Major Project

On

Implementing Location Based Services for Mobile Devices equipped with GPS Capabilities

By

Muchhadia Darshana M.
(06MCE010)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
INSTITUTE OF TECHNOLOGY
NIRMA UNIVERSITY OF SCIENCE & TECHNOLOGY
Ahmedabad 382481
May 2008

Major Project

On

Implementing Location Based Services for Mobile Devices equipped with GPS Capabilities

Submitted in partial fulfillment of the requirements

For the degree of

Master of Technology in Computer Science & Engineering

By

Muchhadia Darshana M.
(06MCE010)

Under Guidance of

Dr. S.N. Pradhan
Mr. Deepak Kakkdia

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
INSTITUTE OF TECHNOLOGY
NIRMA UNIVERSITY OF SCIENCE & TECHNOLOGY
Ahmedabad 382481

May 2008

This is to certify that Dissertation entitled

Implementing Location Based Services for Mobile Devices equipped with GPS Capabilities

Submitted by

Muchhadia Darshana

has been accepted toward fulfillment of the requirement

For the degree of

Master of Technology in Computer Science & Engineering

Dr. S. N. Pradhan	Prof. D. J. Patel
Professor	Head of The Department

Prof. A. B. Patel

Director, Institute of Technology

CERTIFICATE

This is to certify that the Major Project – I entitled "**Implementing Location Based Services for Mobile Devices equipped with GPS Capabilities**" submitted by Ms. Muchhadia Darshana M. (06MCE010), towards the partial fulfillment of the requirements for the degree of **Master of Technology in Computer Science and Engineering** of **Nirma University of Science and Technology, Ahmedabad** is the record of work carried out by her under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Dr. S. N. Pradhan
Professor, Department of Computer Engineering,
Institute of Technology,
Nirma University, Ahmedabad.

Date:

ACKNOWLEDGEMENTS

It give me great pleasure in expressing thanks and profound gratitude to **Dr. S. N. Pradhan**, Professor, Department of Computer Science and Engineering, Institute of Technology, Nirma University, Ahmedabad for his valuable guidance and continual encouragement throughout the Major Project work. I heartily thankful to him for his time to time suggestion and the clarity of the concepts of the topic that helped me a lot during this dissertation.

I like to give my special thanks to **Mr. Deepak Kakdia**, CTO & Vice President, DataCentrica, Ahmedabad for his continual kind words of encouragement, suggestion for model and motivation throughout the Project.

I would like to thank **Prof. D. J. Patel,** HOD (Computer Science & Engineering), Institute of Technology, Nirma University of Science & Technology, Ahmedabad for providing me the facilities in the Nirma campus.

I extend my sincere thanks to my colleagues for their support in my work. I would like to express my gratitude towards my family members who have always been my source of inspiration and motivation.

Muchhadia Darshana M

(06MCE010)

ABSTRACT

Mobile phones and the Internet have revolutionized the communication and lifestyle of people. Mobile devices allow people to access the Internet where ever they are and whenever they want. Mobile location-based system is complex and requires the integration of many different technology components into one seamless system. Increasing capabilities of mobile handheld devices and Personal Digital Assistants (PDAs) in term of processor, power, memory and devices like Global Positioning System (GPS) in new set of applications can be developed. Location based services (LBS) are information services, accessible with mobile devices through the mobile network and utilizes the ability to make use of the location of the mobile device. The location service application is aimed at providing mobile users the map of their current position based on the positioning data from the simulator. Location Based service is one of the promising application and variety of application can be design for users. This project is based on one such application called “Buddy- Tracker”. This application is based on kind of client-server application where server maintains the location of its users provide the appropriate alarm on map.

Buddy- Tracker is a location based service which is community driven. The purpose of the Buddy- Tracker is to create a platform where a set of users can share information about geographical positions between each other. The service consists of a server system and a client application. The server system stores the position data of the client’s location. The client application is developed for mobile phones equipped with GPS capabilities. The client side application is simulated on J2ME wireless toolkit 2.2. This application has identified appropriate communication mechanism for delivering location data to the new generation of smart phones such as Symbian, J2ME and others.

1. INTRODUCTION

Location Based Services or LBS, refers to the wireless services provided to the subscriber based on his current location. The position can be known by getting it from mobile phone network, or from another positioning service, such as GPS. Mobile location-based system is complex and requires the integration of many different technology components into one seamless system. This paper describes on the design and implementation of the three location-based system components, which are the location server simulator, location service application and device client application. The location server simulator processes the location request and returns simulated positioning data to the location service application. The location service application is aimed at providing mobile users the map of their current position based on the positioning data from the simulator. It senses what the display capabilities the requesting device has and delivers the map accordingly [1]. The last component is a small software entity that runs on Java phones such as Motorola A835, Nokia 7610, and SonyEricssonP900. This component is called device client application, issues a location request that starts the data flow and displays the map or text messages on mobile phones.

1.1 Project Background

There are few functions the average cell phone cannot perform these days. From instant messaging to capturing live video of your family, the average cell phone is not much different from a personal computer. But what good are all these features if you can't find your phone? Not much good at all. Using this application user will be able to tell where his phone is at any time by not only seeing its current location, but also by tracking its path on a satellite map of the current

surrounding region! Utilizing powerful new technologies, this application is fast, accurate and reliable. Don't be left behind in the rapid shuffle of meaningless technologies. This application has been commissioned to build a system of tracking a cellular telephone utilizing a variety of technologies. This application receives updates from a GPS (Global Positioning System) system notifying it of the cell phone's location, which are then be uploaded to a server database that in turn will stores this information for access from a home computer. This document delivering the scripts interfacing with the database, any code dealing with the database itself, the source for the ASP.NET server pages, as well as the Java J2ME cell phone applet and any documentation which was written during the period of this development.

1.2 What is Buddy-Tracker?

Buddy-Tracker is meant to share your position in a community, or among friends and family. And user can of course see them if they choose to send their position. The goal of the application is to allow consumers to track the locations of their cell phones. On the large scale, the software consists of two separate modules: a J2ME application utilizing the JSR 179 standard to keep track of the cell phone and a database component accessible by web page which stores information sent by the cell phone [3]. This application allows the user to:

- Log onto an ordinary static web page which will interface with a database containing all the information about the cell phone's location.
- Obtain a sorted listing of said locations presented in easy to understand format.

- Visually present the data utilizing the Google Maps API.

1.3 Motivation for Project

This project focuses on creating an application package that demonstrates the power of current GPS cell phone technology. While the technology shows great potential, the typical consumer has no knowledge of how such a system may benefit him or her or even of the technology's existence. Here objective was to write a user friendly system which will convey the potential benefits of modern cell phone technology to potential users. This application demonstrates these benefits by designing an easy to use, friendly, portable cell phone tracking system. This application believes that this innovation, coupled with the user-friendly interface and clearly inherent convenience perfectly exhibits the capabilities of currently available GPS technology for hand held devices.

1.4 Software Architecture and Technical constraints

The cell phone applet targets the Location API [3] and CLDC J2ME's profile, as it is currently the only Location API offered by Java. The applet is compiled and tested on a Windows XP machine. The database utilizes MySQL and is interfaced with via JAVA scripts. The database was selected due to the availability of documentation, ease of use, and reputation for stability. The corresponding scripting language was chosen as it exhibits great compatibility with MySQL. The database is located on a Windows machine running an IIS (Internet Information Services) server, as these applications are well documented as well as free. The user interfaces with ASP.NET pages which in turn call the aforementioned JAVA scripts. Finally, for a visual representation of the data, the Google Maps API is utilized. The Google Maps API was

chosen because of its open-ended nature and ability to provide the exact mapping tools necessary.

1.5 Usage Constraints

This application is heavily constrained by the range of the positioning service, which currently extends only into metropolitan areas. The personal computer aspect is simplified as much as possible, with the data presented with as little or as much complexity as the user desires. Unfortunately, due to a variety of problems, this application was unable to test the cell phone application on an actual handset, though effort was placed into making the phone application as generic as possible to ensure ease of portability. For a summary of the aforementioned problems see the “Implementation Issues” section of this document.

1.6 Security Considerations

Location capability poses service providers with the challenge of responsibly handling consumers' personal privacy. This is particularly important with 'tracking services' that continuously monitor and log user's location, a location-tracking service for the elderly and children, and other live tracking services using technologies like the GpsGate Server.

1.7 Input

Upon executing the cell phone application, the user is presented with a menu of initial options, which petition the user for his or her username, password, phone number and update frequency. The user then enters the main application menu where the current status of the tracking application is displayed and options for starting and stopping

tracking are provided. The input to the database comes from the cell phone itself in the form of geographical coordinates and is generated periodically without petitioning the user.

1.8 Output

The output takes on two forms. First, the cell phone application has a status screen which notifies the user of the current state of the tracking device. Secondly, output is provided via the website the user interfaces with. The output consists of a listing of locations, compiled by time, which the user's cell phone has been in. Thirdly, this application provides a graphical representation of the above data shown as a map with indicators as to the phone's position.

1.9 Alternatives to Buddy- Tracker

Until recently, there were no products that offered similar functionalities to Buddy- Tracker. Most GPS programs for the cell phone, such as Motorola’s ViaMoto service offered only one-way communication between the cell phone and satellite, not storing the current location anywhere. Fall of 2005, however, brought about a new potential alternative entitled Mologogo. Developed by a group of programmers from various parts of the world communicating only through the internet, it promised the same features as Buddy- Tracker completely free of charge. Here the Application proceeds with the same feature set focusing on making Buddy- Tracker the higher quality alternative, rather than pursuing a strategy of differentiation. Mologogo, while free, is rumored to be unstable and prone to crashes. Furthermore, the data is stored on a server controlled by individuals rather than a corporation. This creates the potential of hackers breaking in and stealing sensitive data, as individuals simply cannot

afford the appropriate security and reliability of service that a company as large as Motorola could provide. Furthermore, Mologogo is designed with Motorola phones in mind, thus limiting its functionality to only those phones. Buddy- Tracker is designed with platform independence in mind, ensuring its functionality on a variety of handsets.

1.10 Application Requirements

The architecture of project is based on Enterprise JavaBeans technology. The

Mobile application is developed in Java 2 Micro Edition.

The system needs different hardware and software components:

- Windows XP SP2, Windows 2003 SP1 or Windows 2000 SP4.
- Internet Information Services (IIS) 5.0 or above.
- Microsoft .NET Framework Version 2.0.
- MySQL Server 4.1 or 5.0.
- MySQL's ODBC-connector driver v3.51.
- Microsoft Data Access Components (MDAC) 2.6
- The web browsers that are currently supported are Internet Explorer 6 or 7 and Mozilla Firefox.
- Franson GPSServer V2.6.0.309
- NetBeans IDE 6.0
- Java Midlet : SUN J2ME Wireless Tool Kit 2.2
- GPS chipset enabled Device/ Mobile device

2. LITERATURE ASSESSMENT

2.1 LOCATION BASED SERVICES

In the following sub sections some major characteristics and definitions on LBS will be given. Here discussion of relation between GIS and LBS and give some Keywords which are useful to describe the LBS Technology.

2.1.1 The Relation of GIS and LBS

The handling of data with positional reference and spatial analysis functions (LBS-services) which give answers to questions like:

- "Where am I?"
- "What is near by?"
- "How can I go to?"

But LBS and GIS have different origins and different. They analyze that Geographic information Systems have been developed during several decades on the basis of professional geographic data applications. Whereas LBS were born quite recently by the evolution of public mobile services. With respect to user groups, GIS can be seen as traditional "professional" systems intended for experienced users with wide collection of functionality. Furthermore GIS systems require extensive computing resources. In contrast, the LBS are developed as limited services for large non-professional user groups [4].

2.1.2 LBS Component

- **Mobile Devices:** A tool for the user to request the needed information. The results can be given by speech, using pictures, text and so on. Possible devices are PDA's, Mobile Phones, Laptops, but the device can also be a navigation unit of car or a toll box for road pricing in a truck.
- **Communication Network:** The second component is the mobile network which transfers the user data and service request from the mobile terminal to the service provider and then the requested information back to the user.
- **Positioning Component:** For the processing of a service usually the user position has to be determined. The user position can be obtained either by using the mobile communication network or by using the GPS. Further possibilities to determine the position are WLAN stations, active badges or radio beacons. The latter positioning methods can especially used for indoor navigation like in a museum. If the position is not determined automatically it can be also specified manually by the user.
- **Service and Application Provider:** The service provider offers a number of different services to the user and is responsible for the service request processing. Such services offer the calculation of the position, finding a route, searching yellow pages with respect to position or searching specific information on objects of user interest (e.g. a bird in wild life park) and so forth.
- **Data and Content Provider:** Service providers will usually not store and maintain all the information which can be requested by users. Therefore geographic base data and location information data will be usually requested from the maintaining authority (e.g. mapping agencies) or business and industry partners (e.g. yellow pages, traffic companies).

2.1.3 LBS Keyword

LBS applications can be characterized by a number of keywords and related questions:

- **Mobile User:** Who or what is mobile? The mobile object can be a person or a device like a car navigation system.
- **Mobile Activities:** What Questions and Problems have users? Such questions do emerge from the user actions: locating, navigating, searching, identifying, event check [4]. Here distinguish three types of spatial scope:
  - Do I need an overview?
  - What is reachable for me?
  - Where am I?
- **Information:** A model of information retrieval is needed to answer the user questions. Such an information process model contains a model of possible questions, defines Queries of geographic base data and location information data, and specifies possible answers.
- **Search and Spatial Analysis:** Which methods and algorithms are suitable for real-time information query in the Internet and spatial data analysis? Further question are:" How to integrate data and information of different scale, quality, data types, prices?" "How is the data availability and actuality?"
- **User Interface:** Is a person using a PDA or mobile phone or something else? How can the user or (navigation) system formulate his needs and can make them more concrete after obtaining an overview?
- **Visualization:** How is the information, returned from LBS, communicated to the user? Speech, text, pictures, pictograms, maps, lists...

- **Technology:** How are service requests and data transferred between user and service provider? Where are the data stored? Which services are provided? Which positioning technology is used?

2.1.4 Push and Pull services

In general one can distinguish two different kinds of location services considering if information is delivered on user interaction or not [4]:

Pull services deliver information directly requested from the user. This is similar to call a website in the Internet by fill in its address in the web browser-address field. For pull services a further separation can be done into **functional services**, like ordering a taxi or an ambulance by just pressing a button on the device, or **information services**, like the search for a close Chinese restaurant.

Push services deliver information which is either not or indirectly requested from the user. Such push services are activated by an event, which could be triggered if a specific area is entered or triggered by a timer. An example for an indirectly requested service is a news service subscription which contains event information with respect to the actual city. A not requested service could be advertisement messages if a specific area in a shopping mall is entered or warning messages if weather conditions change (e.g. hurricane warnings). Since push services are not bound on previous user interaction with the service, they are more complex to establish. Here, the background information like user needs and preferences has to be sensed by the push system.

2.1.5 Adaption - Services respond to context

Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an

application. This can include the user and applications. Various researchers have attempted to classify the different kinds of contexts that are relevant to a user when accessing an information service.

