LOCATION-BASED MOBILE E-COMMERCE FOR TOURIST SERVICES

PHIWA DOUGLAS MZILA

2007

LOCATION-BASED MOBILE E-COMMERCE FOR TOURIST SERVICES

Phiwa Douglas Mzila

20000847

A dissertation submitted in fulfillment of the requirements for the degree of

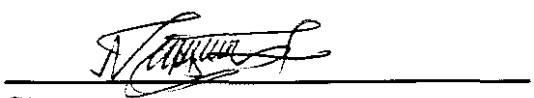
Master of Science (Computer Science)

Department of Computer Science, Faculty of Science and Agriculture,
University of Zululand

2007

DECLARATION

This dissertation represents research work carried out by the author and has not been submitted in any form to another university for a degree. All the sources I have used have been dully acknowledged in the text.



Signature

DEDICATION

To my late sister Philile Rachael Mabaso.

ACKNOWLEDGEMENTS

I would like to express my gratefulness to my supervisor Prof. M.O Adigun and co-supervisor Dr. S.S Xulu for their academic guidance. A special word of thanks also goes to the rest of the Computer Science department staff members, including Dr. O.J Emuoyibofarhe, and to my fellow research students from the University of Zululand, for their valuable comments and contributions.

I wish to thank Mr. Klaas Kabini who gave me assistance and support during the last part of this project. All the best my brother.

I am also thankful to the National Research Fund (NRF) and the Centre for Mobile e-Commerce for funding my studies. Without their financial support, this work would not have been completed.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

There are many instances where tourists become strangers in foreign countries and have no way of discovering available services in place they visit. These services, which are known as location based services, are normally available to only subscribers of those services. For mobile users who are not subscribers this may be unfair, especially if a mobile user is new in that environment, as he or she remains unaware of services that are available. This research proposes a dynamic service discovery mechanism whereby every mobile user is capable of service discovery irrespective of being a subscriber or not. We developed a model for location-based service discovery in a mobile environment. With this model we were able to show how components of a location based application server in mobile commerce application could be modelled in order for them to perform their task, and how services could be advertised in the vicinity of the network coverage area by the LBAS so that the user who is entering that area would be able to access services that are available to him/her We then implement a prototype of the model to showcase the usefulness of the proposed model. Our last objective was to evaluate performance of the prototype system. We conducted usability testing where we asked different people, from different faculties, to take a tour of the system. This group was then interviewed to evaluate the *user-friendliness* and the easiness *to learn* testing of the system. In both aspects, the users were satisfied with system.

LIST OF ABBREVIATIONS

ADT	-	Abstract Data Types
BTS	-	Base Transceiver Station
CComm	-	Centaurus Communication Protocol
CDMA	-	Code Division Multiple Access
CDPD	-	Cellular Digital Packet Data
CPM	-	Context and Profile Manager
DA	-	Discovery Agent
DDM	-	Data Domain Manager
DIA	-	Data Interface Agent
ECC	-	Emergency Control Centre
GIS	-	Geographic Information Systems
GPRS	-	General Packet Radio Service
GPS	-	Global Positioning System
LBAS	-	Location Based Application Server
LBES	-	Location Based Emergency Services
LBS	-	location-based service
MDM	-	Mobile Data Management
MNCA	-	Mobile Network Coverage Area
NCA	-	Network Coverage Area
NSDI	-	National Spatial Data Infrastructure
NSF	-	Nearest Service Finder
OGC	-	Open Geospatial Consortium

PC	-	Personal Computer
PDA	-	Personal Digital Assistant
POI	-	Points of Interest
PP	-	Publishing Processor
RPC	-	Remote Procedure Call
SDMS	-	Spatial Database Management System
SDP	-	Service Discovery Protocol
SDR	-	Service Description Repository
SLM	-	Salutation Manager
SPDE	-	Service Publishing and Discovery Engine
SQL	-	Sequence Query Language
SSPR	-	Service Spatial Properties Repository
TM	-	Transport Managers
UDDI	-	*Universal Description Discovery and Integration*
UI	-	User Interface
UML	-	Unified Modeling Language
WAP	-	Wireless Application Protocol
WDA	-	Wireless Domain Agent
WiFi	-	Wireless Fidelity
XML	-	Extensible Markup Language

GLOSSARY OF TERMS

Agents – Are software entities that carry out some set of operations on behalf of a user or another program with some degree of independence or autonomy, and in so doing employ some knowledge or representation of the user's goals or desires. [IBM]

Service Broker - A service broker is an object interface that manages service requests from, and responsive services provided by, a plurality of clients and servers, respectively, which may reside on different hardware platforms and operating systems and may be connected to computer networks having different network architectures and associated communications protocols

Emulator - An emulator duplicates (provides an emulation of) the functions of one system with a different system, so that the second system behaves like (and appears to be) the first system

Location - geographical location or a specific position or point in physical space

Location-based services – The term location-based services (LBS) denotes applications integrating geographic location (i.e. spatial coordinates) with the general concept of services

e-Commerce - Electronic commerce denotes the seamless application of information and communication technology from its point of origin to its endpoint along the entire value chain of business processes conducted electronically and designed to enable the accomplishment of a business goal

M-Commerce – M-Commerce refers to any transactions, either direct or indirect, with a monetary value implemented via a wireless telecommunication network

Mobile User - Any mobile device user

Points of interest - A Point of Interest (POI) is a location on the surface of an object in given vicinity

Pull/Pull service technology – The ability of users to pull or discover services and suppliers/providers of services to push or advertise their services in a defined coverage area

Repository - A database of information about applications software that includes author, data elements, inputs, processes, outputs and interrelationships

CHAPTER ONE

INTRODUCTION

1.1 OVERVIEW

The growth of e-commerce together with the growing acceptance of mobile devices have directly contributed to the emergence of mobile commerce. Mobile commerce (m-commerce) may be described as conducting e-commerce transactions over a wireless internet. According to Ratsimor, *et al,* (2003) the ability to discover services in a *"given context"* is one of the most critical requirements in m-commerce. An important component of a user's context is the *current location of the user.* Knowledge about the location of a mobile user enables provision of services that are relevant "right here and right now" (Katasonov, 2003). Location-based *m*-commerce incorporates the following features to e-commerce: location-aware technologies, wireless connectivity, and locationalized web-based services to support the processing of location-referent interactions and transactions. Location-referent interactions and transactions are those in which the geographic proximity of the interacting parties is an important transactional consideration (Wyse, 2005).

A mobile phone network operates through a network of local transmitter base stations. Anytime the mobile phone has to send or receive a call or message, it does so via the nearest local transmitter, the geographical location of which

is known to the network operator. It is possible to determine the location of any switched-on mobile phone with reasonable accuracy. This location facility presents a commercial opportunity. The user of a mobile phone could, for example, request a service provider to suggest restaurants or garages in the vicinity of his/her current location. The service provider will use the approximate location of the user to search for the nearest geographical match in their database. The requested information is then sent back to the caller in the form of a message, or even using 3G, a complete map may be sent. This situation has practical applicability in tourism, for example, to locate accommodation, restaurants, etc.

Consider the following scenario that best illustrates how location based services are applicable in real life (adopted from [Boucher, 2005]). A tourist from the USA arrives in Richards Bay by a plane one late Saturday afternoon. Her cellular phone had been switched off. Because of her other commitments, she could not book a hotel room in advance. She gets off the plane and immediately switches on her mobile phone and worries about a suitable place to rest, probably to eat and sleep. She then uses her mobile phone to browse and find out about hotels and other points of interest (POI) around her vicinity. After getting the list of hotels and its information, like telephone numbers she then phones the hotels to book for a room. As she books a room with a single key press, the service provider offers to send a taxi to pick her

up at her current location. She confirms, then sits down and resumes reading a novel whilst waiting for the taxi to arrive.

This scenario describes a rather straightforward use of a typical location-based service (LBS). Finding information about all the different things to do, places to visit, sights to see, places to eat and sleep, and so on, may be difficult and time consuming. Providing this kind of information to tourists can help make their visit a positive experience. It can also encourage tourists to take advantage of services and products offered by the LBS providers that they otherwise may have been unaware of. Obviously, the financial rewards may be considerable (Statistisk, 2005).

A typical LBS request operates on four key elements (You-Heng Hu et al, 2005), namely

- The user's location, obviously.
- Locations of various POI – such as hotels and restaurants.
- Information about each POI – for example, hotel vacancies and costs.
- User context and preferences. (Here the context might be "on holiday, traveling on foot" and preferences might contain his desired hotel accommodations and maximum costs.)

The above key elements play a crucial role in realizing the satisfaction of users in discovering available services within a given vicinity, since the

location of the user will be identified and the information about available services will be stored in the database through which the user's context and preferences in the request can be accessed.

1.2 STATEMENT OF THE PROBLEM

One of the paradoxes in LBSs is that they can be most useful when they are most difficult to implement, that is, when the user is not in his usual environment [Boucher, 2005], e.g. a tourist in a foreign country. Therefore the challenge facing LBS providers is to make sure that their services are properly advertised, so that any potential client will be able to find them easily. The major concern with systems currently employed for location based services in mobile environment is that, when a mobile user moves from one location to another, services available in the new location which are unavailable in the previous location are not automatically provided until the user registers with the system and subscribes to such services. This limits the patronage of such services to only registered users. Hence, there is a need of looking at ways of advertising available services to both registered and unregistered users. A mechanism should be provided to enable a mobile user to discover services available in a given vicinity and to allow a service supplier to advertise his/her service.

1.3 MOTIVATION

Tourism is one of the promising sectors in terms of boosting the economy and it is where the significance of mobile commerce can be optimized. Tourists, as mobile device users, need information about places to visit, where to eat, entertainment, etc. This information is typically found in guidebooks and on the Internet, requiring the tourist to actively search for relevant services in terms of location. Instead, this information can be stored on a database linked to mobile application server. The services are then advertised to suit the current location of each tourist in a mobile network environment. The mobile application server accessed using mobile devices acts as a gateway for tourists to the providers of tourist services.

1.4 RESEARCH GOAL AND OBJECTIVES

The goal of this research is to develop a service discovery mechanism for location based mobile application services suitable for mobile commerce environment. In order to realise this goal the following objectives were identified:

a) Define architectural requirements for a LBS for a m-commerce based on a publish/subscribe paradigm;
b) Design a system that is able to recognise the presence of a mobile device in a given location;
c) Develop a model for LBS with location-awareness.

d) Implement a prototype of LBS model as a proof of concept; and

e) Evaluate performance of the prototype system.

1.5 RESEARCH METHODOLOGY

The methodology adopted in this research is both theoretical and formulative. It begins with a literature review on existing service discovery and services publish/subscribe mechanism models with a view to understanding usage context, suitability for purpose, and interface styles. This is followed by the construction phase where knowledge gathered from the literature becomes the foundation for the proposed model. The third step entails analysis, design and implementation of a prototype as a proof of concept. Finally, the model is evaluated for performance using the prototype implementation as a test-bed.

1.6 ORGANISATION OF THE DISSERTATION

The remainder of this dissertation is organised as follows: Chapter two is a review of relevant literature on location based service application models, and serves as the basis of the location based application server model proposed in this dissertation for mobile commerce. In chapter three, methodology and model development are presented. Chapter four is the implementation of the nearest service finder system as a prototype for providing usefulness of the LBAS. Finally, chapter five presents the conclusion which consists of how the research objectives were achieved, the limitations of the results model developed and suggestions for further work.

CHAPTER TWO

LITERATURE REVIEW

2.1 INTRODUCTION

Tourists are the main consumers of location based mobile commerce services, and the easy availability of mobile devices has made it possible for most people to access these services. Tourism is among the first industries taking advantage of LBS. The main reason for this is that tourism can benefit from the use of mobile technology that provides new services to travelers on the move. The primary functions of LBS for tourism are the localization of persons, objects, places, and routing between these, search for objects in proximity such as restaurants, shops, hotels, or any other points of interest, and information about traveling conditions, such as traffic-related data. The conventional application areas of LBS include mapping, tracking, routing and logistics, electronic yellow pages, data collection and public safety (Beaulieu & Cooper 2001, Maguire 2001, and Zipf & Malaka 2001). However, with the maturing technology in mobile devices, like nodes that are equipped with multimedia capabilities, faster data transfer, cameras, GPS, wireless links, etc, location based services are likely to be in demand. Hence the study of location based services has become very important

There is a huge literature that exists on the study of location based service related issues. Early studies of potential location-based applications can be found in the work done by Dommety, et al (1996). Business opportunities in location based services and their benefits to mobile users are discussed by Rao and Minakakis (2003), and Barnes (2003). An overview of technologies involved in LBS can be found in Hjelm (2002) and Jagoe (2003). A discussion of architectural issues in LBSs is given in Jose et al (2003) and Nicklas and Mitschang (2004). Tilson et al (2004) discuss strategic issues related to an LBS development and service portfolio selection. The issue of dependability in LBS is addressed by Kaasinen (2005). Articles discussing success factors of LBSs can be found in Grajski, et al (2003) and Bisdikian, et al (2001). Unni and Harmon (2003) examine models for LBS in m-commerce. They discussed factors like positioning accuracy, privacy protection, usability of interfaces, price-value correspondence, personalization and many others.

A survey of literature on location-based service applications reveals that up to now LBS related investigations have been carried out not as an object of investigation on its own, rather as peripheral issue. These include work reported in Brookes, et al (2005), Kwon, et al (2004), (MDM. 2004), (Strang and Linnhoff-Popien, 2005). In these works the use of location information tends to be considered as an add-on feature to their basic object of investigation. Location information is thus included as a feature that would increase efficiency, enable personalization, add value, etc. There is generally

an overemphasis on the positive utility side of LBS. The risk of the user's location privacy violation is one of the issues related to the potential negative consequences of using LBS (for a discussion see Markkula 2001). Our interest in this work is in the delivery of location based services to tourists as they are in motion. Tourism, as the sector that commercializes location based services, acts as a basic platform for the real application of mobile commerce services.

In the next section we discuss Location Based Service Application Models from the point of view of how successfully they have incorporated the following features: push and pull technology, automatic location of mobile device, mobility and handling of network disconnections and database management. In section 2.3 we introduce our proposed model that incorporates all the above desirable characteristics.

2.2 LOCATION BASED SERVICE APPLICATION MODELS

In this section we discuss some of the existing schemes with regards to how the following issues are handled: push and pull technology, automatic location of mobile device, mobility and handling of network disconnections and database management. Each of these issues is discussed briefly in this section.

a) *Push and pull service technology:* This deals with ability of users to pull or discover location based mobile services and that of suppliers/providers of the services to push or advertise their services in a defined coverage area. Pull and push service technology is defined by the manner in which information is delivered during interaction between the user and the system. In pull services information is delivered directly from a service provider to a requesting user. This is similar to accessing a website in the Internet by filling 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 the closest restaurant. In push services, delivered information is either not or is indirectly requested by the user. Push services are activated by an event, which could be triggered if a specific area is entered. 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 triggered by changes in if weather conditions. The main difference between push service and

pull service is that in push service it is a must that the user subscribes to a system before he can use it.

b) *Automatic location of mobile device,* Chen and Kotz (2000): Ability of the mobile device to be aware of its current location. There are two general approaches to make the mobile device aware of its current location. Either the system tracks the location by monitoring signals from mobile devices and the mobile device queries a central database to get the current location, or the mobile device passively listens to signals from the cell base station and queries a local database for its current location. In the latter case, if the mobile device only queries a local database for location, it has complete privacy and it can choose to advertise its current location to the world or only to selected third parties.

c) *Mobility and handling of network disconnections,* Castro, et al (2001): This considers the means of preventing frequent interruptions that occur in the mobile environment due to network disconnections during transaction. It is, therefore, necessary to examine how models being reviewed in this work handle this challenge.

d) *Database management* (José Manuel, 2006): Design of information repository as the storage media needs examination of its concepts in mobile computing. Spatial database management is our primary concern in the literature review for this work. Spatial databases have been an active area of research for over a decade, addressing the growing data management and analysis needs of spatial applications such as Geographic Information Systems (GIS). Spatial database management systems aim to make spatial data management easier and more natural to users or applications.

In the subsections below, we discuss how the above issues have been handled in the most representative location based service application schemes. Location based service application models discussed below include Location Based Emergency Services model (LBES) by Gaurav Singh, (2004), LBS Publishing and Discovery Platform by You-Heng Hu et al (2005), Agents2Go by Olga Ratsimor et al (2003), Konark by (Sumi Helal et al, 2003), and Sanitation in Diplomarbeit (2002).

2.2.1 Location Based Emergency Services Model

The Location Based Emergency Services (LBES) model developed by Gaurav Singh, (2004) is one of the models that provide location based services to mobile users in relation to the provision of emergency services. Our interest in LBES model is on its implementation of the *pull and push* feature, and the

way the *database management* feature is implemented. These are the characteristics which are successfully incorporated in this model. As we mentioned in section 2.1, when reviewing a particular scheme, we only discuss those features which are successfully implemented.

In LBES the mobile user initiates the transaction process by specifying his or her request using a mobile device, which is a pull service. Hence we look at how the pull service technology has been applied in LBES. The emergency service initiated by a mobile user could be of any type from man-induced to natural, but every emergency requires a quick and focused response without which the life and/or property of common man may be at risk. The LBES allows users to query the system about their own location, nearest emergency management centre, available services and resources provided by a service supplier. Hence, a user, from emergency site might ask the following possible questions:

- Where am I? (Static)
- Where is the nearest hospital from my current location? (Static)
- Which is the best route to reach there? (Dynamic)

These queries can be executed based on their nature like static query or dynamic query. By dynamic here we mean real time information, that is, when the user sends the query 'How do I reach there?' Then the system or the

mechanism should provide the best possible route based on the dynamic or real time parameters in that area such as traffic conditions, weather etc.

In LBES there could possibly be three main players that is, the user (who reports the incident), the mobile operator or the wireless network provider (helps in proper communication and routing of the call to the nearest emergency centre) and the emergency organization (who is responding to the emergency). These emergency services must be everywhere for the traveling users. These three players are bound together by the geographic information at each step for the user and the emergency organization. Similarly, for the mobile operators the geographic information helps in deciding for the location of the transmitting towers, their coverage area and the cell sizes. Binding all the players with geographic information can be visualized as shown in Figure 2.1.

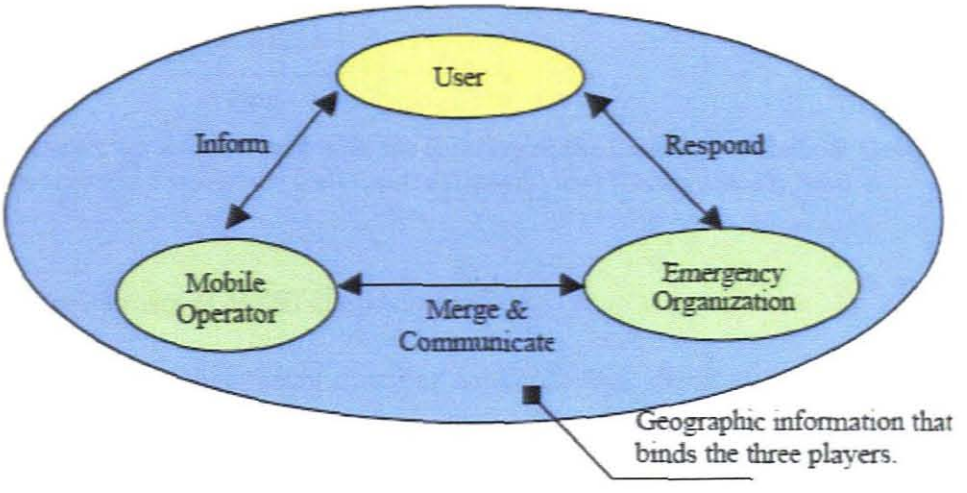


Figure 2. 1 Three Players that are bound together with the Geographic Information

[Gaurav Singh, 2004]

Information and geospatial data required by emergency services for effective and speedy response is obtained from the geospatial database. This is shown in the set up scenario of client server architecture in Figure 2.2. Considering the actual scenario where the user at the time of emergency calls the emergency number (emergency number depends up on country to country), his call is then routed to the nearest emergency control centre (ECC) by the mobile operator. Once the call is routed to the nearest ECC the user might want to have access to the maps with the position of the user or the incident on to it.

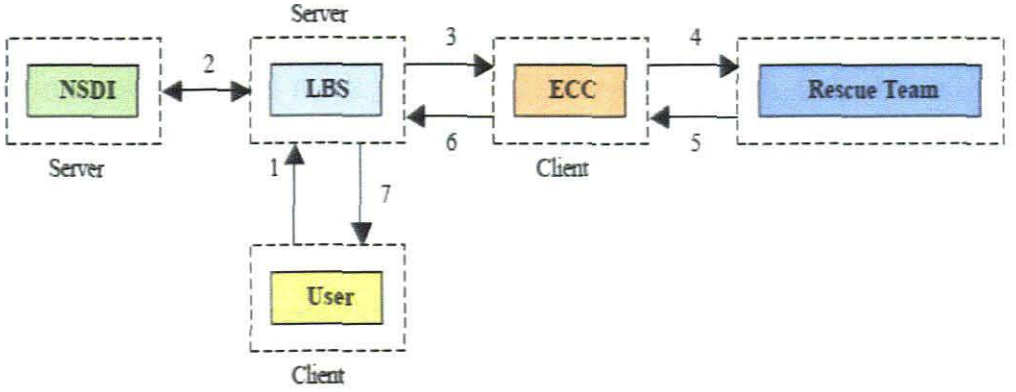


Figure 2. 2 Client server architecture with the Routing of the Emergency Calls & Geospatial Data to the Emergency Call Centre (Broad view) [Gaurav Singh, 2004]

Working of the system (LBES) (Broad view)

1. The user dials the emergency number and informs about the incident and location.

2. Based on the user location (latitude and longitude) the relevant geospatial data is fetched from the NSDI (National Spatial Data Infrastructure).

3. This data is routed to the nearest emergency centre in the same way the call is routed.

4. The emergency control centre sends its rescue team to the site of the incident

5. The rescue vehicle is tracked by the ECC.

6. The route taken by the vehicle is sent to the LBS and

7. Then the information on the incident in a particular area is sent to the users in that area just for the event notification.

Geospatial database for LBES, contains geographic data at the required scale for the large scale emergency such as forest fire, buildings, roads, etc. Some of the main data kept and dealt with in LBES include data on police, fire and rescue, public health department for medical services, urban planning organization etc. Since these organizations are identified as stakeholders in the emergency services, the LBES model shows their parameters and their relations in the database model.

The LBES model features the use of service agents for communicating the participating components of the system. According to Singh (1998), Zipf and Aras (2002), one of the advantages of using agents is that they communicate on what, not how. They autonomously decide what operations to perform and what data to access for others. Also, to perform appropriate and efficient

queries agents can exploit their knowledge of local information. Mobile agents can move between nodes in a network. As mentioned by Calhaus *et al,* (2004), agents help in doing most of the computation at the server side leaving the client side for faster access. Agent technology provides powerful flexibility in such a way that new functionality can be implemented very easily just by adding few more agents. As shown in Figure 2.3, DIA (data interface agent) on receiving the request from the WDA (wireless domain agent) transfer the request to DDM (data domain manager). It also simultaneously triggers actions by the metadata agent which checks the requested metadata of the data domain and the query agent which checks the type of query (static or dynamic) and the type of ontology required to answer the query. If the required ontology is available the query agent further moves to resource agent, but if the required ontology is not available the query agent moves to the ontology builder that builds the required ontology and updates the ontology manager. The resource agent fetches the required data and sends it to the map agent that further helps in the proper visualization of the fetched data and sends it to the router agent of the wireless domain.

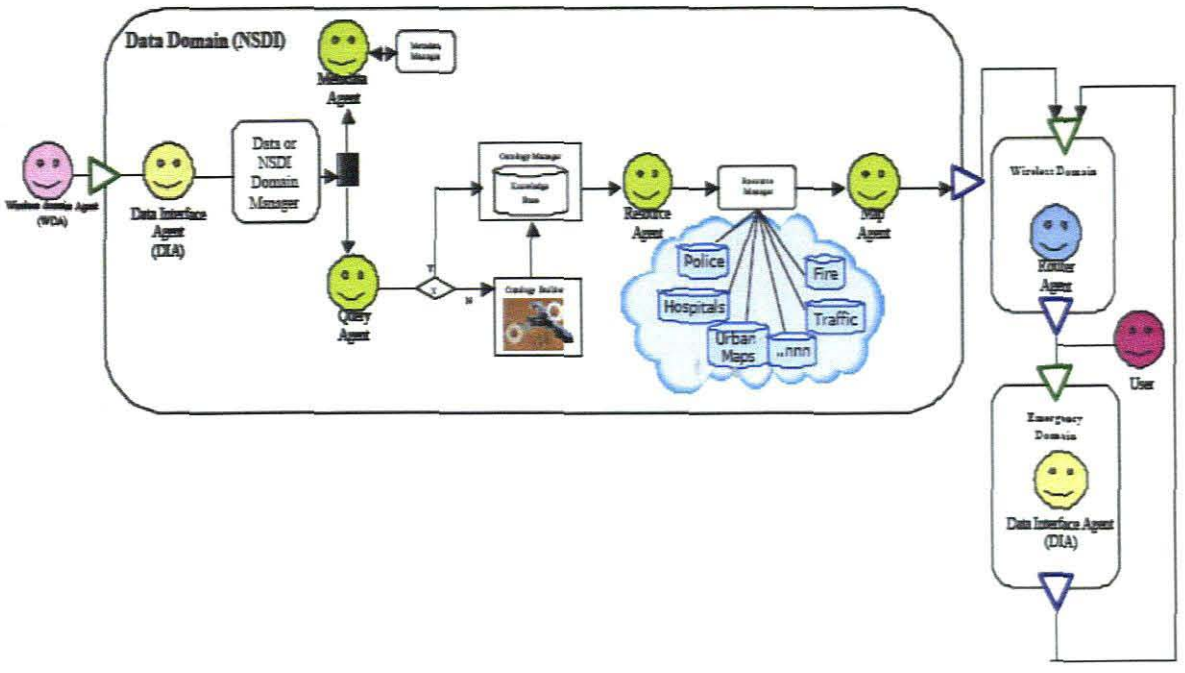


Figure 2. 3 Agents for Database services [Gaurav Singh, 2004]

2.2.2 LBS Publishing and Discovery Platform

Some of the characteristics we based our literature review on are database management, the design of information repository as the storage media in mobile computing and the automatic location of mobile device.

In LBS Publish and Discovery mechanism (You-Heng and Samsung, 2005), Spatial Database Management System (SDMS) is discussed. SDMS stores indexed spatial properties of LBS applications. It is a data type structure for location information, such as Position and Point of Interest (POI) that are modelled as Abstract Data Types (ADT) and encoded in XML. From this model, LBS publishing and discovery platform is regarded as a middleware

which acts between service requestors and service providers. The architecture shown in figure 2.4 consists of three software components, namely: (1) Service Description Repository (SDR), (2) Service Spatial Properties Repository (SSPR), and (3) Service Publishing and Discovery Engine (SPDE).

