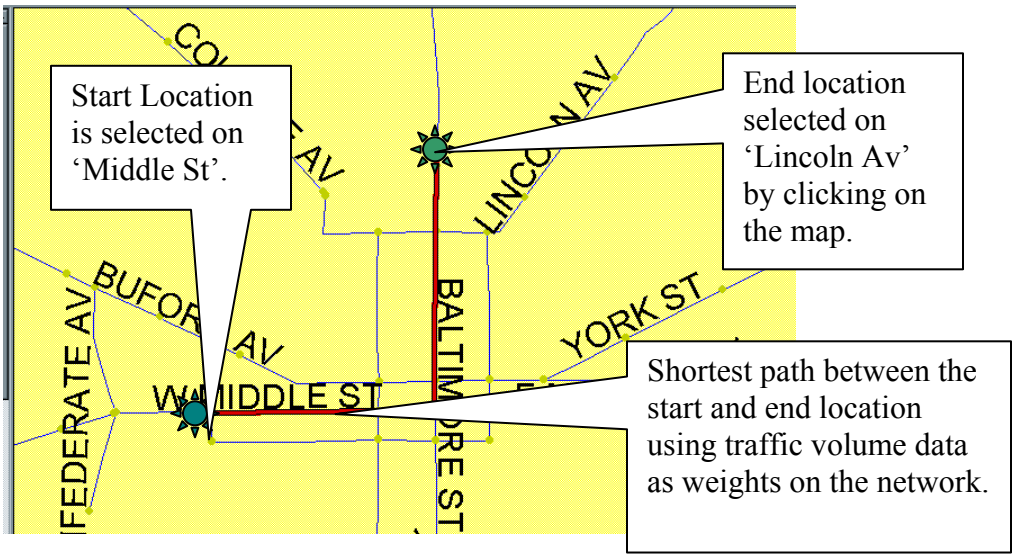


Figure 4.2 Selection of a potential path based on map selection of start and end trip locations.

Once a location pair is selected, only those coordinates need to be stored for the user to ask for a suggested travel path between these locations. To provide this path, server scripts have to calculate the shortest path between the start and end locations on the network. Traffic count values assigned to each link can be used as impedances for the shortest path calculation. These impedances are used to find the least cost path of travel. Such a travel profile that stores only start and end locations will be referred to as a potential shortest path.

Another method of path selection is to have the user select a frequently traveled path. This can be done by tracing out a path in between a start location and an end location along a set of links on the traffic network. The research defines a path between the two locations as a potential preferred path. In Figure 4.3 for example, the user can interactively click on a set of links from the start location to the end location to construct this preferred potential path. Each link ID is then stored and so that future updates can render the updated traffic volumes on these links. These updated traffic conditions in and along this preferred path will give the user pre-travel guidance. The selection methods above describe potential routes. However, sometimes a user might be more interested in a traffic update that covers a larger portion of the traffic network rather than just a path. For example, a client might want to monitor the traffic patterns in a specific area. For such cases, the customized profile may define the boundaries of that area. This is known as an Area of Interest (AOI).

Rectangles, squares, circles or irregular polygons can all be used as AOIs. This research uses rectangular AOIs for simplicity and proof of concept.

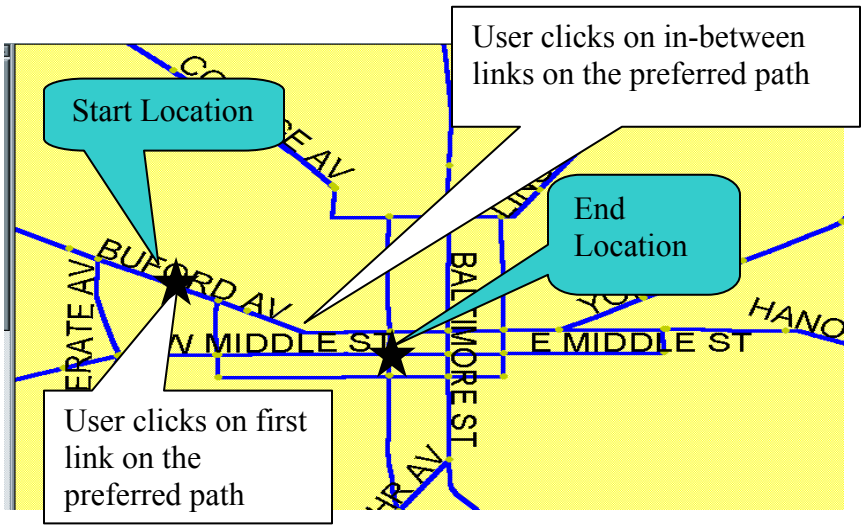


Figure 4.3 Selection of a preferred potential path based on selection of a set of links from the start to the end location

Each AOI has information linking it to the user that created it and an ID that uniquely defines its existence in a geographical database. Figure 4.4 shows an AOI defined as a rectangle by a client and the resulting portion of the network used for traffic updates.

Implementation Basics – Setting Up the Necessary Internet GIS Components

The first step in the implementation process is the selection and usage of an appropriate Internet GIS application. Internet GIS applications usually consist of software packages that allow a server to function as a public map server allowing interactive GIS transactions with spatial data on the server. For this research, the Internet GIS application will provide a template web interface that is shown to the client. This web interface will allow for basic GIS interactions such as pan, zoom and identify.

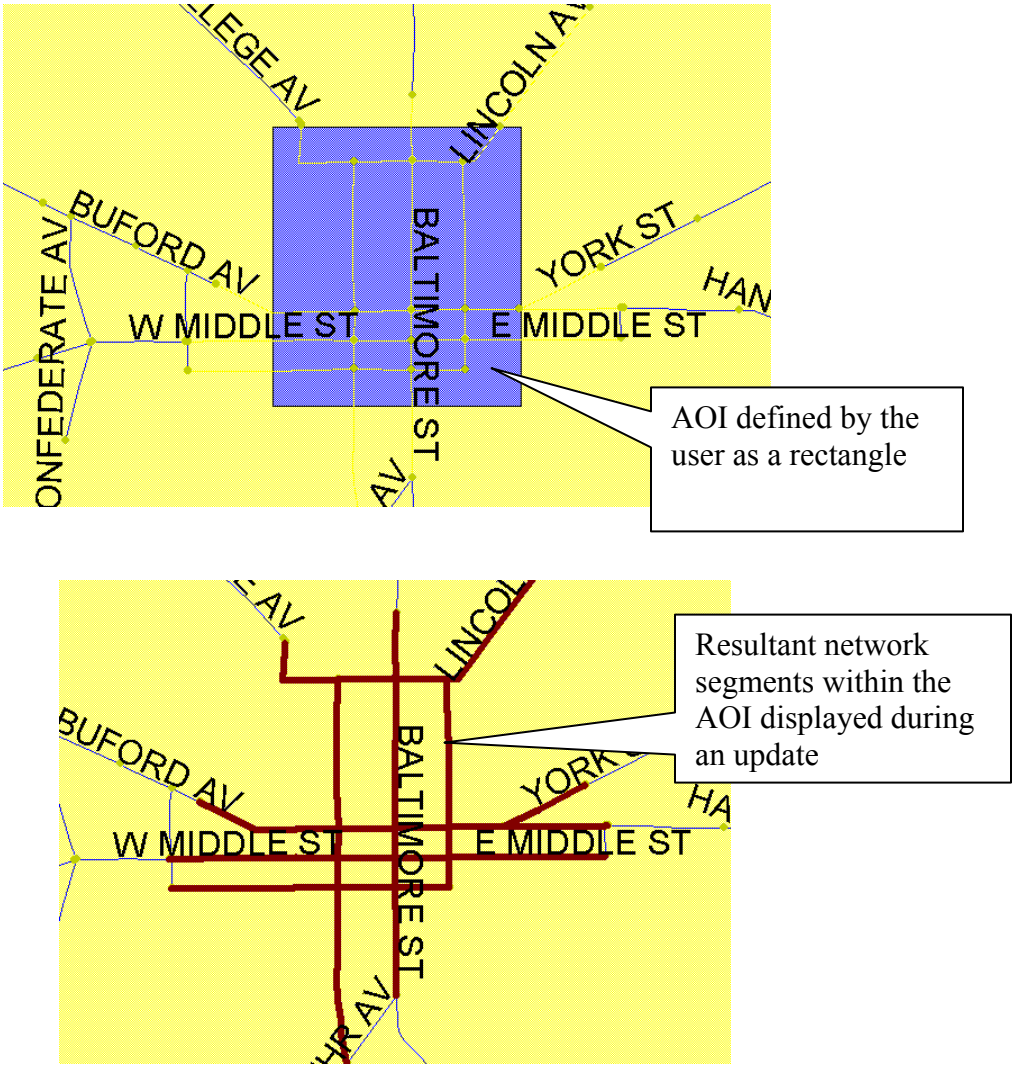


Figure 4.4 Defining an AOI using a rectangle limiting update areas.

Along with this, the web interface should also have the capability to communicate with the server to display spatial data like the background traffic network, AOIs and potential paths. With the basic GIS functionality provided, heavy customization (modifying existing scripts provided by the Internet GIS application) and additional scripting will allow for the creation of AOI and potential paths, which can then be stored and used for traffic updates.

Internet GIS software choices/comparisons

There are two major commercial Internet GIS packages currently available, GeoMedia WebMap and ArcIMS. Geomedia WebMap developed and marketed by Intergraph provides the client with a Component Object Model (COM) component that can be embedded in a web page. This component is called the ActiveCGM map control (http://www.intergraph.com/gis/support/technotes/Gwm2Impl.asp#2). The ActiveCGM control provides the necessary functions for communication between the client and data servers. Basic GIS requests such as pan, zoom and identify are supported. The client loads a web page as an Active Server Page (ASP). ASP pages are special Hypertext Transfer Markup Language (HTML) pages that allow for the instancing and usage of certain server-side components and COM objects. With the ActiveCGM control used within the ASP page loaded on to the client (if the client has an Internet Explorer browser), maps can be rendered on the client. On the server-side, the GeoMedia or MGE data servers must be loaded with the appropriate spatial data. GeoMedia uses the server-side service called GeoMedia WebMap in conjunction with a server-side process called the MapSrvMngr.exe and a number of map server processes each called MapSvr.exe.

The manager (MapSvrMngr) process receives the client ActiveCGM control requests and finds an idle MapSvr.exe to apply GIS functions to the spatial data stored on the data server. Thus, the MapSvr is really the application server that handles and completes all GIS transactions. Figure 4.5 outlines the broad architecture for GeoMedia WebMap based applications.

ArcIMS functions developed by ESRI are similar to Geomedia WebMap, even though the underlying technology is quite different. Figure 4.6 explains the basic ArcIMS architecture. ArcIMS, provides a GIS viewer that the client's browser loads. This viewer is an HTML and DHTML (Dynamic HTML for drawing functions) template that communicates with JavaScript modules that are loaded on the client. The main purpose of the viewer is to provide an interface for spatial data that is stored on the

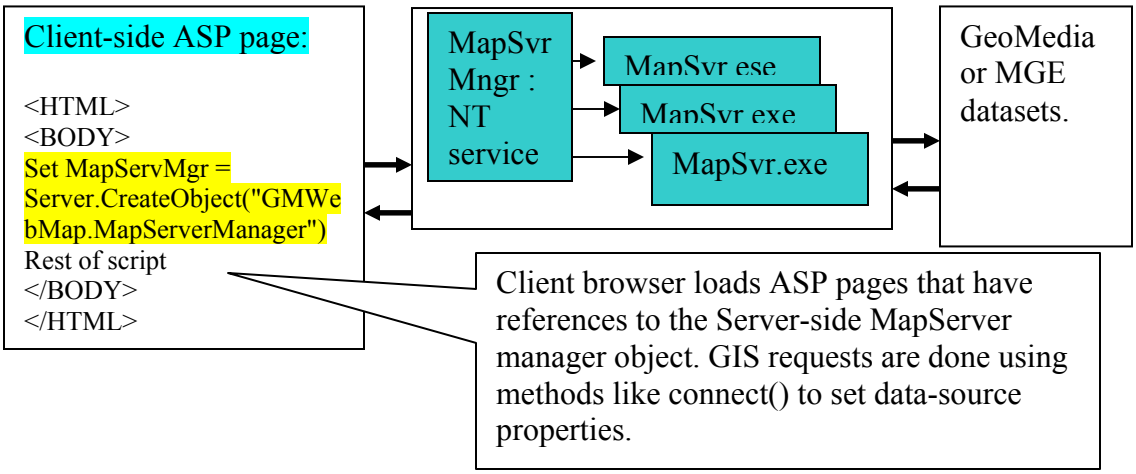


Figure 4.5 GeoMedia WebMap architecture featuring the client-side ASP page and server-side processes. Modified from: http://www.intergraph.com/gis/support/technotes/Gwm2Impl.asp#2

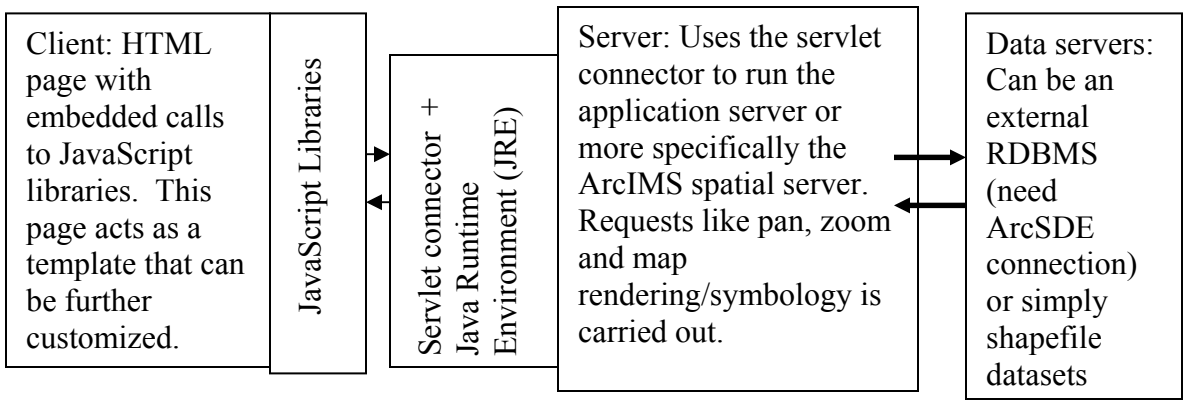


Figure 4.6 Overall ArcIMS architecture showing the flow of data from the client's browser and the data server/sources though the application server. Reproduced from: http://arconline.esri.com/arconline/whitepapers/ims_/arcims31arch.pdf?PID=6

server. Requests like pan, zoom and identification of spatial features can be done by the client using this viewer. The rest of the ArcIMS package is based on Java servlets and dlls that reside on the server. All requests from the client like pan or identify are sent via the TCP/IP network back to the server or more specifically the application server, which acts like a workhorse for rendering these requests.

For this research, ESRI's ArcIMS was used as the Internet GIS package to build the customized profile storage and traffic update application. This product was chosen because of availability. Once ArcIMS was chosen as the Internet package for the deployment of the traffic update application, the next step was to develop the necessary software that allowed the client to load the ArcIMS viewer. Then the client would be able to click a button for creating and storing rectangles that represent the client's area of interest. Potential path profiles would be created in a similar fashion.

As previously mentioned, ArcIMS has an HTML viewer which consists of HTML tags that display a map image showing the necessary GIS datasets along with other tools for interacting with the data. Embedded within this HTML page is a series of calls to JavaScript libraries that are loaded on to the client during the display of the ArcIMS viewer. This allows for considerable modification by the developer. New JavaScript functions can be added and calls made to them to carry out custom functions. Also, existing JavaScript can be modified to add more custom functionality. Table 4.2 lists the major scripting languages that are used to implement the research methodology for storing and using personalized traffic profiles.

The First Step: Setting Up the Map Server (ArcIMS Map Service)

ArcIMS uses map services to group different websites on the server that can be visited by the client. For this research, the map service used for the traffic update and custom profile application is called the 'moresearchsite'. This ArcIMS service lets the developer choose the data that needs to be displayed to the client each time he or she accesses the website. The dataset that was added to the 'moresearchsite' map service was the PDOT street network data represented as two shapefiles. One shapefile contained the network junctions while the other contained the network links. As mentioned in chapter three, the PDOT street level data was also made available as an ArcInfo 8.1 street network for other server-side processing functions. Figure 4.7 illustrates the major datasets used.

Table 4.2 Major scripting languages used for research

Scripting language	Description
JavaScript	Client-side scripting language that allows for programs to be run from the client's browser without initiating new individual processes on the client. Files have a *.js extension.
Microsoft Visual Basic (VB)	VB allows for the development of both standalone executables and macros within Microsoft applications. For the research, VB is used in conjunction with ESRI's ArcObjects, which provides many COM classes needed for geometry creation, storage and shortest path calculation procedures.
ASP	ASPs are scripts that run on the client but use Microsoft's IIS (Internet Information Server) to transmit and execute actions on objects like Microsoft Access or Outlook. ASP may be written in either VB or JavaScript. Files have a *.asp extension.

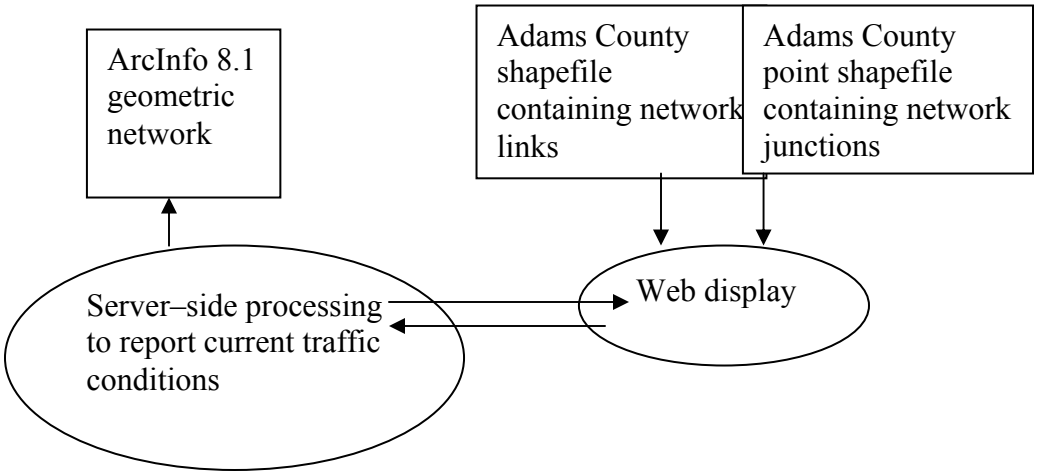


Figure 4.7 The Adams County network for Southeast Pennsylvania

Using ArcIMS to Provide Updated Traffic Information to the Client with Areas of Interests

Once the network junctions and links were added to the ArcIMS map service for display, the next step was to create a storage database to store client profile information. The initial plan for the research was to have separate shapefiles that store each geographic AOI or potential path profile. However, this method was not feasible. Figure 4.8 shows the basic reason for the failure of this method. Using shapefiles to store individual AOIs required that ArcIMS allow online editing (addition or deletion) of new shapes. However, geographic data in an active map service cannot be edited, so this approach was abandoned. Instead of using shapefiles, the strategy employed simply stores the coordinates that define rectangles for an AOI shape in a Microsoft Access database. In the case of potential paths, network link IDs or network junction IDs are stored.

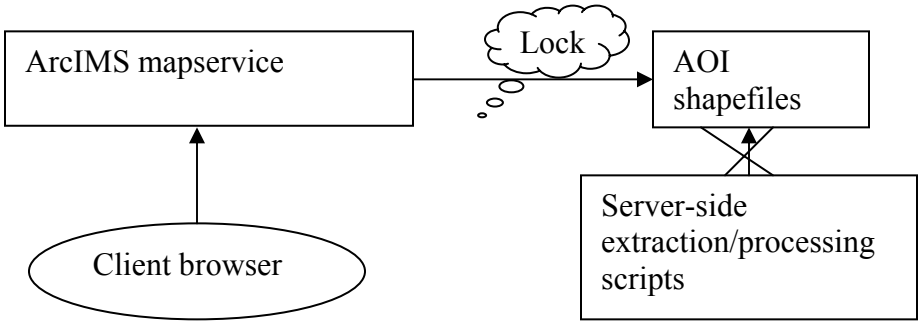
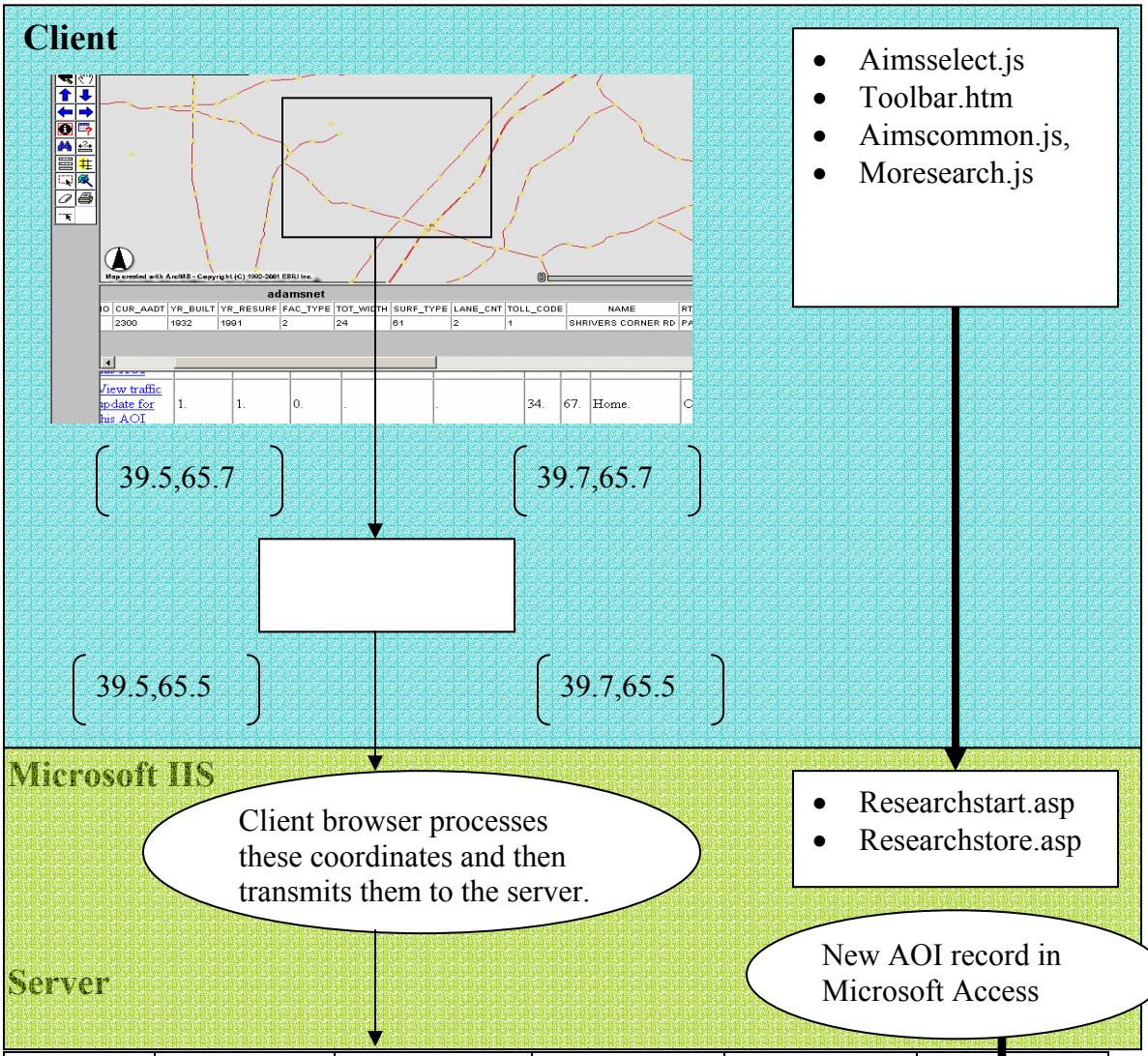


Figure 4.8 ArcIMS locks shapefile storing AOI shapes.

Figure 4.9 illustrates the methodology used to store newly created AOI shapes and potential paths. The client logs into the research website. If it is the client's first visit, he or she is assigned a unique user number. The research website uses this user number to display all existing AOI shapes created by the user along with certain attributes such as descriptive names or comments. JavaScript customization of the client's viewer page allows the client to produce new AOI regions interactively on his or her browser by drawing to the ArcIMS acetate layer. The acetate layer functions as an overlay layer on the map. It displays graphical information that does not belong to a locked geographic data layer. Elements on the acetate layer, therefore, can be edited by individual users using custom scripts.

Using the network links as background reference, the client can then trace out a region defined as a rectangle on the acetate layer to represent a frequently traveled region. The resulting shapes are stored as profiles so that subsequent actions from the same client will display the traffic volume along each network link within this rectangular region.



usernum	MinX	MaxX	MinY	MaxY	Uniqueid
1	39.5	39.7	65.5	65.7	1
1	39.3	39.9	64.5	66.9	2
2	39.6	43.6	65.6	66.8	3
3	39.8	44.4	66	67	4

Figure 4.9 Process for storing a new AOI for future use.

After development of the drawing software, the HTML viewer displayed on the client was modified (again using JavaScript and ASP) to record the bounding coordinates of these newly drawn AOIs.

In Figure 4.9, the user traces out a region defined with Y coordinates from 65.5 to 65.7 while the X coordinates lie between 39.5 and 39.7. These coordinates define the drawn rectangle using the four numbers, minimum X and Y and a maximum X and Y. The research modified the template ArcIMS JavaScript to allow for the recording and transmission of these coordinates from the client directly to the server for permanent storage in a Microsoft Access database. Scripts named 'Toolbar.htm', 'Aimsselect.js', 'Aimscommon.js' and 'Moresearch.js' were either modified or created to allow the template to draw and record these coordinates in the Microsoft Access table 'tuserpoly' (refer to flow in Figure 4.9). The ASP script 'Researchstore.asp' was created to allow this transmission and storage of AOI shape information from the client to the server database. Microsoft Internet Information Server (IIS) was used as the web server. All these scripts are included in Appendix A.

Storage of AOIs within the 'tuserpoly' database creates descriptions of geographic profiles of intended travel regions or interest areas. The research developed a system to allow the users to select individual AOI profiles on their web browsers. Using this system, they can receive updated traffic conditions along the streets or highways that fall within the AOI.

