CHAPTER 3

FRAMEWORK FOR DYNAMIC DATA MANAGEMENT IN LOCATION-BASED SERVICES

3.1 Introduction

The last decade showed an accelerated development of mobile and Internet technologies. Wireless connectivity and mobility are no longer considered a luxury but are a necessity in our everyday life. Users need all available aid from latest technologies to make their busy lives simpler and organized in order to improve their efficiency. Advances in wireless communication technologies and mobile Internet-enabled devices like smart phones and PDAs, have enabled ubiquitous Web-based computing and service distribution [36]. Mobile computing is the technology that provides the valid, precise information to the subscriber in time whenever and wherever [61], which introduced new business models and the development of service architecture. People could connect mobile data with the information system to get useful information at any time through the new generation of intelligent equipment: mobile computer, vehicle, handset and so much as watch, which are of mobile computing functions. What can meet the basic demand of people is that time, place and content information. What's more, the position is the most important information while people are under mobile condition, especially precise position information under urgent situation.

Location-Based Services (LBS) are such an example. Location-based services (LBS) are services (through a combination of hardware devices, communication networks – often wireless – and software applications) that access, provide or otherwise act upon location information. We distinguish between mobile position determination systems that determine the location of a mobile terminal and application-oriented location services, which exploit device location in some application service sought by a client. With rapid advances in ubiquitous connectivity people, vehicles and computers are connected at all times and

location-aware computing will become a critical capability in future computing environments. Location Based Services can be classified in three categories namely Public safety or emergency services, Consumer services, and Enterprise services. Consumer Services contain Navigation Services, Location based advertising, Location based reminders, Family and friend finder, Location based search, and Location based mobile gaming. Enterprises applications include vehicle fleet management and urban traffic analytics. Location based services (LBS) are gaining widespread user acceptance and its usage is increasing day-by-day. Some of the most popular LBS available today are GPS based mobile

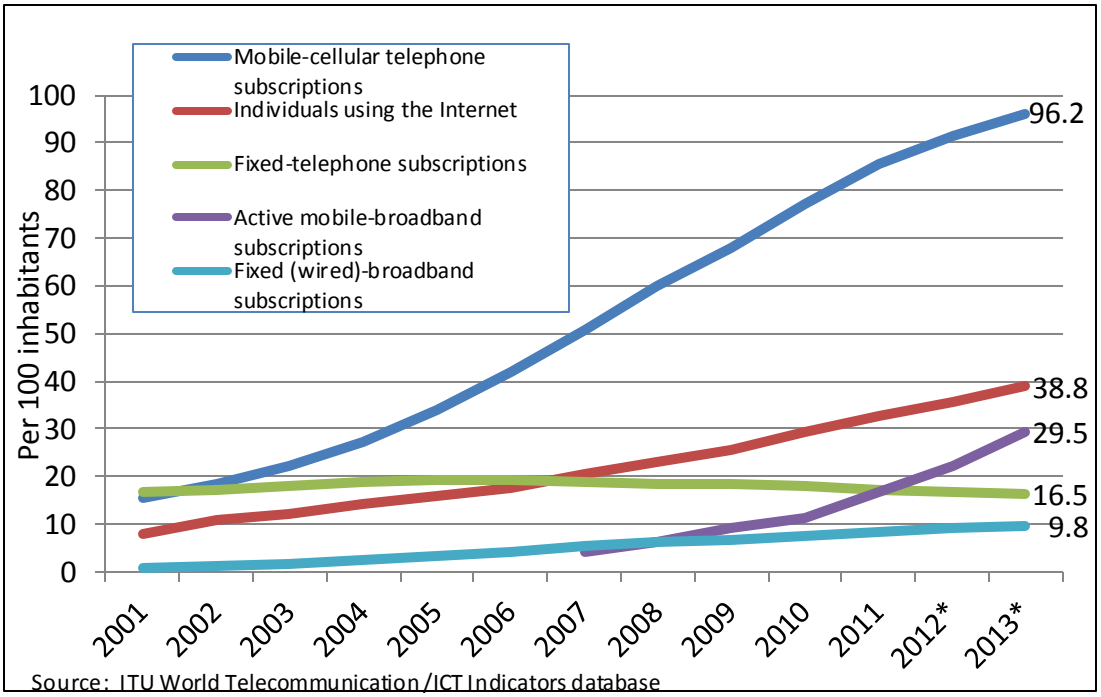

Figure 3.1: Global ICT developments, 2001-2013

navigation systems (e.g. Garmin), location-related social networks (e.g. Facebook), friend queries (e.g. Foursquare) and location based advertisements (e.g. Google). According to the International Telecommunications Union (ITU), there were 6.8 billion mobile subscribers worldwide; Corresponding to a global penetration of 96% in 2013, and 40% mobile subscribers could access the mobile Internet as given in Figure 3.1. Many consumers prefer mobile browsers for banking, travel, shopping, local info, news, video, sports and blogs, and prefer apps for games, social media, maps and music. The increased use of mobile

devices and the use of different spatial applications such as spatial alarms, continuous queries, navigational applications, Location based Social networking etc. have resulted in the explosion of spatial data. It is anticipated that by 2020 the average mobile user will consume one gigabyte of data per day. Thus the data management in Location Based Services is a major research issue.

So, along with all its benefits, LBS also introduce many challenges for data management. Though the wireless network infrastructure has made significant advances in recent years, they still present important limitations. The major hurdles still facing are, intermittent connectivity, a short battery lifetime, limited communications (slow, unreliable, and/or expensive), and limited capabilities in terms of memory, processing, storage, and display size. Moreover, in the case of mobile devices, the resources accessed depend on its location. Due to these factors, classical methods for data management and query processing are insufficient and must be adapted to the new environment [118, 72, 121, and 66]. In this chapter, we introduce a *network assisted* framework for dynamic data management in Location Based Services (LBS).

The rest of the chapter is outlined as follows. Section 3.2 describes the importance of Location, Location modeling, and Location Indexing in the context of Location Based Services; In Section 3.3, we describe the conventional model of Location Based Service delivery. (LBS Delivery model). In Section 3.4, we introduce the proposed architecture of hierarchical database for scalable and dynamic data management in Location Based Services (LBS) followed by the data placement model, the role of proxy and its architecture and data dissemination model that are used in our framework. The factors for system analysis are detailed in Section 3.5 followed by a brief description of related work in Section 3.6

3.2 Importance of Location

Mobile information services or mobile applications, as a whole, can be subdivided into two categories depending on the way information access is controlled [74].

a) General information services that access information without concerning the user' current location
b) Location-based service (LBS) or mobile location-based applications with use the location information as one of the most important parameters.

LBSs 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. Let us consider the example that somebody wants to take a dinner in a restaurant and is therefore searching a restaurant in the Internet. A useful approach to prevent that one gets as search result every restaurant web-page on the world one could restrict the search by adding further search criteria. A good choice is the city where the mobile user is (position), the actual time (evening) or a special type of restaurant (Indian or Chinese). Such kind of restaurant search with respect to position and time can be done by use of a Location Based Service (LBS). Another example, when you walk on the street, you can query where is the proximate restaurant, how to reach it; what is its feature food. And if you come to a strange city and lost your way, you can quickly search the map of your vicinity, marked the target position, the mobile station can automatically display the right route, guide you reach the destination. In all these cases, the location of the mobile user matters. A key characteristic of location based services is that the same service request may need to be answered with completely different results as the user changes his/her location or the targets move. Because of the highly dynamic nature of the problem, traditional information management techniques are not well suited for location based services. Developing proper infrastructure, location management, as well as data management strategies for such services has been a major challenge to both wireless service providers and application developers.