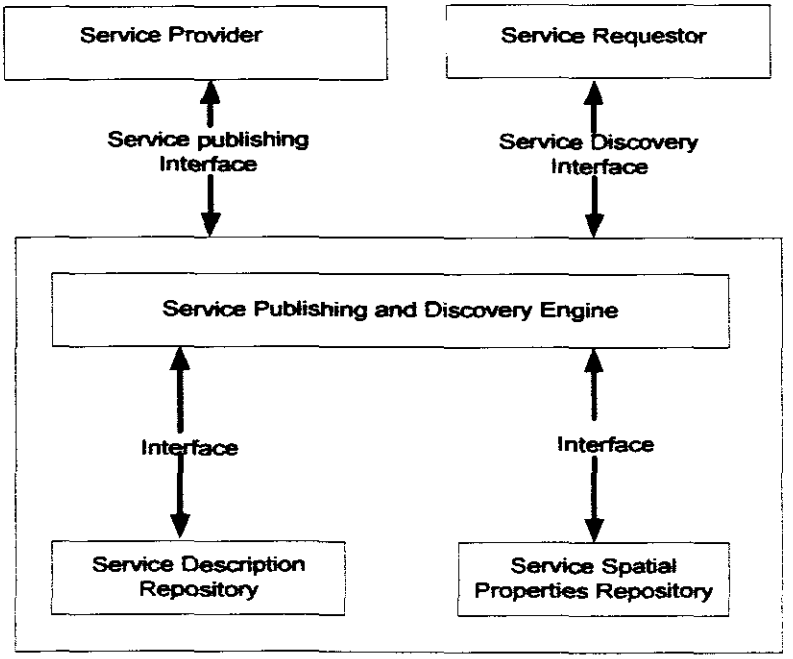


Figure 2. 4 Architecture of publishing and discovery platform for LBS applications
[You-Heng Hu and Samsung Lim, 2005]

Service Description Repository (SDR) – Textual description of LBS application is stored in SDR. In their proposed platform, SDR is a software component that works as a registry of information about businesses and their services. The Universal Description Discovery Integration (UDDI) is used as a standard tool to construct the SDR.

In UDDI, the information about businesses and their services are divided into three categories, namely: (1) White Pages provide contact information and identifiers for a business, such as name, address, phone, fax and email, (2) Yellow Pages provide classification information about the types of the business and the services it offers based on predefined taxonomies. Service Spatial Property Repository (SSPR) is used as data repository for spatial properties of LBS applications.

SSPR provides an extension to SDR to store and index spatial properties of LBS applications. Such properties may include location and coverage range. For the system's seamless interoperability in implementation Heng Hu and Samsung use formal and standardized methods for describing and accessing data as it is a necessary tool. In SPR, OpenLS Abstract Data Types (ADT) and OGC Simple Features Specification for SQL (Open Geospatial Consortium, 2004), are used for this purpose. The spatial properties of LBS applications in the proposed system are modelled by using OpenLS ADT and encoded as XML documents. OpenLS ADT is a fundamental of OpenGIS Location Services (OpenLS) specification, well-known data types and structures for location information, such as Position, Point of Interest (POI) (You-Heng Hu and Samsung Lim, 2005), Address and Route are defined. Position ADT and AreaOfInterestType ADT (Ratsimor et al, 2003) are used to describe the position and coverage range of LBS applications respectively.

The Service Publishing and Discovery Engine (SPDE) module acts as a mediator to dispatch publishing and discovery requests to SDR and SSPR then collect results from them. It is the backbone of the publishing and discovery platform. SPDE provides a UDDI compatible API that extends the UDDI specification to support LBS applications. Extensions in SPDE include three types of information: business information, service information, and information about spatial properties of LBS applications. The core software components of SPDE are the publishing processor (PP) and the discovery agent (DA).

The LBS Publish and Discovery platform is also applicable in wireless network environment. This involves the issue of mobile device location as it is one of the mentioned characteristics. The technology used in this platform for automatic detection or location of a mobile device combines the following communication technologies: General Packet Radio Service (GPRS), Code Division Multiple Access (CDMA) and Wireless Fidelity (WiFi).

2.2.3 Agents2Go

This is the LBS application model that incoperates almost all the mentioned characteristics in section 2.2, that is, Push and pull service technology, Automatic location of mobile device, Mobility/ handling of network disconnections and Database management.

Communication and transaction in Agents2go model are through messages that are exchanged between a mobile device and the Agents2Go Server. These messages are encapsulated in a generic message format. On startup, a mobile device requests an "initial query form" from the Agents2Go Server which is a pull service technology. Through this form, the user can specify a desired request. Upon submission of the request through a send button, the input of the form is converted into a "form data message" and sent to the Agents2Go Server. The "form data message" includes only the values of active components in the form like check boxes, fields, lists, etc. Once a mobile device sends the "form data message" it waits for a response from the Agents2Go Server. The format of a response is a form that a mobile device displays to the user. This form contains a "home" button that causes a mobile device to generate the "initial query form" message to allow the user to enter a new query.

Agents2Go system uses Cellular Digital Packet Data (CDPD) to locate the user in a CDPD (Muthukrishnan and Rittwik, 2001) environment using cell tower network technology. CDPD provides an infrastructure that allows the transmission of data over idle capacity of already existing cellular voice networks. Cellular networks consist of cell towers. Each cell tower has a unique id and defines a geographical boundary called cell, the area that is serviced by that tower. Agents2Go system employs these tower ids to identify a user location. The CDPD module employs a library to obtain periodic status

reports. These reports contain information like cell tower signal strength which is used to minimize packet loss and the tower id which is used for inferring coarse information about current location, etc.

All the messages that are exchanged between the Agents2Go Server and a mobile device are sent using the Centaurus Communication Protocol (CComm) (Kagal, et al 2001). CComm is a layered protocol that provides communication between mobile clients and a server. It handles transmission of data messages and control messages between a server and a client. This layer handles multiple clients, is insensitive to disconnections and is easily portable. It handles all connection/disconnection issues, user identification and authentication.

In the Agents2Go System there is a broker that is associated with a specific Agents2Go Information Repository. An Agents2Go Information Repository is a set of databases that contain information about participating restaurants in a given coverage region. Restaurant information like names, addresses, cuisines etc. of all participating restaurants in that coverage region is distributed among these databases. This information can be classified as static, since it rarely changes. The broker is also responsible for updating frequently changing restaurant information like waiting times and promotions. This kind of information can be classified as dynamic. This dynamic information is maintained within the broker itself. This separation

of dynamic and static information reduces the number of messages that is exchanged between the Agents2Go System components.

2.2.4 Konark

Mobile computing, which is a computing involving handheld devices with wireless connectivity, is totally different from the traditional infrastructure based on desktop wired PCs (Buszko et al, 2001). In Konark model (Sumi Helal et al, 2003), model these differences are well thought-out in designing a service discovery protocol for ad-hoc networks. While discussing the service discovery mechanism it is worth mentioning the significance of the type and range of services targeted in the LBS system. Konark model addresses two of the four issues mentioned above, namely, *Database management* and *Network disconnectivity*.

Communication in Konark is through the network of the participating devices and not the network coverage area provided by network operators. Konark supports two types of networks – partially ad-hoc networks, and completely ad-hoc networks. Partially ad-hoc networks are those where some devices are static and others mobile. An example could be an ad-hoc network in an airport where a printer and a fax-machine are "fixed" while a traveler's laptop is mobile. Another example could be a shopping mall where a shop's server is static while a visitor's PDA is mobile. Completely ad-hoc networks are those where each participating device is mobile and the network has been

formed temporarily. For example, a network formed when a group of people comes together at a conference.

An important design factor in a service discovery protocol is the storage of services and their metadata, that is, information about available services. Therefore there is a need for repository for all available services in the network. Any participating device offering a service has to register its service with this repository and all devices seeking any service have to query it for available services. Konark uses a completely distributed peer-to-peer approach to solve this problem. It specifies that each device will have a local repository that will maintain the local services being offered by that device. Traditionally, service discovery protocols have focused on services provided by devices, such as printers, fax-machines, cameras, audio-systems, etc. With the popularity of handheld devices, and the introduction of mobile commerce, we envision an entirely new set of possible services. These services are becoming location based dependent, and device independent and could vary from services such as games or music, information-related services such as maps or weather, or location based services like find the nearest hotel.

Another important issue in service discovery protocols is how to make service information available to other devices in the network. Konark supports both advertisement and discovery. The service providers can push their services into the network. Advertising depends on a lot of factors such as geographical

or temporal information, that is, location and time based advertising. With the increase in the number and variety of services, locating a particular service becomes increasingly difficult. Depending upon the network size and other factors, the information obtained about available services - either via actively discovering them or by passively caching the advertisements - can be in large quantities. Also, the kind of m-commerce oriented services being targeted requires a high level of user interaction. This makes it necessary to have a registry that stores service information in an extremely manageable and user friendly fashion. Another advantage of this feature would be the support for possible human interaction during advertising and discovery. To handle the above requirement, Konark presents a service registry based on a tree-structure. A basic tree skeleton is used with service classification levels that are generic at the top and become more specific as you move down the tree levels. The tree structure described above is not only used for maintaining services and information about available services, but also for advertising and discovery.

2.2.5 Salutation

In Salutation we have looked at the style used in keeping the participating components glued together to ensure the smooth flow of the information. The Salutation architecture in Figure 2.5 from Diplomarbeit (2002), provides a standard method for application, services and devices called Networked Entities – to advertise their capabilities or request the desired ones. It is

claimed to be a processor, operating system and independent communication protocol. Its key component is the Salutation Manager (SLM). Each Networked Entity has a SLM and uses a remote SLM by means of the Remote Procedure Call (RPC) protocol. The SLM provides services and clients with a transport independent interface (SLM-API). A Networked Entity acts as service, client or both. The different SLMs communicate among themselves using the Salutation Manager Protocol, which uses the Remote Procedure Call. The SLM presents also a transport-independent interface (SLM-TI) to transport dependent entities, called Transport Managers (TM). Each Transport Manager supports one kind of transport, making thus the SLM independent of which one is used. The Salutation Manager and the Transport Manager(s) together perform the service of a service broker.

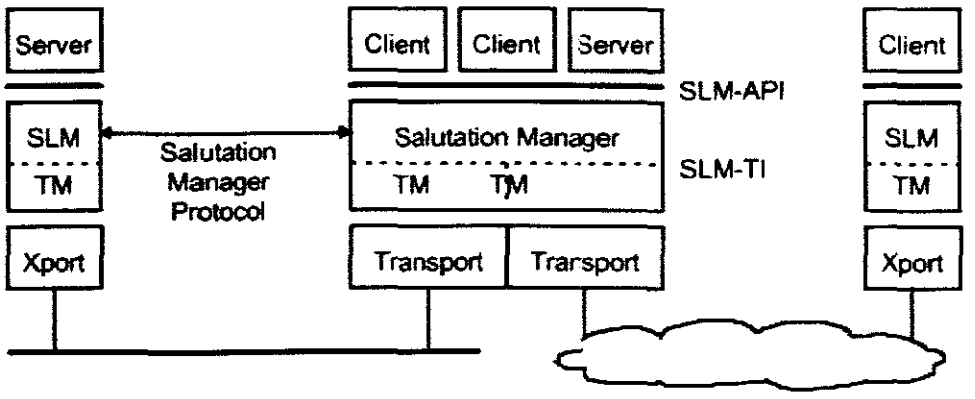


Figure 2. 5 Salutation Architecture [Diplomarbeit, 2002]

The main tasks carried out by a service broker are the following:

Service Registry: A SLM holds a registry of services. At least, it must hold the information about the service connected to it (locally or remotely), but it can also store information about other services on the network. An SLM can even maintain a central service directory for all Salutation equipment in a network.

Service Discovery: The SLM can discover other SLM and the services registered there. This discovery is based on service type comparing, but the SLM can also request services matching specific characteristics.

Availability Check: although there is no lease in Salutation, the SLMs can be requested to periodically check the availability of the registered services.

Service Session Management: In order to allow the use of a discovered service by a client, the SLM creates a virtual data pipe between client and service. This is called a Service Session. The Salutation architecture is rather mature. Several commercial implementations of the protocol exist. It can be used together with Bluetooth's service discovery protocol (SDP).

2.3 THE PROPOSED SOLUTION APPROACH

In this dissertation a Location Based Application Server (LBAS) is proposed as a model for the location based mobile commerce applications. The LBAS system has several distinguishing features that provide advantages defined below, over the existing location based service search systems.

First, it is a platform that allows service suppliers to define and register their services. Also the LBAS system allows service providers to actively participate in the system maintenance through dynamic information updates. It adopts the push service technology for service providers when they advertise their services in respective locations. This feature helps the system to target all mobile users and entails them to subscribe to the system if they are interested in such services. Secondly, it lets service consumers, especially those on the move, to automatically discover available services in their vicinities using Mobile Network Coverage Area (MNCA). LBAS identifies a user's location through network operators and uses this information to refine its searches to provide the most relevant information to the user. This improves the quality of information or services presented to the user and ensures that the service providers are able to send their latest promotions/updates to their users. Thirdly, LBAS database has two component databases, one for mobile service consumers users and the other for all service providers with their location based services. This help in reducing communication overheads in accessing these databases.

2.4 COMPARISON OF LBAS WITH EXISTING SCHEMES.

The model proposed in this work is a contribution to existing research results such as those listed in Table 2.1. LBAS has been developed to prove that location based mobile application services can be advertised, by service suppliers whose services and information are kept in the database, in a given overage area and accessed by every mobile user entering that particular area.

Table 2. 1 Comparing LBAS with some of existing schemes

Model Name	Mobile Device Detection/Location Style	Components communication Style	Network Disconnection Handling
LBS Publish and Discovery Platform	Published services are discovered by specifying the user's current location and other keyword-based criteria using the internet.	General Packet Radio Service (GPRS) and Wireless Fidelity (WiFi)	No clear mechanism defined for handling disconnections.
Agent2Go	Mobile user registers first then will log in for location purposes. Push and pull technology is also applied	Messages through cellular digital packet data (Mobile phone network)	Uses partitioning mechanisms of overlapping regions into disjoint sets: *shared* and *native* type. A *Shared* set contains places that are located in the areas of cell overlap. A *Native* set contains the remaining places that are in the Broker's *coverage region* but not in the cell overlap.
Salutation	Every mobile devices registered for the system is assigned to agent	Employs local area network for communication of components	Disconnection is impossible with salutation because of local area network usage
LBES	Uses device tracking mechanism to monitor the movement and direction of all subscribers of the system. Push and pull service technology is also applied	Offers a simple but powerful agent description facility that allows agents to find each other. Mobile devices are represented by agents and they are the ones that do all necessary tasks on behalf of participating devices.	No clear mechanism defined for handling disconnections.
LBAS: Our Proposed model	Uses automatic mobile device detection mechanism of network coverage area (NCA) of mobile device offered by network operators	Components communicate through short messages through network service providers.	Uses partitioning mechanisms of overlapping regions

CHAPTER THREE

MODEL DEVELOPMENT

3.1 INTRODUCTION

Location based mobile commerce aims to provide targeted information to mobile users in response to their request based on their current location in the exchange of monetary transaction. With the explosion of mobile computing, more and more people use a variety of mobile devices in their daily lives. Recently, electronic commerce has been recognized as a growing sector in the market. The combination of the above concepts has resulted in the emergence of Mobile Electronic Commerce (M- eCommerce). One of the most critical requirements for M-Commerce is the ability to discover services in a given context. An important component of a user's context is his/her current location. For example, a user on arriving at a location that he/she has never visited before should be able to find a local nearest service to his vicinity. But there is still well known limitations with current mobile devices, including limited power supply, smaller screen, limited computing power, limited bandwidth and storage space. These limitations require the development of systems that provide mobile users with high quality, precise and context relevant information to be accommodated within these limitations. It is important that these systems be highly scalable since the

demand for service searches will increase in the future. A search of location based services utilizes a user's current geographical location to refine its search and provide access to locally available services. That is why one of the challenges of location based searches is determining the user's current location. These happen while mobile users are often uncertain, or even completely unaware, of any transactions and exchange of information with relevancy to their current geographical location. Now, for the provision of location based services, an automated detection of the user's current location becomes very helpful and the key problem facing LBS.

Location based services systems are naturally described and implemented as distributed systems. This improves their fault tolerance and scalability. For instance, service information can be grouped by location and managed by a server responsible for the specified geographical region. In such a decentralized scheme user requests are processed at the local server and do not burden the rest of the system. This makes the system more efficient, responsive and scalable.

3.2 FEATURES OF LOCATION BASED APPLICATION SERVER (LBAS)

3.2.1 Overview

LBS need to use real-time positioning methods and accuracy to discover the location of a device. Locations in general can be expressed in spatial conditions 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. 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 (Rao and Minakakis, 2003). The overall goal of the system is to deliver a nearest service to a user (e.g., Find Nearest restaurant, hotel, etc.) based on the user's current location. The system is expected to advertise available services and allow service consumers (mobile users) to automatically discover those services through their mobile handset devices. Location based services are advertised according to zones in which the service supplier wishes them to be available, like towns.

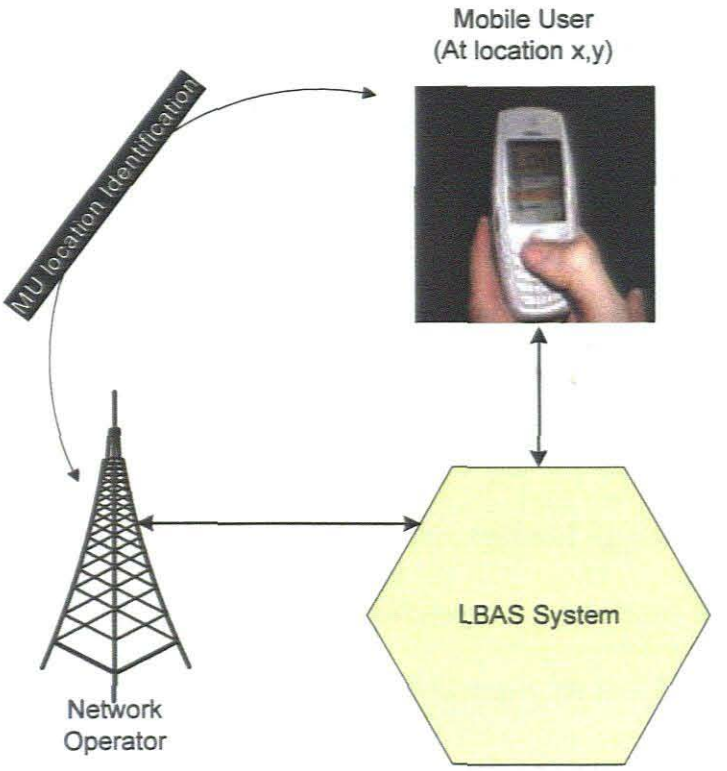


Figure 3. 1 LBAS Context Model

The system in its entirety will incorporate the following services and system components (see the Figure 3.1):

Network Operator: It utilizes Mobile Network Coverage Area (MNCA) to obtain the user location.

LBAS System: Location Based Application Server provides customized information to users according to their location.

Mobile User: Users can either be registered or unregistered in order to get the full use of the LBAS systems.

Registration can either be done on the Internet during site visitation or during service discovery through mobile device. After the registration is complete the users will be provided with customized information from the LBAS system. Applications can be used on any of the several types of positioning methods, such as:

Mobile phone network: This method uses cell ID 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 cell may be anywhere from 2 to 20 kilometers in diameter. Other techniques used along with cell ID can achieve accuracy within 150 meters (Barnes, 2003).

Satellites: The Global Positioning System (GPS), uses a constellation of satellites orbiting the earth. GPS determines the device's position by calculating differences in time taken for signals from different satellites 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, 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 (Hjelm, 2002).

Short-range positioning: This method is applicable in relatively small areas, such as buildings and a local area network can provide locations along with other network systems for services. For example, appropriately equipped devices can use Bluetooth short-range positioning (Jagoe, 2003). The LBAS system has several distinguishing features that provide advantages over the existing location dependent service search systems. First, it is a platform that allows service suppliers to define and register their services. Also the LBAS system allows service providers to actively participate in the system through dynamic information updates. Secondly, it gives service consumers who are mobile users to automatically discover available services in their vicinities.

3.3 LOCATION BASED APPLICATION SERVER DESIGN CHALLENGES

There are different key areas for consideration and challenges that need to be addressed for location based services for increasing their usability and value to users. These are some challenges that will have a crucial effect on this project development:

3.3.1 New network technologies

The expanding development of new network technologies including 2.5 G and 3G technologies will increase the use of location based services. The need for being always connected, the speed of data transfer, and the charging per

volume user-value, will enable the LBS to benefit from these technologies. The ability to advertise or push data to users based on their location and preferences, in a seamless and inexpensive manner, is likely to help location based services to grow.

3.3.2 Standardization

Many efforts are being put in standardizing location based services, both on the network and on the application side. One of such efforts is the 3G series technologies, defining mainly the addition of location based services capabilities to future releases of 3G networks, another is the flexibility in location based services, formed by vendors and interested parties to develop and promote common and ubiquitous solutions for location based services which are network and location determination technology independent. The outcome of these efforts has a huge effect on the success of location based services and affects the technology industry. Operators of these efforts make the required investment to upgrade existing LBS, as well as on the actual availability, usability, and cost of services.

3.3.3 Availability of attractive services

As long as there are attractive services, location based services will not fail but offer easy to use services. Some of these future services are likely to benefit from higher accuracy of location determination technologies. The ability to offer such services requires tight cooperation between mobile

operators, application developers and equipment vendors and mobile users. This requires the understanding of subscriber's preferences and usage habits as well as technology expertise. Although this facilitates the development of LBS, the key is still in attracting the subscribers.

3.3.4 User acceptance

A key question remains whether subscribers will be willing to pay additional fees to use these services. User acceptance surveys provide different answers, most of them debatable. General usage figures based on past experience with other services show that the answer lies in the usability and value services they bring to users. This adds a further dimension to the attractive services mentioned before – services should be tailored and offered to specific user segments, maximizing their value from such services. Operators are in a key position to define and package such services, and tailor them to the needs of their different subscriber segments.

3.4 DESIGN PRINCIPLES

Development of location based services in mobile commerce applications involves mainly three groups of participants, namely, end-user, client (customer) and developer. Therefore design principles in this project have been developed to meet or achieve the goals intended by these participants. Let's define each participant according to his/her involvement and activities in Location Based Application Server system (LBAS).

End-user - A person who uses a LBAS system, as opposed to those who develop or support it. The end-user may or may not know anything about LBAS, how it works, or what to do if something goes wrong.

Client - A person using the location based services offered by LBAS. A client is the buyer of LBAS. For client design goals are: low cost, flexibility and increased productivity

Developer - A person who designs and writes software part of an LBAS. These are generally designers and programmers in the commercial software field. These are design goals for any system developer: minimum number of errors, modifiability, reusability and well defined user-interface.

These are the design goals for end-users: user-friendliness, ease of use and learning, fault tolerance and robustness.

a) User-friendliness

The system should be designed in a fashion that one (user) should not have to recompile the code in order to run a simulation with different parameters or different reactions. The system should be easy to use. Users should be able to search for desired location based services and they should easily understand and interpret the outputs of the system. The system should be effective and efficient for both novice and experienced users.

b) Ease of use and learning

End-user should not need to be either expert in order to use and learn LBAS.

c) Fault tolerance

To achieve the goal the LBAS should be designed such that it can tolerate faults input by the user. Whenever one of the participating components fails the other one must take over, and continue with the transaction.

d) Robustness

Users of LBAS should be able to take corrective actions once an error has been identified. The communication rate between the LBAS and the user should be comprehensive. The system should support all the tasks the user wishes to perform.

e) Low cost

To achieve the goal, the system should be cost effective for clients, for example, using the sms format for communication with its client. Sms's are charged at a standard rate which everyone using mobile device is able to pay.

f) Flexibility

The system should have multiple ways that the client can use to exchange information in interacting with it. This includes the ability of the system to initialize the conversation in which case the user simply responds to requests for information.

g) Increased productivity

Seamless exchange of information for LBAS should be worthy enough in location based services to increase productivity, profitability without compromising software quality

h) Minimum number of errors

The reason for this design goal is to design better interface to minimize possible errors and to minimize the possibility of errors in the source code and to enforce the development standards for all source code being coded.

i) Modifiability

The system should be able to handle the following:

- Readability - The source code of every part of the program should be easy to read to enable later modifications and extensions to the project by others. The program should be well-commented and should satisfy all the aspects of software engineering.

- Writability - The code for the program should be easy to modify. New features should be easily added to the proposed system and the program should be adaptable for different domains.

j) Reusability

Reusability is an important development goal for many software projects that use object-oriented techniques. It is useful for testing. It is an important aspect of application development that enforces a consistent user interface and facilitates the creation of reusable code. Code reuse decreases the cost for new development and provides incremental quality improvements.

k) Well defined-interface

LBAS should have a well-defined interface for location based services to be accommodated by mobile device. This is because the state of mobile user interface (UI) design is pretty bad. These are the challenges to be addressed by this design goal in mobile UI design: the physical input limitations paired with a small screen, result in information display to be very difficult for the mobile designer.

3.5 DESCRIPTION OF THE LBAS SYSTEM MODEL

The system model which comprises of user interfaces (WEB or WAP), LBAS, database server and context and profile manager (CPM), is depicted in Figures 3.2.

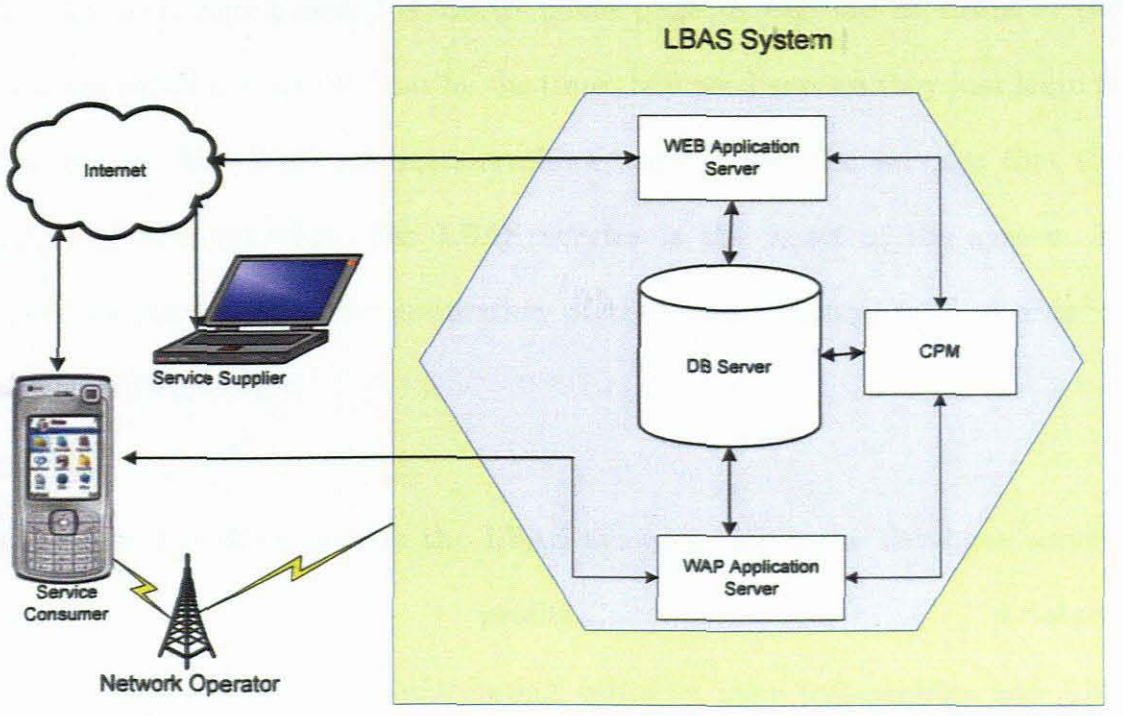


Figure 3. 2 LBAS System Model

The clients communicate with a common server (LBAS) through network operators. This is obvious since a mobile device has to register to any nearest network provider to be available for communication. Service suppliers register their services through internet by visiting the home page and fill in the registration form that requires service information and the location in

which it should be available. These service suppliers can also edit both their information and service information any time they log in to the system. The web application allows users to maintain their personal info and preferences. Web application caters for both service suppliers and mobile users.