The first step in this process allows the user to select an AOI shape that he or she would like to use for a region to receive traffic updates. As shown in the Figure 4.10 flow diagram, the user selects an AOI from one of the records displayed on the bottom of

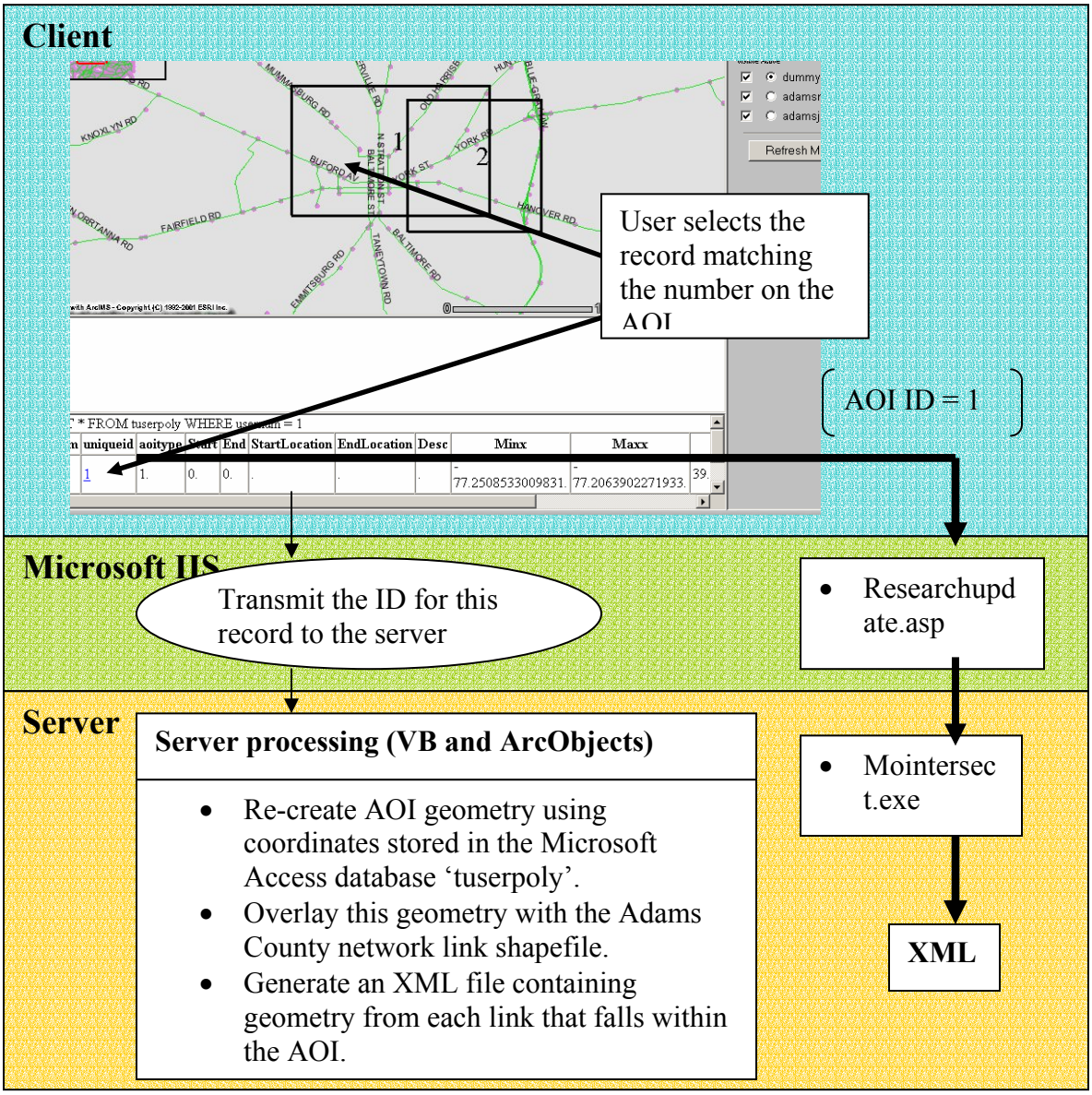


Figure 4.10 Flow diagram for the selection and extraction of traffic volume data within AOI rectangles

the browser screen. Clicking on the table entry that matches the number displayed on the AOI rectangle completes this selection. This reference number or record is then transmitted to the server using an ASP script called 'Researchupdate.asp' (refer to Appendix A).

More specifically, this ASP script transmits the unique ID to a server executable ('mointersect.exe') written with VB and ArcObjects (Figure 4.11). This transmission of the unique ID to the ASP script is done by reading in the unique ID from the client and then querying the minimum and maximum bounding coordinates of the AOI stored in the 'tuserpoly' table. Once these coordinates are read, the ArcObjects/Visual Basic program creates a rectangular geometry shape using these coordinates and overlays this rectangle on the Adams County network link shapefile.

All the network links that fall within the rectangle are recorded by reading their geometry and traffic count values into an eXtensible Markup Language (XML) file as shown in Figure 4.12. XML in this portion of the research was used as a transient geometry storage file. A sample XML file is shown in Figure 4.12 containing all the links resulting from an overlay using an AOI.

The parsing and display of updated traffic volumes follows the creation of the XML file with the traffic links. This parsing and display of these resulting traffic volumes was achieved by running the scripts 'Aimscustom.js' on the ArcIMS client (Figure 4.11). This is shown in the final snapshot of Figure 4.11. Once the client browser parses the XML file, an ArcIMS acetate layer displaying these links is created. The ArcIMS acetate layer is merely a graphic that ceases to exit after the viewer session is completed. Hence, it can be referred to as a dynamic layer.

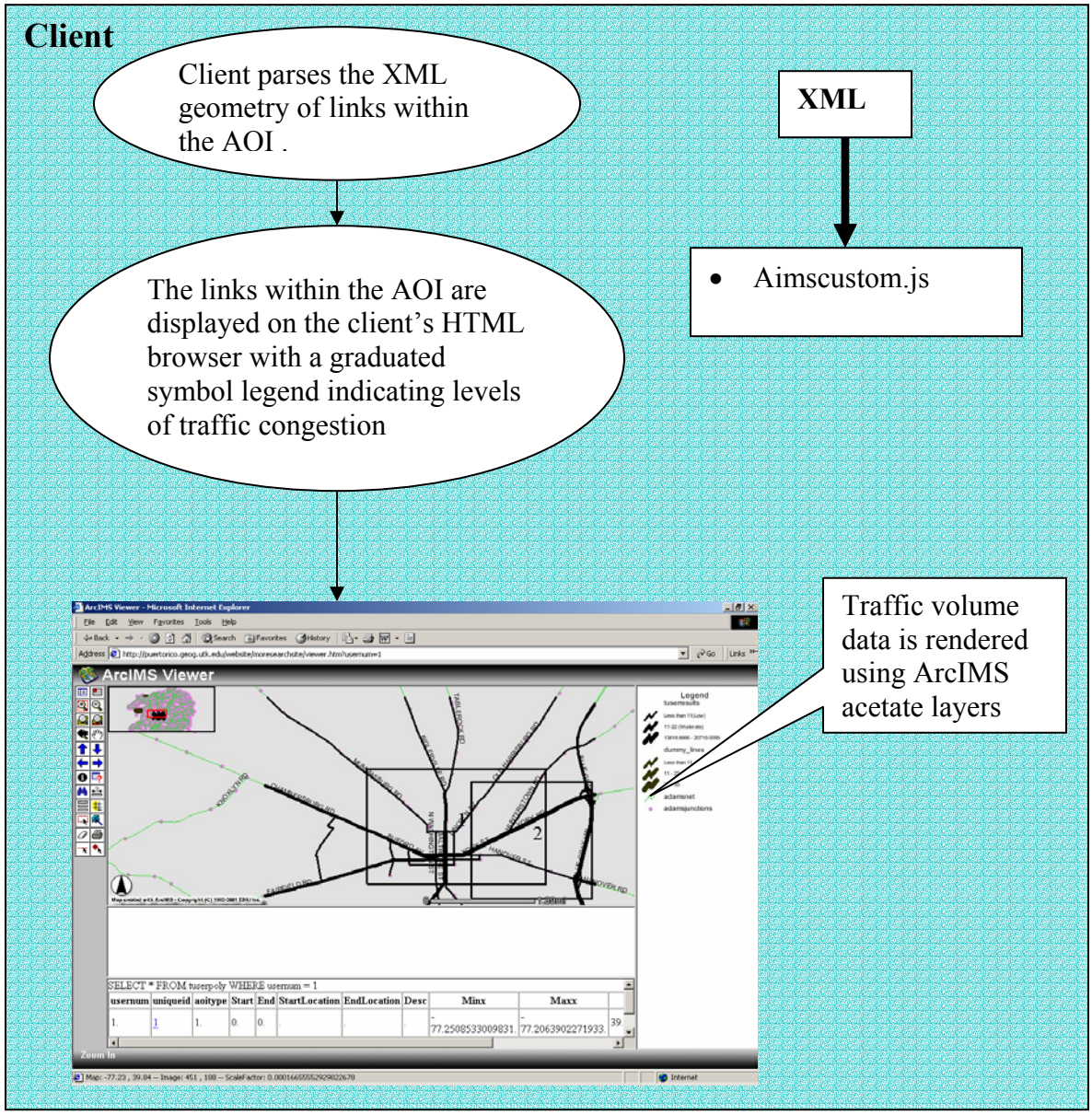


Figure 4.11 Transmitting traffic volume data from the server back to the client using XML

```
<USERSHAPE>
   <OBJ>
      <X>-77.2156699999996</X>
      <Y>39.8367199999993</Y>
      <X>-77.2142899999999</X>
      <Y>39.8388400000003</Y>
       <X>-77.2129499999992</X>
      <Y>39.84087</Y>
      <X>-77.2116499999993</X>
      <Y>39.84274</Y>
      <X>-77.2116499999993</X>
      <Y>39.84274</Y>
      <X>-77.2116499999993</X>
      <Y>39.84274</Y>
      <X>-77.2216499999999</X>
      <Y>39.84273</Y>
     <X>-77.3116777766666</X>
     <Y>39.84284</Y>
     <X>-77.2116499798993</X>
     <Y>39.84274</Y>
     <X>-77.2116499900000</X>
     <Y>39.84474</Y>
     <X>-77.2116499234900</X>
     <Y>39.84234</Y>
     <AADT>3003</AADT>
   </OBJ>
</USERSHAPE>
```

<X> and <Y>tags store coordinates for the link under the <OBJ> tag

Figure 4.12 XML file storing updated traffic links

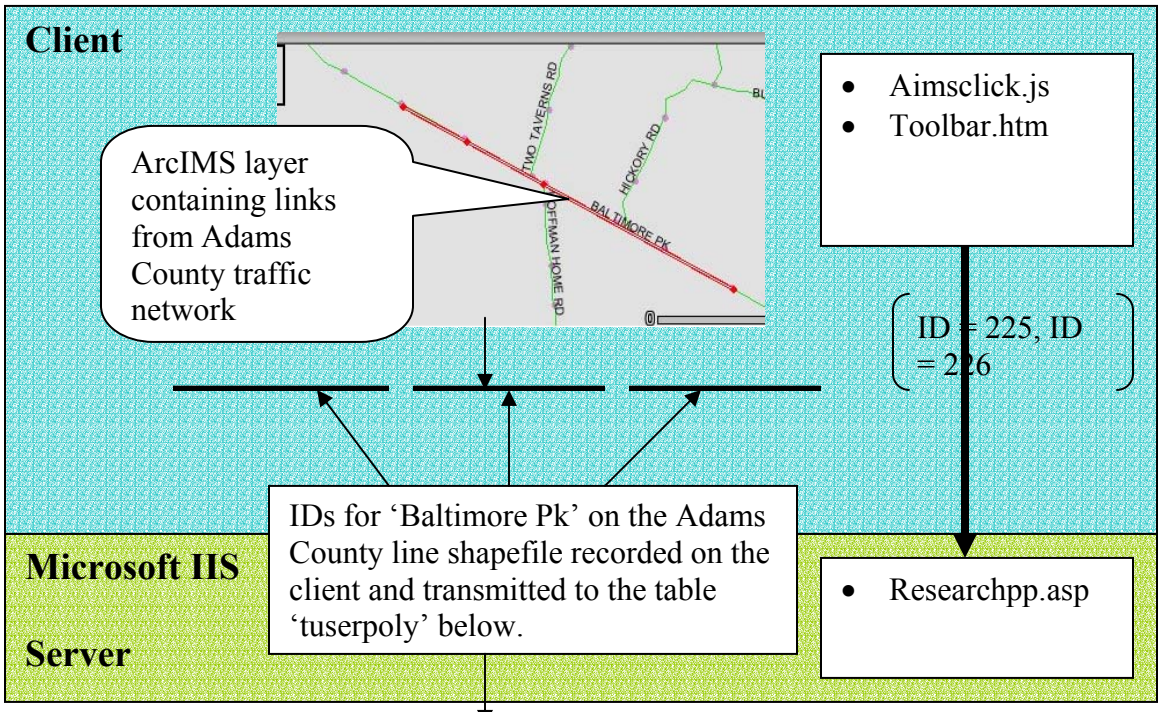
To render the traffic volumes, the acetate layer tells ArcIMS to display the traffic links to the viewer using a certain size and color. After ArcIMS reads and visualizes the XML file by using acetate layers, graduated symbol lines are shown with the zoom levels adjusted so that unnecessary links outside the AOI do not display. It should also be noted that a similar acetate-XML method is used to display previous AOI shapes when the user first logs in with a unique user number.

Using ArcIMS to Provide Updated Traffic Information to the Client with Potential Paths

As described earlier, an AOI is not the only geographic user profile used in the research. Potential paths that describe possible routes for the user are also profiles that can help the user make a pre-trip route choice. The first type of potential path is the preferred path. These profiles are used by the client to view updated conditions on a frequently traveled route. This path is rendered with updated traffic volumes before the client's intended trip.

Preferred Path Updates Using ArcIMS

In addition to storing links that compose a path, it also is possible to store only the origin and destination of a path and use a path algorithm to select the relevant links. Figure 4.13 shows the flow for potential path creation, storage and use in traffic information display. The first step requires the user to select a set of links that start between the user's intended start location and end location on the traffic network and subsequently store these links as profiles on the server. The script 'Researchpp.asp'



User num	MinX	MaxX	MinY	MaxY	aiotype	uniqueid
1	34.5	34.8	48.5	49.7	0	1
1	39.3	39.9	64.5	66.9	1	2
2	39.6	43.6	65.6	66.8	1	3

Usernum	linkid	uniqueid
1	225	1
1	226	1

Network link IDs stored in the loopkup table 'Pplookup'

Figure 4.13 Methodology for creating preferred paths

included in Appendix A accomplishes this task. This script works in conjunction with 'Aimselect.js' and 'Toolbar.htm' on the client browser. Once the user selects a set of links to form a preferred path, the unique ID for each link in the set is stored in the Microsoft Access table 'pplookup'. These are unique ID's for each link on the Adams County network. At the same time, the script 'Researchpp.asp' also inserts a single master entry for this path in the 'tuserpoly' table. Thus, as shown in Figure 4.13, each record in the 'tuserpoly' table for a single preferred path corresponds to a set of links in the 'pplookup' table. These two tables are linked together by the 'unique ID' field in the 'tuserpoly' table using a one-to-many relationship. The contents of the master table 'tuserpoly' are presented at the bottom of the user's screen allowing the user can select respective AOIs or potential paths.

Once the links selected for potential paths have been stored, software for allowing the user to view updated traffic conditions on a potential path is used to display the custom travel information. In Figure 4.14, this process is illustrated by showing the user clicking on a Hyperlink (containing the unique ID of the preferred path) that displays the current traffic volumes. Once in this mode, the server-side script 'Linepp.exe' queries the links within the preferred path from the 'pplookup' table. Using XML as a transient geometry storage file, subsequent XML parsing and conversion to ArcIMS acetate layers re-create the geometry of this potential path for. This potential path, from origin to destination, is shown in graduated symbols. As shown in Figure 4.14, the scripts 'Aimscustom.js' and 'Researchupdate.asp' (refer to Appendix A for code details) have been used for uploading of traffic volume information from the server to the client.

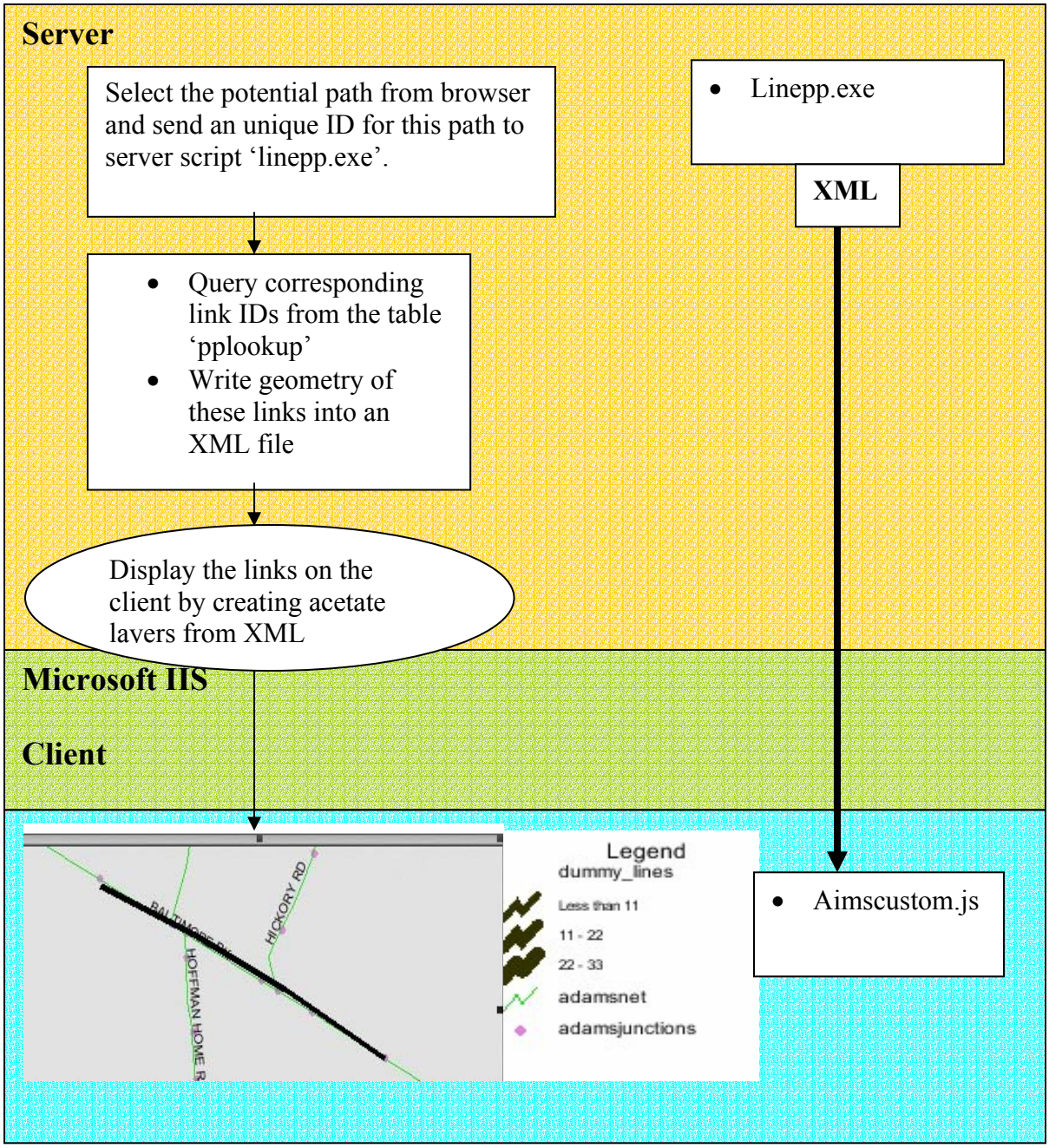


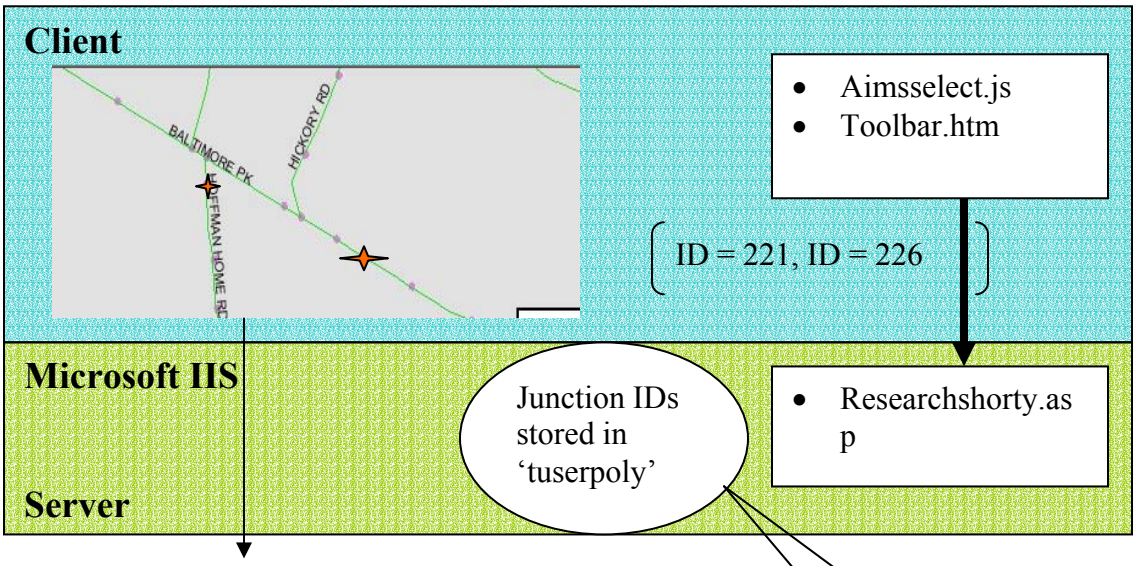
Figure 4.14 Methodology for using preferred paths for traffic updates

Shortest Path Updates Using ArcIMS

The major difference between a potential path and a shortest path is that the latter provides a recommended path based on the current traffic volumes. The ArcIMS HTML client was customized to handle this by allowing the user to select a network junction point. Client-side JavaScripts ('Aimsselect.js' and 'Toolbar.htm') record the unique IDs of these junction points and transmit them back to the server for storage in the 'tuserpoly' table. Refer to figure 4.15 for an illustration of the junction ID storage process.

Once these locations (junction IDs) have been stored as AOIs and preferred paths, server-side scripts for providing updated traffic conditions were developed. Since shortest paths in this research did not necessarily have to display real-time traffic volumes, the set of links that constitute the shortest path between the stored locations based on the current traffic volumes was computed. This computation was performed on the server using the ArcObjects and VB program 'Shortestpath.exe' included in Appendix A. This program reads the start and end IDs for a profile path on the Adams County junction shapefile from the server table 'tuserpoly'. It then identified these as start and end flags on the Adams County geometric network.

The geometric network allowed for shortest path computations. Using ArcObject interfaces, the least cost path between the start and end junctions was computed using the traffic count values as impedances (identified as 'weights' by ArcObjects). This shortest path was extracted as a polyline and converted (just like links from AOI overlays and preferred path construction) into a XML file for display on the client (Figure 4.16).



usernum	MinX	MaxX	MinY	MaxY	aoitype	uniqueid	Start	End
1	36.9	40.4	67	66	2	1	221	256
1	39.3	39.9	66.9	64.5	1	2	456	406
2	39.6	43.6	66.8	65.6	1	3	334	456

Figure 4.15 Creating and storing shortest path profiles

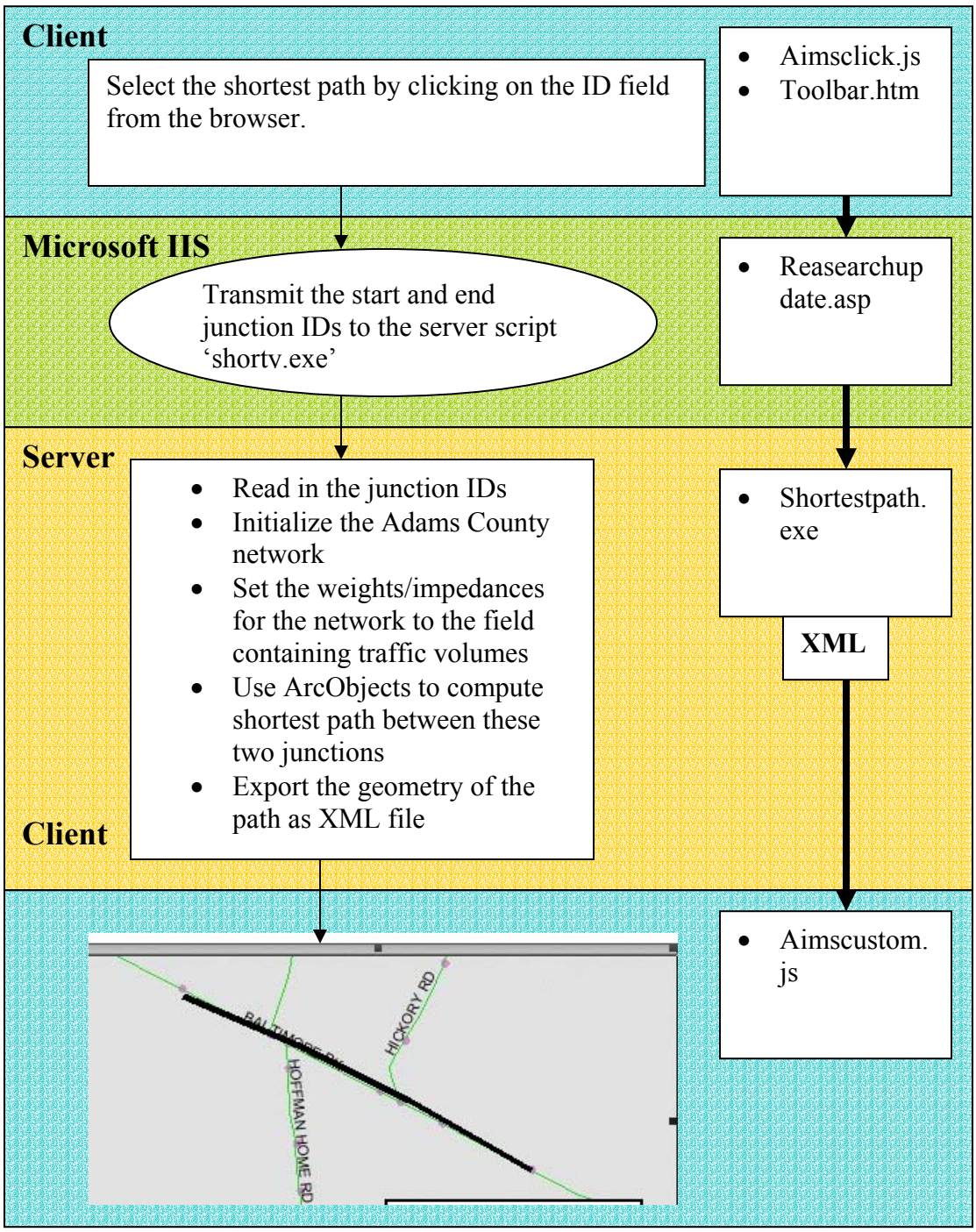


Figure 4.16 Methodology for using shortest paths.