Managing data in LBS faces many challenges

- Constraints of Mobile Environment

  Scarce bandwidth, low-quality communication, frequent network disconnections, and limited local resources, of mobile environment, are some

of the constraints in developing and delivering effective Location Based Services or Location Dependent Information Services.

- Spatial Data

  In Location Based Services (LBS), answers to user queries are highly dynamic. The results of the query can vary with location, because, query results depend on a query's spatial properties. For a location bound query, the query result must be both relevant to the query and valid for the bound location.

- Movement of Mobile Users

  Mobile users normally on the move change their locations, several tasks, such as query scheduling and cache management, are particularly tough in LBS. These challenges have opened many new research problems in LBS data management. Although many traditional data management techniques are applicable to the implementation of LBS, we must reexamine and recreate them to address these issues. Many studies address data management for mobile computing, such as cache management and transaction management.1 However, most existing studies target general data services instead of exploring the special properties of LBSs.

3.2.1 Location Modeling

If we unpack the term "location-based services", it is clear enough that location is an important part of LBS. Location is part of context and it determines what information and services the user may expect. When the mobile client access some kind of Location Based Services, locations must be specified explicitly or implicitly before a client can access information. Location modeling deals with the basic issue of representing space (or more precisely geographic space) for LBS. A location model depends on the system's underlying location identification technique [37]. Two models for representing locations exist:

- ***Geometric model***. The geometric models treat locations and objects as points, areas and volumes within a reference coordinate system, and they can support a range of queries regarding a position, nearest neighbors, and efficient paths among locations. A coordinate is one of n scalar values that define the position of a single point. A coordinate tuple is an ordered list of n coordinates that define the position of a single point. ISO 19111:2007 International Standard requires that the coordinate tuple be composed of one, two or three spatial coordinates. The coordinates are mutually independent and their number is equal to the dimension of the coordinate space. In LBS, system specifies a location as an n-dimensional coordinate (typically n = 2 or 3)—for example, the latitude-longitude pair returned by the GPS System—or a set of coordinates defining an area's bounding geometric shape (such as a polygon). The system can compute regular geometric shapes from concise representations (for example, a circle can be represented by the center and radius). Geometric coordinates can be used to calculate the distance between two geometrically defined positions. Through geometric operations, it can also be determined if two areas overlap, touch each other, or one area contains the other, i.e., topological relations like spatial containment can be derived from the geometry of objects. Hence, geometric coordinates already allow simple spatial reasoning.

  The geometric model's main advantage is its compatibility across heterogeneous systems. However, it can be very costly and complex in terms of the volume of data involved and the need to map the geometric representation to a semantic level suitable for the applications. In addition to describing a coordinate reference system, ISO 19111:2007 International Standards provides for the description of a coordinate transformation or a coordinate conversion between two different coordinate reference systems. With such information, spatial data

referred to different coordinate reference systems can be related to one specified coordinate reference system.

- ***Symbolic modeling using Geographic Identities***. Spatial reference systems using geographic identifiers are not based explicitly on coordinates but on a relationship with a location defined by geographic entities or geographic features. The relationship of the position to the feature may be as follows

  a) Containment, where the position is within the geographic entity, for example in a country;

  b) Based on local measurements, where the position is defined relative to a fixed point or points in the geographic entities or features, for example at a given distance along a street from a junction with another street;

  c) Loosely related, where the position has a fuzzy relationship with the geographic entities or features, for example adjacent to a building or between two buildings.

Logical, real-world entities describe the location space. Entities can be buildings, streets, cities, or system- defined elements such as wireless cells, and is uniquely identifiable by a hierarchical naming system. The symbolic model typically has coarser location granularity than the geometric model because it stresses the representation of relationships between logical entities rather than their precise coordinates. At a semantic level, it is more suitable for LDIS applications. Also, being discrete and well-structured, symbolic location information is easier to manage. Converting locations among heterogeneous systems, however, is difficult.

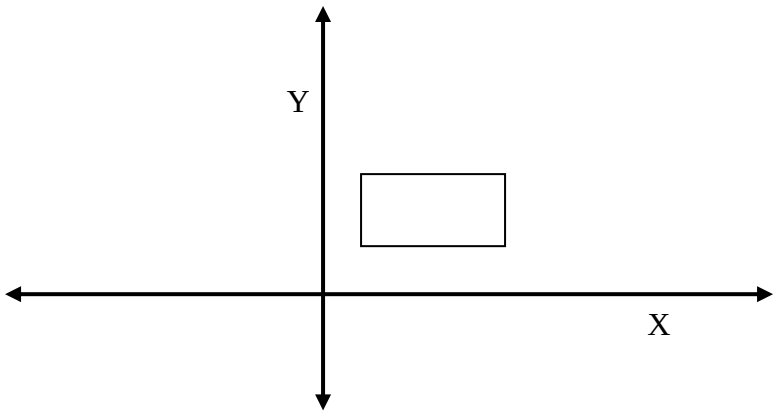

Figure 3.2: Euclidian 2-dimensional coordinate system

There are both pros and cons for both of the reference models. Symbolic models neither provide systemic ways to define locations, nor are extensible at runtime. To add a new location to a symbolic model, extra effort is needed to define its spatial relations with respect to other existing locations. As the number of location objects increases, the spatial relations between these locations increase dramatically. Hence it would be difficult to register all locations, in case the location set is too large. As a result, there might not be enough registered locations to represent precisely the locations in the real environment. [76]

In order to allow users to express all the possible locations, we propose to deploy a geometric location model to specify locations. Geometric model provides all the primitive elements (PE) needed to specify locations. In our proposed architecture we are using Euclidean 2-dimensional model. An example of a geometric coordinate system is the Euclidian 2-dimensional space shown in Figure 3.2 , in which every point is determined by two coordinates, for example, (x, y).

3.2.2 Representations of Locations

By using a geometric location model, three types of spatial expressions can be specified, namely single equation approach, simple predicate approach, and rectangular-based approach. *Table 3.1* shows some examples of the three

approaches. The simple predicate approach can specify only rectangular shapes in 2-D or cubic volume in 3-D spaces, while the equation approach enables users to specify any shapes or volumes they want, provided that they are able to define the equations. The rectangular-based approach states the x and y coordinates in the top left corner and the bottom right corner of a rectangle.

Representation	**Example**
Simple predicate	X > 0 && X < 10 && Y > 0 &&Y < 10
Equations	$X_2 + Y_2 < 10$
Multiple equations	$X_2 + Y_2 < 10$ && $X_2 + Y_2 < 10$
Rectangular-based	(0,0, 10, 10)

Table 3.1 Geometric spatial expressions

The elements used in the first level can be summarized as in Figure 3.3. Variables can be denoted by x, y, and z coordinates, operators are denoted by

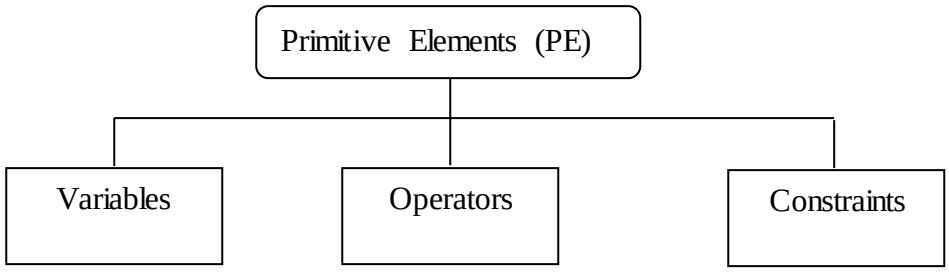

Figure 3.3: Primitive Elements

+,-, *, /, >, <, =, and constraints are denoted by the real number set.