Mobile users can access the LBAS home page to register as users of the location based services so that by the time they need service they just login to the system. The WAP application allows users to use the services that the LBAS system provides. The LBAS service is the heart of the system. It provides functionality for estimation of the user's location and storing of service information.

There are two databases in the LBAS system, within the database server, User profile database and LBS database. User information contains user information and LBS database contains service information. As shown in figure 3.2, the user can connect to the LBAS system via a Web browser and WAP device. The user can register via the web application. Here he can update his preferred services. When a user has registered he can utilize the LBAS service via his WAP device. If the user is unregistered he/she can automatically discover available services by means of his/her mobile device but cannot access it up until he/she subscribes to the system or service. Network operators play an

important role in the LBAS model. In the real world before there is any transaction and initiations in mobile environment, which can be calls or massages, mobile device should be registered to any network operator for the identification of its current position. Therefore in our model network operators are responsible for alerting our system about the current location of mobile users which is addressed by the context and profile manager that dwells in location base application server of the system.

To register with the system, a user logs on to the application server and enters his or her preferences by selecting appropriate fields in profile forms. The location based application server (LBAS) forwards the preferences to the CPM, which aggregates them in user profiles. If the client terminal has a GPS receiver, it sends regular location updates to the CPM. Although we focus here on GPS-based methods in which user location is available at the mobile terminal, we could also use other location tracking methods, such as network-based triangulation, which relies on multiple cell sites for determining a terminal's position. The CPM computes user direction and speed from the user's tracked location history and stores the latest dynamic context values in the user's profile. Mobile users seeking local information about nearby service, for example nearest parks or hotels, connect to the application server to request the information. The application server forwards the query to the CPM, which queries the server for the addresses of

available restaurant Web services. The CPM then sends a request to the identified Web services specifying a service area for the user. The Web services search their databases for the appropriate resources, filtering out those that fall outside the specified area, and return an XML list to the CPM.

3.5.1 Considered issues affecting LBAS

Some issues including mobility management, lightweight client, subject oriented service binding, service handoff, data consistency, and dynamic binding, are taken into consideration in our design of the proposed service discovery protocol.

Mobility Management: In our design, the first Access Point (Coverage Network) of network operators receiving specific service request from a mobile phone becomes the location identification of the user and acts on behalf of this mobile phone during the whole service session. Besides, the location agent tracks the current location of the user. When the user moves across the boundary and enters a new location, it registers itself to the local access point that will report current location of the user to the system. Each time the mobile phone changes its location, the mobile terminal registers itself with the network operator that immediately alerts current the system about the mobile phone's new location.

Subject Oriented Service Binding: To request the service, the user does not need to learn detailed information about this service. All the client needs to do is to request the service by specifying the location of the service and not

any specific information about the system. The system and the service directory will select a proper service in the vicinity of the user. Subject oriented service binding can be transparently achieved.

Data Consistency: A set of data structures is maintained for transparent service binding between the mobile user and the service supplier. Data consistency between these geographically distributed data structures is essential to correctly achieve service discovery. A service supplier sends updates to the service directory, then, the service directory forwards these messages to the related location agent for whoever inquired that service. Each time the user logs in, the information about the local environment and service will be provided according to the location and the time that the user uses the mobile phone.

The LBAS system will first advertise to the user a list of available services based on current conditions and the user's preference recorded in the database if he/she is a registered user, otherwise the system will prompt the user to register with the system before accessing such services. The user then can either choose a service from the list or make new selections from a menu to be shown on the mobile phone screen. The system will make an optimum choice to meet the user's goal by selecting the most appropriate service from the database. Meanwhile the system will also update the database system to reflect the user's new preferences based on his/her current choices. This can also be used to update the criteria in the database system to be used as plans

for building future visitation. In this way, the system is able to remember the user's preferences and provide a more recent candidate list next time the user logs in. Thus the system is designed with a dynamic learning ability in the run-time environment by manipulating the database system. The basic idea in LBAS is the introduction of a mechanism by which the system returns the most nearest service preferred by user.

3.5.2 Proposed Service Discovery Protocol

The proposed service discovery protocol is as follows:

Step 1: Service supplier needs to register itself with the service directory in the system and publish his/her services so that services that he or she offers becomes available to the rest of the defined locations during registration.

Step 2: A mobile user discovers available services, if not registered he/she must register and then sends service request to the system. It is worth mentioning that the concept of location-oriented service binding is considered. That is, in the proposed protocol, clients request a service by specifying the name of the preferred location without detailed knowledge about services that are offered in that location.

Step 3: After receiving service request from the mobile user, the system forwards this request to the database server manager to checkout the available service in the mobile user's current vicinity. Meanwhile, there is a location agent that acts as the broker of the system during the whole session.

Step 4: The database server manager checks if the requested service is available by looking for the advertised services after receiving the service discovery request from the inquiring location agent. A service directory records related information of a mobile user's location and then, sends back the result of the requested service to the location agent.

Step 5: After obtaining the result, the location agent finds out the current location of the requesting mobile user and delivers the result.

3.6 DYNAMIC CONFIGURATION PROVISIONING WITH LBAS

In LBAS, through network operators, the user's current location is automatically identified and services are pushed in an advertising manner to the coverage area whereby every mobile device entering that area will receive information about available services in such vicinity. In this section we give a description in terms of a scenario which indicates how dynamic configuration of services can be achieved with LBAS. We use a scenario of the nearest place to eat (Restaurant) whose prototype implementation details are given in chapter four. In this scenario, we consider an example of a visitor Jack finding a suitable restaurant in a city being visited for the first time.

In the scenario, Jack wants to find "near-by" restaurants of his choice or those along a certain route. Using a mobile terminal, he may enquire about the closest fast food restaurant, say a McDonald's. As a response, a

McDonald's outlet, together with its address/location may be displayed on the user's terminal.

In addition, a map showing its location (or driving instructions) can also be displayed. Jack's current location and preferences, therefore, help to narrow down the search for relevant restaurants instead of searching the whole database of restaurants available in the system. This ensures that at the end the system returns only a list of restaurants serving the type of food preferred by Jack, which are also in the vicinity of his current location. Now, next time Jack travels to another city and plans to visit a restaurant there, the system will be able to adjust itself according to Jack's new location. By doing so the system features the concept of context awareness.

3.7 DESIGN AND PROTOTYPE DEVELOPMENT

The most important objective of implementing the prototype is to prove the LBAS concept. With this, we can show how location based services can be discovered in mobile environment services in a distributed application using the LBAS model by a mobile device, how to advertise available services to a mobile device in a given vicinity, and demonstrate the capability of LBAS to calculate the nearest services and present the information about those services to the mobile user. There is data exchanged among the components where the service supplier advertises his/her services in a given coverage area registered in a service directory. Once the mobile user (tourist) registers

with mobile network operators he/she receives a message displaying nearest available services around his/her vicinity. This information is managed and supplied by the Location Based Application Server (LBAS). The user can either accept system's advertisement or quit the conversation. The system then checks registered and advertised services in the service directory to find the requested nearest restaurant - within the current location of the user. Finally, the system returns information such as the name of the nearest restaurant that meets the search criteria, its physical address as well as the telephone number. Figure 3.3 is a UML activity diagram showing actions taken during a session of service discovery and service advertisement.

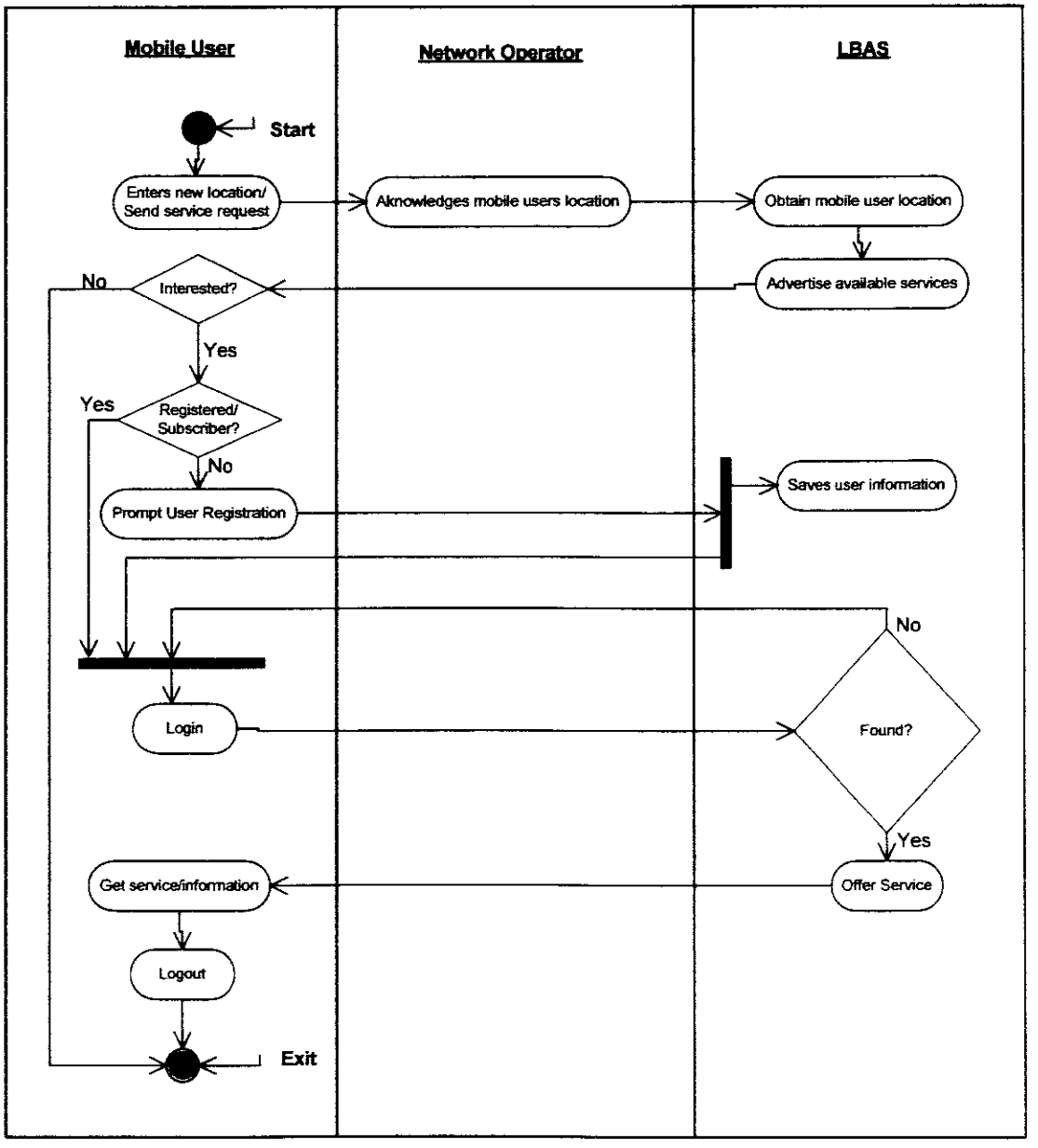


Figure 3. 3 Activity diagram for the Nearest Service Finder System (NSF)

3.8 FUNCTIONALITY DESCRIPTION OF NSF

The tasks that the LBAS service will accommodate are illustrated with the Use Case model of Nearest Service Finder System (NSF) (see Figure 3.4).

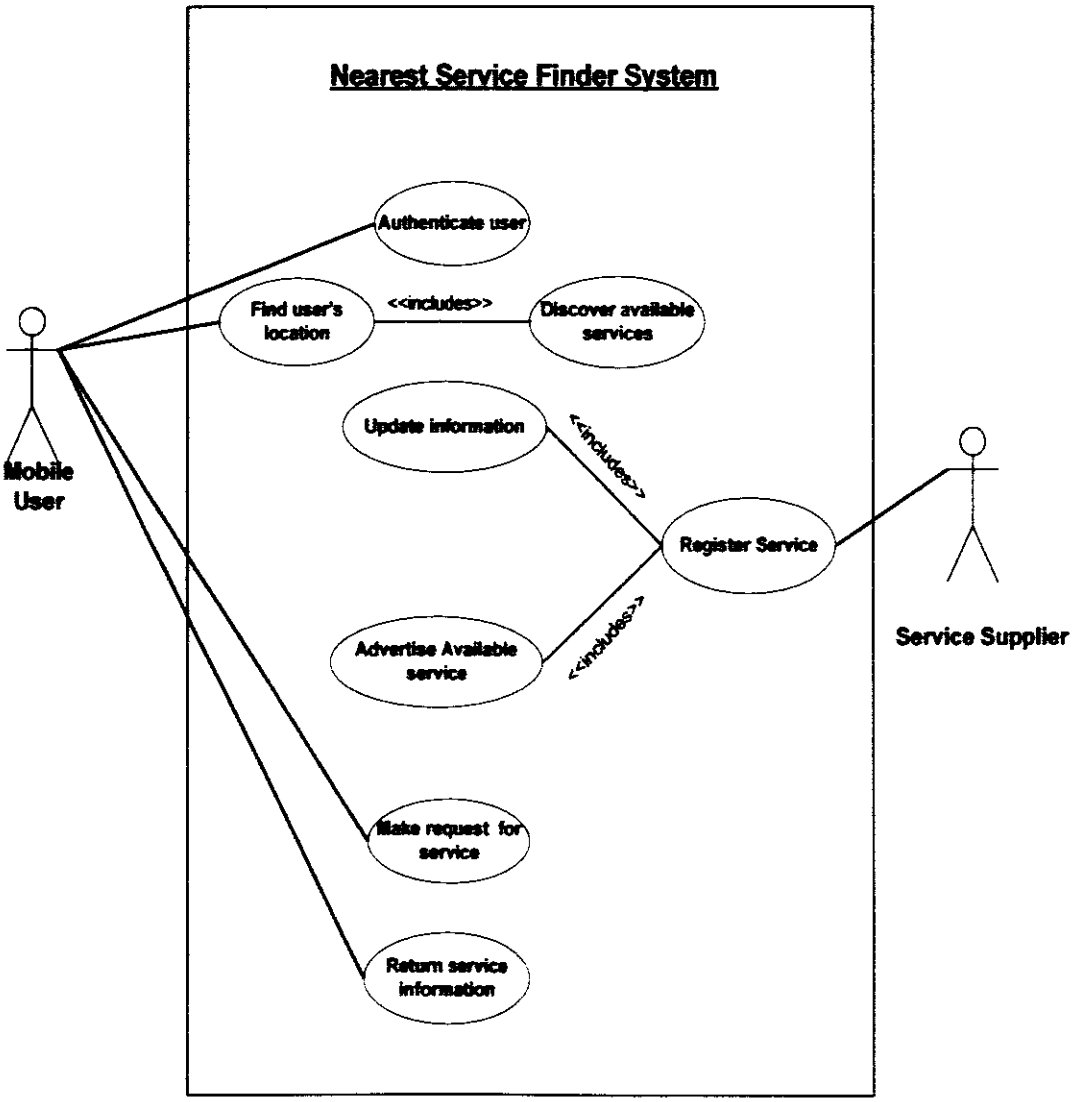


Figure 3. 4 Use case diagram of NSF

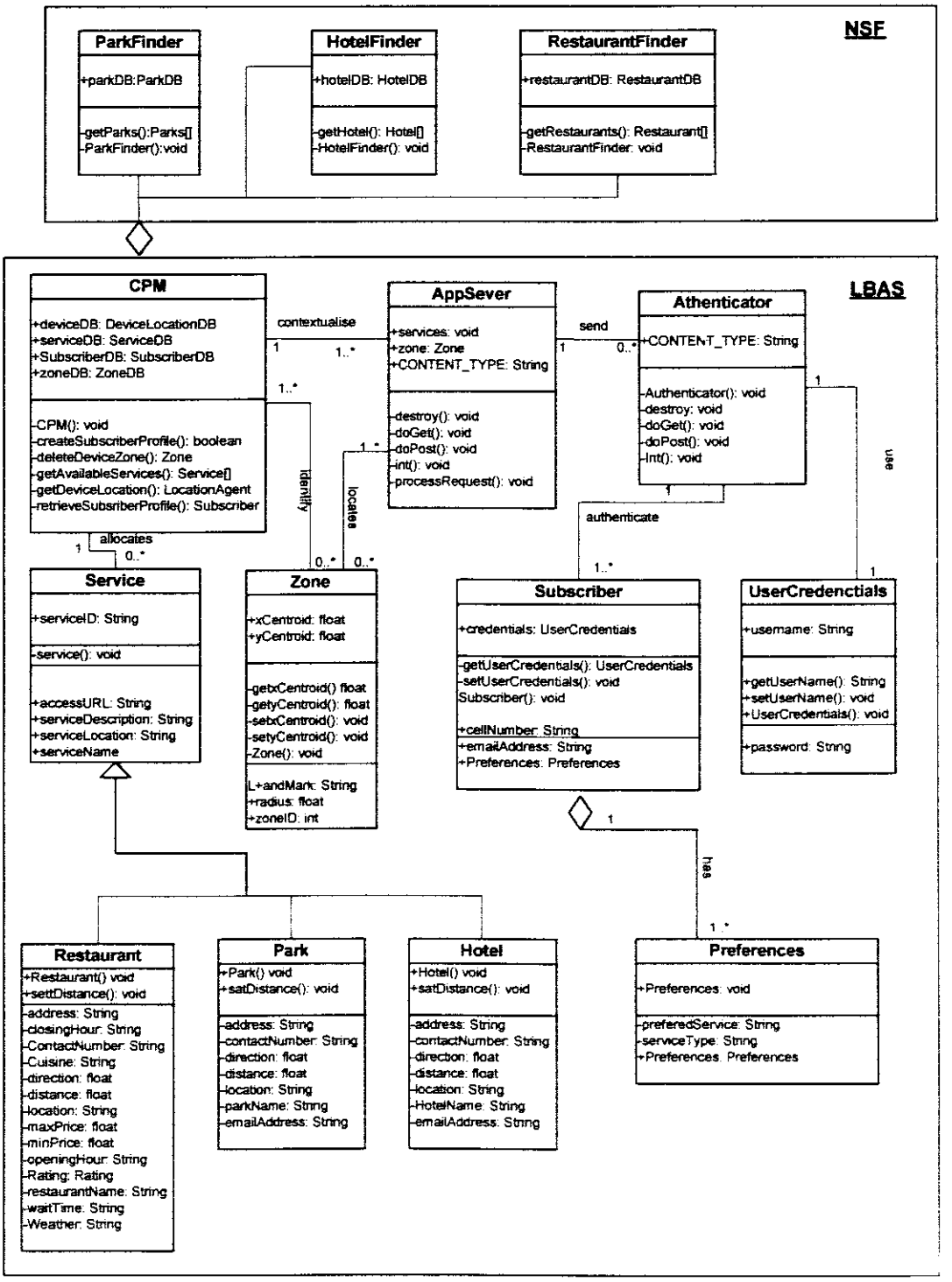


Figure 3. 5 UML Class diagram showing a relationship between the Nearest Service Finder Systems package and the LBAS package.

Figure 3.5 shows the class diagram that illustrates the relationship between all classes participating in the working developed prototype of nearest service finder system. The LBAS system has been implemented by Database Access Logic and Business Logic Package as indicated in figures 3.6 and figure 3.7

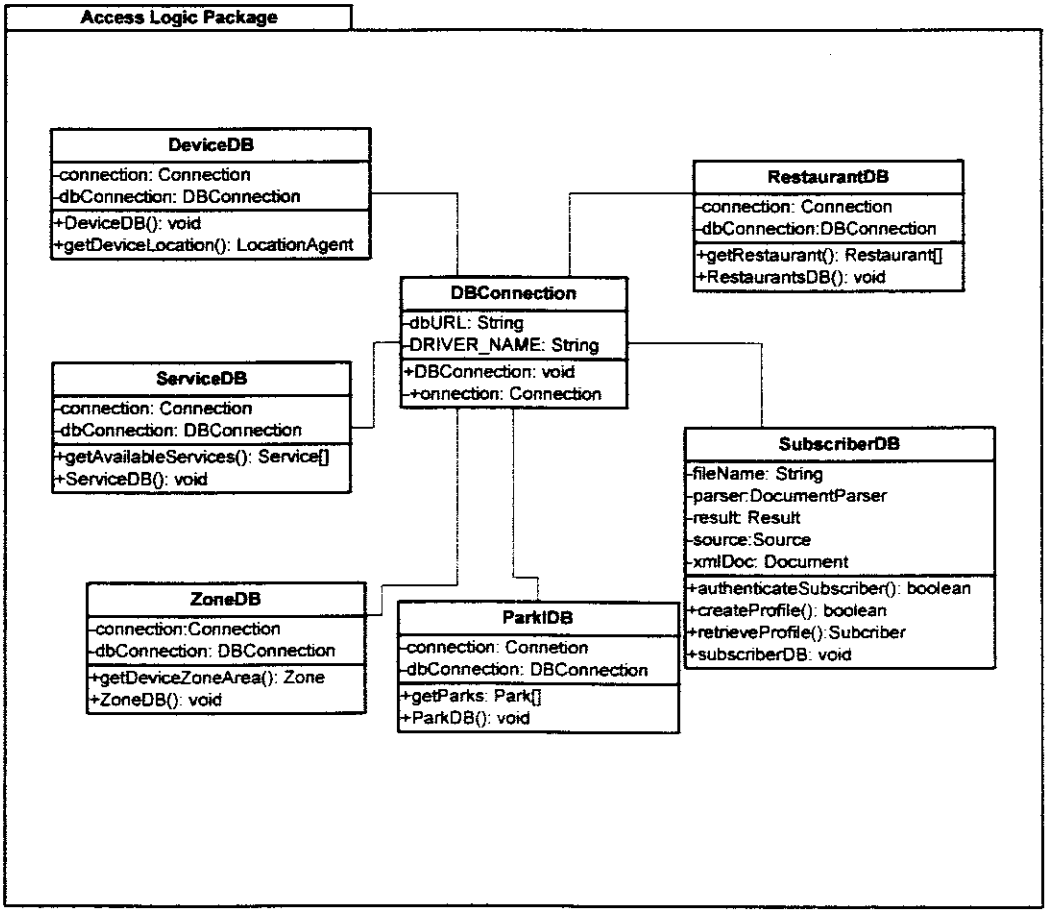


Figure 3. 6 Database Access Logic

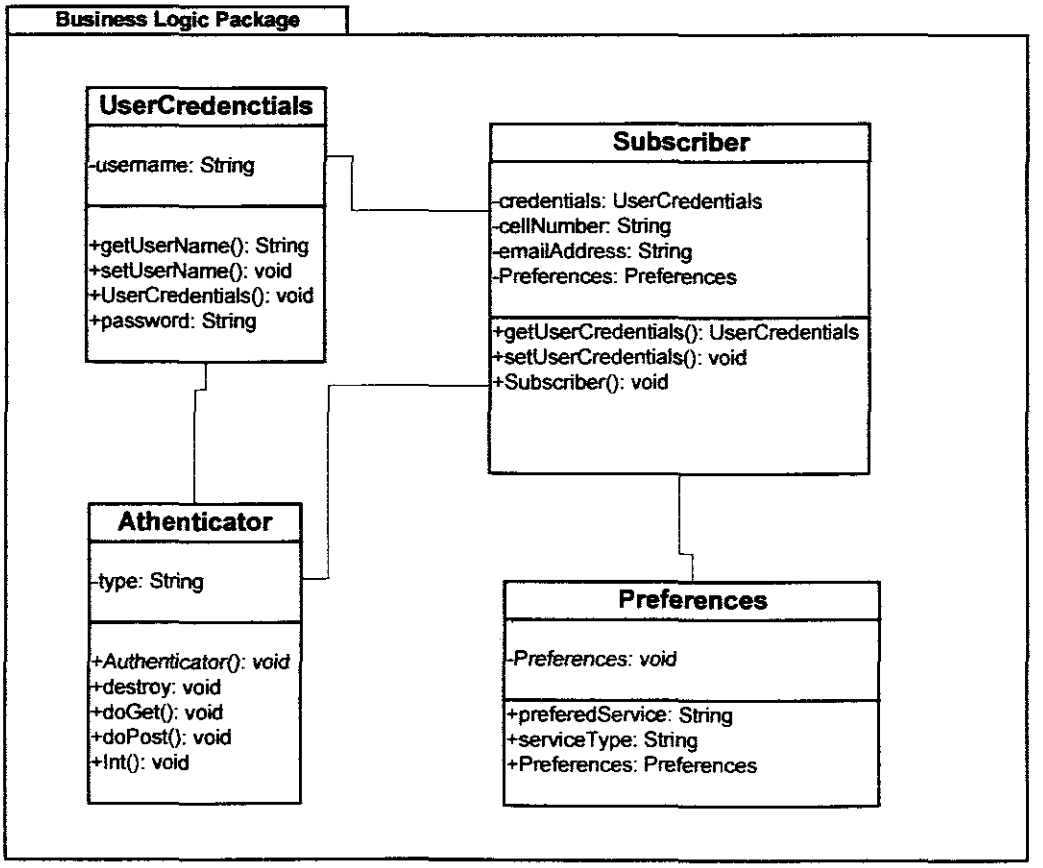


Figure 3. 7 Business Logic Package

3.8.1 Service Discovery

The function discover_Service( ) discovers all services that are advertised and made available to users in that location. See sequence diagram below in Figure 3.8. The mobile user needs to be connected to the network operator to be able to discover available services in his/her location.

