ROVER
Requirements Specification Document

September 2000

ABSTRACT

ROVER is a prototype suite of software applications intended to support interaction and collaboration between mobile agents and coordinate-driven information servers. In traditional WWW servers, resources are provided to users based on a namespace that is purely topological, that is, locality plays no role. In ROVER, the namespace can be dependent upon geometry too, so servers can tailor presentation of material based on the locality. ROVER is a first step towards experimentation with protocols and coordination in such a dynamic environment. It is intended to allow us to explore and demonstrate the desirability of coordinate-driven servers, and to serve as a basis for studying the tools needed to prepare servers to be responsive in such an environment.

We gratefully acknowledge the on-going support of Institute for Advanced Computer Studies.This project is supported by Office of Naval Research under contract N000149910503. The first implementation of ROVER was performed with the help of students in C*MSC 435: Introduction to Software Engineering* in the spring semester of 2000.

1. Introduction

This Requirement Specification Document, hereafter referred to as the RSD, encompasses the requirements of a software application called ROVER. This RSD is intended to define the essential functionality of ROVER, for both the developers building it and the expected users (who are members of an ONR-funded project at the University of Maryland who are studying interconnection of software tools in mobile environments.)

ROVER allows visitors, new students, drivers, map-makers and others to communicate with local web servers in order find what information that might be available in a given locality. Users will be able to enrich those servers with their own information. Users will also be able to specify types of information for which they should receive notification in the event of change.

ROVER will be used to support activities that require location specific information. An example of this is a tourist looking for points of interest. Previously, a tourist who seeks some point of interest must examine his own map, determine where he is and then make a decision on how to navigate. With ROVER, the user will be able to inquire about not only his current whereabouts, but also about what information is available to describe these whereabouts, including directions on how to get to his target destination.

2. General Description of ROVER

The chief objective of this system is to allow the users to find out information about the area that they are in. The entire system can be generally divided into three subsystems: mobile, server and connection (client to server / client to client).

The mobile subsystem itself consists of four major components. The first is its browser. It is a software application programmed for Windows operating system. The mobile unit will be using a freeware called ICE Browser from ICEsoft Company to display the information that they receive. The other three major components of the mobile unit are a GPS, digital compass, and laser range finder. These three components are designed for the acquisition of location data. The absolute position of the current object (mobile computer) will be determined based on the information gathered from these three sources. Moreover, the location of a remote object can be targeted as well. This involves the user pointing the mobile computer towards an object and then the GPS, digital compass, and laser range finder will determine the coordinates of that object. To determine a location relative to the mobile unit, the unit's GPS coordinates, orientation, and distance to the object are required. An algorithm will be used to determine the location of the object in latitude/longitude format.

The server subsystem consists of an upgraded HTTP network and database. The upgraded network will work as follows: There will be number of servers, and each server will cover a local area. All servers run the same set of rules except for the master server and master-backup, which run several extra rules to maintain the network like determining which local server will handle a particular user. The database component can be further divided into three components: data providers, services, and consumers. These components will be used to retrieve information located on a data stores. Data providers are components that represent diverse sources of data such as SQL databases, indexed-sequential files, spreadsheets, document stores, and mail files. Providers expose data uniformly using a common abstraction called the rowset. Services are components that extend the functionality of data providers by implementing extended interfaces where not natively supported by the data store. For example, a cursor engine is a service component that can consume data from a sequential, forward-only data source to produce scrollable data. A relational query engine is an example of a service over OLE DB data that produces rowsets satisfying a Boolean predicate. Consumers are components that consume OLE DB data. Examples of consumers include high-level data access models such as ADO; business applications written in languages like Visual Basic, C++, or Java; and development tools themselves.

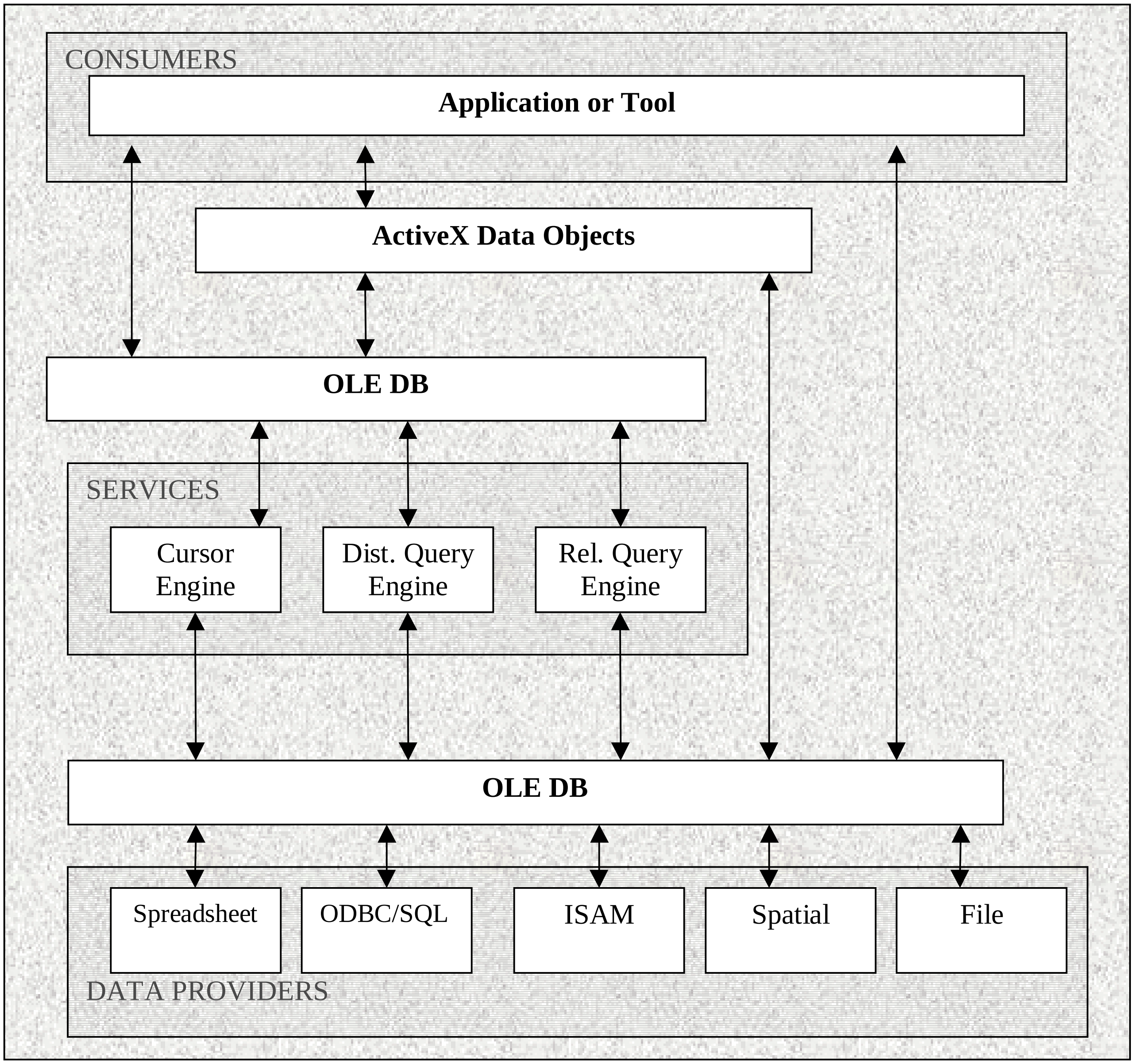

Figure 1: Database components

The connection subsystem consists of two major components (or tasks): notification and messaging. They are both software programs for different tasks. The notification component can be further divided into two smaller components, one resides on the server, and the other one resides on the mobile unit. The notification program installed on the server listens for connection requests from the mobile units and sends back a connection successful message when the mobile user is connected to the server. It will send out pings that tell the mobile units when new information is available to be downloaded. The notification program installed on the user-side will be able to send information to the server and interpret responses that are sent back from the server. Generally speaking, the notification component detects and maintains a connection between the server and mobile units. The messaging component is a layer that resides on the mobile unit. It sends text messages to other mobile units via the connection between servers and mobile units.