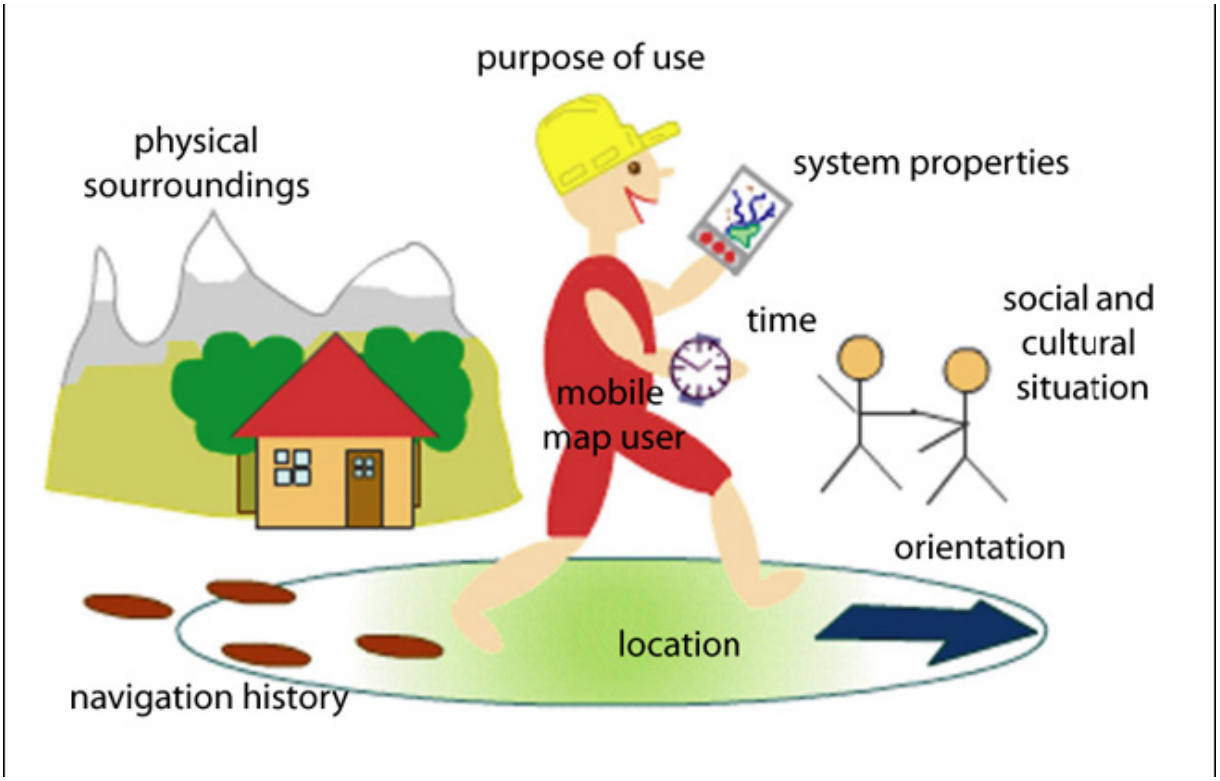

Figure 1.1: Different types of context

Mobile map user: The identity of the user is important to allow the service to consider such issues as:

- Their age and gender, for example children are unlikely to be interested in knowing about bars and pubs
- Their personal preferences, for example which language they want the service to be in
- Who their friends and colleagues are, if they wish to socialize and collaborate

Location: The location is the most commonly considered element of context. It allows information and services to be localized [5]. A user's

location can be absolute, for example describe by a geo-referenced coordinate, or relative, for example a room inside a building.

Time: Time can refer to the instantaneous time of day or longer intervals such as morning, afternoon or evening, day of the week, month, season of the year etc. In an entertainments service time might be used to determine if an event was still valid, for example a concert or a venue is open.

Orientation: The orientation of the user is important to determine the direction a user is heading in and thus what is in front, behind and to either side of them. In a tourist guide this might be used to determine what historical building the user is facing. In a navigation service it is important to check the user is heading in the right direction.

Navigation history: Navigational history allows the users to see where they have been and what they have seen and done. This can be useful in navigation to orientate a user while they're are moving and allow them to backtrack if they get lost. It can also help to build up a profile of what the user is interested in, enhancing the provision of relevant information.

Purpose of use: The purpose of use is defined by the activities, goals, tasks and roles (e.g. a ranger or a tourist) of users. Different types of usage require different

- Types of information,
- Types of presentation, for example maps, text or speech, and
- Modes of interaction

Social and cultural situation: The social situation of a user is characterized by their:

- Proximity to others,
- Social relationships and,
- Collaborative tasks

For example a user might want to ‘follow the crowd’ to find popular venues for example bars, exhibitions, talks, or alternatively avoid others if they are looking for wilderness areas, or they may just want to know where their friends are.

Physical Surroundings: The physical surroundings include such things as the lighting level or how much ambient noise there is. For example direct sunlight will make screens more difficult to read requiring the contrast to be adjusted.

2.2 GLOBAL POSITIONING SYSTEM

GPS tracking is not always possible. Global Positioning System (GPS) indicates that the system can be used everywhere on Earth, on land, at sea and in the air. A GPS device receives signals from the GPS satellites, high in the sky. If it receives strong enough signals from three or more different satellites, it can calculate its position. As these signals are very weak, in some circumstances it can be difficult to receive even three different signals. This is the case in towns with high buildings and relatively narrow streets or under trees with thick foliage. Under these conditions real-time GPS tracking could be troublesome. GPS tracking inside buildings is seldom possible [6].

A GPS device does NOT send any signal, not to the GPS satellites, nor anywhere else. It can only receive. So, if applicants want to know where the GPS device is, they will need a second technology to send this information to them. For this user often use a cellular network. This means that a GPS tracking device must at least contain a GPS receiver and a cellular phone modem.

2.2.1 Different Possibilities to obtain the Position Information

1) The modem can be programmed as a mobile phone with its own phone number, capable of sending SMS short text messages. If applicant calls the

modem's phone number, using own mobile phone, the device answers using call with a SMS message with the co-ordinates of its position. Advantage: user does not need a service contract and user does not have to pay a monthly fee. No-one even needs to know that user uses this possibility. Disadvantage: user will have to translate the co-ordinates in the SMS messages to something more meaningful, i.e. a position on a map.

2) The modem can be programmed with user's mobile phone number and send user a SMS message with its position every x minutes or hours. The interval can often be changed "over the air" by calling the modem and change some settings. Advantage: same as above. Disadvantage: same as above.

3) With the appropriate software and maps installed on user's mobile phone, the SMS messages can be translated to positions on a map on user's mobile phone screen. Advantage: same as above. Disadvantage: user need to buy the software and the maps [4].

4) The modem sends the position information in timely intervals to a server that displays the actual, as well as the past positions, on a map that can be viewed on any computer or mobile phone with Internet access. Advantages: user can follow the GPS tracking device from anywhere in the world [7]. Disadvantages: user needs a subscription to a service that delivers these possibilities. Normally user pays a monthly fee for this. The modem also needs an "always-on" connection with the cellular network for which user will have to pay another monthly fee.

5) As more and more mobile phones nowadays are equipped with a GPS receiver, these phones can perfectly be used as GPS tracking devices. There are already many services available for this purpose.

Great surfaces of this planet are covered by cell phone towers. But most surfaces are not. Here only have to think of the oceans and the

deserts. Even in these regions GPS tracking can be done, but the cell phone modem has to be replaced by a satellite modem. This way the position information can be sent over a communication satellite network (not the GPS satellites!) [6]. Advantage: In principle these kinds of GPS tracking devices can operate everywhere. Disadvantages: These devices are more sophisticated and bigger than GPS tracking devices with a cell phone modem. The devices AND the service are much more expensive.

2.2.2 GPS receiver elements

In a GPS receiver for ordinary use the treatment of all the signals should be different than in a GPS receiver to be used in a Formula in car or a plane. A classic GPS receiver, contrary to software GPS, is built with the following blocks [4]:

- Antenna
- GPS Front-end IC (the analog part)
- GPS Baseband IC (the digital part)
- Application Processor with Memory
- Input/Output

In the front-end IC scientist find a Low Noise Amplifier, a Frequency down Converter and an Analog to Digital Converter. In the Baseband IC are situated the correlates, a micro-processor and ROM and RAM memory. Both are *application specific integrated circuits* (ASICs) and sometimes they are built together into one *integrated circuit*(IC).

GPS Baseband IC

All the hard work of satellite acquisition and tracking, finding the phase transition of the navigation data, measure the pseudo range, demodulation of the navigation data and the decoding of the ephemeris data to obtain the

satellites positions is done in the Base band IC. A GPS receiver to be used in urban canyons must care for multi path mitigation. A receiver to be used in forests will have to treat for weak signals under the thick foliage. Both problems need to be treated differently in different algorithms [6].

2.2.3 GPS Server

The GPS system adds a server to the existing GPS system. The server uses a

GSM-network for communication with the terminal, and the Internet for retrieval of satellite position information. Therefore the GPS server must be placed in between a GSM-network and the Internet. The time for each step in the positioning is measured, except for the data transfer to and from the terminal and the time it takes for the GPS-chip to locate the satellites. These values are instead simulated with fixed values that either can be configured before and/or changed in run-time [10]. The mobile interface is an elementary positioning application developed for the J2ME MIDP 2.0 platform. It displays the position of a GPS-client along with a map of the surroundings. The view can be navigated by zooming and scrolling. The position is retrieved from a GPS-server and the map data from a specialized server.

Using this system the user may simply send visited locations as determined by the GPS server to the user’s home page for storage and for populating a user's database. Acting on timely requests the GPS server modifies the user's preferences upon probing the user for the GPS data and contact information [8].

2.3 J2ME: JSR 179 STANDARDS

The Location API for J2ME specification defines an optional package, javax.microedition.location that enables developers to write wireless location-based applications and services for resource-limited devices like mobile phones, and can be implemented with any common location method. The compact and generic J2ME location APIs provide mobile applications with information about the device's present physical location and orientation (compass direction), and support the creation and use of databases of known landmarks, stored in the device.

JSR 179 requires the Connected Device Configuration (CDC) or version 1.1 of the Connected Limited Device Configuration (CLDC). CLDC 1.0 isn't adequate because it doesn't support floating-point numbers, which the API uses to represent coordinates and other measurements. The Location API doesn't depend on any particular profile -- it can be used with MIDP or the Personal Profile.

The hardware platform determines which location methods are supported. If it doesn't support at least one location provider, LBS won't be possible. Applications can request providers with particular characteristics, such as a minimum degree of accuracy. Some location methods may be free; others may entail service fees. The application should warn the user before any charges are incurred [2].

It is up to the application to determine the criteria for selecting the location method. Criteria fields include: accuracy, response time, need for altitude, and speed. Once the application obtains a Location Provider instance that meets the criteria, it can use that object to obtain the location, in either of two ways:

- Invoke a method synchronously to get a single location.

- Register a listener and get periodic updates at application-defined intervals.

Through the Location API for J2ME, use information about the user's position to build new kinds of applications and services for mobile devices such as cell phones and PDAs, and to enhance existing services. JSR 179 specifies a generic API for obtaining locations, and thus makes porting LBS applications to a wide range of devices much easier. The critical issue that LBS developers must address is the privacy of the customer. To ensure privacy, follow sound programming guidelines and use the security framework in MIDP 2.0.

Determining the Device's Location

To discover the location of the device, LBS must use real-time positioning methods. Accuracy depends on the method used. Locations can be expressed in spatial terms or as text descriptions. A spatial location can be expressed in the widely used *latitude-longitude-altitude* coordinate system. Latitude is expressed as 0-90 degrees north or south of the equator and longitude as 0-180 degrees east or west of the prime meridian, which passes through Greenwich, England. Altitude is expressed in meters above sea level. A text description is usually expressed as a street address, including city, postal code, and so on [2].

Applications can call on any of several types of positioning methods.

- ***Using the mobile phone network***: The current cell ID can be used to identify the Base Transceiver Station (BTS) that the device is communicating with and the location of that BTS. Clearly, the accuracy of this method depends on the size of the cell, and can be quite inaccurate. A GSM cell may be anywhere from 2 to 20 kilometers in diameter. Other techniques used along with cell ID can achieve accuracy within 150 meters.

- ***Using satellites***: The Global Positioning System (GPS), controlled by the US Department of Defense, uses a constellation of 24 satellites orbiting the earth. GPS determines the device's position by calculating differences in the times signals from different satellites take to reach the receiver. GPS signals are encoded, so the mobile device must be equipped with a GPS receiver. GPS is potentially the most accurate method (between 4 and 40 meters if the GPS receiver has a clear view of the sky), but it has some drawbacks: The extra hardware can be costly, consumes battery while in use, and requires some warm-up after a cold start to get an initial fix on visible satellites. It also suffers from "canyon effects" in cities, where satellite visibility is intermittent.
- ***Using short-range positioning beacons***: In relatively small areas, such as a single building, a local area network can provide locations along with other services. For example, appropriately equipped devices can use Bluetooth for short-range positioning.

In addition, location methods can connect to a mobile position center that provides an interface to query for the position of the mobile subscriber. The API to the mobile position center is XML-based. While applications can be fully self-contained on the device, it's clear that a wider array of services is possible when a server-side application is part of the overall service [11].

Some applications don't need high accuracy, but others will be useless if the location isn't accurate enough. It’s acceptable for the location of a tourist walking around town to be off by 30 meters, but other applications and services may demand higher accuracy.

The applications presented “How is it useful?” emphasize that very different types of LBS applications exists and show further that LBS users can be persons or machines. In dependence on skills of a user to

handle electronic devices, the storage capabilities of a device, the user need of applying several services or fulfilling only a specific task a broad range of devices exists. Based on the latter device property LBS devices can be distinguished into **single purpose** and **multi purpose** devices. A **single purpose device** is for instance a car navigation box, a toll box or an emergency remote for old or handicapped people. As well part of that category are devices which call service engineers or rescue teams. But also more advanced systems like augmented reality systems - which might be used by a state inspector for bridges and other buildings - belong to it. **Multi purpose devices** will be used by a broad number of people and will be part of everyday life. Such devices can be mobile phones, smart phones, Personal digital Assistants (PDA's) but also Laptops and Tablet PC's.

3. PROJECT DESIGN

Among the services recently launched by mobile operators to generate new revenue streams, Location Based Services are already being delivered to users on their existing terminals. As the General Packet Radio Service (GPRS), Enhanced Data rates for GSM Evolution (EDGE) and Third Generation (3G) systems evolve, LBS will become more sophisticated and easier to use, making them more appealing to private and business users alike. It is important for mobile operators to start rolling out LBS without delay to keep up with changes in the market and not lag behind the competition.

LBS solution supports the Global System for Mobile communication (GSM), GPRS, and Universal Mobile Telecommunications System (UMTS) networks, offering several levels of accuracy. It thus enables

mobile operators to offer ‘basic’ and ‘premium’ location-based services, depending on the accuracy required by the service and the target users. This solution, which is scalable from a small system (serving a few subscribers) to a large system (suited to many subscribers), can support simple CellID (Cell Identification) through to the more complex Assisted-Global Positioning System (A-GPS). Enhancements are needed to several network nodes, namely the Base Transceiver Station (BTS), Node B, Base Station Controller (BSC), Radio Network Controller (RNC), Mobile services Switching Center (MSC) and Home Location Register (HLR). In addition, two new types of node are needed to support LBS: the Gateway Mobile Location Center (GMLC) and the GPS server. The GMLC receives requests for user location information and forwards them over the network, while the GPS server calculates the user’s position when the GPS positioning method is used [9].

3.1 DISTRIBUTED CHANNELES

• Mobile operator for service delivery and user location: The operators own the network infrastructure through which the location information is retrieved and provide the service to users.

• Positioning equipment manufacturer: This includes the providers of the location-enabling platforms (e.g. location servers and gateways), which retrieve the user’s location information (using an appropriate positioning method) from the network or the user’s handset. They then deliver this information to the applications.

• Location data or geocoding provider: Providers of any kind of geocoded information, such as the locations of streets and buildings.

• Application developers: Develop location applications that combine the information stored in the GMLC with other geographically organized information to create value. LocatioNet, Webraska, and Vicinity/Microsoft are typical application developers.

• Content developers: Include the producers of any kind of information that is geographically referenced (e.g. time-sensitive information, such as cinema or theater start times, traffic information).

