

NEXT-GENERATION LOCATION-BASED SERVICES FOR MOBILE DEVICES

The logo for CSC, consisting of the letters 'CSC' in white on a red rectangular background.

Sidney Shek
CSC
sshek2@csc.com

CSC Grants

February 2010

ABSTRACT

A Location-Based Service (LBS) is a mobile computing application that provides information and functionality to users based on their geographical location. First-generation examples include “show me nearby restaurant”-type applications. Next-generation LBS can provide additional benefits for users and service providers, including:

- Proactively pushing only relevant information to users to help speed up decisions and activities.
- Minimising tedious data entry by integrating applications with advanced sensors such as accelerometers, digital compasses and cameras.
- Allowing service providers to model user behaviour based on their location and route information, which can support improvements of service levels in near real-time and over the longer term.
- Encouraging sharing of location-based information, such as photos and reviews, generated by other service providers and users.

As a result, analysts predict explosive growth in the LBS market over the coming years. Apart from the consumer market, there is a good prospect and potential for this technology to find its application in many industries including health, manufacturing, mining and financial services.

The aim of this research was to provide further insight into possibilities for smartphone next-generation LBS applications, including a greater understanding of implementation and architectural considerations, and future trends in LBS.

This report outlines the current state of LBS technology, research into the important issue of maintaining user privacy, and future trends such as improved visualisation through augmented reality and moves towards context-based computing and Semantic Web.

Four case studies of next-generation LBS are discussed in detail, covering architecture, design challenges and possible solutions. These case studies represent a mix of public and enterprise LBS. A prototype iPhone application for mobile insurance brokers was developed to demonstrate how the above-mentioned next-generation LBS benefits could be applied to enterprise scenarios.

The future of LBS in the consumer and enterprise spaces promises to be very exciting; the ultimate goal of true context-aware computing may not be far away.

Table of Contents

1	Introduction	1
1.1	Benefits of Location-Based Services.....	1
1.2	The Growing Market for Location-Based Services	2
1.3	Project Aims and Outcomes.....	3
2	Background on Location-Based Services.....	4
2.1	Timeline of Location-Based Services	4
2.2	Classifications of Location-Based Services.....	5
2.2.1	Target market	6
2.2.2	Application types	6
2.2.3	Technical capabilities	7
2.3	Characteristics of Next-Generation Location-Based Services.....	8
3	LBS Components and Technologies.....	10
3.1	Location Collection Technologies.....	12
3.1.1	GPS-based solutions.....	13
3.1.2	Mobile network-based solutions	14
3.1.3	Wi-Fi based solution	15
3.2	LBS Middleware Standards.....	15
3.2.1	OpenLS	15
3.2.2	XML (GML, KML) standards for location data.....	16
3.3	Technologies for Storing Location Information	17
3.4	Mobile Device Technology for Next-Generation LBS	19
3.4.1	Easy access to location information	20
3.4.2	Push notifications	20
3.4.3	Availability of sensors	21
3.4.4	Enterprise integration and other features	21
4	Privacy and LBS.....	22
4.1	Why is Privacy a Concern with LBS?	22
4.2	Identification Requirements of LBS	23
4.3	Privacy Solutions.....	23
4.3.1	Anonymisation.....	23
4.3.2	Cryptographic techniques	24
4.3.3	Accuracy filtering	24
4.3.4	Transformation of location request data	25