The major components of the Rover system are:

- Browser
- GPS
- Digital Compass
- Laser Range Finder
- Upgraded HTTP Network
- Data Providers (Database)
- Services (Database)
- Consumers (Database)
- Notification Program
- Messaging Program

Relation of System Components

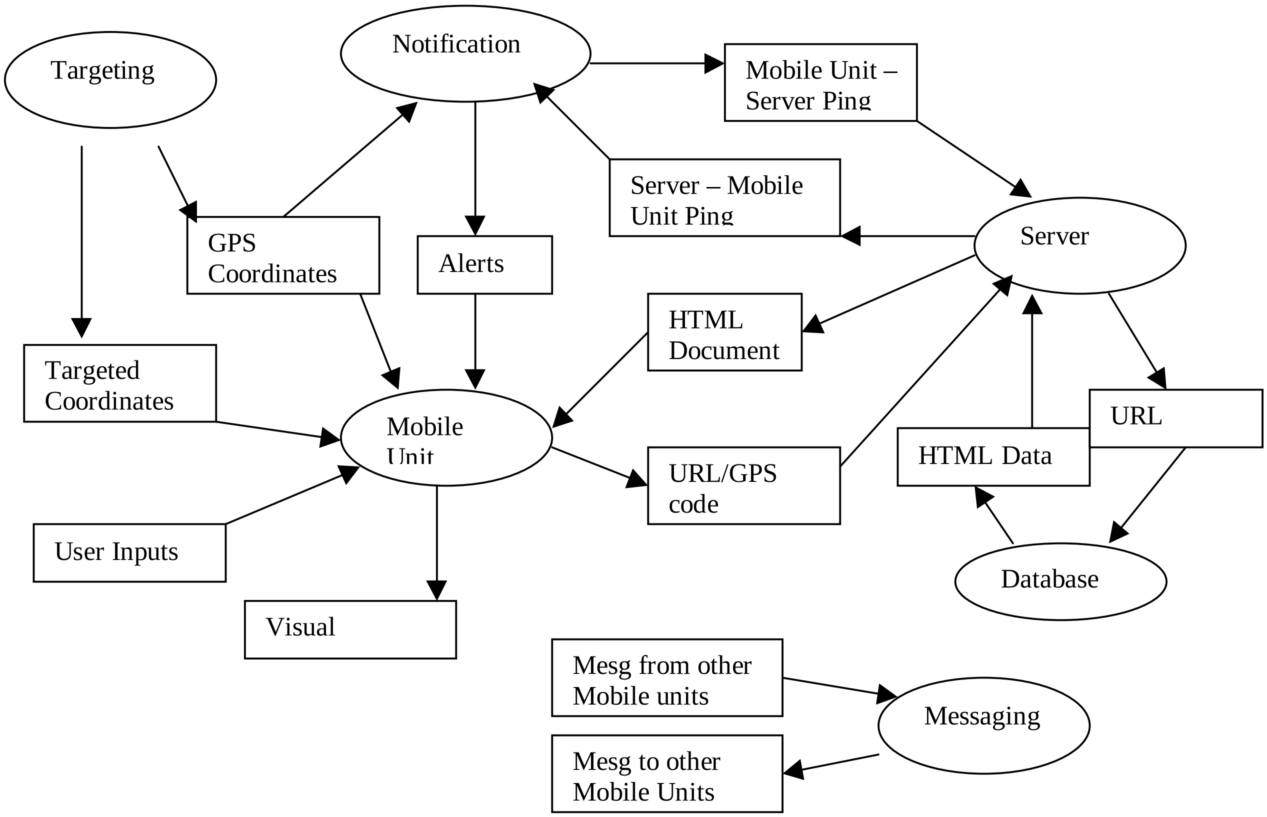

Figure 2: Relation Between System Components

The targeting task in Figure 2 consists of a GPS, a laser range finder, and a digital compass component. These three components detect either the current GPS coordinate of the mobile unit or the targeted coordinate of a remote object. The data containing GPS values is then sent to the mobile unit where browser resides on. Sometimes users would like to manually enter a URL instead of using the laser range finder. In this case, a URL is inputted to the mobile unit. The mobile unit then sends either URL or GPS value to the server where database components reside on (see Figure 1) via upgraded HTTP network. The URL/GPS code is now considered as query that will be sent to the database in order to retrieve corresponding data in HTML format. The HTML data then will be sent to the server, and eventually to the mobile unit where the residing browser component will interpret the data and display the result to the user. Other tasks of the Rover system include the notification and messaging. The notification functions like “ping” between mobile units and servers. It gets the GPS coordinate values from the mobile units (mobile unit gets the GPS value from the targeting task) and passes it to the server network so that the servers know the current locations of the mobile units. Based on the GPS values passed to the servers, the servers will send data back the mobile units in regards to if any refresh of the information is necessary. (Each server is only responsible for a local area, hence if switching server, the information on the mobile units should be refreshed.) The messaging task functions like “instant message”. It delivers text-based messages from one mobile unit to another mobile unit providing both units are on the same local network area.

Principle Interfaces

- Mobile Unit – Server
  The browser resides on the mobile unit requests information from the server, and the server answers the request.

- GPS, Laser Range Finder, Digital Compass – Browser
  GPS, laser range finder, and digital compass are used to provide GPS coordinate location to the browser. The browser provides the ability to launch these devices.

- Server – Database
  Server passes the query (URL/GPS code) to the database. ADO is the data object that will facilitate data exchange between the server and the OLE DB. OLE DB is the interface that will access data from the various data stores. ODBC is the interface between various relational databases and OLE DB (ODBC is part of the OLE DB specification). Result of the query is then passed back to the server.

- Mobile Unit – Notification Program
  The part of the notification program that resides on the mobile unit keeps getting the current GPS location of the mobile unit in order to send the GPS code to the server. Also the notification program receives the data sent back from the server to determine if there is a need to refresh the content shown on browser.

- Server – Notification Program
  The part of the notification program that resides on the server keeps receiving data (GPS coordinate) sent from the mobile unit. The data is in a format of 4 bytes of IP address that will be a mobile unit ID plus 2 floating points for longitudinal and latitudinal position. After server analyzes the GPS value, the notification program residing on the server side will send data back to the mobile unit. The data being sent back is in a format of 4 bytes of IP address that will be the server ID plus 1 byte of server code that can be something meaning “no change” or “refresh”.

- Mobile Unit – Messaging Program
  The mobile unit sends text-based data to the messaging program that resides on the mobile unit. The messaging program also interfaces with the browser component on the mobile unit so that the browser can display the text-based messages.

- Server – Messaging Program
  The server receives the text-based messages from one mobile unit and sends it to another mobile unit as long as both units are on the same local network area.

- Notification Program – Messaging Program
  The messaging program interfaces with the notification program in order to detect which local server the messages should be sent to from the mobile unit based on the current location of the mobile unit.

General Constraints

The GPS and wireless reception impacts the ability of the mobile unit to receive information that is truly location sensitive. And the most accurate result from a civilian GPS device is within 8 feet (based on our test). The laser range finder (at this stage, Yardage Pro 500 is used) has a maximum judging distance of 500 ft. The size and weight of the input and output devices should also be limited. Since it's a portable device, the weight should be as light as possible and the components should be as small as possible. Users can only see a small portion of the information, however, if the output device is small; and if the input device (keyboard) is too small, users will have hard time to type. Moreover, a small hard drive and memory might not be enough for information storage.

3. Specific Requirements

3.1 Functional Requirements

REQUIREMENT #1: PROVIDING CLIENT-SERVER IDS

Introduction
The master server will contain a list of all users on the network. The master server will also contain a list of servers and their coverage areas. The master server must be able to provide client-server IDs to the notification on the mobile unit.

Inputs
The notification program sends the logon request

Processing
The master server will respond to the notification with the address of the server local to the user

Outputs
The local server IDs will be sent to the notification layer.

REQUIREMENT #2: CLIENT SERVERS

Introduction
All web page links will be CGI Post format. All web pages will contain a hidden variable, or variables, which will always be of the same name. The client Server will handle queries from browser and notification.

Inputs
The inputs are query-packets from Mobile browser: 4 byte of IP address is mobile unit ID, 2 floating point for longitude and latitude position, and query-packets from notification tool.

Processing
The client server will separate types of queries and send queries to the database (for more details, see Database requirements)

Outputs

The server will send response packet to mobile unit: 4 byte of IP address is server ID, 1 byte is the server code.

REQUIREMENT #3: CONNECTING EVENT

Introduction
The Notification will pass the local server ID to the Browser-Server function (for more details, see the Master Server requirement). The browser will establish the connection with the server (for more details, see Browser-Server requirement). The notification must be able to notify the user if the mobile unit has established a successful connection to the server.

Processing
The notification program will call the Internet protocol to initialize and get the local server ID. Browser will then establish a connection with server based on the server ID received from Notification. If Notification cannot obtain any local server, it should continue to send connection requests or begin “probing/pinging” until it can. When the connection between mobile unit and server is established, Notification will notify mobile unit.

Outputs
Notification on the server will send acknowledge to mobile unit.

Constraints
The constraints of this function are limited to the abilities of the modem: wave-LAN card and protocol’s reliability.

REQUIREMENT #4: DISCONNECTING EVENTS

Introduction
If the user leaves the current server’s service area, *a master server* (see Server component) will calculate the new server’s ID (based on current location) that the user should be directed to. If the user leaves the range of all servers (no server to forward to), the server will disconnect the user if they leave the area.

Inputs
The inputs are packets that include information about the mobile users current location

Processing
A master server searches and matches the nearest server based on the user’s global coordinates provided by the notification program (see Notification component). If there is no server in the area, the server will disconnect the user.

REQUIREMENT #5: OBTAINING GPS COORDINATES

Introduction
The targeting function must be able to obtain the global coordinates of the user's position. This information must be obtained from the GPS device that is incorporated in the mobile unit.

Inputs
The GPS device, upon query for the user's present position, will return the coordinates of the mobile unit's location. The longitude and latitude will be returned to the targeting software. (At the time of this writing, it has yet to be determined whether the device will directly feed the information to the software through a serial port or if the information will have to be placed in the device manually).

Processing
The information obtained from the GPS device will have to be read in to the targeting software as a string object. No further processing of this data will be necessary at this stage

Outputs
The function returns the ROVER mobile unit's global coordinates.

Constraints
The constraints of this function are limited to the abilities of the GPS device. The device has a margin of error of about 8 feet.

REQUIREMENT #6: BROWSER INTERFACE

Introduction
The mobile unit must be able to provide a browser interface whereby the user may navigate and view HTML documents from a remote server. Because the system requires additional information to be coupled with the URL data that is provided to the server additional interfaces are necessary for managing the interactions with other components of the system. These interfaces are the notification process interface, the server interface, the messaging interface, and the targeting interface.