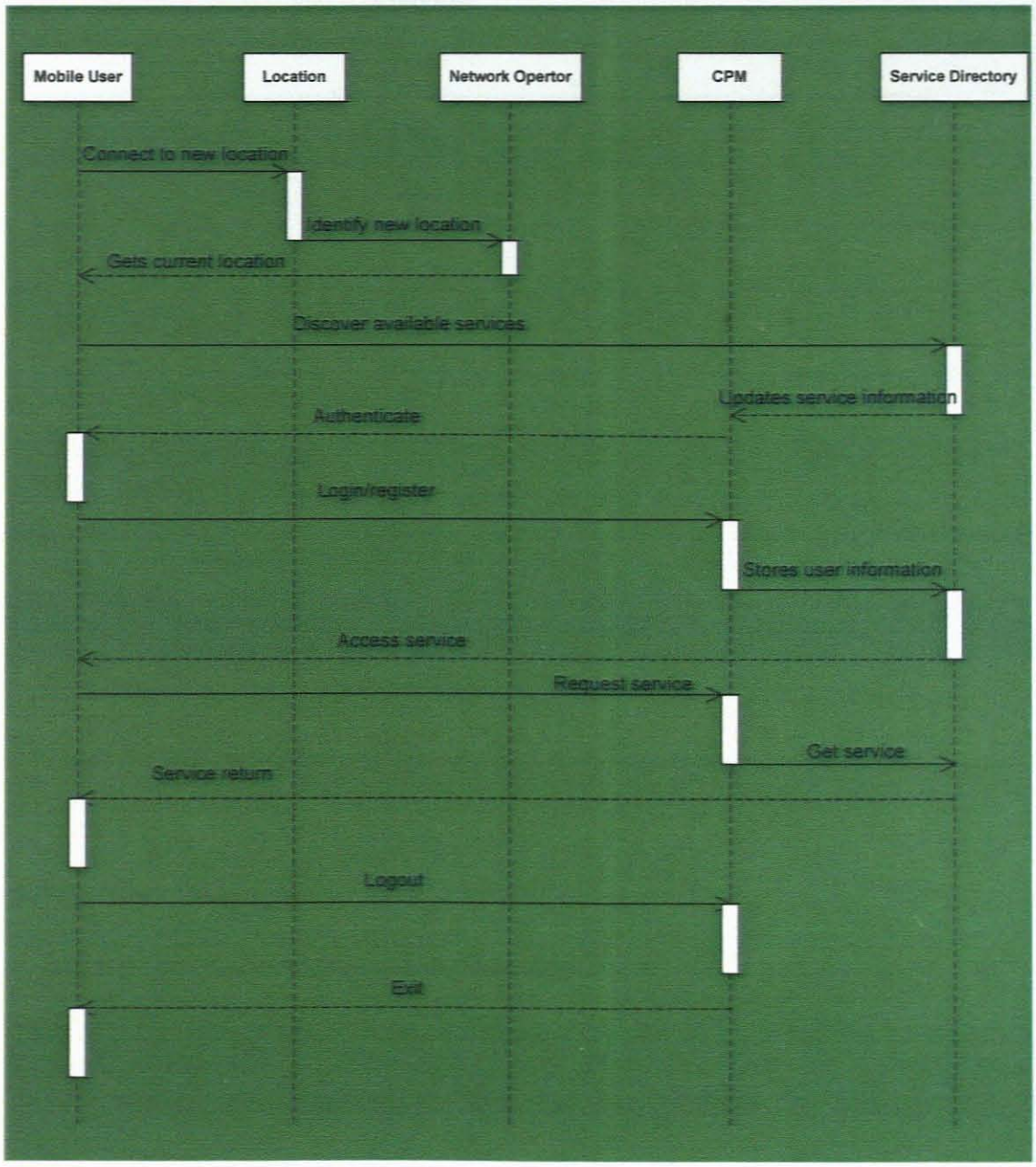


Figure 3. 8 Service discovery sequence diagram

3.8.2 Service Discovery Algorithm

```
Begin

Use zone Id from zone detection

Query zone landmark using zone Id

  If landmark is found
    THEN

          Query all services defined under that landmark
  Else
          No processing
          NotifyUser()
  End If
```

3.8.3 Circle Algorithm

A description of this algorithm is described below:

This algorithm has been adopted to calculate the nearest service to a mobile device which is a restaurant. The function checks the location of all restaurants that are stored in the database. The max radius is set to a certain distance value. All the restaurants within that area are checked and the nearest restaurant is returned. Figure 3.9 gives a graphical illustration of how the circle algorithm works.

Suppose a user wants to find the nearest restaurant. There might be several restaurants in his location, but the set radius value limits the area of interest to few restaurants like three. From these three restaurants the system then returns the nearest one. The pseudo code with algorithm description is as follow:

```
Recordset rs = getRestaurants()

NearestRadius = setValue

Point p1 = getUserLocation()

String NearestRestaurant = "NOT FOUND"

while(rs.moveNext())

{
        Point p2 = rs.getRestaurantPoint()
        double distance = Absolutvalue(p2-p1)

        If

        (distance < nearestRadius)

        {

                nearestRadius = distance

NearestRestaurant = rs.getRestaurantName()
        }
}
return NearestRestaurantInformation
```

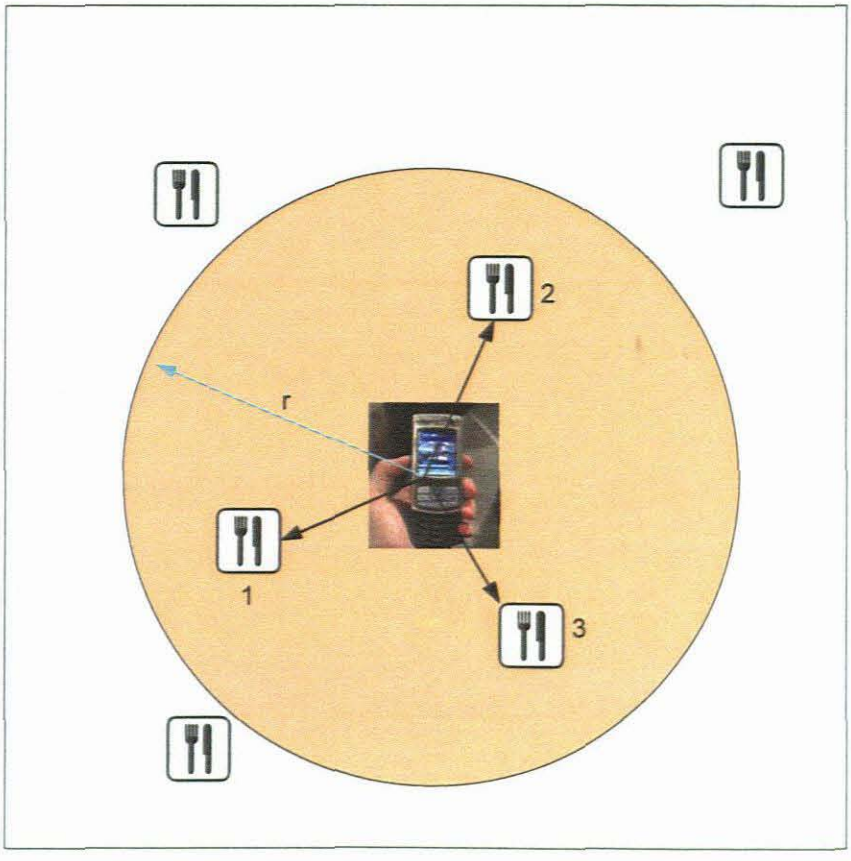


Figure 3. 9 Description of circle Algorithm

3.8.4 Service Registration

The function register_Service allows service suppliers to register their services with the LBAS system which thereafter stores all the information about service supplier and the service itself in the service directory. These services are then made available in the locations defined by service suppliers. After a service has been registered it is automatically advertised to mobile users that are currently in and entering that location which has been defined by the service supplier during service registration.

The pseudo code below describes the service advertisement algorithm. The service registration process is represented by the sequence diagram in Figure 3.10 below

```
If Service not registered
        THEN

                Register service in LBAS

        If service available in LBAS
                THEN

                        Advertise service to location
                        Save service information in service directory
        Else

                        No processing
        End If

Else

                Notify User()
                The service already exists in LBAS
End If
```

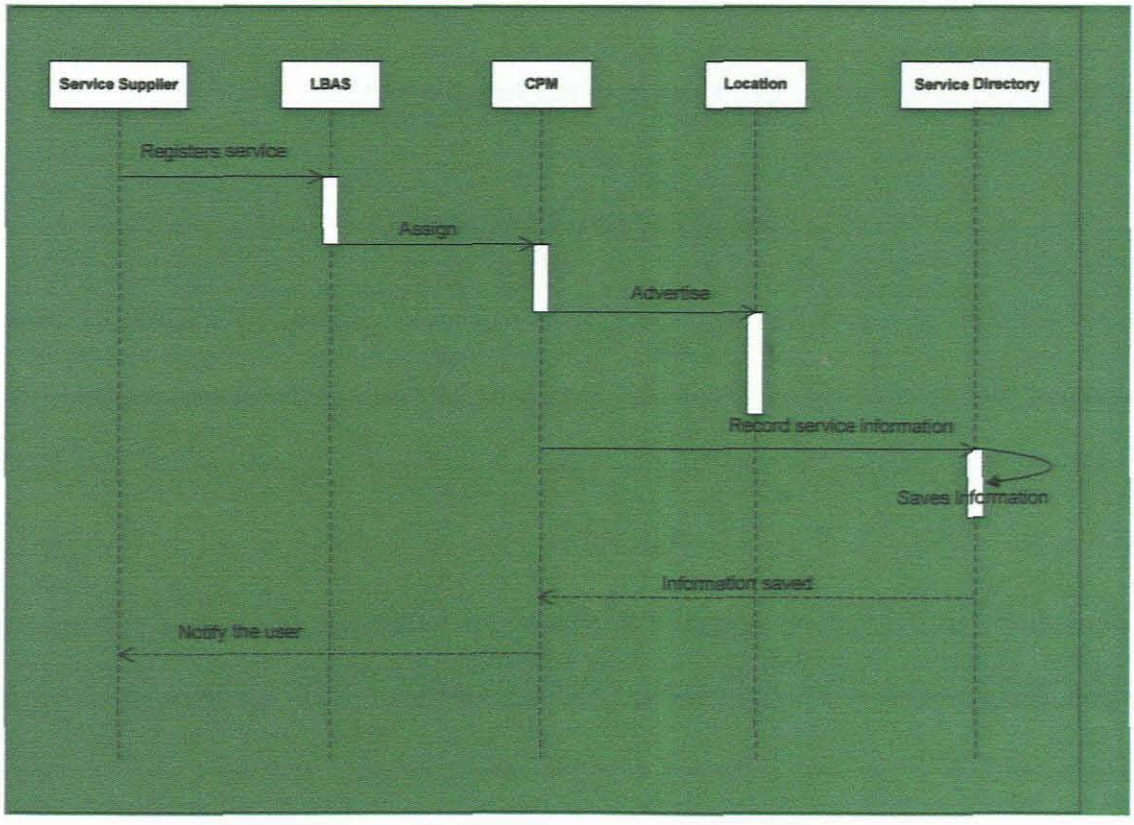


Figure 3. 10 Service registration sequence diagram

CHAPTER FOUR

IMPLEMENTATION

4.1 INTRODUCTION

In the previous chapter the proposed model for location-based service discovery in a mobile environment was presented. The model showed how components of a location based application server in mobile commerce application could be modelled in order for them to perform their task, and how these services could be advertised in the vicinity of the network coverage area by the LBAS so that the user who is entering that area would access services that are available to him/her. To show the usefulness of the proposed model, this chapter presents the implementation of a prototype service, that is, a nearest service finder system like restaurants, hotels and parks services. The LBAS system can accommodate the discovery of more services like nearest garages, hospitals, taxi cabs and etc, but because of time constraints we managed to feature only three services which have been mentioned above. Usability testing for the model and the results obtained are also presented.

4.2 IMPLEMENTATION (ARCHITECTURAL LAYERS)

The implementation architecture consists of three layers as depicted in Figure 4.1, viz: Presentation layer, Business layer and Information layer.

4.2.1 The presentation layer

The presentation layer presents the way interfaces define access to encapsulated system functions without giving the idea of how they are implemented. It accommodates both WEB users and WAP users to ensure equal access to services. The authentication and authorization of clients is encapsulated at this layer through suitable tools like the emulator as the mobile phone interface.

4.2.2 Business layer

The business layer is the heart of the proposed model system which is LBAS. It is the constituent of different modules. The web server consists of web components that handle clients' requests and generate appropriate responses from LBAS. The business layer presents interfaces for registered service suppliers in the service directory that are clearly defined in the WSDL document, other than web services such as user profiling, user authentication, etc. The service supplier system module represents any service supplier system, for example, Restaurant Finder System (RFS) with a service table outlining details about the supplier itself.

4.2.3 Information layer

This layer presents information about service suppliers and user information (user profile).

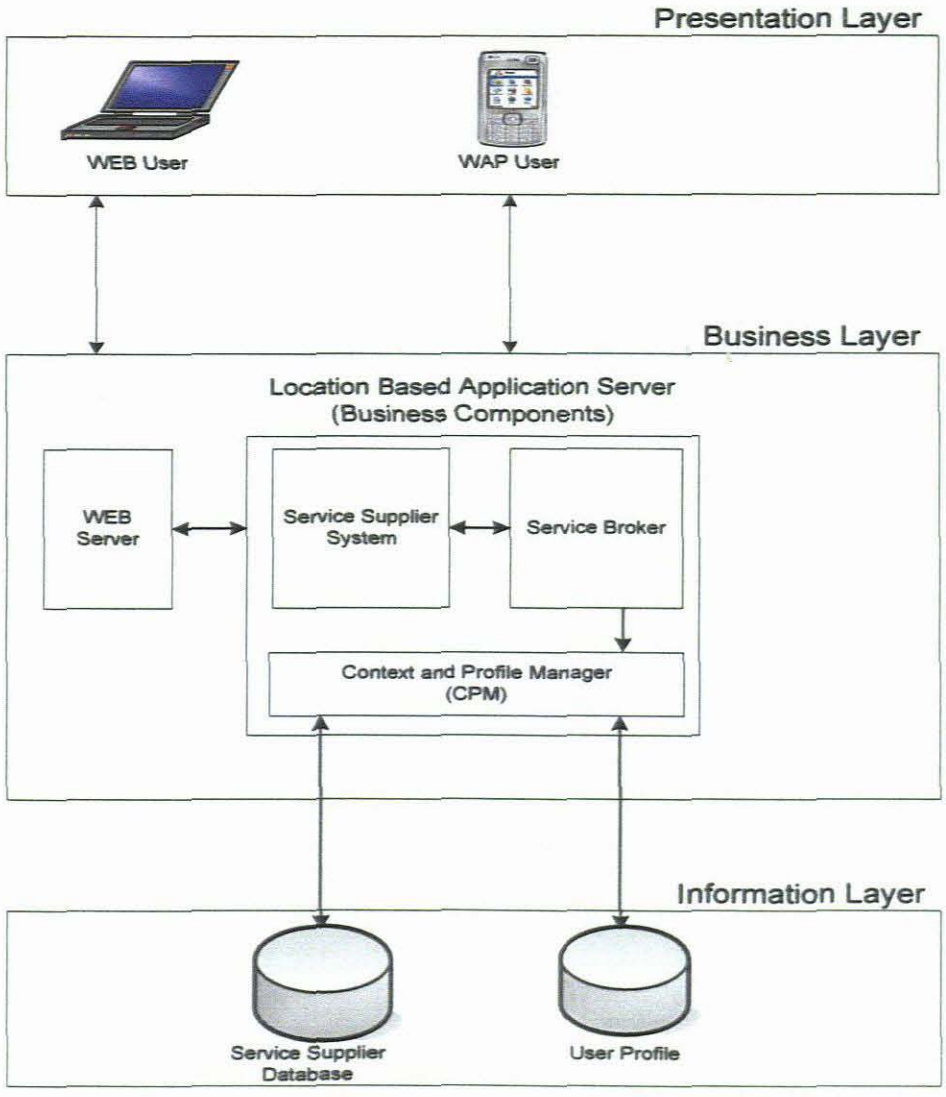


Figure 4. 1 Three Layer Implementation Model Architecture

4.3 ENVIRONMENT SPECIFICATION

To realise the implementation of the Nearest Service Finder prototype the following implementation tools were involved: Borland JBuilder 2005 Enterprise tool, Axis Apache Web Server, Microsoft Visio .Net mobile phone emulator, JBoss Application Server 4.0.5, JAXP (Java API for XML Processing), Servlets framework, JDBC (Java Database Connectivity

framework) and Microsoft Access 2003 tool. We used two desktop machines, one for running our application server and the other one for running the mobile device.

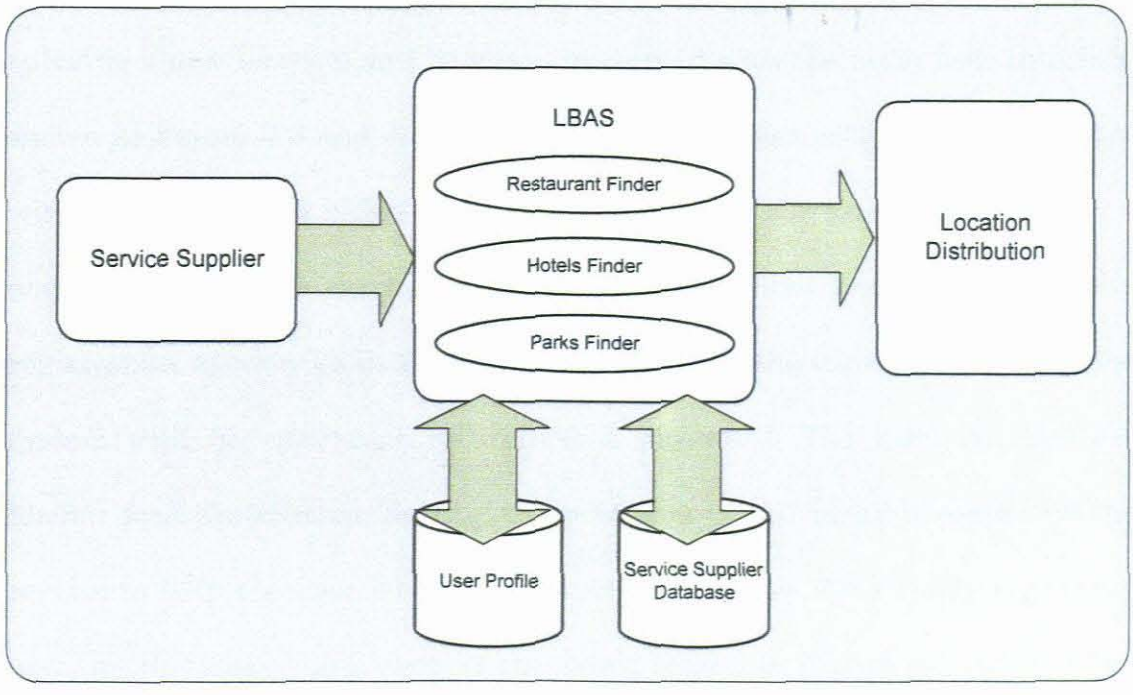


Figure 4. 2 NSF prototype featured into the LBAS architecture.

Figure 4.2 is a schematic diagram showing the Nearest Service Finder system (NSF) prototype plugged into the LBAS architecture. NSF gathers all registered service suppliers in LBAS system. Services registered in LBAS are then advertised by the system broker to distributed locations for any potential mobile user to discover them.

4.4 NSF USER INTERFACE

In designing the graphical user interface of the prototype, we used the Microsoft Visio .Net mobile emulator to make it suitable for our purpose, which is to allow mobile users to discover available services. When the prototype is run, we select location as an indication that a mobile user is entering a new location and he/she is presented with the main user interface shown in Figure 4.3 and 4.4. If it is a first time user, s/he needs to register with the system first before s/he can proceed to use the service. To register, a new user selects the *register* link of Figure 4.5, which leads the user to the registration window as shown in Figure 4.6. Here, the user has to supply the system with his username as well as a password. The user can also set his/her food preferences. Setting the preferences will make it easier for the service to help the user when s/he returns next time. An already registered user, on the other hand, chooses the *Login* button in Figure 4.5, after filling in his /her login information for authentication.

Upon successful login or registration, all available services in that location are displayed in the mobile device with service information – as shown in Figure 4.7 and 4.8 – where the user then selects the service of his/her choice, which ranges from restaurant finder, hotel finder and park finder service in as far as our prototype is concerned.

This information is supplied by the context data source, the CPM component to be specific. As mentioned before, the LBAS system caters for both mobile users and service suppliers; service suppliers can just visit the website and register their service as shown in Figure 4.9 and 4.10. On submission of the form, service registered is saved in the service directory thereafter advertised according to specified locations during registration for discovery by mobile users entering that location.

The outlined screenshots show that a mobile user moving around from one location to another, say, from Mthunzini to Empangeni can automatically discover new available services in the new location. Whenever the mobile device enters a new location, available services are automatically pushed to him whether he is a subscriber or not, but for him to access those services he needs to register as Figure 4.4 shows.