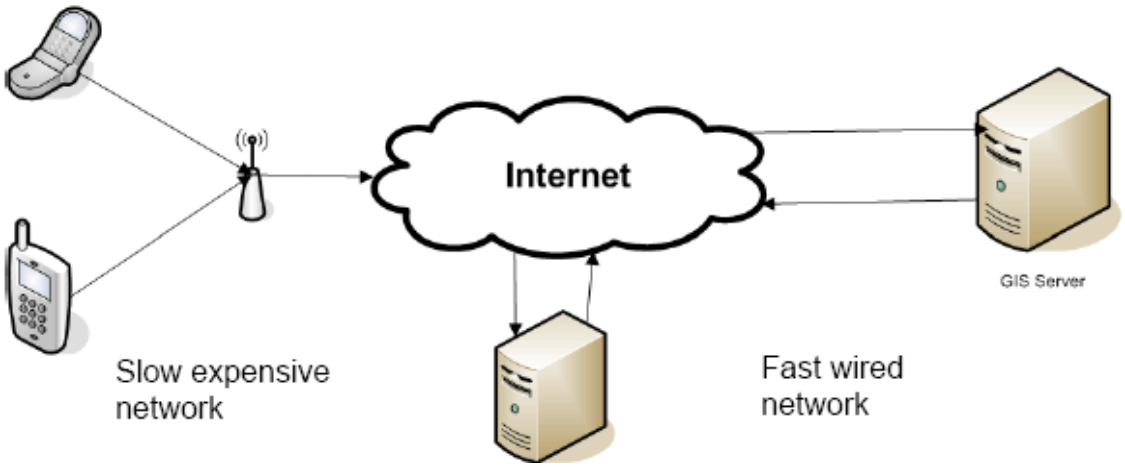

Figure 3.1 High Level Architecture

A multitude of services can be improved by taking the user's location into consideration. Mobile operators have two assets that ensure they are central players in the location-based services value chain: the channel to the user and the "real-time" location of the user. This application solution is tailored for mobile operators who are looking to migrate from voice to a world of services. It includes LBS enablers, such as the GMLC and the GPS server, making it possible to support the full range of accuracies from simple CellID

to complex GPS. In this way, it enables the operators to offer their customers a variety of location-based services.

Basic services, such as “find the nearest POI”, can utilize CellID, whereas premium services, like navigation, fleet management and emergency services should preferably use GPS because of its greater accuracy [9]. This application complies with the 3GPP standards, simplifying interoperability and facilitating its integration into existing mobile networks.

3.2 SYSTEM ARCHITECTURE

GPS employs existing wireless mobile networks to enhance the functionality of traditional GPS. GPS makes use of a dedicated server that is connected to a GPS reference network. The reference network monitors the satellite receivables and forwards this information to the GPS server [12]. The server sends assistance data to mobile receivers through the mobile network. This data includes Doppler shift corrections, satellite navigation messages and approximate receiver positions.

Through compensating for Doppler shifts, the task of signal acquisition is reduced. Less frequency bins must be scanned before acquiring a signal that may be decoded and thus a pseudorange determined. The reduced bandwidth searched allows for greater receiver sensitivity and position determination to be reduced from more than 30 seconds to a few seconds.

The assisted navigation message may be used by an GPS receiver to predict specific bits in the GPS signal’s navigation data. Predicting these bits allows for compensation of the associated phase shift of the modulated C/A code when determining pseudoranges. The compensation allows for coherent integration beyond the 20ms GPS data-bit period to a second or more when the receiver is stationary.

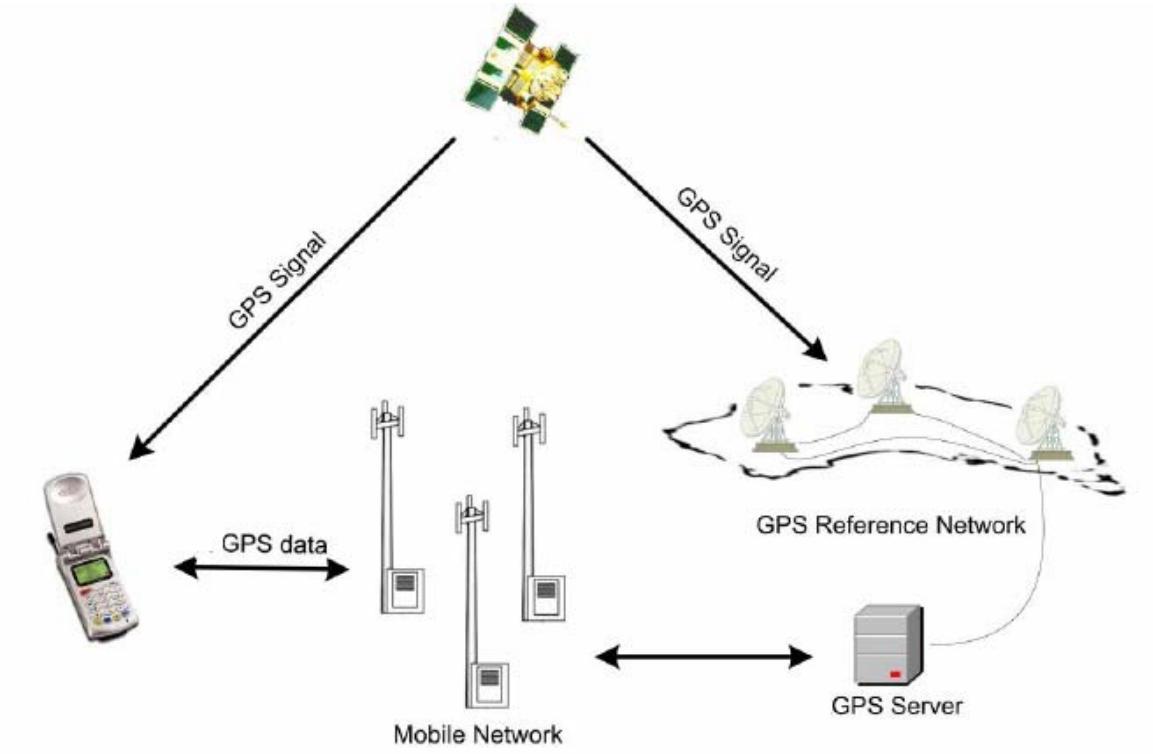

Figure 3.2 System Architecture

Additional information may be utilized in a GPS system. This includes approximate locations, satellite ephemeris and clock corrections. Providing a receiver with up-to-date information also results in achieving a reduced Time-To-First-Fix (TTFF). This allows non GPS specific receivers to make use of some of the advantages of a connection to the GPS reference network.

The mobile network may be used to determine the initial position of a mobile station. A number of methods are used to achieve this including; Angle of Arrival, Time of Arrival or Time Difference of Arrival. These methods assist in determining an approximate position without the accuracy of GPS.

This is achieved through calculating the actual position at a network base Station. The mobile station then only needs to send pseudo range information to the base station. This data may have GPS corrections applied before position calculation, which is sent back to the mobile unit. This form of positioning is referred to as MS-Assisted GPS.

Calculating the position in the receiver is known as MS-Based GPS. MS-Assisted GPS requires less sophisticated hardware in the mobile station since it does not need any manipulation to the determined the pseudo range. This in turn puts less demand of memory and processing requirements.

3.3 SYSTEM PROPERTIES

This relates to the computer infrastructure the user is employing. What type of device they are using and what are its capabilities (e.g. touch screen, color or black and white etc.). Here user have access to a continuous internet connection or if it is only intermittent. Also depends on the bandwidth of the connection and the quality of the positioning information, e.g. the GPS coverage.

Systems that can dynamically change their behavior because of context have been termed variously; reactive, responsive, situated, and context-sensitive and environment directed. However, the term adaptive has become the most commonly used in mobile cartography [4].

Adapt ion can take place at four different levels:

1. **Information level:** the content of the information is adapted. Examples include filtering information by proximity to a user or changing the level of detail of information according to tasks.
2. **Technology level:** Information is encoded to suit different device characteristics (e.g. display size and resolution, network and positioning availability). For example using auditory Driving instructions for users with mobile phones or maps for users with PDAs.

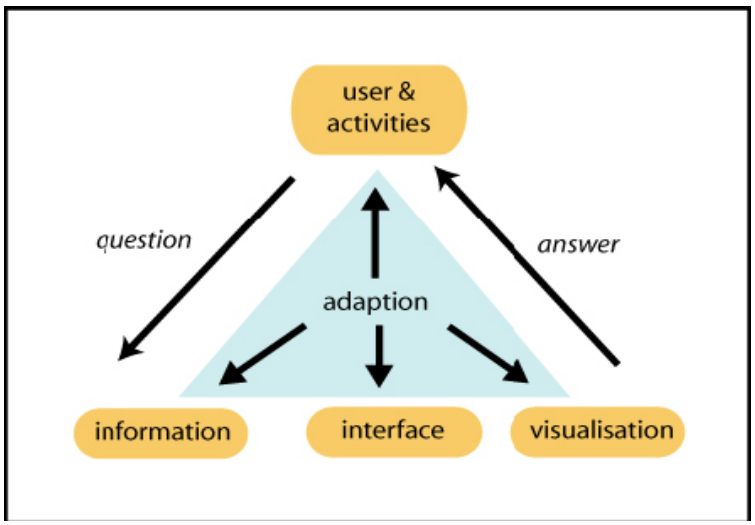

Figure 3.3 Levels of adapt ion for mobile technologies

3. **User interface level:** the user interface is adapted. For example automatically panning and re-orientating a map as the user moves about.

4. **Presentation level:** the visualization of the information is adapted. For example restaurants those are more relevant to a user's preferences in price and taste are shown with more crisp icons and those less relevant use more opaque ones.

3.4 SEQUENCE DIAGRAM

3.4.1 Terminal Location Query

Pattern: Request / Response.

For an application to determine the location of a terminal device, it provides a terminal device address and desired accuracy, and receives the location for the device requested.

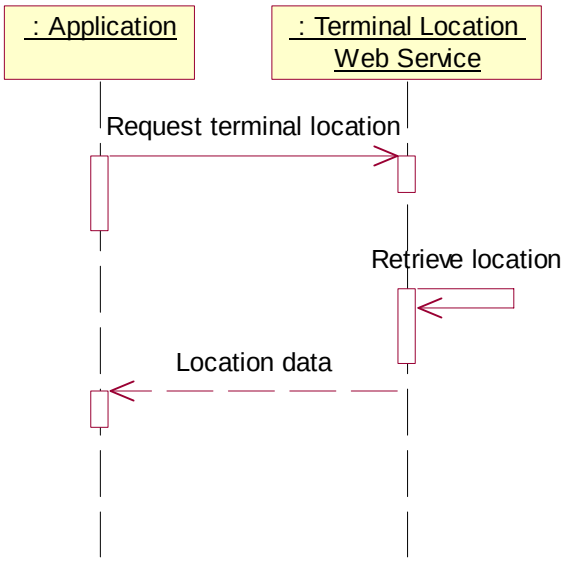

Figure 3.4

3.4.2 Terminal location group query

Pattern: Request / Response.

When an application requires the locations of a set of terminal devices, it may provide an array of terminal device addresses, including network managed group addresses, and receive the location data for the set of devices requested.

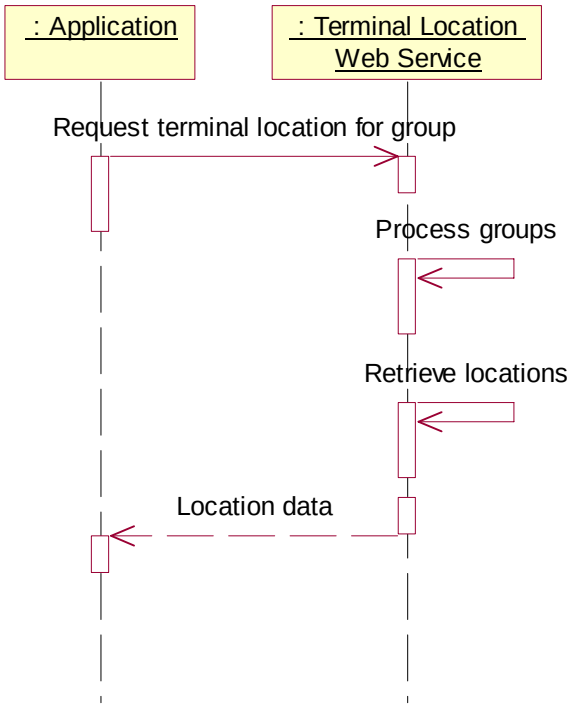

Figure 3.5

3.4.3 Terminal location notification

Pattern: Application Correlated Multiple Notifications.

An application can be notified of a terminal device entering or leaving a geographical area. When a matching event occurs; a notification message will be sent to the application.

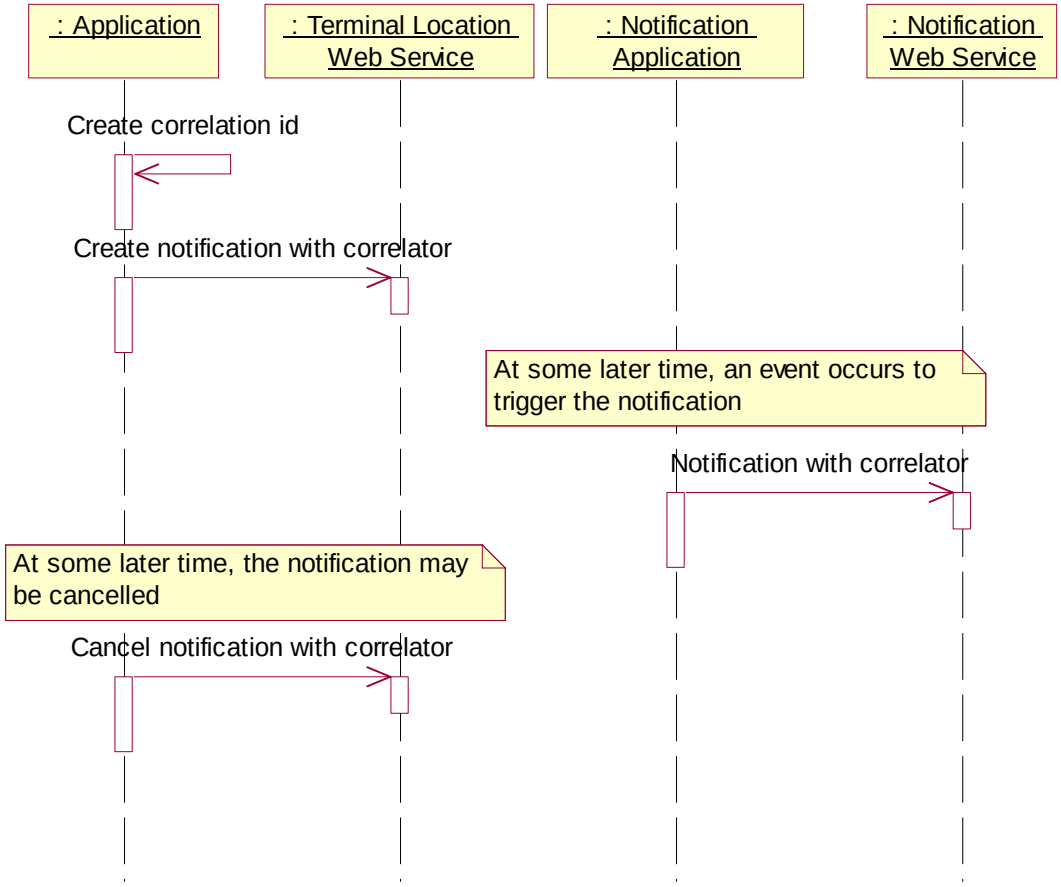

Figure 3.6

3.4.4 Terminal location notification with check immediate

In some applications, the terminal location notification will be used to watch for a specific location change. An example is a 'call when present' service, where the terminal location is checked and determined to be outside the target area, and a notification is set up to notify the application when the terminal enters the target area. Between the time of the original location determination and the time the notification is set up, the terminal could move into the target area, thus the notification on entry into the target area would not be sent.

Using the check immediate flag, after the notification is established, the terminal location will be determined, and if the terminal is in the target area, then a notification will be sent immediately. The following sequence diagram shows this scenario.

This sequence shows:

- The Enterprise Application checks the location of a terminal, and receives its location (in this scenario determining that the terminal is outside the target area).
- The Enterprise Application generates a correlator, and starts a notification with criteria defined to notify the Enterprise Web Service when the terminal enters the target area and the check immediate flag set to true.
- Sets up the notification to monitor terminal location.
- Check the current location of the terminal, and determine if the terminal lies inside the target area.
- In this case, the terminal is in the target area, and a notification is delivered to the Enterprise Web Service.
- The count of notifications is incremented and compared to the notification count limit.

- In this case, a single notification was requested, and the end notification message is sent.
- The startGeographicalNotification operation completes.

This scenario includes the full set of interactions in one sequence, which also shows that the notifications can be received concurrent with the creation of the notification.