CHAPTER 5

RESULTS AND FURTHER RESEARCH

This research implemented an Internet application that provides the user with street level traffic updates in real-time. With the use of browser-based tools, clients are updated on traffic conditions within a predefined region on a network. The client does not have to navigate through the entire network to obtain updated traffic information related to only a portion of the entire network. As a result, browsing and data transmission time for getting this updated traffic information are reduced. In addition, the client only has to use a standard JavaScript enabled web browser, thereby keeping the entire application as a server-side implementation. The first three sections of this conclusion will outline some of the functions provided by the Internet application developed in the research. The remaining two sections will conclude the thesis by discussing some of the GIS software implementation challenges faced while developing the customized web service and further challenges down the road for similar web services.

Creating and Using Rectangular Travel Profiles (AOIs)

The first tool produced by this research is a Graphical User Interface (GUI) provided to the client via the Internet. This GUI allows the user to log in and store frequently traveled regions of the travel network in question. The methodology chapter (chapter four) indicated that this could be done by three methods. The first method is the

generation of traffic profiles as rectangular regions on a street network. Figure 5.1 shows the web browser screen snapshot of a client that created rectangular profiles (AOI shapes) by drawing Dynamic Hypertext Transfer Protocol (DHTML) on his or her browser with the street network as backdrop. Once new AOI's are drawn, the user is allowed to enter some additional descriptive properties in the text box on the lower left corner. This new AOI is immediately added to the master profile storage database, while the attributes are added to the table view in the bottom frame of the viewer. This interactive creation and viewing of profiles give the look and feel of a shopping cart commonly used for online transactions.

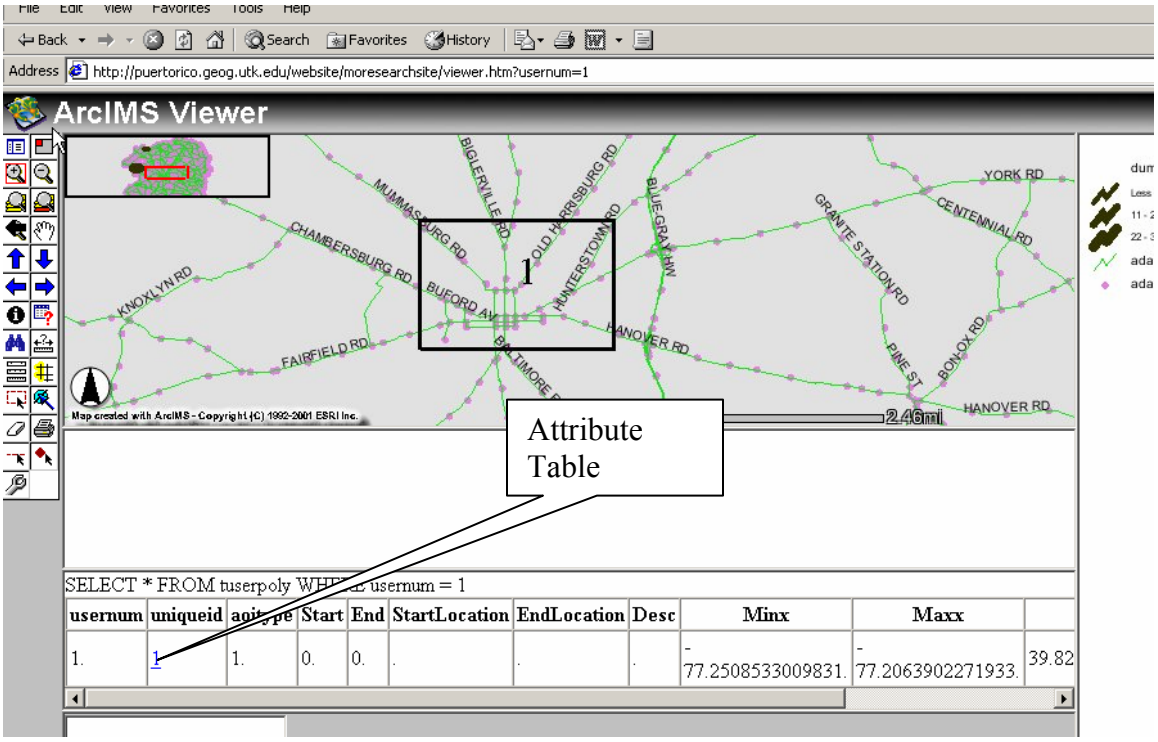


Figure 5.1 AOI profile rectangles drawn and stored for future use

Once such a rectangular AOI region has been stored, the user can exit the application or request updated traffic volumes for street segments that fall within one of these AOIs. Figure 5.2 illustrates this updating process. The request for an update is sent to the server by clicking on the attribute table which returns a graduated symbol graphic representing the traffic volumes on street segments that fall within the selected AOI. In Figure 5.2, this is shown as the streets within the AOI numbered '1' in Figure 5.1. The graduated symbol rendering shows thicker symbols in black that represent heavier traffic volumes along those street segments.

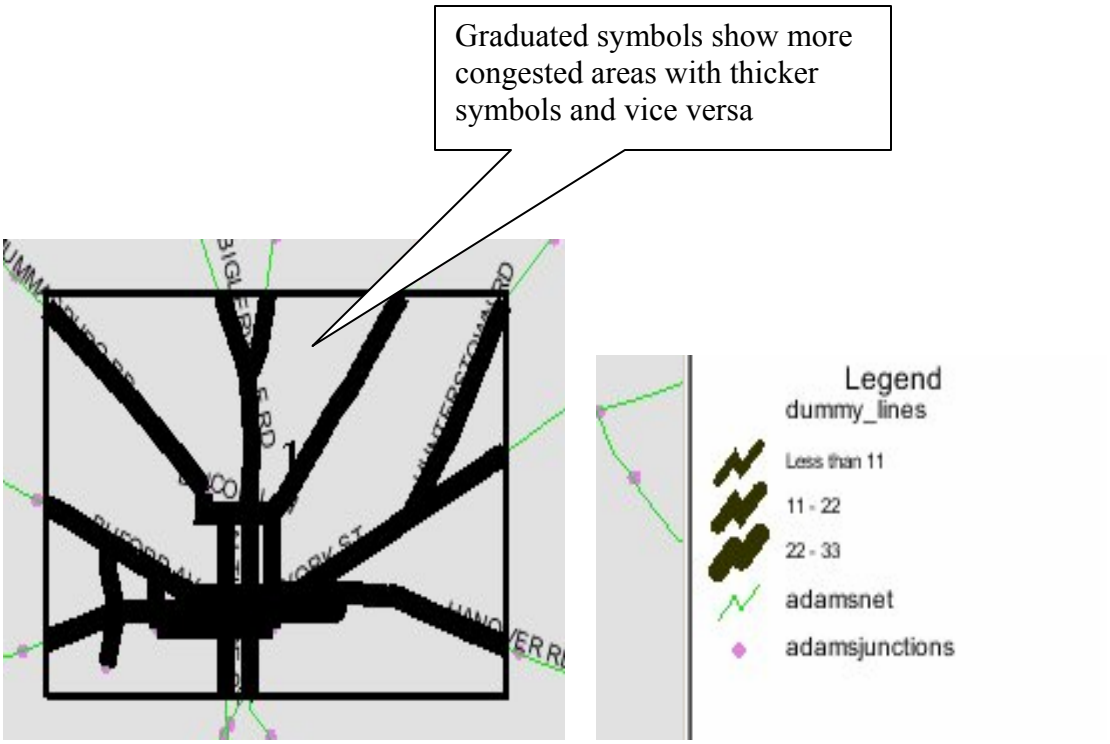


Figure 5.2 Traffic volumes rendered within the AOI identified as '1' in figure 5.1

Thinner symbols indicate relatively lighter traffic congestion conditions.

AOI's are useful for providing updated traffic conditions within a user-defined geographic region. Their utility is more pronounced when no intended travel path is predetermined. For example, viewing traffic volumes in a portion of a downtown area can lead to advanced pre-trip information which can help the public better plan their trips. Heavily congested areas can be avoided. Along with this, utilization of traffic profiles keep the data browsing at a premium since information is presented to the user on a need only basis. However, AOIs are not very useful when the travel path is pre-determined. This means that sometimes, even though updated traffic condition information is desired, the travel path from the start location to the end location is fixed. If these profiles are stored, the user can receive updated conditions only along this pre-determined path. These types of profiles are classified as preferred potential paths.

Creating and Using Potential Preferred Paths

Potential preferred paths provide the client with customized real-time traffic information along an intended travel path. Using a web browser, the user clicks on a start location and constructs a continuous line along the street network. This line should stop at the end location of a frequently traveled route. These routes can be alternate home-work routes or non-periodic travel paths like a shopping mall trip or travel in a new city. Once this profile represented by a path is stored, updated traffic volumes along this path can be graphically rendered using graduated symbols (refer to figure 5.3).

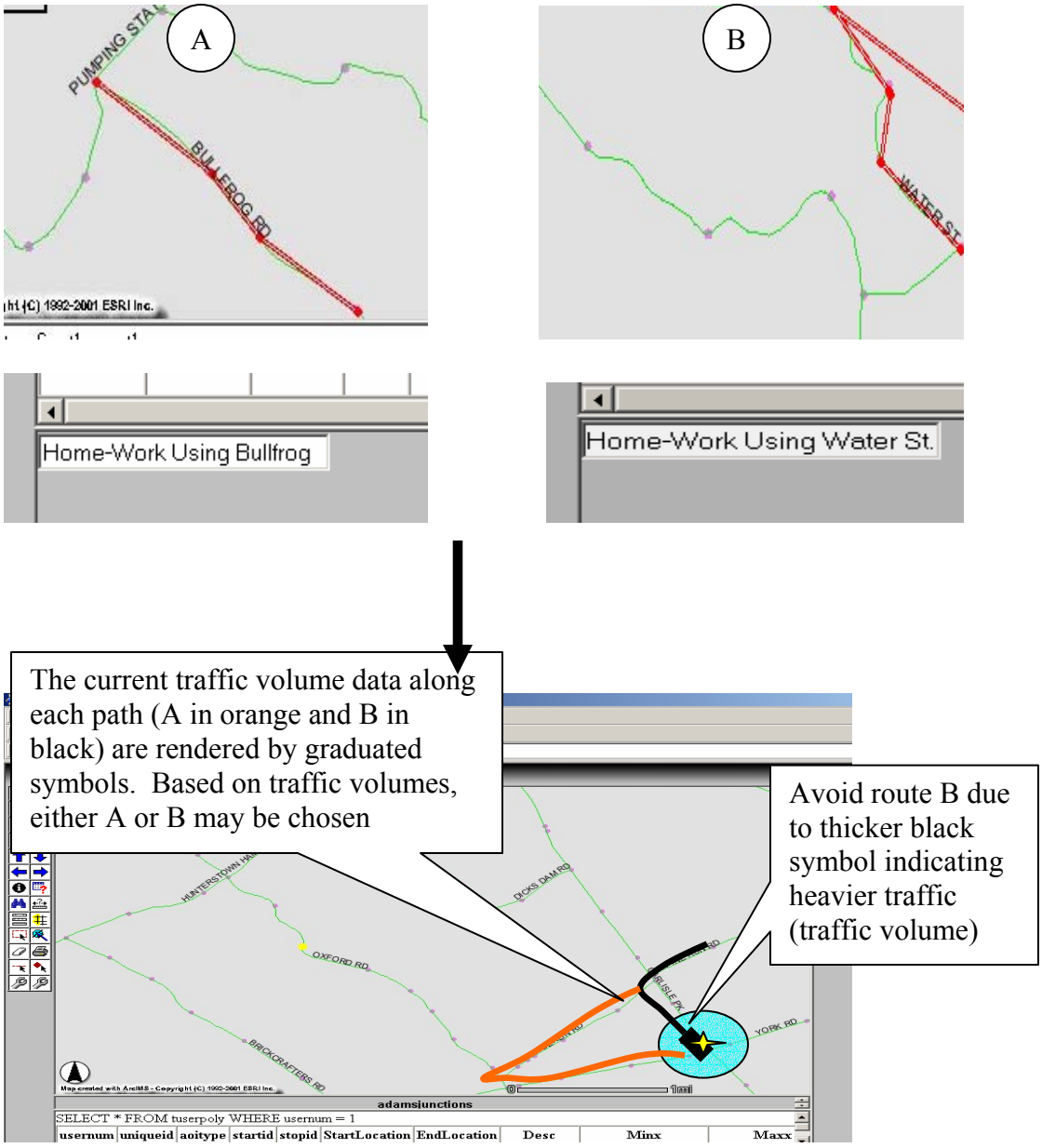


Figure 5.3 Traffic update information using potential preferred paths

In Figure 5.3, either one of the routes stored in profile A or B can be selected by the client. Before the intended trip, the client can inspect the volumes on both the 'Bullfrog' and 'Water Street' route profiles. The route with the least congestion (thinner symbols representing lower traffic) would most likely be the best choice.

Suggested Trip Paths Using Shortest Path Potential Profiles

The third and final profile method implemented in the research is the shortest path route guidance system. The profile creation and storage for shortest paths are less complicated when compared to the tools discussed above. The client simply selects a start and a stop location for an intended travel route. The identification points for these locations are stored as profiles. Before travel, the client can use these pair of locations to receive a suggested path computed by the application. This suggested path is the shortest path between the two locations using the path that follows the lowest traffic congestion (Figure 5.4).

This research successfully implemented a web server application that integrates the concepts of profile management and real-time traffic presentation in using server-side application architecture. While the research accomplishes a proof of concept endeavor, there remain certain challenges from the standpoint of software development, data acquisition and integration into GIS.

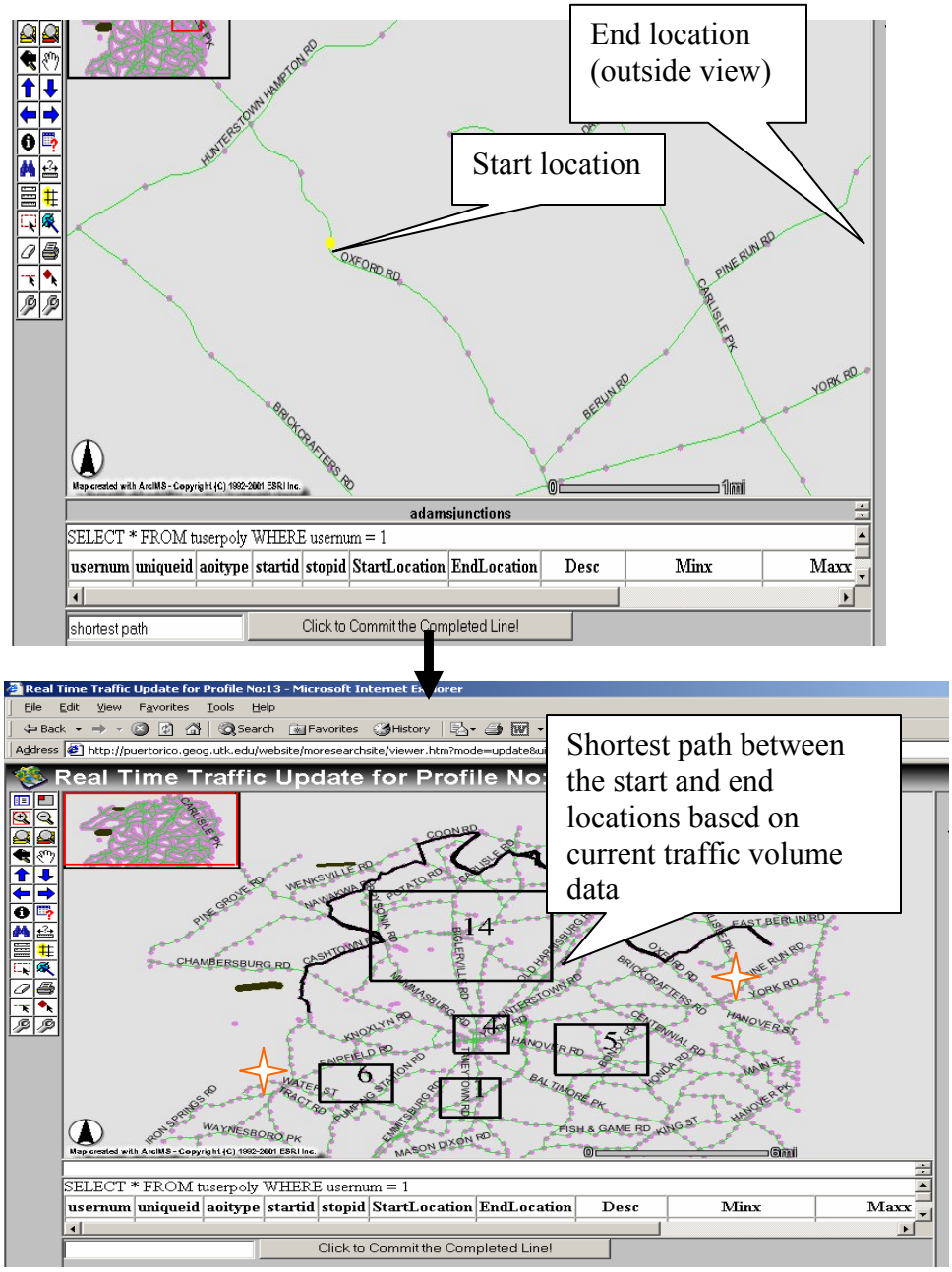


Figure 5.4 Shortest path route guidance

Implementation Challenges

ESRI components were used for much of the implemented traffic update application. ESRI's ArcIMS was used as the Internet GIS software. While this software was adequate for most of the common requests like handling the visualization of street network and the actual display of updated road segments, major customization was required. The customization can be summarized as follows:

1) A communication channel had to be built to allow for the storage of rectangular areas of interests on the server for each client. Since no shapes were stored, the communication channel would receive AOI envelope coordinates drawn for each client and then transmit these coordinates along with other attribute information to the server for storage. This could not be achieved using the traditional ArcIMS data loading of shapefiles into a map service.

2) Transmitting updated traffic volume information to the client also required another communication channel. The major difference from the channel required to store travel profiles was that the response from the server contained geographic information that had to find the client that requested for the information. This was more difficult to implement, but was achieved by producing temporary eXtensible Markup Language (XML) files that contained geometry definitions for links that needed to be updated. These updates are initiated by a call to a server-side executable. Once initiated, the execuatable finds the necessary Microsoft Access tables and creates the XML files containing coordinates that constituted each polyline within an AOI area or a potential path. The current, updated traffic volumes on each link are also encoded within the XML file. This XML file is displayed using dynamic display ArcXML acetate layers.

The major methodology choice for creating both of these communication channels was the usage of ASP scripts that retrieved coordinate and attribute information for AOI shapes and potential paths. Text files were transferred from the client to the server and usually consisted of an ID number or a string containing parameter values. Hence, the research used a method of text file transmission to produce real-time traffic maps and for customization. Even though this methodology proved effective, relying on the customization of the client HTML viewer through an increased usage of client–side JavaScripts might handicap future development in the following ways:

1) Display options are limited to the Netscape or Internet Explorer Document Object Models (DOM). The DOM is a standard that these browsers conform to when the developer gains access to certain objects that allow for drawing and browser related operations on the client. For example, drawing the geometry of updated traffic links requires the usage of the client's browser DOM (either Internet Explorer or Netscape). The research uses a legend that is fixed with three traffic volume intervals: low, moderate and high. The acetate layers drawn via the usage of DOM objects are merely graphics and not GIS layers since no relationship between the data and actual link graphics is developed.

2) Heavy use of client-side Dynamic HTML (DHTML) used for the display of spatial data can obscure the basic utilities provided by Internet GIS software packages like ArcIMS. For example, developing appropriate DHTML layers to show traffic volumes on the server almost repeats the very functions ArcIMS tries to provide. It is like trying to re-create ArcIMS functionality by displaying and querying spatially indexed attribute data.

3) Using shapefiles and ArcSDE with Oracle along with ArcIMS might have been more effective for the implementation. This would take full advantage of existing spatial indexing and RDBMS functionality to manage concurrent requests and spatial transmission and queries. In addition, constant updates of traffic volume or traffic condition information can be facilitated through interactive editing and updating on the server.

The essential technology implemented through the research application began to lay a foundation for future projects integrating server-side storage of travel profiles with real time traffic updates. However, there remain broader areas of concern not directly related to software development or even GIS.

Challenges Down the Road

For customized geographic profiles to be effectively used as real-time traffic reporting tools, other concerns must be addressed. The first problem is the actual method of creating profiles. The research attempted to introduce three methods of simple geographic selection of travel profiles. For example, the preferred path selection tool implemented in the research allowed the user to trace out a frequently traveled path on a street network. Building on this, more advanced methods of profile path selection could be employed. Shifting from merely applying a geographic approach, one could use statistical data of trip origins and destinations to build profile paths between start and end locations. Multi-modal trips can also be built and stored as travel profiles for more complex travel profiles.

However, profiles are merely tools that facilitate the data query and visualization of real-time traffic data. Without the availability of real-time data down to the street level, the usage of travel profiles only marginally shortens unnecessary browsing time and server upload loads. For profile usage applications to become effective, street level data for service regions has to be made available online. Initiatives are underway allowing Traffic Monitoring Service (TMS) and ITS (Intelligent Traffic Systems) programs to collect and distribute traffic volume data for streets as well as major highways in some metropolitan cities. As these programs mature, the need for custom methods of presentation of this vast quantity of traffic data also will grow.

REFERENCES

References

Albaredes, G. *Web Based GIS, New means to access spatial information*. <http://www.autodesk.com>.

ArcIMS: The ArcIMS3 Architecture and Functionality. ESRI White Paper. 2001. <http://arconline.esri.com/arconline/whitepapers.cfm?PID=6>.

ArcIMS versus MapObjects IMS: Architectures, ESRI. White Paper. 2001. <http://arconline.esri.com/arconline/whitepapers.cfm?PID=6>.

Dickman, A. "Two-Tier Versus Three-Tier Apps." *Informationweek* 13 Nov. 1995: 74-80.

ESRI Conference Proceedings, Map Objects and ArcSDE: Twentieth Annual ESRI Conference. 2000. <http://www.esri.com/devsupport/devconn/sde/presentations/uc2000.html.>

ESRI Conference Proceedings, ArcSDE for Oracle Administration: Twentieth Annual *ESRI Conference*. 2000. <http://www.esri.com/devsupport/devconn/sde/presentations/uc2000.html>

Foresman, W.T. *The History of Geographic Information Systems: Perspectives from the Pioneers*. Upper Saddle River, NJ: Prentice Hall Inc, 1998.

Johnson City MPO. *Frequently Asked Questions: Average Annual Daily Traffic.* 2001. <www.jcmpo.org/traffic/faq.htm>.

Gilroy, R., R. Puentes and R. Schuman. *Advanced Traveler* Information Systems: Choosing the Route to Traveler Information Systems *Deployment.* ITS America. 1998 <http://www.fhwa.dot.gov/tfhrc/safety/pubs/its/generalits/choosette.pdf >

Harder, C. *Serving Maps on the Internet: Geographic Information on the World Wide Web.* Redlands, CA: ESRI press, 1998.

Jankowski, P. and T. Nyerges. *Geographic Information Systems for Group Decision Making: Towards a Participatory, Geographic Information Science*. New York, NY: Taylor and Francis, 2001.

Miller, H.J. and S.L. Shaw. *Geographic Information Systems for Transportation-Principles and Applications.* New York, NY: Oxford University Press, 2001.

Minoli, D. and A. Schmidt. *Internet Architectures*. New York, NY: John Wiley & Sons, Inc, 1999.

Morgenthal, J.W. and L. F. Bill. *Enterprise Application Integration with XML and Java*. Upper Saddle River, NJ: Prentice Hall, 2001.

Peng, Z.R. and E. A. Biemborn. *Internet GIS and its Applications*. TR News. 1998 <http://216.239.33.100/search?q=cache:RCH1PbBJfpMC:www.uwm.edu/Dept/CUTS/peng/trnews.htm+Internet+GIS+and+its+Applications+TR+News+1998.&hl=en&ie=UTF-8>

Peng, Z.R.and R. Huang. "Design and development of interactive trip planning for web-based transit information systems." *Transportation Research Part C*, 8.1-6, 2000: 409-425.

Plewe, B. *GIS Online Information retrieval, Mapping and the Internet*. SantaFe, NM: Onward Press, 1997

Sadoski, D. *Client/Server Architectures*. GTE. 1997. <http://www.sei.cmu.edu/str/descriptions/clientserver.html>.

Schussel, G. *Client/Server Past, Present and Future*. 1995 <http://news.dci.com/geos/>.

Sessions, R. *Measuring Scalability.* Object watch newsletter. 2000. <http://216.239.33.100/search?q=cache:a9qPEHOKbQEC:www.objectwatch.com/issue_26.htm+Sessions,+R.+Measuring+Scalability+Objectwatch+newsletter+2000.&hl=en&ie=UTF-8>

Washington State Transportation Center. *Choosing the Route to Traveler Information Systems Deployment: Decision factors for Creating Public/Private Plans*. 1998. <www.itsa.org/.../b23f8115ff328af2852567f3005e7db0/$FILE/ATIS%20Business%20Models%20Sept%2099.ppt>.

Webopedia. 2001. <www.webopedia.com>.

What are Location Services? the GIS perspective. ESRI White Paper. 2000. <www.esri.com>.

APPENDIX

Appendix

Appendix Section	Description
A	Listing of all ASP (*.asp), JavaScript (*.js) and VB code (*.bas,*.vbp) used in the research. Javascript code added by the research in addition to the standard ArcIMS HTML viewer scripts are highlighted in blue
B	Metadata for PDOT traffic CD

A (scripts in sequence)

Script	Purpose
1. researchacetate.asp	Build XML files layers from Access tables for viewing existing polygons.
2. researchpp.asp	Insert new potential preferred paths.
3. researchstart.asp	Display the attribute table on ArcIMS upon every user login
4. researchupdate.asp	Send the ID from client to the server for overlay in the case of AOI or shortest/preferred path in the case of potential paths.
5. aimscommon.js	ArcIMS script for initializing map service. Modified for research.
6. aimscustom.js	ArcIMS script for adding customized acetate layers. Modified to add AOI and network link layers for display.
7. moresearch.js	Launch researchstart.asp from ArcIMS when user logs in.
8. aimsclick.js	ArcIMS script for handling clicks on tools. Launch the ASP script researchpp.asp to store preferred paths. Record ID for preferred paths.
9. toolbar.htm	Add the tools for storing AOIs and potential paths.
10. mointersect.exe	Performs overlays to extract network traffic volumes within AOIs (compiled into exe file)
11. linepp.exe	Fetch traffc volumes and geometry of network links for a preferred path (compiled into exe).
12. shortestpath.exe	Compute shortest path (compiled into exe).