One of the disadvantages of this approach is that it is almost impractical to ask ordinary users, who do not have expertise on specifying locations precisely, to express an equation in order to specify a spatial requirement. Therefore, the

systems based on this approach are often normally assisted by a graphic interface input, allowing users to select areas based on a map. However, a problem is that not every mobile device has a touchscreen allowing users to do so. Even if there is a touchscreen, it is very difficult to select the exact area on a coordinate system, and the situation becomes even worse if the coordinate system is 3-dimensional.

3.2.3 Wireless Infrastructure

Wireless infrastructures have been evolving over time, from analog cellular architectures such as AMPS (1G) to GSM (2G), GPRS (2.5G), and, more recently, UMTS (3G), Bluetooth, Wi-Fi, and WiMax, as alternatives for wireless data communication. [66]. Looking beyond the details of a specific wireless technology, the generally accepted mobile environment infrastructure [117] is considered, which is composed of mobile hosts (MHs) and base stations (BSs). Each BS (or mobile support station, MSS) serves all the MHs within its coverage area or cell. The communication between a MH and the BS that provides it with coverage is wireless (e.g., using cellular network technology or a wireless LAN), and the communication among BSs is wired. Thus, BSs allow the communication between MHs and fixed hosts (FHs) connected to the Internet. In accordance with the previous description, wireless ad-hoc networks are not considered in this work [66]. In a cellular system, the geographical area is divided into adjacent, non-overlapping, hexagonal shaped cells as in Figure 3.4. Each cell has its own transmitter and receiver, called base stations (BS), to communicate with the Mobile Hosts (MH) in that cell.

The term *Mobile Host* refers to any entity that is moving and location is interesting. Various mechanisms are available to get the location of Mobile Host (MH), which usually requires the object to be equipped with some kind of wireless connection. This could be a portable computer with Internet connection used by a passenger travelling in a car, or a PDA connected to a wireless hotspot. Mobile Hosts can be attached to devices such as *GPS,*

(*Global Positioning System*, a free-to-use global network of 24 satellites run by the U.S. Department of Defence, which allows a person to obtain his/her current location) receivers or sensors that feed them with context information.

Similarly, they can be capable of executing different software, such as a Web browser, or a Navigation system, that shows on a map the GPS location of the moving object. Thus, depending on their features, Mobile Host can have different functionalities.

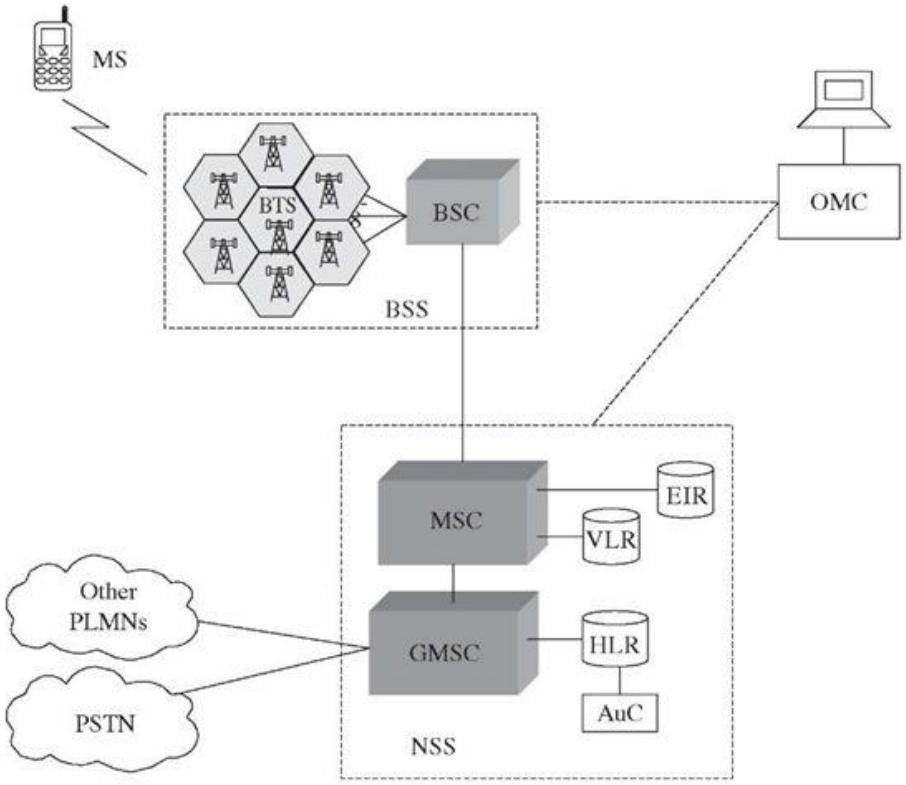

Figure 3.4: Cellular Architecture

Mobile Hosts are always been attached to the Base Station (BS) of the current cell (coverage area) where it is located. As they move, from the present cell to the coverage area of another BS, the connection between the BS and MH has to reset. This process is called *handoff* / *handover*. The mobility of mobile devices introduces an important problem that does not exist in the traditional wired networks. In order to communicate with a mobile device, the cell where it is located must be first obtained. Appropriate architecture has to be defined

to store the location of the Mobile Hosts. For example, the location strategies proposed in the GSM standard use a two-tier system. Each mobile device is associated with a fixed *Home Location Register* (HLR) and a *Visitor Location Register* (VLR). The entry in VLR changes with the location of the device. These two registers keep the location of each Mobile Host. Calls to a device are first directed to the VLR of the caller; if the called up person is not found there, then the called up person's HLR is contacted.

3.2.4 Indexing Locations

Location Based Services (LBS) is an integrated product that has emerged with the development of mobile computing, positioning technologies, and geo-spatial data processing and other technologies. LBS are capable of providing specific spatial information to targeted mobile users that allow users to perform geo-spatial applications at any time and any place [21]. Spatial objects (data) are one of the key components of LBS, as in essence LBS are a kind of data or information services. (In the remaining part of our thesis spatial objects and spatial data are used interchangeably). The locations of spatial objects can be represented by the geometrical model explained in the previous section. A common operation on spatial objects is a search for all objects in an area, for example to find all restaurants that is within 2 miles of a particular point or location. This kind of spatial search occurs frequently in LBS and geo-data applications, and therefore it is important to be able to retrieve objects efficiently according to their spatial location. There has been increasing availability and continuous update of spatial data over the past decades due to the advances in geospatial technology. In the conventional model of LBS, all the spatial data are kept in a central server. Due to the high volume of spatial data and the limitations of wireless mobile devices, management of spatial data for LBS has become an interesting problem for researchers. Mobile Internet access (consumer and business segment) are growing rapidly in recent years. By the end of 2009, voice is overtaken by data

in terms of volume (Tbytes) by Mobile Intranet access which generates the highest traffic volumes.Voice traffic growth should remain limited compared to traffic growth of data from 2010 to 2020.