4.3.5	Privacy policies.....	26
4.4	Addressing Privacy Concerns for Enterprise LBS Users.....	26
4.4.1	Encouraging uptake of LBS via incentive schemes.....	26
4.5	Future Trends in Privacy Research.....	27
5	LBS Case Studies.....	28
5.1	Case Study 1 – Mobile Insurance Broker Application (MoBroker).....	28
5.1.1	Use case scenarios.....	29
5.1.2	Application architecture.....	36
5.1.3	Challenges and possible solutions.....	38
5.1.4	Possible extensions to the prototype.....	40
5.2	Case Study 2 – Public Transport Connection Times application.....	40
5.2.1	Possible application architecture.....	41
5.2.2	Challenges and possible solutions.....	44
5.3	Case Study 3 – Building Induction Manager.....	45
5.3.1	Challenges and possible solutions.....	46
5.4	Case Study 4 – Building Evacuation Assistant.....	47
5.4.1	Challenges and possible solutions.....	47
5.5	Summary.....	47
6	Future Trends Beyond Next-Generation LBS.....	48
6.1	LBS Middleware Service Providers.....	48
6.2	Improvements in Visualisation.....	49
6.3	Context-Aware Computing.....	50
7	Conclusion.....	51
	Bibliography.....	52
	Appendix A – Discussion of Design Decisions in the Mobile Insurance Broker Prototype.....	56
A.1	Implementation of Location Service and Data Store.....	56
A.1.1	Database provider selection.....	56
A.1.2	Accessing spatial database features via Object-Relational Mapping.....	57
A.1.3	Considering use of component-based architecture.....	57
A.2	Client-Server Communications.....	57
A.2.1	Server-side implementation.....	58
A.2.2	Client-side implementation.....	59
A.3	Inertial Navigation System.....	59
A.4	Camera Overlay with Template Photos – “Augmented Reality”.....	60

List of Figures

Figure 3-1 Diagram showing components of an LBS	10
Figure 3-2 Diagram demonstrating a simple B-tree.....	17
Figure 3-3 Diagrams demonstrating an R-tree.	18
Figure 4-1 An example of post-filtering as suggested by [Patrikakis, Voulodimos and Giannoulis, 2009]..	25
Figure 5-1 Screenshot of the overall map view of premises to visit.....	29
Figure 5-2 Screenshot of customer's premises details	30
Figure 5-3 Screenshot showing the question list for the customer's premises with information about the most recent answer	31
Figure 5-4 Screenshot showing a view of the question and summary of answers	31
Figure 5-5 Screenshot showing details of a previous answer.....	32
Figure 5-6 Screenshot showing option to use a template photo when taking a picture	33
Figure 5-7 Screenshot showing list for selecting a template photo to use.....	34
Figure 5-8 Screenshot showing the camera view with a template image overlaid.....	34
Figure 5-9 Screenshot of the map entry screen while in mapping mode.	35
Figure 5-10 Screenshot of the map display screen.	35
Figure 5-11 Screenshot showing the process of requesting assistance from another broker...36	
Figure 5-12 Screenshot showing a push notification being received by the target broker	36
Figure 5-13 Application architecture for the Mobile Insurance Broker LBS	37
Figure 5-14 A possible application architecture for public transport connection times LBS.....	42

List of Tables

Table 2-1 Timeline of key events in the development of LBS.....	5
Table 3-1 Description of LBS application components	11
Table 3-2 Summary of GPS-based location collection technologies	13
Table 3-3 Summary of mobile network-based location collection technologies.....	14
Table 3-4 Summary of Wi-Fi-based location collection technologies	15

Table 5-1 Description of the modules in the application architecture for the Mobile Insurance Broker LBS example38

Table 5-2 Description of modules in the application architecture of the public transport connection times LBS.....43

Table 5-3 Technical challenges and possible solutions for the public transport connection time LBS45

Table 5-4 Challenges and possible solutions for the building induction manager LBS.....46

ACKNOWLEDGMENTS

I would like to express my appreciation and gratitude to the following people for their support, encouragement and feedback during the project:

Paul Gustafson, Bob Hayward, Huseyin Erkoru, Kang Tran, Vivek Srinivasan, Chris Broadfoot and Albert Tang

1 INTRODUCTION

A Location-Based Service (LBS) is a mobile computing application that provides services to users based on their geographical location. These can range from simple “find the closest train station” uses to “tell me when my friends are nearby,” and potentially very soon “tell me if there is a sale in a shop when I walk past.”

First-generation LBSs were reactive and client-server focused; users would ask an application or a system for information and receive a response. With the advent of push notification mechanisms, improved mobile Internet access and a move to collaboration as part of Web 2.0, the next generation of LBS is more proactive and interactive between users. For example:

- Relevant information can be sent to users based on their location rather than users having to manually search for it.
- People can share photos and reviews that are tagged with information in a more peer-to-peer fashion. Data sources are more diverse, coming from various service providers rather than just one.