A

Listing of research scripts

researchacetate.asp

This ASP script sets up the display of existing AOI shapes once the user logs in. An XML file is created from a query to the master table ‘tuserpoly.

```
<%@ LANGUAGE=JAVASCRIPT%>
<!-- #include file="adojavas.inc" -->

<HTML>
<HEAD>
<meta http-equiv="expires" content="0">
</HEAD>
<BODY>


<%

//Script :
//1) query the tuserpoly database
//2) write an XML as xml_<usernum>
//3) XML has rectangle or link ids
var usernum;
var type;


usernum = Request.Querystring("usernum");

selectstring = "SELECT  *  FROM  tuserpoly WHERE usernum = " + usernum;
//selectstring = "SELECT  *  FROM  A37 WHERE usernum = '" + usernum + "'";
conn = Server.CreateObject("ADODB.Connection");
conn.open("motuserpoly","","");
inrecordset = Server.Createobject("ADODB.recordset");
Response.write(selectstring);
inrecordset.open(selectstring,conn);
```

```
//Open a textfile and write to it:
var textfilename = "D:\\ArcIMS\\Website\\moresearchsite\\user" + usernum + ".xml";

var textfileobj = Server.CreateObject("Scripting.FileSystemObject");
var textfile = textfileobj.CreateTextFile(textfilename);


//Loop thru the fields and print rows on the response:

var numfields = inrecordset.Fields.Count;
var xmlheader = '<?xml version=' + '"1.0"' + ' encoding=' + '"ISO8859-1"' + ' ?>';
textfile.WriteLine(xmlheader);

textfile.WriteLine("<USERSHAPE>");

//Response.write("</tr>");
while(!inrecordset.EOF){
      //Response.write("<tr>");
          var aoitype = inrecordset.fields("aoitype");
          var theminx = inrecordset.fields("Minx");
          var themaxx = inrecordset.fields("Maxx");
          var theminy = inrecordset.fields("Miny");
          var themaxy = inrecordset.fields("Maxy");
       var idval = inrecordset.fields("uniqueid");


                    textfile.WriteLine("    <obj>");
                    if (aoitype==1){
                              var theheader = "rect";
                    }else if (aoitype==0)
                    {
                              var theheader = "pp"
                    }else
                    {
                              var theheader = "shorty"
                    }
                    var linestring = "       <type>" + theheader +"</type>";
                    textfile.WriteLine(linestring);
                    var thestring = "        <minx>" + theminx + "</minx>"
                    textfile.WriteLine(thestring);
                    var thestring = "        <miny>" + theminy + "</miny>"
                    textfile.WriteLine(thestring);
                    var thestring = "        <maxx>" + themaxx + "</maxx>"
                    textfile.WriteLine(thestring);
```

```
                var thestring = "       <maxy>" + themaxy + "</maxy>"
                textfile.WriteLine(thestring);
                var thestring = "       <idval>" + idval + "</idval>"
                textfile.WriteLine(thestring);
                textfile.WriteLine("    </obj>");

        inrecordset.movenext();

}

textfile.WriteLine("</USERSHAPE>");
inrecordset.close;
conn.close;

%>
</BODY>
</HTML>
```

researchpp.asp
Insert new potential preferred paths

```
<%@ LANGUAGE=JAVASCRIPT%>
<!-- #include file="adojavas.inc" -->

<HTML>
<HEAD>
<meta http-equiv="expires" content="0">
</HEAD>
<BODY>


<%

//researchpp.asp
var idstring = "";
var thedesc = "";
// Fetch query queryparameters:
thedesc = Request.Querystring("pathdesc");
var usernum = Request.Querystring("usernum");
idstring = Request.Querystring("idstring");
Response.write(usernum);




conn = Server.CreateObject("ADODB.Connection");
conn.open("motuserpoly","","");
inrecordset = Server.Createobject("ADODB.recordset");
inrecordset.open("tuserpoly",conn,adOpenKeyset,adLockOptimistic,adCmdTable);

inrecordset.AddNew;
inrecordset.fields("usernum") = usernum;
inrecordset.fields("aoitype") = 0; // PP type
if (thedesc!=""){
        inrecordset.fields("Desc") = thedesc;
}
```

```
var maxrecordset = Server.Createobject("ADODB.recordset");
var maxsql = "SELECT  max(uniqueid) AS maxid FROM tuserpoly";
maxrecordset = conn.execute(maxsql);

var maxval = maxrecordset.fields("maxid");
inrecordset.fields("uniqueid") = maxval + 1;
var uniqueid = maxval + 1;
maxrecordset.close;
inrecordset.Update;

inrecordset.close;

conn = Server.CreateObject("ADODB.Connection");
conn.open("moconnection2","","");
inrecordset = Server.Createobject("ADODB.recordset");
inrecordset.open("pplookup",conn,adOpenKeyset,adLockOptimistic,adCmdTable);


idstring = idstring + ";

var startpos = 0;
var pos = 0;
var idarray = idstring.split("*");


for (i=0;i<idarray.length;i++){
   inrecordset.AddNew;
   var theid = idarray[i];
         inrecordset.fields("uniqueid").value = uniqueid;
         inrecordset.fields("usernum").value = usernum;
         inrecordset.fields("linkid").value = Number(theid);
         inrecordset.Update;
}

inrecordset.close;

%>
</BODY>
</HTML>
```

researchstart.asp
Display the attribute table on ArcIMS upon every user login

```
<%@ LANGUAGE=JAVASCRIPT%>
<!-- #include file="adojavas.inc" -->

<HTML>
<HEAD>
<meta http-equiv="expires" content="0">
</HEAD>
<BODY>


<%
//researchstart.asp
var usernum;
var type;


usernum = Request.Querystring("usernum");

selectstring = "SELECT  *  FROM  tuserpoly WHERE usernum = " + usernum;
//selectstring = "SELECT  *  FROM  A37 WHERE usernum = '" + usernum + "'";
conn = Server.CreateObject("ADODB.Connection");
conn.open("motuserpoly","","");
//conn.open("moconnection2","","");
inrecordset = Server.Createobject("ADODB.recordset");
Response.write(selectstring);
inrecordset.open(selectstring,conn);

//Loop thru the fields and print rows on the response:

var numfields = inrecordset.Fields.Count;

Response.write("<table border='1' cellspacing='0' cellpadding='2' nowrap
bgcolor='White'>");
Response.write("<tr>");
for (i=0; i<numfields;i++){
        Response.write("<th>");
        Response.write(inrecordset.Fields(i).name);
```

```
        Response.write("</th>");
}
Response.write("</tr>");
while(!inrecordset.EOF){
   Response.write("<tr>");
   for (i=0;i<numfields;i++){
        var field = inrecordset.fields(i).name;
        var thetype = inrecordset.fields("aoitype");
                if (field == "uniqueid") {
                        idvalue = inrecordset.fields(field);
                        Response.write("<td>");
        //var theanchorstring = '<a href="' +
"/mpcasp/agendaedit.asp?"+"uid="+idvalue+"&"+"usernum="+usernum +"">' + "Edit
feature" + "</a>";
                //var theanchorstring = '<a href="' +
"javascript:window.document.location=" + "'http://mpc-
web/mpcasp/agendedit.asp?"+"uid="+idvalue+"&"+"usernum="+usernum +"">' + "Edit
feature" + "</a>";
                var theanchorstring = '<a href="' +
"/website/moresearchsite/default.htm?"+"mode=update"+"&"+"uid="+idvalue+"&"+"use
rnum="+usernum + "&" + "type="+thetype+"" + " TARGET=" +"'_top'"+ '>' + idvalue +
"</a>";
                Response.write(theanchorstring);
                Response.write("</td>");
        } else
        {
                        var value = inrecordset.fields(field);
                Response.write("<td>");
                Response.write(value);
                Response.write(".");
                Response.write("</td>");
        }
        }
   inrecordset.movenext();
   Response.write("</tr>");
}
Response.write("</tr>");
Response.write("</table>");
conn.close;

%>
</BODY>
</HTML>
```

researchupdate.asp

```
<%@ LANGUAGE=JAVASCRIPT%>
<!-- #include file="adojavas.inc" -->
<%Response.AddHeader("Pragma","No-Cache");%>
<HTML>
<BODY>


<%


//researchupdate.asp
var usernum;
var linkindex;
var objectid;  // objectid for the aoi to overlay rectangle with
var inrecordset;
var conn;
var cmd;
var dllobj;

usernum = Request.Querystring("usernum");
uid = Request.Querystring("uid");
type = Request.Querystring("type");

var objShell = new ActiveXObject("WScript.Shell");

//Determine which exe to launch.  Use type for this:
var textstring = "D:\\ArcIMS\\Website\\moresearchsite\\Dev\\mointersect.exe " + uid +
"&" + usernum;

if (type=="1"){
        //var textstring = "D:\\ArcIMS\\Website\\moresearchsite\\Dev\\mointersect.exe "
+ uid + "&" + usernum;
} else if (type=="0"){
        //var textstring = "D:\\ArcIMS\\Website\\moresearchsite\\Dev\\linepp.exe " + uid
+ "&" +usernum;
} else if (type=="2"){
```

```
        //var textstring = "D:\\ArcIMS\\Website\\moresearchsite\\Dev\\shortestpath.exe "
+ uid + "&" +usernum;
}

Response.write(textstring);
var intStatus  = objShell.run(textstring, 0, true);
if(intStatus==0)        {
        Response.Write("sucess");
}

//Assume that the XML file is written out by the exe:


/*Query the Oracle tUserPoly table and fetch attributes*/

selectstring = "SELECT  *  FROM  tuserpoly WHERE usernum = " + usernum;
conn = Server.CreateObject("ADODB.Connection");
conn.open("motuserpoly","","");
inrecordset = Server.Createobject("ADODB.recordset");
Response.write(selectstring);
inrecordset.open(selectstring,conn);

//Loop thru the fields and print rows on the response:

var numfields = inrecordset.Fields.Count;
var field;
var value;
var theid;
var theanchorstring;
var idstring;
Response.write("<table border='1' cellspacing='0' cellpadding='2' nowrap
bgcolor='White'>");
Response.write("<tr>");
for (i=0; i<numfields;i++){
        Response.write("<th>");
        Response.write(inrecordset.Fields(i).name);
        Response.write("</th>");
}
Response.write("</tr>");
while(!inrecordset.EOF){
   Response.write("<tr>");
   for (i=0;i<numfields;i++){

        var field = inrecordset.fields(i).name;
        if (field == "id") {
```

```
                idvalue = inrecordset.fields(field);
                Response.write("<td>");
        //var theanchorstring = '<a href="' + "mpc-
web/mpcasp/agendaedit.asp?"+"uid="+idvalue+"&"+"usernum="+usernum +"">' + "Edit
feature" + "</a>";
                var theanchorstring = '<a href="' +
"/mpcasp/agendaedit.asp?"+"mode=update"+"uid="+idvalue+"&"+"usernum="+usernum
+"" + " TARGET=" +"'_top'"+ '>' + "Edit feature" + "</a>";
                Response.write(theanchorstring);
                Response.write("</td>");
        }else
        {
                   var value = inrecordset.fields(field);
                Response.write("<td>");
                Response.write(value);
                Response.write(".");
                Response.write("</td>");
        }
   }

   inrecordset.movenext();
   Response.write("</tr>");
}
Response.write("</tr>");
Response.write("</table>");
conn.close;
%>
</BODY>
</HTML>
```

Aimscommon.js

```
// aimsCommon.js
/*
*  JavaScript template file for ArcIMS HTML Viewer
*           dependent on aimsXML.js, ArcIMSparam.js, aimsMap.js
*/

aimsCommonPresent=true;


//var theReply = "It didn't work!";
var queryTool = 0;
var legendImage="";
var modeBlurb = modeList[0];

// delimiter to be used between coordinates in strings in ArcXML
request
var coordsDelimiter = " ";
// delimiter to be used between pairs of coordinates in strings in
ArcXML request
var pairsDelimiter = ";";

var chkUnits=false;
var legendTemp = false;
var ovIsVisible=false;

var showBuffer = false;

// character used by browser in decimals - either point or comma
var decimalChar = ((("theChar is" + (10/100)).indexOf("."))==-1) ? ","
: ".";
//alert("Decimal character: " + decimalChar);

/*
**********************************************************************
***************

Common functions

**********************************************************************
***************
*/



// when there is a mapservice to load, it proceeds from here
```

```
function startUp() {

      if (imsURL != "") {
            //alert(imsURL);
            iWidth = parseInt(document.theImage.width);
            iHeight = parseInt(document.theImage.height);
            if (imsURL!=imsOVURL) toggleOVVisible = false;
            getStartExtent();

      }
}

// get the starting extent
function getStartExtent() {
      if (parent.PostFrame.document.forms[0]!=null) {
            var theString = '<ARCXML
version="1.1">\n<REQUEST>\n<GET_SERVICE_INFO renderer="false"
extensions="false" fields="false" />\n';
            theString = theString + '</REQUEST>\n</ARCXML>';
            var theReply="";
            if (getLimitExtent) {
                  if (hasOVMap) {
                        sendToServer(imsOVURL,theString,3);
                  } else {
                        sendToServer(imsURL,theString,3);
                  }
            } else {
                  XMLMode=3;
                  if (hasOVMap) {
                        sendToServer(imsOVURL,theString,998);
                  } else {
                        processXML(msgList[1]);
                  }
            }
      } else {
            alert(msgList[2]);
      }
}

// process the start extent and set up layers
function processStartExtent(theReply) {
      //alert(theReply);
      checkForForbiddenTags(theReply);
      // check for separators in serviceinfo
      var endpos = 0;
      var startpos = 0;
      var pos = theReply.indexOf("<SEPARATORS");
      if (pos!=-1) {

            startpos = theReply.indexOf("ts=",pos);
            if (startpos!=-1) {
                  startpos += 4;
                  endpos = theReply.indexOf(dQuote,startpos);
                  pairsDelimiter = theReply.substring(startpos,endpos);
            }
```

```
            startpos = theReply.indexOf("cs=",pos);
            if (startpos!=-1) {
                  startpos += 4;
                  endpos = theReply.indexOf(dQuote,startpos);
                  coordsDelimiter =
theReply.substring(startpos,endpos);
            }

      //alert("pairsDelimiter="+pairsDelimiter+"\ncoordsDelimiter="+coo
rdsDelimiter);
            checkCoords();
      }

      if (getStartingExtent) {
            getXYs(theReply);
            startLeft=eLeft;
            startRight=eRight;
            startTop=eTop;
            startBottom=eBottom;
      } else {
            eLeft=startLeft;
            eRight=startRight;
            eTop=startTop;
            eBottom=startBottom;
            xDistance = Math.abs(eRight-eLeft);
            var sFactor = xDistance / iWidth
            mapScaleFactor = sFactor;

      }
      if (aimsLayersPresent) {
            getLayers(theReply);
            if (setLayerVisible.length>0) setupLayerVisible();
      }
      if (aimsQueryPresent) {
            if (useStoredQuery) checkStoredQueries(theReply);
      } else {
            useStoredQuery=false;
      }
       xDistance = Math.abs(eRight-eLeft);
       yDistance = Math.abs(eTop-eBottom);
       xHalf = xDistance/2;
       yHalf = yDistance/2;
       panX = xDistance * panFactor;
       panY = yDistance * panFactor;
       if (chkUnits) {
             if (MapUnits=="DEGREES") {
                  if ((eRight > 250) || (eTop > 150)) MapUnits="FEET";
                  // alert(MapUnits);
             }
             chkUnits=false;
       }
       mouseX = 0;
       mouseY = 0;
       pixelX = xDistance/iWidth;
       pixelY = yDistance/iHeight;
```

```
        mapX = eLeft;
        mayY = eTop;
        lastLeft = eLeft;
        lastRight = eRight;
        lastTop = eTop;
        lastBottom = eBottom;
        if (hasOVMap == false) {
              fullLeft = limitLeft;
              fullRight = limitRight;
              fullTop = limitTop;
              fullBottom = limitBottom;
              fullWidth = Math.abs(fullRight - fullLeft);
              fullHeight = Math.abs(fullTop - fullBottom);

        }
      if (aimsLayersPresent) {
            if ((hasTOC) && (showTOC)) {
                  parent.TOCFrame.document.location=appDir+"toc.htm";
            }
      }
      if (aimsGeocodePresent) {
            if (GCLayerCount==0) {
                  if ((useGeocode) || (useReverseGeocode)) {
                        useGeocode=false;
                        useReverseGeocode=false;
                  }
            }
      } else {
                  useGeocode=false;
                  useReverseGeocode=false;
      }
      if (parent.ToolFrame!=null) {
            //alert("Refreshing toolbar");
            parent.ToolFrame.document.location= appDir + "toolbar.htm";
      } else if (hasToolBarOnLayer) {
            // requires custom function getLayerListContent. . .
example in layerlist.js in Hyperlink sample
            var content = getLayerListContent();
            if (isNav) {
                  replaceLayerContent("theToolBar",content);
            } else {
                  content = swapStuff(content,"\\'",sQuote);
                  document.all.theToolBar.innerHTML = content;
            }

      }
      hideRetrieveData();
      if ((ovIsVisible) && (aimsDHTMLPresent)) {
            ovIsVisible = false;
            toggleOVMap();
      }
      if (enforceFullExtent) {
            writeBlankMapXML();
      } else {
```

```
            if ((aimsQueryPresent) && (highlightedOne!="") &&
(queryZoom)) {
                  alert("launching custom query 1");
                  setStartQuery();

            } else {
                  sendMapXML();
                  alert("launching custom query 2");
                  getCommandLineForASP(webParams); //MO:Launch the
commandline parsing for user shapes
                  //If update then Call a function to zoom into the
necessary acetate links, use linkids
            }
      }
}


// request a list of available Image MapServices
function startMap() {
      showRetrieveData();
      if (aimsGenericPresent) {
            // only if aimsGeneric.js is loaded - for generic sample
            getDefaultParams()
            var theText = "<GETCLIENTSERVICES/>";
            sendToServer(catURL,theText,5);

      } else {
            startUp();
      }
}

/*  ************************
*     Extent functions
*     ************************
*/

// get the Map Image width
function getMapWidth () {
      var mapFrameWidth = thePageWin.innerWidth;

      if (mapFrameWidth == null) {
            mapFrameWidth = thePageDoc.body.clientWidth;
      }
      return mapFrameWidth;
}

 //get the Map Image height
function getMapHeight () {
      var mapFrameHeight = thePageWin.innerHeight;

      if (mapFrameHeight == null) {
            mapFrameHeight = thePageDoc.body.clientHeight;
      }
      return mapFrameHeight;
}
```

```
function checkCurrentExtent() {
      var msg = msgList[3] + eLeft + msgList[4] + eBottom + msgList[5]
+ eRight + msgList[6] + eTop;
      var ratio1 = xDistance/fullWidth;
      msg += msgList[7] + ratio1;
      alert(msg);
}


/*  ************************
*     Mode display functions
*     ************************
*/
// write out ModeFrame page
function writeModeFrame(currentMode) {
      parent.ModeFrame.document.location= appDir + "ModeFrame.htm";
}

// write out Mode on dynamic layer
function writeModeLayers(currentMode) {
      var content = '<font face="' + modeLayerFont + '"color="' +
modeLayerShadowColor + '"size=' + modeLayerSize + '><b>' + currentMode
+ '</b></font>';
      replaceLayerContent("theMode1",content);
      content = '<font face="' + modeLayerFont + '"color="' +
modeLayerColor + '"size=' + modeLayerSize + '><b>' + currentMode +
'</b></font>';
      replaceLayerContent("theMode2",content);

}

/*  ***************************************************
*     Various String manipulation Functions
*     ***************************************************
*/

// swap out double quotes for single
function swapQuotes2(inText) {
      var doubleQuote = dQuote;
      var singleQuote = "'";
      var preTemp = "";
      var posTemp = "";
      var nextPos = 0;
      var ePos = inText.length;
      var pos=9;
      while (pos != -1) {
            pos = inText.indexOf(dQuote);
            if (pos!=-1) {
                  nextPos=pos+1;
                  preTemp = inText.substring(0,pos);
                  posTemp = inText.substring(nextPos,ePos);
                  inText = preTemp + sQuote + posTemp;
            }
      }
```

```
	return inText;
}

function swapQuotes(inText) {
	inText = inText.replace(/"/g, "'");
	return inText;
}

// convert hexidecimal rgb number to delimited decimal rgb
function convertHexToDec(hexColor) {
	var pos = hexColor.indexOf(",");
	var decString = hexColor;
	if (pos==-1) {
		pos = hexColor.indexOf("#");
		if (pos!=-1) {
			hexColor = hexColor.substring((pos + 1),(pos + 7));
		}
		//alert(hexColor);
		var redHex = hexColor.substring(0,2);
		var greenHex = hexColor.substring(2,4);
		var blueHex = hexColor.substring(4,6);
		decString = parseInt(redHex,16) + "," +
parseInt(greenHex,16) + "," + parseInt(blueHex,16);

	}
	//alert(decString);
	return decString;

}


// swap out one interior string with another
function swapStuff(oldString,oldStuff,newStuff) {
	var pos = 0;
	var rpos = 0;
	var epos = 0;
	var leftString = "";
	var rightString = "";
	pos = oldString.indexOf(oldStuff);
	while (pos!=-1) {
		epos = oldString.length;
		rpos = pos + oldStuff.length;
		leftString = oldString.substring(0,pos);
		rightString = oldString.substring(rpos,epos);
		oldString = leftString + newStuff + rightString;
		pos = oldString.indexOf(oldStuff);
	}
	leftString=null;
	rightString=null;
	return oldString;
}

/*  ***************************************************
*	Various utility Functions
*	***************************************************
```

```
*/

// disables error checking
function clearError() {
      return true;
}

// reset error checking to default
function resetError() {
      return false;
}


function reloadApp() {
      if (isNav) {
            document.location = "default.htm";
      }
}

// clear out leading spaces in field value list
function clearLeadingSpace(inText) {
      var pos=9;
      while (pos != -1) {
            pos = inText.indexOf('=" ');
            if (pos!=-1) {
                  var lastpos = inText.length;
                  var midend = pos + 2;
                  var midstart = pos + 3;
                  var leftSide = inText.substring(0,midend);
                  var rightSide = inText.substring(midstart,lastpos);
                  inText = leftSide + rightSide;
            }
      }
      return inText;
}

// replace < and > in string with [ and ] to allow display in html page
function untag(inputString) {
      var outString = inputString.replace(/</g,"[");
      outString = outString.replace(/>/g, "]");
      return outString;
}

// replace single quotes with double single quotes
//    set up interior single qoutes and apostrophes for queries
function fixSingleQuotes(inputString) {
      var outString = inputString.replace(/'/g, "''");
      return outString;
}


// parse out record data from XML stream
function parseRecordString(theReply, startpos) {
      var inData = "";
      var pos = theReply.indexOf("<FIELDS ",startpos);
```

```
	if (pos!=-1) {
		startpos = pos + 8;
		xmlEndPos = theReply.indexOf('" />',startpos);
		inData = theReply.substring(startpos,xmlEndPos);
	}
	return inData;
}

// get a list of field names from the returned record
function getFieldNames(recordString) {
	var theStuff = new String(recordString);
	var theList = theStuff.split('" ');
	var fName1 = new Array();
	for (var f=0;f<theList.length;f++) {
		var v = theList[f].split('="');
		fName1[f] = v[0];
	}
	return fName1;

}

// get a list field values from the returned record
function getFieldValues(recordString) {
	var theStuff = new String(recordString);
	var theList = theStuff.split('" ');
	var fValue1 = new Array();
	for (var f=0;f<theList.length;f++) {
		var v = theList[f].split('="');
		if ((v[1]=="") || (v[1]==null)) v[1] = " ";
		if (v[0]==LayerShapeField[ActiveLayerIndex]) v[1]="[" +
ActiveLayerType + "]";
		fValue1[f] = v[1];
	}
	return fValue1;

}

// just get the field value from the lists of fieldnames and
fieldvalues
function getIdValue(fieldNameArray, fieldValueArray) {
	var theValue = 0;
	for (var f=0;f<fieldNameArray.length;f++) {
		if (fieldNameArray[f]==LayerIDField[ActiveLayerIndex]) {
			theValue = fieldValueArray[f];
		}
	}
	return theValue;

}

// just get the interior string from the theReply between preString and
postString
//		starting from startpos
function justGetValue(theReply,preString,postString,startpos) {
	var theValue = "";
```

```
	var pos = theReply.indexOf(preString,startpos);
	if (pos!=-1) {
		pos = pos + preString.length;
		var endpos = theReply.indexOf(postString,(pos));
		if (endpos!=-1) {
			theValue = theReply.substring(pos,endpos);
			xmlEndPos = endpos;
		}
	}
	return theValue;
}

// get one field value from theReply starting from startpos
function justGetFieldValue(theReply,theField,startpos) {
	var preString = theField + '="';
	var returnString = justGetValue(theReply, preString, dQuote,
startpos);
	return returnString;
}

// get the number of features returned in xml response
function justGetFeatureCount(theReply) {
	var theCount = 0;
	var pos = theReply.indexOf("<FEATURECOUNT");
	if (pos!=-1) {
		var theValue = justGetValue(theReply,'count="',dQuote,pos);
		//alert(theValue);
		theCount = parseInt(theValue);
	}
	return theCount;
}

// get all the field values and return a list
function getAllFieldValues(theReply,theField,recCount) {
	var vList = new Array();
	xmlEndPos = 0;
	for (var i=0;i<recCount;i++) {
		vList[i] =
parseFloat(justGetFieldValue(theReply,theField,xmlEndPos));
	}
	return vList;
}

// reset order to numeric
function numberorder(a,b) { return a - b; }

// replace common HTML entitys with the characters they represent
function parseEntity(oldString) {
	//alert(oldString);
	oldString = oldString.replace(/&apos;/g, "'");
	oldString = oldString.replace(/&gt;/g, ">");
	oldString = oldString.replace(/&lt;/g, "<");
	oldString = oldString.replace(/&quot;/g, '"');
	oldString = oldString.replace(/&amp;/g, "&");
	//alert(oldString);
```

```
	/*
	oldString = swapStuff(oldString,"&apos;","'");
	oldString = swapStuff(oldString,"&divide;","/");
	oldString = swapStuff(oldString,"&ge;",">=");
	oldString = swapStuff(oldString,"&gt;",">");
	oldString = swapStuff(oldString,"&le;","<=");
	oldString = swapStuff(oldString,"&lt;","<");
	oldString = swapStuff(oldString,"&ne;","<>");
	oldString = swapStuff(oldString,"&quot;",'"');
	oldString = swapStuff(oldString,"&amp;","&");
	*/

	return oldString;
}

function hideQuotes(oldString) {

}

// replace the five problem characters for the server's XML parser
function makeXMLsafe(oldString) {
	//alert(oldString);
	oldString = oldString.replace(/&/g, "&amp;");
	oldString = oldString.replace(/'/g, "&apos;");
	oldString = oldString.replace(/>/g, "&gt;");
	oldString = oldString.replace(/</g, "&lt;");
	oldString = oldString.replace(/"/g, "&quot;");
	/*
	oldString = swapStuff(oldString,"'","&apos;");
	oldString = swapStuff(oldString,">","&gt;");
	oldString = swapStuff(oldString,"<","&lt;");
	oldString = swapStuff(oldString,'"',"&quot;");
	*/
	//alert(oldString);
	return oldString;
}

// replace +  in string with space to allow parsing of unescaped xml
response
function replacePlus(inText) {
     var re = /\+/g;
      inText = inText.replace(re," ");
     return inText;
}

// replaces comas or spaces in these variables to coordsDelimiter value
      // the variables checked are used for image coords and should be
integer
function checkCoords() {
      var re = /,|\s|\|/g;
      NorthArrowCoords = NorthArrowCoords.replace(re,coordsDelimiter);
      CopyrightCoords = CopyrightCoords.replace(re,coordsDelimiter);
}
```

```
// get the substring between beforeString and afterString, starting at
startpos
//          must be found before limitpos (0 for no limit)
//          caseSensitive = true or false
function
getInsideString(inString,beforeString,afterString,startpos,limitpos,cas
eSensitive) {
      var returnString = "";
      var ucInString = inString;
      var ucBefore = beforeString;
      var ucAfter = afterString;
      if (limitpos==0) limitpos = inString.length;
      if (!caseSensitive) {
            ucInString = inString.toUpperCase();
            ucBefore = beforeString.toUpperCase();;
            ucAfter = afterString.toUpperCase();;
      }
      pos = ucInString.indexOf(ucBefore,startpos);
      //alert(startpos);
      if ((pos != -1) && (pos<limitpos)) {
            pos = pos + ucBefore.length;
            var endpos = ucInString.indexOf(ucAfter,pos);
            returnString = inString.substring(pos,endpos);
      }

      return returnString;
}

// formats date string to "{ts 'yyyy-mm-dd hh:mm:ss'}"
function formatDate(theDateString) {
      //if (theDateString.toUpperCase().indexOf("UTC")==-1)
theDateString + " UTC";
      var v = new Date(theDateString);
      //alert(v);

      var dateString = "";
      if (!isNaN(v.valueOf())) {
            var y = v.getFullYear();
            var mo = v.getMonth() + 1;
            if (mo<10) mo = "0" + mo;
            var d = v.getDate();
            if (d<10) d = "0" + d;
            var h = v.getHours();
            if (h<10) h = "0" + h;
            var mi = v.getMinutes();
            if (mi<10) mi = "0" + mi;
            var s = v.getSeconds();
            if (s<10) s = "0" + s;
            dateString = "{ts \"" + y + "-" + mo + "-" + d;
            if (theDateString.indexOf(":")!=-1) {
                  if (v.getHours() + v.getMinutes() + v.getSeconds()>0)
                        dateString += " " + h + ":" + mi + ":" + s;
            }
            dateString += "\"}";
      }
```

```
      // note: this sets up the formatted date with double quotes,
      //          which will be changed to single by the swapQuotes
function called in sendQueryString()
      return dateString;

}

// format decimal numerics from comma to point
//    SQL format requires English notation
function convertDecimal(theNumString) {
      var replacer = "."
      var re = /,/g;
      var newString = theNumString.replace(re,replacer);
      return newString;
}

// test for forbidden tags for this service
function checkForForbiddenTags(theReply) {
      var startpos = theReply.indexOf("CAPABILITIES forbidden=");
      if (startpos!=-1) {
            startpos = startpos + 24;
            endpos = theReply.indexOf(dQuote,startpos);
            var forbiddenTags = theReply.substring(startpos,endpos);
            //alert(forbiddenTags);
            if (forbiddenTags.indexOf("GET_IMAGE")!=-1) {
                  // No image requests!!!! Abort viewer
                  parent.document.location = "Abort.htm";
            }
            if (forbiddenTags.indexOf("GET_FEATURES")!=-1) {
                  // No id/query requests!!!! Kill buttons
                  aimsSelectPresent=false;
                  aimsQueryPresent=false;
                  aimsBufferPresent=false;
                  aimsIdentifyPresent=false;
                  canQuery=false;
                  useIdentify=false;
                  useSelect=false;
                  useQuery=false;
                  useFind=false;
                  useBuffer=false;
                  useStoredQuery=false;
                  useHyperLink=false;
                  useHyperLinkAny=false;
                  useIdentifyAll=false;
                  useBufferShape=false;

            }
            if (forbiddenTags.indexOf("GET_GEOCODE")!=-1) {
                  // No geocode requests!!!! Kill buttons
                  aimsGeocodePresent=false;
                  useGeocode=false;
                  useReverseGeocode=false;
            }
            if (forbiddenTags.indexOf("GET_EXTRACT")!=-1) {
                  // No geocode requests!!!! Kill buttons
```

```
                    useExtract=false;
                }


        }
    }
```