Due to the high volume of spatial data and its use in LBS in recent years, different data models, query languages and indexing methods have been proposed [123] to effectively retrieve spatial objects. These include B-Trees, R-Trees, Quadtree and KD-Tree etc. Through these spatial indexes, spatial features can be accessed quickly without having to search the entire database. It is generally agreed that among the above mentioned approaches, there is no single approach that can be considered as the best under all circumstances. All of them have their own strength and weakness. To overcome the inherent weakness of all the above approaches, several hybrid methods have been developed. QR-Tree [97] is such a hybrid method, which combines Quadtree and R-Tree to improve performance database update and queries in navigation guidance. Bo Huang and Qiang Wu [21], proposed an improved QR-Tree (iQR-Tree), which increases the access speed and reduces the average response time of the queries.

3.3 LBS Delivery with Client-Server Architecture

The client server model can be defined in this way: Clients request information or a service from a server and that server responds to the client by acting on the request and returning results. In the usual client/server model, one server, sometimes called a daemon, is activated and awaits client requests. An example involving the Internet is a web browser (such as Microsoft Internet Explorer or Mozilla Firefox), which is a client program that requests services (the sending of web pages or files) from a web server in another computer somewhere on the Internet. LBS use the same client-server architecture, where LBS User (Client)

request service from the LBS content provider through the telecommunication

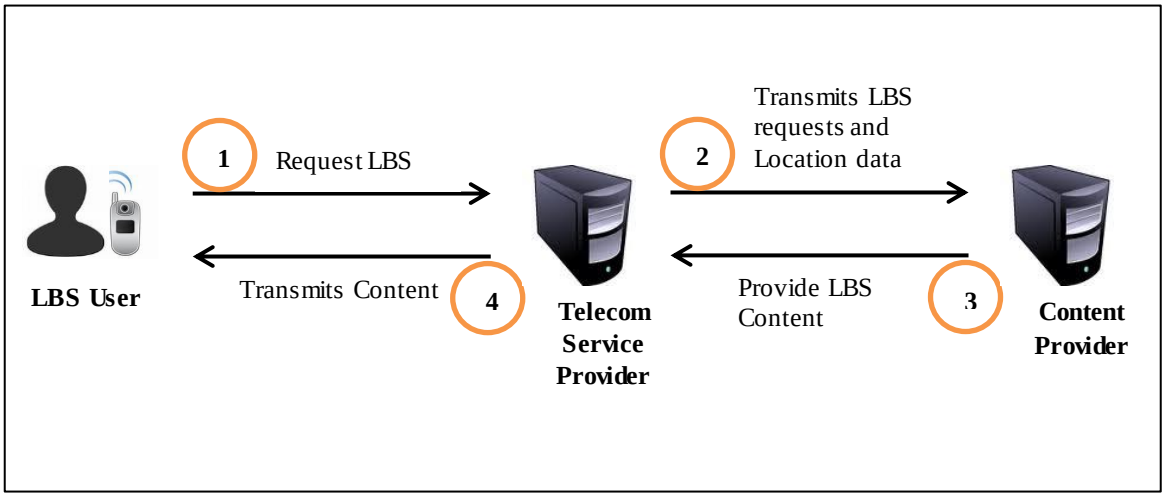

Figure 3.5: LBS Delivery Model

service provider. The structure of the routing used in LBS is presented in Figure 3.5. Following the client/server architecture, LBS are composed of a LBS **client** with his *interface* on one side, and the **server,** with the *database* and some *middleware* on the other side. The web components are distributed in the two sides: *web browser* for the client and *web server* for the server side.

Lot of works have been done on management of data and processing queries in Location Based Services. But, in many cases, a central server is responsible for processing all queries and managing the location data of static and moving objects. Therefore, these proposals are limited to the scalability supported by a single computer. Moreover, it also represents a single point of failure. Moreover, the single computer assumption is not realistic in a large-scale setting. This architecture may be adaptable for services limited to a small geographical area. For example, a centralized computer could probably manage the location information of a fleet of vehicles (e.g., taxis) in a city. However, distributed approaches would be required when there is a large number of moving objects (maybe of different types, such as taxis, buses, people, etc.) and mobile users issuing queries over a large geographical area [66]. Indeed, the moving objects

themselves are distributed through the geographical space by nature. Some proposals already exist to overcome this problem, and future works will probably focus on distributed environments. Our framework deals with such situations by distributing the entire static and moving objects into hierarchical database.

3.4 System Overview

In this section, we first introduce various query models and different classification based on its use in LBS. Then the importance of distributed framework for data placement in local servers as against a centralized server for the service provisioning in Location Based Services is explained. Finally the system architecture and various components of the architecture are explained.

Query types: Queries in Location Based Service can be classified into several categories based on complexity, scope and target objects. In literatures, they are also referred Location Dependent Queries [130]. Information from a centralized broadcasting channel such as BBC world news is an example of Location Independent query. Whereas, localized services such as the nearby restaurants or local weather are examples of Location dependent queries, because they are bound to a location. In LBS, being location is the most important attribute, only Location Dependent queries are considered. Based on the complexity and scope, queries may classified into *Local and non-local queries* and *Simple and General Queries,* and based on scope and target objects it can be classified into *Object Queries and Range Queries* [37].

Local vs. non-local Queries: Local queries are bound to a user's current location (for example, "list the local weather" and "find the nearest restaurant"). Non-local queries are bound to locations other than the user's current location ("find the weather in New Delhi" and "find the nearest station to Taj Mahal," for example). A system can resolve local queries based on the user's local context, whereas for non-local queries, it must first identify the contexts in which it will evaluate the queries.

Simple vs. general queries: Simple queries place simple equality conditions on the underlying data. For example, the query, "download the local weather report," retrieves the weather report associated with the user's current wireless cell. A query like "find the nearest restaurant" can also be viewed as a simple query because "nearest restaurant" is a functional attribute that finds the nearest restaurant based on the user's location. General queries involve complex conditions on the underlying data. They can be further divided into spatially constrained range queries, in which at least one predicate has spatial properties, such as "list the hotels within 10 miles"; and non-spatially constrained queries, which only contain predicates without spatial properties (for example, "list the hotels with a room rate below Rs. 500"). Common spatial operators in spatially constrained queries include *intersect*, *contain*, *within*, and *distance*. Techniques for processing general queries are more sophisticated because the data types (scalar versus vector) and operations are different.

Object Queries: These type of queries target specified objects. Object queries are like traditional type of queries that are issued by explicitly naming the target objects or by giving the object classes and the constraints that must be satisfied. They may be either local or non-local queries. For example "Give me the hotel reservation phone number of the Pleasant Plaza." is an object query since the target object is directly specified. This is a local query, if Pleasant Plaza is located within the same cell of the client. If the target object is not in the neighborhood of the client, or not in the current cell of the client, then the query is a non-local query.

Range queries: These types of queries target objects located within a range restriction and satisfy certain constraints. It is also referred as spatially constrained queries. Range queries are location dependent queries to locate the desired type(s) of object(s) within a specified distance around the client. "List

all the restaurants within 5 miles of my current location." for example, is a typical range query. Range queries normally involve spatial operators. Common spatial operators in range (spatially constrained) queries include *intersect*, *contain*, *within*, and *distance*.