The focus of this research was on this next generation of LBS.

1.1 BENEFITS OF LOCATION-BASED SERVICES

LBS, especially next-generation LBS, provides many benefits for users and service providers including:

1. It aids in filtering the vast amounts of material available on the Internet into relevant information for the user's current context. Users can see important information, enabling them to make informed decisions on the spot, even simple ones such as choosing the best restaurant in an area.
2. By pushing relevant information out to users, it not only supports timely presentation of data, to help speed up decisions and activities, but also it could highlight information that users may not normally be aware of; for example, an application could warn people before they enter a high crime rate district or encounter a temporary road closure or traffic incident.
3. It reduces the amount of manual data entry required by users to access a service; LBSs can automatically obtain location information and other data from sensors on small form-factor devices like smartphones.
4. By sharing location-tagged information, more up-to-date localised information is available to all users.
5. The movement of users (i.e., their location trace) combined with associated tagged information also represents a vast data source for service providers to build models for improving service. For example, researchers at the UCLA Center for Embedded Sensing Systems have developed systems to capture users' movements and modes of transport, thereby allowing a visualisation of carbon footprint by user and by area [Estrin, 2009].

1.2 THE GROWING MARKET FOR LOCATION-BASED SERVICES

As smartphones like the Apple iPhone, Google Android-based devices and Research In Motion's BlackBerry have become increasingly popular, so has the use and ubiquity of LBS. Frost and Sullivan predict 20 percent of users will make use of LBS by 2012 [Vaughan-Nichols, 2009]. Juniper Research [2010] has predicted the size of the mobile LBS market to exceed \$12 billion by 2014. Much of this comes from application sales, with mobile advertising set to drive growth over the next five years. These figures show a large and growing consumer market.

LBS can provide much benefit for enterprise operations such as asset tracking and locating. Realising this potential, many organisations are jumping on the bandwagon using specialised hardware.

However, as more smartphones are deployed into the enterprise, the advanced sensors and features of these devices, such as inbuilt GPS receivers, can be used to provide even more cost-effective LBS to workers and customers. Below are some examples of how next-generation LBS can be applied to various industries:

- **Government – Election day polling booth finder:** Electoral commissions can provide an application to voters to help find their closest polling booth with the minimum wait time, or voters could be sent a notification of when is the best time to vote. Commissions can also gather information on voter numbers at each location, and redistribute voters in near real-time. This could improve throughput on election days and reduce voter frustration over queuing times.
- **Utilities – “Dial before you dig” LBS:** Many utilities have underground cables and pipes that could be accidentally damaged by digging machinery such as backhoes, and this could also endanger lives of workers. While tradespersons should contact appropriate authorities to confirm where to dig, it may be hard to visualise the exact location of pipes and cables based on a map or description. An LBS could be developed using augmented reality to overlay a map of the location of underground pipes and cables over live camera images.
- **Emergency services – Bushfire evacuation LBS:** Early warnings about nearby bushfire fronts are crucial for survival of people and animals during these natural disasters. An LBS could send updates to registered residents in bushfire-prone areas about approaching fire fronts, allowing them to enact their bushfire plans and evacuate as early as possible.
- **Manufacturing and Mining – Equipment inspection log:** To support regular inspection or maintenance of equipment, an LBS could be developed to assist engineers in finding relevant equipment and bringing up relevant information on site such as equipment logs, instructions and warranty information. Updates to equipment logs or requesting spare parts can be done on site electronically, avoiding long turnaround time and minimising paperwork. An advanced application could also send a notification to the closest maintenance engineer when a piece of equipment either has failed or is predicted to fail in the very near future, so that appropriate action can be swiftly taken with minimum disruption to the operation.
- **Health – Ambulance notifier:** For people with known medical ailments, who are fragile and home alone, an application could be developed to detect when an incident has occurred and trigger an emergency call for an ambulance, with information about the

current location and patient details. For example, when an accelerometer detects a sudden fall followed by lack of movement, the patient may be unconscious and hence may need assistance. This type of application could reduce emergency response time and be life saving.