Inputs
Inputs to the browser include HTML documents, GPS coordinates of the target, alert signals from the notification process, and user's input in the form of selecting buttons or keying in URL address from the user interface, or by selecting hypertext from the document display window.

Processing
HTML documents that are received by the browser are displayed in the document display window (monitor). User inputs result in a URL address. GPS coordinates are coupled

with user selected URL's to form a URL/GPS code which is then transmitted to the server. Alert signals trigger the browser to perform an automated refresh. These processes are described in more detail in the requirements for the interfaces listed in Requirement 2 to 6.

Outputs

Outputs to the browser include formatted HTML documents displayed to a screen within a document view window and the URL/GPS code that is transmitted to the server.

Constraints

The browser initializes other processes within the ROVER system, including the notification, messaging and targeting processes. The constraints of browser function are limited to the abilities of the mobile hardware: PC/104, 64 MB RAM and Intel 200MHz processor

REQUIREMENT #7: BROWSER-SERVER INTERFACE

Introduction

The Browser-Server interface is responsible for constructing the URL/GPS code from a user-selected URL and either local or targeted GPS coordinates. The Browser-Server Interface defines the interaction between the browser application and the remote server. Two types of information are being handled in this interface. The browser supplies URL and GPS coordinate information to the server and the server provides HTML documents to the browser.

Inputs

The Browser-Server interface uses a number of inputs, GPS coordinates, a URL, and a HTML document

Processing

The browser obtains current GPS coordinates by means of the Browser-Targeting Interface. The Browser-User Interface provides a URL. Both the GPS coordinates and the URL are used to form the URL/GPS code. The URL/GPS code is then transmitted to the server. Once that data has been cross-referenced to HTML information, the remote server transmits an HTML document back to the browser

Outputs

The Browser-Server Interface outputs a URL/GPS code for the server, and provides an HTML document.

Constraints

The constraints of Browser-Server are limited to the abilities of the Wave LAN card (See Reference for more information about WaveLAN card) and of servers cover areas.

REQUIREMENT #8: BROWSER-NOTIFICATION INTERFACE

Introduction

The Browser-Notification must be able to interact with the notification in the Browser (mobile unit). The browser-notification interface defines the interaction between the browser application and the notification process. Specifically this interface involves the alerts transmitted from the notification process to the browser when the user has moved to GPS coordinates that map to new information in the database

Inputs

The only input is the signal sent from the notification function to the browser application.

Processing

When the notification program on the server side determines that new information is available from the server, it sends a signal to the notification on the mobile unit. Then the notification program on the mobile unit will alert the browser. When the browser receives a signal from the notification process it refreshes, and re-requests the same URL with a new GPS coordinate via the Browser-Server interface

Outputs

The Browser-Notification Interface initiates a process and so does not provide outputs for other interfaces or system components

Constraints

The browser should be initialized before the notification process can signal it. The browser should save old information before refreshing for new information

REQUIREMENT #9: BROWSER-TARGETING INTERFACE

Introduction

The Browser-Targeting function must be able to interact with the Targeting device. The Browser-Targeting Interface defines the interaction between the browser application and the targeting processes. The Browser-Targeting Interface references those processes that access the GPS device in the mobile unit as well as the laser range finder to ultimately provide coordinate that is either local to the user or targeted.

Inputs

The Browser-Targeting Interface receives a flag indicating whether the GPS coordinates to be obtained are either local coordinates or targeted coordinates from the Targeting device.

Outputs

The Browser-Targeting Interface provides a GPS coordinates which is either local or targeted.

Constraints
The constraints are the accuracy of GPS receiver and object location calculation. These constraints are described in more detail in the requirements for the targeting listed in Requirement 17-20

REQUIREMENT #10: BROWSER-MESSAGING INTERFACE

Introduction
The Browser-Messaging Interface provides a means by which the Messaging process can be launched using the same framework as the browser. The messaging process and the browser process are not functionally linked

Inputs
The only input to the Browser-Messaging Interface is initialization

Processing
The browser launches a separate messaging application.

Output
There is no output to the Browser-Messaging Interface

Performance
Performance of browser does not affect the messaging interface because they are two independent applications.

REQUIREMENT #11: BROWSER-USER INTERFACE

Introduction
The Browser-User Interface defines the interaction between the browser application and the User. This interface includes options that the user has available, such as navigation options, and options for accessing targeted GPS information or messaging applications.

Inputs
Inputs include all inputs provided by the user. This includes selected navigation options such as home, forward, back, or refresh. Inputs also include selected hypertext. The Browser-User Interface also takes an HTML document as an input from other components in the browser

Processing
The Browser-User Interface performs navigation commands as follows:

home: Prompts the browser to request a default URL with the local GPS coordinates through the Browser-Server Interface.
back: Goes back to the previous page that the browser has loaded. This command does not result in server access because pages saved in the back button are cached.
forward: When using back to navigate, the browser allows the user to return to the page that the browser had loaded after the one that he is viewing. Similar to back, the forward button accesses cached pages.
refresh: Prompts the browser to reload the same URL that is currently loaded. This will cause the browser to send the current URL to the Browser-Server interface to acquire a new HTML document to be displayed.
HTML pages are received via the Browser-Server Interface. The Browser-User Interface formats these for the document display window to be presented to the user.

Outputs
The outputs for the Browser-User Interface are the visual interface that is displayed to a screen. Outputs also include the URLs that result from user selected actions to be used by the Browser-Server Interface

Constraints
User input that is assumed to be a standard keyboard for now lacks of touch screen technology for the system.

REQUIREMENT #12: RECEIVING INPUT STRINGS

Introduction
The GIS Database Interface must be able to receive the input string from the server, to process it accordingly, and to generate desired results from a variety of data stores.

Inputs
An input string provided by the server will contain location coordinates and possibly, information pertaining to the surrounding areas of those coordinates. An example of a query will be a request of, among others, information such as name of the building or the nearest way to a specific location, etc.

<Coordinates>	<Text>	<Text>	<Text>	<Text>
Location	Query-Item 1	Query-Item 2	Query-Item 3 …..	Query-Item n-1

Processing
The input string will be broken into smaller pieces so that proper queries will be generated

Output
The outputs are broken-up strings that contain data on which queries will be generated for database accessing

Constraints
The input string forwarded by the server does not arrive correctly due to network errors or receiving process's problems

REQUIREMENT #13: ACCESSING DATA FROM DATABASE

Introduction
The GIS Interface must be able to make a connection to a database regardless of its platform and to submit queries in order to retrieve/post information, and to have results or appropriate messages sent back to server.

Inputs
The input here is a bi-product of query processing. At this point, the queries that are generated from the broken-up input strings are passed on to relational or non-relational databases.

Processing
The GIS database interface must be able to determine what kind of database it needs to access, either the local DBMS (Microsoft Access) or non-local DBMS. At this point, one of the followings will occur.

a) Accessing Relational Databases
The ODBC application calls ODBC functions to submit SQL statements and retrieves results. The driver manager loads drivers on behalf of an application. The driver processes ODBC function calls, submits SQL requests to a specific data source, and returns results. This is needed since each DBMS typically has its own call-level programming interface through which it communicates with applications.

b) Accessing Non-Relational Databases
The OLE DB provides a set of interfaces for data access over the enterprise's network. These interfaces take care of data-consumers and database provider functions. An ADO

object then provides the data exposed by the OLE DB to the interface application. Although we will be using Visual Basic, the ADO object is language-neutral.

For example, SQL Server provides an interface called DB-Library while Oracle Server provides the Oracle call interface. Therefore, any application that wishes to access more than one DBMS must be able to translate request and transfer data into each interface it needs to access.

Outputs
The result generated by one of the two DBMS will be forwarded to server.

Constraints
The constraints of this requirement are the failure to communicate with a DBMS and the inability to generate a result from the user query.

REQUIREMENT#14: ENRICHING THE DATA STORE

Introduction
A user from time to time will encounter an area or a site that local or non-local DBMS does not have information about. In the event of this happening, the GIS Database Interface must have the capability to allow the server to transmit data pertaining to that new location and have it incorporated to the local DBMS or non-local DBMS, for future user encounters.

Inputs
The original parsed input string with an appropriate flag indicates that the query is processed as a posting of data to the local DBMDS or non-local DBMS.

Processing
The primary objective is trying to map the new data to the local DBMS. If the data pertains more closely to the data stored in a non-local DBMS, then the appropriate insertion constraints on the non-local DBMS are verified and the data can be posted.

Outputs
The output will consist of a confirmation message to indicate if the attempt to post the new data is successful or not.

Constraints
The primary constraint of requirement is that non-local DBMS does not permit to post data onto their DBMS. To alleviate this realistic constraint, it will be necessary that all attempts should be performed on the local DBMS, where permissions are more likely to exist.

REQUIREMENT #15: QUERY SUCCESS/FAILURE MESSAGE

Introduction
The GIS Database Interface must be able to produce a message that confirms a successful or unsuccessful update on a DBMS or indicates that no result on a particular query on certain areas or sites is generated due to absence of data from both local and non-local DBMSs.

Inputs
The input string will have been parsed and being processed.

Processing
The GIS Database interface must exhaustively attempt to generate a query result or to insert data into a DBMS pertaining to a new site. After trying to access the DBMS a number of times (this number is pre-determined by the system), the error message will be generated.

Outputs
There are two types of error messages: one pertaining to the failure to get a result for a given query of type get-info, and one pertaining to the succeeded/failed attempt to update a DBMS with new information.

