# CHAPTER 1

# INTRODUCTION

The Location-based Service (LBS) is a kind of service derived from the ability in electronic communication to identify and transmit location information. LBS which can be defined as the application of which the service and information provided are determined by the user location. In another word, users are able to receive the most suitable service which is provided by service provider according to their location at a particular time. As the result, both the user and the content provider are able to be benefited from it. LBS is not merely designed and developed for the general business purposes. In addition to satisfy commercial needs, its service can be extended to general consumer application, in which different LBS could be provided to the consumers according to their distinct location. As of lately telecommunication companies in many countries have been actively introducing new applications of information services. It is expected that LBS will play an important role in the telecommunication industry due to its ability of positioning in the mobile telecommunication. It is because LBS aimed at mobile users, and due to the complete difference with respect to the operation interface of web service of mobile devices from that of computers, any complex application service will lead to consumer resistance. As the result, it is necessary for the development of LBS to utilize the mobile devices effectively and to introduce the recommendation mechanism correctly.

## 1.1 Mobile Computing

Mobile computing is a relatively new field of research with little more than three decades of history. During its lifetime, it has expanded from being primarily technical to now also being about usability, usefulness, and user experience. This has led to the birth of the vibrant area of mobile interaction design at the intersections between, among others, mobile computing, social sciences, human-computer interaction, industrial design, and user experience design as shown in figure 1.1. Mobile computing is a significant contributor to the pervasiveness of computing resources in modern civilization. In concert with the proliferation of stationary and embedded computer technology throughout society, mobile devices such as cell phones and other handheld or wearable computing technologies have created a state of ubiquitous and pervasive computing where we are surrounded by more computational devices than people [39]. Enabling us to orchestrate these devices to fit and serve our personal and working lives is a huge challenge for technology developers, and “as a consequence of pervasive computing, interaction design is poised to become one of the main liberal arts of the twenty-first century”[30].

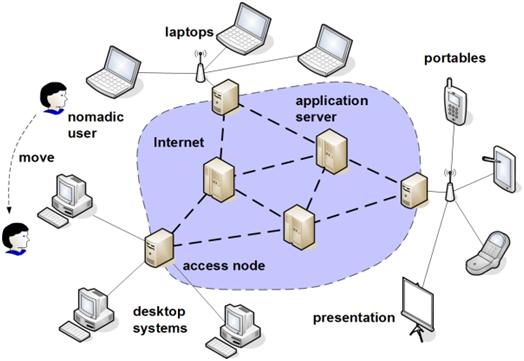


Figure 1.1 Mobile Computing Environments

The field of mobile computing has its origin in a fortunate alignment of interests by technologists and consumers. Since the dawn of the computing age, there have always been technological aspirations to make computing hardware smaller, and ever since computers became widely accessible, there has been a huge interest from consumers in being able to bring them with people [7]. As a result, the history of mobile computing is paved with countless commercially available devices. Most of them had short lifespan and minimal impact, but others significantly pushed the boundaries of engineering and interaction design.

## 1.2 Location-based Services (LBS)

Location services can be defined as services that integrate a mobile device’s location or position with other information so as to provide added value to a user. Location services are mainly used in three areas: military and government industries, emergency services and the commercial sector. The first location system in use was the satellite-based GPS as illustrated in figure 1.2, which allows for precise localization of people and objects of up to 3 meters or more of accuracy. Besides the military use of location data, emergency services have turned out to be an important application field. Analysts and researchers have taken several approaches to classifying LBS applications. A major distinction of services is whether they are person-oriented or device-oriented.

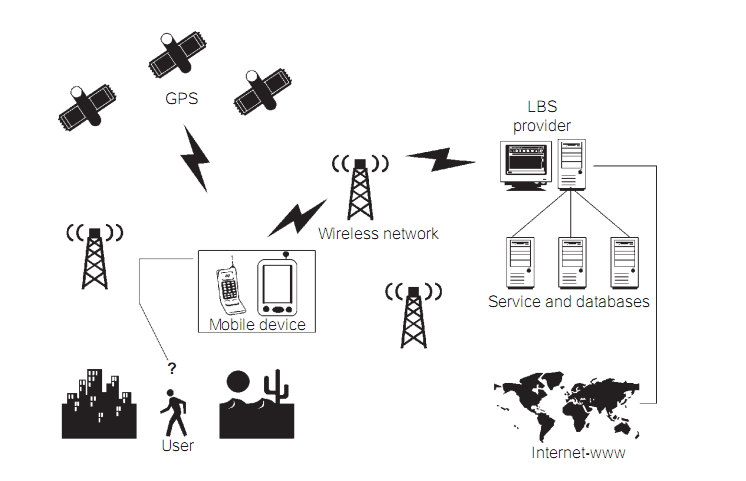


Figure 1.2 Broad Overview of LBS Architecture

Person-oriented LBS comprises all of those applications where a service is user-based. Thus, the focus of application use is to position a person or to use the position of a person to enhance a service. Usually, the person located can control the service (e.g., friend finder application).

Device-oriented LBS applications are external to the user. Thus, they may also focus on the position of a person, but they do not need to. Instead of a person, an object (e.g., a car) or a group of people (e.g., a fleet) could also be located. In device-oriented applications, the person or object located is usually not controlling the service (e.g., car tracking for theft recovery).

In addition to this first classification of services, two types of application design are being distinguished: push and pull services. [12]

Push services imply that the user receives information as a result of his or her whereabouts without having to actively request it. The information may be sent to the use with prior consent (e.g., a subscription-based terror attack alert system) or without prior consent (e.g., an advertising welcome message sent to the user upon entering a new town).

Pull services, in contrast, mean that a user actively uses an application and, in this context, “pull” information from the network. This information may be location-enhanced (e.g., where to find the nearest hotel).

Some services such as a friend finder or date finder integrate both push and pull functionality.

## 1.3 Types of LBS Applications

A reactive LBS application is triggered by the user who, based on his current location, queries the system in search of information. These applications are of the request/response type in which the user queries the system, including the current location, and the system responds with the specific information after searching in other systems and databases. There are many examples of reactive LBS applications.

1. Finding a nearby hotel, friend, service such as taxi, ATMs and the like.
2. Obtaining directions to a place from the current location.
3. Locating people nearby and display their locations on a map.
4. Obtaining local weather information.
5. Sending emergency notifications to police, insurance companies, and roadside assistance companies.

In proactive LBS applications, on the other hand, queries or actions are automatically generated by the LBIS once a predefined set of conditions are met. Since the user does not initiate the request of information, these types of applications require the LBIS to continuously know the location of the user. Conditions are included in the LBIS by the user according to his needs or application needs. In addition to traffic notifications and geofencing, there are many other examples of proactive LBS applications:

1. Fleet management.
2. Real-time tracking of people and/ or assets.
3. Location-based advertising.
4. Turn-by-turn navigation.
5. Real-time friends’ location.
6. Proximity-based actuation.
7. Travel assistance device for riding public transportation and museum-guided visits.

## 1.4 Mobile Development Platforms

With major disruptions from Apple and Google in the form of the iPhone, iPad, and Android, the mobile development landscape has forever changed from a closed API, carrier-centric model to an open API, device-manufacturer and software-developer-centric model. This major shift represents a huge opportunity for LBS application developers as well, because now it’s possible to develop fully featured LBS applications and release them to consumers directly at least most of the time.

But still, mobile development is fragmented between a numbers of different mobile operating platforms, each supporting a different range of functionality and applications. Because there are so many options, it’s important to understand each mobile platform along with market trends when planning LBS applications. Also, resources are constrained on a mobile phone, including screen size, memory, CPU, storage, and input method, making it especially important to properly understand and pick from available programming options.

In this section, all major mobile development platforms, from the iPhone and Android to the relatively Java Micro Edition (Java ME) are presented. The basic development process for each of these platforms look at to get a high-level understanding of the pros and cons of each platform.

Java Platform, Micro Edition (Java ME) differs from the other mobile platforms, in that it is no mobile operating system. Instead it is middle layer between a specific mobile operating system and value added services and applications offered by a service provider.

Android is a software environment built for mobile devices. It’s not a hardware plat-form. Android includes a Linux kernel-based OS, a rich UI, end-user applications, code libraries, application frameworks, multimedia support, and much more. Whereas components of the underlying OS are written in C or C++, user applications are built for Android in Java. Even the built-in applications are written in Java.

Apple, iPhone OS is a mobile operating system developed and distributed by Apple Inc. Originally unveiled in 2007 for the iPhone, it has been extended to support other Apple devices such as the iPod Touch, iPad, iPad mini, and second generation Apple TV. Unlike Microsoft’s Window Phone and Google’s Android, Apple does not license iOS for installation on non-Apple. [15]

## 1.5 Problem Formulation and Motivation

The main promise of location-based services is to provide new services to their customers based on the knowledge of their locations. Examples of these services include continuous live traffic reports (“Continuously, let me know if there is congestion within five minutes of my route”), food and drink finder (“Where is my nearest fast food restaurant”), and location-based advertising (“Send e- coupons to all cars that are within two miles of my gas station”). Due to the abundance of location-based data, location-based services have begun to integrate their functionality with database systems.

This research aims to raise the challenges and provide research directions to enable practical realization of preference-aware location-based services. It aims to enhance the answer of location-based queries. As the query answer may be returned to the users on their mobile devices with limited screen capabilities, it is of essence to enhance the quality of the answer and limit the answer to only those results that are of major interest to the users according to their preferences.

## 1.6 Objectives of the Thesis

The main objective of this research is to provide location-based services to smart phone user to meet according to their preference requirements. Due to resource limitations on mobile devices it is of essence to enhance the quality of the answer and limit the answer to only useful services according to users’ preference and context. The major objectives of this research are as follows;

1. To provide location-based web services to smart phone users without needing to browse using browser applications
2. To behave as a guide for the smart phone users to provide directions and rich information
3. To provide scalable personalized location-based services to users based on their preferences and user desired location range
4. To get services which meet smart phone users’ requirements on the limited mobile screen efficiently and accurately
5. To provide the historical records of the registered user

## 1.7 Organization of the Thesis

This thesis is laid out as follows. In Chapter 1, Mobile computing, location-based services and Web Services technology are generally introduced and the motivation, the objectives of the thesis are included.

Chapter 2 surveys the literature and some existing LBS systems. In this chapter, how other LBS application system applied the methods for their location-based services is also described. This chapter emphasizes on focus background knowledge and theories in order to apply in our proposed system. Then, the details processes of location-based services algorithms are discussed.

Chapter 3 presents the design and implementation of proposed ExcellentService system application which uses proximity detection algorithm based on symmetry approach and distance based location range algorithm to find location range accurately and efficiently. And also, there points out the importance of preference tree matching and simple tree matching algorithms for users preference queries matching in proposed system database. The components and steps taken by the ExcellentService of the proposed system are discussed.

Chapter 4 presents the performance analysis and evaluation proposed system with detail comparison results. Performance analysis and evaluation processes are comparing with the two main algorithms such as Distance Based and Proximity Detection Approach. There evaluation of computational processing time, evaluation of computational responsiveness time, concurrent user’s measurement and records amount results are compared in details with appropriate graphs.

Chapter 5 concludes the thesis with discussion, properties of proposed research work and conclusion. And further extensions of research are also discussed.

# CHAPTER 2

# LITERATURE REVIEW AND BACKGROUNG THEORIES

Many research activities are going in the area of data management techniques for LBS. In this chapter, Location-based services (LBS) with different methods and techniques developed by many researchers have been presented. Firstly, describes LBS with different position technology such as GPS. Most of mobile development platforms provide location APIs to develop location related application with their platform. Different mobile development platforms support LBS applications are presented in Second part. The integration of Web Services with mobile devices has many useful benefits. Location-based applications can be provided from mobile devices with Web Services technologies. Thirdly, provides LBS with Web Service technologies like SOAP, RESTful and XML Web Services. Relational data management technology with spatial support is utilized for location-based services. With different query types, the location and the distance between mobile users are provided. Location-based services with spatial database technologies such as MySQL and Oracle relational database management systems with spatial support are presented in Final part. And then, summaries of the chapter about the above research surveys.

## 2.1 LBS with GPS Positioning

Scalable personalized mobile information pushing platform, which can provide user-friendly and flexible location-based service by combining the GPS location-based services and the latest Web 2.0 technologies is presented by the authors. It proposed a Location-based Data and Service Middleware based on Service-Oriented Architecture in order to implement Mobile Information Pushing System involved in a variety of formats of data integration and conversion, as well as a combination of a wide range of services. Then, it proposed a novel 3-D Tag-Cloud module, so that it can visualize useful retrieval information even in the limited mobile screen. Especially, it designed multi-dimensional collaborative filtering algorithms in order to achieve dynamic personalized recommendation and mobile information sharing. It also developed a dynamic restaurant mobile location-based recommendation and discount coupons pushing system by cooperating with some restaurants. [23]

The authors in have described as GPS-based tracking system that collects the geographic co-ordinates from the GPS receiver carried by the user and transforms it to comprehensive location by using GIS software. Their proposed tracking system shows that the behavior and activity of a person can be deduced from the track data. Nevertheless, some security concerns are associated with GPS-based tracking such as revealing location information and the possibility of editing tracked data that may pose risks to innocent person. [25]

The authors in developed a system called commotion, which used GPS technology to learn over time about the user frequently visited locations, and allowed users to label these and to attach reminders to these places. The reminders could be audio or text based to-do-lists containing a number of items. Again however, the hardware required, which included a laptop, made it unrealistic to use such a system for a mass market, and since GPS was used for place detection, many places could not be covered. [29]

The authors in paper proposed a localization improvement algorithm in GPS interfering spots by integrating information of multiple sensors such as gyroscope and compass in the smart phone. The proposed algorithm is implemented in a smart phone and the performance is evaluated on a campus. The proposed algorithm has better performance than only the GPS location information in GPS interfering spots and maintains reasonable performance in open spaces where the GPS receiver is accurate. [36]

The research paper presented implementation of Location-based Services in Android using GPS and Web Services. With the help of A-GPS in phones and through Web Services using GPRS, Location-based Services can be implemented on Android based smart phones to provide these value-added services: advising clients of current traffic conditions, providing routing information, helping them find nearby hotels. [28]

The researchers in paper proposed an architecture of location-based services which uses GPS. In this paper the authors presented requirements for location-based reminders resulting from a qualitative study performed at small area in the city, and elaborate how these results are influencing ongoing design of a more comprehensive location-based reminder system. [35]

In research paper, a technique is given to send GPS coordinates of a mobile through a SMS to other mobile phones. Two algorithms, Kalman Filter and Velocity Renovation, which can be used in conjunction with GPS, are used as a basic for location tracking. The first coordinates are generated from a GPS assisted mobile on Google map, this location is then sent through SMS to another person. [27]

## 2.2 LBS on Android Platform

In this research the researchers showed how WalkTheDroid, a location simulator based on the Android emulator, uses recorded location tracks and device profiles to carry out different virtual test scenarios simultaneously. By inducing the location information on application layer, the simulator can analyze every android application separately. Although only android programs can be used for simulation, the general concepts for energy-efficient background tracking remain valid for all smartphone platforms since they use the same positioning technologies as Android such as GPS, WiFi and Cell-ID positioning. Because the simulator is based on the emulator, it behaves like a real Android device, and hence can be used for location-based testing without changing the code of the test application. Therefore WalkTheDroid provides a versatile and flexible simulator not only for the evaluation of energy- efficiency in different background tracking algorithms but also provides a tool for virtually walking around to test location-based applications. [11]

This approach is based on neighborhood information derived from the Voronoi diagram of the underlying spatial data set and can handle fundamental spatial query types, such as k nearest neighbor and range queries, as well as more advanced query types like reverse k-nearest neighbor, aggregate nearest neighbor, and spatial skyline. It evaluated VN-Authbased on real-world data sets using mobile devices (Google Droid smart phones withAndroid OS) as query clients. Compared to the current state-of-the-art approaches (i.e., methods based on Merkle Hash Trees), its experiments show that VN-Auth produces significantly smaller verification objects and is more computationally efficient, especially for queries with low selectivity. [26]

Based on the Android mobile phone, this study proposed the design and implementation of LBMMPS (Location-based Mobile Multimedia Pusher System), which pushes incident multimedia content to friends online in real time. The user can share incident videos, photos, and messages with other friends online in real time. By integrating GPS coordinates of incidents into Google Map, users can indicate the location where an incident occurs via a map interface. LBMMPS records the location of a moving trajectory of a special incident, and shows the incident's movements on the map. [22]

This approach presented an advertisement delivery system, called “OPAS,” and the service it provides. The system is based on mobile terminals held by shoppers in an underground mall. The terminals have a passing-by networking function. The paper also proposes location determination algorithms that make use of this system. An experimental OPAS has been built using a MANET (Mobile Ad Hoc Network) emulator and Android terminals, and has been used to evaluate the effectiveness of the proposed location determination algorithms. [40]

These authors proposed a hierarchical positioning algorithm, which provides a general algorithmic optimization in order to extend existing positioning APIs to energy-efficiently track a user's position without diminishing accuracy. The algorithm dynamically deactivates different positioning technologies and only activates the positioning method with the least energy consumption that at the same time provides sufficient accuracy to determine topological relationships. In that way, the algorithm can reliably and accurately determine, if the user leaves or enters predefined geographic areas to trigger events for proactive LBSs while preserving valuable energy resources. First results on Android handsets show a reduction in energy consumption of up to 90 percent in comparison to conventional GPS tracking. [10]

In this research, a new system for positioning and multimedia reporting, named electronic tour guide, is presented. The mobile device of the system communicates with the server, from which it receives electronic tourist maps, and the GPS satellites. Each location on the map with predetermined irregular shape avoids overlap with other areas and thus it increases the accuracy of the visited location. The device is guiding the tourist using the coordinates received from the GPS signals and the objects that are preloaded as electronic maps [31]. The main functions of the system are;

* 1. It is displaying the content according to user's movement, position and preferred selections.
  2. It generates reports for guiding using existing date stored from the server, the current position, the previous movement and the paid services.
  3. It reports and prepares the multimedia travel book based on the paid services.
  4. It prepares the daily trip report from images, text and audio files downloaded from server or recorded by user.
  5. It submits the recorded data for future individual, group or public usage.