- **Health – Doctor’s patient list:** An LBS could be developed to support medical specialists that service multiple hospitals by presenting a list of their patients in the hospital that they are supposed to look after. This could include patients that were admitted overnight that doctors were not aware of. The application could also provide the exact location of patients, including floor level and bed number. This kind of application could improve service levels by reducing patients’ waiting time. A feature could also be added to allow users to find nearby medical specialists if additional consultations are required.

1.3 PROJECT AIMS AND OUTCOMES

Motivated by the opportunities described above, the aim of this research grant was to investigate development of next-generation LBS. This included:

- Gaining further insight into possibilities for smartphone next-generation LBS applications, especially for enterprise scenarios. An understanding of key differentiators between first and next-generation LBS was developed.
- Gaining an understanding of components of LBS and underlying technologies. This included location collection technologies such as GPS, relevant middleware standards, storage of location-based data, and technologies on current smartphone platforms (review).
- Developing a prototype iPhone LBS application (MoBroker) based on a case study of a mobile insurance broker. This served as an example of using next-generation LBS concepts to streamline business processes, and aided in understanding the technical challenges of building an enterprise LBS application.
- Further investigating LBS application architecture, design challenges and possible solutions through thought experiments on three other case studies.
- Investigating the challenge of protecting the privacy of LBS users, and proposed solutions from academic literature. This also included investigating possible mechanisms for encouraging uptake of LBS in enterprises.
- Gaining insight into future trends of LBS technology.

This report presents the findings of the research.

2 BACKGROUND ON LOCATION-BASED SERVICES

This chapter provides some background information on LBS. This includes a timeline of key events in history leading to next-generation LBS, classifications of LBS to help distinguish between them, and some typical examples of LBS.

2.1 TIMELINE OF LOCATION-BASED SERVICES

LBSs represents a convergence of several technologies: location collection, map visualisation, and applications on mobile devices. Below are some key events in the development of these technologies that have led to LBS; these may not be the first implementations but they represent the popularisation of the technology. The rows are coloured based on the related technology (**blue** for location collection, **orange** for applications on mobile devices, and **green** for map visualisation).

Date	Event	Justification
1996	The White House authorises that Selective Availability of GPS signals will be phased out [Humerfelt, 2009]. This was implemented on 2 May 2000 [Wikipedia Global Positioning System, 2010].	This gave civil applications access to high-precision location information that would be useful for navigation and LBS.
1998	In the USA, cellular network providers are required to provide the location of 911 calls as part of Enhanced 911. Initially callers can be located to within a cell [Oakes, 1998].	Primitive location information on all mobile phone users in the US became available, allowing cellular networks to provide location-based services (e.g., go2's local search application released in May 2002).
2000	Java Micro Edition (Java ME) standards first approved [Java Community Process JSR 68, 2010].	This represented a major step in allowing application developers to write programs to run on embedded systems including mobile phones.
2003	JSR 179 Location API reaches final release [Java Community Process JSR 179, 2010]	This represents the first API standard for providing access to location information on Java ME based devices.
2004	Qualcomm announces first successful test of Assisted GPS on mobile phones [www.3G.co.uk, 2004].	GPS on mobile phones means reduced reliance on cellular networks to provide location information, and also potentially improved accuracy where GPS signals can be used.

Feb 2005	Google Maps goes live. Google Maps API is released shortly afterwards in June [Google, 2010].	Although other map sites, such as MapQuest, became available in 1996, Google Maps provided a highly interactive and user-friendly interface that surpassed other sites. Also, the Google Maps API allowed Web developers to integrate maps on their site for many types of mashups and geotagging of information.
2006	Google Maps for Mobiles released [Wikipedia Google Maps, 2010].	This allowed Google Maps to be run on any Java-based mobile device.
June 2007	Apple releases the first iPhone [Wikipedia History of the iPhone, 2010]	Although it was not the first device to incorporate many features such as GPS and accelerometers, it popularised the concept of smartphones for consumers with applications such as Maps. This was possible through the change in philosophy to a large touchscreen interface based on finger gestures.
August 2008	Apple releases the App Store for iPhone [Wikipedia App Store, 2010]	The App Store and iPhone SDK encouraged the development of applications to run on mobile phones by allowing developers to earn money and by giving users an easy means of finding and purchasing applications.
October 2008	The first Android-based phone is released (HTC Dream) [Wikipedia HTC Dream, 2010]	The release of Android promised to allow many manufacturers to produce devices that rival the Apple iPhone. This is witnessed by the large number of devices now available.
June 2009	IBM Seer application for Wimbledon released, running on Wikitude augmented reality browser [Mobilizy, 2009].	This represents a major application of new visualisation techniques for location-based data.