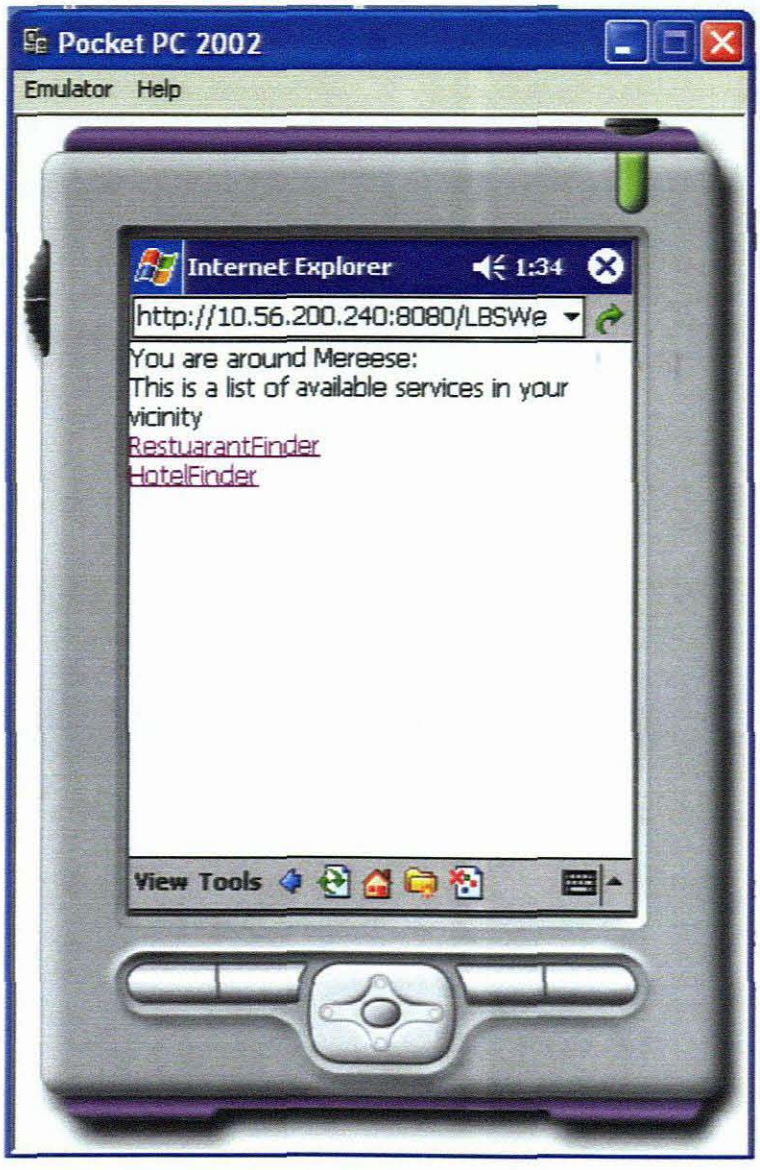


Figure 4. 3 Screenshot for advertised and discovered available service

Figure 4. 4 Screenshot for advertised and discovered available service

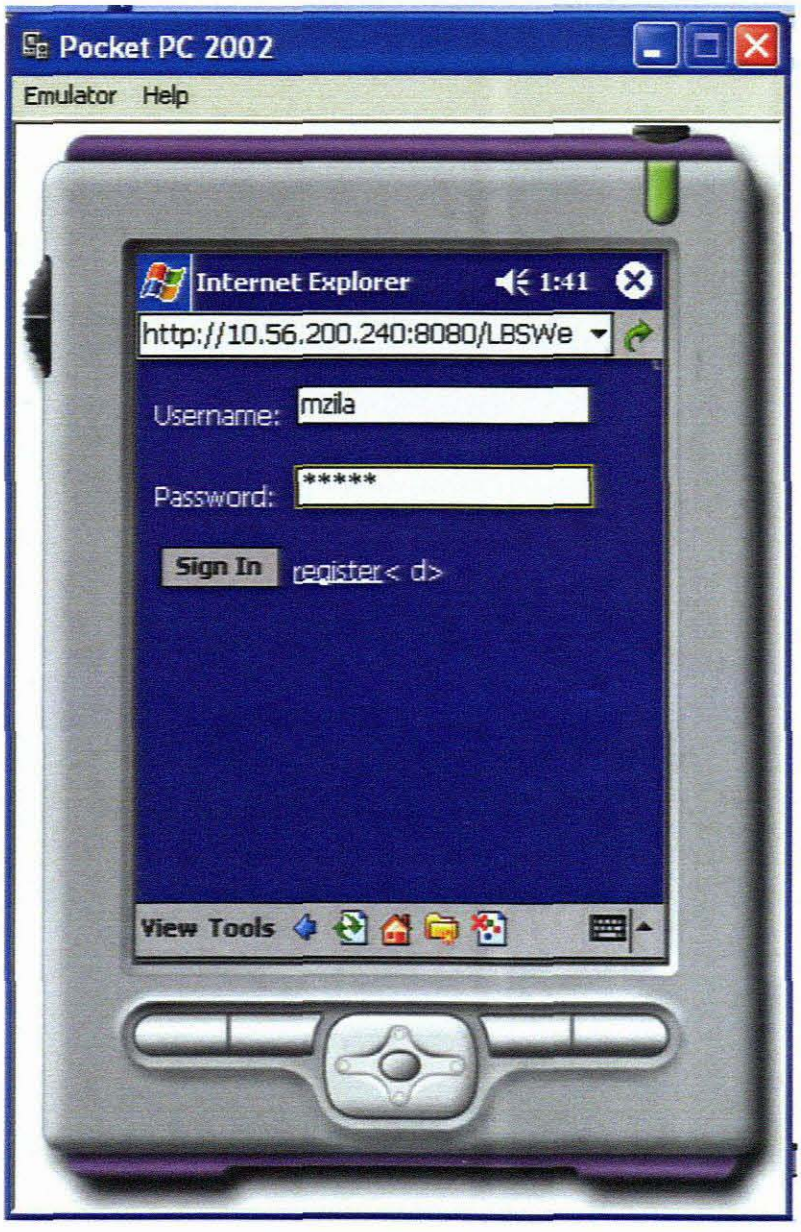


Figure 4. 5 Authentication

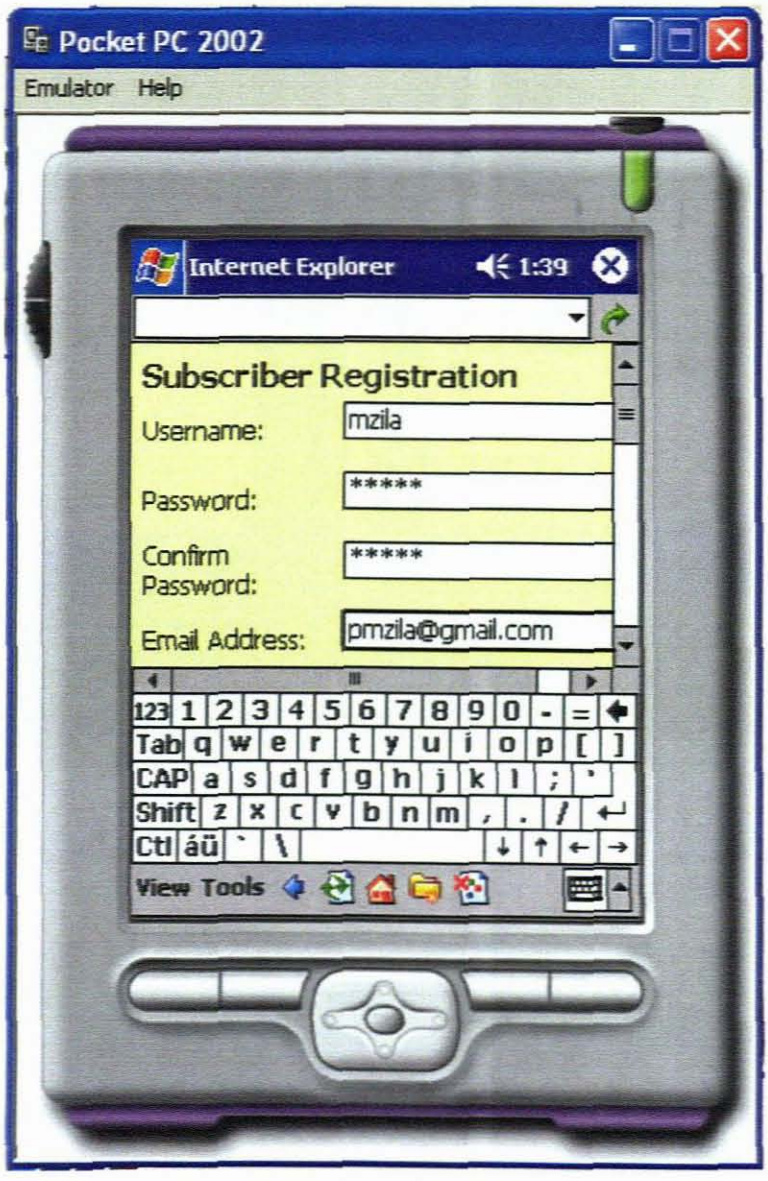


Figure 4. 6 New user registration

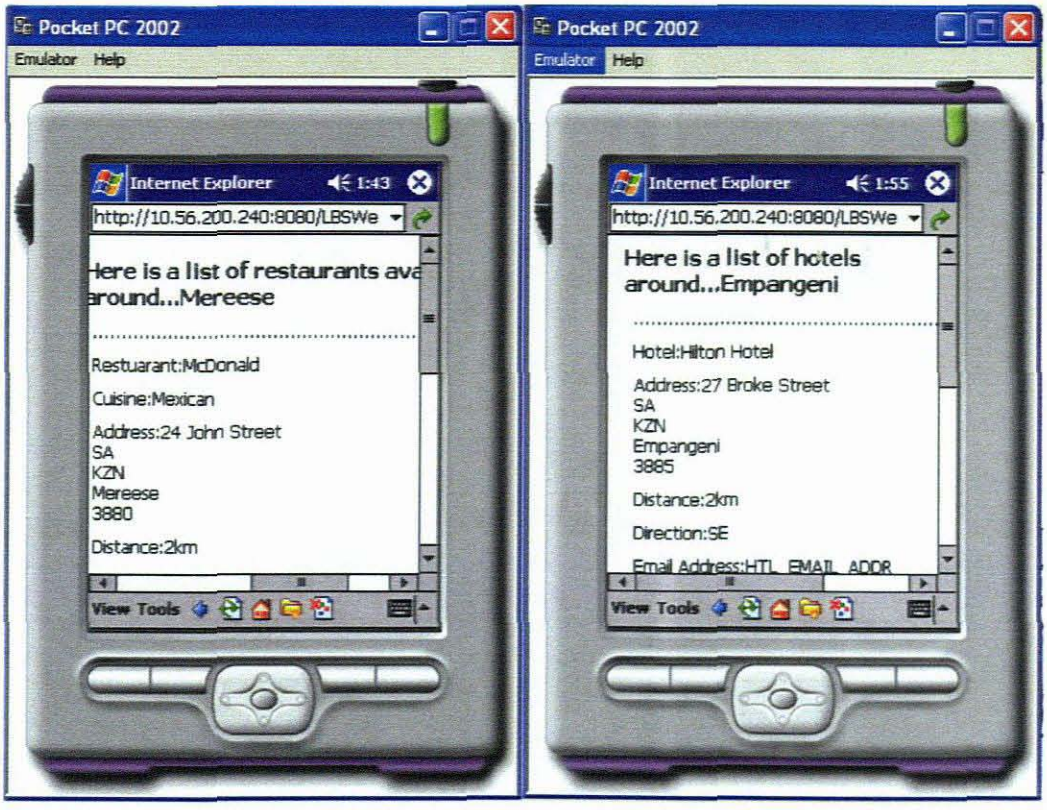


Figure 4. 7 Returned service information

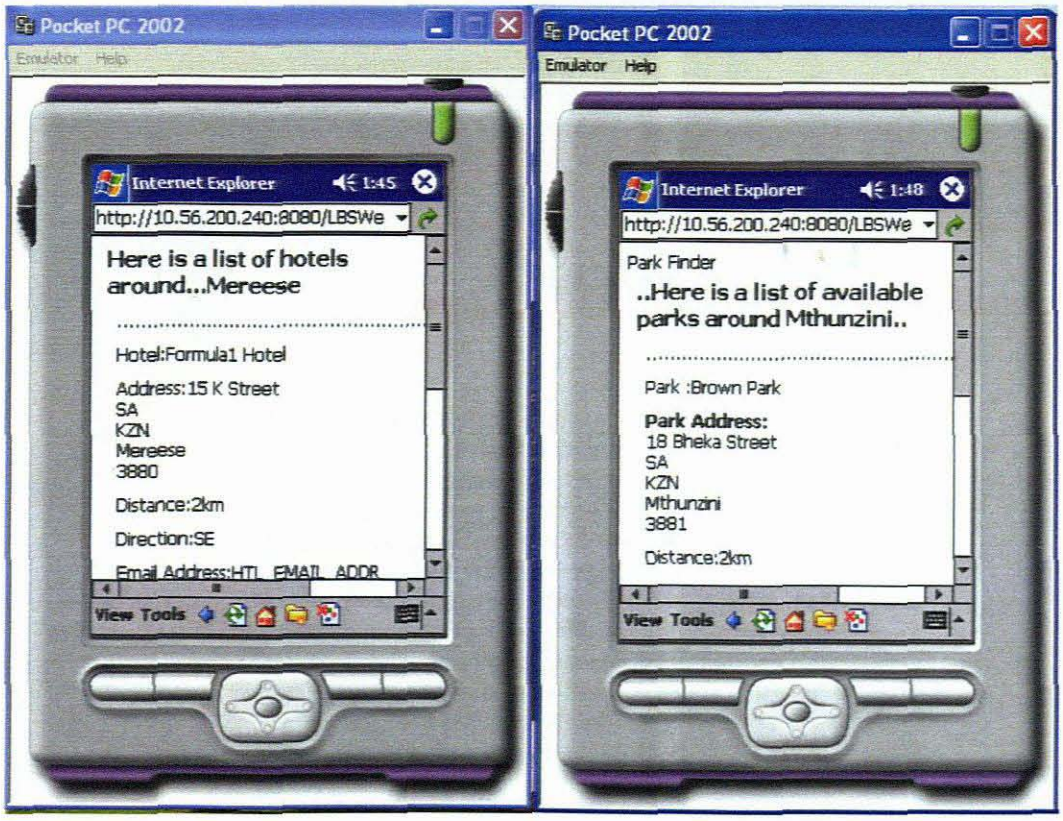


Figure 4. 8 Returned service information

Figure 4. 9 Home page for service registration

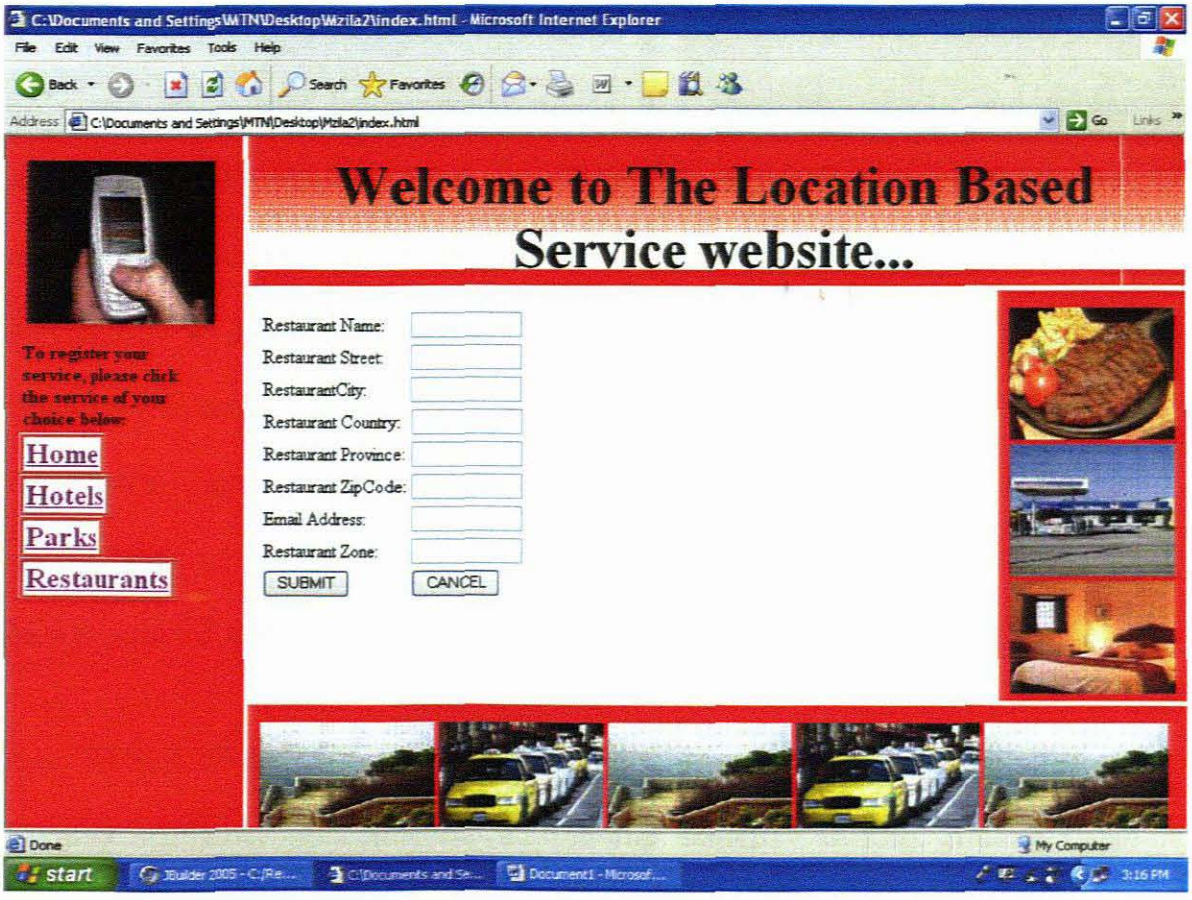


Figure 4. 10 Registration form

4.5 PERFORMANCE EVALUATION

For performance evaluation, we conducted usability testing where we asked different people, from different faculties to take a tour of the system. We then interviewed the same group of people to evaluate the user-friendliness of the system and easy to learn testing. We categorised these people according to their level of understanding the location based service systems and computer related software systems. We had a group of novices, normal and expert

people in computer science related systems. We regarded all interviewees from the faculties of Arts and Education as novices because normally they only use computers for personal needs like checking emails and typing assignments. We also regarded interviewees from the faculties of Commerce and Law as normal candidates and those from the faculty of Science and Agriculture as experts because of their exposure to such systems. From their responses in answering the drafted questionnaire, firstly we tried to find out from three groups defined above after they used the system about accessibility to services if the user is new (not registered) and after the user has registered and login using his/her username and password.

4.6 RESULTS ANALYSIS

Twenty students were interviewed for our usability testing. Among these twenty students, eleven were from the Science and Agriculture faculty, six from Education and Art faculties and three were from the Commerce and Law faculty. They were divided into three groups as follows:

Group A	11 - Science and Agriculture – Experts
Group B	6 - Education and Art - Novices
Group C	3 - Commerce and Law- Normal

Familiarity to LBAS

In trying to get interviewees' exposure and their familiarity with location based services applications and systems; 20 out of 20 interviewees, none was familiar or have used such systems.

Access to advertised services

If non-registered – Not easy is 20 out of 20

If registered – Easy to access is 20 out of 20

Necessity of LBAS

Group A	Strongly Agree	-	7/11
	Agree	-	4/11
	Strongly Disagree	-	0
	Not Sure	-	0
Group B	Strongly Agree	-	2/6
	Agree	-	4/6
	Strongly Disagree	-	0
	Not Sure	-	0
Group C	Strongly Agree	-	1/3
	Agree	-	2/3
	Strongly Disagree	-	0
	Not Sure	-	0

User-friendliness

Group A	Extremely user-friendly	-	4/11
	Moderately user-friendly	-	7/11
	Complicated	-	0
	Extremely complicated	-	0
	Not sure	-	0
Group B	Extremely user-friendly	-	1/6
	Moderately user-friendly	-	2/6
	Complicated	-	2/6
	Extremely complicated	-	0
	Not sure	-	1/6
Group C	Extremely user-friendly	-	0/3
	Moderately user-friendly	-	3/3
	Complicated	-	0
	Extremely complicated	-	0
	Not sure	-	0

Easy to use and learn

Group A	Yes	-	11/11
	No	-	0/11

Group B	Yes	-	6/6
	No	-	0/6
Group C	Yes	-	3/3
	No	-	0/3

User Satisfaction

Group A	Yes	-	11/11
	No	-	0/11
Group B	Yes	-	4/6
	No	-	2/6
Group C	Yes	-	2/3
	No	-	1/3

Figure 4.11 and 4.12 indicates the access level to actual services that are provided by LBAS to both registered and non-registered users. One of our objectives with this system is to limit un-registered users from accessing the services after they have been advertised to them. The essence of this analysis is to ensure that only registered users can access LBAS services. Firstly the interviewee was asked to try and access services before registering then the same person registered first and after this survey the access level was conducted to see if the objective is achieved.

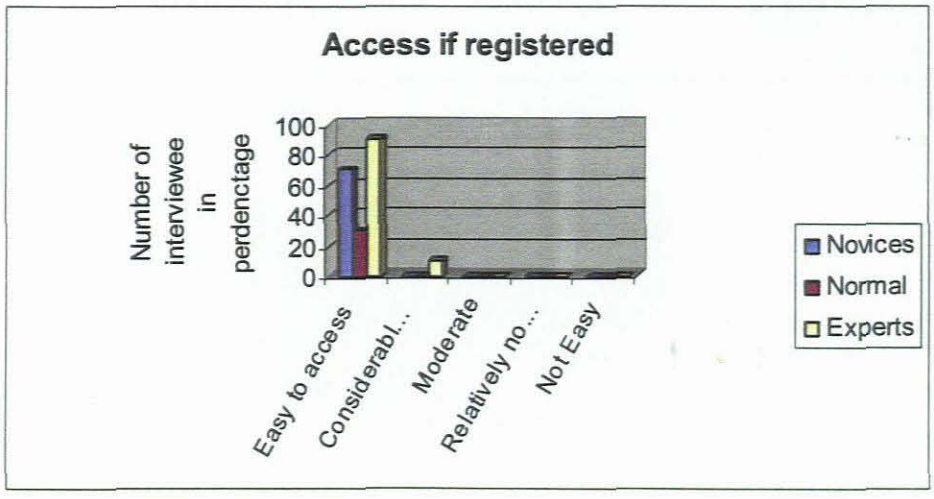


Figure 4. 11

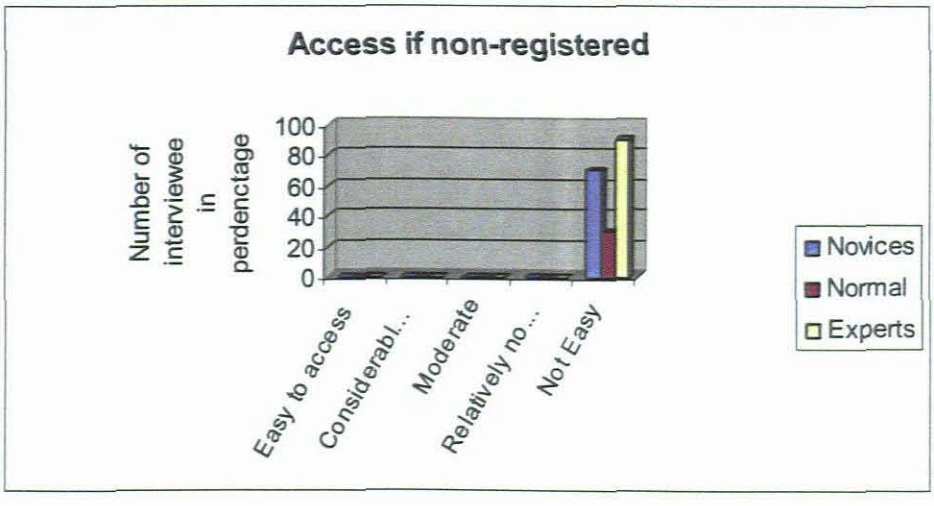


Figure 4. 12

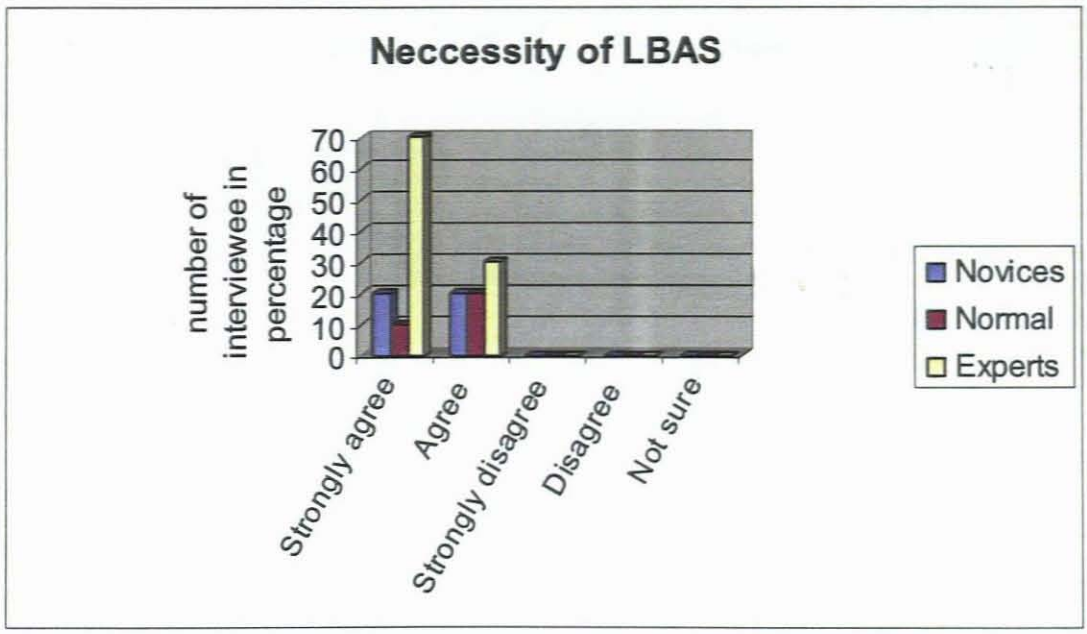


Figure 4. 13

Figure 4.13 shows the percentage of users who think there is a great need for LBAS system and from the bar chart we then realized that though people are not exposed to location based service systems but they saw a great need for such systems to help in services availability awareness whenever a mobile user leaves or enters a new location. Figure 4.14 is the bar chart that indicates the level of user-friendliness of the system during the interaction between the user and the system.

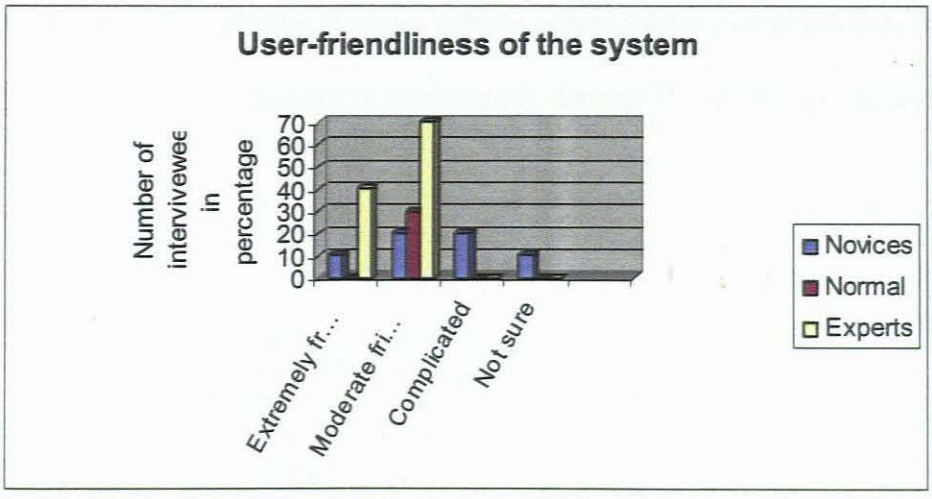


Figure 4. 14

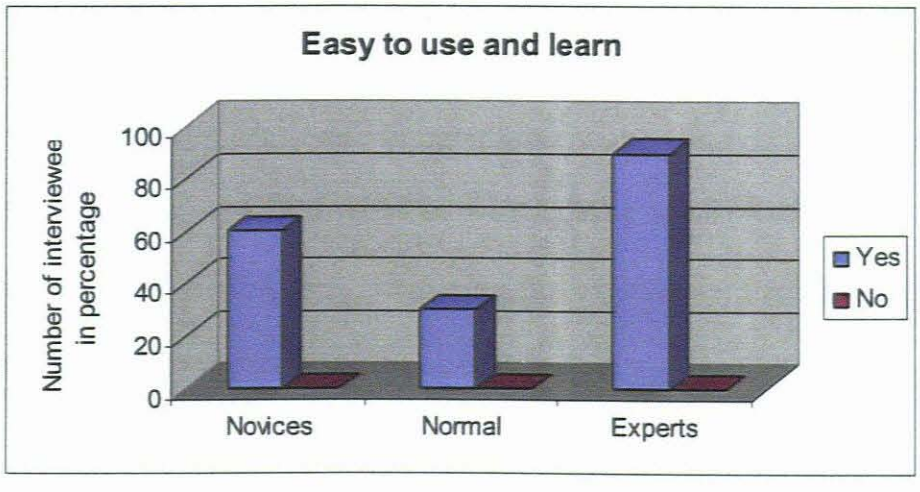


Figure 4. 15

The bar chart in figure 4.15 shows the results about usability and learneabilty of the system, where we asked whether it is easy or not to use and learn functionality of the system.

Finally we carried out the test on user's satisfaction, to establish whether they found the service they were looking for during their tour of the system.

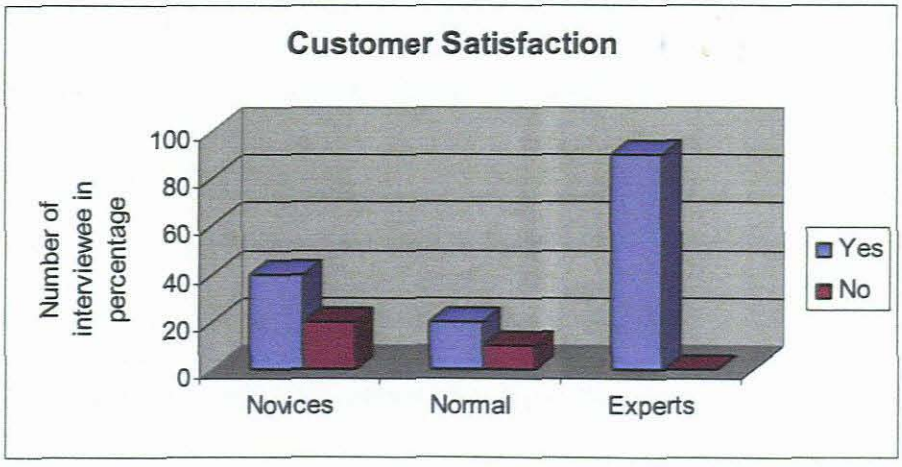


Figure 4. 16

Figure 4.16 shows the user's satisfaction after using the system. This is based on whether the user did find what he was looking for from the system, were the services advertised to him, what was his experience through the tour in as far as LBAS is concerned.

CHAPTER FIVE

CONCLUSIONS AND FUTURE WORK

5.1 CONCLUSIONS

The goal of this research was to develop a service discovery mechanism for location based mobile application services suitable for mobile commerce environment. To achieve the goal of this research work a number of objectives were set. Firstly, it was to define architectural requirements for a LBS for m-commerce based on a publish/subscribe paradigm. To achieve this objective, a literature review was conducted from which relevant work were explored to compile necessary requirements for our developed model. Secondly, it was to the design a location based service discovery mechanism, which raised the research question, how can a mobile user discover a service and service supplier advertise his/her service in a given vicinity? This objective was achieved by designing a model for location-based service discovery in a mobile environment. With this model we were able to show how components of a location based application server in mobile commerce application could be modelled in order for them to perform their task, and how services could be advertised in the vicinity of the network coverage area by the LBAS so that the user who is entering that area would access services that are available to him/her Our next objective of the research was to implement a prototype of

the model to showcase the usefulness of the proposed model. A prototype of a nearest service finder was implemented. Our last objective was to evaluate performance of the prototype system. For performance evaluation, we conducted usability testing where we asked different people, from different faculties, to take a tour of the system. This group was then interviewed to evaluate the user-friendliness of the system and easy to learn testing. In both aspects, the users were satisfied with system.

From our literature review, it was found that there is a high demand for systems that provide location based services to mobile users. Unfortunately, there are very few mobile users that have been exposed or who are aware of such systems and services. The model we developed, the so-called LBAS, will prove to be a useful contribution towards satisfying the need for systems that provide location based services. The main contribution of this dissertation was to closely examine the mechanism whereby every mobile user is capable of service discovery irrespective of being a subscriber or not.