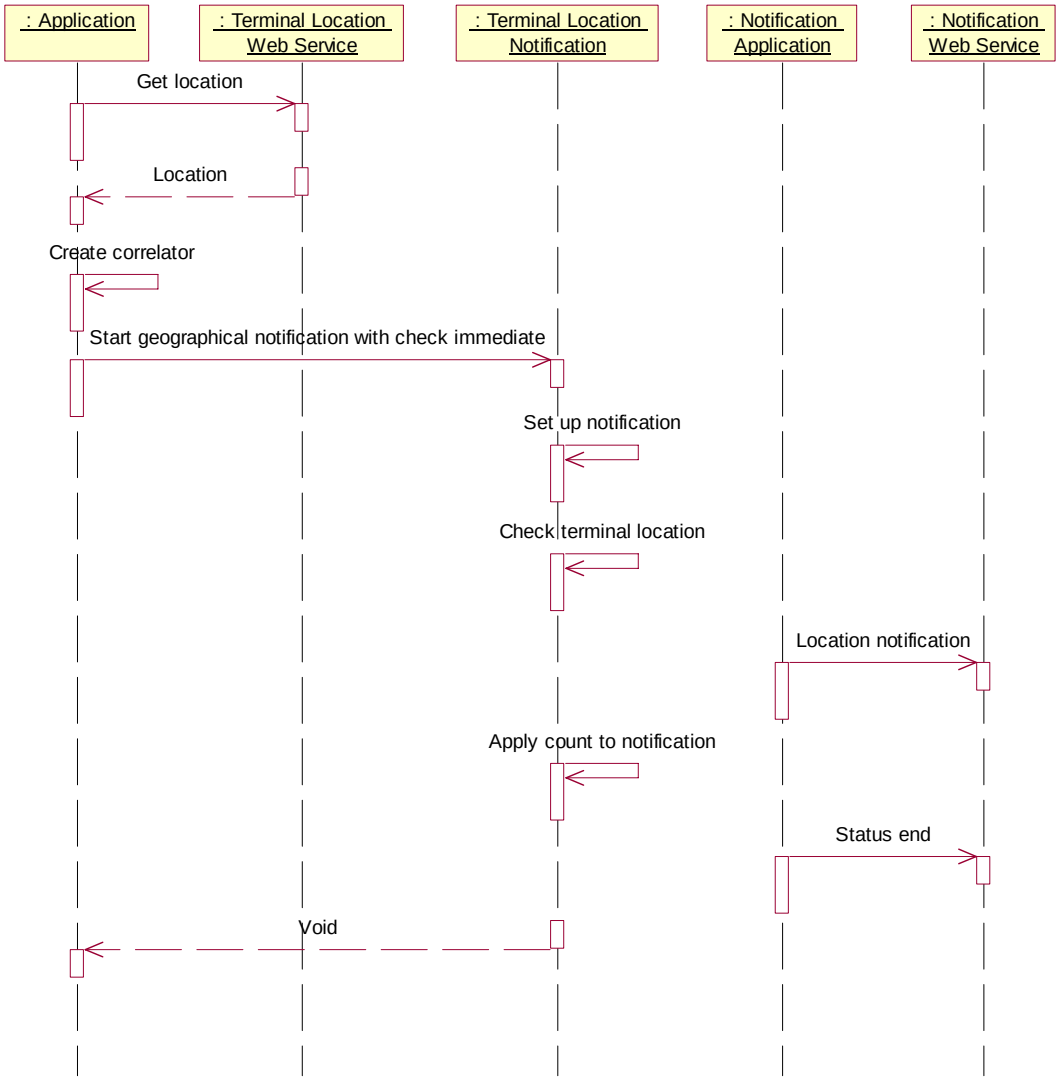

Figure 3.7

3.4.5 Terminal location periodic notification

Pattern: Application Correlated Multiple Notifications.

An application can be notified of a terminal device location on a periodic basis. At each interval, a notification message will be sent to the application.

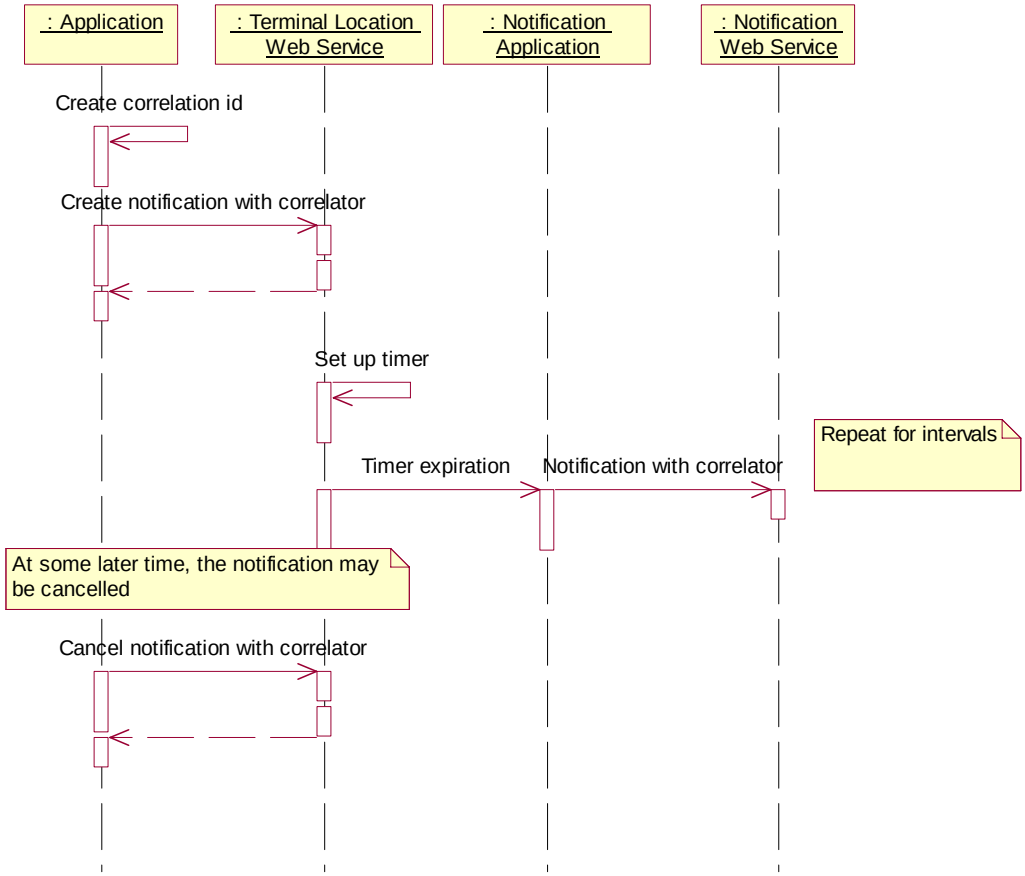

Figure 3.8

3.5 DEPLOYMENT DIAGRAM:

The clients connect to GPS Server and first send the name of their destination position. After that, at regular intervals, they send queries about the next step that they need to go to, in order to reach their destination position. The GPS Server gets the client position information from satellite. Using its, the GPS Server calculates the shortest route that the client can take in order to reach its destination and then returns this next step information to them. The database server stores all the states and session information about each query from different clients.

Use case diagram:

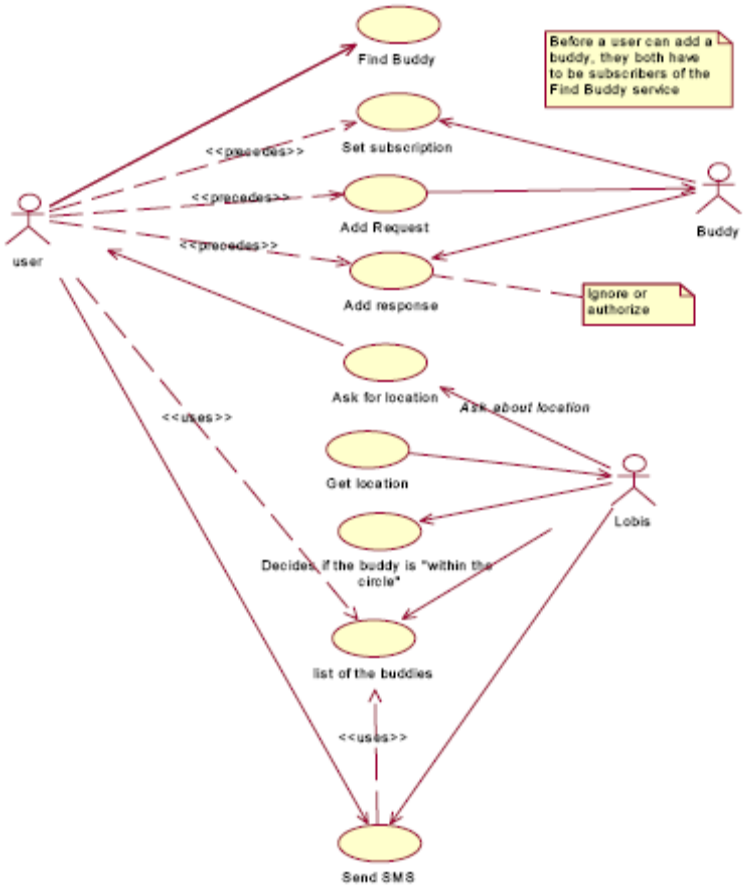

Figure 3.9

Specifications: This service allows users to find the location of other subscribed users.
Actor: Web& WAP User, System, Buddy
Pre-condition: The user may activate or deactivate this service through a WAP device
Post-condition: The user gets the information from chosen service.
Event Flow:
1. The user access his mobile phone (login)
2. The user sets his status (On-/off-line and away)
3. The user is able to add other buddies that have consented.
4. The user can choose to be visible or invisible to other users.
5. The system provides the user with a list of other buddies near him.
Variation: In order to locate a buddy, the buddy must be on your list. Messages can be sent to buddies in any status.

User Profile Admin:

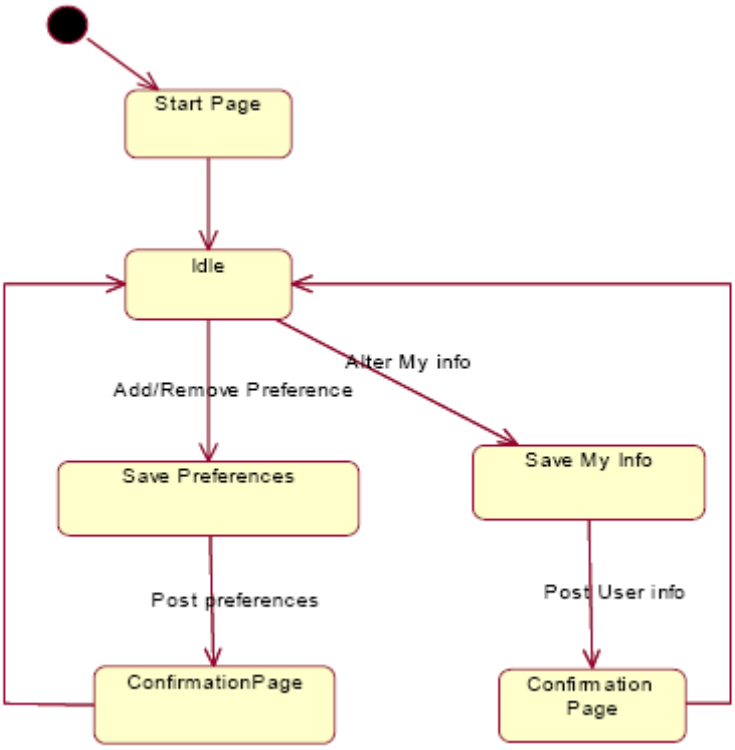

Figure 3.10 State chart diagrams for user login

Servers

The system has three types of servers: satellite server, database server and GPS server. The GPS server is the main server admin focus on. Satellite server tells the GPS Server what it observes from the client. The database server stores static map information, the states and session information about each query from different clients.

The following tables describe the details of these servers. The types of server are more conceptual than physical.

Satellite Server

Description	This server represents a GPS satellite
Functions	The only function of the satellite server is to detect the

	position of each client.	
Type	It is considered as an external server with 100% reliability and availability. (pretty much like a credit-card info server in E-commerce)	
Invocations	It does not invoke any server It gets invoked only by the GPS server	
State management	Does not store any state information	
Interface definitions	Satellite Interface	
	Operations	String getNewLocation (Position currentStreetName, String clientID, Vector streetOptions)

Database Server

Description	This server contains all the information about the different streets in the city
Functions	Database management
Type	It is considered as an external server with 100% reliability and availability.
Invocations	It does not invoke any server It gets invoked only by the GPS server
State management	Does not store any state information
Interface definitions	All the database management will be handled through contained managed persistence

GPS Server

Description	This is main server. Here this application implement the main functionalities of the system.
Functions	Inform the client's current position

	Suggest the optimal route to reach the destination (return an next step object which contains the next street name as well as other necessary information) Query the satellite server for the position of the client(based on the possible set of streets the client can turn into from the original street) Manages the sessions of its clients	
Invocations	It invokes the GPS Satellite It invokes the Database Server It is invoked by the GPS client	
State management	For each client it holds: The session ID The entire route to the desired destination The actual position of the client The history movement of all clients in their life cycle	
Interface definitions	GPSServer Interface	Described in the GPSServer Interface definition below

Client

The clients are with Mobile station/ laptop/ Blackberry (GPS enabled).

Description	The client is supposed to be a GPS device located in some vehicle
Functions	The only function of the satellite server is to detect the position of each client.
Invocations	It invokes the GPS server
State management	Does not store any state information

Some supplement about clients:

- Main purpose: To reach their destination/know their current location
- It invokes the GPS server. Every time it comes to an intersection or every 15 seconds.
- If destination address not in database, client needs to select another destination.
- Information lost, the client has to resend the information

Exceptions/ Errors Generated

The following are exceptions in system. Notice that here there is an assumption in project is that the system has database integrity, so no need to consider exceptions about no-meaningful data.

- Destination Address not in Database
- Invalid ID
- Conflicting ID
- Satellite Connection Timeout
- Cannot identify next step
- Information Lost
- Cannot connect to satellite
- Cannot connect to database

Interface

The following are interfaces, each of which has some methods.

GPS Server Interface

Route getService(String destination) // Gps server returns a unique id for this client which is a data member of object Route;

Boolean stopService(String id)

Route getNextStep(String id)

RouteFinder Interface

Route getRoute(Position currentLocation, Position destinationLocatin)

Satellite Interface

Position getNewPosition(Position currentLocation, Position [] alternatePosition, Position destinationLocation, String id)

Database

There are two data domains in database. One is map, the other is client.

- Map domain identifies the information about the map. Two optional designs are offered now. Map information is entity.
- Client domain identifies the information about each session of each client. Client information is session.

4. PROJECT IMPLEMENTATION

4.1 J2ME APPLICATION

4.1.1 Introduction

The cell phone application itself is designed on the J2ME platform. It utilizes the MIDP, CLDC 1.2, and JSR-179 API profiles. When approaching the design of this applet, it was immediately apparent that speed and memory usage would be of paramount importance, as the average cell phone does not have either. The applet is therefore designed with brevity and reliability in mind [3].

4.1.2 Design

The applet is composed of three parts: the main application driver, a MovementTracker object, and a SendData object. The main application driver serves primarily to instantiate the MovementTracker and SendData objects. Shortly after execution, the user is presented with a menu where he or she enters data concerning the username, password, update frequency and phone number. The data is then stored in private variables and the user is passed into the main application screen. The main application screen consists of a status window, which is updated with the current status of the tracker. The status varies depending on a boolean running value, any errors present, or current state of the program. When the program is first executed, and the initial options are entered, the status window reads

“Initialized”. If the running boolean is true, then the status window displays “Tracking”. If it is false, the window displays “Tracking stopped”. If at any point the cell phone loses its tracking device, such as when the phone goes out of range of a GPS satellite, or triangulation cannot be performed, an exception is thrown which causes the status window to display “Provider lost”. The user must then restart the application. Though it would be possible to attempt to reacquire a new location provider every set period, such a course was deemed inefficient as the creation of a new LocationProvider object is extremely expensive and should not be performed with any kind of frequency or performance will suffer. With this solution, the user can reestablish the communication when they have reached an area which will allow access to some GPS positioning provider. Upon selecting “Begin Tracking” from the soft menu, the program checks the initialized boolean and if false, the MovementTracker object is instantiated with the aforementioned options passed in as parameters. If it is true, then this is a case where tracking has been stopped and the user wishes it to resume, thus the running boolean of the MovementTracker is set to true. If the user selects “Stop Tracking” from the soft menu, the program checks the initialized boolean, and if true, sets the running boolean of MovementTracker to false, preventing it from performing updates.