Table 2-1 Timeline of key events in the development of LBS

2.2 CLASSIFICATIONS OF LOCATION-BASED SERVICES

There are many examples of LBS, each with different technical challenges. It would be helpful to classify LBS into categories in order to identify their similarities. Classifications of LBS based on *target market*, *application type* and *technical capabilities* are discussed below.

2.2.1 Target market

There are three general target markets for LBS: **publically accessible for the mass market**, **publically accessible for niche markets**, and **internal enterprise applications** [Rao and Minakakis, 2004]. The characteristics of these categories are described below:

1. **Publically accessible (Mass market):** These applications are aimed at general consumers, such as “find me the closest restaurant” applications. Specific user registrations to use these applications are not required. These LBSs must be highly scalable to handle large numbers of requests, and performance must be reasonable for the location to be covered. For example, the design of the application should take into account that users in Australia may be accessing servers in the USA and hence may experience significantly greater network latency. It is also important to maintain high availability as there is no specific list of users to warn if the system needs to be shut down for regular maintenance; public users would only see an unavailable system and the bad experience may lead them not to return.
2. **Publically accessible (Niche market):** These applications are still aimed at the public but are targeted at specific customers. For example, past customers of a shop or retail outlet may be notified of specials or sale items when they walk nearby. The performance, scalability and availability issues are less important in this case. Instead, ensuring privacy of users becomes a major concern.
3. **Internal enterprise applications:** These applications are used internally inside an organisation. Some common examples include supply chain management and inventory tracking, personnel tracking and information retrieval. Many of these applications may warrant the use of custom hardware such as RFID tags or special mobile devices. However, with the advent of the latest smartphones, there is an opportunity to reduce the need to purchase custom-made devices. Using mobile phones in this environment raises privacy issues of employees, as well as technical challenges related to protecting private internal data in a public mobile phone network.

2.2.2 Application types

LBS can be classified based on the main purpose of the application. Below are some examples that cover many of the LBSs available today:

- **Navigation and routing:** These are perhaps the most recognisable applications. They provide directions to users to navigate from an origin to a destination, possibly with specific instructions based on mode of transport.
- **Entertainment:** Many games and social networking services can make use of LBS. For example, the Twitter application can capture the location of tweets. Some treasure-hunting-style games have also been created.
- **Information services:** These provide information to users. For example, a mass market application may be “find me the closest restaurants,” or a niche market application may be “get me the session times for movies at the closest cinema.” Tourism applications also feature here, as well as transport applications (e.g., find me the closest train station and the timetables for the next few trains).

- **Accident and emergency services:** These provide information about people during emergencies, such as “report my vehicle’s location to the police” or “patient Smith needs immediate medical assistance at his home.”
- **Supply chain management and tracking:** These are typically internal applications, such as for goods tracking or delivery truck tracking and navigation (e.g., direct drivers to the most optimal route). In terms of public applications, one could classify location-sensitive billing/micropayment applications in this category (e.g., purchasing train tickets based on entry and exit stations).

LBS applications can also combine some of the above features. For example, navigation applications to the nearest train station may also provide information about approaching trains.

2.2.3 Technical capabilities

LBSs can also be differentiated based on technical features as follows:¹

- **Network- versus device-centric:** Prior to GPS-enabled phones, mobile network providers could deliver LBSs to users by triangulating their position based on the signals sent to mobile base stations. This “network-centric” focus was obviously dependent on mobile network providers and how they exposed location information to LBS developers.