Aimscustom.js

```
// aimsCustom.js
/*
*  JavaScript template file for ArcIMS HTML Viewer
*           dependent on aimsXML.js, ArcIMSparam.js, aimsCommon.js,
aimsMap.js,
*           aimsLayers.js, aimsDHTML.js
*           aimsClick.js, aimsNavigation.js,
*/

// global variables
      aimsCustomPresent=true;
      // change these to send XML response to custom function.
      // use numbers >= 1000 and match in useCustomFunction()
      // defaults are defined in aimsXML.js and use standard functions

      // xml response mode for selection
      selectXMLMode = 6;
      // xml response mode for identify
      identifyXMLMode = 7;
      // xml response mode for query
      queryXMLMode = 8;
      // xml response mode for find
      findXMLMode = 14;
      // xml response mode hyperlink
      hyperlinkXMLMode = 15;



// custom function for handling clicks
//          flow redirected here when
//          toolMode set to >=1000
function customMapTool(e) {
                        if (toolMode == 1001) {
                                    // insert code here
                                    return false;

                        }
                        if (toolMode == 1002) {
                                    // insert code here

                        }

}

// send  XML response to custom function
```

```
//          flow  redirected here when
//          XMLMode >=1000
function useCustomFunction(theReply) {
      if (XMLMode==1001) {
            // insert code here
      } else if (XMLMode==1002) {
            // insert code here
      } else {
            alert(msgList[55] + XMLMode + msgList[56]);
      }
      hideLayer("LoadData");
}

// add custom stuff to Map XML request. . . between selection and
geocode
function addCustomToMap1(){
      var customString = "";

      //MO: modified here to read the XML file produced by
researchstart:
      alert("custom to map");
      //Get usernum even though getcommandlineline extracts it again!
      var cmdString2 = webParams.toUpperCase();
      var pos = cmdString2.indexOf("USERNUM=");
      if (pos!=-1) {
            startpos = pos + 8;
            endpos = webParams.indexOf("&",startpos);
            if (endpos==-1) endpos = webParams.length;
            usernum = webParams.substring(startpos,endpos);
            alert("usernum at customtomap");
            alert(usernum);
      }
      //Query the database first and get the rectangle bounds first:
      var aspurl =
"http://puertorico.geog.utk.edu/researchasp/researchacetate.asp?" +
"Usernum=" + usernum;
      var win1 = parent.MoFrame;
      win1.document.open();
      win1.document.location = aspurl;

    for (i=0;i<2000;i++)

      //Read the XML file just produced:
    var xmlDoc = new ActiveXObject("Microsoft.XMLDOM");
    xmlDoc.async = "false"
    var loadstring = "user" + usernum + ".xml";
    xmlDoc.load(loadstring);
    nodes = xmlDoc.documentElement.childNodes
    for (i=0;i<nodes.length;i++){
      thetype = nodes.item(i).childNodes.item(0).text;
      //alert(thetype);
        //Mo: check if update!
          //If update display aoi being updated only, use uid-> AOI
identifier:
```

```
      if(thetype=='rect'){
            var minx = nodes.item(i).childNodes.item(1).text;
            var miny = nodes.item(i).childNodes.item(2).text;
            var maxx = nodes.item(i).childNodes.item(3).text;
            var maxy = nodes.item(i).childNodes.item(4).text;
            var theid = nodes.item(i).childNodes.item(5).text;

                  var intminx = parseFloat(minx);
                  var intmaxx = parseFloat(maxx);
                  var intminy = parseFloat(miny);
                  var intmaxy = parseFloat(maxy);

                  var numtextx = (intminx + intmaxx)/2;
                  var numtexty = (intminy + intmaxy)/2;
                  var textx = numtextx + '';
                  var texty = numtexty + '';
                  //alert(textx);
                  //alert(texty);

            //Write this layer as acetate:

            customString += '<LAYER type="ACETATE" name="usershape">\n';
            customString += '<OBJECT units="DATABASE">\n<LINE coords="'
+ minx + " " + miny + ';' + minx + " " + maxy + ';';
            customString += maxx + " " + maxy + ';';
            customString += maxx + " " + miny + ';';
            customString += minx + " " + miny + '">\n';
            customString += '<SIMPLELINESYMBOL type="SOLID" color="' +
'#ff0000' + '" width="3"/>\n</LINE>\n</OBJECT>\n';
            customString += '<OBJECT units="DATABASE">\n';
            customString += '<TEXT coords="' + textx + " " + texty + '"
label="' + theid + '">\n<TEXTMARKERSYMBOL fontstyle="regular"
fontsize="30" font="Times New Roman" /></TEXT>\n';
            customString += '</OBJECT>\n';
            customString += '</LAYER>\n';
            //alert(customString);
        }

      }

      getCommandLineForASP(webParams);
      return customString;
}

// add custom stuff to Map XML request. . . between clickpoints and
copyright
function addCustomToMap2(){
      var customString = "";
    var cmdString2 = webParams.toUpperCase();
      var pos = cmdString2.indexOf("MODE=");
      if (pos!=-1) {
            startpos = pos + 5;
            endpos = webParams.indexOf("&",startpos);
            if (endpos==-1) endpos = webParams.length;
            momode = webParams.substring(startpos,endpos);
```

```
            alert(momode);

      }
      if (momode!="update") return;  // no lines to draw if not an
update
      var pos = cmdString2.indexOf("UID=");
      if (pos!=-1) {
            startpos = pos + 4;
            endpos = webParams.indexOf("&",startpos);
            if (endpos==-1) endpos = webParams.length;
            momodeid = webParams.substring(startpos,endpos);
            alert(momodeid);

      }
      //Type is used for AOI overlay or potential path:
      var pos = cmdString2.indexOf("TYPE=");
      if (pos!=-1) {
            startpos = pos + 5;
            endpos = webParams.indexOf("&",startpos);
            if (endpos==-1) endpos = webParams.length;
            momodetype = webParams.substring(startpos,endpos);
            alert(momodetype);

      }

      //Query the database first and get the rectangle bounds first:
      var aspurl =
"http://puertorico.geog.utk.edu/researchasp/researchupdate.asp?" +
"Usernum=" + usernum + "type=" + momodetype;
      var win1 = parent.MoFrame;
      win1.document.open();
      win1.document.location = aspurl;

      for (i=0;i<2000;i++)

      //Read the XML file just produced:
      var xmlDoc = new ActiveXObject("Microsoft.XMLDOM");
      xmlDoc.async = "false"
      var loadstring = "update-" + usernum + momodeid + ".xml";
      //var loadstring = "update-11.xml";
      xmlDoc.load(loadstring);
      nodes = xmlDoc.documentElement.childNodes

      //Initialize the color ramps:
      //Three ramps   light-> 4-7617:3
      //Mod-> 7618-15235:6
      //Heavy-> 15235 -22841:9


      for (i=0;i<nodes.length;i++){

         customString += '<LAYER type="ACETATE" name="usershape">\n';
           customString += '<OBJECT units="DATABASE">\n<LINE coords="'
               //alert(nodes.item(i).childNodes.length);
         for (j=0;j<nodes.item(i).childNodes.length-1;j++){
```

```
                  //alert(j);
               var x = nodes.item(i).childNodes.item(j).text;
               customString += x + " ";
               if (j==(nodes.item(i).childNodes.length -2)) {
                     var aadt = nodes.item(i).childNodes.item(j+2).text;
                     //alert(aadt);
                     //Get the last y:

                 } else {
                     var y = nodes.item(i).childNodes.item(j+1).text;
                     customString += y + ';';
                     //Check for last item and read as AADT:
                 }

                 j++;
            }
            //Classfy aadt:
            if (aadt>=4 && aadt<=7617) {
                     var thewitdh = 3;
            } else if (aadt>7617 && aadt<=1523) {
                     var thewitdh = 6;
            } else  {
                 var thewitdh = 9;
            }

            customString += '">\n';
            customString += '<SIMPLELINESYMBOL type="SOLID" color="' +
'#6D67FF' + '" width="' + thewitdh + '"/>\n</LINE>\n</OBJECT>\n';
              customString += '</LAYER>\n';

              if (i==1) {
               alert(customString);
              }

          }



 //Mo:Zoom into this AOI using etops:

 return customString;
}

// add custom stuff to Map XML request. . . under modeOnMap
function addCustomToMap3(){
      var customString = "";
            /*
            customString += '<LAYER type="ACETATE" name="theMode">\n';
            customString += '<OBJECT units="PIXEL">\n<TEXT coord="5,' +
(iHeight-10) + '" label="This is a test">\n';
            customString += '<TEXTMARKERSYMBOL fontstyle="BOLD"
fontsize="12" font="ARIAL" fontcolor="' + modeMapColor + '" ';
            customString += 'threed="TRUE" glowing="' + modeMapGlow +
'" />\n</TEXT>\n</OBJECT>';
            customString += '\n</LAYER>\n';
```

```
            alert(customString);
            */
      return customString;
}

// add custom stuff to Map XML request. . . on top of everything
function addCustomToMap4(){
      var customString = "";

      return customString;
}

// extract layers to download
function extractIt() {
      hideLayer("measureBox");
      alert(msgList[51]);
}
```

Moresearch.js

```
// Moresearch.js
/************************************************
*
* Javascript to parse command line:
* Launch window to ASP
* Query oracle database and select preselected user polygons
************************************************/


//List of global variables that are modified:
//starttop,startleft,startbottom,startright all set to 0 along with the
limits

//List of entry points:
//12/12/01 amiscommon.js::processStartExtent()


function getCommandLineForASP(cmdString) {

      alert("moagenda.js: starting");
      //var usernum = "";     Note: usernum is defined in AIMSparams.js

      //Process command line:

      var cmdString2 = cmdString.toUpperCase();
      var pos = cmdString2.indexOf("QUERYZOOM=");
      var startpos = 0;
      var endpos = 0;

      if (pos!=-1) {
            startpos = pos + 10;

            endpos = cmdString.indexOf("&",startpos);
            if (endpos==-1) endpos = cmdString.length;
            querystatus = cmdString.substring(startpos,endpos);
            alert(querystatus);
      }
      var pos = cmdString2.indexOf("USERNUM=");
      if (pos!=-1) {
            startpos = pos + 8;
            endpos = cmdString.indexOf("&",startpos);
            if (endpos==-1) endpos = cmdString.length;
```

```
            usernum = cmdString.substring(startpos,endpos);
            alert(usernum);
      }
      if (usernum != "") {
            // Mo: dont need this now
        /*
            //var userquerystring = 'TUSERPOLY.USERNUM = ' + usernum;
            alert("processing querry");
            var userquerystring = 'ADAMSNET.OBJECTID = 1';
            alert(userquerystring);
            modeBlurb = "Query";
            QueryZoom = true;
            setActiveLayer(2);
            sendQueryString(userquerystring);
            alert("sent query");
            */
      }

      //Launch query with username in ASP:
      //Write data to MoFrame.htm
      var aspurl =
"http://puertorico.geog.utk.edu/researchasp/researchstart.asp?" +
"Usernum=" + usernum;
      alert(aspurl);
      //var aspwin =
window.open(aspurl,"aspwindow","width=575,height=120,scrollbars=yes,res
izable=yes");
      //MO: change to resultframe in mainpage
      var win1 = parent.MoFrame;
      win1.document.open();
      win1.document.location = aspurl;
      //Query the layer again to show shopping cart:
      /*
       Call aimsquery::sendquerystring
       modify aimsidentify::display attributedata() to show shopping
cart urls
      */

}
```