Upon instantiation, five parameters, containing the update interval in seconds, the user name, the password, the phone number, and a hard-coded radius respectively, are passed to the MovementTracker object. The MovementTracker uses the radius parameter to set criteria for picking a LocationProvider which is a JSR-179 defined object which in turn allows for an interface to some physical device inside the phone which can provide the current geographic coordinates. A

LocationListener is then set up to poll the device for coordinates every some set period of time specified by the update interval. The option exists to set a timeout value for the LocationListener in cases of static interference or a defective tracker, but due to a lack of test hardware set the LocationListener not to have a timeout as it is difficult to gauge the appropriate timeout. The user name and password are simply stored as public values to be accessed by the SendData object later on. Every time the LocationListener issues an interrupt, a custom designed handler processes the interrupt. In instantiating the handler, a SendData object is also instantiated with access to its superclass. The handler establishes whether admin should process the update (if the running boolean is false or the new coordinates are less than the radius passed in at instantiation from the current location admin do not perform an update as that would pollute the server with repetitive data). If this is an update admin should process, the new coordinates are stored in the phone and the handler uses the SendData object to send the data composed of user name, password, phone number, and latitude and longitude coordinates, to the server and the applet goes into the background until another interrupt is issued. The entire design of the MovementTracker is multi-threaded so as not to interrupt any other functions the program may be performing.

The SendData object merely connects to the server using the CLDC 1.2 profile formats the data so that it may be processed by the server, sends the actual data to the server and closes the connection.

4.2 DATA TYPE DEFINITION

4.2.1 Latitude and Longitude values

Latitude and longitude values used in the present document follow the conventions of the ISO 6709 [3] specification, as it applies to latitudes and longitudes specified using decimal degrees.

Latitude values are expressed as floating point numbers in the range -90,0000 to +90,0000, using decimal degrees (as opposed to minutes and seconds). Positive values indicate locations north of and on the equator. Negative values indicate locations south of the equator.

Longitude values are expressed as floating point numbers in the range -180,0000 to +180,0000, using decimal degrees (as opposed to minutes and seconds). Positive values indicate locations east of and on the prime meridian (Greenwich). Negative values indicate locations west of the prime meridian up to the 180th meridian.

4.2.2 Accuracy values

Two accuracy values are used in some of the operations. These values express the desire of the application for the location information to be provided by the Web Service. The choice of values may influence the price that the Service Provider charges.

- The 'requested accuracy' expresses the range in which the application wishes to receive location information. This may influence the choice of location technology to use (for instance, cell sector location may be suitable for requests specifying 1000 meters, but GPS technology may be required for requests below 100 meters).

- The 'acceptable accuracy' expresses the range that the application considers useful, if the location cannot be determined within this range, then the application would prefer not to receive the information. For instance, a taxi tracking service to determine the closest taxi to a person may not be useful if the accuracy cannot be

provided within 1000 meters to provide prompt service. This will also reduce customer satisfaction issues, since results that are not useful can be handled appropriately for billing (e.g. Service Provider may choose not to bill for these).

The ‘maximum_age’ expresses the maximum age of location information that the application considers useful. This can be used by the service provider to supply cached location information rather than always to do a direct network location request.

The ‘response time’ expresses the expected response time from an application point of view. If the network is unable to respond within the desired time frame, the application would prefer not to have the information as it may no longer be useful.

The ‘tolerance’ expresses the priority of response time versus accuracy. If the application is delay tolerant the network is expected to return a location with the requested accuracy even if this means not complying with the requested response time. The application can also indicate that it is more important that the location information is returned within the requested time even if this implies that the requested accuracy can not be fulfilled. An indication of ‘no delay’ implies that the application expects the service provider to return any current location estimate immediately.

In triggered notifications, a tracking accuracy is defined. This accuracy refers not to the accuracy for the area being checked against, but rather for the accuracy of the technology used to track the terminal. For instance, a fine grained tracking accuracy would be suitable for tracking the terminal entering a specific location, like a person arriving at a destination building. A coarse grained tracking accuracy would be appropriate for determining when a person has arrived at a city after a plane trip or a truck is in the vicinity of a warehouse.

4.2.3 EnteringLeavingCriteria enumeration

Indicator for whether the notification is related to entering an area or leaving an area.

Enumeration value	Description
Entering	Terminal is entering an area
Leaving	Terminal is leaving an area

4.2.4 LocationInfo structure

Location information represented as a coordinate.

Element name	Element type	Optional	Description
latitude	float	No	Location latitude
longitude	float	No	Location longitude
altitude	float	Yes	Location altitude
accuracy	int	No	Accuracy of location provided in meters
timestamp	dateTime	No	Date and time that location was collected

4.2.5 RetrievalStatus enumeration

Enumeration of the location items that are related to an individual retrieval in a set.

Enumeration value	Description
Retrieved	Location retrieved, with result in **currentLocation** element
NotRetrieved	Location not retrieved, **currentLocation** is not provided (does not indicate an error, no attempt may have been made)

4.2.6 LocationData structure

Data structure containing device address, retrieval status and location information. As this can be related to a query of a group of terminal devices, the **reportStatus** element is used to indicate whether the information for the device was retrieved or not, or if an error occurred.

Element name	Element type	Optional	Description
reportStatus	RetrievalStatus	No	Status of retrieval for this terminal device address
currentLocation	LocationInfo	Yes	Location of terminal. It is only provided if **reportStatus**=**Retrieved**.
errorInformation	common:ServiceError	Yes	If **reportStatus**=**Error**, this is the reason for the error. Error due to privacy verification will be expressed as POL0002 in the ServiceError
address	anyURI	No	Address of the terminal device to which the location information applies

4.2.7 DelayTolerance enumeration

Enumeration of the delay tolerance items that forms part of the location request.

Enumeration value	Description
NoDelay	The server should immediately return any location estimate that it currently has. If no estimate is available, the server shall return the failure indication and may optionally initiate procedures to obtain a location estimate (e.g. to be available for a later request).
LowDelay	Fulfillment of the response time requirement takes precedence over fulfillment of the accuracy requirement. The server shall return any current location estimate with minimum delay. The server shall attempt to fulfill any accuracy requirement, but in doing so shall not add any additional delay (i.e. a quick response with lower accuracy is more desirable than waiting for a more accurate response).
DelayTolerant	Fulfillment of the accuracy requirement takes precedence over fulfillment of the response time requirement. If necessary, the server should delay providing a response until the accuracy requirement of the requesting application is met. The server shall obtain a current location with regard to fulfilling the accuracy requirement.

4.3 GOOGLE MAP API

4.3.1 Browser Compatibility

Google Maps supports recent versions of Mozilla Firefox, Internet Explorer, and Safari and Opera limited. It is not universally supported, however. For example, it is not compatible with IE 5.0. The Google Maps API provides a global method - GBrowserIsCompatible() - to check compatibility, but it does not exhibit any default behavior when it

detects an incompatible browser. Thus, the detection of an incompatible browser currently does not lead to the display of any warning messages.

4.3.2 Problems with Internet Explorer (IE)

4.3.2.1 JavaScript

Our website sometimes crashes when prompted to display a Google Map in early versions of IE. This is because certain browsers may have “quirks” regarding JavaScript. As an example, early versions of IE (pre-v5.5) do not allow JavaScript inside tables, which this application utilizes at the time of the writing of this document.

4.3.2.2 Polylines

In the earlier stage of development process, the polylines in maps were unable to be displayed in Internet Explorer. The reason behind this behavior was determined to be the browser incompatibility between Google Maps API and IE.

4.3.3 Limitations

4.3.3.1 The Satellite and Hybrid Modes

The satellite mode and the hybrid mode of Google Map are not available in some closer zoom levels in some cities and places due to infrequent satellite activity. As observed, it seems that the availability is directly related to the population of the city in question. Therefore, the satellite mode and hybrid mode are limitedly supported in some places in LBS service.

4.3.4 Current Nature of Google Maps

Currently, Google Maps is continually being tested by many users and is continually improved. It is in open Beta. There is a possibility that some modification made to Google Maps by Google may cause an incompatibility with this application, which in the worst case could be resolved by the redesigning of some aspect of application. Also, the

Beta nature of Google Maps implies that Google is not obligated to continue providing this free service. It is also possible that the service may be charged by Google or worse yet, be discontinued.

4.3.5 Degrade of Quality Result from High Access Rate

Although there is no limit on the number of pages view per day using the Maps API, more than 500,000 page views per day may lead to service degraded, according to Google Maps API documentation. If anyone expects more than 500,000 page views per day, here need to contact Google Maps API for provision additional capacity to handle the traffic.

4.3.6 Google Maps API Key and Directories

A single Google Maps API key is only valid for a single directory on web server. Moreover, a Google account is required to sign up for a Google Maps API key. Therefore, the URLs for application test pages and services are limited by the amount of keys who can sign up for. This application can only sign up for one Map API key and obtains one URL that works with Google Maps API.

4.4 NMEA FORMATE

In the 1980s the National Marine Electronics Association (NMEA) developed the NMEA 0183 Interface Standard for data exchange between marine electronic devices. Today, most global positioning systems use the NMEA interface for data exchange [13]. Thus the J2ME GPS data parser will work on any GPS receiver that implements this standard.

NMEA Sentences

$GPRMC,214434,A,3753.666,N,12203.162,
W,0.0,0.0,270901,15.4,E,A*33
$GPGGA,214616,3753.667,N,12203.167,W,
1,04,5.6,121.1,M,-27.4,M,,*7
$GPGSA,A,3,01,03,20,22,,,,,,,,,7.
4,5.6,1.5*36
$GPGSV,3,1,10,01,69,062,47,03,12,106,
37,04,12,279,00,08,12,250,00*77
$GPGLL,3753.667,N,12203.167,W,
214616,A,A*54
$GPBOD,,T,,M,,*47

The $GPGGA sentence contains the topological location information. The relevant fields are as follows:

Field				
1	Message		Header	($GPGGA)
3	-	4	Latitude,	North/South
5	-	6	Longitude,	East/West
10 Altitude in meters				

The latitude 3753.667 denotes 37 degrees, 53 minutes, and 66.7 seconds. There are a couple of problems with this format. First, the longitude and latitude numbers are not base 10 numbers, so difficult for an application to determine relative distances. Second, the KVM does not support floating or double primitive types or the respective wrappers, making any decimal value an incorrect numerical format.

The answer is to use fixed-point integer calculations to handle decimal values. For example, the value of 100.234 is 100234, with a fixed point of three. The GpsParserImpl class instance normalizes the longitude and latitude coordinates by using the following code:

```
int nc = (10000 * k[0]) + (10000 * k[1])/60 + (1000 * k[2])/3600;
```

Where k[0] denotes the degrees (37), k[1] the minutes (53), and k[2] the seconds (667). The normalized integer value for this latitude is 379019.

To find the speed and direction of the target, developer extract information from $GPRMC (fields 8 and 9, respectively). If the altitude field is blank, it means the GPS receiver able to pick up the signal from a fourth satellite, which is necessary for 3D tracking.

In this scenario the target is not moving, so the speed and direction fields contain 0.0. The J2ME application developer may use this information in interesting ways. For instance, client could display all targets moving over 60 m.p.h. in dark blue, or all targets moving north in green. This type of color-coding and dimensional reduction of information allows an enormous amount of location information to be displayed to the user of a limited mobile device. This will undoubtedly be an important area of future growth, research, and standardization for the mobile industry.

4.4.1 Accessing GPS Data through the Serial Port

Opening the InputStream

Plugging the GPS receiver into the laptop or mobile device starts a stream of GPS data flowing over the serial port. Reading the input stream from the serial port is no different than reading an input stream from a file or an HTTP connection. As mentioned previously, user can use either HttpReader, which functions as a GPS data stream, or SerialReader to open a connection to an actual GPS receiver. Only difference in the code:

SerialReader:

```
String URI = "comm:1;baudrate=4800;bitsper char=8";
InputConnection inputCon = (InputConnection) Connector.open(URI);
InputStream is = inputCon.openInputStream();
```

HttpReader:

```
String URI = "http://xspsoft:8080/GPS.html";
HttpConnection inputCon = (HttpConnection) Connector.open(URI);
InputStream is = inputCon.openInputStream();
```

Note that a SerialReader object opens a connection to the stream on communication port 1; however, user's port number may be different. If user choose the wrong port, the KVM will throw an IOException stating that the system find the specified file. If the port is occupied, the IOException states that the handle is invalid.

Each GPS receiver sends data at a certain baud rate. Ensure that the baud rate in the String URI matches the baud rate of user GPS

receiver. If any of the bytes in the input stream are above 128, the baud rate is incorrect. The input stream will come through as garbage. Also make sure there are no spaces between the semicolons or the program will throw an exception.

Parsing the InputStream

The abstract class GpsParser contains the base code to read the GPS data stream. A concrete, direct subclass instance instantiates either the SerialReader or HttpReader, depending on the preference. GpsParser returns an NMEA sentence as a character array, one line at a time, through the use of the Template Method and an abstract method denoted as command(char[] c).

The GpsParserImpl class contains an example of a concrete implementation of the command instance method, which is executed for each NMEA sentence flowing through the serial port. In the GpsParserImpl class, the match Command class method determines whether the current NMEA sentence is the GPGGA sentence. If it is, the command method scans and tokenizes the data into a vector.

Invoking the GpsParserImpl constructor causes the GPS data stream to continually update the Coordinate class with the respective longitude and latitude information. Now any application can invoke the Coordinate accessor class methods getLatitude, getLongitude, getAltitude, getSpeed, and getDirection to get the most current position of the target [13].

4.5 FRAMEWORK DESIGN

Figure 4.1 shows a typical application using the framework. The application is required to provide a concrete implementation of the interface LocationListener which is drawn as “MyLocationListener” above. The framework interface “IOConnector” describes a generic connector and the concrete class “BTConnector” included with the framework implements the facilities of connecting and communicating with a Bluetooth device. NMEAParser is a library used by the framework to parse NMEA sentences read from the GPS receiver. The class Location in the framework stores all Location specific information – coordinates, name, type (placeholder for pointer application specific data structure) etc. and its only purpose is as a aggregating the Location specific data. All operations over the Location objects are performed through the LocationProvider class which is the center of the framework. The following design decisions to make the framework generic and useful across a wide range of applications:

- Use of interfaces IOConnector and LocationListener to abstract out the connection and Location Event Listener parts from the framework which are either dependent on the connection type or are specific to the application.
- Since the application might have its own requirements on when and how frequently should the location be updated (e.g. to conserve power), assuming a fixed interval of update, or even assuming a periodic update might have been limiting. So LocationProvider provides a single function “RefreshLocation” that the application is expected to call when an update is desired.
- Defining what constitutes a Location was the most challenging decision. This application looked at the kind of scenarios where user would be using the framework. User might want to define a room as a location, or a building or a campus. From this user concluded that maybe cuboids would best represent a Location (region). But then the problem was what default size to choose. Should the application

assume a default size and let the user correct whenever it makes a bad guess. But then the Location object would keep changing which might not be desirable. Also the overhead required for this generic model both in terms of computational complexity, storage required as well as user interaction was too high. Here the application represents a location as cylinder whose base is specified by a center (latitude, longitude, altitude) and radius (horizontal tolerance) and height (vertical tolerance). To get user input for all parameters at Location creation, thus avoiding runtime complexity. This model also simplifies the implementation of operations for Location objects.

- Overlapping locations: It is possible for a user to define two locations that share a region of space. It could be the user's intention or this could be the result of overestimating one of the tolerance parameters. To permit the creation of such objects while providing a function to determine if two location objects overlap. If the application decides to disallow such objects all it need to do is use the check overlap function at construction time [14].