REQUIREMENT #16: RETURNING QUERY'S RESULT FROM DATABASE

Introduction
The GIS Database Interface must be able to generate a result and send it back to the user, via the server, in a formatted way. A pre-formatted HTML page captures the results based on the user queries and is sent back to the user with the requested data

Processing/Outputs
Form structures embedded in HTML code will be filled with the query result and will be sent to the server, so that it can be forwarded to the client.

Constraints
The query may generate a large number of pages

REQUIREMENT #17: OBTAINING THE MOBILE UNIT'S BEARING

Introduction
The targeting module must be able to correctly calculate distances to remote object. To do this, it needs the mobile unit's bearings. The reasons for determining a user's bearings

are as follows. The device must know the proper hemisphere in which the mobile unit is located.

Inputs
The targeting software will obtain the mobile unit's bearings from a digital compass. The compass will give the targeting software function the direction in which the unit is facing, as a string. (EX: N, NE, SW, etc).

Processing
The information obtained from the digital compass will have to be read in as a string.

Outputs
This function returns the ROVER mobile unit's directional heading. The output from this function will be used by other functions of the targeting module and the output will be sent to the Mobile unit when a query for the user's position is made.

Constraints
The constraints of this function are limited to the abilities of the digital compass. The device's margin of error is yet to be determined.

REQUIREMENT #18: OBTAINING THE DISTANCE FROM THE LASER RANGE FINDER

Introduction
The targeting module must be able to correctly calculate the remote object's coordinates in Longitude and Latitude. To do this, it needs the distance of a remote object. Based on the mobile unit's bearings, obtained from the digital compass, the distance to the remote object is either added or subtracted from the mobile unit's coordinates to obtain the coordinates of the remote object.

Inputs
The targeting software will obtain the mobile unit's distance to a remote object from a laser range finder. The range finder will give our targeting software function the distance to an object in feet. This information will be read into the software as a string.

Processing
The information obtained from the laser range finder will have to be read in as a string.

Outputs
This function returns the distance of the distant object (in feet). The output from this function will be used by other functions of the targeting module and the output will be sent to the Mobile unit when a query about an object is made.

Constraints

The constraints of this function are limited to the abilities of the laser range finder. The device's margin of error is yet to be determined.

REQUIREMENT #19: CALCULATING COORDINATES OF TARGETED OBJECT

Introduction
The targeting unit must be able to obtain the coordinates of the object being targeted. This will be achieved with the aid of the distance functionality for the module.

Inputs
Apart from the querying information from the unit, this function requires input from other targeting module functions. The information needed is retrieving current GPS coordinates, correct heading to determine the relative latitude and longitude, and distance from the range finder.

Processing
The value retrieved from the laser range finder will be calculated onto the coordinates received from the GPS device. To add or subtract from the GPS coordinates is dependent upon the heading reading. Once the correct rule (add or subtract) is applied, the coordinates of the targeted object will be given.

Output
This function returns the coordinates of the object targeted by the Laser Range Finder. The output will be sent to the Mobile unit when a query for an object's position is made.

Constraints
The constraints of this function are limited to the abilities of the GPS, Digital Compass, and Laser Range Finder devices. The GPS device has a margin of error of about 8 feet.

REQUIREMENT #20: UPLOADING FAVORITE AND BUDDY LISTS FROM NOTIFICATION

Introduction
The notification program must be able to upload data from favorite and buddy functions to the server.

Inputs
Notification gets Favorite and Buddy lists from Browser and Messaging

Processing
Favorite and Buddy lists are uploaded to Notification on the server. Server will store and inform other server components to allocate space to accommodate the new mobile user.

Constraints
The constraints of this program are limited to the abilities of server's and mobile unit's storage.

REQUIREMENT #21: NOTIFYING USER OUT OF COVERED AREA

Introduction
When the mobile user is close to the edge of the service area, Notification on Server must be able to send out: "About to leave current server area" notification to the mobile user. If the user leaves the range of all servers (no server to forward to), the program on the server will send out an "About to disconnect from total coverage area" notification.

Inputs
Notification on Server will obtain user's global coordinates from Notification on Browser

Processing
Once the connection between the mobile units and the server is established, Notification on the mobile unit will, at a constant rate, ping the server it is currently connected to make sure it is still connected to the server. Notification on the mobile unit will send out pings that include information about the mobile user's current location. If the mobile user nears the edge, an appropriate message will be sent.

Outputs
Notification on the mobile unit will generate an appropriate message to alert the mobile user. This message will be forward to browser.

Constraints
The constraints of this program are the accuracy of GPS receiver and object location calculation

REQUIREMENT #22: NOTIFYING FAVORITE EVENT

Introduction
This function will allow mobile users to select their favorite items that they would like to receive notifications of if the server has information about them. Notification must be able to notify the user when there is a match in the server's information and the user "favorite" information.

Inputs

The user interface in Browser will send to the notification program a “favorite” list. The list includes information about a “hot spot” in the area or information on a restaurant that the user is fond of.

Processing
The favorite list is uploaded to Server and is processed. The server’s notification program will cross reference “favorites” to information located in the server’s database and send out notification.

Outputs
Appropriated messages will be sent to Browser

REQUIREMENT #23: NOTIFYING BUDDY EVENT

Introduction
The Buddy Notification function will allow users to enter their friends’ information. This function must be able to alert the user when a “buddy” is currently in or out of the area.

Inputs
Users enter the names of friends, family members, or colleagues into mobile units.

Processing:
This information is then uploaded to the server’s notification program. All “buddies” are placed into a database and cross-referenced with each other by the server notification program.

Outputs
The notification program of the server will send out an appropriate message to inform user that a “buddy” is currently in or out of his area

REQUIREMENT #24: MANAGING AND UPDATING WEB PAGES

Introduction
The notification program must be able to manage and update Web pages to Browser.

Inputs
Sever will send a header packet with the type of information that it is sending (hard or soft refresh).

Processing:
The pages are sent directly to the browser, which will handle the information appropriately.

Outputs

The pages are sent to Browser

REQUIREMENT #25: SENDING MESSAGE

Introduction
The messaging program must be able to package the text message and send it to the notification layer with IP address of the recipient

Inputs
The user generates the contents of the text message in the user browser interface and clicks on the send message button to transmit the message.

Processing/Outputs
The message is packaged and sent to the notification layer to be transmitted to the recipient

Constraints
The constraint of this function is packet size, which is the max length of the message.

REQUIREMENT #26: REPORTING UNDELIVERED MESSAGE

Introduction
The notification program must be able to report the failure to Browser if a message cannot be delivered.

Processing/Outputs
The messaging translates the signal and sends an appropriate message "message was not sent…" to the user.

REQUIREMENT #27: RECEIVING MESSAGE

Introduction
The messaging program must be able to unpack the text message and send it to the user.

Inputs
The inputs are messages for user from the notification program

Processing/Outputs
The messages are displayed on the user interface

Constraints
The constraint of this function is packet size, which is the maximum length of the message.

3.2 External Interface Requirements

3.2.1 User Interface Requirements

The browser that resides on the mobile unit is decided to be ICE Browser, a free program developed by ICEsoft Company. For demonstration purpose, Microsoft Internet Explorer 5 is used in this document because the ICE Browser is not yet available.

3.2.1.1. Login Screen

The login screen allows the users to enter personal password. The idea behind this screen is to prevent illegal usage.

Figure 3: Login Screen

- **Enter Password Field**
  The Enter Password Field allows the users to enter personal password in order to preventing illegal usage.

- **Enter Button**
  The Enter Button takes the user to the Main Menu (figure 4).

- **Reset Button**
  The Reset Button deletes the password string entered by user at once.

3.2.1.2. Main Menu Screen

The main menu screen has two frames. The top frame displays current information including date, time, and current location in coordinate format. This frame will be updated periodically to reflect the most current information. The exact time period is not determined at current stage. The bottom frame is the main menu allowing users to input.