Aimsclick.js

```
// aimsClick.js
/*
*  JavaScript template file for ArcIMS HTML Viewer
*           dependent on aimsXML.js, ArcIMSparam.js, aimsCommon.js,
aimsMap.js,
*           aimsLayers.js, aimsDHTML.js
*           aimsNavigation.js
*/

aimsClickPresent=true;

var onOVArea = false;

// Global vars to save mouse position
var mouseX=0;
var mouseY=0;
var x1=0;
var y1=0;
var x2=0;
var y2=0;
var zleft=0;
var zright=0;
var ztop=0;
var zbottom=0;

var totalMeasure=0;
var currentMeasure=0;
var lastTotMeasure=0;

// variables for interactive clicks
var clickCount = 0;
var   clickPointX = new Array();
var clickPointY = new Array();
var clickMeasure = new Array();
      // type - 1=Measure; 2=SelectLine ; 3=SelectPolygon
var clickType = 1;

var shapeSelectBuffer = false;

var panning=false;
var zooming=false;
var selectBox=false;
var blankImage = "images/map.gif";
```

```
var leftButton =1;
var rightButton = 2;
if (isNav) {
      leftButton = 1;
      rightButton = 3;
}


/*  ***************************************************
*     Point click functions
*     used by Measure and Select by Line/Polygon
*     ***************************************************
*/

// put a point at click and add to clickCount
function clickAddPoint() {
      var theX = mouseX;
      var theY = mouseY;
      getMapXY(theX,theY);
      clickPointX[clickCount]=mapX;
      clickPointY[clickCount]=mapY;
      clickCount += 1;
      selectCount=0;
      totalMeasure = totalMeasure + currentMeasure;
            //var u = Math.pow(10,numDecimals);
            //if (totalMeasure!=0) totalMeasure =
parseInt(totalMeasure*u+0.5)/u;

      clickMeasure[clickCount]=totalMeasure;
      legendTemp=legendVisible;
      legendVisible=false;
      var theString = writeXML();
      var theNum = 99;
      sendToServer(imsURL,theString,theNum);

}

// zero out all clicks in clickCount
function resetClick() {
      var c1 = clickCount;
      clickCount=0;
      clickPointX.length=1;
      clickPointY.length=1;
      currentMeasure=0;
      totalMeasure=0;
      lastTotMeasure=0;
      clickMeasure.length=1;
      selectCount=0;

      legendTemp=legendVisible;
      legendVisible=false;
      var theString = writeXML();
      var theNum = 99;
            //showRetrieveMap();
      sendToServer(imsURL,theString,theNum);
```

```
	if (toolMode==20) updateMeasureBox();

}

// remove last click from clickCount
function deleteClick() {
	var c1 = clickCount;
	clickCount=clickCount-1;
	selectCount=0;
	if (clickCount<0) clickCount=0;
	if (clickCount>0) {
		totalMeasure = clickMeasure[clickCount]
		clickPointX.length=clickCount;
		clickPointY.length=clickCount;
		clickMeasure.length=clickCount;

	} else {
		totalMeasure=0;
		clickMeasure[0]=0;
	}
	currentMeasure=0;
	if (c1>0) {
		legendTemp=legendVisible;
		legendVisible=false;
		var theString = writeXML();
		var theNum = 99;
		sendToServer(imsURL,theString,theNum);
	}

}


//keep track of currently selected tool, and display it to user
// set the imsMap cursor tool
function clickFunction (toolName) {
	if (hasLayer("measureBox"))
		hideLayer("measureBox");
	switch(toolName) {
	// Zooming functions
	case "zoomin":
		// zoom in mode
		toolMode = 1;
		panning=false;
		selectBox=false;
		if (isIE)   {
			document.all.theTop.style.cursor = "crosshair";
			theCursor = document.all.theTop.style.cursor;
		}
		modeBlurb = modeList[0];
		//if (useTextFrame) parent.TextFrame.document.location=
appDir + "text.htm";
		break
	case "zoomout":
		// zoom out mode
```

```
            toolMode = 2;
            panning=false;
            selectBox=false;
            if (isIE)   {
                  document.all.theTop.style.cursor = "crosshair";
                  theCursor = document.all.theTop.style.cursor;
            }
            modeBlurb = modeList[1];
            //if (useTextFrame) parent.TextFrame.document.location=
appDir + "text.htm";
            break
      case "zoomlast":
            zoomBack();
            panning=false;
            zooming=false;
            selectBox=false;
            break
      case "zoomactive":
            //alert(LayerExtent[ActiveLayerIndex]);
            var q = LayerExtent[ActiveLayerIndex].split("|");
            panning=false;
            zooming=false;
            selectBox=false;

      //zoomToEnvelope(parseFloat(q[0]),parseFloat(q[1]),parseFloat(q[2
]),parseFloat(q[3]));

            var l = parseFloat(setDecimalString(q[0]));
            var b = parseFloat(setDecimalString(q[1]));
            var r = parseFloat(setDecimalString(q[2]));
            var t = parseFloat(setDecimalString(q[3]));
            var w = r-l;
            var h = t-b;
            // add a bit of a margin around the layer
            var wm = w * (5/100);
            var hm = h * (5/100);
            l = l - wm;
            r = r + wm;
            b = b - hm;
            t = t + hm;
            zoomToEnvelope(l,b,r,t);
            break
      case "fullextent":
            fullExtent();
            break

      // Pan functions
      case "pan":
            // pan mode
            toolMode = 3;

            zooming=false;
            selectBox=false;
            if (isIE)   {
                  document.all.theTop.style.cursor = "move";
```

```
                  theCursor = document.all.theTop.style.cursor;
            }
            modeBlurb = modeList[2];
            //if (useTextFrame) parent.TextFrame.document.location=
appDir + "text.htm";
            break

      // Identify-Hyperlink functions
      case "identify":
            // identify mode - layer attributes - requires
aimsIdentify.js
            panning=false;
            zooming=false;
            selectBox=false;
            shapeSelectBuffer = false;
            if (canQuery) {
                  toolMode = 4;

                  if (isIE)   {
                        document.all.theTop.style.cursor = "crosshair";
                        theCursor = document.all.theTop.style.cursor;
                  }
                  modeBlurb = modeList[3];
            } else {
                  alert(msgList[46]);
            }
            //alert("Function Not Implemented");



            showGeocode=false;
            if (useTextFrame) parent.TextFrame.document.location=
appDir + "text.htm";
            hideLayer("measureBox");
            break
      case "researchpp":
            // identify mode - layer attributes - requires
aimsIdentify.js
            ppset = true; // variable set so that select is handled
different, needs to be reset in aimsselect.js
            //alert(ppset);
            panning=false;
            zooming=false;
            selectBox=false;
            shapeSelectBuffer = false;
            if (canQuery) {
                alert("setting tool mode = 10");
                  //toolMode = 4;
                  toolMode = 21;

                  if (isIE)   {
                        document.all.theTop.style.cursor = "crosshair";
                        theCursor = document.all.theTop.style.cursor;
                  }
                  modeBlurb = modeList[3];
```

```
            } else {
                  alert(msgList[46]);
            }
            //alert("Function Not Implemented");



            showGeocode=false;
            if (useTextFrame) parent.TextFrame.document.location=
appDir + "text.htm";
            hideLayer("measureBox");

            //Mo: Enter the description for the path:
            //var descwin = parent.TextFrame;
            //descwin.document.open();
            //descwin.document.writeln('<html><meta http-
equiv="Content-Type" content="text/html; charset=' + charSet +
'"><HEAD>');
            //descwin.document.writeln('        </head>');
            //descwin.document.writeln('        <body>');
            //descwin.document.writeln('        Please enter a
descriptor for the path ');
            //descwin.document.writeln('<form name="thepath">');
            //descwin.document.writeln('<input name="pathval" size="25"
maxlength="1000" >');
            //descwin.document.writeln('      </form></body></html>');
            //descwin.close;



            break

   case "researchppstop":
            // Launch the asp and clear ppset and pparray
            //sendShapeSelect(1);

            var textwin = parent.PathFrame;
            var pathdesc = textwin.document.thepath.pathval.value;
            alert(pathdesc);
            alert(usernum);
            var aspurl =
"http://puertorico.geog.utk.edu/researchasp/researchpp.asp?" +
"usernum=" + usernum + "&" + "idstring=" + pparray + "&" + "pathdesc="
+ pathdesc;
            alert(aspurl);
            alert(pparray);
            var win1 = parent.MoFrame;
          win1.document.open();
          win1.document.location = aspurl;
            alert(pparray);
            ppset = false; // variable set so that select is handled
different, needs to be reset in aimsselect.js
            pparray = "";

            break
```

```
      case "identifyall":
            // identify drill mode
            panning=false;
            zooming=false;
            selectBox=false;
            shapeSelectBuffer = false;
            toolMode = 5;
            if (canQuery) {
                  if (isIE)   {
                        document.all.theTop.style.cursor = "crosshair";
                        theCursor = document.all.theTop.style.cursor;
                  }
                  //modeBlurb = modeList[19]; // identify all
                  modeBlurb = modeList[20]; // identify visible
features
                  //modeBlurb = modeList[3]; // identify
            } else {
                  alert(msgList[46]);
            }
            //alert("Function Not Implemented");
            showGeocode=false;
            drawSelectBoundary=false;
            if (useTextFrame) parent.TextFrame.document.location=
appDir + "text.htm";
            hideLayer("measureBox");
            break
      case "hyperlink":
            // hyperlink mode - requires aimsIdentify.js
            var isOk = false;
            var j=-1;
            panning=false;
            zooming=false;
            selectBox=false;
            shapeSelectBuffer = false;
            toolMode = 15;
            modeBlurb = modeList[9];
            showGeocode=false;
            if (useTextFrame) parent.TextFrame.document.location=
appDir + "text.htm";
            hideLayer("measureBox");
            var isOk = checkHyperLinkLayer(ActiveLayerIndex)
            if (isOk) {
                  if (canQuery) {


                        if (isIE)   {
                              document.all.theTop.style.cursor =
"crosshair";
                              theCursor =
document.all.theTop.style.cursor;
                        }

                  } else {
                        alert(msgList[46]);
```

```
                  }
                  //alert("Function Not Implemented");
            } else {
                  currentHyperLinkLayer="";
                  currentHyperLinkField="";
                  alert(msgList[47]);

            }
            break

      case "hyperlinkany":
            // hyperlink mode - requires aimsIdentify.js
            var j=-1;
            panning=false;
            zooming=false;
            selectBox=false;
            shapeSelectBuffer = false;
            toolMode = 30;
            modeBlurb = modeList[9];
            showGeocode=false;
            if (useTextFrame) parent.TextFrame.document.location=
appDir + "text.htm";
            hideLayer("measureBox");
                  if (canQuery) {


                        if (isIE)   {
                              document.all.theTop.style.cursor =
"crosshair";
                              theCursor =
document.all.theTop.style.cursor;
                        }

                  } else {
                        alert(msgList[46]);
                  }
                  //alert("Function Not Implemented");
            break

      // Measure-Unit function
      case "measure":
            panning=false;
            zooming=false;
            selectBox=false;
            shapeSelectBuffer = false;
            if (clickCount>0) {
                  if (totalMeasure==0) resetClick();
            }
            toolMode = 20;
            if (isIE)   {
                  document.all.theTop.style.cursor = "crosshair";
                  theCursor = document.all.theTop.style.cursor;
            }
            modeBlurb = modeList[12];
            if (clickType==1) {
```

```
                  //if (useTextFrame) parent.TextFrame.location= appDir
+ "measure.htm";
                        showLayer("measureBox");
                        updateMeasureBox();
            }
            showGeocode=false;
            break

      case "setunits":
            if (useTextFrame) {
                  parent.TextFrame.location = "setUnits.htm";
            } else {
                  window.open((appDir +
"setUnits.htm"),"OptionWindow","width=575,height=120,scrollbars=yes,res
izable=yes");
            }
            break

      // Graphic Selection functions
      case "shape":
            panning=false;
            zooming=false;
            selectBox=false;
            shapeSelectBuffer = false;
            toolMode = 21;
            if (isIE)   {
                  document.all.theTop.style.cursor = "crosshair";
                  theCursor = document.all.theTop.style.cursor;
            }

            modeBlurb = modeList[13];

            showGeocode=false;

            hideLayer("measureBox");
            break

      case "selectbox":
            panning=false;
            zooming=false;
            // select mode - requires aimsSelect.js
            if (canQuery) {
                  toolMode = 10;
                  queryTool=0;
                  clickCount=0;
                  showBuffer=false;
                  if (isIE)   {
                        document.all.theTop.style.cursor = "crosshair";
                        theCursor = document.all.theTop.style.cursor;
                  }

                  modeBlurb = modeList[4];
            } else {
                  alert(msgList[46]);
            }
```

```
            //alert("Function Not Implemented");
            showGeocode=false;
            if (useTextFrame) parent.TextFrame.document.location=
appDir + "text.htm";
            hideLayer("measureBox");
            break

      case "researchstore":
            panning=false;
            zooming=false;
            alert("handling agendastore in clickfunction");
            // select mode - requires aimsSelect.js
            if (canQuery) {
                  toolMode = 10;
                  researchset = true; // variable set so that select is
handled different, needs to be reset in aimsselect.js
                  alert(researchset);
                  queryTool=0;
                  clickCount=0;
                  showBuffer=false;
                  if (isIE)   {
                        document.all.theTop.style.cursor = "crosshair";
                        theCursor = document.all.theTop.style.cursor;
                  }

                  modeBlurb = "Select Rectangle";
            } else {
                  alert("Cannot query Service\nIdentify, Select, and
Query functions are disabled.");
            }
            //alert("Function Not Implemented");
            showGeocode=false;
            if (useTextFrame) parent.TextFrame.document.location=
appDir + "text.htm";
            hideLayer("measureBox");
            break


      case "selectpoint":
            panning=false;
            zooming=false;
            shapeSelectBuffer = false;
            // select mode - requires aimsSelect.js
            if (canQuery) {
                  toolMode = 11;
                  queryTool=0;
                  clickCount=0;
                   resetClick();
                  if (isIE)   {
                        document.all.theTop.style.cursor = "hand";
                        theCursor = document.all.theTop.style.cursor;
                  }

                  modeBlurb = modeList[5];
            } else {
```

```
                  alert(msgList[46]);
            }
            //alert("Function Not Implemented");
            showGeocode=false;
            showBuffer=false;
            hideLayer("measureBox");
            break

      case "selectline":
            panning=false;
            zooming=false;
            shapeSelectBuffer = false;
            // select mode - requires aimsSelect.js
            if (canQuery) {
                  toolMode = 12;
                  queryTool=0;
                  if (isIE)   {
                        document.all.theTop.style.cursor = "crosshair";
                        theCursor = document.all.theTop.style.cursor;
                  }
                  hideLayer("measureBox");
                  if (useTextFrame) {
                        parent.TextFrame.document.location= appDir +
"selectline.htm";
                  } else {
                        Win1 =
open("selectline.htm","QueryWindow","width=575,height=150,scrollbars=ye
s,resizable=yes");
                  }
                  modeBlurb = modeList[6];
            } else {
                  alert(msgList[46]);
            }
            //alert("Function Not Implemented");
            showGeocode=false;
            showBuffer=false;
            break

      case "selectpoly":
            panning=false;
            zooming=false;
            shapeSelectBuffer = false;
            // select mode - requires aimsSelect.js
            if (canQuery) {
                  toolMode = 13;
                  queryTool=0;
                  if (isIE)   {
                        document.all.theTop.style.cursor = "crosshair";
                        theCursor = document.all.theTop.style.cursor;
                  }
                  hideLayer("measureBox");
                  if (useTextFrame) {
                        parent.TextFrame.document.location= appDir +
"selectpoly.htm";
                  } else {
```

```
                        Win1 =
open("selectpoly.htm","QueryWindow","width=575,height=150,scrollbars=ye
s,resizable=yes");
                  }
                  modeBlurb = modeList[7];
            } else {
                  alert(msgList[46]);
            }
            //alert("Function Not Implemented");
            showGeocode=false;
            showBuffer=false;
            break

      case "selectshape":
            panning=false;
            zooming=false;
            shapeSelectBuffer = false;
            // select mode - requires aimsSelect.js
            if (canQuery) {
                  toolMode = 16;
                  queryTool=0;
                  if (isIE)   {
                        document.all.theTop.style.cursor = "crosshair";
                        theCursor = document.all.theTop.style.cursor;
                  }
                  hideLayer("measureBox");
                  if (useTextFrame) {
                        parent.TextFrame.document.location= appDir +
"select.htm";
                  } else {
                        Win1 =
open("select.htm","QueryWindow","width=575,height=150,scrollbars=yes,re
sizable=yes");
                  }
                  modeBlurb = modeList[8];
            } else {
                  alert(msgList[46]);
            }
            //alert("Function Not Implemented");
            showGeocode=false;
            showBuffer=false;
            break
      ///*
      case "buffershape":
            panning=false;
            zooming=false;

            // interactive shape buffer - not implemented
            if (canQuery) {
                  toolMode = 17;
                  //toolMode = 16;
                  queryTool=0;
                  shapeSelectBuffer = true;
                  if (isIE)   {
                        document.all.theTop.style.cursor = "crosshair";
```

```
                        theCursor = document.all.theTop.style.cursor;
                  }
                  hideLayer("measureBox");
                  if (useTextFrame) {
                        parent.TextFrame.document.location= appDir +
"shapeBuffer.htm";
                  } else {
                        Win1 =
open("shapeBuffer.htm","QueryWindow","width=575,height=150,scrollbars=y
es,resizable=yes");
                  }
                  modeBlurb = modeList[11];
            } else {
                  alert(msgList[46]);
            }
            //alert("Function Not Implemented");
            showGeocode=false;
            showBuffer=false;
            break
            //*/
      // Geocode Function
      case "geocode":
            panning=false;
            zooming=false;
            selectBox=false;
            shapeSelectBuffer = false;
            // geocode mode - requires aimsGeocode.js
            hideLayer("measureBox");
            modeBlurb = modeList[14];
            setupGeocode();
            //parent.TextFrame.document.location= appDir +
"addmatch.htm";
            break

      // Query - Search - Find functions
      //if ((toolName=="attributesel") || (toolName=="query")) {
      case "query":
            // query mode - requires aimsQuery.js
            panning=false;
            zooming=false;
            selectBox=false;
            shapeSelectBuffer = false;
            queryStartRecord=1;
            //toolMode=
            queryTool=51;
            if (canQuery) {
                  LayerFields.length=1;
                  LayerFieldType.length=1;
                  LayerFieldCount=0;
                  toolMode=8;
                  modeBlurb=modeList[15];

                  fieldIndex=0;
                  setQueryString="";
                  hideLayer("measureBox");
```

```
            queryForm();
      } else {
            alert(msgList[46]);
      }
      showGeocode=false;
      showBuffer=false;
      break

case "storedquery":
      // storedquery mode - requires aimsQuery.js
      panning=false;
      zooming=false;
      selectBox=false;
      shapeSelectBuffer = false;
      queryStartRecord=1;
      queryTool=1;
      toolMode=51;
      modeBlurb="Search";
      if (canQuery) {
            toolMode=51;
            modeBlurb=modeList[16];
            fieldIndex=0;
            setQueryString="";
            hideLayer("measureBox");
            getStoredQueries();
      } else {
            alert(msgList[46]);
      }
      showGeocode=false;
      showBuffer=false;
      break

case "find":
      //find
      toolMode=9;
      panning=false;
      zooming=false;
      selectBox=false;
      shapeSelectBuffer = false;
      queryStartRecord=1;
      queryTool=1;
      if (canQuery) {
            LayerFields.length=1;
            LayerFieldType.length=1;
            LayerFieldCount=0;

            fieldIndex=0;
            setQueryString="";
            hideLayer("measureBox");
            modeBlurb = modeList[17];
            findForm();
      } else {
            alert(msgList[46]);
      }
      showGeocode=false;
```

```
      showBuffer=false;
      break

case "clearsel":
       clearSelection();
       break

// Buffer function
case "buffer":
      //buffer - requires aimsBuffer.js
      if (useBuffer) {
            if (checkSelected()) {
                  toolMode = 25;
                  shapeSelectBuffer = false;
                  modeBlurb = modeList[18];
                  writeBufferForm();
            } else {
                  showBuffer=false;
                  alert(msgList[48]);
            }
      } else {
            alert(msgList[49]);
      }
      break

case "options":
      writeOptionForm();
      break

// Print function
case "print":
       printIt();
       break

// custom modes
case "dbidentify":
      panning=false;
      zooming=false;
      selectBox=false;
      shapeSelectBuffer = false;
      // identify mode - requires custom db query - not in basic
      if (canQuery) {
            toolMode = 40;
            if (isIE)   {
                  document.all.theTop.style.cursor = "hand";
                  theCursor = document.all.theTop.style.cursor;
            }

            modeBlurb = modeList[3];
      } else {
            alert(msgList[46]);
      }
      //alert("Function Not Implemented");

      showGeocode=false;
```

```
            if (useTextFrame) parent.TextFrame.document.location=
appDir + "text.htm";
            hideLayer("measureBox");
            break


      case "extract":
             extractIt();
             break

      case "legend":
            if (aimsLegendPresent) {
                  if (imsURL!="") {
                        if (hasTOC) {
                              if (legendVisible) {

                                    legendVisible=false;
                              //writeLayerList();

      parent.TOCFrame.document.location=appDir+"toc.htm";
                              } else {
                                    legendVisible=true;
                                    getLegend();
                              }
                        } else {
                              legendVisible=true;
                              getLegend();
                        }
                  } else {
                        alert(msgList[45]);
                  }
            } else {
                  alert(msgList[50]);
            }
            break

      case "layerlist":
            // put LayerList in separate window
            writeLayerListForm();
            break

      default:
            alert(msgList[51]);
      }
      modeName=modeBlurb;
      if (useModeFrame) {
            writeModeFrame(modeBlurb);
      } else if ((drawFloatingMode) && (modeLayerOn)) {
            writeModeLayers(modeBlurb);
      } else if ((modeRefreshMap) && (drawModeOnMap)) {
            //var theString = writeXML();
            sendMapXML();
      }
}
```

```
// check for mouseup
function chkMouseUp(e) {
      if ((toolMode == 1) && (zooming)) {
                  stopZoomBox(e);

      }
      if ((toolMode == 2) && (zooming)) {
                  stopZoomOutBox(e);
      }
      if ((toolMode == 3) && (panning)) {
                  stopPan(e);

      }
      if ((toolMode == 10) && (selectBox)) {
                  stopSelectBox(e);
      }

      return false;

}

// perform appropriate action with mapTool
function mapTool (e) {
      var theButton= 0;
      // get the button pushed... if right, ignore... let browser do
the popup... it will anyway
      if (isNav) {
            theButton = e.which;
      } else {
            theButton =window.event.button;
      }
      if (theButton==leftButton) {
            getImageXY(e);
            if ((mouseX>=0) && (mouseX<iWidth) && (mouseY>=0) &&
(mouseY<iHeight)) {
                  //if ((!isNav) || (!is5up)) {
                  if ((hasOVMap) && (ovIsVisible) &&
(mouseX<i2Width+ovBoxSize) && (mouseY<i2Height) && (ovMapIsLayer)) {
                              //alert(mouseX + ", " + mouseY);
                              ovMapClick(mouseX,mouseY);

                              window.status = "On OV Map Area";
                        //}
                  } else {
                        //alert(mouseX + "," + mouseY);

                        switch(toolMode) {
                        case 1:
                                    startZoomBox(e);
                                    return false;
                                    break

                        case 2:
                                    startZoomOutBox(e);
                                    return false;
```

```
                              break
                        case 3:
                              startPan(e);
                              return false;
                              break

                        case 4:
                              identify(e);
                              break

                        case 5:
                              // identify all
                              identifyAll(e);
                              break

                        // custom modes
                        /*
                        case 6:
                              // route - requires custom route routine
- not in default
                              routeClick = routeClick + 1;
                              if (routeClick > 2) routeClick = 2;
                              setRouteXY()
                              writeRoutePage();
                              break
                        case 7:
                              // proximity - requires custom proximity
routine - not in default
                              proxCount=0;
                              proximitySearch(e);
                              break

                        */
                        case 10:
                              //select(e);
                                    startSelectBox(e);
                                    return false;
                                    break
                        case 11:
                              //select point
                              if (checkIfActiveLayerAvailable()) {
                                    select(e);
                              }
                              break
                        case 12:
                              //select line
                              if (checkIfActiveLayerAvailable()) {
                                    clickType=2;
                                    clickAddPoint();
                                    //Mo: added here:

                                    if (useTextFrame) {
                                          if
(parent.TextFrame.document.title!==modeList[60]) {
```

```
      parent.TextFrame.document.location= appDir + "selectline.htm";
                                          }
                                    }
                              }
                              break
                        case 13:
                              //select polygon
                              if (checkIfActiveLayerAvailable()) {
                                    clickType=3;
                                    clickAddPoint();
                                    if (useTextFrame) {
                                          if
(parent.TextFrame.document.title!==modeList[7]) {

      parent.TextFrame.document.location= appDir + "selectpoly.htm";
                                          }
                                    }
                              }
                              break
                        case 15:
                              // hyperlink
                              hyperLink(e);
                              break
                        case 16:
                              //select shape
                              if (checkIfActiveLayerAvailable()) {
                                    clickType=2;
                                    clickAddPoint();
                                    if (useTextFrame) {
                                          if
(parent.TextFrame.document.title!==modeList[8]) {

      parent.TextFrame.document.location= appDir + "select.htm";
                                          }
                                    }
                              }
                              break
                              ///*
                        case 17:
                              //buffer shape -
                              if (checkIfActiveLayerAvailable()) {
                                    clickType=2;
                                    clickAddPoint();
                                    if (useTextFrame) {
                                          if
(parent.TextFrame.document.title!=modeList[11]) {

      parent.TextFrame.document.location= appDir + "shapeBuffer.htm";
                                          }
                                    }
                              }
                              break
                              //*/
                        case 20:
```

```
                              // measure
                              clickType=1;
                              clickAddPoint();
                              break
                        case 21:
                              // shape
                              clickType=4;
                              clickAddPoint();
                              break
                        case 30:
                              // hyperlink
                              hyperLinkAny(e);
                              break
                        case 40:
                              // db identify - requires custom db query
- not in default
                              if (aimsDBPresent) {
                                    matchDBLinkLayer(dbLinkLayer);
                                    dbIdentify(e);
                              }
                              break
                        default:
                              if (toolMode>=1000) {
                                    customMapTool(e);
                              }
                        }
                  }
            }
      }
}

// update measureBox layer
function updateMeasureBox() {
      if (isNav4) {
            var theForm =
document.layers["measureBox"].document.forms[0];
      } else {
            //var theForm = document.measureBox.forms[0];
            var theForm = document.forms[0];
      }
      var j = 1;
      for (var i=0;i<sUnitList.length;i++) {
            if (ScaleBarUnits==sUnitList[i]) j=i;
      }
            var u = Math.pow(10,numDecimals);
            var tMeas = 0;
            if (totalMeasure!=0) tMeas =
parseInt(totalMeasure*u+0.5)/u;
      theForm.theMeasTotal.value = tMeas + " " + unitList[j];
      theForm.theMeasSegment.value = currentMeasure + " " +
unitList[j];
      showLayer("measureBox");

}
```

```
########################################
#      toolbar.htm                     #
#                                      #
########################################




<META HTTP-EQUIV="Pragma" CONTENT="no-cache">
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-
1">
<HTML>
<HEAD>

      <SCRIPT LANGUAGE=JAVASCRIPT>
            var t = parent.MapFrame;
            document.writeln('<TITLE> ' + t.titleList[12] +
'</TITLE>');
            var lBreak = "<br>";
            var isSecond=true;
            var currModeName="zoomin";
            var version = navigator.appVersion;
            var browser = navigator.appName;
            //alert(browser);
            function revertToolPic() {
                  // reset tool icons to non-set mode
                  //alert(lastMode);
                  if (parent.MapFrame.useZoomIn)
document.zoomin.src="images/zoomin_1.gif";
                  if (parent.MapFrame.useZoomOut)
document.zoomout.src="images/zoomout_1.gif";
                  if (parent.MapFrame.useIdentify)
document.identify.src="images/identify_1.gif";
                  if (parent.MapFrame.usePan)
document.pan.src="images/pan_1.gif";
                  if (parent.MapFrame.useMeasure)
document.measure.src="images/measure_1.gif";
                  if (parent.MapFrame.useSelect) {
                        document.rect.src="images/select_rect_1.gif";
                        document.line.src="images/select_1.gif";
                  }
                  if (parent.MapFrame.useBufferShape)
document.buffershape.src="images/buffershape_1.gif";
                  if (parent.MapFrame.useHyperLink)
document.hyperlink.src="images/hotlink_1.gif";
                  if (parent.MapFrame.useHyperLinkAny)
document.hyperlink.src="images/hotlink_1.gif";
                  if (parent.MapFrame.useIdentifyAll)
document.identifyall.src="images/identifyall_1.gif";
            }
            function setToolPic(functName) {
                  // set clicked button icon to set mode
                  //if (functName!=parent.MapFrame.modeName) {
```

```
                        revertToolPic();
                        parent.MapFrame.focus();
                        //alert(functName);
                        if (functName=="Zoom In") {

      document.zoomin.src="images/zoomin_2.gif";
                        } else if (functName=="Zoom Out") {

      document.zoomout.src="images/zoomout_2.gif";
                        } else if (functName=="Identify") {

      document.identify.src="images/identify_2.gif";
                        } else if (functName=="Pan") {
                              document.pan.src="images/pan_2.gif";
                        } else if (functName=="Measure") {

      document.measure.src="images/measure_2.gif";
                        } else if (functName=="Select Rectangle") {

      document.rect.src="images/select_rect_2.gif";
                        } else if (functName=="Buffer Shape") {

      document.buffershape.src="images/buffershape_2.gif";
                        } else if (functName=="Select Line/Polygon") {
                              document.line.src="images/select_2.gif";
                        } else if (functName=="HyperLink") {

      document.hyperlink.src="images/hotlink_2.gif";
                        } else if (functName=="Identify All") {

      document.identifyall.src="images/identifyall_2.gif";
                        } else if (functName=="researchstore") {
                              alert("settingtoolpic");

      document.researchstore.src="images/select_line_2.gif";
                        } else if (functName=="researchpp") {
                              alert("settingtoolpic");

      document.researchstore.src="images/select_point_2.gif";
                        } else if (functName=="researchppstop") {
                              alert("settingtoolpic");

      document.researchstore.src="images/wrench.gif";
                        }
                  //}
            }

            function openGeoNetwork() {
                  var Win1 =
window.open("http://www.geographynetwork.com","","scrollbars,resizable,
toolbar,width=750,height=580");
            }

      </SCRIPT>
</HEAD>
```

```
<BODY BGCOLOR="Silver" TEXT="Black" LINK="White" VLINK="White"
LEFTMARGIN=0 TOPMARGIN=0 RIGHTMARGIN=0 ALINK="White"
onload="setToolPic(parent.MapFrame.modeBlurb)">
      <DIV ALIGN="center">
            <TABLE BORDER="1" CELLSPACING="0" CELLPADDING="1"
ALIGN="CENTER" VALIGN="MIDDLE" BGCOLOR="White" BORDERCOLOR="Gray"
BORDERCOLORLIGHT="Silver" BORDERCOLORDARK="Black">

                  <SCRIPT TYPE="text/javascript"
LANGUAGE="JavaScript1.2">
                        /*
                        if (parent.MapFrame.useGeoNetwork) {
                              document.writeln('<tr><TD COLSPAN="2"
align="center" valign="middle">');
                              document.writeln('<IMG
SRC="images/GN_tool1.gif" WIDTH=18 HEIGHT=18 HSPACE=0 VSPACE=0 BORDER=0
ALT="Geography Network" onmousedown="openGeoNetwork()"
onmouseover="window.status=\'Geography Network\'">');
                              document.writeln('</TD></tr>');
                        }
                        */
                        document.write('<tr>');
                        if ((parent.MapFrame.hasTOC) &&
(parent.MapFrame.aimsLegendPresent)) {
                              // Legend toggle. . . requires
aimsLegend.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/togglelegend.gif" width=16 height=16 hspace=1 vspace=0
border=0 alt=" ' + t.buttonList[14] + '" name="legend"
onmousedown="parent.MapFrame.clickFunction(\'legend\');"
onmouseover="window.status=\' ' + t.buttonList[14] + '\'">');
                              isSecond = !isSecond;
                              document.writeln('</td>');
                        }
                        if (parent.MapFrame.hasOVMap) {
                              // Overview Map toggle . . . requires
overview map
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/overview_html.gif" width=16 height=16 hspace=1 vspace=0
border=0 alt=" ' + t.buttonList[15] + '" name="refmap"
onmousedown="parent.MapFrame.toggleOVMap();"
onmouseover="window.status=\' ' + t.buttonList[15] + '\'">');

                              isSecond = !isSecond;
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }
                        if (parent.MapFrame.useZoomIn) {
                              // Zoom In . . . requires
aimsNavigation.js
```

```
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/zoomin_1.gif" width=16 height=16 hspace=1 vspace=0 border=0
alt=" ' + t.buttonList[16] + '" name="zoomin"
onmousedown="parent.MapFrame.clickFunction(\'zoomin\');
setToolPic(\'Zoom In\');" onmouseover="window.status=\' ' +
t.buttonList[16] + '\'">');
                              isSecond = !isSecond;
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }
                        if (parent.MapFrame.useZoomOut) {
                              // Zoom Out . . . requires
aimsNavigation.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/zoomout_1.gif" width=16 height=16 hspace=1 vspace=1
border=0 alt=" ' + t.buttonList[17] + '" name="zoomout"
onmousedown="parent.MapFrame.clickFunction(\'zoomout\');
setToolPic(\'Zoom Out\');" onmouseover="window.status=\' ' +
t.buttonList[17] + 't\'">');
                              isSecond = !isSecond;
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }
                        if (parent.MapFrame.useFullExtent) {
                              // Full Extennt . . . requires