## 2.3 LBS with Different Web Services Technologies

Location-based Web Services are information services accessible with mobile devices through the mobile network and utilizing the ability to make use of the location of the mobile device. By extending existing web services, location-based web services offer rich information services. SOAP and REST are two main types of web service technology. This two web services can be combined with location-based services to give the effective information to smartphone users. This section describes research works of the location-based services with different web service technologies such as SOAP and REST.

## 2.3.1 SOAP-based Location-based Web Services

This research proposed a GIS Web services architecture based on SOAP, XML and HTTP. A prototype is developed to expose GIS functionalities through standard Web interface. It is implemented using Microsoft Visual Studio .NET 2003, Oracle Spatial 10g database, Oracle MapViewer, and other supporting tools. The proposed Web services approach is found to be practical and promises a lowering of the entry barrier to adopt GIS for development. [3]

The authors in this paper proposed a privacy enhanced location-based service (PE-LBS) architecture which allows a mobile node to request location-based services via a proxy server hiding the network location of the mobile device while providing service accountability. The architecture is composed of six modules: location acquisition hardware, XML data record parser, XML service request, transport module, LBS proxy and service modules. Privacy Enhanced Technologies has been carefully integrated to enhance the privacy of our architecture by protection of personal identifiable information. One of the components of the architecture is the LBS Proxy Server, responsible of processing SOAP (message envelope) requests and generates response. When the SOAP request is received by a server, it gets bound to the class specified in the request. The proxy server works as a SOAP Dispatcher, by determining which class should handle a given request, and loading that class, if necessary. The SOAP server acts as an intermediary between a SOAP client and the requested service provider. [32]

## 2.3.2 RESTful-based Location-based Web Services

In the last years two important architectural and technological solutions have been proposed for improving Web applications design. The REST architectural style has been defined to describe the WWW architecture and to guide its future evolution preserving the fundamental characteristics of scalability. At the same time AJAX and related technologies have changed the way many Web applications are designed and implemented enhancing responsiveness and pervasiveness. The integration of these two approaches seems promising, but several issues must be tackled. For testing possible solutions, the authors designed and implemented a framework for building mobile Web applications using the REST architecture and AJAX technologies: the MIRAJ framework. It was used to develop a basic set of Location-Based-Services for tourist guidance in an archaeological park located in Metaponto, Italy [1].

Multimodal dialogue application framework can be used to implement specific mobile applications and dynamic HTTP-based REST services. These infrastructures can overcome the technical limitation imposed by current mobile device hardware and software. In addition, new services of independent providers can be added easily. The authors created a custom HTTP/REST Meta Web Service for mobile location-based scenarios and explained how this service integrates into a multimodal dialogue framework. In addition, a real-world application scenario is provided and explained generic dialogue framework and a specific implementation of a new Meta Web Service. With the new Meta Web Service with RESTful technology, meaningful information can be provided for travelers or tourists. [33]

In this work, LBS architecture with RESTful Web Service from the perspective of service design and horizontal service integration is discussed. The main goals of this approach are to produce a LBS design framework that is usable across a wide variety of LBS scenarios, is based on well-established and widely used Web technologies, and provides open and standardized access to location services. The approach, called Tiled Feeds, can be used to create LBS that are open and accessible to basic Web clients, and can be easily integrated and remixed with other LBS to provide rich, personalized views and services. The main starting point for this approach is Web Architecture and the underlying architectural style of RESTful Web Service. This system started from the assumption that spatial information can be represented as features with a spatial association, and that these features can be exposed through the most popular pattern of RESTful Web services today- feeds. This approach is also an ideal starting point for building LBS that can be used across a variety of mobile delivery platforms, either as Web-based applications designed for mobile Web browsers, or as native applications being backed by Web-based services. [34]

## 2.4 LBS with MySQL Spatial Support

Design and implementation for location-based service platform is presented in this research. This platform is divided into client side and server side. Client side of this platform software is developed based on the following techniques: mobile SVG based JSR226 and JSR179 specification, Bluetooth, mobile media, map slicing, map layering, and J2ME. Sever side is developed by XML, J2EE, and MySQL. This platform can provide the following services: information of map segment query, shortest path to destination query by Dijkstra algorithm, bus route query, and route navigates. A general architecture and modular for locationbasedservices is also provided based on this platform with interactive, multimedia and location- aware. Many valuable LBS can be enabled on this platform by some suitable modification. [24]

This research provided detail on the implementation of a location awareness system which gives the user's current location, sends this location using SMS (Short Message Service) plus sharing location with friends and family and views them on Google maps. Users can also take benefit of this application in emergency situations by using emergency feature of this application. To get the location coordinates, application is using GPS (Global Positioning System) as location provider. The application design has five parts: a mobile client, a web server, a database, GPS system and a map service. A mobile client which consists of a mobile and GPS receiver finds the location of the user to get aware of his location. In order to share this location the mobile client sends this location to the web server from where other users can get this location if they have the authentication provided by the user. The mobile client is implemented using J2ME which is one of the most promising software platforms for mobile devices. The web server side programming is done using PHP and database is maintained by MYSQL. [16]

## 2.5 Positioning of the User Location

One essential aspect in LBS is fixing the location of users through their mobile device. Location of users is regarded as the spatial context in LBS applications. Such location data are often used as a key in real applications, from which the overall context informs mobile applications. In this chapter the various technologies currently available to provide location data about mobile devices used for LBS are discussed; these include satellite positioning systems (the most commonly used being the United States operated Global Positioning System (GPS)), positioning methods based on mobile telecommunications networks (network-based), technologies used for short range positioning (such as Bluetooth and tags), and hybrid positioning solutions. The principle and characteristics of each technology are described as well as their strengths and weaknesses.

The purpose and type of LBS application determines what level of accuracy and consistency of position fixing is acceptable. User location data can be classed into different levels of granularity, which is closely related to different types of applications. Location data can be point-oriented, line-oriented and area-oriented. Point-oriented location data with high accuracy can be at a single point with coordinates, or for a very small area with an indicator of accuracy (e.g., x, y). Position fixing to a point can be used for applications such as emergency services and other services requiring an accurate fix of user location. Line-oriented location data can be used for LBS applications with line-based geographic features (i.e., roads, rails and rivers), such as vehicle/ fleet management services. Area-oriented location data can have different levels of accuracy (medium to low) depending on the area covered. Such location data can be sufficient for those LBS applications that only need to know whether users are within a certain service area, such as providing local visitor information, weather reports, tariff services and congestion alert services.

Thus, depending on the service requested by a user, the positioning technology might be switched, changing the overheads and latency along with the accuracy and consistency. However, it needs to be pointed out that the accuracy of position fixing is not the only factor needed to ensure that applications deliver more useful information with less error.

## 2.5.1 Positioning Technologies

There are a number of positions fixing technologies as well as combined hybrid solutions to locating users. Positioning methods can be generally categorized into network-based, device-based and hybrid methods. They can also be differentiated as to whether they are integrated or stand-alone positioning infrastructures, and again depending on whether they are satellite-based, network-based (cellular) or indoor infrastructures. Device-based positioning is usually based on GPS and hence both a satellite-based and stand-alone infrastructure. Network-based positioning can be regarded as integrated, as the network is also used for communication and data transmission with gateways to other networks and systems. Various positioning technologies are discussed based on the categories of device-based, network-based and hybrid integrated. In addition, a range of technologies which can be used to locate users within a comparatively short range are described.

For device-based positioning (also known as terminal-based or receiver-based positioning), the mobile device determines its position using signals it receives. In other words, signal measurements and the computation to determine a position are performed by the receiver located within a mobile device. No network connection is required. There is more privacy in using device-based methods. However, there is high power consumption for mobile devices to carry out positioning and positioning stops once signals can no longer be received (such as indoors or in tunnels). Global Navigation Satellite Systems (GNSS) are considered as a class of device-based positioning. The GPS is the most well-known of these.

Network-based positioning methods use the transmitter base stations of a mobile telecommunications network to locate a mobile device by measuring the signal travelling to and from a set of base stations. Through signal measurements, the direction and/or length of an individual radio path can be computed and the position of a mobile device can be derived using computational geometry. Connection to server-side services is needed to position mobile devices when using network-based methods. There is low power consumption and fewer requirements made of the mobile devices. Usually, a range of network-based positioning methods can be operated using mobile telecommunication networks. The networks are used both for positioning and communication and must share the bandwidth. Such network-based positioning can work in indoor environments as long as there is sufficient signal strength. If position data with high accuracy are required, there can be high costs in terms of signaling overhead and thus reduced capacity for voice/data transmission.

A range of other technologies can also be used for positioning, particularly for comparative short range positioning. Such technologies include Bluetooth technology, Radio Frequency Identification (RFID) and Wireless Local Area Networks (WLANs).

## 2.5.2 Global Positioning System (GPS)

The Global Positioning System (GPS) is a satellite-based global radio navigation system. GPS consists of 21 constellation satellites plus three spare ones orbiting the Earth every 12 hours. The GPS satellites are in orbit at high altitude (about 20200km). The technology used by GPS enables the system to fix any position in the world at any time of the day when there is a clear view of the necessary number of satellites. The GPS configuration assures the visibility of five to eight satellites at any point on the Earth at the same time. Currently, the GPS provides both military and civilian positioning services. Over the years, GPS receivers have become small enough and cheap enough to be used by the general public as handheld gadgets or incorporate into mobile phones. The GPS can thus be used in a wide range of applications, such as precise positioning, navigation, surveying, mapping, and engineering and of course for LBS. The general design principles of the GPS can be summarized as follows;

1. Be suitable for all classes of platforms whether they be aircraft, ships or land-based vehicles as well as other satellites.
2. Provide positioning in real-time along with the ability to determine the time and velocity.
3. Reference all positioning to a single global geodetic datum.
4. Be capable of preferentially providing higher accuracies to certain classes of user.
5. Be scalable to unlimited numbers of users worldwide.
6. Has low unit cost with low power overhead.

Many LBS applications (such as in-car navigation) are supported by GPS positioning. Locating users through GPS-enabled devices is a device-based positioning method. A GPS-enabled mobile device means that a satellite navigation receiver is built into the device. Devices with built-in GPS can offer a higher degree of location accuracy that can be within ±3to ±15 meters. However, such accuracy may not be consistent in all environments. The accuracy may vary from rural areas to high-rise urban areas due to its need for a clear view of the sky and signals from three or four GPS satellites. There are still other disadvantages for such devices used in mobile situations, such as the higher power consumption and an increased cost in the manufacture of the device.

## 2.5.2.1 Basic Principles of GPS

The basic principle of positioning using GPS is trilateration based on distance measurements (range measurements), using satellites as reference points, although there have been advances in the data processing over the year since the system became operational as well as enhancements to the satellite constellation, such as the Wide Area Augmentation System (WAAS) which increases accuracy. A position on the Earth is determined by the distance measured between the GPS receiver at a particular position and a range of satellites. Mathematically four such measurements are needed to calculate an exact position. Three simultaneous measurements from three different satellites can narrow down the position to two points on the intersection of three spheres as shown in figure 2.1. In theory, three satellites ranges are sufficient to determine a position. In practice, the fourth satellite is used as an additional measurement to eliminate what is known as receiver clock offset. In situations where the altitude is already known, the position can be determined using one less satellite measurement.

The distance from a position on the Earth to a satellite is determined by the time taken for the radio signal to travel between that satellite and the GPS receiver: distance= speed of light (approximately 300 000 km/hr) \* travel time. The radio signal generated by GPS includes carrier signals, Pseudo-Random Noise (PRN) codes and navigation messages.

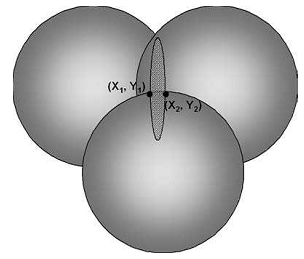


Figure 2.1 Measurements Two Position Points Determined by the Three Satellite.

## 2.5.2.2 Principle of Differential GPS

To achieve more accurate position measurement than standard GPS, a relative (differential) positioning technique has been developed known as Differential GPS (DGPS). In point (absolute) positioning, the position is determined through a single receiver and multiple satellites. The configuration of the satellites and any remaining errors will directly affect the accuracy of the point positioning. The basic principle of DGPS is to remove these errors by determining the difference between the measured distance at an unknown point whose position is to be fixed and an approximation of the true distance as calculated by a receiver at a known position.

Thus DGPS uses at least two receivers for positioning. One GPS receiver acts as a reference receiver, which is place at a known stationary position with accurately surveyed coordinates (x, y, z). The other GPS receiver, known as the user receiver, is at an unknown position that can be mobile. Both reference and the user receiver track the same satellites simultaneously. The difference between the two simultaneous measurements by both receivers is calculated. Thus, the systematic error sources, which are common to the reference receiver and the user receiver, can be determined. DGPS positioning is an effective way of increasing positioning accuracy by removing the systematic errors. The basic concept of DGPS is illustrated in figure 2.2. It is important that both the reference and user receivers track the same satellites to generate their measurements, otherwise severe errors may result.

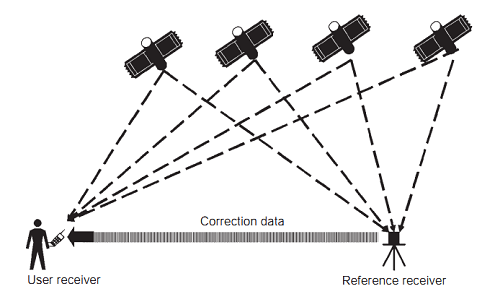


Figure 2.2 The Basic Principle of Differential GPS.

Positioning with DGPS is implemented by DGPS services. DGPS services can enable users to obtain higher positioning accuracy. They are usually provided by governments, industries and professional organizations. However, in order to use such services, additional hardware is required to be able to receive and process the differential corrections. DGPS services normally involve some type of wireless transmission system.

## 2.5.3 Network-Based Positioning Technologies

In principle, network-based positioning technologies use a network of base stations to locate a mobile device by measuring the signal travelling between the mobile device and a set of base stations. The direction and/or length of individual radio paths can be computed through signal measurements, and the position of a mobile device can then be determined from geometric relationships. In this section, the principles of some commonly used network-based positioning technologies are described. These technologies include Network Cell Identification (Cell-ID) technology, Angle of Arrival (AOA) technology and time delay methods, including Time of Arrival (TOA), Time Difference of Arrival (TDOA) and Enhanced-Observed Time Difference method (E-OTD). One of the main advantages in locating the position of a user’s mobile device for LBS through the use of these network-based positioning technologies is that they are based on the telecommunication networks servicing mobile devices with voice and data and do not require any extra software or hardware to be installed in mobile devices. The term mobile device generally refers here to a handheld, mobile wireless-enabled device such as a mobile phone or PDA that is wireless enabled. A mobile device is also known in some texts as a terminal or a mobile station.

## 2.5.4 Short Range Positioning Technologies

The positioning technologies discussed above can be used for large area coverage: GPS positioning has global coverage and network-based positioning methods cover the operational area of whole mobile telecommunications networks. There is, however, a range of technologies that can be used to position devices over comparatively short ranges. They are frequently employed for positioning in indoor environments. These technologies include Wireless LANs (WiFi), Bluetooth, Radio Frequency Identification (RFID), Ultra Wide Band positioning, as well as ultrasonic, infrared, camera-assisted and sensor-assisted technologies.

## 2.5.4.1 WiFi-Based Positioning

WiFi is a wireless networking standard defined in the IEEE 802.11 series. WiFi are also known as Wireless Local Area Networks (WLANs). WiFi-enabled devices (often equipped with a WiFi adapter) can be connected to a wireless network via access points on the network with a speed of up to 54 Mbps. These local networks are often connected to a larger network such as a corporate network. The basic principle of using WiFi technology to position a mobile device (referred to as WiFi-based positioning) is illustrated in figure 3.3. The location of a device is determined by measuring the strength of signals received at two or more access points. The signals communicated between mobile devices and access points are often referred to as beacons, which contain packets of information. There are two ways of transmitting these signals for positioning: uplink and downlink. For uplink, a WiFi-enabled mobile device generates beacons. These beacons are received by those access points that are in range, that are able to receive the signal, and the network establishes the position. For downlink, access points transmit beacons containing their ID along with other information. The device continuously receives beacons from those access points in the nearby vicinity, detects the best quality signal for trans-mission and identifies the access point for positioning.

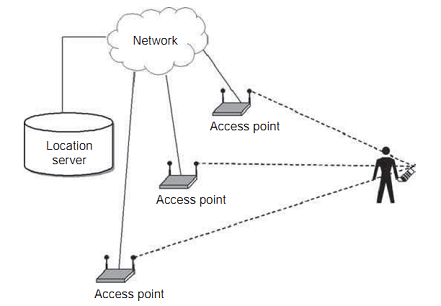


Figure 2.3 Illustration of WiFi-based Positioning

WiFi-based positioning only works in areas where there is network coverage from access points. This method can work in indoor environments. It is not suitable for a very large scalable implementation; therefore, the coverage is relatively small although it may be larger than other indoor positioning technologies such as Bluetooth. Currently, WiFi technology is widely used in offices, homes, airports, cafes, hotels and public places. WiFi-based positioning can also be used for outdoor environments, usually in dense urban areas. The accuracy of this method is generally not very high, and also depends on the density of access points and the strength of signals transmitted.

## 2.5.4.2 Bluetooth Technology Used for Positioning

Bluetooth technology provides an ad hoc approach for low power, short range wireless connections for voice and data transmission between various devices within a nominal 10 m range. It operates in the globally available unlicensed 2.4 GHz ISM radio frequency band. Bluetooth-enabled devices can be linked up for ad hoc networking with other Bluetooth-enabled devices when they come within range. Most mobile devices are currently fitted with Bluetooth-enabled transmitter chips.