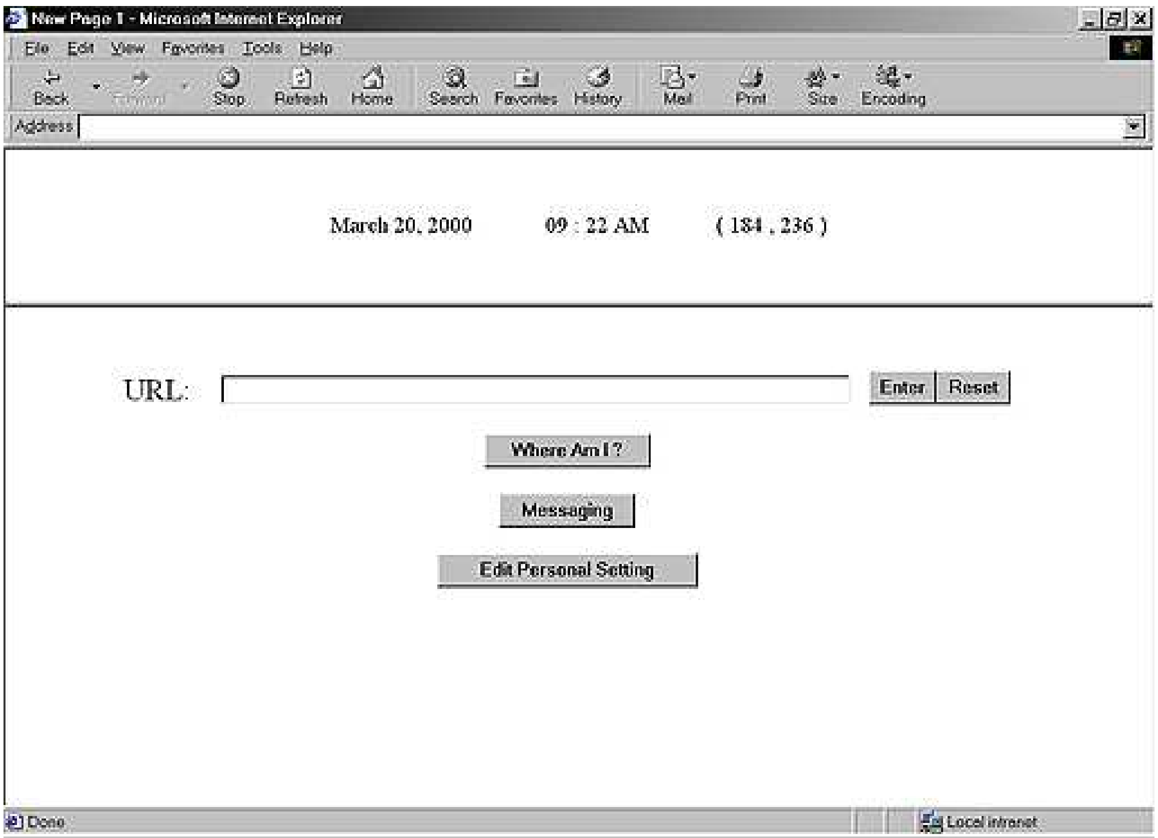

Figure 4: Main Menu Screen

- **URL Field**
  This field allows users to enter the URL they are looking for. After clicking Enter Button, the result page will show (figure 5). The Reset Button simply deletes the whole string users enter in the URL Field.

- **Where Am I ? Button**
  This button will take users to the result page (figure 6) where users will find his/her current location in both coordinate and address formats.

- **Messaging Button**
  This button will take users to the messaging screen (figure 7) where users can read the current local events and can send messages to his/her friends who are on the same local network.

- **Edit Personal Setting**
  This button will take users to the personal setting screen (figure 8) where users can select his/her interests.

3.2.1.3. URL Search Result Screen

This screen has two frames, top frame displays the current information while bottom frame displays the search result where the query is entered on the main menu (figure 4).

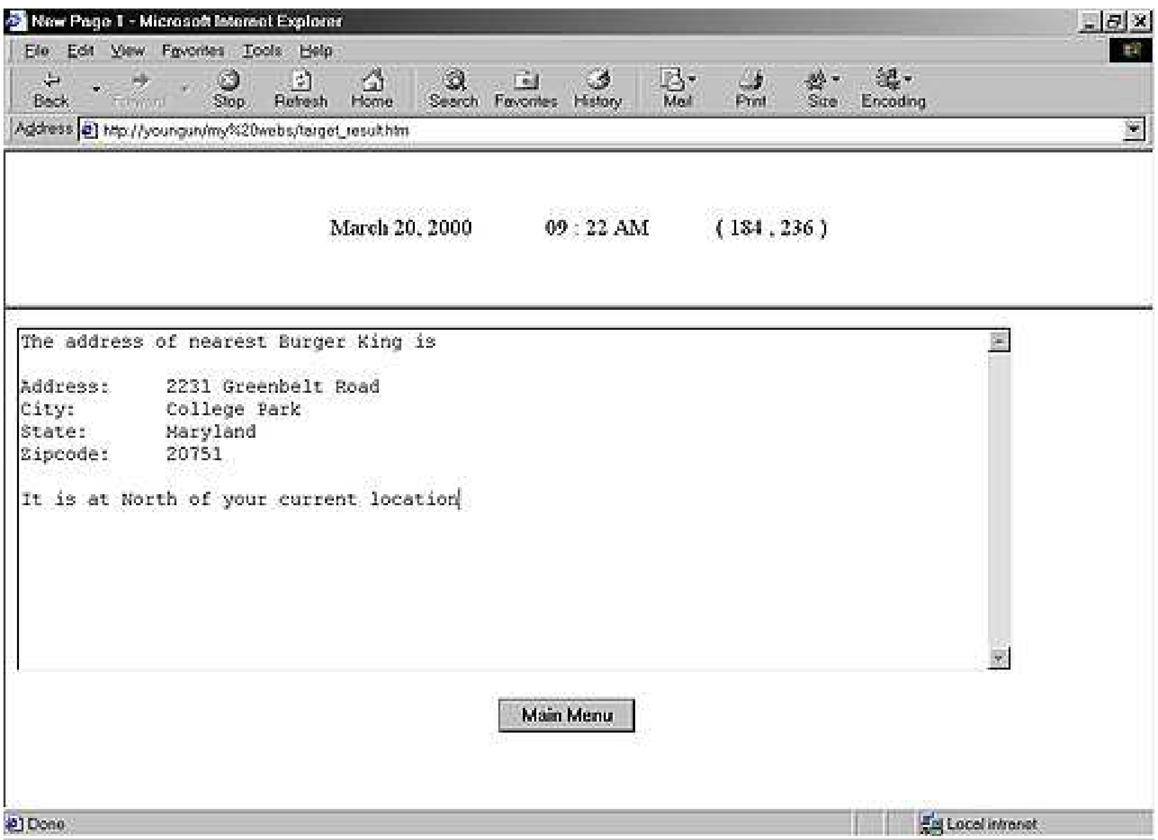

Figure 5: URL Search Result Screen

- **URL Search Result Field**
  This field displays the search result that the query is entered from the main menu URL field (figure 4). For example, string "burgerking.com" is entered from the main menu, then the URL Search Result Field here will return the address of the

nearest Burger King store. Notice that the search function will only be performed in our own database, not World Wide Web.

- **Main Menu Button**
  This button brings the users back to the main menu screen (figure 4).

3.2.1.4. Where Am I? Result Screen

This screen again has two frames, top frame still displays the most current information while bottom frame displays the search result on the current location of the Rover mobile unit.

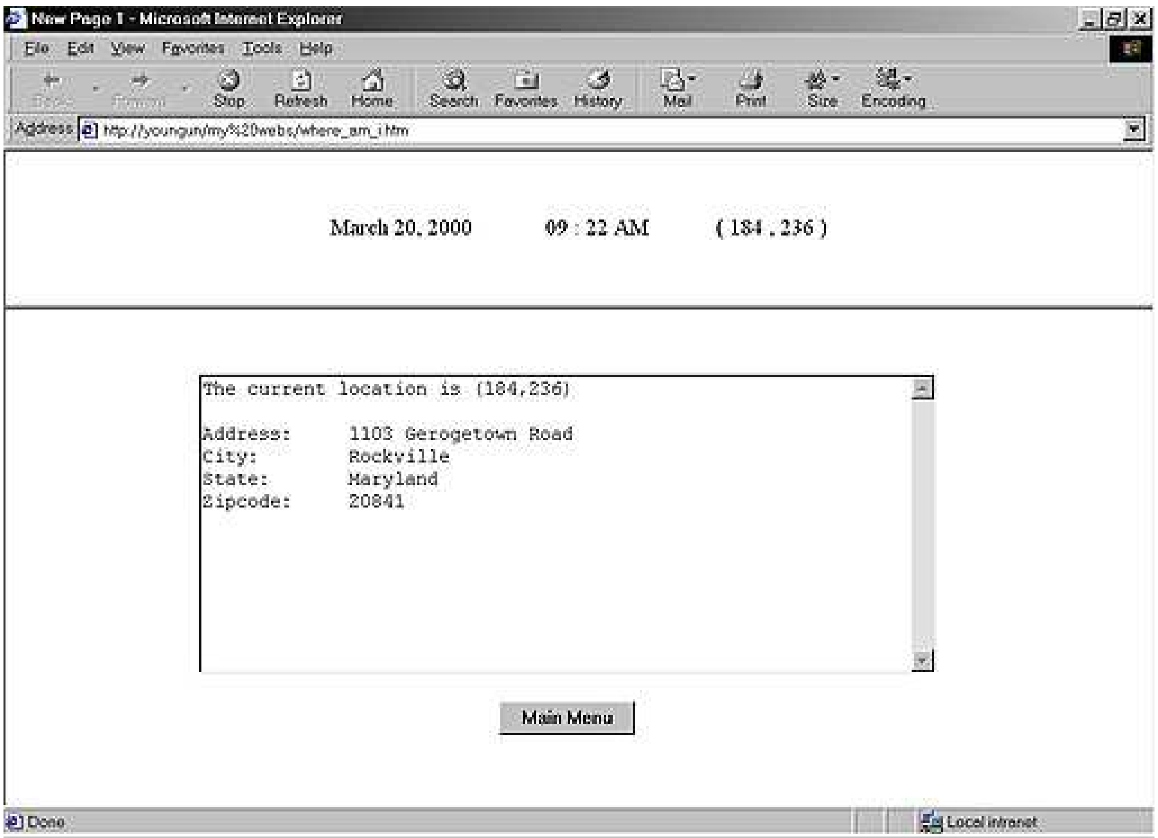

Figure 6: Where Am I Result Screen

- **Search Result Field**
  This field displays the current location of the mobile unit in both coordinate and address format. The coordinate in this field will be the same as the coordinate on the top frame. The address displayed is the nearest possible address of the current position. For example, assume a user is standing in the middle of a street, then the address displayed in this field will be the address of the nearest building.