We analyzed the mechanism whereby the mobile phone network operates through a network of local based transmitters scattered all over the world. Every time a mobile phone or other similar device sends or receives a message, it does so via the nearest local transmitter, the geographical location of which is known to the network operator. There are two different

aspects that need to be highlighted concerning our work. Firstly, it regards the use of specific location as the triggering tool to advertise available services in that given vicinity, aiming at reducing the communication overheads. Secondly, concerning the classification of services according to locations in which they are available, since it may become a basis for future research.

5.2 LIMITATIONS

The implemented system had some limitations. One of them is its inability to determine the exact location of the mobile device using device coordinates and the coordinates of the service. Furthermore for service discovery, implementation could not support the use of location as the automatic services discovery component. The latter is the limitation that can be eliminated by acquiring more time for implementation.

5.3 FUTURE WORK

The results obtained from the performance evaluation of the model presented in this dissertation helped in exposing some areas that still need to be improved in order for the model to meet its full potential. The future work may consist of addressing the limitations highlighted in section 5.2.

REFERENCES

[Barnes, S.J. 2003]. Known by the network: The emergence of location-based mobile commerce. In E.P. Lim and K. Siau (Eds.) Advances in Mobile Commerce Technologies, Hershey: Idea Group Publishing, 171–189.

[Beadle, H., Harper, B., Maguire, G., and Judge, J. 1997]. Location aware mobile computing. In Proc. International Conference on Telecommunications (ICT'97), Melbourne, Australia, IEEE, 1319-1324.

[Beaulieu, M. & Cooper, M. 2001], *Wireless Internet Applications and Architecture.* Addison-Wesley.

[Berger, S., Lehmann, H. & Lehner, F. 2003]. Location-based Services in the Tourist Industry. *Information Technology & Tourism* 5: 4, 243–256.

[Bisdikian, C., Christensen, J., Davis, II, J., Ebling, M. R., Hunt, G., Jerome, W., Lei, H., Maes, S., and Sow, D. 2001] Enabling location-based applications. In M. Devarakonda, A. Joshi, and M. Viveros (Eds.) Proc. 1st international workshop on Mobile commerce, New York: ACM Press, 38–42.

[Boucher, S., 2005]. *Software-Engineering Issues in Location-Based Services.* Mémoire de Diplôme d'Etudes Approfondies en Informatique, Université Libre de Bruxelles, Brussels, Belgium, 2005.

[Brookes W., E. Lawrence, R. Steele, and E. Chang, 2005]. Proc. 4th International Conference on Mobile Business, Sydney, Australia, 11-12 July 2005. Los Alamitos: IEEE Computer Society.

[Buszko D., Lee W., and Helal A., 2001] "Decentralized Ad-Hoc Groupware API and Framework for Mobile Collaboration," Proceedings of the ACM International Conference on Supporting Group Work (Group'01).

[Calhaus, F.; Pereia, C.; Costa, P. & Botelho, L. 2004], Agent Technology Extended with Mobile devices
.http:// iscte.pt/~luis/papers/ExtendedAgentsTech.pdf [accessed on 02-10-2005].

[Castro P., Chiu P., Kremenek T., and Muntz R., 2001] A probabilistic location service for wireless network environments. In Ubiquitous Computing 2001, Atlanta, GA, September 2001.

[Guanling Chen and David Kotz, 2000] A Survey of Context-Aware Mobile Computing Research, Dartmouth Computer Science Technical Report TR2000-381

[Daniel Barbara, 1999] A Survey on Transactions on Knowledge and Data Engineering, IEEE VOL. 11, NO.1, Mobile Computing and Databases.

[Dao, D., Rizos, C., and Wang, J. 2002]. Location-based services: Technical and business issues. GPS Solutions 6(3), 169–178.

[Diplomarbeit Dr.-Ing., 2002] Discovery, Filtering and Management of Services in Automobile Ad Hoc Networks Master's thesis, Institute of Communication Networks, Munich University of Technology.

[Directive, 2002] Directive on privacy and electronic communications of the European Parliament and of the Council, 12 July 2002.

[Dommety, G. and Jain, R. 1996] Potential networking applications of Global Positioning Systems (GPS). CS Dept., The Ohio State University, Tech. Rep. TR-24.

[D'Roza, T. and Bilchev, G. 2003] An overview of location-based services. BT Technology Journal 21(1), 20–27.

[Eriksson, I. 2003]. Working Together for the Future of European Tourism. Final report of the Working Group "Mobile Services for Tourism".

[Grajski, K. A. and Kirk, E. 2003] Towards a mobile multimedia age location based services: A case study. Wireless Personal Communications 26(2-3),105–116.

[Herring, J. 2001] Editorial: Quality is the future of geoprocessing. GeoInformatica 5(4), 323-325.

[Hjelm, J. 2002]. Creating Location Services for the Wireless Web. New York: John Wiley & Sons.

[Jagoe, A. 2003]. Mobile Location Services: The Definitive Guide. Upper Saddle River, NJ: Prentice Hall.

[Jose, R., Moreira, A, Rodrigues, H., and Davies, N. 2003] The AROUND architecture for dynamic location-based services. Mobile Networks and Applications 8(4), 377-387.

[Joshi A., Weerawarana S., and Houstis E.N, 1997] "On Disconnected Browsing of Distributed Information", in Proc. IEEE Research Issues in Data Engineering (RIDE '97), pp 101-107,

[José Manuel, 2006] Design and Implementation of a Device Description Repository Telefónica' Position Paper Date 31/05/2006 Cantera Fonseca

[Kaasinen, E. 2005]. User acceptance of location-aware mobile guides based on seven field studies. Behaviour and Information Technology 24(1), 37-49.

[Kaasinen, E. 2003] User needs for location-aware mobile services. Personal and Ubiquitous Computing 7(1), 70–79.

[Kagal L., Korolev V., Chen H., Joshi A., Finin T., 2001]. "Centaurus: A Framework for Intelligent Services in Mobile Environment", in Proc. IEEE The 21st International Conference on Distributed Computing Systems (ICDCS-21), pp 195-201

[Katasonov A, Veijalainen J, Markkula J, 2003], "Mobile E-Commerce and Location-Based Services": Technology and Requirements Proceedings of the 9th Scandinavian Research Conference on Geographical Mobile Location, University of Jyväskylä, Information Technology Research Institute, 2003

[Köhne, F., Totz, C., and Wehmeyer, K. 2005] Consumer preferences for location based service attributes: a conjoint analysis. Int. J. Management and Decision Making 6(1), 16–32.

[Kwon Y.J., Bouju A., and Claramunt C., 2004] Proc. 4th Workshop on Web and Wireless Geographical Information Systems, Koyang, Korea, 25-27 November 2004, LNCS vol. 3428, Berlin: Springer.

[Maguire, D. 2001] Mobile Geographic Services Come of Age: ESRI Dives into Wireless Markets. *GeoInformatics* 4, March (2001) 6–9.

[Markkula, J. 2001] Dynamic geographic personal data - New opportunity and challenge introduced by the location-aware mobile networks. Cluster Computing 4(4),369-377.

[MDM. 2004] Proc. Intl. Conf. on Mobile Data Management, Berkeley, USA, 19- 22January 2004. Los Alamitos: IEEE Computer Society.

[Muthukrishnan S., Rittwik Jana, 2001], "Location Based Services in a Wireless WAN using Cellular Digital Packet Data (CDPD)", to appear in Proc. 2nd ACM International Workshop on Data Engineering for Wireless and Mobile Access (MobiDE01).

[Nicklas, D. and Mitschang, B. 2004] On building location aware Applications using an open platform based on the NEXUS augmented world model. Software and Systems Modeling 3(4), 303-113.

[Open Geospatial Consortium, 2004], OpenGIS Location Services (OpenLS): Core Services [Parts 1-5] (OLS Core) 1.0, http://www.opengeospatial.org/specs/?page=specs, accessed Jan 2005.

[Osman, Z., Maguire, M., and Tarkiainen, M. 2003] Older users' requirements for location based services and mobile phones. In L. Chittaro (Ed.) Proc. Mobile HCI'03, Udine, Italy, LNCS vol. 2795, Berlin: Springer, 352–357.

[Rao, B. and Minakakis, L. 2003]. Evolution of mobile location-based services. Communications of the ACM 46 (12), 61–65.

[Ratsimor O, Joshi A and Finin T, 2003] Agents2Go: An Infrastructure for Location-Dependent Service Discovery in the Mobile Electronic Commerce Environment

[Rekesh, John, 1999] "UPnP, Jini and Salutation - A look at some popular coordination framework for future network devices." Technical Report, California Software Lab, http://www.cswl.com/whiteppr/tech/upnp.html. [Accessed August 2005]

[Sal] Salutation Consortium. Salutation Architecture Specification 2.0c. www.salutation.org, June 2006.

[Schiller, J and Voisard, A, 2004] Location-Based Services, Morgan Kaufmann. Publishers Incl. San Francisco, CA, USA, Forth Edition

[Sharma, V.; Gahlout, S.S.; Kumar, A.; Roy, A.K. & Gupta, A. 2004], GIS Based Emergency Planning and Response System for Gujarat State, Global Spatial Data Infrastructure -7th Conference, Bangalore, India.

[Singh Gaurav, 2004] Framework for Location Based Emergency Services in India Institute Of Remote Sensing, National Remote Sensing Agency (NRSA), DEPARTMENT OF SPACE, DEHRADUN, INDIA

[Singh, M.P. 1998], Agent Communication language: Rethinking the principles.http://www.csc.ncsu.edu/faculty/mpsingh/papers/mas/computer-acl-98.pdf [accessed03-10-2005].

[Statistisk S, 2005] Satellittregnskap for turisme, 1996-2003. Available at: www.ssb.no/emner/09/01/turismesat, [Accessed on March 2006]

[Strang T. and Linnhoff-Popien C., 2005] Proc. 1st Intl. Workshop on Location- and Context-Awareness at Pervasive'05, Oberpfaffenhofen, Germany, May 12-13, 2005, LNCS vol. 3479, Berlin: Springer.

[Sumi Helal , 2003], Konark – A Service Discovery and Delivery Protocol for Ad-Hoc Networks Computer and Information Science and Engineering Department University of Florida, Gainesville, FL-32611, USA

[Tao Gu, Qian H. C., Yao J. K., Pung H. K., 2003] "An Architecture for Flexible Service Discovery in OCTOPUS", Proceedings of the 12th International Conference on Computer Communications and Networks (ICCCN), Dallas, Texas, October 2003.

[Tilson, D., Lyytinen, K., and Baxter, R. 2004] A Framework for selecting a location based service (LBS) strategy and service portfolio. In Proc. 37th Hawaii International Conference on System Sciences, Los Alamitos: IEEE Computer Society, CD-ROM

[Unni, R. and Harmon, R. 2003] Location-based services: Models for strategy development in m-commerce. In D.F. Kocaoglu and T.R. Anderson (Eds.) Proc. Portland International Conference on Management of Engineering and Technology (PICMET'03), Portland State University, 416–424.

[Workshop on Usability] Utility and Ethics of Location Based Services at NordiCHI'04, Tampere, Finland, 23 October 2004. Web-page http://www.uta.fi/hyper/projektit/ken/. [Accessed March 2006]

[Wyse Jim, 2004] Primary Area of Research, Location-Based Mobile Commerce, University of Newfoundland Available at: www.mum.ca/wyse/research/researchinterest [Accessed June 2006]

[You-Heng Hu and Samsung Lim, 2005] A Novel Approach for Publishing And Discovery of Location-based Services; Proceedings of SSC 2005 Spatial Intelligence, Innovation and Praxis: The national biennial Conference of the Spatial Sciences Institute,

[Yu, S.; Spaccapietra, S.; Cullot, N. & Aufaure, M.A. 2003], User Profiles in Location Based Services: Make Human More Nomadic and Personalized. http://lbdwww.epfl.ch/e/publications_new/ articles.pdf/NG2I03.pdf (accessed on 13-09-2005)

[Zipf, A. & Aras, H. 2002], Proactive Exploitation of the spatial context in LBS through interoperable integration of GIS-services with a Multi agent system, 5th AGILE conference on Geographic Information Science, Spain.

[Zipf, A. & Malaka, R. 2001]. Developing location based services for tourism – The service providers view. In Sheldon, P. J., Wöber, K. W. & Fesenmaier, D. R. (eds.): *Information and Communication Technologies in Tourism*, 83–92. 8th International Congress on Tourism and Communications Technologies in Tourism (ENTER 2001), Montreal, Canada. Springer.

Appendix

A. User Manual

Start site/service discovery

As you enter a new location, you will always first be met with LBAS start page, as shown in Figures A1,A2 and A3 which indicate all available services in that location, for example, in Figure A1 only Restaurant Finder Service and Hotel Finder Service are available in Mareese location. If you enter the Mthunzini location you will be met with advertised services in Mthunzini which are Hotel Finder Service, Park Finder Service and Restaurant Finder Service. You are given the option to choose which service you are interested in, and if you click the option of your choice you will be authenticated by the system to provide your login details or sign in depending on whether you are a registered user or unregistered user respectively as shown in Figure A5.

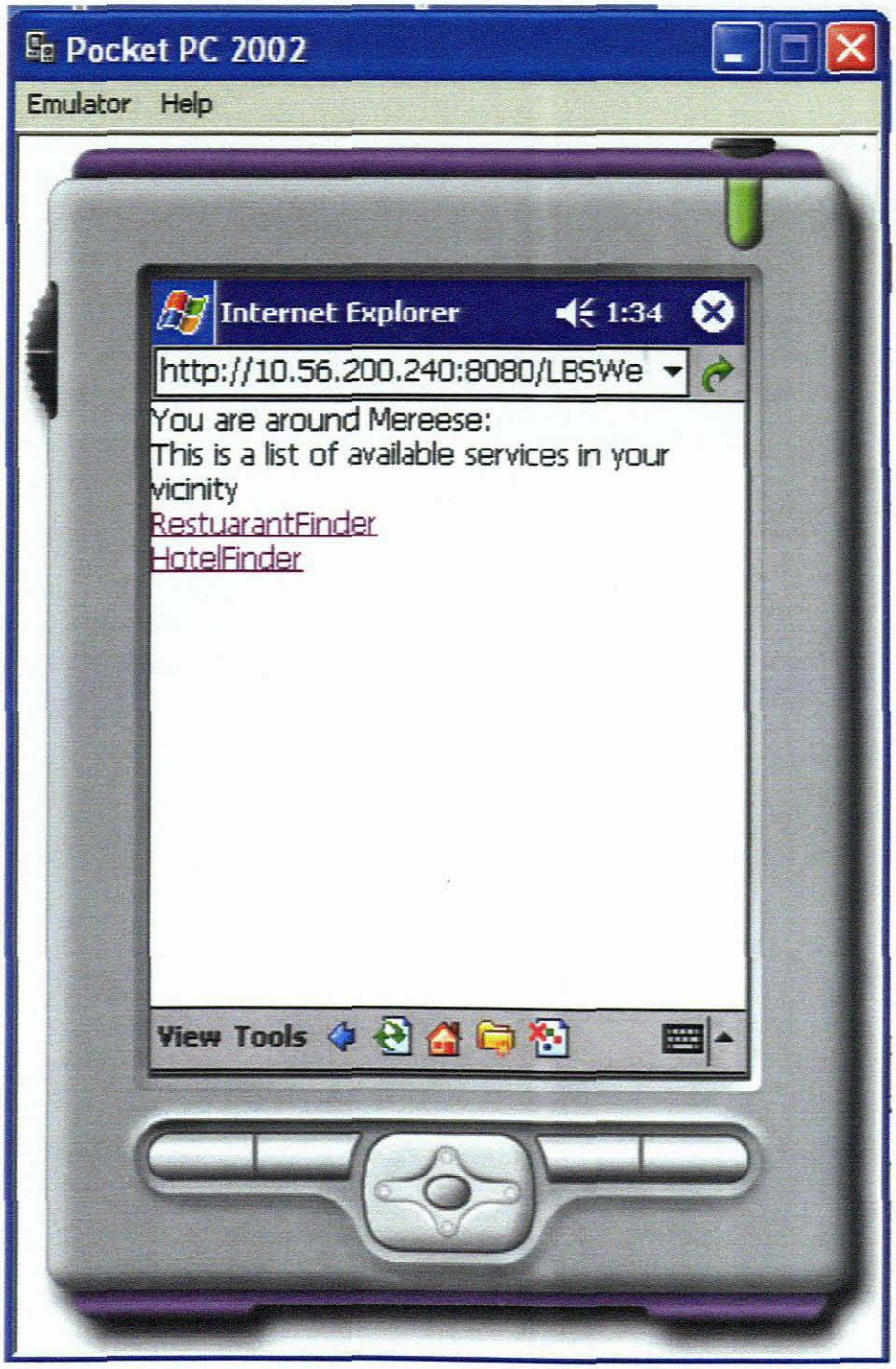

Pocket PC 2002
Emulator Help
Internet Explorer 1:34
http://10.56.200.240:8080/LBSWe
You are around Mereese:
This is a list of available services in your vicinity
RestuarantFinder
HotelFinder
View Tools

Figure A 1

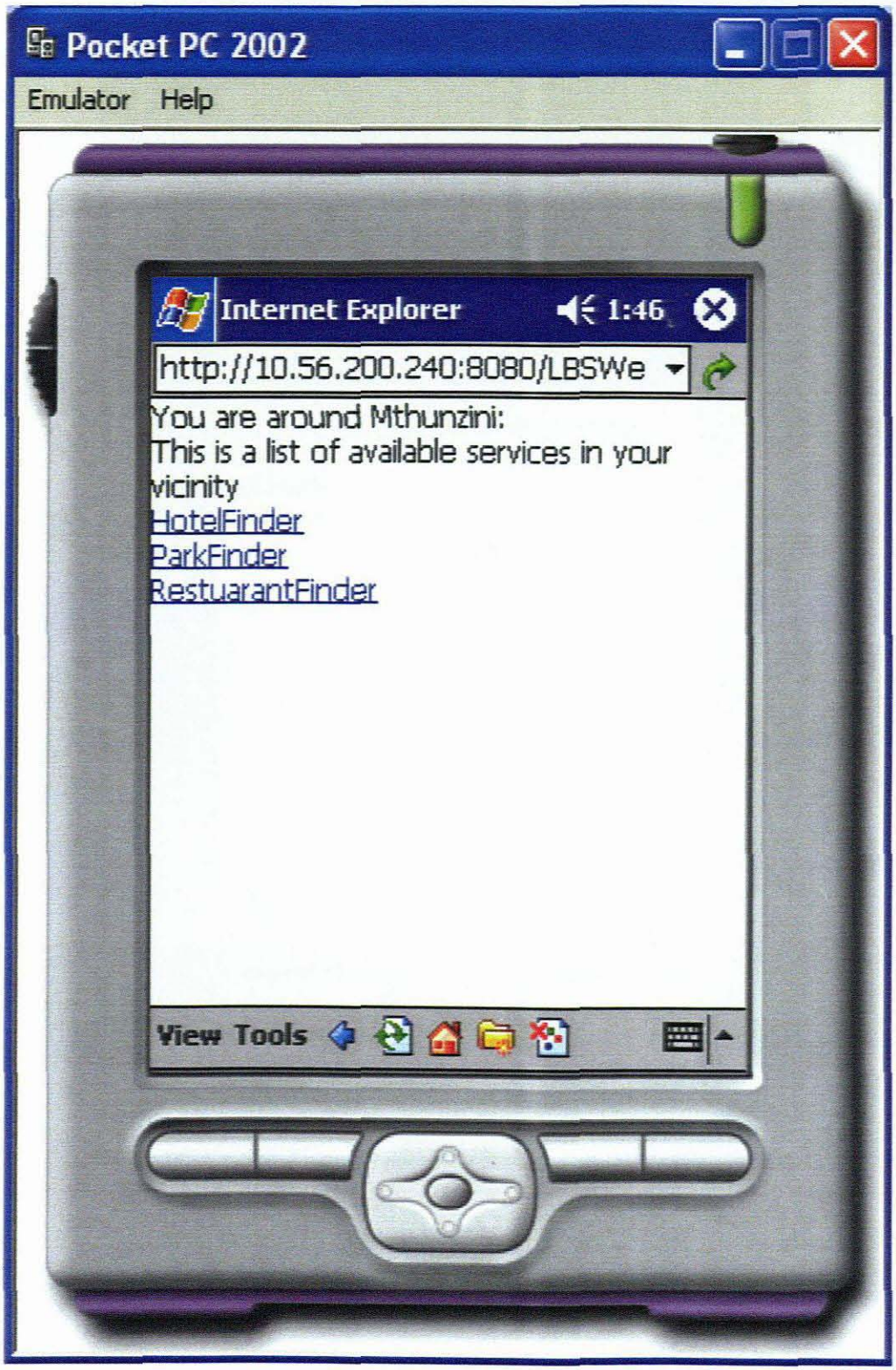

Pocket PC 2002
Emulator Help
Internet Explorer 1:46
http://10.56.200.240:8080/LBSWe
You are around Mthunzini:
This is a list of available services in your vicinity
HotelFinder
ParkFinder
RestuarantFinder
View Tools

Figure A 2

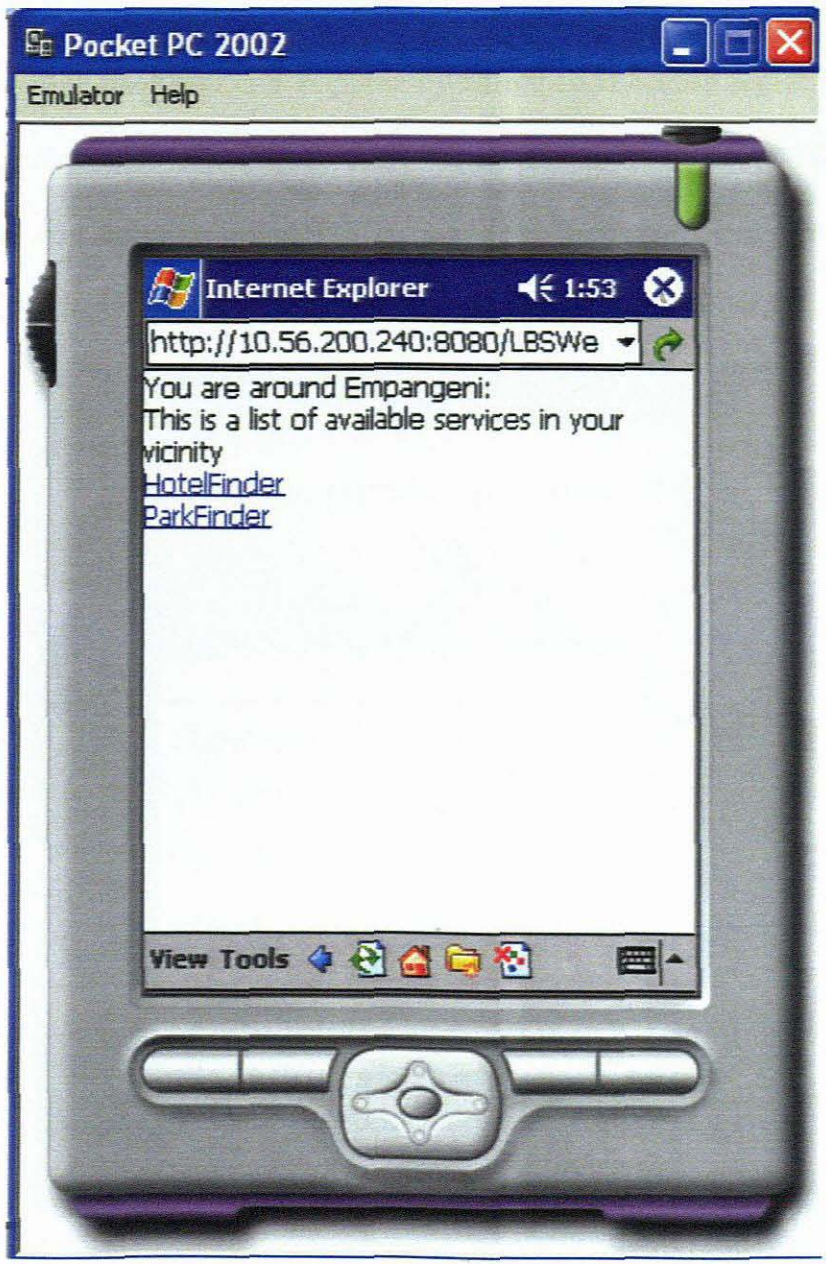

Pocket PC 2002
Emulator Help
Internet Explorer
1:53
http://10.56.200.240:8080/LBSWe
You are around Empangeni:
This is a list of available services in your vicinity
HotelFinder
ParkFinder
View Tools

Figure A 3

Registration of new user

If you are a new user you can register by filling in the user registration form by clicking *Register* (see Figure A5). When fields have been filled in (see Figure A4), you have to click "OK" in order to register as a user of LBAS. After that the login page will automatically appear (see Figure A5).