New devices are now capable of determining their location via GPS and digital compasses. This “device-centric” situation reduces the dependency on mobile network providers and hence has allowed more freedom in LBS application development, resulting in the explosion of applications available to date.

Assisted GPS, where GPS is combined with information from mobile networks, is also common. This reduces initial synchronisation time. However the location information is still available and controlled by the mobile device rather than the mobile network provider.

- **Reactive versus proactive:** Reactive LBSs are those that require a request from the user, whereas a proactive LBS sends pertinent information to users when they reach certain locations. Most LBSs are reactive, such as the common “find me the closest restaurant”-style applications. With Research In Motion and Apple supporting push notifications to their devices, it is possible to develop new LBSs that proactively send information to users without the LBS application running.
- **Location versus route/trace information:** LBSs commonly store or process a user’s current or previously recorded location, which represents a single data point. Now systems are starting to record more than a single data point. Route or location trace information, along with speed and direction, are being stored [Brilingaite, Jensen and Zokaite, 2004]. This extra information allows systems to “predict” the future location of users to support proactive LBS, as well as provide historical information that may be useful to the service provider such as for traffic modelling [Estrin, 2009].
- **Single target versus multi-target:** Single target LBSs represent “single user request” applications such as “find my nearest train station.” Multi-target applications are those that

¹ The following list is based on a similar list presented in [Kupper, Treu and Linnhoff-Popien, 2006].

involve multiple users such as “find my nearest friends”-type applications. This is in line with moves towards more collaborative computing as part of Web 2.0 and social networking. Another example of multi-target LBS is “geocasting,” where location-based notifications are sent to users within an area and not just to a single user [Kim, et al. Efficient Geocast, 2008].

- **Central versus distributed knowledge:** Many current LBSs are based on client-server architecture, where location and map information is stored in a somewhat central location. One possible step is to move towards distributed knowledge sharing. A simple example may be location-based searches that combine results for “nearby restaurants for dinner,” “hotels for accommodation” and “rental car providers for next-day travelling.” By using Bluetooth for peer-to-peer computing, mobile devices themselves may store and share location-based information.
- **Indoor versus outdoor:** Different types of location collection technologies have different accuracies, and some do not work indoors (e.g., standard GPS, which needs a view of satellites). For LBSs that require low precision indoors, it is possible to use other technologies such as mobile network or Wi-Fi triangulation. For higher precision, custom-made devices may be required and standard location collection APIs may not suffice (e.g., Bluetooth or Wi-Fi access points may be added with custom logic to determine a user’s location based on the access point they are connected to or perhaps even based on distance calculations determined from packet round trip times).

2.3 CHARACTERISTICS OF NEXT-GENERATION LOCATION-BASED SERVICES

Now that classifications and characteristics of different LBSs have been discussed, the “next-generation LBS” in the context of this report can be modelled. Its functional characteristics should embrace the following:

- **Being proactive**, where relevant information is *pushed out* to users based on their location: Notifications may be sent to individual users or geocasted to a group of users within an area.

Pushing information out to users has many benefits. Firstly, by having data pre-loaded when the user views an application, it can be made to appear more responsive. Secondly, it reduces the need to regularly poll for information, which reduces power consumption of portable devices. Thirdly, by reducing the polling required, it can reduce the load on the server component. Finally, it allows users to be notified when the LBS application is not running.

- Capable of **storing and using location trace information**: This represents historical information about users, and could be used to predict their future location and intent. The predictions may allow LBSs to deliver more targeted information to users.

Also, LBS service providers can plot location trace information from many customers to gather data about common routes and requests in near real-time. This can be used to improve service models, thereby allowing LBS service providers to fine tune services. For example, if many public transport commuters are travelling on the same route during a special event, more buses could be allocated on-the-fly to service needs.

- Being multi-target, with an emphasis on **collaboration between users**: Next-generation LBS applications could incorporate communications between multiple users, such as the ability to find nearby users and to geocast messages to many users at once. Also, localised information entered by users can be easily accessed by others, which could improve decision-making.

Collaboration could also be supported through peer-to-peer networking when users are close to each other. This would allow data to be shared easily between users who are working together.