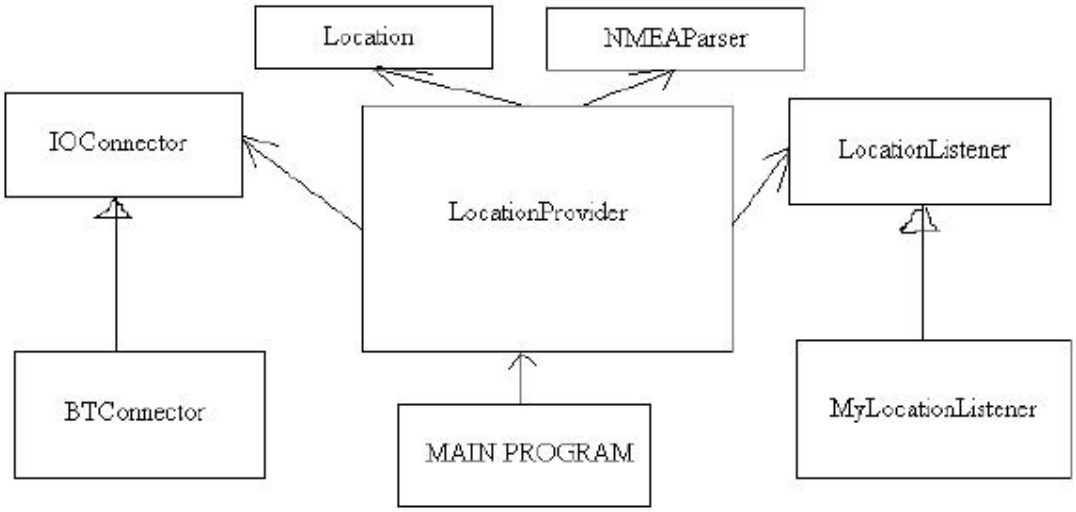

Figure 4.1 System Framework

Description of classes constituting the framework:

- Location: Provides storage for location specific information (name, coordinates, horizontal tolerance, vertical tolerance etc.). Also permits

a “type” parameter which can be used to store application-specific information.

- LocationProvider: Provides all kinds of operations for Location, maintains the current location information and is responsible for invoking the listener callback functions when location events happen.
- **IOConnector:** Interface for a generic connector. Has a method for Read/Write, checking the connection status.
- **BTConnector:** An implementation of IOConnector interface for Bluetooth. Provides extra methods for device discovery.
- **LocationListener**: Interface for a generic listener. Has event callback for LocationEntry, LocationExit, and LocationUpdate.
- **NMEAParser**: Provides parsing for GPS NMEA sentences and is used by LocationProvider.

5. PROJECT RESULTS

This application has a more community based approach when compared to Vehicle Tracker. No user may be deleted by other users, and every user can create and join groups. As administrator of an application of Buddytracker administrator can delete the complete application and thereby all users belonging to it, administrator can’t delete individual users. The reason for this is that unless created by the SiteAdmin every user in BuddyTracker has created his account by himself in the login-window. On the other hand the SiteAdmin should have the rights to delete users somehow, as every active user demands one license, hence this compromise.

BuddyTracker is meant to share your position in a community, or among friends and family. And user can of course see them if they choose to send their position.

5.1 USER ROLES IN BUDDYTRACKER

The different users in an instance of BuddyTracker are:

User

Everyone that's registered on an application is a user. Like in VehicleTracker a user can be active in many different Instances, but in BuddyTracker he can't be deleted by another user with admin-rights, only removed from a Group or a Buddy List.

Group Owner

This is a user that has created a Group, and he's the only one that can appoint or remove Administrator-rights and decide whether this Group is public or private. All registered users can create their own Groups.

Group Administrator

A user that's been given Administrator rights by a Group Owner.

Member

This is a user that's Member of a Group, he can leave the Group any time he wish, and can also be expelled by the Group Owner or a Group Administrator. To join a Group a user has to apply for membership.

BuddyTracker Web GUI

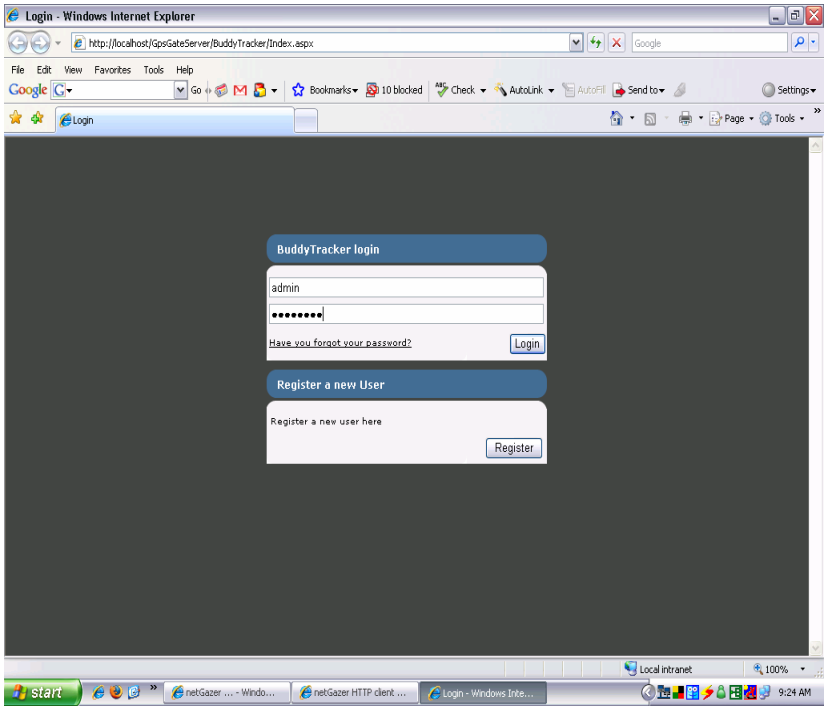

Figure 5.1 Login form for Administrator/user

This is the login window where a registered user can login to an application Instance of BuddyTracker, and a new user can register. When registering enter the user name, password, and an e-mail address that can be used to retrieve the password if it is forgotten.

Map/Home - with Buddy List and Group List

This is the first page user come to when logging in. It consists of 2 lists to the left, Buddy List and Group Lists. And to the right is the Map where user can see the positions of himself and his Buddies if user has Buddy List selected, or the Members of a Group in the Group List, if user is not the owner or a Member of any Groups the list will be empty.

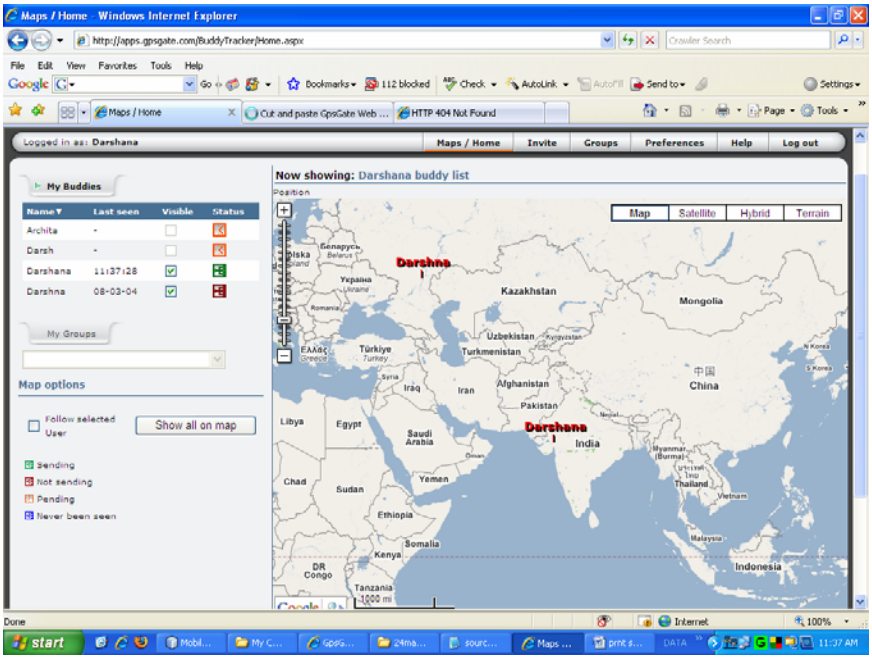

Figure 5.2 Position of Buddies on map

Sending-status

Looking on your Buddy List or a Group List when user have several other users on them user will see there is a color-coded system for sending-status.

Green = Online

Means a user have sent his position within the last 20 minutes.

Red = Offline

The user isn't currently broadcasting, either his GPS or his connection to the server isn't working, maybe he has no Internet-connection on his cell phone at the moment.

Blue = Never been seen

This user has never sent a position to this server, and therefore has no position on map either.

Orange = Pending

This user has applied to a Group but has not yet received a confirmation/rejection on the application. If accepted he will be positioned on the map and change in the Group List to whatever status

he is in. If a user loose the connection to the server there is a delay of 20 minutes until he is set as status Offline, this time frame is to prevent too many small changes in the status, as driving through a tunnel or switching between Wireless Networks can mean short temporary connection losses.

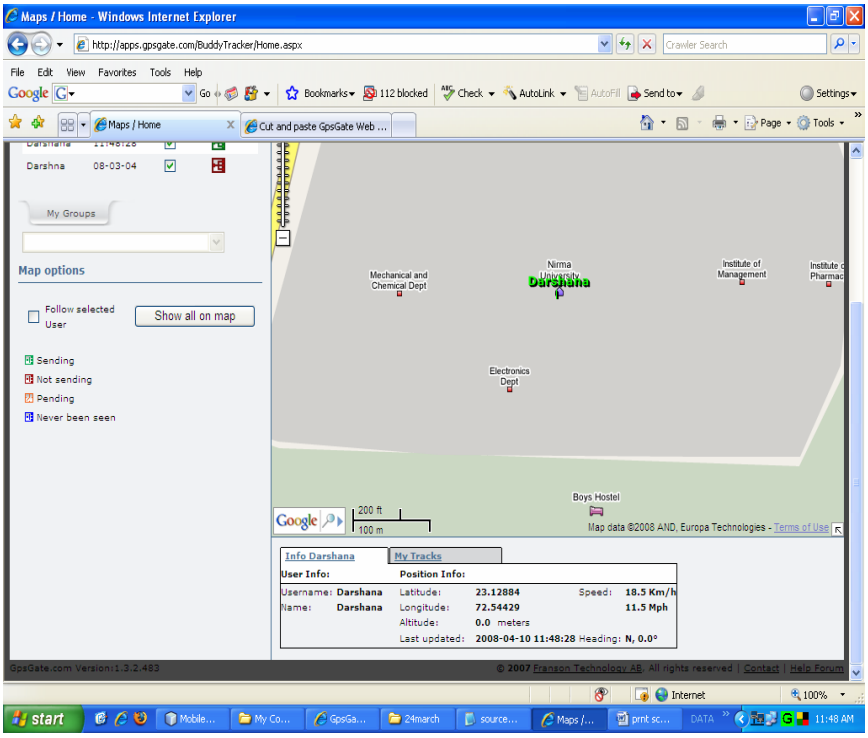

Figure 5.3 Buddy information

A user can broadcast his position manually by using the ClickPosition-feature. He selects his position on the small map by putting the marker there and clicking the Change-button. He can also set his position by writing it as longitude/latitude in the text-fields. ClickPosition is basically a way for a user to send his position when he currently is not able to send his position using the GpsGate Client. Note that using ClickPosition will send the position for this user until a new position is added either manually or by the GpsGate-client, but that as with the GpsGate Client not sending, after 20 minutes he will change status from Online to Offline.

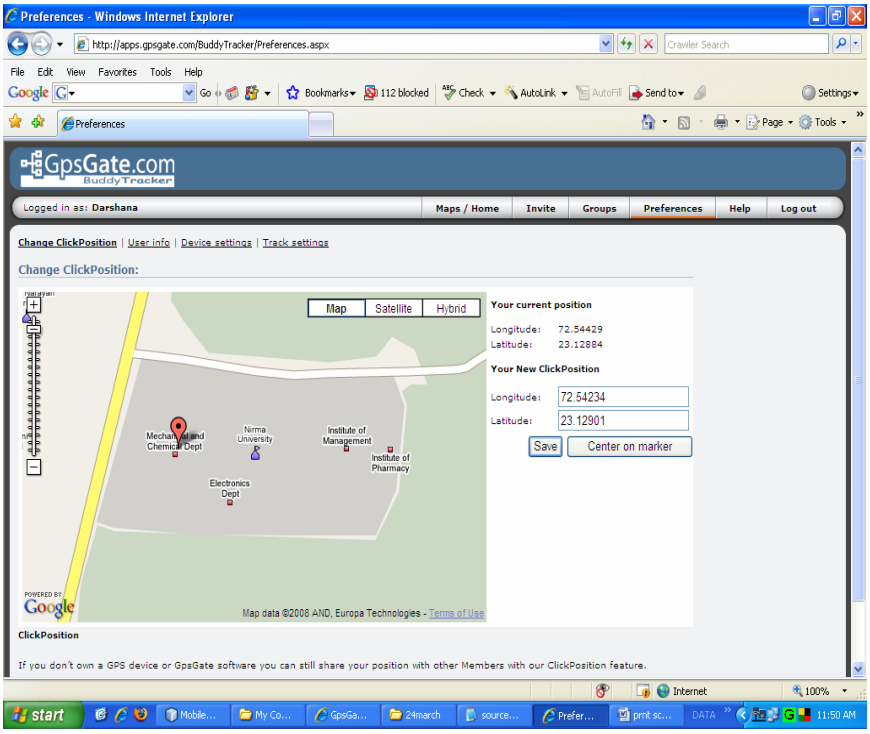

Figure 5.4 Preferences: Change of Buddy's Posotion

Tracks

The Tracks option in BuddyTracker gives the opportunity to see positions between two points in time as a track on the map. Of course to show tracks of buddy would have to send his position at the selected times. A Track basically consists of all the recorded positions (waypoints) with a straight line drawn between them. This means that the more often admin record buddy's position, the"smoother" the track will be. The track as seen on the map consists of a Start- and Stop marker, and a line between them. Admin can either select between recorded tracks in a list (always named after the start time), or manually set a start- and stop time. Admin show the track on the map by clicking on the Show Track-button, the track will be displayed and the map will automatically resize itself to show the whole track. Observe that for now there is a limit to 5000 waypoints for a track. It is always the user that is logged on to his own account that sees his tracks, and no one else's.

The time limit for one recorded track to stop and another one to start is two hours. This means if buddy send his position for two hours, stop sending for one hour, and then start sending again for another three hours, it will be available as one track in the list 6 hours long. If buddy have a gap in his positions, perhaps if buddy have traveled in a tunnel or momentarily shut of his GPS or the device with the GpsGate Client installed. These gaps will be filled in as a straight line between the two closest positions. Something that can disturb his tracks is the use of the function Click Position in the BuddyTracker GUI and GPS Simulator in the GpsGate Client, as these manual position-functions are treated as valid waypoints.