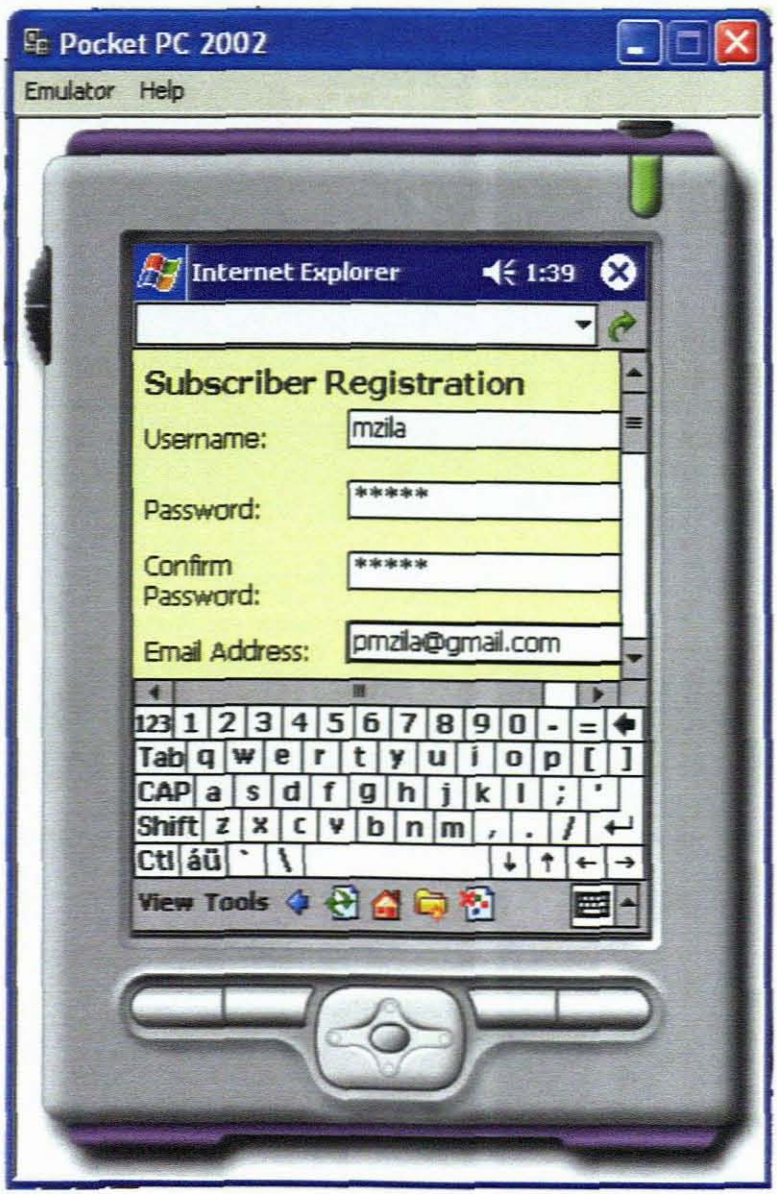


Figure A 4

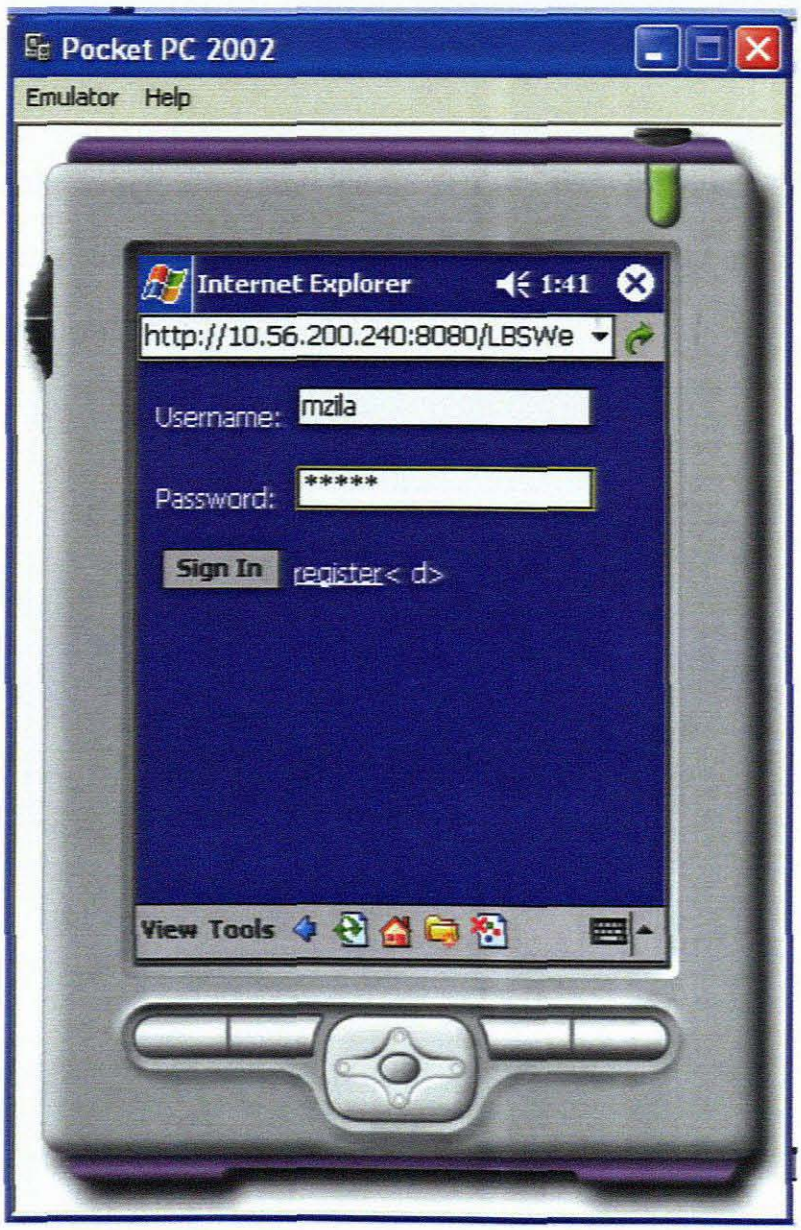

Pocket PC 2002
Emulator Help
Internet Explorer 1:41
http://10.56.200.240:8080/LBSWe
Username: mzila
Password: *****
Sign In register< d>
View Tools

Figure A 5

Service Information

After successful registration and log in, you are then provided with the page that shows the information about all available services in your location as depicted by Figure A6, A7, A8 and A9.

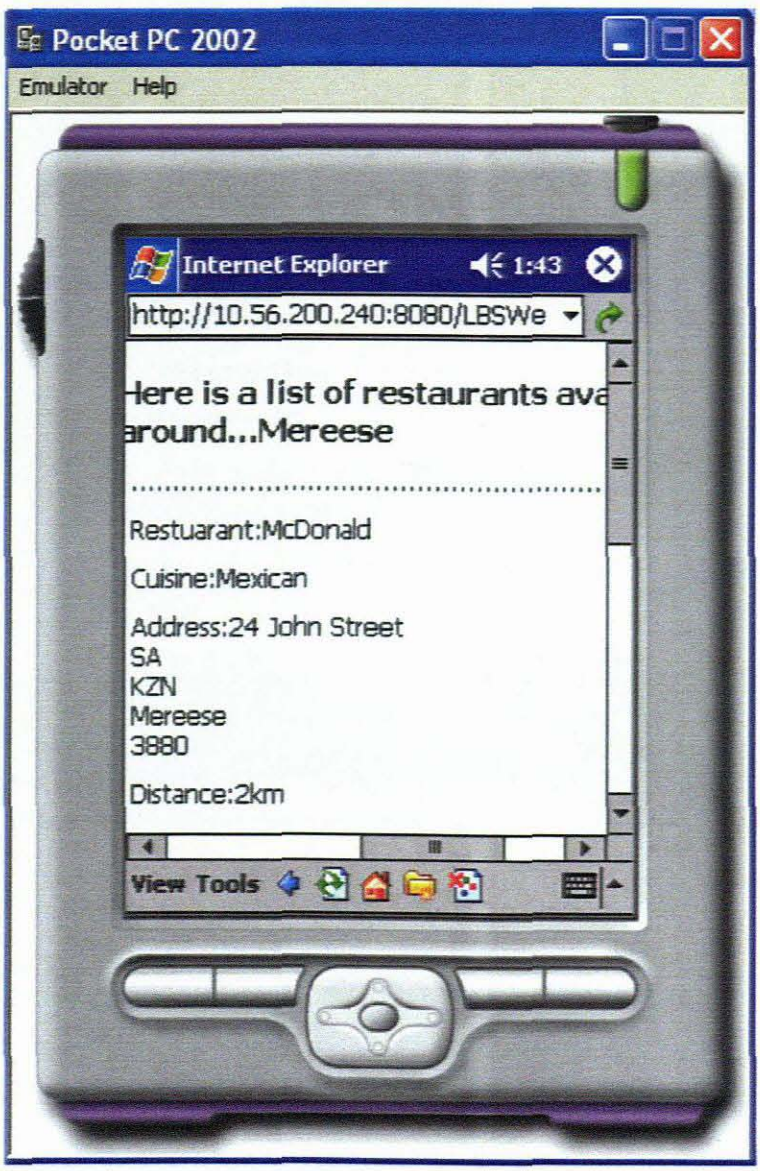


Figure A 6

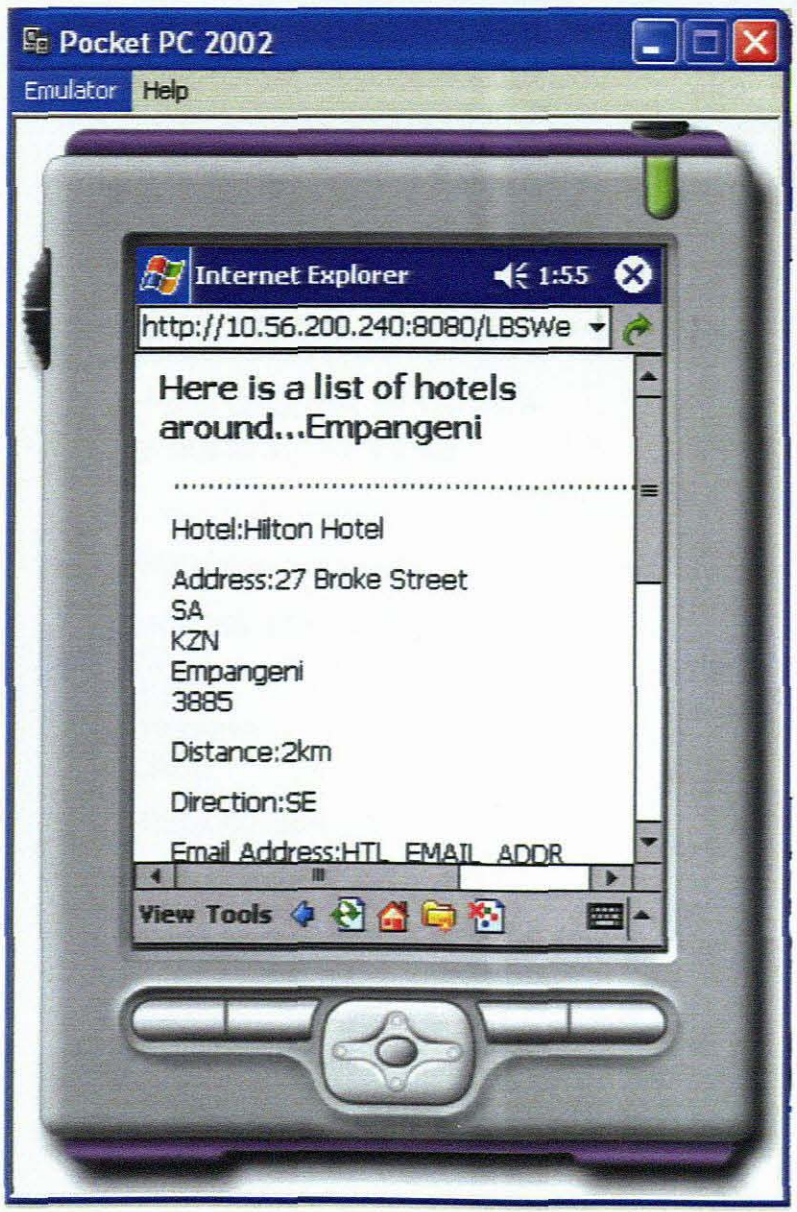

Pocket PC 2002
Emulator
Help
Internet Explorer
1:55
http://10.56.200.240:8080/LBSWe
Here is a list of hotels around...Empangeni
Hotel:Hilton Hotel
Address:27 Broke Street
SA
KZN
Empangeni
3885
Distance:2km
Direction:SE
Email Address:HTL_EMAIL_ADDR
View Tools

Figure A 7


Pocket PC 2002
Emulator Help
Internet Explorer 1:45
http://10.56.200.240:8080/LBSWe
Here is a list of hotels around...Mereese
Hotel:Formula1 Hotel
Address:15 K Street
SA
KZN
Mereese
3880
Distance:2km
Direction:SE
Email Address:HTL_EMAIL_ADDR
View Tools

Figure A 8

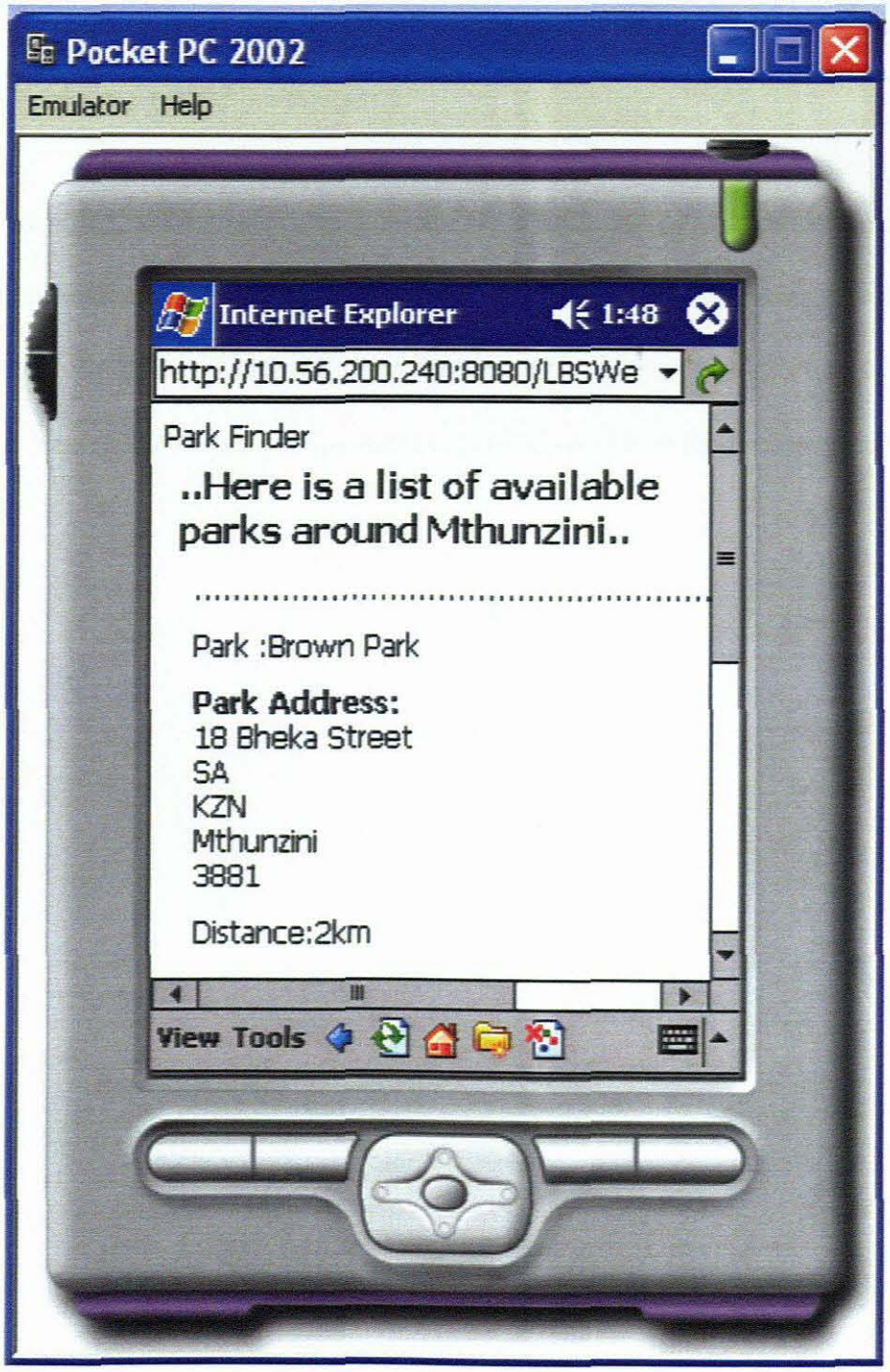

Pocket PC 2002
Emulator Help
Internet Explorer
1:48
http://10.56.200.240:8080/LBSWe
Park Finder
..Here is a list of available parks around Mthunzini..
Park :Brown Park
Park Address:
18 Bheka Street
SA
KZN
Mthunzini
3881
Distance:2km
View Tools

Figure A 9

Service registration/Advertisement

If you want to advertise your service, you have to visit the Location based Service Website that will allow you to register your services. Figure A10 shows the home page of LBAS. And Figure A11 shows the table that a service provider has to fill in for his service according to the defined locations.


C:\Documents and Settings\MTN\Desktop\Mzila2\index.html - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address C:\Documents and Settings\MTN\Desktop\Mzila2\index.html
Go Links
Welcome to The Location Based Service website...
To register your service, please click the service of your choice below:
Home
Hotels
Parks
Restaurants
We offer you an outstanding opportunity to reach your customers. All you need to do is to register your service and we shall advertise it to all potential mobile users according to their locations. We accept all services that are location based sensitive. You can now be alert about all available service next to your location, meaning where-ever you go we are there. These services ranges from Resaturant Finder Service, Hotel Finder Service, Mall Finder Service, Gas Filling Station Finder Service, Taxi Finder Service and Park Finder Service. Now with our system you will never be in trouble if you arrive in a strange place, with us you are not a stranger. Mobile users have the opportunity to register as our customers through their mobile device once they are located in their current locations. Our Slogan : WHERE-EVER YOU GO WE ARE THERE!
Done
My Computer
start
3:16 PM

Figure A 10

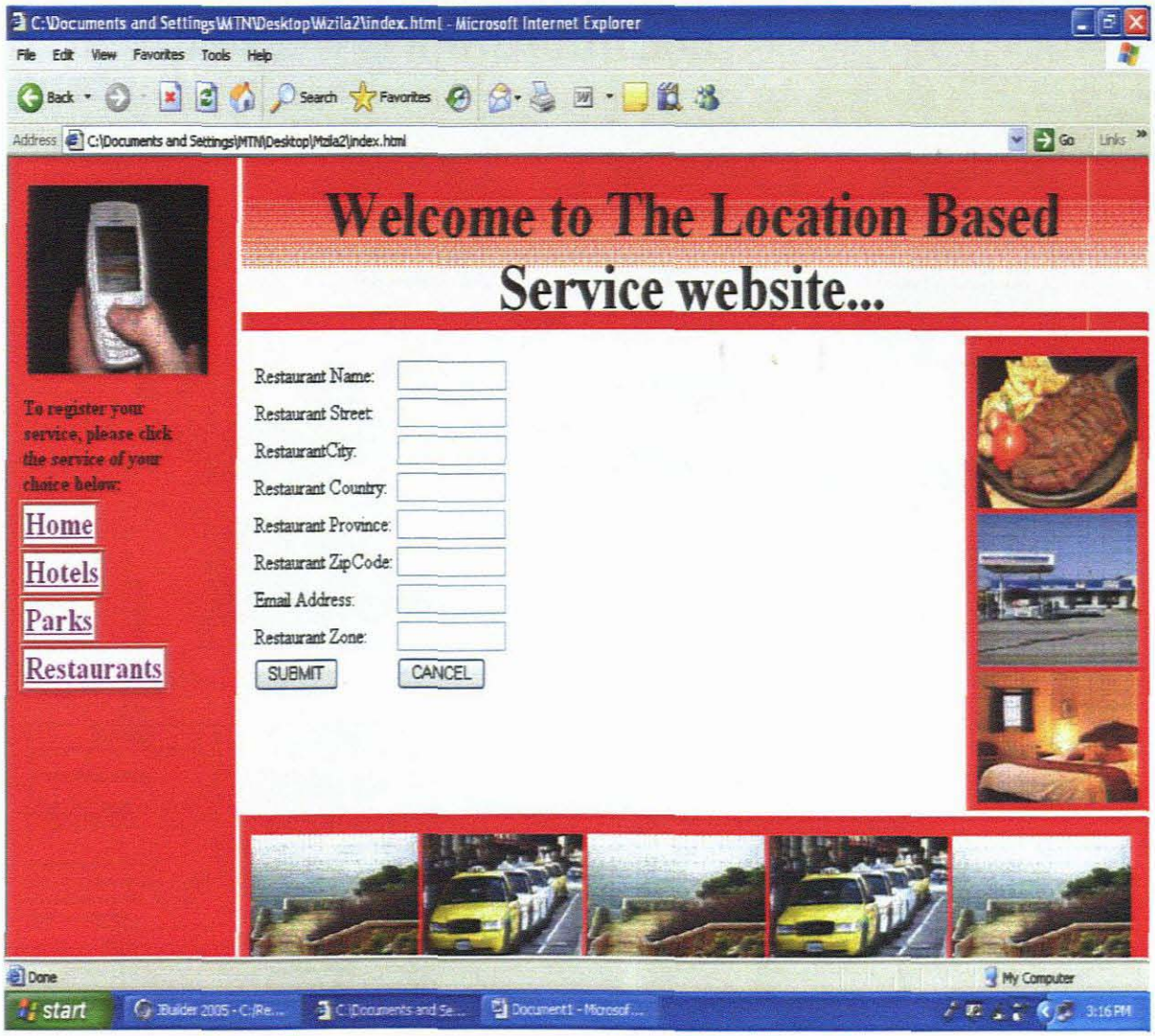

C:\Documents and Settings\MTN\Desktop\Mzila2\index.html - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address C:\Documents and Settings\MTN\Desktop\Mzila2\index.html
Go Links
Welcome to The Location Based
Service website...
To register your service, please click the service of your choice below:
Home
Hotels
Parks
Restaurants
Restaurant Name:
Restaurant Street:
RestaurantCity:
Restaurant Country:
Restaurant Province:
Restaurant ZipCode:
Email Address:
Restaurant Zone:
SUBMIT
CANCEL
Done
My Computer
start
3:16 PM

Figure A 11

B. Class Diagram Definition

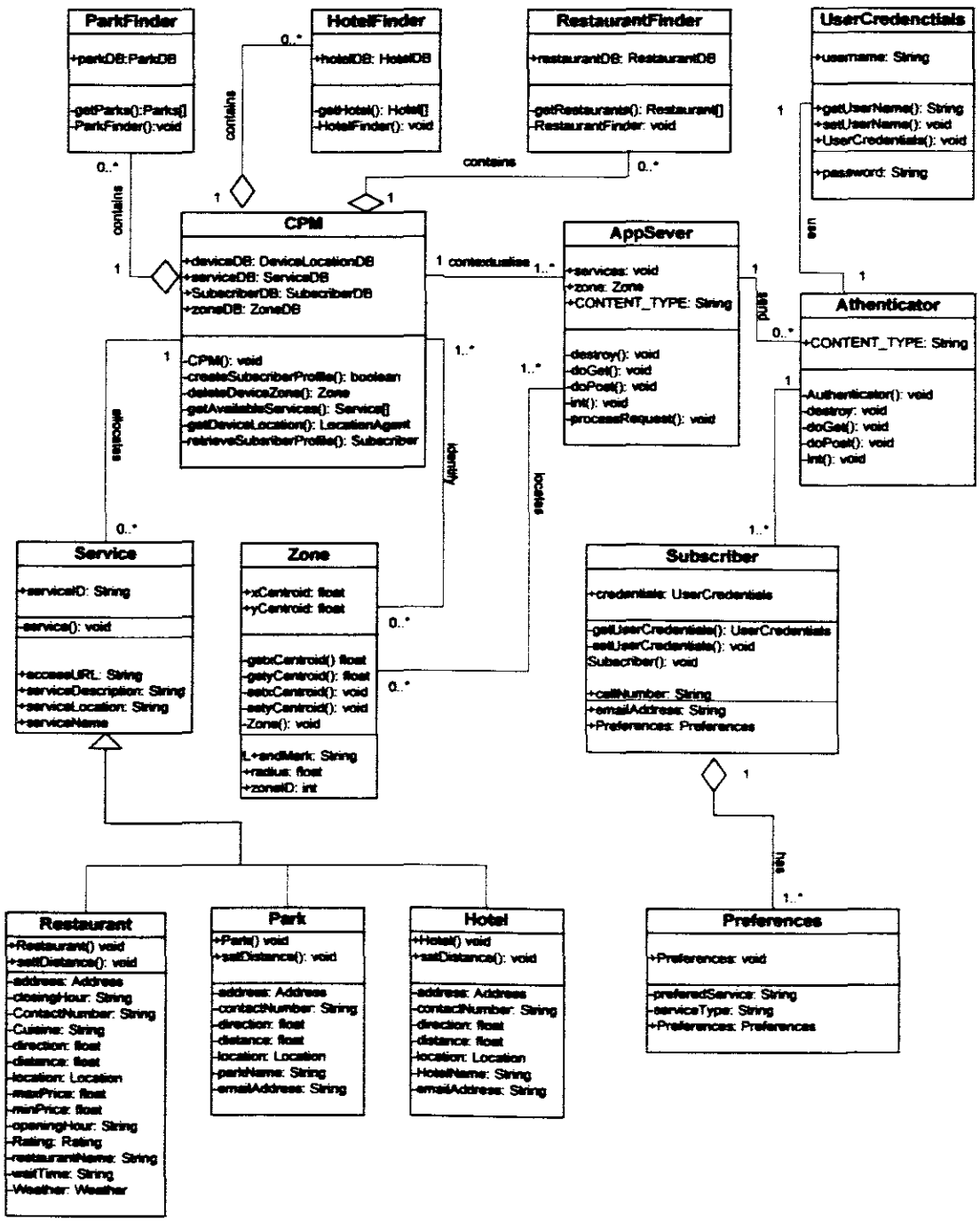


Figure B 1

ParkFinder: This class represents the system that finds all the parks registered in service directory that are defined by park service suppliers and distributed according to locations in which they are available.

RestaurantFinder: This class represents the system that finds all the restaurants registered in service directory that are defined by restaurants service suppliers and distributed according to locations in which they are available.

HotelFinder: This class represents the system that finds all the hotels registered in service directory that are defined by hotel service suppliers and distributed according to locations in which they are available.

Subscriber: This is the class that represents mobile users, users that subscribe to the system after they have discovered services. It contains user information.

UserCredentials: This class contains information entered by the subscriber during subscription, which later is needed by the system to authenticate the subscriber.

Authentication: This class represents the process by which the system familiarize itself with user credentials to identify if the user can access the service or not.

CPM: This class is context aware, it is responsible for allocating service in locations where they are supposed to be discovered and it associates the user profile and preferences with what should be delivered to the concern user and the location or zone. It is called Context and Profile Manager.

AppServer: This class is responsible for running all classes within the server.

Preferences: This class contains user preferences

Hotel: This class contains information for each hotel

Restaurant: This class contains information for each restaurant

Park: This class contains information for each park

Zone: This class defines all locations in which services are available and advertised by the system to mobile users.

Services: This class contains all location based services that need to be pushed to users for discovery purposes.

C. Usability Testing Questionnaires

The usability testing was conducted by:
Mr. P.D Mzila (20 000847)
Masters Student (Researcher)
Department of Computer Science
University of Zululand
Date: ___/___/___

Purpose: To evaluate the necessity of the system- Does it solve the identified problem?
To test out the user-friendliness and Learnability to use of the system

Tools used for conducting test:

a) LBAS System
b) Questionnaire
c) Students from the University of Zululand
(Novices, Normal and Expects)

Novices: Students from Art and Education Departments
Normal: Students from Law and Commerce Department
Experts: Students from Science Department

Dear Respondent

Please read and answer by tick the following questions based on your experiences when you were using the system..

Questions

1. Please pick your faculty

- ☐ Science and Agriculture
- ☐ Education
- ☐ Commerce and Law
- ☐ Art

2. How often have you been using location based service application?

- ☐ Regularly
- ☐ Moderate
- ☐ Never

3. How do you rate the access of discovered services?

- ☐ Easy to access
- ☐ Considerably easy
- ☐ Moderate
- ☐ Relatively not easy
- ☐ Not easy

4. After using this System (LBAS), do you think there is necessity for location based service discovery?

- ☐ Strongly agree
- ☐ Agree
- ☐ Strongly disagree
- ☐ Disagree
- ☐ Not sure

5. How do you rate the quality of interaction between user and the system?

- ☐ Extremely friendly
- ☐ Moderate friendly
- ☐ Complicated
- ☐ Extremely complicated
- ☐ Not sure

6. Do you think it is easy to learn and use this system?

- ☐ Yes
- ☐ No

7. After using the system were you satisfied about services offered?

- ☐ Yes
- ☐ No

Thank you for your time in helping to improve our system. The knowledge gain from your response is highly appreciated.