Remote and Local data: Similarly, the data involved in LBS can be classified into Remote and Local, based on the location from where the data are served. Non-local queries normally serve data from remote locations, whereas, local service station such as area map or tourist attractions is an example of local data.

Based on the above classification of Query, data and its typical usage pattern [130], has classified the Information service applications in LBS along four dimensions.

- **Source of data**: *Remote* vs. *local*

  The data can be from a remote database, or from a local service station such as area map or tourist attractions.

- **Period of validity**: *static* vs. *dynamic*

  The period of validity of the data can be relatively static such as various kinds of traffic timetables, or dynamic such as the current traffic condition.

- **Target audience**: *public* vs. *personal*

  The intended target audience of the data can be to the general public such as a public announcement, or to a particular person such as email or personalized news.

- **Valid Scope**: Area or areas within which the query result is valid. Valid scopes are important in developing effective data placement, indexing, cache replacement, and cache invalidation methods. In the geometric location model, a valid scope often takes the shape of a polygon in a 2D space.

3.4.1 Proposed Model with Hierarchical Database

Though several methods have been proposed for efficiently managing spatial data in mobile environment, the spatial data or objects are assumed to reside in a database on a centralized server. LBS queries from clients residing in different wireless cells under different Base Stations (BS) are directed to the central server. After executing the queries, the results are sent back to the clients as depicted in *Figure 3.6*. Since all clients are communicated to the same central server, it poses high communication overhead.

As we discussed above, the spatial data, that are the central point of any LBS query, is increasing enormously due to LBS applications such as Location Based Social networking. Indexing the objects based on locations using the various schemes described above, of course, increases the access efficiency and performance of LBS query provisioning. However, as we mentioned earlier, majority of the schemes consider the data served from a central server. Clearly, this access strategy imposes high communication head and processing power on the LBS system. To overcome the data management issues, several distributed database system architectures have been proposed. As per the

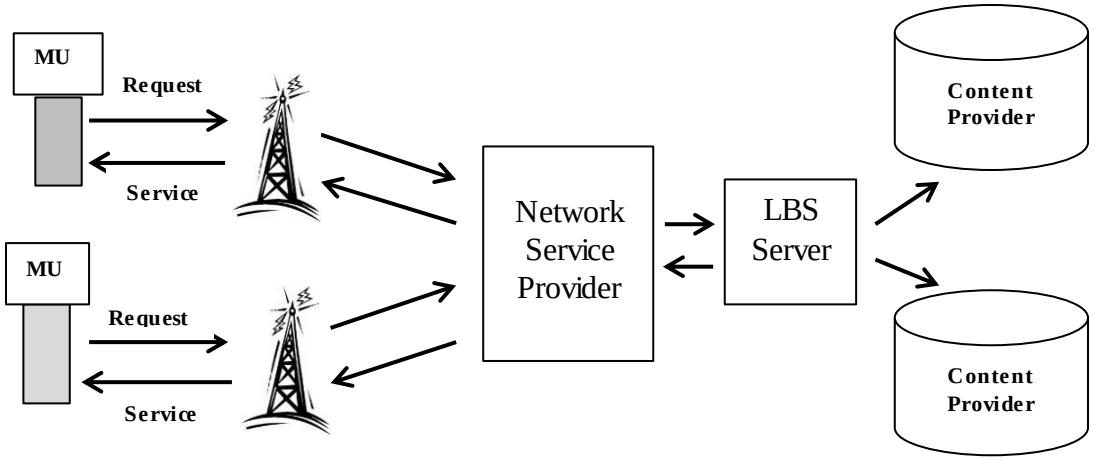

Figure 3.6 Centralized Service provisioning in LBS

scheme, the entire geographical area is divided into several service areas.

Each area may contain thousands of spatial objects which the LBS systems are interested in. The entire world of objects is indexed as a hierarchical structure with increasing resolution in a hierarchical manner. Before detailing the proposed method, different types of data and query methods in LBS are to be explained.

3.4.2 System Architecture

The cellular architecture is modeled by a homogeneous hexagonal network coverage model as shown in *figure 3.7*. Each cell is assumed to have at least one Base transceiver station (BTS). These cells are connected each other with a Base Supporting Station (BSS). The mobile client within a cell

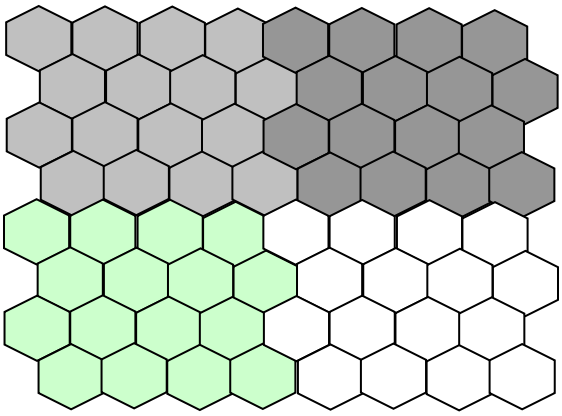

Figure 3.7: Regions representing location areas (LA) and individual cells (in this figure there are 4 LA, each consisting of 16 cells)

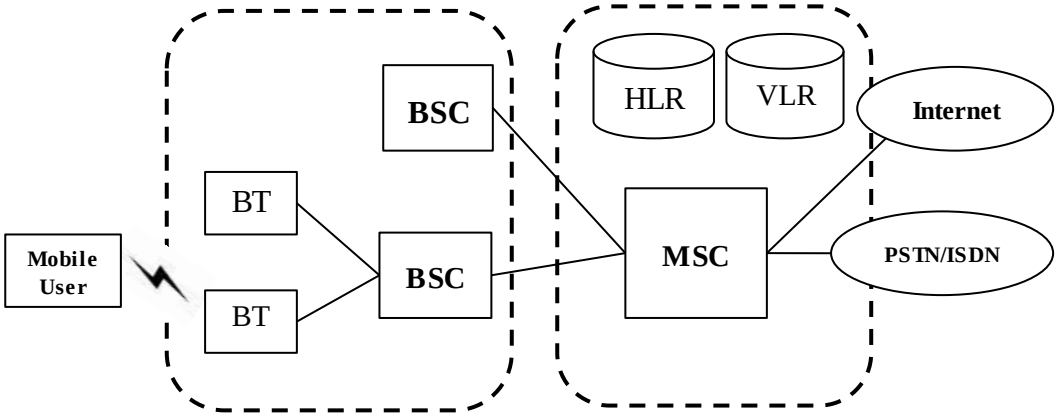

Figure 3.8: Mobile Communication System

communicates with the BTS of the cell through wireless channel as in Figure 3.8. As per our distributed architecture, it is assumed that a spatial database is attached with each cell. The databases of different neighboring cells are organized into a hierarchical tree structure as shown in Figure 3.9. It is also assumed that each internal node (database) contains the cumulative information of its sub-nodes. Proper update methods such as replication have

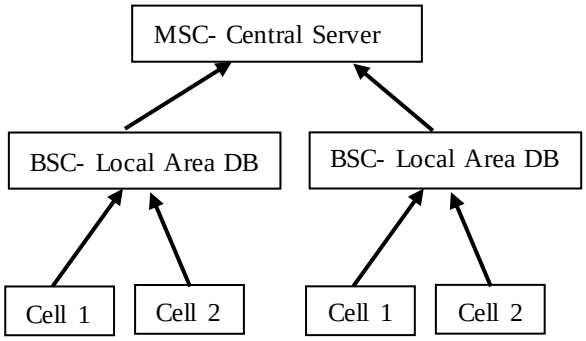

Figure 3.9: Hierarchical Tree Structure

to be devised for maintaining up-to-date information in all databases.