- **Main Menu Button**
  This button brings the users back to the main menu screen (figure 4).

3.2.1.5. Messaging Screen

This screen has three frames. The top frame still displays the current information. The lower-left frame displays the events of the current local network area. The lower-right frame displays a list of user's friends who are currently on the same local network area as the user.

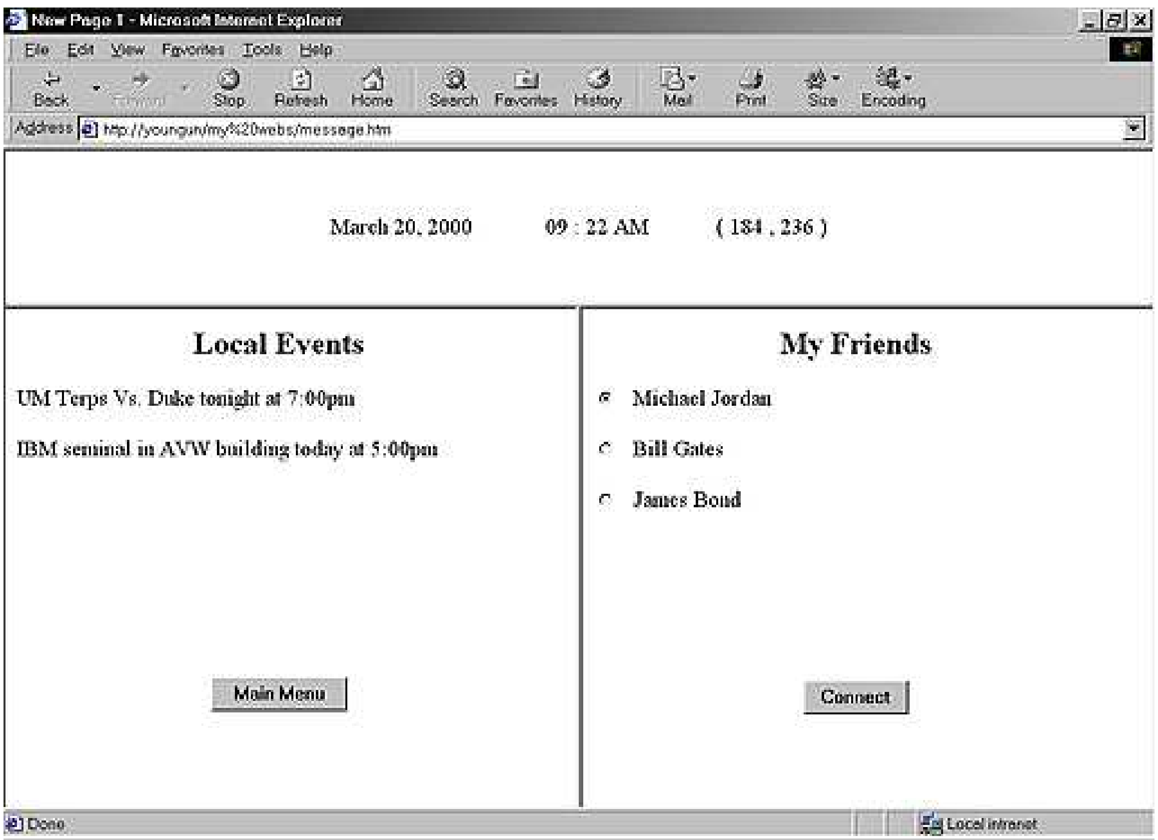

Figure 7: Messaging Screen

- **Local Events Frame**
  The local events frame displays the events that match the interests of the user on the local network area. The user's interests are pre-defined at the personal setting screen (figure 8).

- **Main Menu Button**
  This button brings users back to the main menu screen (figure 4).

- **My Friends Frame**
  This frame displays a list of people who match the names pre-defined by the user and are at the same local network area as the user. A user can select a person from this list to send messages.

- **Connect Button**
  The connect button will make connection between the user's mobile unit and the other person's mobile unit. Messages can be sent to each other after connection is established. The users will be transferred to the Chat Screen (figure 10).

3.2.1.6. Personal Setting Screen

This screen allows the users to select their interests and modify personal friends list. The settings here will be used as a mask to match the search result passed back from the database.

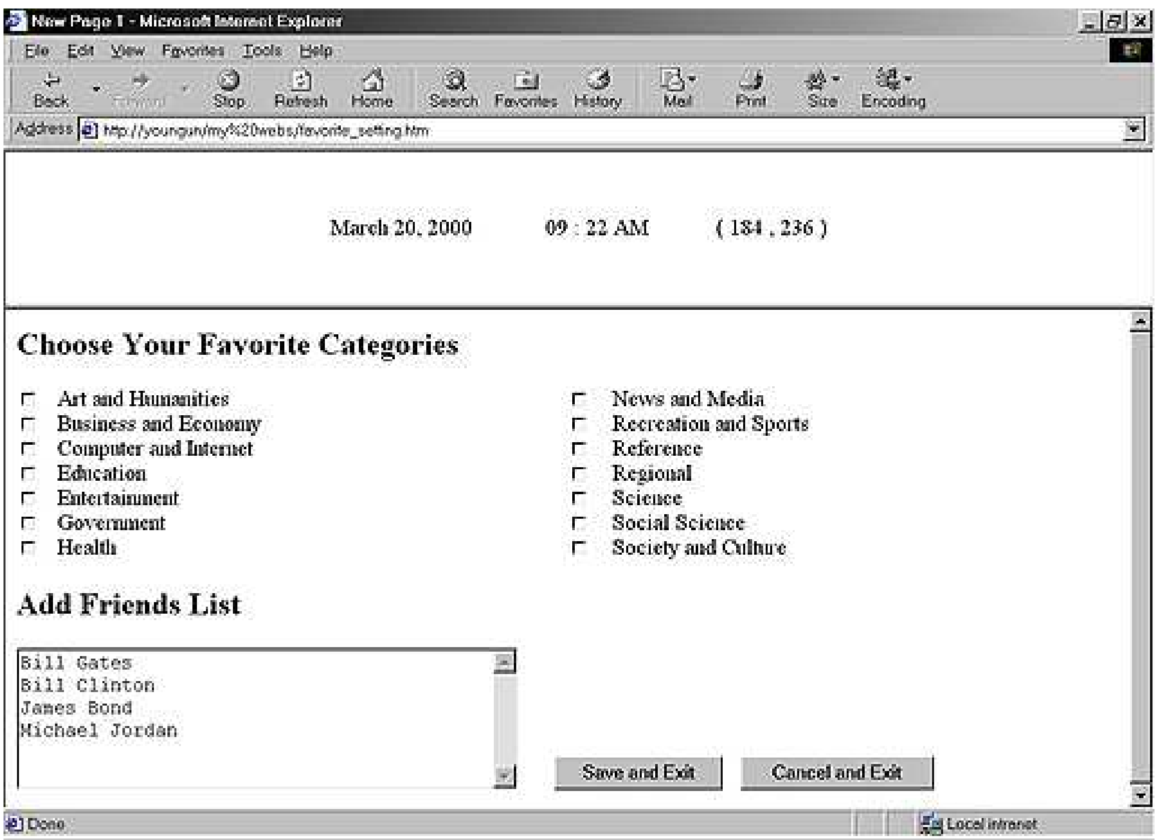

Figure8: Personal Setting Screen

- **Choose Your Favorite Categories Field**
  There are many categories in this field for the users to choose. For example, if a user selects sports, then all the sports related events would be displayed on the messaging screen (figure 7) in the future.

- **Add Friends List Field**
  This field allows the users to enter a list of people whom the users want them to be shown on the messaging screen (figure 7) when they enters the same local network area as the users.
  The format of friends list has not been determined at this stage. For demonstration purpose, people's name is used. The format will be justified for the final product.

- **Save and Exit Button**
  Save the settings and exit to the main menu screen (figure 4).

- **Cancel and Exit Button**
  Do not save the settings and exit to the main menu screen.

3.2.1.7. Targeting Result Screen

The Targeting Result Screen is very similar to the URL Search Result Screen (figure 5). The Targeting Result Screen is shows the results if the users use the Laser Range Finder to find information of a particular building (location).