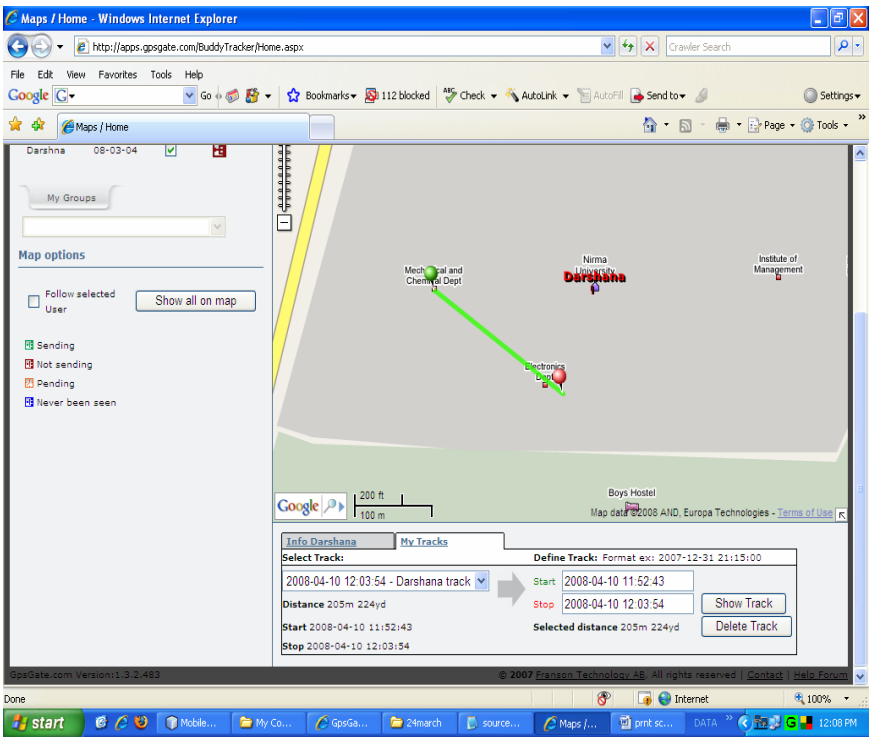

Figure 5.5 Recorded track of Buddy with Date / Time

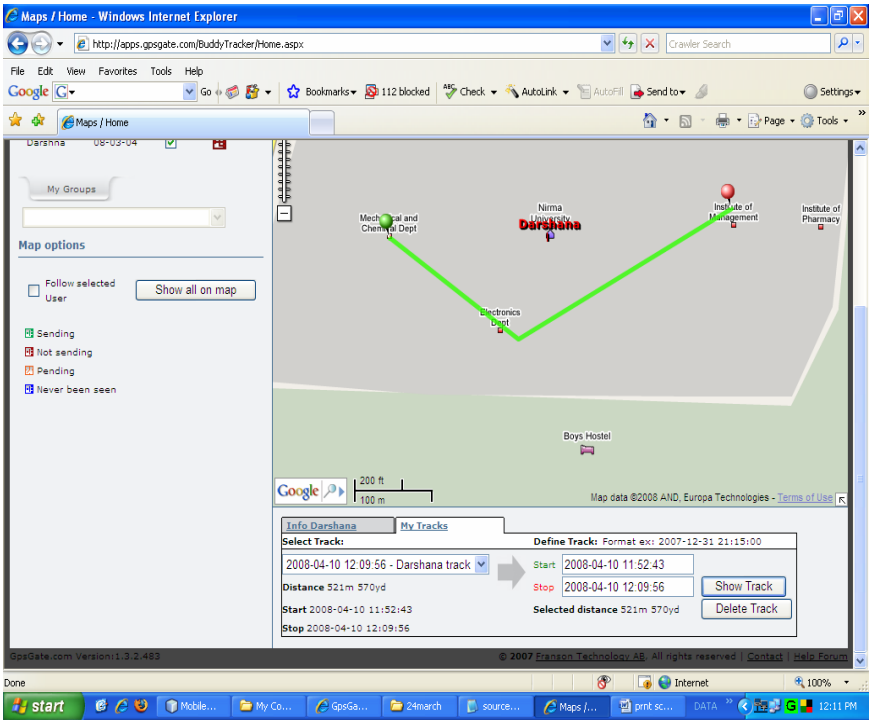

Figure 5.6 Recorded track of Buddy with Date/Time (2)

5.2 CLIENT SIDE RESULTS

Types of Client

- Windows laptop or stationary computer
- Windows Mobile phone or device (Pocket PC)
- Mobile Phone
- GPS tracking Device
- Blackberry

Testing in Windows OS/ Client side

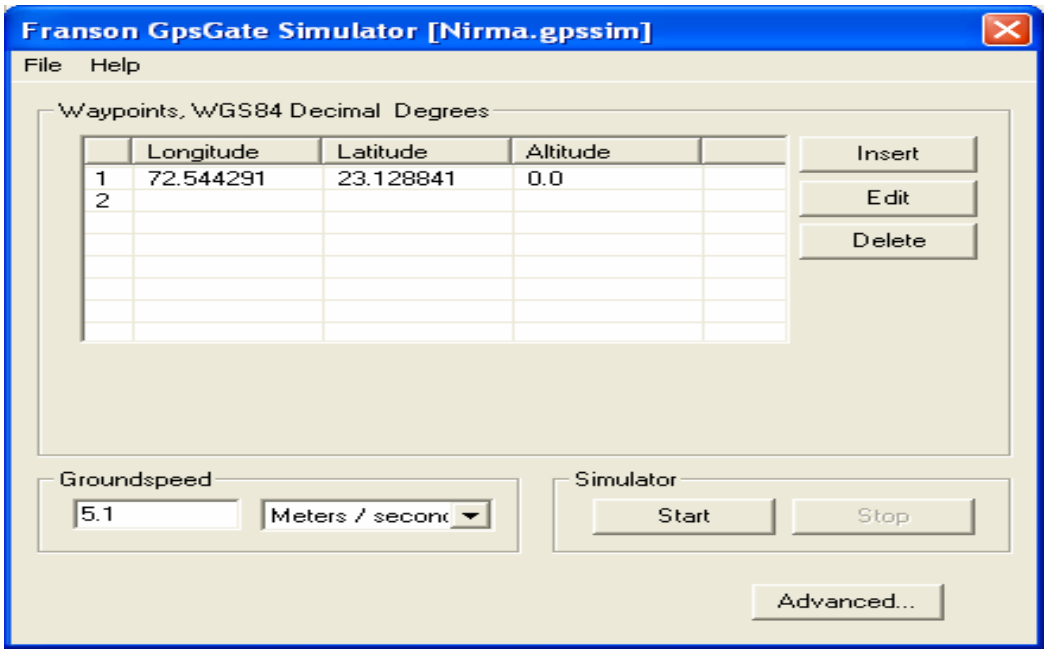

Figure 5.7 Input position manually and sending Longitude, Latitude and Altitude to server.

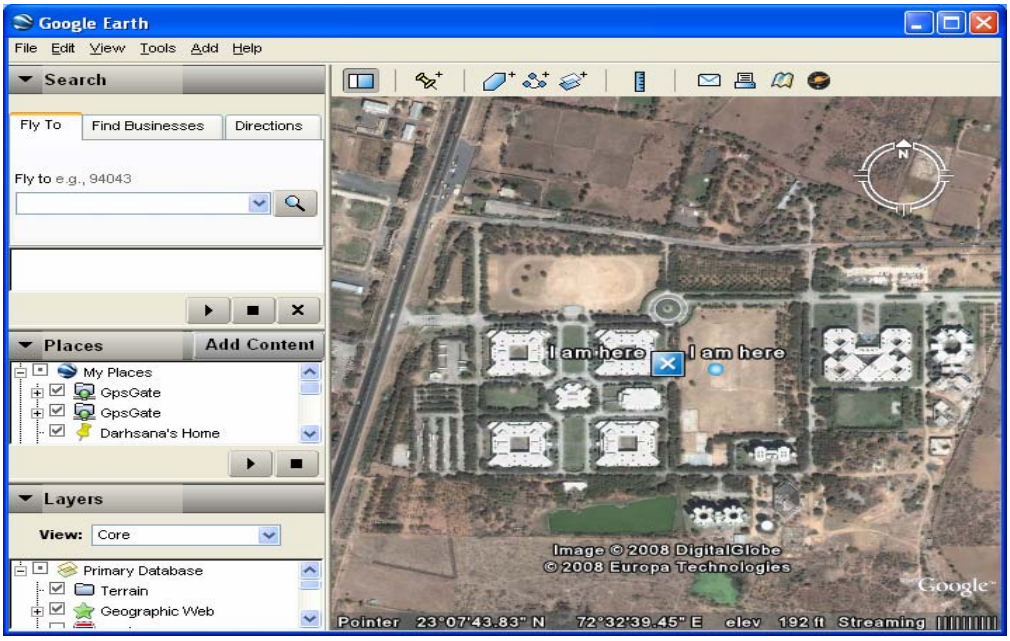

Figure 5.8 Output In Google Earth

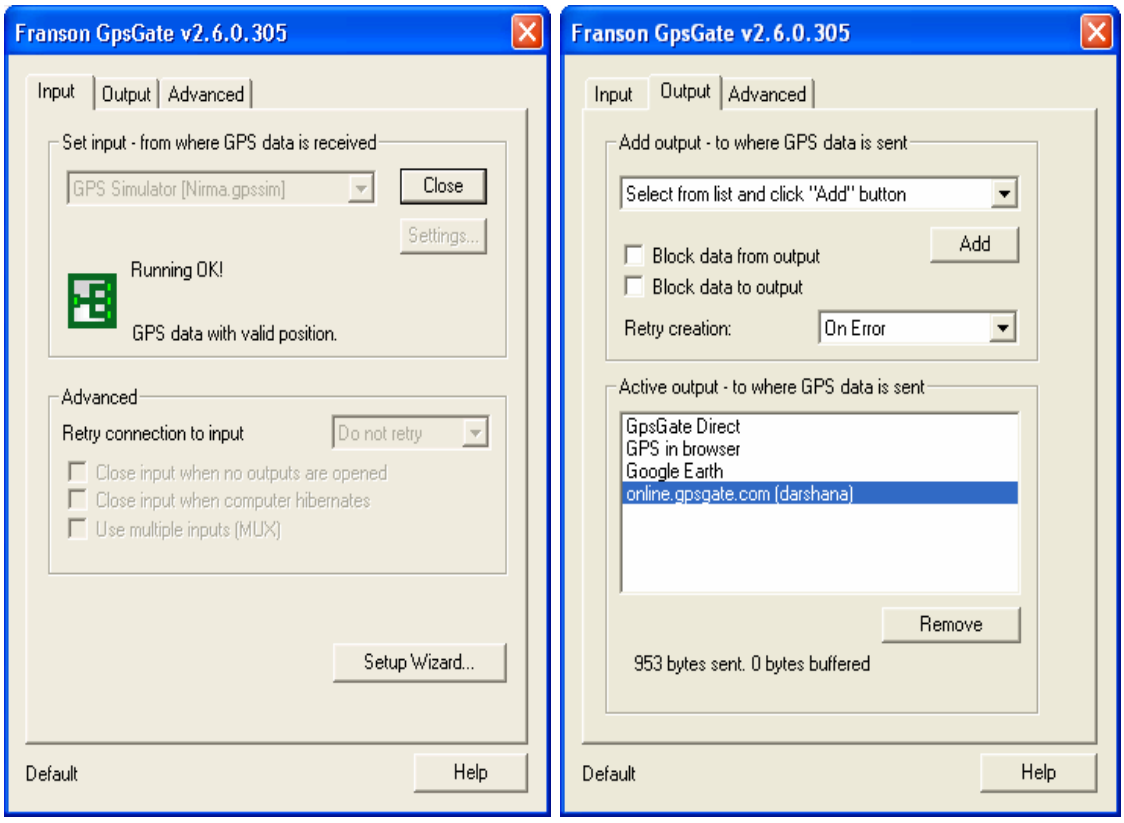

Figure 5.10 Input/Output in Server Setting

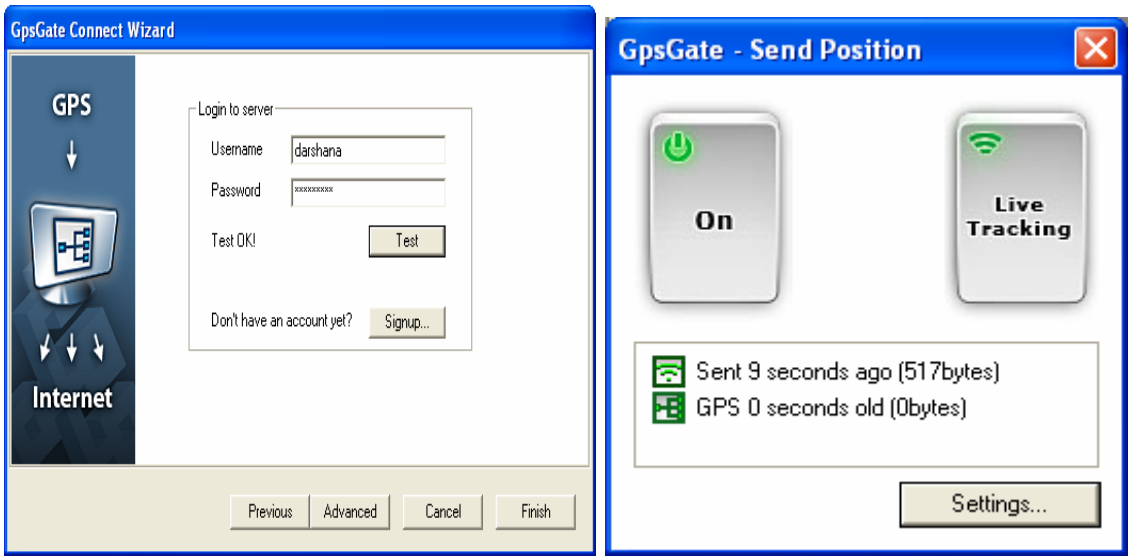

Figure 5.11 Connection to GPS server

Simulation in J2ME WTK 2.2

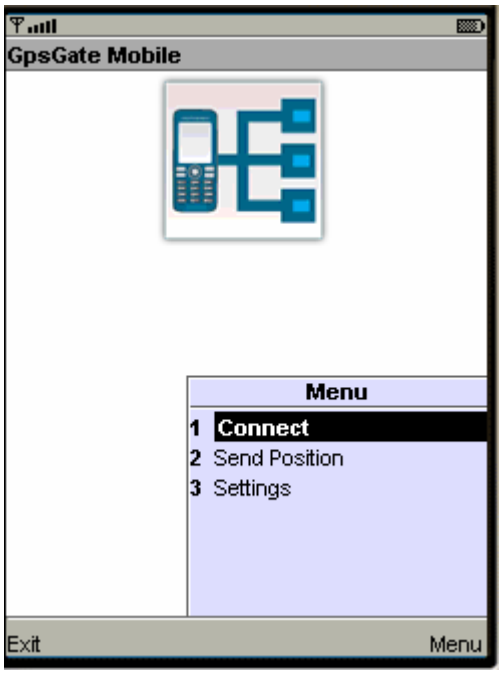

Figure 5.12 Connection to GPS server

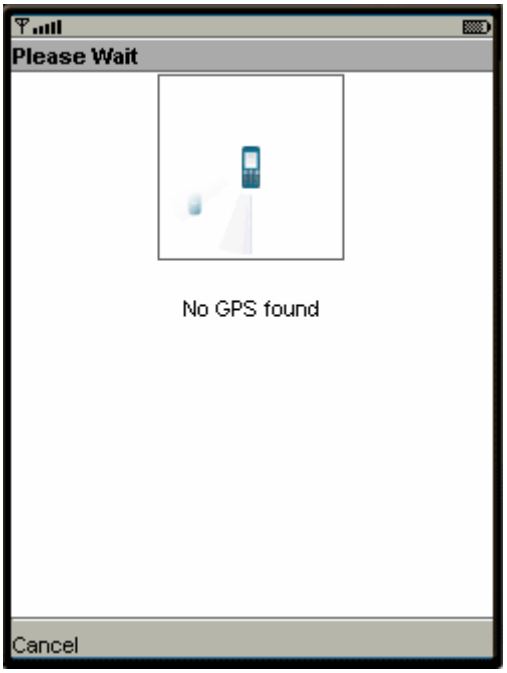

Figure 5.13 Finding GPS device

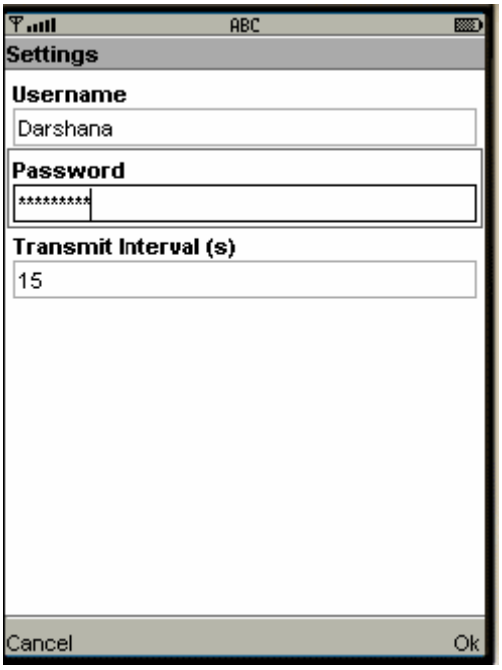

Figure 5.14 Settings in Client's Handset

5.3 INTERFACE TESTING

5.3.1 Interface: TerminalLocation

Request the location for a terminal.