- **Making use of sensors** available in smartphones such as cameras and accelerometers, to provide extra information that can be fed into a cloud of location-tagged information: This is important for collaborative LBSs as it minimises manual data entry, which can make the application more appealing. Some examples include:
 - Using the camera on smartphones to upload geo-tagged images. In the near future, it may also be possible to add face detection functionality, which could trigger a function to add friend's names to tags in addition to standard text tag entry.
 - Through a combination of accelerometers and location collection, it is possible to determine velocity and route information based on movements of the device.
 - As a more sophisticated example, KDDI Corporation recently released a mobile phone in Japan that was capable of detecting specific types of actions that were performed by users purely based on the vibration of the device [Fitzpatrick, 2010]. It is capable of differentiating between walking, climbing stairs and even emptying a waste bin. This is done by recording accelerometer movements, and matching them with known patterns for actions. The matching is done via a sophisticated algorithm run on a server. Apart from potential privacy issues, this demonstrates the possibilities of recording valuable information using built-in sensors in mobile phones.

3 LBS COMPONENTS AND TECHNOLOGIES

This chapter presents a discussion of components in the next-generation LBS architecture, including descriptions of recent technology advances in academia and industry. This chapter also examines the technology behind LBS components, including (1) how **location information** can be obtained, (2) relevant **middleware standards**, (3) **technologies for storing location data**, and (4) how **smartphone technology can be applied to next-generation LBS**. These are important considerations when designing an LBS application.

LBSs contain a number of components including maps and Geographic Information System (GIS) information, location collection services, and LBS application-specific subcomponents. The architecture of an LBS can be generalised as shown in Figure 3-1. The description of each component is also tabulated in Table 3-1.