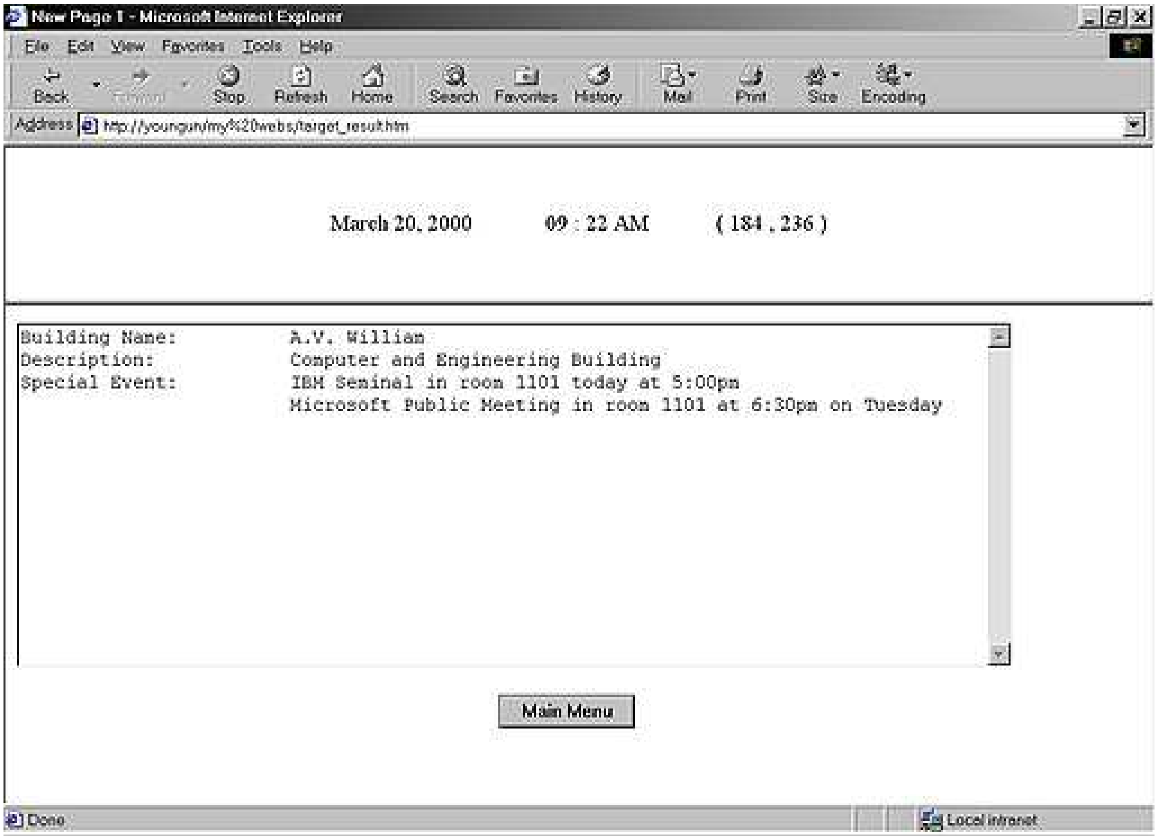

Figure 9: Targeting Result Screen

- **Targeting Result Field**
  This field displays information of a particular location. It will display the location name, a general description of the location, and special events of the location, if any.
  At this point, the exact type of information is not finalized. Name, description, and events are used for the prototype. A decision on the information type will be determined for the final product.

- **Main Menu Button**
  This button takes the users back to the main menu screen (figure 4).

3.2.1.8. Chat Screen

This screen again has two frames. Top frame displays the same information as the previous screens. The bottom frame functions as a "chat room". Users can type and read data via this screen, and the data will be transferred between the user's mobile unit and the other person's mobile unit as long as both units are on the same local network. When the end sends message to the users, a pop notification screen will appear.

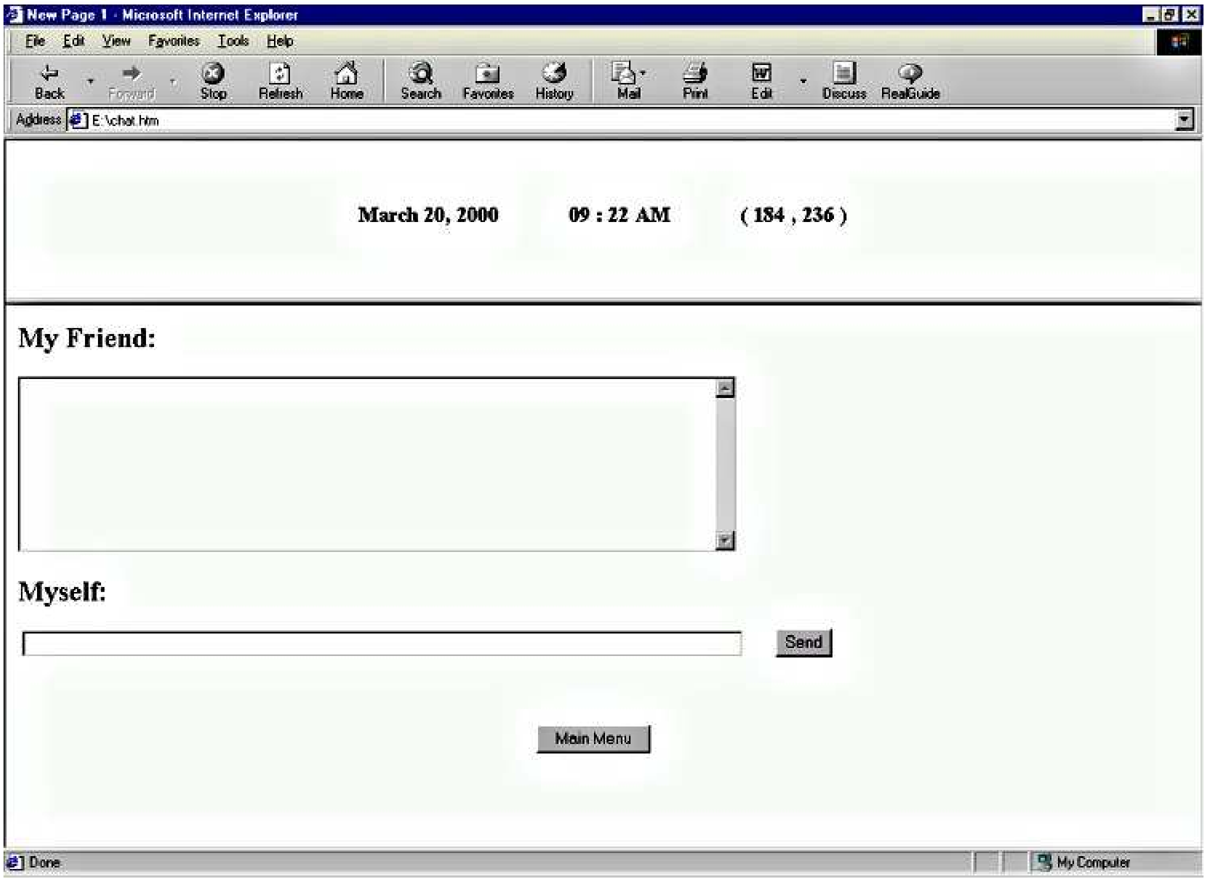

Figure 10: Chat Screen

- **My Friend Field**
  This field displays the data the other end types.

- **Myself Field**
  This field is where users can input data.

- **Send Button**
  This button sends the string in the Myself Field to the other end after being clicked.

- **Main Menu Button**
  This button brings users back to the main menu (figure 4).

3.2.2 Hardware Requirements

- Mini VGA display for the mobile unit (if it this hardware can not be obtained a desktop VGA monitor will be used instead)
- Ports (Unsure if port will be serial or parallel)
- For full implementation of this product, multiple servers are recommended. However, the prototyping stage can be implemented with one server.
- Mobile computer with PC/104 motherboard
- GPS device, laser range finder, and digital compass
- WavLan card for wireless communication between the mobile unit and the LAN
- Input device (mouse and keyboard - for prototype)
- All hardware devices should be small and portable as possible

3.2.3 Software Requirements

- Apache Web Server
- Microsoft Windows 95/98/NT Operating system
- Microsoft Visual Basic
- JDK 1.2
- GNU project C++ Compiler
- Vypress messaging software
- Microsoft Universal Data Access
- ICE Browser
- CGI scripting

3.3 Miscellaneous Requirements

3.3.1 Maintenance Requirements

- The mobile unit ID and server ID must be sent in every communication over the WaveLan card. Each message should be checked to make sure this is correct based on the connection established or message should be discarded.
    - Periodic updates for the browser to add appropriate features
    - The application must allow updates in an event that new features are added to ROVER. These integrated updates will allow for testing of both the new feature and the ability of the whole unit to work along with the new feature.
    - The GPS and Laser Range-finder and Digital Compass units should be easily connected or disconnected so that repairs and replacements can be easily made.

3.3.2 Security Requirements

- The lines of communication should remain secure because there are no user interfaces at either end node.
- A user must enter ID and password to access a mobile unit. A mobile unit will automatically shut down if the idle time exceeds 30 minutes.

3.3.3 Cost Requirements

The hardware has been obtained free of charge from various colleagues of Professor Purtilo. The messaging software (Vypress) and browser (from ICEsoft Company) can also be obtain free of charge. All other software that is needed to build ROVER will be implemented with a cost of time and effort spent by the developers.

4. Testing / Demonstration

4.1 Scavenger

4.1.1 Scavenger Overview

The Scavenger application is a multi-player game that extends the idea of the traditional scavenger hunt. In a scavenger hunt, each player is given a clue to decipher that leads him to the location of another clue. This clue, once deciphered, leads to yet another location and clue, and the process continues until the end of the chain of clues is reached. In some variations, the player is required to gather some item at each location along the way, but in all cases, the first player to successfully reach the end of the chain of clues is the winner.

The Scavenger application allows this idea to be extended and executed more easily. Each player is given a mobile unit, equipped with a browser built to interface with the class technology. At the start location of the hunt, the mobile unit uses the technology to report the players' locations to a server also equipped with the class technology, and the server returns browsable content that is the equivalent of a clue. This content could be some text, photograph, or other multimedia information that provides a hint to the location of the next clue.

The problem is to develop a useful test to verify the functionality of the class technology. The Scavenger application will need to test both the individual components and the unit as a whole. In addition, the application should require only basic functionality of the class technology, but also include optional tests for advanced functions should they be implemented in the final product.

4.1.2 Description of the Scavenger Application

Application Functionality

Once the players head off in search of the location/clue hinted at, the game begins. When a player approaches the location hinted at by the current clue, his mobile unit sends information about his location to a server and a new clue is displayed on his mobile unit, featuring browsable content that hints at the location of the next clue. This chain of search-and-discover continues in the same fashion as a traditional scavenger hunt until some player receives the final clue, reaches the final location, and wins the game.