In on-demand access, a mobile client submits a request, consisting of a query and the query's associated location, to the server. The server locates the appropriate data and returns it to the mobile client either individually or by multicast. As the most fundamental data access mechanism, on-demand access can be used for all types of queries. We have identified three main issues in on-demand access: data placement, data replication, and query classification.

3.4.3 Data Placement in Distributed Architecture

The first issue developers must face when dealing with location dependent data is where to place them in the system architecture. In a centralized solution, all data go into a central database (*for example, the root in Figure 3.9*).

Obviously, this scheme is unattractive in terms of performance and robustness because the database can become a bottleneck or a single point of failure.

Alternatively, we can use a distributed placement solution. Under this scheme, the local server is the data manager and wireless information server for a single cell. Each cell is assumed to have a unique local server which provides services to all clients within that cell. The local server of a cell manages the information of the data objects within that cell.

A higher level server keeps the information of all data objects of all cells that come as the child nodes of that server to form a hierarchy. For example, in Figure 3.10, we would store data objects of cell 1 in the local server of cell 1 and data objects that fall into cells 1 to 4 at Server 1. In our architectural

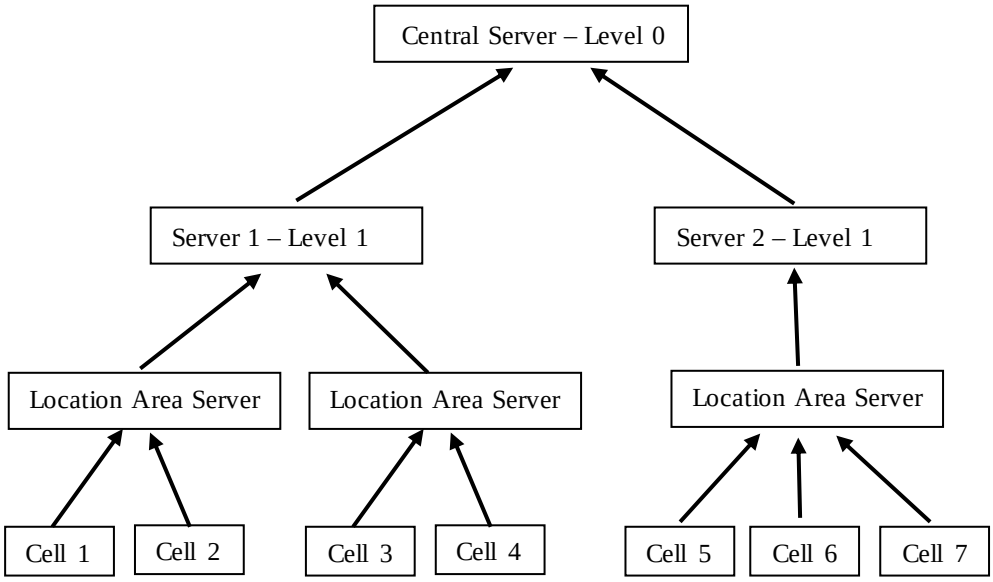

Figure 3.10: Hierarchical Data Placement

design, a client always sends requests to the local server of the cell where the client resides. The target objects may be available right at the client cache, at the local server of the same cell, or from appropriate higher level servers, if the scope the query overlaps with the neighboring cells, or from central server.

The two data placement solutions represent valid scope of data objects and process queries differently. A centralized solution is not suggested as it cannot scale to a large system because it is impractical for the central database to store the valid scopes of all data objects and serve queries from all cells. In a distributed solution, the data objects are distributed across many cells, which

ease the storage demand on a single cell. Furthermore, data placement partly implies the valid scope of data objects. For example, in Figure 3.10, if we use a cell-based symbolic location model, a data instance placed in a base station implies that it is valid for that cell and thus no explicit valid scope need be stored. A data instance stored at wired server 1 implies that the instance is valid in some or all of cells 1 to 4 but invalid in cells 5 to 7. Thus, we can use a more compact representation. In summary, a distributed method results in a more uniform workload and more efficient querying; however, managing valid scopes is more complex. The data placement solution used also affects query processing. In a centralized solution, the centralized data server processes all types of queries, although not always efficiently. Although a distributed solution simplifies processing of simple queries, it complicates the processing of general queries. To process a local simple query, the system searches the servers from the corresponding base station up the database hierarchy until it finds the relevant instance. The length of the path from the query point to the solution is typically quite short. Moreover, the smaller candidate set and less resource contention make query latencies in distributed solutions shorter than in centralized solutions. The simplest way to process a non-local simple query such as “find the local news for New Delhi” is to redirect it to the specified remote cell(s) and then follow the same process used for a local query. On the other hand, a general query such as “find the nearest hotels with a room rate below Rs. 500” can incur an evaluation of the query on many nodes, because, for example, the system might need to process such a query at several distributed nodes and combine the intermediate results returned from the nodes to get the final answer.

3.4.4 Collocated Proxy

The architecture described above should address the following question: How should the system process various types of queries? As discussed earlier, there are object queries, range queries, map queries etc. The information about the static and mobile objects inside the cell is maintained by the database server of

each cell. The database server manages the location data of mobile users and static objects, such as gas stations, restaurants, etc. Location queries are registered and evaluated at the database server. In the case of object queries, the targeted object may be available from the local server or from other higher level servers, depending on the spatial distribution of objects. So, in the case of a range query, whose scope is fully within a cell, query is processed by the local server, whereas in the case of a range query, whose scope overlaps with the neighboring cells, the local server of that cell cannot process and hence is directed to the higher levels. What technique can be used to classify the queries? Who will direct queries to process in the local servers as well in remote higher level servers? What kind of caching mechanism is used?

To address these questions efficiently, we introduce the concept of Collocated Service Proxy. Each local database of the cell is attached with a service proxy server. When a user issues a query to the database server, it creates a service proxy object for this user, if there is no proxy already exists for this user. It assumes that the proxy is location-aware with GPS enabled devices. The service proxy performs tasks such as accepting data request to access mobile services on behalf of the user, convert query into standard SQL statements, calculate the spatial intersection of the query scope with the neighboring cells, search the local database for the cached result that matches the request, forwarding the query to the higher level if the query scope (range) overlaps with the scope of the neighboring cells., formatting the data request to various application formats, and forwarding the result to the mobile user. The personal proxy, that is, the proxy created on a per-user basis, is a "mobile proxy" which moves with the user during location handoffs to minimize the network signaling cost while maintaining the required quality of service. The leaf nodes of the tree structure are the local database of each cell, and these databases contain the information about the data objects of that cell.

3.4.5 Proxy Architecture

The proxy object plays an important role in our dynamic data management system. When a user wants to submits a query to the DMS of the local server, server creates a proxy object for that user. Then, the proxy will perform tasks such as accepting service request (query) from the client, analyzing the scope of the query, routing the request to the local or parent servers according to the scope of the query and returning the results to the client.