Although there is no Bluetooth specification designed to sup-port positioning services, Bluetooth technology can be used to locate user mobile devices. Various solutions have been developed based on Bluetooth technology for short range and indoor positioning. Bluetooth-enabled devices can transmit signals containing information such as device identity and profile. Such signals can be picked up by a host device and used to identify the presence of other devices when they are within the 10m communication range. Signals can also be processed and computed to determine the position of a device. Position information should be able to be exchanged between Bluetooth devices locally or with the location server in a network.

The main advantage of Bluetooth-based positioning is that it can be deployed rapidly, with easy maintenance and low cost. Positioning with Bluetooth technology can be used by those LBS applications where short range coverage with approximate area accuracy is sufficient. For example, this might be short range ‘push’ information services and alerts. Another advantage of Bluetooth-based positioning is that it is designed to consume low levels of power and is suitable for small mobile devices. Moreover, the internal authentication and encryption provided by Bluetooth technology provides for better security than other wireless technologies, which rely on external means of security.

## 2.6 Context in Location-Based Services

Another important aspect is to understand how context is used in LBS. In order to provide tailored data and information services to users in mobile situations, most LBS applications have some level of context awareness. Context-awareness can be used for improving system design, identifying relevant content, enhancing communication and in delivering services. Therefore, understanding context is central to LBS, and there are a number of concepts that need to be addressed. Important context information can include user location and the surrounding environment, user situations during activities, time of day and date, and the technologies involved (devices, networks and systems). Context can be spatial-temporal, and can also be related to user characteristics, personal preferences and behaviors. The information and services delivered to users in LBS are strongly influenced by and linked to these relevant contexts.

LBS applications are user focused and task specific. The main purpose of understanding context in LBS is to enhance the ability of service providers to supply users in mobile situations with information and data services that are viewed as having a high level of utility. It is, therefore, important for application designers to choose what context(s) to use in their applications and to determine what context-aware behavior(s) to support in their applications. It is generally viewed that applications provided via a mobile device can support better services for users when the mobile device being serviced can identify more contexts through its usage. Broadly speaking, using mobile devices can benefit from awareness of context in two ways: firstly, through adaptation of content to changes in the environment; and secondly, through improvement of the interaction with users. Context information provides a means to filter the flow of information from service provider to users, which can help address the problem of information overload and can be used to give additional meaning to the information requests made by a user.

Research into context awareness, in general, focuses on location, mobility and time [2]. Location can be a particular x, y position, a surrounding environment or a meaningful place such as home or office. Mobility can reflect users in situations where they are continuously on the move or users being in a certain place. Time can add another dimension to context, either as a momentary instant or as a fixed or floating period; time can be absolute or relative. Context can also relate to user ability and preferences, and to the nature of activities being undertaken. Furthermore, technologies employed in LBS contribute context as well. For example, this can include type and functionality of a mobile device, network connectivity and communication media supported. Context has a direct influence on many respects of LBS, such as communication between users and mobile devices, content provided for specific applications, system design, service delivery and much more. Therefore, context-awareness is one of the key means of enhancing LBS in order to better meet user requirements. Use of context will inevitably need to be pragmatic, the general challenge being the identification of relevant context(s) in terms of situation environment and time that can be sufficiently and usefully captured. Both situation and environment are to a large degree characterized by continuity over time. Thus the context history of a user, the time-line, can itself be an important means of mining contextual information for given situations or environment. [4]

Firstly, in this section, context and context-awareness are defined from a broader perspective. This means that LBS is then considered and a view of context as an interaction concept between user, technology and environment is developed. Next, environment as context (location and time), technology as context and the user as context are looked at individually. In the final part of this section the dynamics of context are considered.

## 2.6.1 Context and Context-Awareness

Context is important to every aspect of human life. With the fast development of wireless communication and increasing use of mobile devices, context-awareness has become an increasingly important consideration in developing systems and applications [3, 4 and 5]. Being aware of the context(s) in which systems and applications are run can improve their ability to adapt and react to user situations, such as surrounding locations, people and objects such as other devices. Thus, such context-awareness is better able to support computer use in applications of mobile and ubiquitous computing, and in different environments. The early focus of research was mainly on the design of devices but has broadened to a consideration of context and context-awareness in systems and applications.

Context is generally regarded as data or information which describes the situation that is relevant to and has influence on the state of users, systems and applications. ‘Context is any information that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves’ [6]. Accordingly, any information can be viewed as context if it can be used to characterize the user situation in an interaction. Where you are, who you are with and what resources are nearby have also been highlighted as three important aspects of context in mobile distributed computing [7, 8 and 9].

Context-aware is often described as the ability to discover changes in user situations (context), being responsive to them and using acquired context to provide services in meeting user needs [7, 6 and 10]. In context-aware systems, contextual information can either be obtained by requiring users to enter data into the system, or can be acquired by sensing devices, monitoring patterns of use and the surrounding environment. Such contextual information is then used to dynamically tailor the response of the system. By being context-aware systems and applications can better support their users with features such as supporting the presentation of information to users, triggering the automatic execution of a service, or tagging information with some aspect of context (such as location or time) to better facilitate later retrieval. Many context-aware systems and applications use physical location as the main contextual element [11]; there are some that combine location with other features such as user preferences. As stated above, context history stored within a context-aware system is widely believed to be useful but in reality is rarely used. Instead most applications tend to use small amounts of contextual information in a piecemeal way.

## 2.6.2 Dynamics of Context

One important aspect of context in LBS is its dynamic characteristics that are embedded within the context. The three main elements in the concept (environment, technology and user) have been used as three main strands to define environment context, technology context and user context in LBS. Due to mobility being a fundamental aspect, LBS needs to be studied through the real-time dynamic interaction between user, environments and technologies. Therefore, it is important to understand the context in LBS from this dynamic perspective.

Context is not constant unchanged information. One obvious and unique dynamic aspect in LBS is the environment. The environment is full of changes, such as changes of location, time and situation. Such dynamic characteristics in environment context require LBS applications to be aware of the continuous change in the context. Another aspect of dynamics in context is in user context, which includes variation in user preferences, in user emotional and physiological conditions, all of which can change over time and from place to place. Personal preferences and behaviors can also vary according to different activities and usage situations. The technology context contributes to a further aspect of the dynamics in context. The availability and usage of a range of technologies to deliver LBS is more likely to change in mobile situations, which will affect the ability of a provider to deliver consistent and uninterrupted services. Although these three forms of context are often only considered in application development individually, they are actually interwoven. And so it is with the dynamic nature of these contexts. Such interwoven dynamics have effects on the way users interact with devices and with surrounding environments. It is essential in the development of LBS to establish a clear understanding of contexts within a dynamic real-time interaction framework (i.e., how users interact with technology and with environment in practice).

## 2.7 Web Services Technologies

A Service Oriented Architecture can be achieved in many different ways, but Web Services are definitely the most popular one. In fact they are so common that the two terms are often used interchangeably.

Web Services are a realization of the SOA paradigm with existing Internet and WWW technologies, foremost the Hypertext Transfer Protocol (HTTP) and the eXtensible Markup Language (XML) [12]. HTTP is used as the transport mechanism for XML based service message exchanges. Web Services are not exclusively bound to HTTP yet its widespread deployment in existing Internet environments makes it the primary transport protocol for today's Web Services.

Two important architectural concepts form the Web Services application stack. The more formal SOAP approach that builds on a large set of standards defined by the W3C, OASIS and other standard bodies. And the less specified REST approach.

## 2.7.1 SOAP based Web Services

The three core technologies of SOAP based Web Services are Simple Object Access Protocol (SOAP), Web Service Description Language (WSDL) and Universal Description, Discovery and Integration (UDDI). Together they are used as a direct application of the SOA find-bind-execute cycle as shown in figure 2.4. The Service Provider must describe its SOAP based Web Service with WSDL (low level) and UDDI (high level for lookup). A Service Consumer can look for a specific service in an UDDI-Registry. If the Service Consumer finds an appropriate service, it can bind to it with the information provided by the registry and invoke it via its WSDL service description. The actual data exchange between the service endpoints is carried out with SOAP. SOAP and WSDL are standardized by the W3C, whereas UDDI is a standard maintained by OASIS.

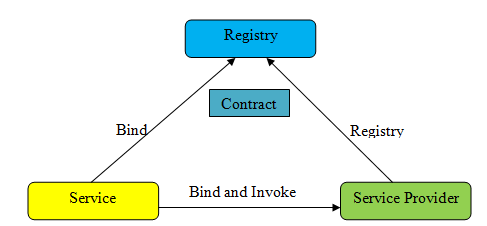


Figure 2.4 SOA Find-Bind-Execute Cycle.

## 2.7.1.1 Simple Object Access Protocol (SOAP)

SOAP (originally an acronym for Simple Object Access Protocol, now it is just a name [13] is the key technology for SOAP based Web Services and responsible for exchanging data between to service endpoints. Like its companion technologies WSDL and UDDI it is built on top of XML and provides a schema in the form of the SOAP-specification that allows for system independent data exchange. It has been standardized in two different versions, SOAP 1.1 and SOAP 1.2.

A SOAP document is referred to as a SOAP message or SOAP envelope [13]. Each SOAP envelope consists of a header and a body. The header carries meta-information, quality of service constraints and routing directives relevant for the service exchange. The body contains the payload in form of a (partial) XML document. Binary data can be included encoded in the message body.

SOAP can be used for Remote Procedure Calls (RPC) as well as literal document exchange. For RPC, the body contains XML serialized method calls, which are answered by the remote station with a SOAP message containing the return values. This way Web Services can be used as an alternative to programming language specific RPC methods like amongst others Java RMI, CORBA or Microsoft DCOM.

Today much more common than RPC is literal document exchange. With literal document exchange, XML serialized documents are transferred in the SOAP body. This could be e.g. a booking request containing all the relevant booking data generated by the calling Web Service, which gets answered by the called service with a confirmation or an error message.

An important concept of the SOAP messaging framework is the SOAP processing model. Every SOAP message originates at an initial SOAP sender and reaches a specific ultimate receiver via zero or more SOAP intermediaries. The way a SOAP message travels in a network is described as its message path. The message path information is included in the SOAP header. [14]

## 2.7.1.2 Web Services Description Language (WSDL)

Together with SOAP, WSDL forms the heart of every SOAP based service architecture. It is used for describing Web Services with XML based metadata information in such a way that other services can automatically use them. This way it is a key enabler for a Service Oriented Architecture, as it allows for programmatically exploring and using a service. The currently most widely deployed version is WSDL 1.1. Each WSDL document consists of a set of definitions (<definitions>-root-tag) that describe the what, the how and the where of a specific service.

## 2.7.1.3 Universal Description, Discovery and Integration (UDDI)

The UDDI specification provides a description of Web Services and allows them to be listed in an UDDI-registry. Much like in a classified directory, service consumers can look up Web Services in an UDDI-registry and get all the relevant information for connecting to them. An UDDI-registry can be run by a company for providing services to their customers, as well as for internal usage. Microsoft, IBM and SAP used to maintain the Universal Business Registry, which served as a broker place for Web Services all over the world, but was discontinued in 2006.

Since WSDL serves as the “technical instruction manual” for SOAP based Web Service, UDDI needs to tightly integrate with it. This is done via mappings in the UDDI-document that directly map WSDL entities to UDDI entities [14]. Additionally to the technical mapping of the service, the UDDI document provides meta-information that describes the service and allows for criteria based discovery.

## 2.7.2 REST based Web Services

The idea of Web Services based on the Representational State Transfer (REST) principle is to directly rely on the technologies already provided by the World Wide Web, rather than creating new ones. The key technologies for REST are [15] as follows;

1. Uniform Resource Identifiers (URIs).
2. HTTP.
3. XML.

While SOAP-based Web Services also use these specifications as the fundament for higher level protocols, REST directly relies on them for providing a Service Oriented Architecture. In contrast to SOAP based services, which reduce the application layer HTTP protocol to a transport protocol that could easily be substituted by any other protocol for transportation like e.g. SMTP, XMPP or JMS [16], REST Web Services directly leverage the methods provided by HTTP for CRUD (Create, Read, Update, Delete) operations. In the REST world the HTTP methods are called verbs, which the developer can use to describe the intended action. The relationship between CRUD-operations and HTTP methods is shown in table 2.1. [17]

Table 2.1 Relationship between HTTP methods and CRUD Operations

|  |  |
| --- | --- |
| **General Action** | **HTTP Method** |
| Create | PUT |
| Read | GET |
| Update | POST |
| Delete | DELETE |

The proper use of these HTTP methods (verbs) provides RESTful Web Services with a uniform interface, as they specify the allowed operations which always leave the service in a consistent state [18, 19]. Key properties of RESTful Web Services are [18, 17].

## 2.7.2.1 Stateless

REST-based Web Services are stateless, which means that each request travelling from client to server must always contain all the necessary context to understand the request. A good test whether a Web Service should be designed RESTful could be to test if the interaction with the service survives a restart of the host without any impact.

## 2.7.2.2 Resource Identification with URIs

RESTful Web Services are built with URIs that uniquely identifies resources. The service provider publishes a service under an unique URI which can be accessed by the service consumer over HTTP by using the appropriate verbs.

## 2.7.2.3 Communication through the transfer of representations of resources

The response of a request is the representation of a REST Web Service resource and is usually transferred in the form of pure XML, although any other format like e.g., JavaScript Object Notation (JSON) is also imaginable.

As a must in any Service Oriented Architecture, RESTful Web Services too have to be described in order to be properly invoked. Opposed to SOAP based Web Services which precisely describe the exchanged messages in a WSDL-file, there exists, no such specification for RESTful services. But WSDL can be “borrowed” from SOAP services – Instead of the traditional SOAP-binding, the newly introduced HTTP-binding of WSDL 2.0 could be used to describe the service on the basis of HTTP [20].

Another alternative is to describe the invocation in a humanly understandable format and make it publicly available (e.g., in the form of an HTML-page, or a WSDL-file that is read by a human designing the service consumer). [21]

## 2.8 Android

Android is a mobile operating system running on the Linux kernel. It was initially developed by Google and later the Open Handset Alliance. It allows developers to write managed code in the Java language, controlling the device via Google-developed Java libraries. Figure 2.5 displays the main architecture of the Android operating system.

Android application development is done in Java using the Android Java libraries, preferably in Eclipse (an open source development environment), and applications can be submitted to the Android Market. There’s no approval process in the Android Market. Submitted applications go live instantly, unlike those for the Apple App Store.

Google has created a powerful and robust mobile operating system in the form of Android. The open nature of Android is already disrupting legacy mobile OSs as well as even the iPhone. Many device manufacturers are moving from legacy platforms to Android and releasing new Android phones.

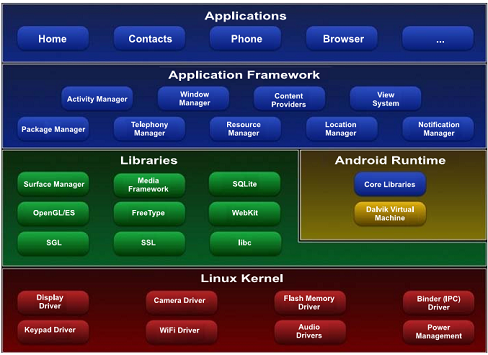


Figure 2.5 Major Components of the Android Operating System

Learning Android development is easy, with Java developers making the quickest jump. Android Java libraries are intuitive, and it helps greatly that the platform is open source. Here are some major properties of the Android platform:

1. **Multi-touch on a limited number of Android devices**; curiously, multi-touch capabilities have been left out of the Android SDK until recently except for a few phones just coming out, unlike the iPhone. It’s claimed that this feature will be enabled widely in the near future because some developers created multi-touch demos on Android phones by hacking the OS.
2. **Multitasking:** Android supports multitasking, but this requires a bit of explanation. On a Windows desktop, you can freely start, end, and switch between actively running programs. They continue running the same way whether they’re in focus or in the background. In Android, the one don’t have to shut down programs because the system will do it for them when it runs low on memory.

Unlike Windows, Android makes a clear distinction between a program that’s doing work (like downloading a file or playing some music) and one that’s sitting around waiting for the user to return. The commonly quoted “Android can run programs in the background” case is the one where the background program is doing work.

## 2.9 Computation of Location Range Query

Spatial databases have witnessed an increasing number of applications recently, partially due to the fast advance in the fields of mobile computing and embedded systems and the spread of the Internet. For example, it is quite common these days that people want to figure out the driving or walking directions from their handheld devices (mobile phones or PDAs). However, facing the huge amount of spatial data collected by various devices, such as sensors and satellites, and limited bandwidth and/or computing power of handheld devices, how to deliver light but usable results to the clients is a very interesting, and of course, challenging task.

For our purpose, light refers to the fact that the representation of the query results must be small in size, and it is important for three reasons. First of all, the client-server bandwidth is often limited. This is especially true for mobile computing and embedded systems, which prevents the communication of query results with a large size. Moreover, it is equally the same for applications with PCs over the Internet. In these scenarios, the response time is a very critical factor for attracting users to choose the service of a product among different alternatives, (e.g., Google Map versus MapQuest), since long response time may blemish the user experience. This is especially important when the query results have large scale. Second, clients’ devices are often limited in both computational and memory resources. Large query results make it extremely difficult for clients to process, if not impossible. This is especially true for mobile computing and embedded systems. Third, when the query result size is large, it puts a computational and I/O burden on the server. The database indexing community has devoted a lot of effort in designing various efficient index structures to speed up query processing, but the result size imposes an inherent lower bound on the query processing cost. If we return a small representation of the whole query results, there is also the potential of reducing the processing cost on the server and getting around this lower bound. As we see, simply applying compression techniques only solves the first problem, but not the latter two.