amisLayers.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/fullextent_1.gif" width=16 height=16 hspace=1 vspace=1
border=0 alt=" ' + t.buttonList[18] + '"
onmousedown="parent.MapFrame.clickFunction(\'fullextent\');"
onmouseover="window.status=\' ' + t.buttonList[18] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }

                        if (parent.MapFrame.useZoomActive) {
                              // Zoom to Active Layer . . . requires
aimsLayers.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/zoomactive_1.gif" width=16 height=16 hspace=1 vspace=1
border=0 alt=" ' + t.buttonList[19] + '"
onmousedown="parent.MapFrame.clickFunction(\'zoomactive\');"
onmouseover="window.status=\' ' + t.buttonList[19] + '\'">');
```

```
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }

                        if (parent.MapFrame.useZoomLast) {
                              // Zoom to previous extent . . . requires
aimsLayers.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/zoomlast.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[20] + '"
onmousedown="parent.MapFrame.clickFunction(\'zoomlast\');"
onmouseover="window.status=\' ' + t.buttonList[20] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }

                        if (parent.MapFrame.usePan) {
                              // Pan Button . . . requires
aimsNavigation.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/pan_1.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[21] + '" name="pan"
onmousedown="parent.MapFrame.clickFunction(\'pan\');
setToolPic(\'Pan\');" onmouseover="window.status=\' ' +
t.buttonList[21] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');

                        }

                        if (parent.MapFrame.usePanNorth) {
                              // Pan North . . . requires
aimsNavigation.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/pan_north.gif" width=16 height=16 hspace=1 vspace=1
border=0 alt=" ' + t.buttonList[22] + '"
onmousedown="parent.MapFrame.panButton(2);"
onmouseover="window.status=\' ' + t.buttonList[22] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
```

```
                        }

                        if (parent.MapFrame.usePanSouth) {
                              // Pan South . . . requires
aimsNavigation.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/pan_south.gif" width=16 height=16 hspace=1 vspace=1
border=0 alt=" ' + t.buttonList[23] + '"
onmousedown="parent.MapFrame.panButton(4);"
onmouseover="window.status=\' ' + t.buttonList[23] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }

                        if (parent.MapFrame.usePanWest) {
                              // Pan West . . . requires
aimsNavigation.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/pan_west.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[24] + '"
onmousedown="parent.MapFrame.panButton(1);"
onmouseover="window.status=\' ' + t.buttonList[24] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }

                        if (parent.MapFrame.usePanEast) {
                              // Pan East . . . requires
aimsNavigation.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/pan_east.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[25] + '"
onmousedown="parent.MapFrame.panButton(3);"
onmouseover="window.status=\' ' + t.buttonList[25] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }
                              // only one of these next two
(useHyperLink or useHyperLinkAny) can be true - useHyperLink takes
priority
                        if (parent.MapFrame.useHyperLink) {
                              // HyperLink . . . requires
aimsIdentify.js
```

```
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/hotlink.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[26] + '" name="hyperlink"
onmousedown="parent.MapFrame.clickFunction(\'hyperlink\');
setToolPic(\'HyperLink\');" onmouseover="window.status=\' ' +
t.buttonList[26] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }
                        if (parent.MapFrame.useHyperLinkAny) {
                              // HyperLink . . . requires
aimsIdentify.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/hotlink.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[26] + '" name="hyperlink"
onmousedown="parent.MapFrame.clickFunction(\'hyperlinkany\');
setToolPic(\'HyperLink\');" onmouseover="window.status=\' ' +
t.buttonList[26] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }
                              // only one of these next two
(useIdentify or useIdentifyAll) can be true - useIdentify takes
priority
                        if (parent.MapFrame.useIdentify) {
                              // Identify . . . requires
aimsIdentify.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/identify_1.gif" width=16 height=16 hspace=1 vspace=1
border=0 alt=" ' + t.buttonList[27] + '" name="identify"
onmousedown="parent.MapFrame.clickFunction(\'identify\');
setToolPic(\'Identify\');" onmouseover="window.status=\' ' +
t.buttonList[27] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }

                     if (parent.MapFrame.useIdentifyAll) {
                              // Identify All - identify on all visible
feature layers. . . drill down
                              document.write('<td align="center"
valign="middle">');
```

```
                              document.write('<img
src="images/identifyall_1.gif" width=16 height=16 hspace=1 vspace=1
border=0 alt="' + t.modeList[20] + '" name="identifyall"
onmousedown="parent.MapFrame.clickFunction(\'identifyall\');
setToolPic(\'Identify All\');" onmouseover="window.status=\'' +
t.modeList[20] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }

                        //if ((parent.MapFrame.useQuery) &&
(version.indexOf("MSIE 5")==-1)) {
                        if (parent.MapFrame.useQuery) {
                              // Query . . . requires aimsQuery.js
                              // IE 5.0 has big problems with the query
stuff
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/query_1.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[28] + '"
onmousedown="parent.MapFrame.clickFunction(\'query\');
setToolPic(\'ClearOut\');" onmouseover="window.status=\' ' +
t.buttonList[28] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }

                        if (parent.MapFrame.useStoredQuery) {
                              // Search . . . requires aimsQuery.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/search.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[29] + '"
onmousedown="parent.MapFrame.clickFunction(\'storedquery\');
setToolPic(\'ClearOut\');" onmouseover="window.status=\' ' +
t.buttonList[29] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }

                        if (parent.MapFrame.useFind) {
                              // Find . . . requires aimsQuery.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/find_1.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[30] + '"
```

```
onmousedown="parent.MapFrame.clickFunction(\'find\');
setToolPic(\'ClearOut\');" onmouseover="window.status=\' ' +
t.buttonList[30] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }

                        if (parent.MapFrame.useMeasure) {
                              // Measure
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/measure_1.gif" width=16 height=16 hspace=1 vspace=1
border=0 alt=" ' + t.buttonList[31] + '" name="measure"
onmousedown="parent.MapFrame.clickFunction(\'measure\');
setToolPic(\'Measure\');" onmouseover="window.status=\' ' +
t.buttonList[31] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }

                        if (parent.MapFrame.useSetUnits) {
                              // Set Units
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/setunits.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[32] + '"
onmousedown="parent.MapFrame.clickFunction(\'setunits\');"
onmouseover="window.status=\' ' + t.buttonList[32] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }

                        if (parent.MapFrame.useBuffer) {
                              // Buffer . . . requires aimsBuffer.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/buffer_1.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[33] + '" name="buffer"
onmousedown="parent.MapFrame.clickFunction(\'buffer\');
setToolPic(\'Buffer\');" onmouseover="window.status=\' ' +
t.buttonList[33] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }
```

```
                        if (parent.MapFrame.useSelect) {
                              // Graphic Selection tools . . . requires
aimsSelect.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/select_rect_1.gif" width=16 height=16 hspace=1 vspace=1
border=0 alt=" ' + t.buttonList[34] + '" name="rect"
onmousedown="parent.MapFrame.clickFunction(\'selectbox\');
setToolPic(\'Select Rectangle\');" onmouseover="window.status=\' ' +
t.buttonList[34] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/select_1.gif" width=16 height=16 hspace=0 vspace=1 border=0
alt=" ' + t.buttonList[35] + '" name="line"
onmousedown="parent.MapFrame.clickFunction(\'selectshape\');
setToolPic(\'Select Line/Polygon\');" onmouseover="window.status=\' ' +
t.buttonList[35] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }
                        ///*
                        if (parent.MapFrame.useBufferShape) {
                              // Buffer Shape. . . requires
aimsSelect.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/buffershape_1.gif" width=16 height=16 hspace=1 vspace=1
border=0 alt="Buffer Shape" name="buffershape"
onmousedown="parent.MapFrame.clickFunction(\'buffershape\');
setToolPic(\'Buffer Shape\');" onmouseover="window.status=\'Buffer
Shape\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }
                        //*/

                        if (parent.MapFrame.useClearSelect) {
                              // Clear selection . . . requires
aimsIdentify.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/clearhighlight_1.gif" width=16 height=16 hspace=1 vspace=1
```

```
border=0 alt=" ' + t.buttonList[36] + '"
onmousedown="parent.MapFrame.clearSelection();"
onmouseover="window.status=\' ' + t.buttonList[36] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }
                        if (parent.MapFrame.useGeocode) {
                              // Geocode. . . requires aimsGeocode.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/geocode.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[37] + '"
onmousedown="parent.MapFrame.clickFunction(\'geocode\');"
onmouseover="window.status=\' ' + t.buttonList[37] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }
                        if (parent.MapFrame.usePrint) {
                              // Print. . . requires aimsPrint.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/print_1.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[38] + '"
onmousedown="parent.MapFrame.clickFunction(\'print\');"
onmouseover="window.status=\' ' + t.buttonList[38] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }
                        if (parent.MapFrame.useExtract) {
                              // Extract. . . zip and ship. . .
requires aimsCustom.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/zipship.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[39] + '"
onmousedown="parent.MapFrame.clickFunction(\'extract\');"
onmouseover="window.status=\' ' + t.buttonList[39] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }
                        if (parent.MapFrame.allowOptions) {
                              // Options. . . requiers
aimsOptions.js... allowOptions is set to true in this file
```

```
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/wrench.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[40] + '"
onmousedown="parent.MapFrame.writeOptionForm();"
onmouseover="window.status=\' ' + t.buttonList[40] + '\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
                              if (isSecond)
document.write('</tr><tr>');
                        }
                        if (parent.MapFrame.canLoad) {
                              // Load MapService. . . requires
aimsGeneric.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/open_1.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt=" ' + t.buttonList[41] + '"
onmousedown="parent.MapFrame.loadMapForm();"
onmouseover="window.status=\' ' + t.buttonList[41] + '\'">');
                              document.writeln('</td>');
                        }
                        if (parent.MapFrame.hasresearchStore) {
                              //Mo: Make user AOI as rectangle or
circle
                              // Graphic Selection tools . . . requires
aimsSelect.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/select_line_1.gif" width=16 height=16 hspace=1 vspace=1
border=0 alt="Draw AOI" name="researchstore"
onmousedown="parent.MapFrame.clickFunction(\'researchstore\');
setToolPic(\'researchstore\');" onmouseover="window.status=\'Make AOI
rectangle\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');

                        }
                        if (parent.MapFrame.hasresearchpp) {
                              //Mo: start clicking
                              // Graphic Selection tools . . . requires
aimsSelect.js
                              document.write('<td align="center"
valign="middle">');
                              document.write('<img
src="images/select_point_1.gif" width=16 height=16 hspace=1 vspace=1
border=0 alt="Draw AOI" name="researchstore"
onmousedown="parent.MapFrame.clickFunction(\'researchpp\');
setToolPic(\'researchpp\');" onmouseover="window.status=\'Select
Potential Path\'">');
                              isSecond = !isSecond
                              document.writeln('</td>');
```

```
                                if (isSecond)
document.write('</tr><tr>');

                            }
                            if (parent.MapFrame.hasresearchpp) {
                                //Mo: stop adding links
                                // Graphic Selection tools . . . requires
aimsSelect.js
                                document.write('<td align="center"
valign="middle">');
                                document.write('<img
src="images/wrench.gif" width=16 height=16 hspace=1 vspace=1 border=0
alt="Draw AOI" name="researchstorestop"
onmousedown="parent.MapFrame.clickFunction(\'researchppstop\');
setToolPic(\'researchppstop\');" onmouseover="window.status=\'Commit
the selected links\'">');
                                isSecond = !isSecond
                                document.writeln('</td>');


                            }
                            document.writeln('</tr>');
                            document.writeln('</TABLE>');


                    </SCRIPT>


        </DIV>
</BODY>
</HTML>
```

```
################################################
#           mointersect.bas:: mointersect.exe  #
################################################


Attribute VB_Name = "intersect"
Dim m_outputfeatureclass As IFeatureClass
Dim m_Attributecoll As Collection

Public Sub Main()

'Recieve parameters:
Dim uid As Long
Dim usernum As Long
Dim tokenarray As Variant
Dim adams As IFeatureClass


Open "c:\logintersectexe.txt" For Output As #1
strValue = Command()
Write #1, strValue
Call ParseDelimitedString(strValue, tokenarray, "&")


uid = CLng(tokenarray(0))
usernum = CLng(tokenarray(1))

'Load the fclasses corresponding to:
Set adams = openShapeFileWorkspace2("D:\\reserachdata\\networkshapes",
"adamsnet")


'Construct a polygon from a query to tuserpoly to retrieve record for
id = XX

Dim tuserpoly As Recordset
Dim tuserpoly_base As Database
'Dim thequerydef As DAO.QueryDef
Dim selectstring As String
Set tuserpoly_base =
OpenDatabase("D:\ArcIMS\reserachdata\tuserpoly.mdb")

selectstring = "SELECT * FROM Tuserpoly WHERE uniqueid=" & uid

Set tuserpoly = tuserpoly_base.OpenRecordset(selectstring)

'No need to loop, there should be only one record with this uid!

Dim maxx As Double
Dim maxy As Double
Dim minx As Double
Dim miny As Double

tuserpoly.MoveFirst
maxx = tuserpoly.Fields("Maxx").Value
```

```
maxy = tuserpoly.Fields("Maxy").Value
minx = tuserpoly.Fields("Minx").Value
miny = tuserpoly.Fields("Miny").Value

'MsgBox miny
tuserpoly.Close
tuserpoly_base.Close

'Construct geometry with these coordinates:
Dim pGeom As IPolygon
Set pGeom = New Polygon
Dim pSegmentColl As ISegmentCollection
Dim pSegment As ISegment
Dim pPoint As IPoint
Dim pEnv As IEnvelope

'MO: Have to set rectangle !!! adding segments does not work

Set pEnv = New Envelope
pEnv.XMax = maxx
pEnv.XMin = minx
pEnv.YMax = maxy
pEnv.YMin = miny

Set pSegmentColl = pGeom
pSegmentColl.SetRectangle pEnv


'Set tuserpoly =
openShapeFileWorkspace("D:\ArcIMS\reserachdata\usershapes",
"tuserpoly")

Call DoIntersection(pGeom, adams, uid, usernum)

Exit Sub

End Sub
Private Sub DoIntersection(pIntersectFeature As IPolygon, pOverlayLayer
As IFeatureClass, objectid As Long, usernum As Long)
'On Error GoTo Errorhandler
  Dim pIntersectFClass As IFeatureClass
  Dim pFilter As IQueryFilter
  Dim pIntersectFCursor As IFeatureCursor
  'Dim pIntersectFeature As IFeature
  Dim pIntersectTopo As ITopologicalOperator2
  Dim pSpatialFilter As ISpatialFilter
  Dim pIntersectFields As IFields
  Dim pIntersectArea As IArea
  Dim pOverlayFCursor As IFeatureCursor
  Dim pOverlayFClass As IFeatureClass
  Dim pOverlayFields As IFields
  Dim pOverlayFeature As IFeature
  Dim pOverlayArea As IArea
  Dim pOutputfClass As IFeatureClass
  Dim newArea As IArea
```

```
  Dim theProportion As Double
  Dim newGeometry As IGeometry
  Dim pOverlayIndex As Long
  Dim pIntersectIndex As Long
  Dim pOldFieldsCount As Long
  Dim pNewFields As IFields
  Dim item As Long, lCount As Long, bFlag As Boolean
  Dim newField As IField, pOldField As IField, pCollName As String
  Dim pNewFeatureCursor As IFeatureCursor, pNewBuffer As IFeatureBuffer
  Dim openstring As String
  Dim polypointcol As IPointCollection
  Dim aadtval As Long


  'Open the tuserresults database:
  'Fetch the Access tables:
  'Dim tuserresults As Recordset
  'Dim tuserresults_base As Database
  'Dim selectstring As String

 'Set tuserresults_base =
OpenDatabase("D:\ArcIMS\reserachdata\tuserresults.mdb")

  'selectstring = "SELECT * FROM tuserresults"

  'Set tuserresults = tuserresults_base.OpenRecordset(selectstring)

  'MO: Write the XML file:
  openstring = "D:\ArcIMS\Website\moresearchsite\update-" & usernum &
objectid & ".xml"
  Open openstring For Output As #2

  Set pOverlayFClass = pOverlayLayer
  Set pOverlayFields = pOverlayFClass.Fields
  Print #2, "<USERSHAPE>"

  'Create the new shapefile or geodatabase layer

  'Set pOutputfClass = Editshape()

  'MsgBox pIntersectFeature.GeometryType
  'MsgBox pIntersectFeature.IsEmpty
  'MsgBox pIntersectFeature.Envelope.XMax
  'MsgBox pIntersectFeature.Envelope.XMin
  'MsgBox pIntersectFeature.Envelope.YMax
  'MsgBox pIntersectFeature.Envelope.YMin
  Set pIntersectTopo = pIntersectFeature

  'get the intersecting layer features
   Set pSpatialFilter = New SpatialFilter
   pSpatialFilter.GeometryField = "shape"
   Set pSpatialFilter.Geometry = pIntersectFeature
   pSpatialFilter.SpatialRel = esriSpatialRelIntersects
```

```
  Set pOverlayFCursor = pOverlayFClass.Search(pSpatialFilter, False)


  'Cycle through the features intersecting that buffer poly
  Set pOverlayFeature = pOverlayFCursor.NextFeature
  If (pOverlayFeature Is Nothing) Then
    'MsgBox "Nothing"
  End If

    'tuserresults.MoveFirst
    'tuserresults.Edit
    While Not pOverlayFeature Is Nothing

      Set newGeometry = pIntersectTopo.intersect(pOverlayFeature.Shape,
2)
      aadtval = pOverlayFeature.Value(3)
      'Loop thru all the points in the collection and write X and Y's

      Set polypointcol = newGeometry

      Print #2, "    <OBJ>"
      For i = 0 To (polypointcol.PointCount - 1)
        thex = "        <X>" & polypointcol.Point(i).X & "</X>"
        they = "        <Y>" & polypointcol.Point(i).Y & "</Y>"
        Print #2, thex
        Print #2, they

      Next i
      thevalue = "        <AADT>" & aadtval & "</AADT>"
      Print #2, thevalue
      Print #2, "    </OBJ>"
      'Write this to tuserresults table:
      'tuserresults.AddNew
      'All we nedd is the linkid correspomding to the intersected link:
      'tuserresults.Fields("linkid") = pOverlayFeature.Value(0)
      'tuserresults.Update
      Set pOverlayFeature = pOverlayFCursor.NextFeature
    Wend
    'tuserresults.Close
    'tuserresults_base.Close
 Print #2, "</USERSHAPE>"
 Exit Sub


Errorhandler:
  'MsgBox Err.Number & " " & Err.Description, vbCritical, "error in
intersection"
  Write #1, Err.Number & " " & Err.Description, vbCritical, "error in
intersection"
End Sub

Private Function AddTheFields(pIntersectFClass As IFeatureClass, _
 pOverlayFClass As IFeatureClass, pShortNames As Boolean) As IFields
On Error GoTo Errorhandler
  Dim pNewFields As IFields
```

```
  Dim newField As IField
  Dim newFieldEdit As IFieldEdit
  Dim pIntersectFields As IFields
  Dim pOverlayFields As IFields
  Dim pFieldsEdit As IFieldsEdit
  Dim item As Long
  Dim pOverlayField As IField
  Dim theIndex As Long
  Dim pGeoDef As IGeometryDefEdit
  Dim pGeoDataset As IGeoDataset
  Dim pFldEdt As IFieldEdit
  Dim aIndex As Long
  Dim pNewName As String

  Set pNewFields = New Fields
  Set pIntersectFields = pIntersectFClass.Fields
  Set pOverlayFields = pOverlayFClass.Fields
  Set pFieldsEdit = pNewFields
  For item = 0 To pOverlayFields.FieldCount - 1
    Set pOverlayField = pOverlayFields.Field(item)
    'Skip certain fields
    If pOverlayField.Type <> esriFieldTypeGeometry And
pOverlayField.Type <> esriFieldTypeOID _
     And UCase(pOverlayField.Name) <> "SHAPE_LENGTH" And
UCase(pOverlayField.Name) <> "SHAPE_AREA" Then

      On Error Resume Next
      theIndex = -1
      theIndex = pIntersectFields.FindField(pOverlayField.Name)
      If theIndex > -1 Then
        'Try to find alternative name
        pNewName = Left(pOverlayField.Name, 8) & "_1"
        theIndex = -1
        theIndex = pIntersectFields.FindField(pNewName)
        If theIndex > -1 Then
          'Skip Thefield if a field of the same name is already in
theFile
          Set AddTheFields = Nothing
          Exit Function
        End If
        theIndex = -1
        theIndex = pOverlayFields.FindField(pNewName)
        If theIndex > -1 Then
          'Skip Thefield if a field of the same name is already in
theFile
          Set AddTheFields = Nothing
          Exit Function
        End If
      Else
        pNewName = pOverlayField.Name
      End If

      Set newField = New Field
      Set newFieldEdit = newField
      With newFieldEdit
```

```
        .Name = pNewName
        .AliasName = pOverlayField.Name
        .Type = pOverlayField.Type
        .IsNullable = pOverlayField.IsNullable
        .Length = pOverlayField.Length
        .Precision = pOverlayField.Precision
      End With
      pFieldsEdit.AddField newField
    End If
  Next item

  Set AddTheFields = pFieldsEdit

  Exit Function

Errorhandler:
  'MsgBox Err.Number & " " & Err.Description, vbCritical, "error in
addfields"
End Function

Public Function openShapeFileWorkspace(Location As String, datasetname
As String) As IFeatureClass

Write #1, "Call to openshapefile"
Dim pInShpWorkspaceName As IWorkspaceName
Set pInShpWorkspaceName = New WorkspaceName
pInShpWorkspaceName.PathName = Location
pInShpWorkspaceName.WorkspaceFactoryProgID =
"esriCore.ShapefileWorkspaceFactory"

Dim pInShpFeatCLSNm As IFeatureClassName
Set pInShpFeatCLSNm = New FeatureClassName
Dim pShpDatasetName As IDatasetName
Set pShpDatasetName = pInShpFeatCLSNm
pShpDatasetName.Name = datasetname
Set pShpDatasetName.WorkspaceName = pInShpWorkspaceName

Dim pName As IName
Dim pInShpFeatCls As IFeatureClass
Set pName = pInShpFeatCLSNm
Set pInShpFeatCls = pName.Open  ' Shape tuserpoly

Set openShapeFileWorkspace = pInShpFeatCls
Write #1, "Completed openshapefile"
End Function
Public Function openShapeFileWorkspace2(Location As String, datasetname
As String) As IFeatureClass

'Open the sde workspace and return the featureclass from tUserPoly
Dim pPropset As IPropertySet
Set pPropset = New PropertySet

Dim pFact As IWorkspaceFactory
Dim pWorkspace As IWorkspace
```

```
Dim pApp As IAppDisplay
Set pApp = New AppDisplay

With pPropset
.SetProperty "DATABASE", Location
End With
Write #1, "1"
Set pFact = New ShapefileWorkspaceFactory
Write #1, "2"

Set pWorkspace = pFact.Open(pPropset, 0)
Write #1, "3"
Dim pFeatureWorkspace As IFeatureWorkspace
Write #1, "4"
Set pFeatureWorkspace = pWorkspace
Write #1, "5"

Set openShapeFileWorkspace2 =
pFeatureWorkspace.OpenFeatureClass(datasetname)
Write #1, "6"
'  Error in dll for window handle
End Function



Public Function Editshape() As IFeatureClass
'Open tuserresults for editing and return as a featureclass:
'MsgBox "Start editshape"
Dim shpfeatureclass As IFeatureClass
Set shpfeatureclass =
openShapeFileWorkspace("D:\ArcIMS\reserachdata\usershapes",
"tuserresults.shp")

' get the workspace and start editing
  Dim pDataset As IDataset
  Set pDataset = shpfeatureclass
  Dim pWorkspace As IWorkspace
  Set pWorkspace = pDataset.Workspace
  Dim pWorkspaceEdit As IWorkspaceEdit

  Set pWorkspaceEdit = pWorkspace
  pWorkspaceEdit.StartEditing True
  pWorkspaceEdit.StartEditOperation

  ' Dont need this as closeeditshape() closes the edit session
  'Dim pFeat As IFeature
  'Set pFeat = shpfeatureclass.CreateFeature
  'pFeat.Store

  'pWorkspaceEdit.StopEditOperation
  'pWorkspaceEdit.StopEditing True

  Set Editshape = shpfeatureclass
  'MsgBox "End edit shape"
```

```
End Function

Private Function newsegment(pSeg As ISegment, pPoint As IPoint, minx As
Variant, maxx As Variant, miny As Variant, maxy As Variant) As ISegment

Set pSeg = New esriCore.Line
Set pPoint = New Point
pPoint.X = minx
pPoint.Y = miny
pSeg.FromPoint = pPoint
pPoint.X = maxx
pPoint.Y = miny
pSeg.ToPoint = pPoint
Set newsegment = pSeg

End Function
```

```
################################################
#           loadpp.bas:: linepp.exe            #
################################################


Attribute VB_Name = "loadpp"

Public Sub Main()

Dim uid As Long
Dim usernum As Long
Dim tokenarray As Variant
Dim adams As IFeatureClass



Open "c:\logintersectexe.txt" For Output As #1
strValue = Command()
Write #1, strValue
Call ParseDelimitedString(strValue, tokenarray, "&")


uid = CLng(tokenarray(0))
usernum = CLng(tokenarray(1))

'Load the fclasses corresponding to:
Set adams = openShapeFileWorkspace2("D:\reserachdata\networkshapes",
"adamsnet")

Dim pp As Recordset
Dim pp_base As Database
'Dim thequerydef As DAO.QueryDef
Dim selectstring As String
Dim polypointcol As IPointCollection
Dim aadtval As Long

Set pp_base = OpenDatabase("D:\ArcIMS\reserachdata\pplookup.mdb")
selectstring = "SELECT * FROM pplookup WHERE uniqueid=" & uid

openstring = "D:\ArcIMS\Website\moresearchsite\update-" & usernum & uid
& ".xml"
Open openstring For Output As #2

Set pp = pp_base.OpenRecordset(selectstring)

'Loop thru the pp recordset and fetch each linkid:
Print #2, "<USERSHAPE>"

pp.MoveFirst
For i = 0 To pp.RecordCount

   Dim theid As Long
   Dim pFilter As IQueryFilter
   Dim adamscursor As IFeatureCursor
   Dim adamsfeature As IFeature
```

```
   Dim polyseg As IPolyline

   theid = pp.Fields("linkid").Value
   Set pFilter = New QueryFilter
   pFilter.WhereClause = "objectid=" & theid

   'Now search the corresponding link in the adams theme:
   Set adamscursor = adams.Search(pFilter, 2)
   Set adamsfeature = adamscursor.NextFeature
   While Not adamsfeature Is Nothing
        Set polyseg = adamsfeature.Shape
        Set polypointcol = polyseg
        aadtval = adamsfeature.Value(3)

        Print #2, "    <OBJ>"
         For j = 0 To (polypointcol.PointCount - 1)
            thex = "        <X>" & polypointcol.Point(j).X & "</X>"
            they = "        <Y>" & polypointcol.Point(j).Y & "</Y>"
            Print #2, thex
            Print #2, they

        Next j
        thevalue = "        <AADT>" & aadtval & "</AADT>"
        Print #2, thevalue
        Print #2, "    </OBJ>"

        Set adamsfeature = adamscursor.NextFeature

        'MsgBox polyseg.FromPoint.X

   Wend


Next i


Print #2, "</USERSHAPE>"

Close #2









End Sub
Public Function openShapeFileWorkspace2(Location As String, datasetname
As String) As IFeatureClass

'Open the sde workspace and return the featureclass from tUserPoly
Dim pPropset As IPropertySet
Set pPropset = New PropertySet
```

```
Dim pFact As IWorkspaceFactory
Dim pWorkspace As IWorkspace

Dim pApp As IAppDisplay
Set pApp = New AppDisplay

With pPropset
.SetProperty "DATABASE", Location
End With
Write #1, "1"
Set pFact = New ShapefileWorkspaceFactory
Write #1, "2"

Set pWorkspace = pFact.Open(pPropset, 0)
Write #1, "3"
Dim pFeatureWorkspace As IFeatureWorkspace
Write #1, "4"
Set pFeatureWorkspace = pWorkspace
Write #1, "5"

Set openShapeFileWorkspace2 =
pFeatureWorkspace.OpenFeatureClass(datasetname)
Write #1, "6"
'  Error in dll for window handle
End Function
```

```
################################################
#           shortestpath.bas:: shortestpath.exe #
################################################



Attribute VB_Name = "shorty"
Public ipNetworkCollection As esriCore.INetworkCollection
Public m_ipGeometricNetwork As esriCore.IGeometricNetwork
Public m_ipPoints As esriCore.IPointCollection
Public m_ipEnumNetEID_Junctions As esriCore.IEnumNetEID
Public m_ipEnumNetEID_Edges As esriCore.IEnumNetEID
Public m_ipPolyline As esriCore.IPolyline
Public m_ipPointToEID As esriCore.IPointToEID
Public m_ipMap As esriCore.IMap

Public Sub Main()
'Calculate the shortest path given two junctions with objectid's

'Fetch the necessary points for stops:
Dim startoid As Integer
Dim endstopoid As Integer
Dim mpointstart As IPoint
Dim mpointstop As IPoint
Dim uid As Integer


Open "c:\logshortyexe.txt" For Output As #1

On Error GoTo ErrorHandler

strValue = Command()
Write #1, strValue
Call ParseDelimitedString(strValue, tokenarray, "&")

Dim tuserpoly As Recordset
Dim tuserpoly_base As Database

uid = CLng(tokenarray(0))
usernum = CLng(tokenarray(1))

'Fetch the start and stopid's from the tuserpoly database:
Set tuserpoly_base =
OpenDatabase("D:\ArcIMS\reserachdata\tuserpoly.mdb")

selectstring = "SELECT * FROM Tuserpoly WHERE uniqueid=" & uid

Set tuserpoly = tuserpoly_base.OpenRecordset(selectstring)

tuserpoly.MoveFirst

startoid = tuserpoly.Fields("startid").Value
```

```
stopoid = tuserpoly.Fields("stopid").Value

tuserpoly.Close
tuserpoly_base.Close


'startoid = 25
'stopoid = 27


Dim ipNetwork As esriCore.INetwork
Dim ipFeatureClassContainer As esriCore.IFeatureClassContainer
Dim ipFeatureClassadadams As esriCore.IFeatureClass ' Adams street
polylines
Dim ipFeatureClassjunctions As esriCore.IFeatureClass ' Junctions on
network
Dim pInFeatureClass As IFeatureClass
Dim pSearchFeatureCursor As IFeatureCursor
Dim pFeature As IFeature
Dim count As Integer

'Launch the pathfinder modules:
Set ipNetworkCollection =
OpenAccessNetwork("D:\ArcIMS\reserachdata\PDOT.mdb", "Street network2")
'Fetch the featureclass with Aadt values:
'MsgBox "Opened Access"
'Count = ipNetworkCollection.GeometricNetworkCount

Set m_ipGeometricNetwork = ipNetworkCollection.GeometricNetwork(0)

'Get the Network
Set ipNetwork = m_ipGeometricNetwork.Network
Set ipFeatureClassContainer = m_ipGeometricNetwork
Set ipFeatureClassadadams = ipFeatureClassContainer.Class(0)
Set ipFeatureClassjunctions = ipFeatureClassContainer.Class(1)

Set pInFeatureClass = ipFeatureClassjunctions
Set pSearchFeatureCursor = pInFeatureClass.Search(Nothing, 2)

Set pFeature = pSearchFeatureCursor.NextFeature

'Search for the

'Dim pPoints As IMultipoint
'Set pPoints = New esriCore.Multipoint
Set m_ipPoints = New esriCore.Multipoint
Set mopointstart = New esriCore.Point
Set mopointstop = New esriCore.Point

Do While Not pFeature Is Nothing


    If (pFeature.Value(0) = startoid) Then
        'MsgBox pFeature.Value(0)
        Set mpointstart = pFeature.ShapeCopy
```

```
    End If
    If (pFeature.Value(0) = stopoid) Then
        'MsgBox pFeature.Value(0)
        Set mpointstop = pFeature.ShapeCopy

        Exit Do
    End If

 Set pFeature = pSearchFeatureCursor.NextFeature
Loop

m_ipPoints.AddPoint mpointstart
m_ipPoints.AddPoint mpointstop

Call SolvePath("TrafficVolume")
Call PathPolyLine

'Read the polyline into XML:

openstring = "D:\ArcIMS\Website\moresearchsite\update-" & usernum & uid
& ".xml"
Open openstring For Output As #2

Dim polypointcol As IPointCollection
Dim polyseg As IPolyline
Dim shortyfeature As IFeature

Print #2, "<USERSHAPE>"

Set polyseg = m_ipPolyline
Set polypointcol = polyseg

Print #2, "    <OBJ>"
For j = 0 To (polypointcol.PointCount - 1)
    thex = "        <X>" & polypointcol.Point(j).X & "</X>"
    they = "        <Y>" & polypointcol.Point(j).Y & "</Y>"
    Print #2, thex
    Print #2, they

Next j
aadtval = 10
thevalue = "        <AADT>" & aadtval & "</AADT>"
Print #2, thevalue
Print #2, "    </OBJ>"
Print #2, "</USERSHAPE>"



ErrorHandler:
Write #1, Err.Description

End Sub
```

```
Public Function OpenAccessNetwork(AccessFileName As String,
FeatureDatasetName As String) As IFeatureDataset

  Dim ipWorkspaceFactory As esriCore.IWorkspaceFactory
  Dim ipWorkspace As esriCore.IWorkspace
  Dim ipFeatureWorkspace As esriCore.IFeatureWorkspace
  'Dim ipFeatureDataset As esriCore.IFeatureDataset

  ' After this Sub exits, we'll have an INetwork interface
  ' and an IMap interface initialized for the network we'll be using.

  ' close down the last one if opened
  'CloseWorkspace

  ' open the mdb
  Set ipWorkspaceFactory = New esriCore.AccessWorkspaceFactory
  Set ipWorkspace = ipWorkspaceFactory.OpenFromFile(AccessFileName, 0)

  ' get the FeatureWorkspace
  Set ipFeatureWorkspace = ipWorkspace

  ' open the FeatureDataset
  Set OpenAccessNetwork =
ipFeatureWorkspace.OpenFeatureDataset(FeatureDatasetName)

  ' initialize Network and Map (m_ipNetwork, m_ipMap)
  'If Not InitializeNetworkAndMap(ipFeatureDataset) Then Err.Raise 0,
"OpenAccessNetwork", "Error initializing Network and Map"

End Function

Public Sub SolvePath(WeightName As String)

  Dim ipNetwork As esriCore.INetwork
  Dim ipTraceFlowSolver As esriCore.ITraceFlowSolver
  Dim ipNetSolver As esriCore.INetSolver
  Dim ipNetFlag As esriCore.INetFlag
  Dim ipaNetFlag() As esriCore.IEdgeFlag
  Dim ipEdgePoint As esriCore.IPoint
  Dim ipNetElements As esriCore.INetElements
  Dim intEdgeUserClassID As Long
  Dim intEdgeUserID As Long
  Dim intEdgeUserSubID As Long
  Dim intEdgeID As Long
  Dim ipFoundEdgePoint As esriCore.IPoint
  Dim dblEdgePercent As Double
  Dim ipNetWeight As esriCore.INetWeight
  Dim ipNetSolverWeights As esriCore.INetSolverWeights
  Dim ipNetSchema As esriCore.INetSchema
  Dim intCount As Long
  Dim i As Long
  Dim vaRes() As Variant

  ' make sure we are ready
```

```
  Debug.Assert Not m_ipPoints Is Nothing
  Debug.Assert Not m_ipGeometricNetwork Is Nothing

  'Mo:Initialize : Added from PF
  Dim ipNetworkCollection As esriCore.INetworkCollection
  Dim count As Long
  Dim ipFeatureClassContainer As esriCore.IFeatureClassContainer
  Dim ipFeatureClass As esriCore.IFeatureClass
  Dim ipGeoDataset As esriCore.IGeoDataset
  Dim ipLayer As esriCore.ILayer
  Dim ipFeatureLayer As esriCore.IFeatureLayer
  Dim ipEnvelope  As esriCore.IEnvelope, ipMaxEnvelope As
esriCore.IEnvelope
  Dim dblSearchTol As Double
  Dim dblWidth As Double, dblHeight As Double

  Set m_ipMap = New esriCore.Map

 ' Add each of the Feature Classes in this Geometric Network as a map
Layer
  Set ipFeatureClassContainer = m_ipGeometricNetwork
  count = ipFeatureClassContainer.ClassCount
  Debug.Assert count > 0   ' then Exception.Create('No (network)
feature classes found');

  For i = 0 To count - 1
      ' get the feature class
      Set ipFeatureClass = ipFeatureClassContainer.Class(i)
      ' make a layer
      Set ipFeatureLayer = New esriCore.FeatureLayer
      Set ipFeatureLayer.FeatureClass = ipFeatureClass
      ' add layer to the map
      m_ipMap.AddLayer ipFeatureLayer
  Next

  ' Calculate point snap tolerance as 1/100 of map width.
  count = m_ipMap.LayerCount
  Set ipMaxEnvelope = New esriCore.Envelope
  For i = 0 To count - 1
    Set ipLayer = m_ipMap.Layer(i)
    Set ipFeatureLayer = ipLayer
    ' get its dimensions (for setting search tolerance)
    Set ipGeoDataset = ipFeatureLayer
    Set ipEnvelope = ipGeoDataset.Extent
    ' merge with max dimensions
    ipMaxEnvelope.Union ipEnvelope
  Next

  ' finally, we can set up the IPointToEID ...
  Set m_ipPointToEID = New esriCore.PointToEID
  Set m_ipPointToEID.SourceMap = m_ipMap
  Set m_ipPointToEID.GeometricNetwork = m_ipGeometricNetwork

  ' set snap tolerance
  dblWidth = ipMaxEnvelope.Width
```

```
  dblHeight = ipMaxEnvelope.Height

  If dblWidth > dblHeight Then
    dblSearchTol = dblWidth / 100#
  Else
    dblSearchTol = dblHeight / 100#
  End If

  m_ipPointToEID.SnapTolerance = dblSearchTol

  'Mo: Initialization complete

  ' instantiate a trace flow solver
  Set ipTraceFlowSolver = New esriCore.TraceFlowSolver

  ' get the INetSolver interface
  Set ipNetSolver = ipTraceFlowSolver

  ' set the source network to solve on
  Set ipNetwork = m_ipGeometricNetwork.Network
  Set ipNetSolver.SourceNetwork = ipNetwork

  ' make edge flags from the points

  ' the INetElements interface is needed to get UserID, UserClassID,
  ' and UserSubID from an element id
  Set ipNetElements = ipNetwork

  ' get the count
  intCount = m_ipPoints.PointCount
  'MsgBox intCount
  Debug.Assert intCount > 1

  ' dimension our IEdgeFlag array
  ReDim ipaNetFlag(intCount)

  For i = 0 To intCount - 1
    ' make a new Edge Flag
    Set ipNetFlag = New esriCore.EdgeFlag
    'MsgBox m_ipPoints.Point(i).X
    Set ipEdgePoint = m_ipPoints.Point(i)
    ' look up the EID for the current point  (this will populate
intEdgeID and dblEdgePercent)
    m_ipPointToEID.GetNearestEdge ipEdgePoint, intEdgeID,
ipFoundEdgePoint, dblEdgePercent
    'MsgBox intEdgeID   ' else Point (eid) not found
    'MsgBox dblEdgePercent
    ipNetElements.QueryIDs intEdgeID, esriETEdge, intEdgeUserClassID,
intEdgeUserID, intEdgeUserSubID
    Debug.Assert (intEdgeUserClassID > 0) And (intEdgeUserID > 0)  '
else Point not found
    ipNetFlag.UserClassID = intEdgeUserClassID
    ipNetFlag.UserID = intEdgeUserID
    ipNetFlag.UserSubID = intEdgeUserSubID
    Set ipaNetFlag(i) = ipNetFlag
```

```
    'Mo:
    Set ipEdgePoint = Nothing
  Next

  ' add these edge flags
  ipTraceFlowSolver.PutEdgeOrigins intCount, ipaNetFlag(0)

  ' set the weight (cost field) to solve on

  ' get the INetSchema interface
  Set ipNetSchema = ipNetwork
  Set ipNetWeight = ipNetSchema.WeightByName(WeightName)
  Debug.Assert Not ipNetWeight Is Nothing

  ' set the weight (use the same for both directions)
  Set ipNetSolverWeights = ipTraceFlowSolver
  Set ipNetSolverWeights.FromToEdgeWeight = ipNetWeight
  Set ipNetSolverWeights.ToFromEdgeWeight = ipNetWeight

  ' initialize array for results to number of segments in result
  ReDim vaRes(intCount - 1)

  ' solve it
  ipTraceFlowSolver.FindPath esriFMConnected, esriSPObjFnMinSum,
m_ipEnumNetEID_Junctions, m_ipEnumNetEID_Edges, intCount - 1, vaRes(0)

  ' compute total cost
  m_dblPathCost = 0
  For i = LBound(vaRes) To UBound(vaRes)
    m_dblPathCost = m_dblPathCost + vaRes(i)
  Next
  'MsgBox m_dblPathCost
  'MsgBox m_ipEnumNetEID_Edges.count
  'MsgBox m_ipEnumNetEID_Junctions.count
  ' clear the last polyline result
  Set m_ipPolyline = Nothing
'MsgBox "Done SP"
End Sub

Sub PathPolyLine()

  Dim ipEIDHelper As esriCore.IEIDHelper
  Dim count As Long, i As Long
  Dim ipEIDInfo As esriCore.IEIDInfo
  Dim ipEnumEIDInfo As esriCore.IEnumEIDInfo
  Dim ipGeometry As esriCore.IGeometry
  Dim ipNewGeometryColl As esriCore.IGeometryCollection
  Dim ipSpatialReference As esriCore.ISpatialReference

  ' if the line is already computed since the last path, just return it
  'If Not m_ipPolyline Is Nothing Then
    'Set PathPolyLine = m_ipPolyline
    'Exit Sub
  'End If
```

```
  Set m_ipPolyline = New esriCore.Polyline
  Set ipNewGeometryColl = m_ipPolyline

  ' a path should be solved first
  Debug.Assert Not m_ipEnumNetEID_Edges Is Nothing

  ' make an EIDHelper object to translate edges to geometric features
  Set ipEIDHelper = New esriCore.EIDHelper
  Set ipEIDHelper.GeometricNetwork = m_ipGeometricNetwork
  Set ipSpatialReference = m_ipMap.SpatialReference
  Set ipEIDHelper.OutputSpatialReference = ipSpatialReference
  ipEIDHelper.ReturnGeometries = True

  ' get the details using the  IEIDHelper classes
  Set ipEnumEIDInfo =
ipEIDHelper.CreateEnumEIDInfo(m_ipEnumNetEID_Edges)
  count = ipEnumEIDInfo.count

  ' set the iterator to beginning
  ipEnumEIDInfo.Reset

  For i = 1 To count

    ' get the next EID and a copy of its geometry (it makes a Clone)
    Set ipEIDInfo = ipEnumEIDInfo.Next
    Set ipGeometry = ipEIDInfo.Geometry

    ipNewGeometryColl.AddGeometryCollection ipGeometry

  Next  ' EID

  ' return the merged geometry as a Polyline
  'Set PathPolyLine = m_ipPolyline

End Sub
```