5.3.1.1 Operation: getLocation

This operation is intended to retrieve the location for a single terminal. The accuracy requested is the desired accuracy for the response. The acceptable accuracy is the limit acceptable to the requester. If the accuracy requested cannot be supported, a fault will be returned to the application. If the accuracy of the location is not within the acceptable accuracy limit, then the location will not be returned, instead a fault will be returned. The URI provided is for a single terminal, not a group URI. If a group URI is provided, a fault will be returned to the application. If tolerance is indicated this affects the priority of accuracy, response time and maximum acceptable age.

Input message: getLocationRequest

Part name	Part type	Optional	Description
requester	anyURI	Yes	It identifies the entity that is requesting the information. The application invokes this operation on behalf of this entity. However, it should NOT be assumed that the application has authenticated the **requester**. If this part is not present, the requesting entity is the application itself.
address	anyURI	No	Address of the terminal device for which the location information is requested
requestedAccuracy	int	No	Accuracy of location information requested
acceptableAccuracy	int	No	Accuracy that is acceptable for a response
maximumAge	common:TimeMetric	Yes	Maximum acceptable age of the location information that is returned
responseTime	common:TimeMetric	Yes	Indicates the maximum time that the application can accept to wait for a response
tolerance	DelayTolerance	No	Indicates the priority of response time versus accuracy

Output message: getLocationResponse

Part name	Part type	Optional	Description
result	Location Info	No	Location of the terminal for which location information was requested

5.3.1.2 Operation: getTerminalDistance

This operation is intended to determine the distance of a terminal from a location. The URI provided is for a single terminal, not a group URI. If a group URI is provided, a fault will be returned to the application.

Input message: getTerminalDistanceRequest

Part name	Part type	Optional	Description
Address	anyURI	No	Address of terminal to check
Latitude	float	No	Latitude of the location to measure from
longitude	float	No	Longitude of the location to measure from
requester	anyURI	Yes	It identifies the entity that is requesting the information. The application invokes this operation on behalf of this entity. However, it should NOT be assumed that the application has authenticated the **requester**. If this part is not present, the requesting entity is the application itself.

Output message: getTerminalDistanceResponse

Part name	Part type	Optional	Description
Result	int	No	Distance from terminal to the location specified in meters

5.3.1.3 Operation: getLocationForGroup

The getLocationForGroup operation initiates a retrieval activity, where one or more terminals, or groups of terminals, may have their locations determined. The accuracy requested is the desired accuracy for the response. If the accuracy requested is not supported, a fault will be returned to the application. If the location retrieved is not within the acceptable accuracy limit, then the location data will contain a ServiceError.

If tolerance is indicated this affects the priority of accuracy, response time and maximum acceptable age.

The Web Service may return a result set that does not include complete information, allowing the Web Service implementation to choose to deliver a partial set of results to accommodate other conditions, such as avoiding timeouts. In this case, a result will be marked **NotRetrieved** for each address for which location retrieval was not attempted.

Input message: getLocationForGroupRequest

Part name	Part type	Optional	Description
requester	anyURI	Yes	It identifies the entity that is requesting the information. The application invokes this operation on behalf of this entity. However, it should NOT be assumed that the application has authenticated the **requester**. If this part is not present, the requesting entity is the application itself.
addresses	anyURI [1..unbounded]	No	List of URIs to get location for, including group URIs
RequestedAccuracy	int	No	Accuracy of location requested in meters
AcceptableAccuracy	int	No	Accuracy that is acceptable for a response in meters
maximumAge	common:TimeMetric	Yes	Maximum acceptable age of the location information that is returned
ResponseTime	common:TimeMetric	Yes	Indicates the maximum time that the application can accept to wait for a response
tolerance	DelayTolerance	No	Indicates the priority of response time versus accuracy

Output message: getLocationForGroupResponse

Part name	Part type	Optional	Description
result	LocationData [1..unbounded]	No	Set of results for the request

5.3.2 Interface: TerminalLocationNotificationManager

Set up notifications for terminal location events using geographical based definitions.

5.3.2.1 Operation: startGeographicalNotification

Notifications of location changes are made available to applications. The number and duration of notifications may be requested as part of the setup of the notification or may be governed by service policies, or a combination of the two.

If **checkImmediate** is set to true, then the notification will be set up, and then the current value of the terminal location will be checked. If the terminal location is within the **radius** provided and the criteria is **Entering** or is outside the **radius** and the criteria is **Leaving**, a notification will be sent to the application. This notification will count against the **count** requested. This addresses the case where the location of the device changes during the time the notification is being set up, which may be appropriate in some applications.

The **correlator** provided in the **reference** must be unique for this Web Service at the time the notification is initiated, otherwise a fault will be returned to the application.

If the **frequency** requested is more often than allowed by the service policy, then the value in the service policy will be used. If the **duration** requested exceeds the time allowed in the service policy, then the value in the service policy will be used. If the notification

period (**duration**) ends before all of the notifications (**count**) have been delivered, then the notification terminates. In all cases, when the notifications have run their course (by **duration** or **count**), an end of notifications message will be provided to the application.

Service policies may govern what **count** values can be requested, including maximum number of notifications allowed and whether unlimited notifications can be requested (i.e. either by not specifying the optional **count** message part or by specifying it with a value of zero). If the **count** value requested is not in policy, a fault will be returned.

The criteria will be met when the terminal enters the area defined as the circle of the **radius** provided around the point provided (**latitude**, **longitude**). The **trackingAccuracy** provided will determine how fine grained is the determination of where the terminal is located. A **trackingAccuracy** with a high value (coarse grained tracking) may result in more or less notifications (false or missed notifications) than actual entries and exits from the area defined.

Service policies govern what values can be provided for **trackingAccuracy**, including a minimum number (in meters) that can be requested. If the value provided is not within policy, a fault will be returned.

Input message: startGeographicalNotificationRequest

Part name	Part type	Optional	Description
reference	common: SimpleReference	No	Notification endpoint definition
requester	anyURI	Yes	It identifies the entity that is requesting location event notification. The application invokes this operation on behalf of this entity. However, it should NOT be assumed that the application has authenticated the **requester**. If this part is not present, the requesting entity is the application itself.
addresses	anyURI [0..unbounded]	No	Addresses of terminals to monitor
Latitude	float	No	Latitude of centre point
longitude	float	No	Longitude of centre point
Radius	float	No	Radius of circle around centre point in meters
TrackingAccuracy	float	No	Number of meters of acceptable error in tracking distance
Criteria	EnteringLeavingCriteria	No	Indicates whether the notification should occur when the terminal enters or leaves the target area
CheckImmediate	boolean	No	Check location immediately after establishing notification
frequency	common:TimeMetric	No	Maximum frequency of notifications (can also be considered minimum time

			between notifications)
Duration	common:TimeMetric	Yes	Length of time notifications occur for, do not specify to use default notification time defined by service policy
Count	int	Yes	Maximum number of notifications. For no maximum, either do not specify this part or specify a value of zero.

5.3.2.2 Operation: startPeriodicNotification

Periodic notifications provide location information for a set of terminals at an application defined interval. The accuracy requested is the desired accuracy for the response. If the accuracy requested is not supported, a fault will be returned to the application.

Input message: startPeriodicNotificationRequest

Part name	Part type	Optional	Description
reference	common: SimpleReference	No	Notification endpoint definition
requester	anyURI	Yes	It identifies the entity that is requesting location event notification. The application invokes this operation on behalf of this entity. However, it should NOT be assumed that the application has authenticated the **requester**. If this part is not present, the requesting entity is the application itself.
addresses	anyURI [1..unbounded]	No	Addresses of terminals to monitor
RequestedAccuracy	int	No	Accuracy of location requested in meters
frequency	common:TimeMetric	No	Maximum frequency of notifications (can also be considered minimum time between notifications)
Duration	common:TimeMetric	Yes	Length of time notifications occur for, do not specify to use default notification time defined by service policy

5.3.2.3 Operation: endNotification

The application may end a notification (either type) using this operation. Until this operation returns, notifications may continue to be received by the application. An end of notification (Location**EndRequest**) message will not be delivered to the application for a notification ended using this operation.

Input message: endNotificationRequest

Part name	Part type	Optional	Description
correlator	string	No	Correlator of request to end

5.3.3 Interface: TerminalLocationNotification

Notification interface to which notifications are delivered.

5.3.3.1 Operation: locationNotification

When the location of a monitored device changes a notification is delivered to the application with the new location information. If a group identifier was used, the terminal device URI is provided, not the group URI.

Input message: locationNotificationRequest

Part name	Part type	Optional	Description
correlator	string	No	Correlator provided in request to set up this notification
data	LocationData [1 .. unbounded]	No	Location information for terminal
criteria	EnteringLeavingCriteria	Yes	Indicates whether the notification was caused by the terminal entering or leaving the target area. (This part is provided for geographical notifications, not for periodic notifications)

5.3.3.2 Operation: locationError

The location error message is sent to the application to indicate that the notification for a terminal, or for the whole notification, is being cancelled by the Web Service.

Input message: locationErrorRequest

Part name	Part type	Optional	Description
correlator	string	No	Correlator provided in request to set up this notification
address	anyURI	Yes	Address of terminal if the error applies to an individual terminal, or not specified if it applies to the whole notification
reason	common:ServiceError	No	Reason notification is being discontinued

5.3.3.3 Operation: locationEnd

The notifications have completed for this correlator. This message will be delivered when the duration or count for notifications have been completed. This message will not be delivered in the case of an error ending the notifications or deliberate ending of the notifications (using **endNotification** operation).

Input message: locationEndRequest

Part name	Part type	Optional	Description
Correlator	string	No	Correlator provided in request to set up this notification

5.4 POSITION ERRORS

5.4.1 Ephemeris errors

Each satellite transmits its position in the navigation message modulated onto the L1 signal. This orbital path for each satellite cannot be perfectly modelled and hence slight inaccuracies may occur with the indicated satellite position.

5.4.2 Clock errors

Even though atomic clocks are used by GPS satellites, a slight error results over the course of a single day. The atomic clock inaccuracy is approximately between 8.64 and 17.28 ns each day. This leads to errors ranging from 2.59m to 5.18m (NetLibrary Inc. 2002).

The clock accuracies are monitored frequently from the control segment and this offset is uploaded to the respective satellite. Each satellite then transmits this correction in the navigation message.

To remove the clock drift, a receiver may apply the clock correction data received in the navigation message. This will remove a portion of the clock error but may still result in range errors of a few meters.

5.4.3 Atmospheric errors

The Earth's atmosphere delays the transmission of the GPS signal. Since pseudorange measurements depend up the transmission time of the signal, an error in the approximated distance is induced. To remove this error, the speed(s) at which the signals propagate through the atmosphere must be known. The Earth's atmosphere is comprised of two essential layers that affect the transmission of GPS signals. These are the ionosphere and troposphere.

5.5 Privacy

Location information can be highly sensitive:

Target marketing: Mobile users' locations can be used to classify customers for focused marketing efforts.

Embarrassment: One customer's knowledge of another's location may lead to embarrassing situations.

Harassment: Location information can be used to harass or attack a user.

Service denial: A health insurance firm might deny a claim if it learned that a user visited a high-risk area.

Legal restrictions: Some countries regulate the use of personal data.

The user's location may not always be available, for any of several reasons. The device is cut off from any of the location methods it supports, in a tunnel or on an airplane for example. No location provider that the device supports is available. Depending on the method used, determining the location may take a long time. Keep the user informed. Location service fees, typical of

network-assisted location methods, can add up quickly, so don't overuse fee-based services. Be sensitive to privacy concerns. Tell customers about the information being collected on them and how it will be used. Offer customers the choice of what location information to disclose, and when appropriate an option not to participate. Allow customers to review their permission profiles so that they know what they are permitting. Protect location information so that it cannot be accessed by unauthorized persons.

GPS Phones:

- P535, P526 ASUS
- BlackBerry8800 (GSM/GPRS), 7100i (iDEN), 7520 (iDEN) BlackBerry
- M700, X500+ E-TENX500 glofiish
- FALCOM -MAMBO 2
- Pocket LOOX T Series Fujitsu Siemens
- GSmarti300, GSmartq60 Gigabyte
- iPAQhw6900 series HP
- HTC P3300, HTCHTC P3600HTC Advantage X7500HTC S420, HTC S730, HTC P4550, HTC P5500
- JAQ4 i-mate
- ImCoSys-I-SM1
- A701, A501 Mio Technology
- MC35, E815 Motorola
- Nokia N95, Nokia 6110, NavigatorNokia E90 Communicator
- SGH-i600 (GPS version) Samsung

6.
CONCLUSION

Developing a framework for mobile positioning client on PDA platform was inspired by availability of J2ME development environment and increasing number of MIDP applications on small communication devices, limited by memory and processing power and with the support for HTTP. With the rapid development of next generation networks and end-user devices as well as location aware computing, it is becoming obvious, that the location based services will have one of the leading roles in the mobile communications of the future.

The implementation of the GPS server can on a whole be regarded as a success. Even though one part of the assistance data can't be determined properly the server's performance exceeded expectations. The GPS server meets the performance requirements thanks to the supervisor/worker implementation. The communication between the MLC and terminal was implemented successfully.

Calculations: All vector, matrix, and positioning algorithms were implemented and tested. When a complete positioning was made using real data in the correct format, the retrieved position was within the required margin of error. The simulation tools The SMLC simulation tool was used both as a test tool of concurrent communication and performance of the GPS server. In the future, it will be possible to use the SMLC simulation tool to measure the response time of a complete positioning communication sequence of a live GPS server. Due to lack of terminals that supported SUPL communication the WAP-push component weren't properly tested.

Trace tool: During the early stages of the project the purpose of the application part was more or less arbitrary. Consequently, it was decided that a map display and positioning was to be developed. It was to consist of a web application and a mobile application that communicate with a map data server. The mobile application was decided to be bare bone, while the web application should over account features, allowing the user to store and track mobile phone number etc… These ideas were however revised roughly halfway through on account of Mobile Arts requesting a tool for tracing positioning. The trace tool now implements much of the functionality requested by Mobile Arts.

To conclude, this project has identified appropriate communication mechanism for delivering location data to the new generation of smart phones such as Symbian, J2ME and others. Besides that, it has also defined the issues, barriers and opportunities to the deployment of device based location service applications.

References:

1. “Positioning Simulator for Location-based Service”, Anusuriya Devaraju & Simon Beddus, Research and Venturing, UK.

2. “J2ME and Location-Based Services”, http://developers.sun.com/mobility/apis/articles/location/

3. Java Specification Requests, JSR 179 Standards

4. “Foundations of Location Based Services”, by Stefan Steiniger, Moritz Neun and Alistair Edwardes

5. Schiller, J. H., 2003. *Mobile Communication*. Addison Wesley.

6. “Assisted GPS Solution in Cellular Networks” By Gidon Lissai ,November 2006

7.” Location Based Information Service”, by Tahani Siddik

8. “World Intellectual Property Organization”, http://www.wipo.int/pctdb/en/wo.jsp?WO=2001%2F078427&IA=WO2001%2F078427&DISPLAY=DESC

9. “Location-Based Services Strengthen the Strategic Position of Mobile Operators”, by Faggion, A. Trocheris, Strategy white paper.

10. “Developing A-GPS as a Student Project “, by Bahram Bahar, Adam Bolcsfoldi, Jonas Falkevik, Department of Information Technology, Uppsala University.

11. “JSR 179 Location API for J2ME™version 1.0.1”, http://www.forum.nokia.com/info/sw.nokia.com/id/f1957ea1-5a79-406e-be49-306cfd15d2da.htm

12.” Innovative applications of GPS: Mobile assistance for public transport networks “, by Cameron A. Lindberg, Curtin, University of technology.

13. “Targeting GPS - Integrating J2ME, GPS, and the Wireless Web”, By Shane Isbell

14. “Location Aware Programming Framework”, CSE237B Course Project, Fall 2004 by Apurva Sharma