Usability refers to the question of whether the user could derive meaningful knowledge from the query results. Note that more results do not necessarily imply better usability. On the contrary, too much information may do more harm than good, which is commonly known as the information overload problem. As a concrete example, suppose that a user issues a query to her GPS device to find restaurants in the downtown Boston area. Most readers having used a GPS device could quickly realize that the results returned in this case could be almost useless to the client for making a choice. The results (i.e., a large set of points) shown on the small screen of a handheld device may squeeze together and overlap. It is hard to differentiate them, let alone use this information! A properly sized representation of the results will actually improve usability. In addition, usability is often related to another component, namely, query inter-activeness that has become more and more important. Inter-activeness refers to the capability of letting the user provide feedback to the server and refine the query results as he or she wishes. This is important as very often, the user would like to have a rough idea for a large region first, which provides valuable information to narrow down her query to specific regions. In the above example, it is much more meaningful to tell the user a few areas with high concentration of restaurants (possibly with additional attributes, such as Italian versus American restaurants), so that she could further refine her query range.

With the advance of wireless communication technology, it is quite common for people to view maps or get related services from the handheld devices, such as mobile phones and PDAs. Range queries, as one of the most commonly used tools, are often posed by the users to retrieve needful information from a spatial database. However, due to the limits of communication bandwidth and hardware power of handheld devices, displaying all the results of a range query on a handheld device is neither communication-efficient nor informative to the users. This is simply because that there are often too many results returned from a range query. In view of this problem, we present a novel idea that a concise representation of a specified size for the range query results shall be computed and returned to the user. Such a concise range query not only reduces communication costs, but also offers better usability to the users, providing an opportunity for interactive exploration. The usefulness of the concise range queries is confirmed by comparing it with other possible alternatives, such as distance base range query calculation.

## 2.9.1 Range Query Generating Algorithm

Since the circle is a frequently used component in pictures and graphs, a procedure for generating either full circles or circular arcs is included in most graphics packages. More generally, a single procedure can be provided to display either circular or elliptical curves.

Properties of Circle Range is defined as the set of points that are all at a given distance r from a center position (xc,yc). This distance relationship is expressed by the Pythagorean Theorem in Cartesian coordinates as

(2.1)

We would use this equation to calculate the position of points on a circle circumference by stepping along the x axis in unit step from xc – r to xc + r and calculating the corresponding y values at each position as

(2.2)

But this is not the best method for generating a circle. One problem with this approach is that it involves considerable computation at each step. Moreover, the spacing between plotted pixel positions is not uniform. We could adjust the spacing by interchanging x and y (stepping through y values and calculating x values) whenever the absolute value of the slope of the circle is greater than 1. But this simply increases the computation and processing required by the algorithm.

Another way to eliminate the unequal spacing is to calculate points along the circular boundary using polar coordinates r and . Expressing the circle equation in parametric polar from yields the pair of equations

(2.3)

When a display is generated with these equations using a fixed angular step size, a circle is plotted with equally spaced points along the circumference. The step size chosen for depends on the application and the display device. Larger angular separations along the circumference can be connected with straight line segments to approximate the circular path. For a more continuous boundary on a raster display, we can set the step size at 1/ r. This plots pixel positions that are approximately one unit apart.

Computation can be reduced by considering the symmetry of circles. The shape of the circle is similar in each quadrant. We can generate the circle section in the second quadrant of the *xy* plane by noting that the two circle sections are symmetric with respect to the y axis. And circle sections in the third and fourth quadrants can be obtained from sections in the first and second quadrants by considering symmetry about the x axis. We can take this one step further and note that there is also symmetry between octants. Circle sections in adjacent octants within one quadrant are symmetric with respect to the 45 line dividing the two octants. These symmetry conditions a point at position (*x, y*) on a one-eight circle sector is mapped into the seven circle points in the other octants of the *xy* plane. Taking advantage of the circle symmetry in this way, we can generate all pixel positions around a circle by calculating only the points within the sector from *x*=0 to *x*=*y*.

Determining pixel positions along a circle circumference using either equation (2.1) or equation (2.3) still requires a good deal of computation time. The Cartesian equation (2.1) involves multiplications and square root calculations, while the parametric equations contain multiplications and trigonometric calculations. More efficient circle algorithms are based on incremental calculation of decision parameter, are in the Bresenham line algorithm, which involves only simple integer operations.

Bresenham’s line algorithm for raster displays is adapted to circle generation by setting up decision parameters for finding the closet pixel to the circumference at each sampling step. The circle equation (2.1), however, is nonlinear, so that square root evaluations would be required to compute pixel distances from a circular path. Bresenham’s circle algorithm avoids these square root calculations by comparing the squares of the pixel separation distances.

A method for direct distance comparison is to test the halfway position between two pixels to determine if this midpoint is inside or outside the circle boundary. This method is more easily applied to other conics; and for an integer circle radius, the midpoint approach generate the same pixel positions as the Bresenham`s circle algorithm. Also, the error involved in locating pixel positions along any conic section using the midpoint test is limited to one-half the pixel separation.

## 2.9.2 Proximity Detection Algorithm with Symmetry Approach

As in the raster line algorithm, we sample at unit intervals and determine the closet pixel position to the specified circle path at each step. For a given radius and screen center position, we can first set up our algorithm to calculate pixel positions around a circle path centered at the coordinate origin (0,0). Then each calculated position is moved to its proper screen position by adding to and to . Along the circle section from to in the first quadrant, the slope of the curve varies from 0 to . Therefore, we can take unit steps in the positive direction over this octant and use a decision parameter to determine which of the two possible positions is closer to the circle path at each step. Positions in the other seven octants are then obtained by symmetry.

To apply the midpoint method, we define a circle function:

(2.4)

Any point on the boundary of the circle with radius satisfies the equation. If the point is in the interior of the circle, the circle function is negative. And if the point is outside the circle, the circle function is positive. To summarize, the relative position of any point can be determined by checking the sign of the circle function.

(2.5)

The circle-function tests in equation (2.5) are performed for the mid-positions between pixels near the circle path at each sampling step.

The midpoint between the two candidate pixels at sampling position is Assuming we have just plotted the pixel at we next need to determine whether the pixel at position or the one at position is closer to the circle. Our decision parameter is the circle function Equation (2.4) evaluated at the midpoint between these two pixels:

(2.6)

If , this midpoint is inside the circle and the pixel on scan line is closer to the circle boundary. Otherwise, the midpoint is outside or on the circle boundary, and we select the pixel on scan line .

Successive decision parameters are obtained using incremental calculations. We obtain a recursive expression for the next decision parameter by evaluating the circle function at sampling position :

(2.7) equation 2.7 – 2.6 gives,



or

(2.8)

Where, is either or , depending on the sing of .

Increments for obtaining are either (if is negative) or . Evaluation of the terms and can also be done incrementally as

(2.9)

At the start position, these two terms have the values and , respectively. The initial decision parameter is obtained by evaluating the circle function at the start position.

or

(2.10)

If the radius r is specified as an integer, we can simply round to(for r an integer) since all increments are integers. (2.11)

The midpoint method calculates pixel positions along the circumference of a circle using integer additions and subtractions, assuming that the circle parameters are specified in integer screen coordinate as follows:

|  |
| --- |
| **Algorithm : Proximity Detection Algorithm** |
| 1. Input radius and circle center, and obtain the first point in the circumference of a circle centered on the origin as  2. Calculate the initial value of the decision parameter as  3. At each position, starting at , perform the following test: If , the next point along the circle centered on is and  Otherwise, the next point along the circle is and  where and .  4. Determine symmetry points in the other seven octants.  5. Move each calculated pixel position onto the circular path centered on and plot the coordinate values:  ,  6. Repeat steps 3 through 5 until . |

Figure 2.6 Proximity Detection Algorithm with Symmetry Approach

The proximity detection algorithm in figure 2.6, first inputs user desired location range denotes as *r* and current location point(lat, long) denotes as (x, y) and then obtain the first surrounding area point of location circle range centered on the origin as (0, r). Then, the algorithm calculates the boundary area decision parameter value denotes as 5/4-r. The algorithm calculates the surrounding location range points based on boundary area decision parameter. Finally, the algorithm moves each calculated location points onto the circular path centered on user current location point. In this algorithm, the shape of the location range is circle shape. This algorithm can be reduced computation time by considering the symmetry of circles. The shape of the circle is similar in each quadrant. The algorithm can generate the circle section in the second quadrant of the XY plane by noting that the two circle sections are symmetric with respect to the Y axis. And circle sections in the third and fourth quadrants can be obtained from sections in the first and second quadrants by considering symmetry about X axis. The system can take this one step further and note that there is also symmetry between octants. Circle sections in adjacent octants within one quadrant are symmetry with respect to the 45 degree line dividing the two octants. Taking advantage of the circle symmetry in this way, we can generate all points positions around a circle by calculating only the points within the sector from x=0 to x=y. After calculating all users desired surrounding range distance points, the system extracts requested services within surrounding range. The step by step process flow of the symmetry-based location range calculation is presented in figure 2.7.



Figure 2.7 Process Flow of the Symmetry based Location Range Calculation.

There are altogether five steps are as follow;

1. Step 1: Requests the services within specifying location range.
2. Step 2: Calculates boundary value of the specified location range.
3. Step 3: Calculates location points in the first octant of first quadrant.
4. Step 4: Uses symmetry approach to obtain all location points.
5. Step 5: Produces all location points around and within location range.

The point positions along a circle path within 10 Km are calculated using proximity detection algorithm and the symmetry positions in the first quadrant is shown in figure 2.8.

r = 10

p0 = 1 – r = -9 (if r is integer round p0 = 5/4 – r to integer)

Initial point (x0, y0) = (0, 10)

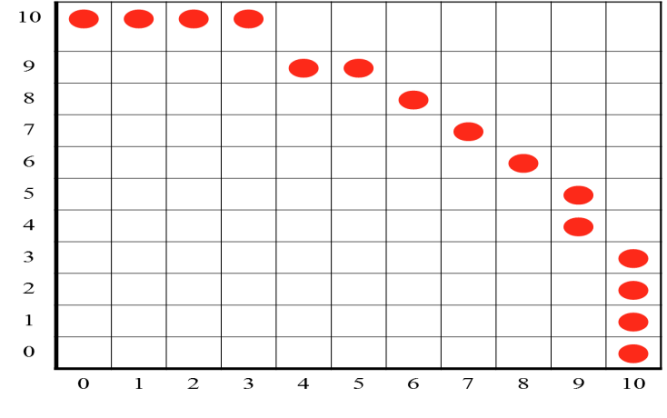
 

Figure 2.8 Calculated Point Positions (solid circle) along a Circle Path with r=10Km.

Once the surrounding area is determined, an information request is issued to the database server to retrieve the area’s requested services. By extracting only the services within a certain range the system are allowing for the possibility that database server could make use of spatial data structures in order to find the relevant data set. When this data set is returned to the LBS application, a filtering step takes place to select only those services that are within a pre-determined distance from the mobile user. This algorithm proves to be more efficient compared to an exhaustive computation of distance over all available services. Figure 2.9 shows requested range and the relevant information area within predefined location range.

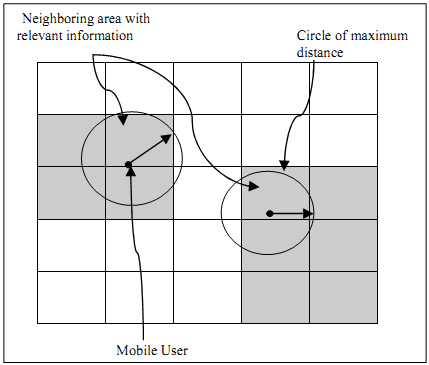


Figure 2.9 Selection of relevant area within specified location range.

## 2.9.3 Location Range Calculation using Distance-based Approach

A range query is a common database operation that retrieves all records where some value between an upper and lower boundary. In this section the range query with distance based approach is used to find the particular location within that range. The distance based range queries are used to answer the query within the specified region.

With the growing popularity of GPS enabled mobile devices and the advances in wireless technology, the efficient processing of continuous range queries, which is defined as retrieving the information of moving objects inside a user-defined region and continuously monitoring the change of query results in this region over a certain time period, has been of increasing interest. Continuous range query processing is very important due to its broad application base. The range queries are important type of query, used to find the objects within the range. In case of distance based range queries, consider an object O, a positive value called range r, and a query point q, . The is defined as the distance between the object O and a query point q, that is less than or equal to range r. Consider the continuous monitoring of moving range queries over static data objects. The following real time example will clearly explain this scenario. Suppose a family travelling by car. They need to reach their final destination by a certain time, while they want to continuously monitor restaurants within 10 km of their current location.

|  |
| --- |
| **Algorithm : Distance-based Location Range Algorithm** |
| 1. Initialization: r = distance range;  q = user current location point ;  l = location points in all database records in R;  M= Ø;  2. for j = 0 to l do  3. d=acos[  4. If then  5. ;  6. else  7. return 0;  8. endif  9. endfor  10. return ; |

Figure 2.10 Location range algorithm with distance-based approach

**2.10 Extracting Optimal Answer Using Tree Matching**

Several algorithms have been proposed to address the problem of finding the minimum set of operations (i.e., the one with the minimum cost) to match one tree with another. All the formulations have complexities above quadratic. In [37], a solution based on dynamic programming is presented with the complexity of *O (n1n2h1h2)*, where *n1* and *n2* are the sizes of the trees and *h1* and *h2* are the heights of the trees. In [38], Yang et al. proposed a simple tree matching algorithm, which makes use of dynamic programming to calculate the maximum number of node-pair between two trees. This algorithm does not allow cross-layer matching and nodes replacement.

**2.10.1 Preference Tree Matching Algorithm**

Tree matching algorithm call the simple tree matching to get optimal answers based on user preference query. Finally, the algorithm shows optimal k-answers from the result set. Let P=RP:<P1,P2,…,Pm> and D=RD:<D1,D2,….,DL> be preference query tree and database record tree, where RP and RD are the roots of P and D respectively in figure 2.11.

|  |
| --- |
| **Algorithm: Preference Matching Algorithm** |
| 1. Initialization : m = number of preferences  in P;  l = number of database tree  records in D;  R= Ø;  2. If m does not contain in l then  3. return 0;  4. else  5. for j = 1 to l do  6. r = SimpleTreeMatching ( P, )  7. R{r}= R{r} U  8. endfor  9. return R{r};  10. endif |

Figure 2.11 Preference matching algorithm

The step-by-step matching and filtering the optimal available services with user preferences is shown in figure 2.12.

Sorting Results with Similarity

Return Optimal Top-k answers

Matching Preferences with Database records

User Preferences

Figure 2.12 Process flow of the preference matching and filtering

There are altogether four steps are as follow;

1. Step 1: Input user preferences of smartphone user.
2. Step 2: Match preferences of user query with database records to calculate similarity values.
3. Step 3: Order similarity values from preference matching.
4. Step 4: Filter optimal top-k answers to respond users.

## 2.10.2 Simple Tree Matching Algorithm

Simple tree matching algorithm calculates the similarity by using dynamic programming to produce the greatest matching, the algorithm complexity is *O(mn),* where *m* and *n* are the size of *A* and *B*. The specific algorithm [38] is shown in figure 2.13.

Let’s *A* and *B* are two trees, *i* and *j* for two nodes in *A* and *B*, respectively. Following [38], a mapping, *M*, between two trees as follow: For any node pair *( i , j)*∈ *M*( (neither *i* nor *j* is root), let (*parent(i),parent(j)*) ∈ *M* .The maximum matching of two trees is the matching which has the maximum number of matching pairs.

Let A=RA :< A1,…..,Am> and B=RB :< B1,….,Bn> be two trees, where RA and RB are the root of A and B, respectively; Ai and Bj be the ith and jth node of the first-level subtrees of A and B, respectively. If the symbols of RA and RB are the same, then the maximum matching of A and B (i.e. W (A,B) is M ( < A1,A2,….,Am>, < B1,B2,…,Bn> )+1 , where M ( < A1,A2,..,Am>, < B1,B2,..,Bn> ) = max (M ( < A1,A2,.., Am-1>, < B1,B2,..Bn-1 >) +W( Am, Bn), M ( < A1,A2,…,Am-1 >, <B1,B2,…,Bn>), M ( < A1, A2,…,Am>, < B1, B2, …, Bn-1>)). If RA and RB contain distinct symbols, then W (A, B) =0.

|  |
| --- |
| **Algorithm : SimpleTreeMatching(A,B)** |
| 1. if the root of the two trees A and B contain distinct symbols then 2. return 0; 3. else m= the number of first-level subtrees of A; 4. n= the number of first-level substrees of B; 5. Initialization: M[i,0] = 0 for i = 0,………..,m; 6. M[0,j] = 0 for j = 0,……......,n; 7. for i = 1 to m do 8. for j = 1 to n do 9. M[i,j]=max (M[i,j-1], M[i-1,j], M[i-1,j-1] +W[i,j]), 10. where W[i,j]= SimpleTreeMatching (Ai,Bj) 11. endfor 12. endfor 13. return (M[m,n]+1); 14. Endif |

Figure 2.13 Simple tree matching algorithm

First, the roots of two trees are compared. If they contain distinct symbols, the two trees totally do not match. If they contain the same symbols, the algorithm recursively to find the maximum matching between the first-level subtrees of the two trees, and save the matching in the *W* matrix. Then, we calculate the value of matrix *M* according to *W*. To better understand the algorithm, here presents a running example (the maximum matching between two trees show in figures 2.14 (a) and (b) to explain the algorithm implementation.

N1

N10

N2

N3

N4

N9

N8

N7

N6

N5

N11

N13

N12

(a)

N14

N15

N17

N16

N18

N19

N20

N21

N22

N23

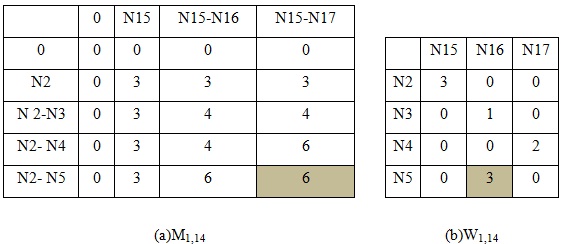
N24

N25

(b)

Figure 2.14 Sample trees for simple tree matching

First, the roots, nodes N1 and N14 are compared. Since they contain identical symbol a, M1,14[4,3]+1, the maximum matching value of A and B is returned. M1,14 is calculated based on W1,14 matrix. Each element W1,14[i,j] of W1,14 is the maximum matching of the ith and jth first-level subtrees of A and B, respectively. W1,14[i,j] is recursive calculations based on its M matrix. For exampleW1,14[5,16] is calculated recursively by building matrixes (e)-(h), and all the relevant elements have been marked with a shadow. The 0th row and 0th column of the matrix M are initialized to 0. The construction of matrix Mx,y and Wx,y , are shown in figure 2.15, where x and y represent the matched node of A and B, respectively.