A traditional scavenger hunt typically has some person responsible for setting up the game: he invents the clues, hides them at the proper locations, and starts the players off by giving them the first clue. The analogous role in regards to the Scavenger application is the administrator. This person authors the browsable multimedia clues, associates each with a location, and configures the server to display these clues when the players reach the associated locations.

Product Perspective

The application tests the basic functionality of the class technology by verifying communication between the database, server and mobile unit, but also includes features that take full advantage of additional functional components. Players can use the messaging component of the class technology to communicate remotely via the mobile units. The notification tools of the class technology indicate to players when other players come within their vicinity. The targeting component of the class technology allows players to remotely target a location and verify whether it is a location that should be explored to find the next clue.

Product Interfaces

The application is meant as a demonstration of the class technology and relies upon that technology for a great deal of its underlying functionality. The application will not be tied to a particular platform or operating system, except insofar as the class technology has need of a particular environment.

Because of the integral nature of the class technology to the application, the requirements for the class technology should be considered requirements of the application as well, except where superceded by explicit requirements given in this document.

User Characteristics

There are two classes of user, indicated above, for the application that is explicitly defined here.

- **Player**: The role of the player is to participate in a scavenger hunt using the application. The player interacts with the application via the software on the mobile units. The exact nature of this software is described elsewhere, but includes a browser that can report the player's location to a server and display location-specific content using the class technology, as well as support further functions of the class technology such as messaging, active notification, and remote targeting.
- **Administrator**: The role of the administrator is to initialize the application with data and put it into a state ready to support a game of scavenger hunt among the players. The administrator interacts with the application indirectly by preparing content that will be viewed by the players during a game and preparing this content for distribution by the server(s) by adding it to the database of content that the server(s) draw upon during operation. The administrator need not have an active role during the game.

General Constraints

The general requirement of the application is that it demonstrates the function of each part of the class technology. Therefore, the application is constrained only by the ability of the class technology to function. The following outline indicates what parts of the application demonstrate the function of each component of the class technology and the necessity of each component for the application. It is listed by component.

- **Servers**: The operation of the application is fundamentally dependent on the function of servers that use the class technology to transmit content based upon the location of the client. The ability of players to receive appropriate clues when they near important locations during the game demonstrates the proper function of the servers. The demonstration of this component is inseparably linked with the mobile unit and database components. The application requires this component in order to function.
- **Mobile Units**: The ability of players to interact with servers by using mobile units to report their location and to browse content transmitted by the servers demonstrates the proper function of the mobile units. The application requires this component in order to function.
- **GIS Database Interface**: The ability of the administrator to prepare content for the game and store it so that it may be retrieved by servers and of players to see the prepared content retrieved by the servers from the database demonstrate the proper function of this component. The application requires this component in order to function.
- **Notification Tools**: The ability of players to be notified, without requiring active input, when other players are in their vicinity and when they approach the location of a clue demonstrates the proper function of the notification tools component. This component is not required for the application to function. If this component does not work, the ability of players to know other players in their vicinity can be removed, and the notification of players when they approach clue locations can be made to require an active input.
- **Messaging Tools**: The ability of players to transmit free-text messages to each other demonstrates the proper function of the messaging tools. This component is not required by the application. If this component is not functional, the free-text messaging feature can be removed from the application.
- **Targeting**: The ability of players to remotely indicate a location and receive from the servers an indication of the merit of its exploration demonstrates the proper function of this component. This component is not required by the application.

If this component is not functional, the ability of players to remotely target a location can be removed from the application.

4.1.3 Scavenger Application Requirements

Gameplay Requirements

The following requirements describe the function of the application while being used by a player during the course of a game of scavenger hunt.

Clue Display

The application displays to each player the player's current clue. The current clue is defined as the most recent clue received by the player that has not been solved. Clues are solved by going to the hinted location and receiving the next clue. Because clues are received in a linear fashion, each player has exactly one current clue at any given time.

Clues are browsable content prepared by the administrator and are displayed by the application in the manner implied by the use of that term, meaning that they may, but are not required to, contain textual or multimedia data, hyperlinks to other documents, multiple pages of information, or additional player interaction. The exact manner of presentation of a clue is left to the discretion of the developer and may depend on the implementation of the mobile unit component of the class technology.

Clue Discovery

The application detects when a player has solved a clue by traveling to the hinted location and gives that player the next clue. Depending on the functionality of the notification tools component of the class technology, this process occurs either passively or actively. If the notification tools are functional, the clue is revealed to the player when he reaches the key location, requiring no input. In the other place, the player actively queries the server via input to the mobile unit about whether he has reached the key location, and the server responds by revealing the new clue, if appropriate.

The application only reveals clues in the correct linear order determined by the administrator. If a player is searching for clue three and reaches the location of clue five, the application will not reveal clue five or indicate that a clue is present. Similarly, if a player is searching for clue three and goes back to the location of clue two, the application will not redisplay clue two to the player.

Game Completion

When a player wins the game by completing the hunt, the application indicates to all players that the game has been completed and displays post-game browsable content prepared by the administrator. The exact manner of indication and display is left to the discretion of the developer.

Depending on the subdivision of functionality, this requirement may be dependent upon the notification tools or messaging tools component of the class technology. This requirement may be modified or omitted if a necessary component is not functional.

Player Proximity

The application indicates to each player all other players within close proximity to him. "Close proximity" is defined to be within a certain straight-line range determined by the administrator before the game. The exact manner of indication (audio/visual, textual/graphical, etc.) is left to the discretion of the developer.

This requirement is dependent upon the notification tools component of the class technology and may be modified or omitted based upon the functionality of that component.

Free-Text Messaging

The application allows players to send textual messages to each other during a game and displays to each player the messages he receives. The method of message entry and manner of message display is left to the discretion of the developer.

This requirement, and certain of its parameters, such as which players a given player may message at a given time, is dependent upon the function of the messaging tools component of the class technology. This requirement may be modified or omitted based upon that component's functionality.

Remote Targeting

The application allows players to query the value of exploring remote locations and responds to those queries by indicating whether the remote location is "hot" or "cold", that is, whether it is closer to the next clue or farther than the player's current location. At the discretion of the administrator, this feature may be turned off for a game. The exact manner of indication is left to the discretion of the developer.

This requirement is dependent upon the targeting component of the class technology. This requirement may be modified or omitted based upon the functionality of that component.

Administrative Requirements

The following requirements describe the function of the application when being used by the administrator to prepare a scavenger hunt for play.

Clue Creation

The administrator is able to create clues of browsable content and enter them into the content database from which the servers will later draw to relay the clues to players during a game. The exact structure of the browsable content and its entry into the database are dependent upon the GIS database interface component of the class technology.

Location Association

The administrator is able to (and must) associate with each clue a location. A player reaching this location results in the clue being revealed to the player by the application, provided that the clue is part of the hunt in which the player is participating and the clue is the next appropriate clue for the player.

Clue Order

The administrator is able to assign each clue to at least one scavenger hunt and for each hunt, of which the clue is a part, is able to (and must) assign to the clue it's position in the order of clues in the hunt.

Game Parameters

The administrator is able to prepare non-clue data and define parameters for games and hunts. Such data and parameters includes but is not limited to: post-game content displayed when a player has won a game, the definition of "close proximity" for purposes of player proximity notification, and enabling/disabling the remote targeting feature for a game.

4.2 Campus Tour

The Campus Tour model for which testing will evaluate is designed to provide location-based information about various Campus features. The user can query information from the server. The Campus Tour model is designed to take advantage of all of the functionalities that ROVER provides. The major difference between this demonstration and the others is that the data will be compiled from actual campus information.

Most major colleges and universities offer prospective students the option to tour their campuses and thoroughly investigate what the school has to offer. The Campus Tour Demonstration caters the ROVER unit to conduct such tours in a timely and efficient manner. It does this by providing location-based information about campus such as points of interest, emergency personnel, and the major buildings. It offers more flexibility, not restricting the students to a group, and allowing the students to tour at their own pace.

The user could have input a query into the Rover mobile unit and have the result displayed on the output device. All the information that the user needs will be instantly accessible.

The ROVER unit also comes with an instant messaging service for communication between ROVER users and a built-in distance measuring system, and the user has everything they need to learn all about campus easily and comfortably.

Before ROVER can be tested as a completed product, each component must be tested separately. Prototype testing should proceed in the following order: Module, Integration, Functional, Performance, Acceptance, Installation, and Regression. Then, once all components are in working order, ROVER can be tested as a unit. Using the UMCP campus as the touring site, the testing team will select a portion of campus and sample data about that part of campus. Members of the testing team will assess ROVER's abilities by walking around within the specified area

The ROVER unit must ensure that all components both independently and collectively behave as outlined in the Requirements Specification Document. The major focus of the testing, however, will be to ensure that the server, notification, and the mobile units work in harmony since they are the crux of the system. The unit must be able to query location-based information from the server and the server must be able to dynamically send new information to the unit.

Data will be input into the Database for the Server to query and update. During testing, the input will be dynamically entered as the user walks around a specific piece of campus. This input will include the changed GPS coordinates or an instant message.

All information is output to the Browser component. Testing must ensure that the information displayed by the browser is accurate according to the current location. Or, if an instant message is being sent or the laser finder is in use, the Browser component should display it in a readable format.