The proxy cooperates with the underlying location management system, so that it is location-aware, and knows the location of the mobile user when the

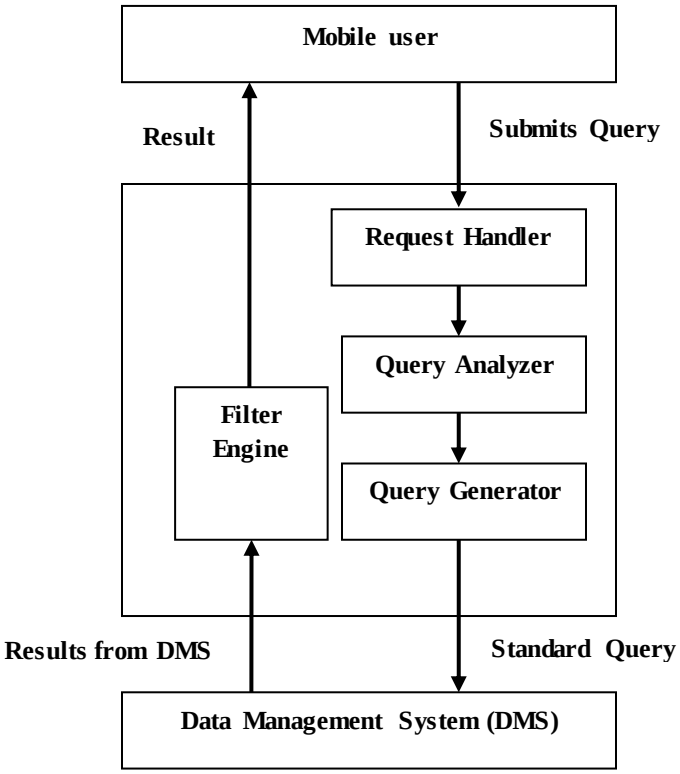

Figure 3.11: Proxy Architecture

client submits a query. It is also assumed that the proxy is aware of the scope of the current cell. All service requests from the client are routed through the proxy to the database.

Proxy has the following components

- Request Handler
- Query Analyzer
- Query Generator
- Filter Engine

The main steps involved in each component is explained in the following section

3.4.6 Query Processing and Data Dissemination Model

In this section, we explain the main steps of our query processing approach.

The user device contains an application that is used to issue queries as well as keeps the retrieved data up to date. When the user issues a query, it creates a user proxy agent at the local server collocated with local database and the query is processed according to the following steps as shown in Figure 3.11.

Request Handler receives the query and transforms the query to intermediate specifications in the following way

i) It obtains the location of the querying object (reference object)

ii) ii) It obtains the target class objects from the query

iii) iii) It also obtains the time and maximum speed of the reference object. This is useful for calculating the buffer area that can be included in the result set based on a predictive calculation. However, we are not considering these parameters in this thesis.

Query Analyzer handles two things. First it identifies the spatial area of the query from the location dependent constraints of the query. During this step it checks whether the query boundary overlaps with the neighboring cells. Each local server and DMS is aware of its location area and location area of its

neighboring cells. If the query boundary is completely within the cell, it is executed by the local server. If the boundary intersects with neighboring cells having the same location area, then it is routed to the next higher level. If the cells are having different location area then it is passed to the topmost level as shown in Figure 3.10**.** The information about the boundary intersection and query specification is forwarded to the query generator.

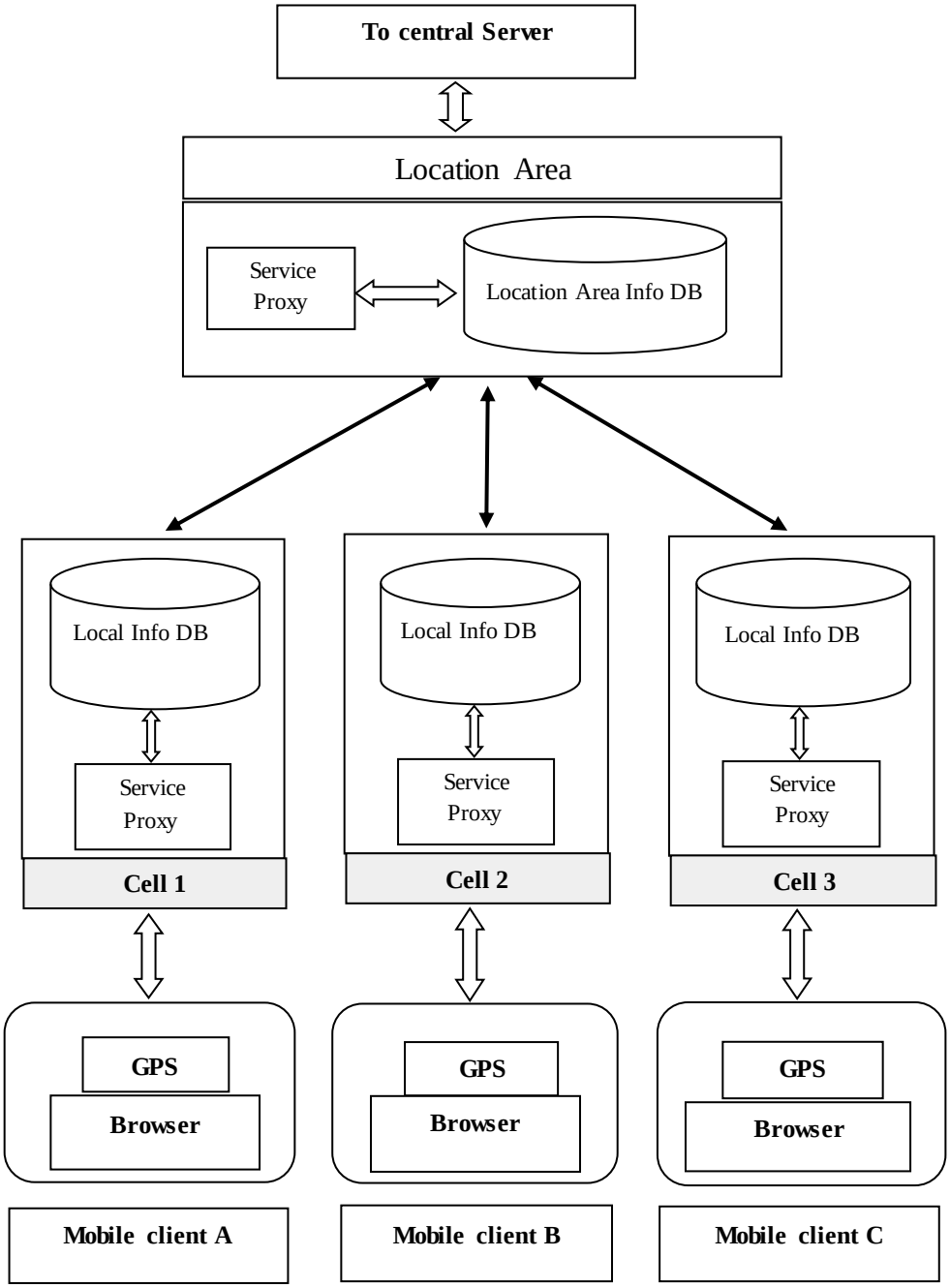

Figure 3.12: Service Provisioning Architecture for LBS using Proxies

The query generator submits the query to the appropriate DMS based on the information obtained from Query analyzer along with the query specifications. It generates appropriate type of standard queries suitable for the DMS. (Spatial query for spatial database or standard SQL queries for relational databases). It also selects the DMS, to submit the query as we described before. If the query boundary falls within the cell, then the query is submitted directly to local DMS, otherwise it is routed to the higher levels with the information given by Query Analyzer.