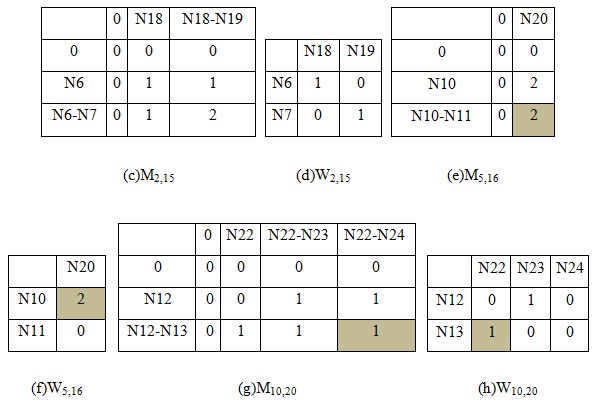


Figure 2.15 (a) M matrix for the first-level subtrees of N1 and N14; (b) W matrix for the firstlevel subtrees of N1 and N14; (c)-(h) M matrix and W matrix for the lower-levels subtrees.

## 2.11 Chapter Summary

This chapter presents location-based services applications developed by researchers with different methods and techniques. The wireless network through the user’s query and LBS provider’s response can be communicated between mobile device and server. The LBS support spatial databases such as MySQL, Oracle and their service components researches that are used to resolve the query and provide the tailored response to the user are provided. Since LBS are heterogeneous technology, they comprise of a number of sub-architectures to do with wireless communication, web services, the Internet, GPS and so on. This chapter concludes the discussion about related technologies of location-based services in the first section. One essential aspect in LBS is fixing the location of users through their mobile device. Location of users is regarded as the spatial context in LBS applications. Such location data are often used as a key in real applications, from which the overall context informs mobile applications. In order to provide tailored data and information services to users in mobile situations, most LBS applications have some level of context awareness. Important context information can include user location and the surrounding environment, user situations during activities, time of day and date, and the technologies involved (devices, networks and systems). Web services technologies have advanced the development of Internet-based applications by facilitating the integration and interoperability of services provided by different organizations.

In the second section, user location range calculations are presented with proximity detection algorithm with symmetry approach and distance based location range calculation to calculate user specified location range based on smartphone user current location. Proximity detection algorithm with symmetry approach is more efficient computation time than distance based location range calculation. And also, there express the usage of preference tree matching and simple tree matching algorithms in details for matching with representative field databases.

# CHAPTER 3

# design and implementation OF LOCATION-BASED SERVICES (Excellentsercvice) system

People can receive various types of data services via their portable devices due to the recent advancements of Wi-Fi technologies. Android smart phone is very popular around the world and supports users various service components. Location-based Service (LBS) is recognized as a promising technology in the upcoming ubiquitous era. LBS become more attractive service area in the smart phone technology in mobile computing. LBS are a platform that provides information services based on the current or a known location. Context and preference aware location-based services are more attractive among smart phone users. LBS web services effectively recognize and respond the location specific web resources to the mobile user based on their geographical position. It provides a user with contents customized by the user’s current location which are retrieved from a database.

Towards the goal of realizing location-based services, the system architecture of location-based services system (ExcellentService, for short) that delivers personalized services to its customers based on the user desired location and preference query is presented. ExcellentService will replace the traditional scheme of “one size fits all” of existing location-aware database systems. Instead, ExcellentService tailors its functionalities and services based on the preferences of each customer. The user requests the available services via mobile device as his/her predefined location range and preferences. Before reporting the query answer, ExcellentService will check the user preferences and desired location. The system maps user preference query and extracts the optimal answers related to user preference query. Then, the system returns the most relevance query answers to the users. Thus the proposed system would not report expensive hotels, hotels without user interest amenities, or hotels with conflicting dietary offerings and hotels without desired location.

## 3.1 ExcellentService Conceptual Design

The conceptual design of the ExcellentService is shown in figure 3.1. Firstly, mobile phone users request the available services with current location from their smart phones with his / her preference queries. The server side gets user request from ExcellentService layer. The location selection module calculates location range based on user specified location range and extracts the available services within location range. Then, extracted services are matched and filtered based on user preferences and returns optimal top *k* answers to the smart phone users. This conceptual design is divided into front-end services and back-end services.

We will be using the internet for our main source of usage, which will send the users data and the service of process from mobile to the server and then share their requested information (queries) back to the users. ExcellentService consists of four parts: the repository, the mobile client, web client and a map service of Google Map API, shown in figure 3.1.

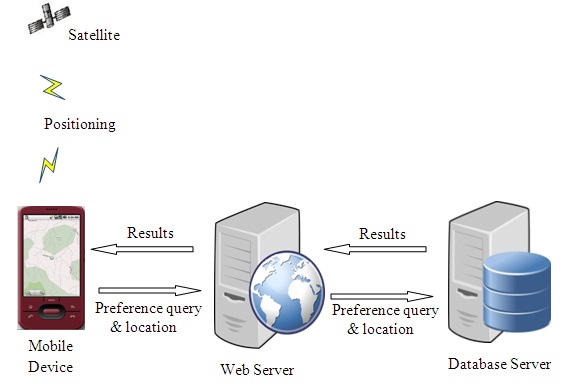


Figure 3.1 Conceptual Design of (ExcellentService) System

Firstly, mobile phone users request the available services with current location from their smart phones with his / her preference queries. The server side gets user request from LBS layer. The LBS calculates location range based on user specified location range and extracts the available services within location range. Then, extracted services are matched and filter based on user preferences and sorted the services based on matching preference level and return optimal top *k* answers to the smart phone users. The smart phone user gets the suitable answers with the specification of their service requirements. The system has three components which are expressing details in the following.

## 3.2 ExcellentService Android Application

The use case diagram for ExcellentService android application is shown in figure 3.2.



Figure 3.2 Use Case diagram for ExcellentService android application.

ExcellentService is developed on the Android platform. The Android Smart phone user can use ExcellentService application directly from their android phone. The users have to register their personal information in registration form of ExcellentService. After registration has succeeded, the mobile user can log into ExcellentService for their purposes. The android phone user can send request service with their preference query and view details of requested service. The user can click to visualize information details on the map and on the timeline website. The mobile phone users can insert category and their current location from anywhere only there needs availing of Internet in order to get the correct current location.

## 3.3 LBS Web Services

The use case diagram for LBS web service is shown in figure 3.3.



Figure 3.3 Use Case diagram for LBS Web Service.

Firstly, the web service receives data from the android application with user specified location range. The LBS web service calculates the boundary value of the specified location range and the location points of in the first octant of first quadrant. Then, LBS web service uses symmetry approach to obtain all location points and produces all location point around and with user specified location range. Symmetry based proximity detection algorithm is used for above actions. The extracted services are matched preferences of user query with database records to calculate similarity values and filter based on user preferences and sorted the services based on matching preference level and return answers to the smart phone users. Preference matching algorithm is used for these actions.

## 3.4 ES-DB Database

The ES-DB database schema is shown in figure 3.4.



Figure 3.4 ES-DB database schema.

The database name, ES-DB, for software prototype ExcellentService is created in MySQL database. The ES-DB database is created by eight tables. This diagram is the standard entity relationship model, widely used in general-purpose design. Every entity has a unique identifier attribute, named ID, which is shown in figure 3.4. In our system, category table is created for storing available services such as hotel, bank, restaurant and hospital provided by ExcellentService. In this table consists of ID and name for each service. The same features (such as name, address, phoneno, url, rating, latitude, longitude etc) of these services (such as hotel, bank, restaurant and hospital) are stored in the information table. The hotel, bank, restaurant and hospital table are created for storing the difference features. The difference features are pricehigh, pricelow, cuisine etc. The personal information of users can be stored in the user table. The personal information are name, email, password, phoneno, DOB, profession, city. In order to know the user’s use case processes of when and where, there the history table is created.



Figure 3.5 Structure of ExcellentService software prototype

Figure 3.5 describes the structure of ExcellentService software prototype. Most of the Android platform based applications are created with layers activities. In our proposed system there are altogether three basic activities but only two of them are mainly in functional and the last one GPS on activity is system structure purpose only.

## 3.5 Design and Implementation of Proposed System

Implementation of our proposed ExcellentService System entirely based on system process flow diagram which described in figure 3.6. Only when WiFi/GPS on, module the system will connect with ExcellentServiceDB. Incoming user data will be directly checked by ExcellentServiceDB and that user will be arrived to Welcome Page only when input data are correct. If not the user needs to register. In the Welcome Page the user can see welcome screen and available services. There, the user can search four kinds of objects such as Hotel, Restaurant, Bank and Hospital. The user’s search query is controlled with five parameters: Name, Range, Rating, Price H/L and Cuisine. After the user has filled with appropriate parameters and clicked on search button then available searched lists will be displayed. There, the user can manually select each of available searched lists in order to get details of them. Also, user can get the selected object’s location by pressing Map View button.



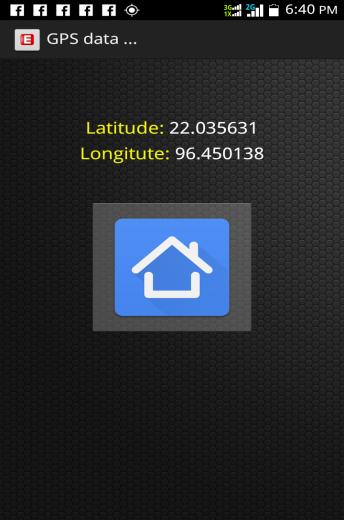
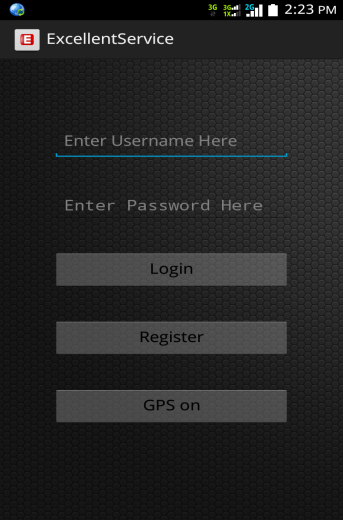
Figure 3.6 ExcellentService System Process Flow Diagram

Our system consists of LBS web service, MySQL database and ExcellentService client on android smartphone device. In this research we had implemented connection between PHP (server) and Android (client) using HTTP and JSON. The data from android goes to webserver (PHP) to database server (MySQL). PHP is used here because of the interaction it can offer with the databases. On Android, HTTP protocol is used to connect with the webserver (PHP). JSON (JavaScript Object Notation) is a lightweight text-based open standard designed for human-readable data interchange and it is used in this application to send data from Android device to PHP Script.

When the application is executed, it connects the device to the PHP script on the server. PHP script fetches the response data which is encoded to JSON format and then sent back to the device. The data is parsed and displayed according to the requirement.

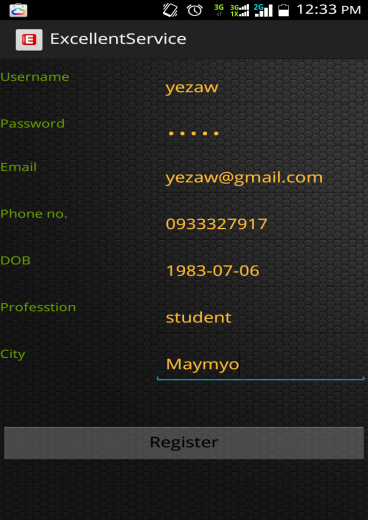
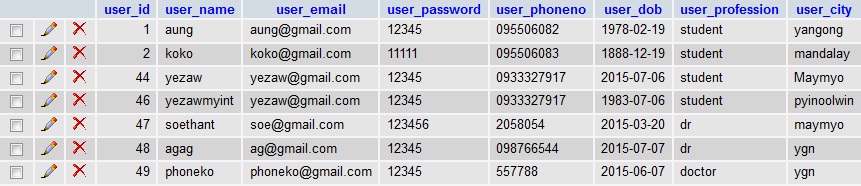
1. **Login Activity:** This class enables the user to login to the application so as to use the application. The user can login by entering the username and password. If the user enters the wrong credentials he/she is not allowed to login.
2. **Register Activity:** This class lets the user to register with the application. The user has to register by giving a username, a valid password and an email address etc. Each registered user will have a unique username. All the user details are stored in a database located on a server. This enables the user to login from any device as the data is centrally located on a server. Certain constraints have been set on the password and the user name is put for a valid format.

The Android Smart phone user will launch ExcellentService application in their android home screen. ExcellentService will show login screen to find available services provided by ExcellentService. Firstly, the smart phone user needs to click the GPS On button and opens the GPS and WiFi in his/her smart phone. If smart phone user is not registered, they must register their personal information with ExcellentService registration. After registration is successful, the mobile user signs in to use services in ExcellentService. Figure 3.7, 3.8 and 3.9 show Main UI screen, Home Screen, the registration screen and Login screen of ExcellentService. Figure 3.8 (b) shows the database server which is applied with MySQL for our system. There, all registered users list can be seen.

(a) (b)

Figure 3.7 (a) Main UI screen (b) Home screen

(a) (b)

Figure 3.8 (a) Registration screen (b) Registered users list in database

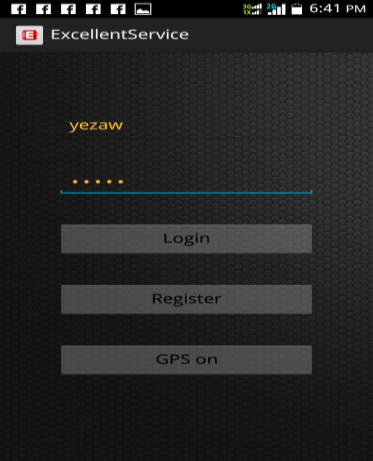


Figure 3.9 Login screen

After smart phone user has signed in ExcellentService, the main user interface of ExcellentService will show. The smart phone user can use the available service finder of ExcellentService to find their requirements. Figure 3.10 shows welcome screen of ExcellentService. In our system, there include four categories such as Hotel, Hospital, Restaurant and Bank as shown in that figure.

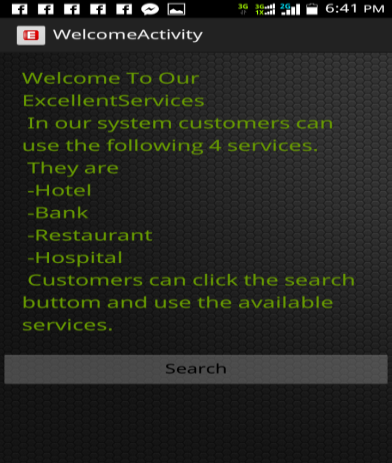


Figure 3.10 Welcome screen

The mobile user can search the object with his/her preference query in ExcellentService as shown in figure 3.11 (a). Then he/she will get the list of objects with his/her preference query result and its execution time using Distance based Algorithm as shown in figure 3.11 (b). And then he/she can click the each object name to get the details data. Figure 3.11 (a), (b) and (c) show searching of hotel and its relevant search result lists with execution time by using Distance based Approach Algorithm and details of them.