D. Source Code

Access layer

```
package com.mzila.lbs.accesslogic;

import java.sql.*;

public class  DBConnection
{
      private final String DRIVER_NAME =
"sun.jdbc.odbc.JdbcOdbcDriver";
      private String dbURL;
      private Connection connection;


      public DBConnection(String dbURL)
      {
            this.dbURL = dbURL;
            try
            {
                  Class.forName(DRIVER_NAME);
                  System.out.println("Driver loaded");
                  connection  =
DriverManager.getConnection(this.dbURL);
                  System.out.println("Connetion to " + this.dbURL
+"successful");
            }
            catch(ClassNotFoundException cnfex)
            {
                  System.out.println("Failed to load driver:" +
DRIVER_NAME + ":" + cnfex.getMessage());
                  System.exit(1);
            }
            catch(SQLException sqlex)
            {
                  System.out.println("Failed to connect to  " +
this.dbURL + " database:" + sqlex.getMessage());
            }
      }
      public Connection getConnection()
      {
            return connection;
      }
}
package com.mzila.lbs.accesslogic;
import java.sql.*;
import com.mzila.lbs.businesslogic.*;

public class DeviceLocationDB
{
DBConnection dbConnection = new DBConnection("jdbc:odbc:LBSDB");
Connection connection;
```

```
public DeviceLocationDB()
        {
                connection = dbConnection.getConnection();
        }
        public  LocationAgent getDeviceLocationInfo(int deviceID)
        {
                LocationAgent locAgent = null;
                try
                {
String locationQuery = "SELECT * FROM DEVICE_LOCATIONS WHERE DEVICE_ID
= ?";
PreparedStatement locationStmt =
connection.prepareStatement(locationQuery);

locationStmt.setInt(1,deviceID);

ResultSet rs = locationStmt.executeQuery();

while(rs.next())
                        {
locAgent = new LocationAgent();

Location location = new Location();


locAgent.setDeviceType(rs.getString("DEVICE_TYPE"));

location.setLongitude(rs.getFloat("DEVICE_XCOORD"));

location.setLatitude(rs.getFloat("DEVICE_YCOORD"));
locAgent.setLocation(location);

                        }
                }
 catch(SQLException sqlex)
                {
 sqlex.printStackTrace();
                }
 return locAgent;
        }

}




package com.mzila.lbs.accesslogic;

import java.sql.*;
import java.util.*;
import com.mzila.lbs.businesslogic.*;

public class ServicesDB
{
```

```
DBConnection db = new DBConnection("jdbc:odbc:LBSDB");
Connection connection;
        public ServicesDB()
        {
                connection = db.getConnection();
        }
        public Service[] getAvailableServices(int zoneID)
        {
                Service [] services =  null;
                ArrayList serviceList  = new ArrayList();
                Service service = null;
                try
                {
String serviceQuery = "SELECT * FROM SERVICES WHERE ZONE_ID = ?";
PreparedStatement serviceStmt =
connection.prepareStatement(serviceQuery);

                        serviceStmt.setInt(1,zoneID);

                        ResultSet rs =  serviceStmt.executeQuery();

                        while(rs.next())
                        {
                             service = new Service();

service.setServiceName(rs.getString("SERVICE_NAME"));

service.setAccessURL(rs.getString("ACCESS_URI"));

service.setServiceDescription(rs.getString("SERVICE_DESC"));

                             serviceList.add(service);
                        }

                        serviceList.trimToSize();

                        services = new Service[serviceList.size()];
                        for(int i = 0; i < services.length;i++)
                        {
                                services[i] =
(Service)serviceList.get(i);
                        }
                }
                catch(SQLException sqlex)
                {
                     System.out.println("Failed to retrieve available
services"+sqlex.getMessage());
                }
                return services;
        }
}



package com.mzila.lbs.accesslogic;
```

```
import com.mzila.lbs.businesslogic.*;

import javax.xml.parsers.*;
import javax.xml.transform.*;
import javax.xml.transform.dom.*;
import javax.xml.transform.stream.*;
import org.w3c.dom.*;
import java.io.*;


public class SubscriberDB
{

        DocumentBuilderFactory  parserFactory;
        DocumentBuilder parser;
        Document xmlDoc;
        TransformerFactory transFactory;
        Transformer trans;
        Source source;
        Result result;
        String fileName =
"C:\\ResturantFinderService\\LocationBasedServicesFinderProject\\Data
Source\\subscribers.xml";
        public SubscriberDB()
      {
                  try
                        {
                                    parserFactory =
DocumentBuilderFactory.newInstance();
                                    parser =
parserFactory.newDocumentBuilder();
                                    transFactory =
TransformerFactory.newInstance();
                                    trans = transFactory.newTransformer();
                         }
                         catch(Exception ex)
                         {
                                    ex.printStackTrace();
                         }
      }
        public boolean authenticateSubscriber(Subscriber subscriber)
        {

               boolean isValid = false;
               UserCredentials cr = null;
               try
               {
                        xmlDoc = parser.parse(new File(fileName));

                        Element root = xmlDoc.getDocumentElement();

NodeList subscriberList =  root.getElementsByTagName("subscriber");

                        for(int i = 0; i <
subscriberList.getLength();i++)
                        {
```

```
Element subscriberElement = (Element)subscriberList.item(i);

Element username =
(Element)subscriberElement.getElementsByTagName("username").item(0);
Text tUserName = (Text)username.getFirstChild();
                                String sUserName =
subscriber.getUserCredentials().getUserName();

Element password =
(Element)subscriberElement.getElementsByTagName("password").item(0);
Text  tPassword  = (Text)password.getFirstChild();
String sPassword = subscriber.getUserCredentials().getPassword();


if((tUserName.getNodeValue().equals(sUserName)) &&
(tPassword.getNodeValue().equals(sPassword)))
                                {
                        cr =  new UserCredentials();

cr.setUserName(tUserName.getNodeValue());

cr.setPassword(tPassword.getNodeValue());
                        }

                        }
                        if(cr != null )
                                isValid = true;
                        else
                                isValid = false;
                }
                catch(Exception ex)
                {
                        ex.printStackTrace();
                }
                return isValid;
        }
      public boolean  createProfile(Subscriber sub)
      {
                boolean isCreated =  false;
            try
            {


                  xmlDoc = parser.parse(new File(fileName));


 Element subscribers = xmlDoc.getDocumentElement();

Element  subscriber = xmlDoc.createElement("subscriber");
                        subscribers.appendChild(subscriber);

                        //Adding  a username
                        Element  username =
xmlDoc.createElement("username");

username.appendChild(xmlDoc.createTextNode(sub.getUserCredentials().get
UserName()));
```

```
                        subscriber.appendChild(username);

                        //Adding a password
Element  password = xmlDoc.createElement("password");

password.appendChild(xmlDoc.createTextNode(sub.getUserCredentials().get
Password()));
                        subscriber.appendChild(password);

                        //Adding an email address
Element emailAddress =  xmlDoc.createElement("emailAddress");

emailAddress.appendChild(xmlDoc.createTextNode(sub.getEmailAddress()));
                        subscriber.appendChild(emailAddress);

                        //Adding a cellphone number
Element cell = xmlDoc.createElement("cellNumber");

cell.appendChild(xmlDoc.createTextNode(sub.getCellNumber()));
                        subscriber.appendChild(cell);

                        //Adding preferences
Element preference = xmlDoc.createElement("preferences");
                        subscriber.appendChild(preference);

String [] preferedServices =
sub.getPreferences().getPreferedServices();

                    for(int i = 0;i < preferedServices.length;i++)
                    {
Element prefService =  xmlDoc.createElement("preferedService");

      prefService.appendChild(xmlDoc.createTextNode(preferedServices[i]
));
                                    preference.appendChild(prefService);

                    }

String [] cuisines = sub.getPreferences().getCuisine();
                        for(int i = 0;i < cuisines.length;i++)
                    {
Element cuisine = xmlDoc.createElement("cuisine");

      cuisine.appendChild(xmlDoc.createTextNode(cuisines[i]));
                                    preference.appendChild(cuisine);
                    }
String [] foodTypes = sub.getPreferences().getFoodType();
                    for(int i = 0;i < foodTypes.length;i++)
                    {
Element foodtype = xmlDoc.createElement("foodType");

      foodtype.appendChild(xmlDoc.createTextNode(foodTypes[i]));
                                    preference.appendChild(foodtype);
                    }

Element priceLow = xmlDoc.createElement("priceLow");
```

```
      priceLow.appendChild(xmlDoc.createTextNode(String.valueOf(sub.get
Preferences().getPriceLow())));
                        preference.appendChild(priceLow);

Element priceHigh = xmlDoc.createElement("priceHigh");

      priceHigh.appendChild(xmlDoc.createTextNode(String.valueOf(sub.ge
tPreferences().getPriceHigh())));
                        preference.appendChild(priceHigh);

                  Source source = new DOMSource(xmlDoc);
                  Result result = new StreamResult(new File(fileName));

                  trans.transform(source,result);
                      if(sub.getUserCredentials().getUserName() != " ")
                                isCreated  = true;
            }
            catch(Exception ex)
            {
                        ex.printStackTrace();
            }
                return isCreated;
      }
      public Subscriber retrieveProfile(Subscriber credentials)
      {
            Subscriber subscriber = null;
            try
            {
                  xmlDoc = parser.parse(new File(fileName));

NodeList subscriberList = xmlDoc.getElementsByTagName("subscribers");
                  for(int i = 0;i < subscriberList.getLength();i++)
                  {
Element subscriberElement = (Element)subscriberList.item(i);

Element username =
(Element)subscriberElement.getElementsByTagName("username").item(0);
                        Text t = (Text)username.getFirstChild();


if(credentials.getUserCredentials().getUserName().equals(t.getNodeValue
()))
                        {
                              subscriber = new Subscriber();
                              Preferences pref = new Preferences();

Element email =
(Element)subscriberElement.getElementsByTagName("emailAddress").item(0)
;
Text txtEmail  = (Text)email.getFirstChild();

      subscriber.setEmailAddress(txtEmail.getNodeValue());

Element cell  =
(Element)subscriberElement.getElementsByTagName("cellNumber").item(0);
Text txtCell = (Text)cell.getFirstChild();
```

```
subscriber.setCellNumber(txtCell.getNodeValue());

Element preferenceElement  =
(Element)subscriberElement.getElementsByTagName("preferences").item(0);

                                        //Querying services
NodeList prefServ  =
preferenceElement.getElementsByTagName("preferedService");
String []  services = new String[prefServ.getLength()];
                                        for(int x = 0;x <
prefServ.getLength();x++)
                                {
                                        Element servicePref =
(Element)prefServ.item(x);
Text txtSP = (Text)servicePref.getFirstChild();
                                        services[x] = txtSP.getNodeValue();
                                }

pref.setPreferedServices(services);

                                        //Querying cuisines
NodeList  cuisinesList =
preferenceElement.getElementsByTagName("cuisine");
String [] cuisines = new String[cuisinesList.getLength()];
                                        for(int y = 0;y <
cuisinesList.getLength();y++)
                                {
Element cuisine = (Element)cuisinesList.item(y);
Text txtC = (Text)cuisine.getFirstChild();
cuisines[y] = txtC.getNodeValue();
                                }
        pref.setCuisine(cuisines);

                                        //Querying food types
NodeList  foodTypeList =
preferenceElement.getElementsByTagName("foodType");
String [] foodTypes = new String[foodTypeList.getLength()];
                                for(int z = 0;z <
foodTypeList.getLength();z++)
                                {
Element foodType = (Element)foodTypeList.item(z);
Text txtFood = (Text)foodType.getFirstChild();
                                foodTypes[z] = txtFood.getNodeValue();
                                }
                                        pref.setFoodType(foodTypes);

                                        //Querying price low
Element priceLow =
(Element)preferenceElement.getElementsByTagName("priceLow").item(0);
Text txtPLow = (Text)priceLow.getFirstChild();

pref.setPriceLow(Float.parseFloat(txtPLow.getNodeValue()));

                                        //Querying price high
Element priceHigh =
(Element)preferenceElement.getElementsByTagName("priceHigh").item(0);
```

```
Text txtPHigh = (Text)priceHigh.getFirstChild();

pref.setPriceHigh(Float.parseFloat(txtPHigh.getNodeValue()));


subscriber.setPreferences(pref);
                        }

                    }
            }
            catch(Exception ex)
            {
                  ex.printStackTrace();
            }
                return subscriber;
        }

}
```

Business layer

```
package com.mzila.lbs.businesslogic;

public class DynamicContext implements java.io.Serializable
{
      private Location location;
      private Time timeStamp;
      private float speed;
      private float direction;
      private Weather weather;
      private int distance;

      public DynamicContext()
      {
      }
      public void setLocation(Location geocodes)
      {
            this.location = geocodes;
      }
      public void setTimeStamp(Time timeStamp)
      {
            this.timeStamp = timeStamp;
      }
      public void setSpeed(float speed)
      {
            this.speed = speed;
      }
      public void setDirection(float direction)
```

```
        {
                this.direction = direction;
        }
        public Location getLocation()
        {
                return location;
        }
        public Time getTimeStamp()
        {
                return timeStamp;
        }
        public float getSpeed()
        {
                return speed;
        }
        public float getDirection()
        {
                return direction;
        }
}

package com.mzila.lbs.businesslogic;

public class Location implements java.io.Serializable
{
        private float longitude;
        private float latitude;
        public Location()
        {

        }
        public void setLongitude(float longitude)
        {
                this.longitude = longitude;
        }
        public void setLatitude(float latitude)
        {
                this.latitude = latitude;
        }
        public float getLongitude()
        {
                return longitude;
        }
        public float getLatitude()
        {
                return latitude;
        }

}


package com.mzila.lbs.businesslogic;

public class LocationAgent implements  java.io.Serializable
{
        private int deviceID;
        private String deviceType;
```

```
        private Location location;


        public LocationAgent()
        {

        }
        public void setDeviceID(int deviceID)
        {
                this.deviceID = deviceID;
        }
        public int getDeviceID()
        {
                return deviceID;
        }
        public void setDeviceType(String deviceType)
        {
                this.deviceType = deviceType;
        }
        public String getDeviceType()
        {
                return deviceType;
        }
        public void setLocation(Location deviceLocation)
        {
                this.location = deviceLocation;
        }
        public Location getLocation()
        {
                return location;
        }

}



package com.mzila.lbs.businesslogic;

public class Preferences implements java.io.Serializable
{
      private String [] cuisine;
      private String [] foodType;
      private float priceHigh;
      private float priceLow;
      private int cuisinePreferenceLevel;
      private String [] preferedServices;


      public Preferences()
      {
      }
      public void setCuisine(String [] cuisine)
      {
            this.cuisine = cuisine;
      }
```

```
	public void setFoodType(String [] foodType)
	{
		this.foodType = foodType;
	}
	public void setPriceHigh(float priceHigh)
	{
		this.priceHigh = priceHigh;
	}
	public void setPriceLow(float priceLow)
	{
		this.priceLow = priceLow;
	}
	public void setCuisinePreferenceLevel(int cuisinePreferenceLevel)
	{
		this.cuisinePreferenceLevel = cuisinePreferenceLevel;
	}
	public void setPreferedServices(String [] services)
	{
		this.preferedServices = services;
	}
	public String [] getPreferedServices()
	{
		return preferedServices;
	}
	public String [] getCuisine()
	{
		return cuisine;
	}
	public String[] getFoodType()
	{
		return foodType;
	}
	public float getPriceHigh()
	{
		return priceHigh;
	}
	public float getPriceLow()
	{
		return priceLow;
	}
	public int getCuisinePreferenceLevel()
	{
		return cuisinePreferenceLevel;
	}
}



package com.mzila.lbs.businesslogic;

public class Service implements java.io.Serializable
{
	private String serviceID;
	private String serviceName;
```

```
        private String accessURL;
        private String serviceLocation;
        private String serviceDescription;

        public Service()
        {
        }
        public void setServiceName(String serviceName)
        {
                this.serviceName = serviceName;
        }
        public void setAccessURL(String accessURL)
        {
                this.accessURL = accessURL;
        }
        public void setServiceLocation(String serviceLocation)
        {
                this.serviceLocation = serviceLocation;
        }
        public void setServiceDescription(String serviceDescription)
        {
                this.serviceDescription = serviceDescription;
        }
        public String getServiceName()
        {
                return serviceName;
        }
        public String getAccessURL()
        {
                return accessURL;
        }
        public String getServiceLocation()
        {
                return serviceLocation;
        }
        public String getServiceDescription()
        {
                return serviceDescription;
        }
}
```

```
package com.mzila.lbs.businesslogic;


public class Subscriber implements java.io.Serializable
{
      private UserCredentials credentials;
      private String emailAddress;
```

```
        private String cellNumber;
        private Preferences  preferences;

        public Subscriber()
        {
        }
        public void setUserCredentials(UserCredentials credentials)
          {
              this.credentials = credentials;
        }
        public void setEmailAddress(String emailAddress)
        {
              this.emailAddress = emailAddress;
        }
        public void setCellNumber(String cellNumber)
        {
              this.cellNumber = cellNumber;
        }
        public void setPreferences(Preferences  preferences)
        {
              this.preferences = preferences;
        }
        public UserCredentials getUserCredentials()
        {
              return credentials;
        }
        public String getEmailAddress()
        {
              return emailAddress;
        }
        public String getCellNumber()
        {
              return cellNumber;
        }
        public Preferences getPreferences()
        {
              return preferences;
        }
}
```

```
package com.mzila.lbs.businesslogic;

public class UserCredentials implements java.io.Serializable
{
        private String username;
        private String password;
        public UserCredentials()
        {
              setUserName(null);
              setPassword(null);
```

```
	}
	public UserCredentials(String username,String password)
	{
		this.username = username;
		this.password = password;
	}
	public void setUserName(String username)
	{
		this.username = username;
	}
	public void setPassword(String password)
	{
		this.password = password;
	}
	public String getUserName()
	{
		return username;
	}
	public String getPassword()
	{
		return password;
	}


}


package com.mzila.lbs.accesslogic;

import com.mzila.lbs.utils.*;
import com.mzila.lbs.businesslogic.*;
import java.util.*;
import java.sql.*;

public class ZoneDB
{
	DBConnection dbConnection = new
DBConnection("jdbc:odbc:LBSDB");
	Connection connection;

	public ZoneDB()
	{
		connection = dbConnection.getConnection();
	}
	public Zone getDeviceZoneArea(float deviceXCoord,float
deviceYCoord)
	{
		Zone reqZone = null;
		ArrayList zoneList = new ArrayList();
		MyUtils utils = new MyUtils();
		try
		{
			String zoneQ = "SELECT * FROM ZONE";
			Statement stmt = connection.createStatement();

			ResultSet rs = stmt.executeQuery(zoneQ);
			Zone zone = null;
```

```
                        while(rs.next())
                        {
                                zone = new Zone();
                                zone.setZoneID(rs.getInt("ZONE_ID"));

zone.setRadius(rs.getFloat("ZONE_RADIUS"));

zone.setxCentroid(rs.getFloat("ZONE_XCOORD"));

zone.setyCentroid(rs.getFloat("ZONE_YCOORD"));

zone.setLandmark(rs.getString("ZONE_LANDMARK"));
                                zoneList.add(zone);
                        }
                        zoneList.trimToSize();

                        for(int i = 0; i < zoneList.size();i++)
                        {
                                zone = (Zone)zoneList.get(i);

if(utils.calculateAvDistance(zone.getxCentroid(),zone.getyCentroid(),de
viceXCoord,deviceYCoord) < zone.getRadius())
                                {
                                        reqZone = zone;
                                }
                        }


                }
                catch(SQLException sqlex)
                {
                        sqlex.printStackTrace();
                }
                return reqZone;
        }
}
```

```
package com.mzila.lbs.businesslogic;

public class Hotel
{
        private String hotelName;
        private Address address;
        private String location;
        private Weather weather;
        private Rating hotelRating;
        private String homePageURL;
```

```
private String contactNumber;
private String emailAddress;

public Hotel()
{
}
public String getHotelName()
{
        return hotelName;
}
public void setHotelName(String hotelName)
{
        this.hotelName = hotelName;
}
public Address getAddress()
{
        return address;
}
public void setAddress(Address address)
{
        this.address = address;
}
public String getLocation()
{
        return location;
}
public void setLocation(String location)
{
        this.location = location;
}
public Weather getWeather()
{
        return weather;
}
public void setWeather(Weather weather)
{
        this.weather = weather;
}
public Rating getHotelRating()
{
        return hotelRating;
}
public void setHotelRating(Rating hotelRating)
{
        this.hotelRating = hotelRating;
}
public String getHomePageURL()
{
        return homePageURL;
}
public void setHomePageURL(String homePageURL)
{
        this.homePageURL = homePageURL;
}
public String getContactNumber()
{
        return contactNumber;
```

```
        }
        public void setContactNumber(String contactNumber)
        {
                this.contactNumber = contactNumber;
        }
        public String getEmailAddress()
        {
                return emailAddress;
        }
        public void setEmailAddress(String emailAddress)
        {
                this.emailAddress = emailAddress;
        }
}



package com.mzila.lbs.businesslogic;

public class Park implements java.io.Serializable
{
        private String parkName;
        private Address parkAddress;

      public Park()
      {
      }
        public void setParkName(String parkName)
        {
                this.parkName = parkName;
        }
        public void setParkAddress(Address address)
        {
                this.parkAddress = address;
        }
        public String getParkName()
        {
                return parkName;
        }
        public Address getParkAddress()
        {
                return parkAddress;
        }
}

package com.mzila.lbs.businesslogic;

public class Restuarant implements java.io.Serializable
{
      private String restuarantName;
      private String cuisine;
      private Address address;
        private String locationName;
        private Location location;
      private String contactNumber;
```

```
private Rating rating;
private float minPrice;
private float maxPrice;
private String openingHour;
private String closingHour;
private float distance;
private float direction;
private String waitTime;
private Weather weather;
private int selectionRate;

public Restuarant()
{
}
public void setRestuarantName(String restuarantName)
{
      this.restuarantName = restuarantName;
}
public void setCuisine(String cuisine)
{
      this.cuisine = cuisine;
}
public void setAddress(Address address)
{
      this.address = address;
}
public void setLocationName(String location)
{
      this.locationName = location;
}
public void setContactNumber(String contactNumber)
{
      this.contactNumber = contactNumber;
}
public void setRating(Rating rating)
{
      this.rating = rating;
}
public void setMinPrice(float minPrice)
{
      this.minPrice = minPrice;
}

public void setMaxPrice(float maxPrice)
{
      this.maxPrice = maxPrice;
}
public void setOpeningHour(String openingHour)
{
      this.openingHour = openingHour;
}
public void setClosingHour(String closingHour)
{
      this.closingHour = closingHour;
}
public void setDistance(float distance)
{
```

```
        this.distance = distance;
}
public void setDirection(float direction)
{
        this.direction = direction;
}
public void setWaitTime(String waitTime)
{
        this.waitTime = waitTime;
}
public void setWeather(Weather weather)
{
        this.weather = weather;
}
public void setSelectionRate(int selectionRate)
{
        this.selectionRate = selectionRate;
}


public String getRestuarantName()
{
        return restuarantName;
}
public String getCuisine()
{
        return cuisine;
}
public Address getAddress()
{
        return address;
}
public String getLocationName()
{
        return locationName;
}
public String getContactNumber()
{
        return contactNumber;
}
public Rating getRating()
{
        return rating;
}
public float getMinPrice()
{
        return minPrice;
}
public float getMaxPrice()
{
        return maxPrice;
}
public String getOpeningHour()
{
        return openingHour;
}
public String getClosingHour()
```

```
        {
                return closingHour;
        }
        public float setDistance()
        {
                return distance;
        }
        public float getDirection()
        {
                return direction;
        }
        public String getWaitTime()
        {
                return waitTime;
        }
        public Weather getWeather()
        {
                return weather;
        }
        public int getSelectionRate()
        {
                return selectionRate;
        }

}
```

Web services

```
package com.mzila.lbs.webservices;

import com.mzila.lbs.businesslogic.*;
import com.mzila.lbs.accesslogic.*;


public class CPM
{
        ZoneDB zoneDB = new ZoneDB();
        DeviceLocationDB devlocDB = new DeviceLocationDB();
        SubscriberDB subscriberDB = new SubscriberDB();
        ServicesDB  serviceDB = new ServicesDB();
        public CPM()
        {
            try
            {
                  jbInit();
            }
            catch (Exception ex)
            {
                  ex.printStackTrace();
            }
```

```
	}

	  public LocationAgent getDeviceLocation()
	  {
		int deviceID = 200 + ((int)(Math.random()*3));
		return devlocDB.getDeviceLocationInfo(deviceID);
	  }
	  public Zone detectDeviceZone(float deviceXCoord,float
deviceYCoord)
	  {
		return
zoneDB.getDeviceZoneArea(deviceXCoord,deviceYCoord);
	  }
	  //
	  public Subscriber retrieveSubscriberProfile(Subscriber
credentials)
	  {
		return subscriberDB.retrieveProfile(credentials);
	  }
	  public boolean createSubscriberProfile(Subscriber subscriber)
	  {
		return subscriberDB.createProfile(subscriber);
	  }
	  public Service [] GetAvailableServices(int zoneID)
	  {
		return serviceDB.getAvailableServices(zoneID);
	  }

	private void jbInit() throws Exception
	{
	}


}



package com.mzila.lbs.webservices;

import com.mzila.lbs.accesslogic.HotelDB;
import com.mzila.lbs.businesslogic.Hotel;

//User Defined Packages
public class HotelFinder
{
	  HotelDB hotelDB = new HotelDB();
	public HotelFinder()
	{
	}
	  public Hotel[] getHotel(String location)
	  {
		return hotelDB.getHotels(location);
	  }
}
```

```
package com.mzila.lbs.webservices;

//User Defined Packages
import com.mzila.lbs.accesslogic.ParkDB;
import com.mzila.lbs.businesslogic.Park;

public class ParkFinder
{
        ParkDB parkDB = new ParkDB();
      public ParkFinder()
      {
      }
        public Park [] GetParks(String location)
        {
                return parkDB.getParks(location);
        }
}
```

```
package com.mzila.lbs.webservices;

//User Defined Packages
import com.mzila.lbs.accesslogic.RestuarantDB;
import com.mzila.lbs.businesslogic.Restuarant;

public class RestuarantFinder
{
        RestuarantDB restuarantDB = new RestuarantDB();

        public RestuarantFinder(){}
        public Restuarant[] GetRestuarants(String location,String
cuisine)
        {
                return restuarantDB.getRestuarants(location,cuisine);
        }
}
```