B

Metadata for PDOT CD

Identification_Information:
Citation:
Citation_Information:
Originator:
Pennsylvania Department of Transportation, Bureau of Planning and Research, Geographic Information Division
Publication_Date: 20010116
Title:
PennDOT - State maintained roadways in Adams County, Pennsylvania
Edition: 2001
Publication_Information:
Publication_Place: Harrisburg, PA
Publisher: Pennsylvania Department of Transportation
Online_Linkage:
<ftp://www.pasda.psu.edu/pub/pasda/padot2001/state/padot-stroads-adams_2001.zip>
Description:
Abstract:
State-owned and maintained public roads by county within Pennsylvania. Includes fields describing pavement type, traffic volumes and other information as detailed below.
Purpose:
Public information and support for transportation planning, design and development.
Supplemental_Information: This data is intended for use at 1:24,000 or smaller scale.
Time_Period_of_Content:
Time_Period_Information:
Single_Date/Time:
Calendar_Date:
20010116
Currentness_Reference: publication date
Status:
Progress: Complete
Maintenance_and_Update_Frequency: Bi-annual
Spatial_Domain:
Bounding_Coordinates:
West_Bounding_Coordinate: -77.471603
East_Bounding_Coordinate: -76.958397
North_Bounding_Coordinate: 40.069698
South_Bounding_Coordinate:

39.719891
Keywords:
Theme:
Theme_Keyword_Thesaurus: None
Theme_Keyword: culture transportation roads
Theme_Keyword: highways hiways
Place:
Place_Keyword_Thesaurus: None
Place_Keyword: Pennsylvania
Place_Keyword:
Adams County
Access_Constraints:
None.
Use_Constraints:
The user shall indemnify, save harmless, and, if requested, defend the COMMONWEALTH, their officers, agents, and employees from and against any suits, claims, or actions for injury, death, or property damage arising out of the use of or any defect in the FILES or any accompanying documentation.<p>

The COMMONWEALTH excludes any and all implied warranties, including warranties or merchantability and fitness for a particular purpose.<p>

The COMMONWEALTH makes no warranty or representation, either express or implied, with respect to the FILES or accompanying documentation, including its quality, performance, merchantability, or fitness for a particular purpose. The FILES and documentation are provided "as is" and the USER assumes the entire risk as to its quality and performance.<p>

The COMMONWEALTH will not be liable for any direct, indirect, special, incidental, or consequential damages arising out of the use or inability to use the FILES or any accompanying documentation.<p>

The USER is granted permission to translate and add value to the FILES for the use of the FILES on its computer hardware; provided, however, that the USER annually notify the UNIVERSITY / COMMONWEALTH of any customizing or value-adding work done.<p>

Any customized or value added versions of the files will contain the following disclaimer:<p>

THIS IS NOT A PENNSYLVANIA DEPARTMENT OF TRANSPORTATION APPROVED FILE. THE DEPARTMENT OF TRANSPORTATION RETAINS THE MASTER FILES.<p>

THE USER AGREES AND UNDERSTANDS THAT IT MAY NOT FURTHER DISTRIBUTE THE FILES TO A THIRD PARTY.<p>

If you have any questions or problems, contact the ORGANIZATION where you acquired the data.<p>

Point_of_Contact:
Contact_Information:
Contact_Organization_Primary:
Contact_Organization:
Pennsylvania Department of Transportation, Bureau of Planning and Research, Geographic Information Division
Contact_Person: Bob Grugan
Contact_Address:
Address_Type: mailing address
Address: P.O. Box 3654
City: Harrisburg
State_or_Province: PA
Postal_Code: 17101-3654
Country: USA
Contact_Voice_Telephone:
(717) 772-3305
Native_Data_Set_Environment:
Microstation Design File (.DGN)

Data_Quality_Information:
Attribute_Accuracy:
Attribute_Accuracy_Report: Verified visually by comparison against source materials.
Logical_Consistency_Report: Topology exists
Completeness_Report:
Contains all officially recognized features within the Commonwealth.
Positional_Accuracy:
Horizontal_Positional_Accuracy:
Horizontal_Positional_Accuracy_Report:
Digitized from sources conforming to National Mapping Accuracy Standards for 1:24,000 scale maps.
Quantitative_Horizontal_Positional_Accuracy_Assessment:
Horizontal_Positional_Accuracy_Value: 100
Horizontal_Positional_Accuracy_Explanation:
Estimated accuracy is +/- 100 Feet.
Lineage:
Source_Information:
Source_Citation:
Citation_Information:

Originator: Geological Survey (U.S.)
Publication_Date: Unknown
Title: USGS 7.5 minute topographic maps
Geospatial_Data_Presentation_Form: map
Publication_Information:
Publication_Place: Reston, VA
Publisher:
United States Geological Survey
Source_Scale_Denominator: 24000
Type_of_Source_Media: paper
Source_Time_Period_of_Content:
Time_Period_Information:
Single_Date/Time:
Calendar_Date:
Unknown
Source_Currentness_Reference: publication date
Source_Citation_Abbreviation: topo quads
Source_Contribution: Features for digitization.
Source_Information:
Source_Citation:
Citation_Information:
Originator: Geological Survey (U.S.)
Publication_Date: 1993-1996
Title: Digital orthophoto quarter quadrangle
Geospatial_Data_Presentation_Form: map
Publication_Information:
Publication_Place: Reston, VA
Publisher:
United States Geological Survey
Source_Scale_Denominator: 12000
Type_of_Source_Media: on-line
Source_Time_Period_of_Content:
Time_Period_Information:
Range_of_Dates/Times:
Beginning_Date: 1992
Ending_Date:
1995
Source_Currentness_Reference: Image acquisition
Source_Citation_Abbreviation: orthophotos
Source_Contribution: position verification.
Process_Step:
Process_Description:
Digitized into Bentley Microstation system from most recently available 7.5 minute USGS topographic quadrangles on a stable (mylar) base. Updated information is added from construction plans and maps as this information

becomes available. Converted to ESRI Shapefile and Geomedia Warehouse formats for distribution.
Process_Date: 2000-2001
Process_Step:
Process_Description: Verified visually through overlay with orthophotos as possible.
Process_Date: 2000-2001
Process_Step:
Process_Description: Exported to ESRI Shapefile for distribution.
Process_Date:
2001

Spatial_Data_Organization_Information:
Direct_Spatial_Reference_Method: Vector

Spatial_Reference_Information:
Horizontal_Coordinate_System_Definition:
Geographic:
Latitude_Resolution: 0.0002743
Longitude_Resolution: 0.0002743
Geographic_Coordinate_Units: Decimal degrees
Geodetic_Model:
Horizontal_Datum_Name: North American Datum of 1983
Ellipsoid_Name: GRS80
Semi-major_Axis: 6378137.0
Denominator_of_Flattening_Ratio:
298.26

Entity_and_Attribute_Information:
Overview_Description:
Entity_and_Attribute_Overview:
Road entity described with the following attribution: ACCESSCTRL: Access control: 1=Full Control; 2=Partial Control; 3=No Control ADTT_CUR: Current truck annual average daily traffic CTY_CODE: County code - see below for county codes and county names CUR_AADT: Current annual average daily traffic DLY_VMT: Daily vehicle miles traveled DLYTRKVMT: Daily truck vehicle miles traveled FAC_TYPE: Facility Type Indicator: 1=One-way; 2=Two-way FEDAID_SYS: Federal Aid System - see table below for a table of Federal Aid System codes FUNC_CLS: Functional Class - see table below for a table of functional class codes JURIS: Jurisdiction (indicates maintenance responsibility for road): 1=State; 2=Turnpike; 5=Non-state federal aid LANE_CNT: Number of traffic lanes NAME: Street name NHS_IND: National Highway System codes: N=Not on National Highway System; Y=On National Highway System RT_NO_PRFX: Route prefix: I=Interstate; PA=PA Traffic Route; US=US Traffic Route RT_NO_SFFX: NA SEG_NO: Segment number SIDE_IND: Directional

indicator (right/left side of highway): 1=Even numbered segments (right side); 2=Odd numbered segments (left side) ST_RT_NO: State route number SURF_TYPE: Pavement surface type - See below for Pavement Surface Type codes. TOLL_CODE: Toll indicator: 1=Toll bridge TOT_WIDTH: Total pavement width TRAF_RT_NO: Traffic route number TRK_PCT: Truck percentage WKDYTRKCUR: Current weekday truck volume YR_BUILT: Year road was built YR_RESURF: Year road was last surfaced

The attributes below are control information generated during data export process. Please disregard.

FID: System generated feature identification integer Shape: System generated feature geometry LENGTH_FT: NA MAPID: NA MSLINK: MGE link ID number

Federal Aid System Codes:

Code Designation

1 Federal aid system open to traffic 2 Federal aid system notyet built or not open to traffic 8 Nonfederal aid open to traffic 9 Nonfederal aid not yet built or not open to traffic

Functional Class Codes:

Code Functional Class

01 Rural principal arterial - interstate 02 Rural principal arterial - other 06 Rural minor arterial 07 Rural major collector 08 Rural minor collector 09 Rural local 11 Urban principal arterial - interstate 12 Urban principal arterial - other freeways 14 Urban other principal arterial 16 Urban minor arterial 17 Urban collector 19 Urban local 99 Ramp

Pavement Surface Type Codes: CodePavement Surface Type 20 Earth - unimproved 30 Earth - graded, drained 40 Stabilized - (soil, gravel, or stone) 51 Bituminous Surface Treatment 52 Mixed Bituminous - intermediate type 53 Bitum Penetration - intermediate 61 Bituminous Pavement - high type 62 Bituminous Pavement on PCC Base 71 Plain Portland Cement Concrete Pavement 72 Reinforced Portland Cement Concrete 73 Continuously Reinforced / Prestressed 74 Concrete Over Concrete - bonded 75 Concrete Over Concrete - unbonded 76 Concrete Over Bituminous Pavement 80 Brick / Block Pavement 99 Undefined Surface Type

Entity_and_Attribute_Detail_Citation: PennDOT
Detailed_Description:

Entity_Type:
Entity_Type_Label: Roadway
Entity_Type_Definition: public roadways
Entity_Type_Definition_Source: U.S. Bureau of Census
Attribute:
Attribute_Label: CTY_CODE
Attribute_Definition: County code number
Attribute_Definition_Source: Self-evident
Attribute_Domain_Values:
Unrepresentable_Domain:
<pre> PennDOT Code County Name

01 Adams 02 Allegheny 03 Armstrong 04 Beaver 05 Bedford 06 Berks 07 Blair 08 Bradford 09 Bucks 10 Butler 11 Cambria 12 Cameron 13 Carbon 14 Centre 15 Chester 16 Clarion 17 Clearfield 18 Clinton 19 Columbia 20 Crawford 21 Cumberland 22 Dauphin 23 Delaware 24 Elk 25 Erie 26 Fayette 27 Forest 28 Franklin 29 Fulton 30 Greene 31 Huntingdon 32 Indiana 33 Jefferson 34 Juniata 35 Lackawanna 36 Lancaster 37 Lawrence 38 Lebanon 39 Lehigh 40 Luzerne 41 Lycoming 42 McKean 43 Mercer 44 Mifflin 45 Monroe 46 Montgomery 47 Montour 48 Northampton 49 Northumberland 50 Perry 51 Pike 52 Potter 53 Schuylkill 54 Snyder 55 Somerset 56 Sullivan 57 Susquehanna 58 Tioga 59 Union 60 Venango 61 Warren 62 Washington 63 Wayne 64 Westmoreland 65 Wyoming 66 York 67 Philadelphia </pre>

Distribution_Information:
Distributor:
Contact_Information:
Contact_Organization_Primary:
Contact_Organization: Pennsylvania Spatial Data Access (PASDA)
Contact_Address:
Address_Type: mailing address
Address: Land and Water Building
City: University Park
State_or_Province: PA
Postal_Code: 16802
Country: USA
Contact_Electronic_Mail_Address:
pasda@psu.edu
Resource_Description:
PennDOT roadways for Adams County
Distribution_Liability:
The USER shall indemnify, save harmless, and, if requested, defend those parties involved with the development and distribution of this data, their officers, agents, and employees from and against any suits, claims, or actions for injury, death, or

property damage arising out of the use of or any defect in the FILES or any accompanying documentation. Those parties involved with the development and distribution excluded any and all implied warranties, including warranties or merchantability and fitness for a particular purpose and makes no warranty or representation, either express or implied, with respect to the FILES or accompanying documentation, including its quality, performance, merchantability, or fitness for a particular purpose. The FILES and documentation are provided "as is" and the USER assumes the entire risk as to its quality and performance. Those parties involved with the development and distribution of this data will not be liable for any direct, indirect, special, incidental, or consequential damages arising out of the use or inability to use the FILES or any accompanying documentation.
Standard_Order_Process:
Digital_Form:
Digital_Transfer_Information:
Format_Name: ArcView Shapefile
Format_Version_Number: 3.x
File_Decompression_Technique: Zip compression
Digital_Transfer_Option:
Online_Option:
Computer_Contact_Information:
Network_Address:
Network_Resource_Name:
<ftp://www.pasda.psu.edu/pub/pasda/padot2001/state/padot-roads-adams_2001.zip>
Access_Instructions:
<http://www.pasda.psu.edu>
Fees:
none

Metadata_Reference_Information:
Metadata_Date: 20010116
Metadata_Review_Date:
20010127
Metadata_Contact:
Contact_Information:
Contact_Organization_Primary:
Contact_Organization: Pennsylvania Spatial Data Access (PASDA)
Contact_Person: Christopher Pfeiffer
Contact_Position: Metadata Coordinator
Contact_Address:
Address_Type: mailing address
Address: 141 Land and Water Building
City: University Park
State_or_Province: PA

Postal_Code: 16802
Country: USA
Contact_Voice_Telephone: 814-865-8792
Contact_Facsimile_Telephone: 814-865-3378
Contact_Electronic_Mail_Address:
cxp7@psu.edu
Metadata_Standard_Name: FGDC Content Standards for Digital Geospatial Metadata
Metadata_Standard_Version: FGDC-STD-001-1998

Generated by `mp` version 2.5.2 on Tue Feb 20 16:47:55 2001

Vita

Moinak Chatterjee was born in Calcutta, India. He received his B.S.E.E in Electrical and Computer Engineering from the University of Tennessee in 1998 and completed an M.S. in Geography with a specialty for GIS/Transportation by 2003. Moinak has worked on several projects relating to Geographical Information Systems and software development during the past several years.