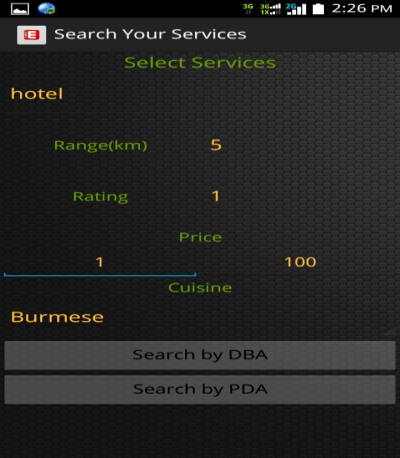
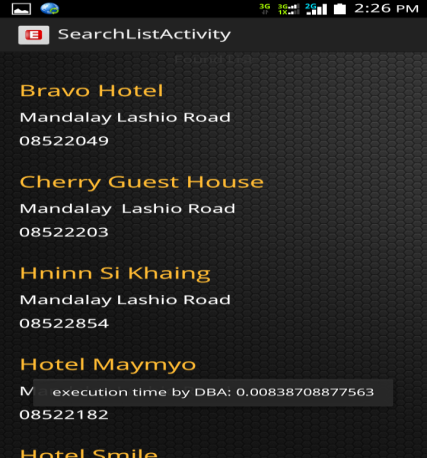
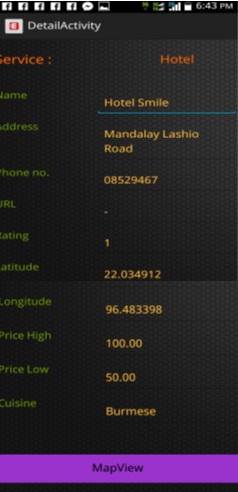
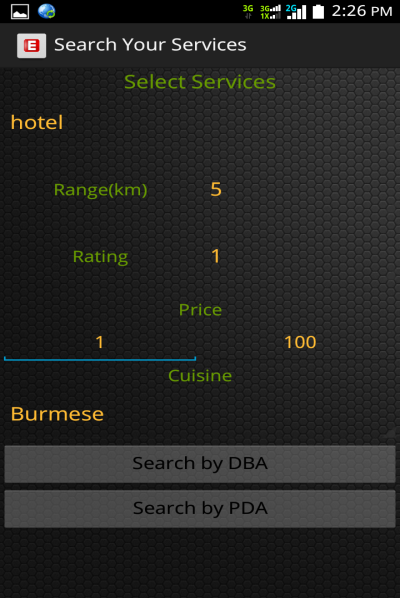
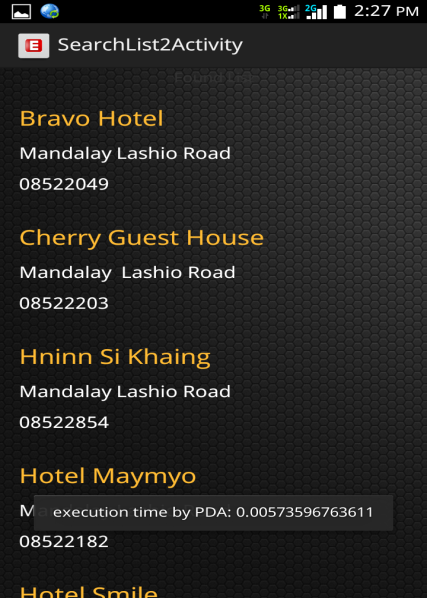
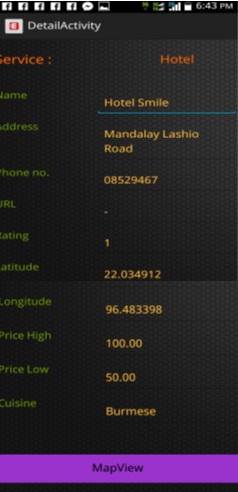
   (a) (b) (c)

Figure 3.11 (a) Search hotel (b) Search result list hotel (c) Detail of hotel

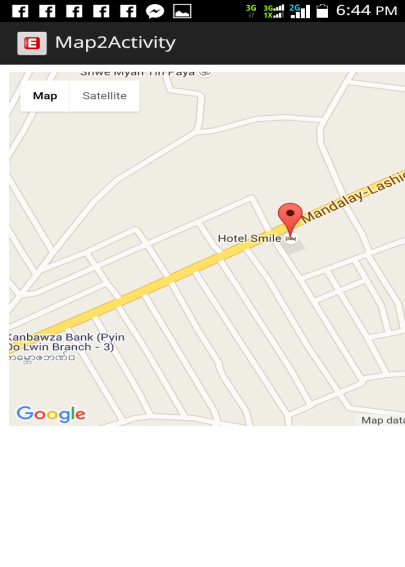
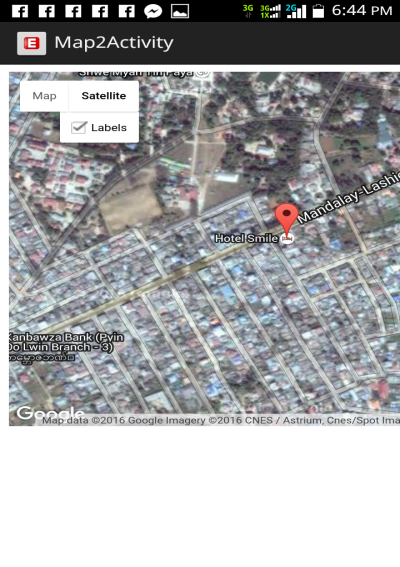
Also, in figure 3.12 (a), (b) and (c) show searching of hotel and its relevant search result lists with execution time by using Proximity Detection Algorithm and details of them.

(a) (b) (c)

Figure 3.12 (a) Search hotel (b) Search result list hotel (c) Detail of hotel

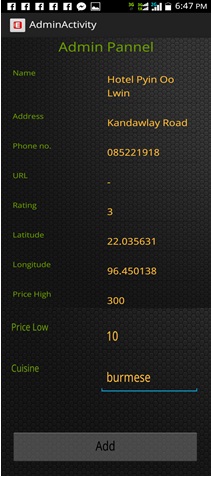
And then the user can click map view button and see the object location point on the map and also in satellite view. Figure 3.13 (a) and (b) show the map and satellite views of the object location point which the user’s selected.

(a) (b)

Figure 3.13 (a) Map view (b) Satellite view

The administrator can add the objects (such as hotel, restaurant, bank and hospital) in the ExcellentService to the web server with the object`s location as shown in figure 3.14 (a). Then the data will be entered to the related table as shown in figure (example for hotel) 3.14 (b) and (c).

(a) (b) (c)

Figure 3.14 (a) Adding Object (b) Object Data in Information Table (c) Object Data in Table

## 3.6 Chapter Summary

This chapter presents ExcellentService, which is the implementation of location-based services framework on android platform. Server-side integration of the ExcellentService is highlighted with listing data. The server returns the requested services with RESTful services. We had implemented connection between PHP (server) and Android (client) using HTTP and JSON. The data from android goes to webserver (PHP) to database server (MySQL). JSON (JavaScript Object Notation) is a lightweight text-based open standard designed for human-readable data interchange and it is used in this application to send data from Android device to PHP Script. And also, the implementation outcome results of proposed system are described in details.

**CHAPTER 4**

**SYSTEM EVALUATION OF LBS**

In order to show the capability of the prototype implementation, an experimental case study is carried out. With this study, it is aimed to show the advantage of the prototype implementation compared to the traditional location-based web services system. It is also aimed to show the comparative advantages implementation modules. This chapter presents performance evaluation of the Efficient Location-Based Services system (ExcellentService) by means of an experiment.

**4.1 Experimental Setup**

We implemented and tested system (ExcellentService) with sample Hotel finder scenario. Our system consists of a location-based services with Web Server and ExcellentService client on android smartphone device. Table 4.1 illustrates hardware and software specifications of our system (ExcellentService). The analysis of the LBS framework mainly focuses on mobile peer, since the user experience is more important. ExcellentService is mainly deployed on Desktop/Laptop computers with have extensive resources including powerful processing and most likely persistent broadband connectivity. The performance of ExcellentService is related to the design of services. This experiment primarily analyzed the performance on mobile information delivery to show the usefulness of a proactive application.

Table 4.1 Hardware and Software Specification of Experimental Test Bed

|  |  |  |
| --- | --- | --- |
| **Components** | **Specification** | |
| **Name** | **Description** |
| **LBS Database Server** | Central Processing Unit (CPU) | Intel (R) Core (TM) i3 Processor 3227U CPU@1.90 GHz) |
| Memory (RAM) | 4GB |
| Operating System | Windows 7 Ultimate |
| **Web Server** | Main Domain Address: www.mmgreenhackers.com | <https://natmout.com:2083/cpsess2675090552/>  frontend/x3/index.html |
| **LBS Mobile Client** | Mobile Platform | Android Platform |
| Platform Version | Android 4.1 (JellyBean) |

We conducted a series of experiments to evaluate the performance of the proposed LBS, under various system conditions. All of the experiments were implemented in android and Java servlet on a Intel® Core™ i3 Processor CPU@ 3227U (1.90 GHz) machine with 4GB memory running on windows7 operating system. The data sets queried ranges from 20 to 120 records. There are 20, 40, 60, 80, 100 and 120 records at each query. Each query was done 20 times and the average recorded. We performed experiments about the connection establishment time, processing time and service reception time of our ExcellentService system with various criteria. All these experiments relied on the connection speed with our hosting server.

**4.2 Evaluation of Computational Processing Time**

In this experiment, we evaluated processing time of the LBS framework implemented by ExcellentService software prototype. We conducted this experiment with various location ranges and number of records. Processing time tests to determine end to end timing of various time critical business processes and transactions, while the system is under low load, but with a production sized database.

(4.1)

where,

Tp= Processing Time

Ts= Start Time

Te= End Time

Table 4.2 Execution Times of PDA and DBA

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Location Range (km) | 1 km | 3km | 5km | 7km | 10km |
| Execution Time PDA (ms) | 3.96 | 4.34 | 4.93 | 5.12 | 6.41 |
| Execution Time DBA (ms) | 5.71 | 6.91 | 7.15 | 7.65 | 8.56 |

As shown on this table 4.2 and figure 4.1, we could analyzed that the larger location range area the processing time of PDA is the lesser than DBA by comparing in a range of 1km to 10km. The processing time of PDA is nearly one-fourth time faster than DBA, according to these experimental results.

Figure 4.1 Processing Time Comparison between DBA and PDA

Table 4.3 and 4.4 illustrate relevant product records comparison between Proximity Detection Approach (PDA) and Distance Based Approach (DBA) with data amount and location ranges. In figures 4.2 and 4.3 the number of records amount for PDA and DBA is slightly increased when the location ranges is greater. And there, we can point out that, the greater the location range, the more accurate records in both of them. These experimental results are basically relying on the user’s current location point.

Table 4.3 Number of records (DBA) based on Location Ranges

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Number of records (DBA) | | | | | | | | | | |
| Location Range (km) | 1 km | 2km | 3km | 4km | 5km | 6km | 7km | 8km | 9km | 10km |
| Hotel | 5 | 16 | 21 | 22 | 22 | 23 | 24 | 24 | 24 | 24 |
| Restaurant | 1 | 13 | 16 | 17 | 17 | 17 | 17 | 17 | 17 | 17 |
| Bank | 0 | 5 | 6 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| Hospital | 0 | 2 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |

Table 4.4 Number of records (PDA) based on Location Ranges

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Number of records (PDA) | | | | | | | | | | |
| Location Range (km) | 1 km | 2km | 3km | 4km | 5km | 6km | 7km | 8km | 9km | 10km |
| Hotel | 2 | 15 | 21 | 22 | 22 | 23 | 24 | 24 | 24 | 24 |
| Restaurant | 1 | 11 | 15 | 16 | 17 | 17 | 17 | 17 | 17 | 17 |
| Bank | 0 | 5 | 6 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| Hospital | 0 | 2 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |

Figure 4.2 Number of Records Amount for DBA

Figure 4.3 Number of Records Amount for PDA

In table 4.5 and 4.6 show processing time comparison between DBA and PDA with location ranges. In figure 4.4 and 4.5 represent the processing time of the two approaches is slightly increased when the requested location range is greater. Even though the processing time is increased because of data amount and location range is greater, the processing time of PDA is nearly one-fourth times faster than the processing time of DBA.

Table 4.5 Execution Time of DBA with location range

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Execution Time of DBA (ms) | | | | | |
| Location Range (km) | 1 km | 3km | 5km | 7km | 10km |
| Hotel | 4.61 | 6.39 | 8.45 | 9.83 | 12.42 |
| Restaurant | 4.82 | 6.18 | 7.56 | 7.92 | 11.91 |
| Bank | 3.92 | 6.02 | 6.58 | 7.56 | 9.25 |
| Hospital | 3.68 | 5.93 | 6.92 | 7.25 | 9.13 |

Table 4.6 Execution Time of PDA with location range

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Execution Time of PDA (ms) | | | | | |
| Location Range (km) | 1 km | 3km | 5km | 7km | 10km |
| Hotel | 3.32 | 4.85 | 6.25 | 8.45 | 10.81 |
| Restaurant | 2.96 | 4.93 | 6.74 | 7.02 | 9.38 |
| Bank | 3.08 | 4.61 | 5.19 | 5.92 | 8.16 |
| Hospital | 2.54 | 4.12 | 5.01 | 5.86 | 8.01 |

Figure 4.4 Processing Time of DBA with location range

Figure 4.5 Processing Time of PDA with location range

In table 4.7, 4.8 and figure 4.6, 4.7 show the effect of concurrent requests on processing time in DBA and PDA. When many user requests arrived in ExcellentService, the processing time of the system increases. But within short range location the processing time is not different mostly even though number of current users increase.

Table 4.7 Effect of concurrent requests in DBA

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Execution Time of DBA (ms) | | | | | |
| Location Range (km) | 1 km | 3km | 5km | 7km | 10km |
| 1 User | 5.71 | 6.91 | 7.15 | 7.65 | 8.56 |
| 2 Users | 6.59 | 7.36 | 7.83 | 8.62 | 8.84 |
| 3 Users | 7.67 | 8.42 | 9.17 | 9.72 | 10.53 |
| 4 Users | 9.81 | 10.31 | 11.02 | 11.58 | 13.72 |
| 5 Users | 11.98 | 12.29 | 13.87 | 14.11 | 14.98 |

Figure 4.6 Effect of Concurrent user Requests in DBA

Table 4.8 Effect of Concurrent user Requests in PDA

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Execution Time of PDA (ms) | | | | | |
| Location Range (km) | 1 km | 3km | 5km | 7km | 10km |
| 1 User | 3.96 | 4.34 | 4.93 | 5.12 | 6.41 |
| 2 Users | 4.28 | 5.35 | 5.91 | 6.33 | 6.79 |
| 3 Users | 6.84 | 7.18 | 7.52 | 7.93 | 8.14 |
| 4 Users | 8.68 | 9.13 | 9.52 | 10.71 | 12.21 |
| 5 Users | 10.85 | 11.82 | 13.92 | 14.15 | 14.67 |

Figure 4.7 Effects of Concurrent Requests in PDA

**4.3 Evaluation of Computational Responsiveness Time**

We evaluated responsiveness time of ExcellentService to the user smartphone in these experiments. In these experiments, we conducted experiments with ExcellentService by means of various location ranges and comparison of preferences user’s requests in study. Response time is the length of the time between an indication of the end of an inquiry and the display of the result of the response at a user terminal.

(4.2)

Where,

Tresp= Response Time

Treq= Request Time

(Te-Ts)= Processing Time

Trep= Reply Time

Figure 4.8 Response Time Comparisons between DBA and PDA with Location Ranges

Figure 4.8 shows response time comparison between DBA and PDA with location ranges. According to this figure, even though the response time increases based on location range, response time of PDA is nearly one-fourth times faster than DBA.

**4.4 Chapter Summary**

This chapter has experimentally evaluated the proposed system framework implemented by ExcellentService software prototype. The experiments were divided into four parts: evaluation of computational processing time, evaluation of computational responsiveness time, concurrent user’s measurement and records amount results. The results show that the performance of message delivering and battery consumption is acceptable for developing information services on mobile devices and it is entirely depend on the connection speed of our host server. In addition, the system selects suitable algorithms and approaches that are jointly optimized to achieve high availability and efficiency. Moreover, user satisfaction also brought the opportunity of getting feedback about the system functionalities and performance.

# CHAPTER 5

# CONCLUSION AND FUTURE WORK

In this thesis, the design and development location-based service framework and its software prototype, ExcellentService, are implemented. The general idea is to provide smartphone users the available location-based services based on their specified location range and preferences. In the preceding chapters, the specific improvement of the proposed framework in various domain areas has been described. In this final chapter, the proposed framework is summarized and the study of this work is completed by drawing conclusions and giving directions of future research.

**5.1 Discussion**

Obviously, LBS should bring enormous business opportunities for each party involved in this field of location-based framework. These players include hardware and software vendors, content and on line service providers, wireless network and infrastructure providers, wireless handset vendors and map portal sites. Only a common specification and agreements among these partners do ensure the consumer satisfaction and fast deployment of its services. Investors are now considering what business plans could support revenue-raising LBS. Sources of revenue for service providers, for example, may include subscription fees for LBS, advertising push services, connection fees, fees for content, transaction fees or margins on the price of products ordered, fees for the quality of the offered service. In some cases, such as for emergency 911 services, the operators may collect revenue to pay for the services through regular phone subscription fees.

The majority of LBS require high accuracy at a reasonable cost to optimize the return on investment. The cost of implementing any technique depends on handset device modification, infrastructure modification, maintenance activity, network expansion plans, accuracy and high performance, etc. All this cost should be refunded by the subscription fees to push services and most of pull services from the client and the application service providers too.

In order to improve LBS systems to meet market expectations from a testing point of view, the following primary issues have to be considered for a better RON (Return on Investment): Functionality, Usability, Performance, Scalability, Security and Privacy, Interoperability, Accuracy, Precision and Quality of Service. In this chapter, we will investigate only the issues implied directly to our research such as scalability, interoperability, quality of service.

**5.1.1 Scalability**

Intelligent choices must be about hardware, software and the design of the applications to handle potential growth in the number of customers and providers for the LBS applications. Load testing should be done to evaluate better the scalability and the performance of LBS and confirm that the system can handle a high volume of simultaneous users and/or transactions while maintaining adequate response times from providers and admin database.

We can reach scalable LBS because scopes may range from small locations to wide-area coverage without any significant effect on the system‘s performance. Response time-out problems typically result from incorrect server application, design and/or database problems or network limitations.

Even lack of resources is a problem such as RAM, disk space, CPU, bandwidth, limited number of sessions. So, we must verify that all servers can operate under extreme conditions. The direct way to undertake load testing on the server is to manually vary the inputs, e.g. number of clients, frequency of requests, and mixture of requests then measure the change in performance. For example, the delay between sending and getting back the appropriate answer are subject to change between the android emulator and the LBS Web Services technology.

We had started our prototype with a LBS provider, in MySQL, using java servlets to access them and Web Services technology with web application to get users requests. The LBS server had implemented two different interfaces. One is used for the users to collect their preferences and one for the provider to build their catalog of services. We confirmed that we had improved scalability with the following issues:

1. XML and web application technology will speed up the HTTP connection to the server.
2. The login ID of the user is enough to be linked to his profile once sent with the initial service request.
3. Provider’s catalog is very important to include all useful metadata about the services handled by each provider. Rather than sending the user‘s request to all available services in LBS server and wait for few appropriate responses because some services did not cover the area or meet user‘s requirements, it would be better to check first the catalog file in the LBS server and contact only the services that can meet the requirements. In that case we can save time and resources.
4. Besides, the administrator can easily input the metadata of any new services via this catalog interface and save the result as XML file and parse it via mobile client.
5. The java code had been prepared to include a class called Connections with JSON driver and servlet to the LBS server.
6. Web Service solution can save lot of time and avoid any human intervention thus increment the numbers of services and users in a dynamic way. These web services use REST for mobile protocols and we had already agreed on their usefulness in the literature at all levels.

Finally, scalability is a very big task to respect. For the time being, we had thought to study it but it is not realistic.