The result that contains results of objects from the buffer area, based on the predictive calculation. However in this paper, we are not using the predictive calculation. Filter engine filters out the result from the buffer area and only relevant result is presented to the user.

3.4.7 Proxy Service Provisioning

The service provisioning architecture is shown in Figure 3.12. Each Mobile Station (Mobile user / Client) communicates with local database of the system through the collocated personal proxy. The request for service is processed as mentioned in the above section and the result is returned to the Mobile Station (MS).

Hierarchical Replication Based on Location: Data replication is used to improve reliability and reduce costs, such as network traffic and response time. Using data replication, the system creates one or more copies of the data, which it places at different locations in the network. A key issue in data replication is placement—that is, what are the replication units, how many replicas should the system create, and where should it put them? To apply these solutions to the LBS replication problem, we must estimate the access patterns for each data instance at the cell level. This is relatively easy for on-demand access but difficult for broadcast data.

In LBS, location dependency provides for an innovative replication scheme. In our framework, the database is organised in a hierarchical structure. All base

data i.e., location independent data is fully replicated at the first level server (Central Server). At the top layer, we only replicate base data; data that do not depend upon location. At the next higher level, we have data that are the same for that set of locations and so on. At the lowest level the server will keep only data related with its cell. Thus, we can replicate data in a hierarchical fashion. This strategy expedites the data access for queries related with the lower levels.

In addition, for location-dependent data, access patterns might be time dependent and periodically repetitive. For instance, people tend to access local weather in the morning and news in the evening. Due to time differences between areas, the access pattern probably slowly migrates from one cell to another. Existing automatic replica migration protocols can adapt to user access behavior changes in distributed systems and Internet environments. We do not yet know, however, whether these protocols would work well for location-dependent data. We do not address this issue further in this section.

3.5 Analysis

Since the LBS involve heterogeneous technologies, the research on LBS has different dimensions. These intertwined technologies necessitated the work of numerous standardisation bodies on producing the interoperable standards related to LBS technologies and services. To name a few, the following organisations presently contribute to the LBS-related standards: the Third Generation Partnership Project and its second initiative (3GPP and 3GPP2); the Open Mobile Alliance (OMA); the European Telecommunications Standards Institute (ETSI); the GSM Association (GSMA); the Internet Engineering Task Force (IETF); the International Telecommunication Union (ITU); the European GNSS Supervisory Authority (GSA). The number of involved standardisation groups indicates a substantial increase in the common interest devoted to the LBS field in its current development state [130]. But still the field lacks a standardization and platform for evaluating and comparing the research findings. In most of the cases simulations are used, which vary significantly due to the heterogeneous technological

intersection. There are certain standards proposed for the analysis of our framework. The framework may be analyzed based on the Utility, Accuracy, Latency, and Feasibility of the system. The Utility of the framework is defined according to how much the system is useful to deliver the service to the client. The framework has the ability to perform a task with precision which shows the Accuracy. Latency is referred as the time taken to receive an answer by the client after the query has been issued. Feasibility is referred as to how it is practically adaptable to the present technology and useful to the mobile users. In the following chapters, the framework is used to implement the range query processing and peer-to-peer architecture for privacy preserving query processing.

The framework has several advantages and some of the success factors and their description are listed below

- Latency: Since the data are stored and accessed from the local server, time taken for query processing is always less than the time needed to access a central server.
- Accessibility: The user gets the frequently accessed data from the Local Server itself to avoid data loss and to maintain efficiency.
- Simplified Data Maintenance: Since the data are partitioned based on the geographical location and services, the maintenance of the data is easy.
- Data volume: It is designed to keep the data both in local server and central server
- Scalability: The framework is flexible enough to function well when it (or its context) is changed in size or volume in order to meet a user need.
- Query processing: Query classification is done by the proxy server, which makes the framework user-friendly.

3.6 Related Works

The key element in Location Based Services is the Location Dependent Data (LDD). Appropriate architecture is necessary to dynamically manage the static and moving Data for the efficient processing of Location queries in Location Based

Service. Some works have been done towards the developing appropriate architectures for storage and management of location-dependent data. One of the initial works was done by Imielinski and Nath [68], the term *dataspace* was used to denote a new kind of physical space that is connected to the network. As per the architecture, data is stored where it is produced, and so it is completely distributed. Thus, for example, messages can be attached to locations (e.g., sensors) and users can query these locations to obtain different types of location-dependent data (e.g., pollution information, announcements, etc.).

SHIOW-YANG WU and KUN-TA WU [130] proposed a service framework, system architecture and dynamic data management strategies for location based services in mobile computing environments. A technique, called *neighbouring replication*, to store replicas of data in different servers with the goal of increasing the efficiency of range queries which may span the spatial boundaries of several local servers. Several strategies for the dynamic data management were introduced which balances the update cost and query cost.

Gupta et al. [55], suggested a hierarchical framework for handling queries, with a focus on data distribution issues. He proposed a hierarchical framework for handling continuous queries on location-dependent data from a single zone (*single-ZQ*) or a set of neighbour zones (*multiple-ZQ*). Similar type of work was also proposed Madria et al. [95] and Ryu et al. [124].

Some works proposed caching methods for queries on location-dependent data. Zheng et al. [155], Gao and Hurson [41], and Hu et al. [63] proposed different kinds of frameworks for caching the location dependent data for efficient storage and processing in LBS. Ku et al. [83] studied data caching for location-based spatial queries in *P2P*environments. Chow et al. [32] proposed a *mobile cooperative caching* to share query results and reduce the communications with the spatial database server.

Nikos Pelekis et al. designed a framework called "HERMES" for Location Based Data Management [108]. Yubin Xu et al. proposed a model for LBS based disaster and emergency management [20]. Stevenson G. et al. have done a framework

"LOC8", a location model supporting a range of expressive representations for spaces, spatial relationships, and positioning systems. The authors constructed a programming framework for exploring location data's multifaceted representations and uses [54]. Maria Luisa Damiani et al. proposed a framework "PROBE" for the personalized cloaking of private locations [99]. MobIS: A pragmatic framework for location based services provides portable and easy to use mobile LBS framework for developing mobile applications and maintaining content [82].

3.7 Conclusion

In this chapter, we proposed a dynamic data management system with a hierarchically distributed architecture, in which every mobile user interacts with the lowest level of database hierarchy. We introduce the concept location aware mobile service proxy, which plays a pivotal role in classifying and routing queries to the different level of the hierarchical database. The architecture assumes that the proxy is location-aware, means that the proxy knows the position of the client whenever the client requests for any kind of services. The service proxy performs tasks such as accepting data requests to access mobile services on behalf of the user, search the local database for the cached result that matches the request, forwarding the query to the higher level if the query cannot be satisfied by the local database, formatting the data request to various application formats, and forwarding the result to the mobile user. The personal proxy, that is, the proxy created on a per-user basis, is a "mobile proxy" which moves with the user during location handoffs to minimize the network signaling cost while maintaining the required quality of service. We have developed algorithms and protocols for accepting and routing data request, searching and caching the results, moving the proxy with user during location handoff to minimize the network signaling cost and continuity of the service. We evaluate the efficiency of our approach using simulation and compare it with the conventional centralized server approach and a

central service proxy. Further, this framework is used to implement the Range Query Processing technique used in the next section.