**5.1.2 Interoperability**

Given the many players/operators, providers and manufacturers involved in the business of LBS, there should be interoperability solution adjusted via standards such as XML. Many LBS include several components such as the mobile application itself, web application, map databases providers, user databases, etc. These components cannot only be located on different physical machines but also on different platforms. Some strategic considerations are followed to adjust indirectly the interoperability. These include the range of coverage and scalability of applications, the degree of service quality that can be established and maintained at a reasonable cost; and the careful alignment of the overall technology costs with the types of services that customers will pay for.

To ensure interoperability among all these prototypes, we decided to adopt XML family language as a standard to transmit the useful information.

RESTful Web Service XML stream can be parsed both unified output files and adjusted the related combination towards a unique portrayal from LBS provider. Besides, web services technology, JSON packages, REST are important as well.

Furthermore, the 3-tiers architecture with a mediator database is very useful to tackle the heterogeneity at the application layer in a smooth way, using catalog or mapping/integration tables for example. All of the above can play a role of wrappers or mediator component to the whole system architecture.

**5.1.3 Quality of Service**

It varies based on the type of LBS application. For example, driving directions may require an accuracy of 30 yards, while location-sensitive billing or mobile yellow pages may only need to locate a user within a range of 250 yards. In this system, nearest service finding system requires an accuracy of 1 km range as default and user specified range to locate nearest services within user specified range.

Our platform tested from several different perspectives including functionality, usability, network performance, server capacity, performance, security and privacy, scalability and interoperability. As a matter of fact, our application should offer;

1. High performance: delivering answers in a second
2. Scalable architecture: to support thousands of concurrent users and terabytes of data
3. Reliability: capable of delivering up to 99.999% up-time
4. Up to date information: support the delivery of real-time and dynamic information
5. Mobility: availability from any device and from any location
6. Open platform: support common standards and protocols
7. Security: manage security and privacy services
8. Interoperability: with any platform, technology or CRM (Customer Relationships Management).

As for the radius, it should be chosen by the user in his preferences (10Km for example) or fixed by the administrator based on each type of service. In that case, we can define more precisely the term “nearest” from the user’s point of view. This latter had direct influence on the answer (e.g. nearest restaurants). The radius is different for pedestrian users or drivers.

Finally, users’ preferences are necessary for delivering adequate answer and thus increasing the quality of LBS application.

**5.2 Conclusion**

Providing dynamic location-based service and improving the information retrieve accuracy especially in the limited mobile screen have become the important research areas in the development of LBS. Current location-based applications only provide services based only on the location context while ignoring various forms of user preferences. In this thesis, we have discussed the rigidness in current location-based applications that provide services based only on the location context while ignoring various forms of user preferences and surrounding context. To overcome such rigidness, we introduced the system architecture of a Location-based Services that delivers personalized services to its customers based on the surrounding context. LBS tailors its functionalities and services based on not only user desired location range but also the preferences of each user. The system is evaluated and analyzed based on processing and response time with DBA and PDA algorithms.

This analysis is needed to select the most appropriate implementation technology that suits limited mobile environment capabilities and to facilitate providing mobile Web Services in a continuous light weight processing manner. According to evaluation results, we observed that PDA can compute specified location ranges nearly one-fourth times faster than distance based approach. The system responds users request accurately by filtering with preference matching and filtering algorithm. The system provides smart phone users with reasonable performance and suitable services to meet user’s satisfaction.

## 5.3 Limitations of Proposed Research

The first disadvantage of our research is that, it is entirely rely on the connection speed of our Myanmar’s Internet connection speed. If the connection is not in the best condition, the GPS location point’s results will be worst. As we mention above chapters, we are rely on the Google Map services, so our process phones must be suitable with Google Services in order to get the latest map information. Our proposed research cannot give the direct optimized way (road map) for the user as it is not a navigation system. For the GPS location point, the entire usage phones must have the appropriate GPS service also.

**5.4 Further Extension**

The system can be extended for push-based location-based services to send notification alert to smart phone users within specified location range. Geofencing is the next natural step of LBS. A geofence is a virtual area defined by a set of boundaries, like a leisure center area or neighborhood boundaries. This way, ExcellentService takes into account geo-content, which refers to the context that is relevant to a particular area. The location service provider uses the geographical position of the user to provide services to the end users. However, a mobile device is still resource constrained, and some applications usually demand more resources than a mobile device can afford. The information retrieval in mobile devices is a tedious task due to the limited processing capability and low storage space available. To alleviate this, a mobile device should get resources from an external source. In this system, web services are used as the external resource for the ExcellentService. As a further extension, we can extend our web services into cloud computing platforms. Hence the advent of cloud computing in location-based services increases the user’s information retrieve capability by overcoming the mobile’s storage space and processing capability. The message can be exchanged between user and location service provider in mobile device accessing the cloud by minimizing cost, data storage and processing power. Our main goal is to provide dynamic location-based service and increase the information retrieve accuracy especially on the limited mobile screen by accessing cloud computing technology.

**REFERENCES**

[1] ABUM02 Location, Not Quite Here Yet, London, 2002.

[2] Aditya Arya , Mrs. RupaliSyal, “Case Study of Mobile Clients for Location-based Services”, 2010.

[3] Ahmad, Z. ;Palaniappan, S. ; Woon Wei Lee ; Yew Kok Han , “Geographical Information System-supported location-based services: Seamless content & functionality integration using web services”, International Conference on Computing & Informatics, 2006. ICOCI '06. , Page(s): 1 – 3

[4] Ahmad, Z. ;Palaniappan, S. ; Woon Wei Lee ; Yew Kok Han, “Geographical Information System-supported location-based services: Seamless content & functionality integration using web services”, International Conference on Computing & Informatics, 2006. ICOCI '06. , Page(s): 4 – 6

[5] AnusuriyaDevaraju, Simon Beddus , “Positioning Simulator for Location-based Service ”, 2012.

[6] AnusuriyaDevaraju, Simon Beddus, “Building Location-based Service with Java Technologies “

[7] Atkinson, P.: Man in a briefcase: the social construction of the laptop computer and the emergence of a type form. In Journal of Design History, 18(2), 2005, pp. 191-205.

[8] Ayhan, S. ;Comitz, P. ; LaMarche, R., “Implementing geospatially enabled aviation web services”, Integrated Communications, Navigation and Surveillance conference, 2008. ICNS 2008, Page(s): 1 – 8

[9] Barbeau, S. ; Labrador, M.A. ; Perez, A.J. ; Winters, P.L. ; Georggi, N.L. ; Aguilar, D. ; Perez, R.A. “Dynamic Management Real-Time Location Data on GPS-Enable Mobile Phone “, The Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, 2008. UBICOMM '08. , Page(s): 343 - 348

[10] Bareth, U. ;Kupper, A., “Energy-Efficient Position Tracking in Proactive Location-Based Services for Smartphone Environments”, IEEE 35th Annual Computer Software and Applications Conference (COMPSAC), 2011 , Page(s): 516 - 521

[11] Bareth, U., “Simulating Power Consumption of Location Tracking Algorithms to Improve Energy-Efficiency of Smartphones”, IEEE 36th Annual Computer Software and Applications Conference (COMPSAC), 2012 , Page(s): 613 – 622

[12] BratislavPredic, Dragan Stojanovic , “A Framework for Handling Mobile Objects in Location-based Services” , 8th AGILE Conference on GIScience, 2005

[13] Brown, Barry, Sellen, Abigail and O'Hara, Kenton P. ,” A Diary Study of Information Capture in Working Life.”, Proceedings of the ACM CHI 2000 Human Factors in Computing Systems Conference, In: Turner, Thea, Szwillus, Gerd, Czerwinski, Mary, Peterno, Fabio and Pemberton, Steven (eds.) April 1-6, 2000, The Hague, The Netherlands. pp. 438 -445

[14] C.Ahn and Y.Nah , “Design of Location-based Web Service Framework for Context-Aware Applications in Ubiquitous Environments.”, 2010 IEEE InternationalConference on Sensor Networks, Ubiquitous, and Trustworthy Computing.

[15] Chandra, A. ; Jain, S. ; Qadeer, M.A., “Implementation of location awareness and sharing system based on GPS and GPRS using J2ME, PHP and MYSQL”, 3rd International Conference on Computer Research and Development (ICCRD), 2011 ,Volume: 1, Page(s): 216 – 218

[16] Chandra, A. ; Jain, S. ; Qadeer, M.A., “Implementation of location awareness and sharing system based on GPS and GPRS using J2ME, PHP and MYSQL”, 3rd International Conference on Computer Research and Development (ICCRD), 2011 , Volume: 1, Page(s): 219 - 220

[17] Chen, G. and Kotz, D. (2000) A survey of context-aware mobile computing research. Dartmouth Computer Science Technical Report TR2000-381, Dartmouth College, Hanover, NH.

[18] Chen, G. and Kotz, D. , “ A survey of context-aware mobile computing Research”, Dartmouth Computer Science Technical Report TR2000 -381 , Dartmouth Col lege, Hanover, NH, 2000 , Page(s): 18 – 20

[19] Ching, W. ; Rue Jing Teh ; Binghao Li ; Rizos, C., “UniwideWiFi based positioning system”, IEEE International Symposium on Technology and Society (ISTAS), 2010 , Page(s): 180 – 189

[20] Chinthaka, Eran, “Enable REST with Web services, Part 1: REST and Web services in WSDL 2.0 “, MAY 2007. URL, http://www.ibm.com/developerworks/webservices/library/ws-rest1/, last viewed 2008-03-24.

[21] Chi-Yi Lin ; Ming-Tze Hung ; Wei-Hsun Huang , “A Location-Based Personal Task Management Application for Indoor and Outdoor Environments”, 15th International Conference on Network-Based Information Systems (NBiS), 2012 , Page(s): 582 – 587

[22] Chu-Sing Yang ; Pang-Wei Tsai ; Ming-Yi Liao ; Chun-Chia Huang ; Yeh, C.E., “Location-Based Mobile Multimedia Push System”, International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2010, Page(s): 181 – 184

[23] Fan Yang, Zhi-Mei Wang, “A mobile location-based information recommendation system based on GPS and WEB2.0 services", Journal WSEAS Transactions on Computers, Volume 8, Issue 4, pp. 725-734, April 2009 ,World Scientific and Engineering Academy and Society (WSEAS) Stevens Point, Wisconsin, USA

[24] JunhuiZhao ;CaiMu Zheng ; Di Zhou, “Design and Implementation of a Location-Based Service Platform”, 10th International Conference on Advanced Communication Technology, 2008. ICACT 2008. , Volume: 1, Page(s): 529 - 533

[25] K. Michael, A. Mcnamee, M. G Michael , H. Tootell, “Location-Based Intelligence-Modeling Behavior in Humans using GPS”, The Proc. Of IEEE International Symposium on Technology and Society, New York, 2006.

[26] Ling Hu ; Wei-Shinn Ku ; Bakiras, S. ; Shahabi, C., “Spatial Query Integrity with Voronoi Neighbors”, IEEE Transactions on Knowledge and Data Engineering, Volume: 25 , Issue: 4, Page(s): 863 – 876

[27] M. Zahaby, P. Gaonjur and S. Farajian, “Location Tracking In GPS using Kalman Filter Through Sms”, in EUROCON 2009, IEEE, (2009) May 18-23.

[28] ManavSinghal, Anupam Shukla , " Implementation of Location-based Services in Android using GPS and Web Services” , Internationla Journal of Computer Science Issues, Vol.9, Issue 1, No.2, January 2012

[29] Marmasse, N. and Schmandt, C., “Location-Aware Information Delivery with ComMotion” , Lecture , 2003.

[30] McCullough, Malcolm (2004): Digital Ground: Architecture, Pervasive Computing, and Environmental Knowing. The MIT Press

[31] Mladenovic, V. ;Lutovac, M.M. ; Lutovac, M.D., “Electronic tour guide for Android mobile platform with multimedia travel book”, 20th Telecommunications Forum (TELFOR), 2012 , Page(s): 1460 – 1463

[32] MuracevicDzenana and FahrudinOrcevic , “Integration of Spatial Data with Business Intelligence Systems", Proceedings of the IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), December 2008, pp. 447-452

[33] N. Deblauwe, P. Ruppel, “Combining GPS and GSM Cell-ID positioning for Proactive Location-based Services,” Proc. of Fourth Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services, pp. 1-7, Aug. 2007.

[34] O’Hare, G.M.P. and O’Grady, M.J. (2003) Gulliver’s Genie: a multi-agent system for ubiquitous and intelligent content delivery. Computer Communications, 26, 117–87.

[35] Priyanka Shah, RutaGadgil, Neha Tamhankar, " Location-based Reminder Using GPS for Mobile (Android)", ARPN Journal of Science and Technology vol2, No.4, May 2012

[36] Soyoung Hwang and Donghui Yu, " GPS Locatization Improvement of Smartphones Using Built-in Sensors", International Journal of Smart Home, Vol.6, No.3, July, 2012

[37] T.K.C. “The Tree-to-Tree Correction Problem.” Journal of the ACM, 26(3), pp. 422-433, 1979

[38] W. Yang, “Identifying Syntactic Differences Between Two Programs,” Software Practice and Experience, vol. 21(7), pp. 739-755, 1991

[39] Weiser, Mark (1991): The Computer for the 21st Century. In Scientific American, 265 (3) pp. 94-104

[40] Yamaguchi, T. ;Takami, K., “Determining a user's location in underground malls using opportunistic advertisement delivery systems”, IEEE Region 10 Conference TENCON 2011 - 2011 , Page(s): 498 – 502

**Author`s Publications**

[1] Wai lin Aung, Lieutenant Commander, Ye Zaw Myint, Captain, “Web-based Intrusion Detection Systems”, Science and Technology Development Journal, Volume VI, Issue I, September 2014, Defence Services Technology Academy.

[2] Wai lin Aung, Lieutenant Commander, Ye Zaw Myint, Captain, “Prevention of SQL Injection Vulnerabilities”, Science and Technology Development Journal, Volume VI, Issue I, September 2014, Defence Services Technology Academy.

[3] Ye Zaw Myint, Captain, Wai lin Aung, Lieutenant Commander, “Architecture and Components of Android Operating System”, Science and Information Journal, Volume VII, Issue III, September 2015, Defence Services Academy.

[4] Soe Thant, Major, Wai lin Aung, Lieutenant Commander, Ye Zaw Myint, Captain, “Approach To The Detection of Web-Based Attacks”,6th ICSE 2015,YTU.

[5] Wai lin Aung, Lieutenant Commander, Ye Zaw Myint, Major, “Efficient Location-Based Services System Using Proximity Detection And Preference Tree Matching Algorithms In Android Platform”, Science and Technology Development Journal, Volume V, Issue II, March 2016, Defence Services Technology Academy.

[6] Wai lin Aung, Lieutenant Commander, Ye Zaw Myint, Major, “Computer Forensics အေၾကာင္းခြဲျခမ္းစိတ္ျဖာေလ့လာျခင္း”, Science and Technology Development Journal, Defence Services Technology Academy. (Pending)

**Appendix A**

**Searching Method in Android Platform**

Search2Activity.java

**package** com.engineer4myanmar.json;

**import** java.io.BufferedReader;

**import** java.io.IOException;

**import** java.io.InputStream;

**import** java.io.InputStreamReader;

**import** java.util.ArrayList;

**import** java.util.HashMap;

**import** java.util.List;

**import** org.apache.http.HttpEntity;

**import** org.apache.http.HttpResponse;

**import** org.apache.http.NameValuePair;

**import** org.apache.http.StatusLine;

**import** org.apache.http.client.ClientProtocolException;

**import** org.apache.http.client.HttpClient;

**import** org.apache.http.client.methods.HttpGet;

**import** org.apache.http.client.utils.URLEncodedUtils;

**import** org.apache.http.impl.client.DefaultHttpClient;

**import** org.apache.http.message.BasicNameValuePair;

**import** org.json.JSONArray;

**import** org.json.JSONException;

**import** org.json.JSONObject;

**import** android.app.Activity;

**import** android.app.ProgressDialog;

**import** android.content.Context;

**import** android.content.Intent;

**import** android.content.SharedPreferences;

**import** android.os.AsyncTask;

**import** android.os.Bundle;

**import** android.util.Log;

**import** android.view.View;

**import** android.widget.AdapterView;

**import** android.widget.AdapterView.OnItemSelectedListener;

**import** android.widget.ArrayAdapter;

**import** android.widget.EditText;

**import** android.widget.LinearLayout;

**import** android.widget.Spinner;

**import** android.widget.Toast;

**public** **class** Search2Activity **extends** Activity **implements** OnItemSelectedListener {

**public** **static** String *ipaddress1* = "mmgreenhackers.com";//"192.168.1.100";

Person person = **null**;

Spinner spServices;

Spinner spRange;

Spinner spRating;

EditText etMin;

EditText etMax;

Spinner spCuisine;

LinearLayout llRating;

LinearLayout llPrice;

LinearLayout llMaxMin;

LinearLayout llCuisine;

String input\_services = "";

String input\_range = "";

String input\_rating = "";

String input\_min = "";

String input\_max = "";

String input\_cuisine;

String sjson = "";

JSONObject jObj;

JSONParser jsonParser = **new** JSONParser();

// change here your ip/folder/php

**private** **static** String *url\_search* = "http://" + *ipaddress1*

+ "/esdb/search3.php";

**private** ProgressDialog pDialog;

**final** ArrayList<String> Alist = **new** ArrayList<String>();

ArrayList<HashMap<String, String>> resultList = **new** ArrayList<HashMap<String, String>>();

String finalResult="";

// /////////////////////////////////////////////////////////////////////

@Override

**public** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.*activity\_search2*);

llRating = (LinearLayout)findViewById(R.id.*llRating* );

llPrice = (LinearLayout)findViewById(R.id.*llPrice* );

llMaxMin = (LinearLayout)findViewById(R.id.*llMaxMin*);

llCuisine = (LinearLayout)findViewById(R.id.*llCuisine*);

spServices = (Spinner) findViewById(R.id.*spServices*);

ArrayAdapter<CharSequence> adapter = ArrayAdapter.*createFromResource*(

**this**, R.array.*services\_arrays*, R.layout.*spinner\_item*);

adapter.setDropDownViewResource(android.R.layout.*simple\_spinner\_dropdown\_item*);

spServices.setAdapter(adapter);

spRange = (Spinner) findViewById(R.id.*spRange*);

ArrayAdapter<CharSequence> adapter1 = ArrayAdapter.*createFromResource*(

**this**, R.array.*range\_arrays*, R.layout.*spinner\_item*);

adapter.setDropDownViewResource(android.R.layout.*simple\_spinner\_dropdown\_item*);

spRange.setAdapter(adapter1);

spRating = (Spinner) findViewById(R.id.*spRating*);

ArrayAdapter<CharSequence> adapter2 = ArrayAdapter.*createFromResource*(

**this**, R.array.*ratings\_arrays*, R.layout.*spinner\_item*);

adapter.setDropDownViewResource(android.R.layout.*simple\_spinner\_dropdown\_item*);

spRating.setAdapter(adapter2);

etMin = (EditText) findViewById(R.id.*etMin*);

etMax = (EditText) findViewById(R.id.*etMax*);

spCuisine = (Spinner) findViewById(R.id.*spCuisine*);

ArrayAdapter<CharSequence> adapter3 = ArrayAdapter.*createFromResource*(

**this**, R.array.*cuisine\_arrays*, R.layout.*spinner\_item*);

adapter.setDropDownViewResource(android.R.layout.*simple\_spinner\_dropdown\_item*);

spCuisine.setAdapter(adapter3);

spServices.setOnItemSelectedListener(**this**);

}

**public** **void** funSearchNow(View v) {

HashMap<String,String> hashMap = **new** HashMap<String,String>();

// new registerJSONdbTask().execute(url\_register);

input\_services = String.*valueOf*(spServices.getSelectedItem());

// TO DO hospital and bank data omit

input\_range = String.*valueOf*(spRange.getSelectedItem());

input\_rating = String.*valueOf*(spRating.getSelectedItem());

input\_cuisine = String.*valueOf*(spCuisine.getSelectedItem());

input\_min = etMin.getText().toString();

input\_max = etMax.getText().toString();

//Bundle with hashMap

hashMap.put("name",input\_services);

hashMap.put("range",input\_range);

hashMap.put("rating",input\_rating);

hashMap.put("cuisine",input\_cuisine);

hashMap.put("min",input\_min);

hashMap.put("max",input\_max);

/////////////////////////////////////////

String lat;

String lng;

lat = getLocation("lat", "22.024104");

lng = getLocation("lng", "96.447339");

hashMap.put("x0",lat);

hashMap.put("y0",lng);

hashMap.put("r",input\_range);

Intent intent = **new** Intent(getApplicationContext(),

SearchListActivity.**class**);

intent.putExtra("hashMap",hashMap);

startActivity(intent);

}

**public** **void** funSearchPDA(View v) {

HashMap<String,String> hashMap = **new** HashMap<String,String>();

// new registerJSONdbTask().execute(url\_register);

input\_services = String.*valueOf*(spServices.getSelectedItem());

// TO DO hospital and bank data omit

input\_range = String.*valueOf*(spRange.getSelectedItem());

input\_rating = String.*valueOf*(spRating.getSelectedItem());

input\_cuisine = String.*valueOf*(spCuisine.getSelectedItem());

input\_min = etMin.getText().toString();

input\_max = etMax.getText().toString();

//Bundle with hashMap

hashMap.put("name",input\_services);

hashMap.put("range",input\_range);

hashMap.put("rating",input\_rating);

hashMap.put("cuisine",input\_cuisine);

hashMap.put("min",input\_min);

hashMap.put("max",input\_max);

/////////////////////////////////////////

String lat;

String lng;

lat = getLocation("lat", "22.024104");

lng = getLocation("lng", "96.447339");

hashMap.put("x0",lat);

hashMap.put("y0",lng);

hashMap.put("r",input\_range);

Intent intent = **new** Intent(getApplicationContext(),

SearchList2Activity.**class**);

intent.putExtra("hashMap",hashMap);

startActivity(intent);

}

**public** String getLocation(String key, String default\_value) {

SharedPreferences sharedPref = **this**.getSharedPreferences(

"com.engineer4myanmar.json", Context.*MODE\_PRIVATE*);

String val = sharedPref.getString(key, default\_value);

**return** val;

}

**public** **void** onItemSelected(AdapterView<?> parent, View view, **int** pos,

**long** id) {

**switch**(pos)

{

**case** 0:

showViews();

**break**;

**case** 1:

hideViews();

**break**;

**case** 2:

showViews();

**break**;

**case** 3:

hideViews();

**break**;

}

}

**public** **void** onNothingSelected(AdapterView<?> arg0) {

}

**public** **void** hideViews()

{

llRating.setVisibility(4);

llPrice.setVisibility(4);

llMaxMin.setVisibility(4);

llCuisine.setVisibility(4);

}

**public** **void** showViews()

{

llRating.setVisibility(0);

llPrice.setVisibility(0);

llMaxMin.setVisibility(0);

llCuisine.setVisibility(0);

}

// ////////////////////////////////////////////////////////////////////

@SuppressWarnings("null")

**public** String readJSONFeed(String URL, List<NameValuePair> params) {

StringBuilder stringBuilder = **new** StringBuilder();

HttpClient client = **new** DefaultHttpClient();

String paramString = URLEncodedUtils.*format*(params, "utf-8");

URL += "?" + paramString;

HttpGet httpGet = **new** HttpGet(URL);

**try** {

HttpResponse response = client.execute(httpGet);

**if** (response != **null**) {

// Log.e("JSON", String.valueOf(response.toString()));

} **else** {

Log.*e*("JSON", String.*valueOf*(response.toString()));

Log.*e*("JSON", "HTTPGET ERROR");

}

StatusLine statusLine = response.getStatusLine();

**int** statusCode = statusLine.getStatusCode();

// Log.e("JSON", String.valueOf(statusCode));

**if** (statusCode == 200) {

HttpEntity entity = response.getEntity();

InputStream content = entity.getContent();

BufferedReader reader = **new** BufferedReader(

**new** InputStreamReader(content));

String line;

**while** ((line = reader.readLine()) != **null**) {

stringBuilder.append(line);

}

// Log.e("JSON ! GOT IT \*\*\*\* ", stringBuilder.toString());

} **else** {

Log.*e*("JSON", "Failed to download file");

}

} **catch** (ClientProtocolException e) {

e.printStackTrace();

} **catch** (IOException e) {

e.printStackTrace();

}

**return** stringBuilder.toString();

}

}

**Appendix B**

**User Interface (Android Platform)**

Activity\_search.xml

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

xmlns:tools=*"http://schemas.android.com/tools"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*

android:background=*"@drawable/back"*

android:orientation=*"vertical"* >

<LinearLayout

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:orientation=*"vertical"* >

<TextView

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center"*

android:gravity=*"center"*

android:textColor=*"#669900"*

android:textSize=*"20dip"*

android:text=*"Select Services"* />

<Spinner

android:id=*"@+id/spServices"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center"*

android:entries=*"@array/services\_arrays"*

android:gravity=*"center"*

android:prompt=*"@string/services\_prompt"* />

</LinearLayout>

<LinearLayout

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:layout\_marginTop=*"15dp"*

android:orientation=*"horizontal"* >

<TextView

android:layout\_width=*"0dp"*

android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center"*

android:layout\_weight=*"1"*

android:gravity=*"center"*

android:textSize=*"17dip"*

android:textColor=*"#669900"*

android:text=*"Range(km)"* />

<Spinner

android:id=*"@+id/spRange"*

android:layout\_width=*"0dp"*

android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center"*

android:layout\_weight=*"1"*

android:entries=*"@array/range\_arrays"*

android:gravity=*"center"*

android:prompt=*"@string/range\_prompt"* />

</LinearLayout>

<LinearLayout

android:id=*"@+id/llRating"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:layout\_marginTop=*"15dp"*

android:orientation=*"horizontal"* >

<TextView

android:layout\_width=*"0dp"*

android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center"*

android:layout\_weight=*"1"*

android:gravity=*"center"*

android:textColor=*"#669900"*

android:textSize=*"17dip"*

android:text=*"Rating"* />

<Spinner

android:id=*"@+id/spRating"*

android:layout\_width=*"0dp"*

android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center"*

android:layout\_weight=*"1"*

android:entries=*"@array/ratings\_arrays"*

android:gravity=*"center"*

android:prompt=*"@string/rating\_prompt"* />

</LinearLayout>

<LinearLayout

android:id=*"@+id/llPrice"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:layout\_marginTop=*"15dp"*

android:orientation=*"horizontal"* >

<TextView

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center"*

android:gravity=*"center"*

android:textSize=*"17dip"*

android:textColor=*"#669900"*

android:text=*"Price"* />

</LinearLayout>

<LinearLayout

android:id=*"@+id/llMaxMin"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:orientation=*"horizontal"* >

<EditText

android:id =*"@+id/etMin"*

android:layout\_width=*"0dp"*

android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center"*

android:layout\_weight=*"1"*

android:gravity=*"center"*

android:textColor=*"#ffbb33"*

android:text=*"1"*

android:hint=*"Minimum"* />

<EditText

android:id =*"@+id/etMax"*

android:layout\_width=*"0dp"*

android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center"*

android:layout\_weight=*"1"*

android:gravity=*"center"*

android:textColor=*"#ffbb33"*

android:text=*"100"*

android:hint=*"Maximum"* />

</LinearLayout>

<LinearLayout

android:id=*"@+id/llCuisine"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:orientation=*"vertical"* >

<TextView

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center"*

android:gravity=*"center"*

android:textColor=*"#669900"*

android:textSize=*"17dip"*

android:text=*"Cuisine"* />

<Spinner

android:id=*"@+id/spCuisine"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center"*

android:entries=*"@array/cuisine\_arrays"*

android:gravity=*"center"*

android:prompt=*"@string/cuisine\_prompt"* />

</LinearLayout>

<LinearLayout

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:orientation=*"vertical"* >

<Button

android:id =*"@+id/btnSearch"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center"*

android:gravity=*"center"*

android:onClick=*"funSearchNow"*

android:text=*"Search by DBA"* />

<Button

android:id =*"@+id/btnSearch2"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center"*

android:gravity=*"center"*

android:onClick=*"funSearchPDA"*

android:text=*"Search by PDA"* />

</LinearLayout>

</LinearLayout>

**Appendix C**

**Distance Based Algorithm**

Web Service search3.php

<?php

$time\_start = microtime(true);

require\_once \_\_DIR\_\_ . '/db\_user\_connect.php';

$con = new DB\_CONNECT();

$con->connect();

if (isset($\_GET['catalog'])){

$catalog = $\_GET['catalog'];

$rating =$\_GET['rating'];

$p1 = $\_GET['p1'];

$p2 = $\_GET['p2'];

$cui = $\_GET['cuisine'];

$x0 = $\_GET['x0'];

$y0 = $\_GET['y0'];

$r = $\_GET['r'];

function getDistance($lat1, $lng1, $lat2, $lng2)

{

// radius of earth; @note: the earth is not perfectly spherical, but this is considered the 'mean radius'

$radius = 6371.009; // in kilometers

//elseif ($unit == 'mi') $radius = 3958.761; // in miles

// convert degrees to radians

$lat1 = deg2rad((float) $lat1);

$lng1 = deg2rad((float) $lng1);

$lat2 = deg2rad((float) $lat2);

$lng2 = deg2rad((float) $lng2);

// great circle distance formula

return $radius \* acos(sin($lat1) \* sin($lat2) + cos($lat1) \* cos($lat2) \* cos($lng1 - $lng2));

}

$sql = "SELECT information.name AS

info\_name,

$catalog.\*,

information.\*

FROM

information

inner join

$catalog

on

information.id= $catalog.info\_id

WHERE

information.rating = '$rating'

AND $catalog.cuisine = '$cui'

OR $p1 BETWEEN $catalog.price\_high

AND $catalog.price\_low

AND $p2 BETWEEN $catalog.price\_high

AND $catalog.price\_low

GROUP BY

information.name " ;

//echo $sql;

//Starting query

$res = mysqli\_query($con->myconn,$sql) or die(mysqli\_error());

//Extracting rows from result

$row = mysqli\_num\_rows($res);

//if there is data?

if($row> 0){

//create two dimensional array for every data resulted

$response["result"]=array();

while ($row = mysqli\_fetch\_assoc($res)){

//create new empty array to store data from one row

$found = array();

$found['info\_name'] = $row['info\_name'];

$found['address'] = $row['address'];

$found['phone\_no'] = $row['phone\_no'];

$found['lat'] = $row['lat'];

$found['lng'] = $row['lng'];

$found['cui'] = $row['cuisine'];

$x1 = $found['lat'];

$y1 = $found['lng'];

//echo $x1.$y1;

if( $r >= getDistance($x0, $y0, $x1, $y1))

{

//}else{//Do Nothing} //skip these data

//save found array into another two dimensional array result

array\_push($response["result"],$found);

}

}

$time\_end = microtime(true);

$execution\_time = ($time\_end - $time\_start);

//Add success flag 1 into two dimentional array response

$response["success"]=1;

$response["execution\_time"]=$execution\_time;

//Creat Final JSON by encoding two dimentional array response and ECHO it

echo json\_encode($response);

}

else{

//if no rows or data success Flags =0 and add to two dimentional array $response

$response["success"]=0;

//add error message in two dimentional array $response

$response["message"]="Not found";

//(only two error code included)

echo json\_encode($response);

}

// check for empty result from query

if ($res){

// success

$response["success"] = 1;

// echoing JSON response

//echo json\_encode($response);

} else {

// no logins found

$response["success"] = 0;

$response["message"] = "No detail found";

// echo no users JSON

echo json\_encode($response);

}

?>

**Appendix D**

**Proximity Detection Algorithm**

Search4.php

<?php

//place this before any script you want to calculate time

$time\_start = microtime(true);

// include db connect class

require\_once \_\_DIR\_\_ . '/db\_user\_connect.php';

// connecting to db

$con = new DB\_CONNECT();

$con->connect();

if (isset($\_GET['catalog'])){

//Read input data according to their name and save in variables

$catalog = $\_GET['catalog'];

$rating =$\_GET['rating'];

$p1 = $\_GET['p1'];

$p2 = $\_GET['p2'];

$cui = $\_GET['cuisine'];

$x0 = $\_GET['x0'];

$y0 = $\_GET['y0'];

$r = $\_GET['r'];

/\*

\* Function to draw circle by PDA Algorithm

\*/

function midPointCircleAlgo($radius, $xC, $yC)

{

$P;

$x, $y;

$P = 3 - 2 \* $radius;

$x = 0;

$y = $radius;

draw($x,$y,$xC,$yC);

while($x<=$y){

$x++;

if($P<0)

{

$P+=4\*$x+6;

}else{

$P+=4\*($x-$y)+10;

$y--;

}

draw($x,$y,$xC,$yC);

}

}

/\*

\* Function to Draw Circle

\*/

function draw($x, $y, $xC, $yC){

$xC = 320+$xC;

$yC = 240-$yC;

putpixel($xC + $x,$yC + $y,1);

putpixel($xC + $x,$yC - $y,1);

putpixel($xC - $x,$yC + $y,1);

putpixel($xC - $x,$yC - $y,1);

putpixel($xC + $y,$yC + $x,1);

putpixel($xC - $y,$yC + $x,1);

putpixel($xC + $y,$yC - $x,1);

putpixel($xC - $y,$yC - $x,1);

}

/\*

\* Checking lat long is within in circle range

\*/

function isWithinInCircle($x,$y){

if(isbetween($x,$y)){

return true;

}else{

return false;

}

}

$sql = "SELECT information.name AS

info\_name,

$catalog.\*,

information.\*

FROM

information

inner join

$catalog

on

information.id= $catalog.info\_id

WHERE

information.rating = '$rating'

AND $catalog.cuisine = '$cui'

OR $p1 BETWEEN $catalog.price\_high

AND $catalog.price\_low

AND $p2 BETWEEN $catalog.price\_high

AND $catalog.price\_low

GROUP BY

information.name " ;

//echo $sql;

//Starting query

$res = mysqli\_query($con->myconn,$sql) or die(mysqli\_error());

//Extracting rows from result

$row = mysqli\_num\_rows($res);

//3-Debug information save in search3.log (rows from result)

$myfile = fopen("search3.log", "a") or die("Unable to open file!");

fwrite($myfile, "\n Roll ".$row." \n");

fclose($myfile);

//if there is data?

if($row> 0){

//if the lat and long are valid???

/\*

\* Draw Circle with PDA algorithm

\*/

midPointCircleAlgo($r, $x0, $y0);

//create two dimensional array for every data resulted

$response["result"]=array();

while ($row = mysqli\_fetch\_assoc($res)){

//create new empty array to store data from one row

$found = array();

$found['info\_name'] = $row['info\_name'];

$found['address'] = $row['address'];

$found['phone\_no'] = $row['phone\_no'];

$found['lat'] = $row['lat'];

$found['lng'] = $row['lng'];

$found['cui'] = $row['cuisine'];

$x1 = $found['lat'];

$y1 = $found['lng'];

//echo $x1.$y1;

if(isWithinInCircle($x1, $y1))

{

//}else{//Do Nothing} //skip these data

//save found array into another two dimensional array result

array\_push($response["result"],$found);

}

}

$time\_end = microtime(true);

$execution\_time = ($time\_end - $time\_start);

//Add success flag 1 into two dimentional array response

$response["success"]=1;

$response["execution\_time"]=$execution\_time;

//Creat Final JSON by encoding two dimentional array response and ECHO it

echo json\_encode($response);

}

else{

//if no rows or data success Flags =0 and add to two dimentional array $response

$response["success"]=0;

//add error message in two dimentional array $response

$response["message"]="Not found";

//Creat Final JSON by encoding two dimentional array response and ECHO it

//(only two error code included)

echo json\_encode($response);

}

}

// check for empty result from query

if ($res){

// success

$response["success"] = 1;

// echoing JSON response

//echo json\_encode($response);

} else {

$response["success"] = 0;

$response["message"] = "No detail found";

// echo no users JSON

echo json\_encode($response);

}

?>