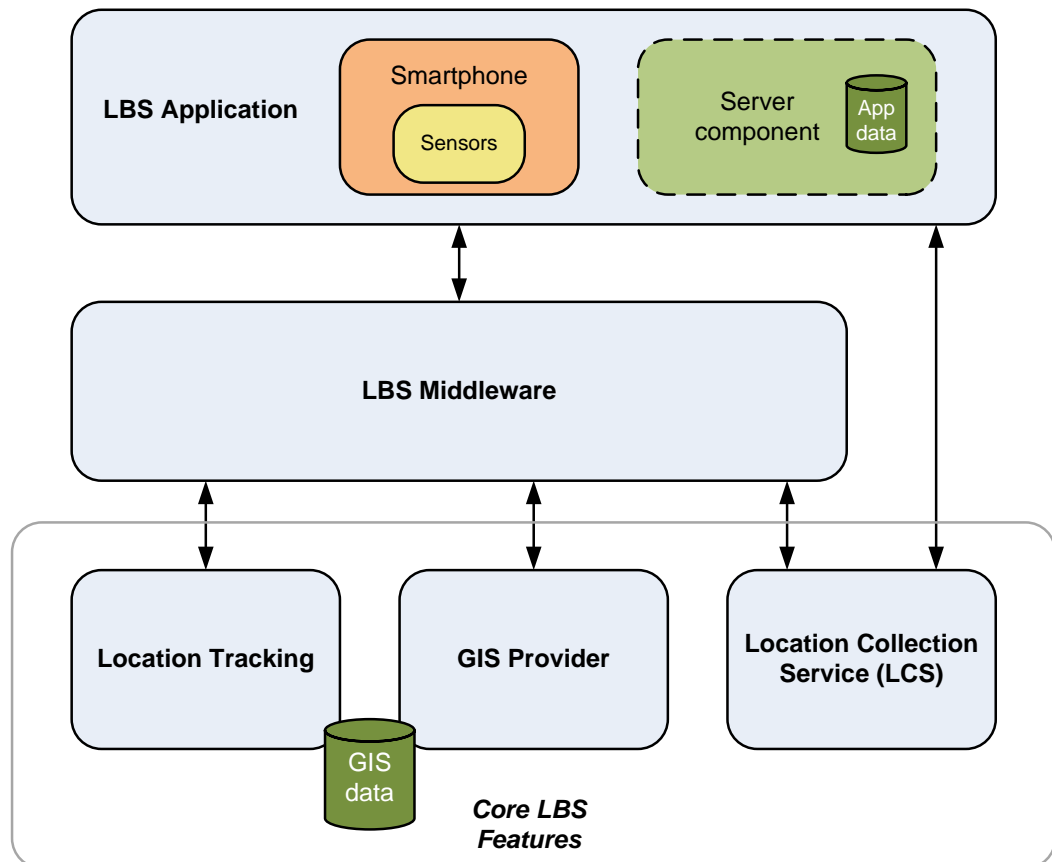


Figure 3-1 Diagram showing components of an LBS

Component	Description
LBS Application	This represents a specific application such as a “find my friends” application. This consists of a smartphone component, which has a number of sensors, and potentially a server component that includes application-specific data (such as location-tagged information). ² A brief comparison of currently available smartphones, with a focus on potential for next-generation LBS applications, is presented in Section 3.4.
LBS Middleware	This wraps access to Core LBS Features (Location Tracking, GIS Provider and Location Collection Services) to provide a consistent interface to LBS applications. The OpenLS specification represents one standard for LBS middleware and is described in Section 3.2.
Location Tracking	This component stores the location trace of individual users. This represents a fundamental component in next-generation LBS as it contains the data that allows a user’s route to be determined and potentially predicted. In particular, this component would typically support the following functionality: <ul style="list-style-type: none"> • Keep records on user’s current and past locations. • Notify other components when a specific user has moved, or when they move in or out of an area. This supports location-based notifications being sent to users. • Determine which users are within a defined location. This supports geocasting features. • Queries of location trace to generate user movement models.
GIS Provider	This component provides geospatial functionality for many LBSs including map information, map visualisation and directory services. Google Maps with its API can be considered a GIS provider. Other examples include deCarta ³ (commercial) and OpenRouteService.org ⁴ (open source).
Location Collection Service (LCS)	This component performs location collection to get a latitude and longitude for a specific user. Depending on the technology, this component may be accessed via the LBS Middleware (e.g., mobile network triangulation via a service provider) or directly (e.g., via GPS receiver in the smartphone). A brief introduction to the various technologies for the Location Collection Service is described in Section 3.1.

Table 3-1 Description of LBS application components

² In a distributed LBS, there may not be a server component; instead, the application-specific data may reside in a distributed peer-to-peer network.

³ See <http://www.decarta.com>

⁴ See <http://www.openrouteservice.org>

3.1 LOCATION COLLECTION TECHNOLOGIES

Global Positioning System (GPS) typically comes to mind when location collection technologies are discussed. GPS and A-GPS have a clear advantage in terms of accuracy. However, there are several other options available with different accuracies, such as mobile network-based and Wi-Fi-based technologies. Briefly, GPS-based technologies provide highest accuracy at the expense of power consumption. Mobile network-based technologies provide lower accuracy, but since they do not rely on an extra receiver, they use less power. Wi-Fi based technologies sit in the middle of the continuum but rely on Wi-Fi access points to be available nearby. Not all applications require high accuracy; hence, one can save on power consumption and extend a smartphone's battery life by using a lower accuracy technology.

The tables below present characteristics of some location collection technologies grouped into (1) **GPS**, (2) **mobile network** and (3) **Wi-Fi-based solutions**.⁵

⁵ The tables were compiled from various sources including [Ahn, et al., 2008] and [Xin, 2009].

3.1.1 GPS-based solutions

	GPS	Assisted GPS
Description	The device's position is triangulated based on signals from at least four GPS satellites based on the known position of the satellites, the time that messages from the satellites were sent and the time that they were received.	This is an enhanced form of GPS commonly used on smartphones, in which an "assistance" server on the mobile network provides information such as accurate GPS satellite orbit information, accurate timestamps or possibly snapshots of GPS signals. This can allow GPS accuracy with initial location information within seconds, thereby making it practical for use in LBSs.
Accuracy	5-10 m	5-10 m
Pros	Highly accurate. No dependency on a mobile network provider.	Highly accurate, and allows GPS to be used in more areas, such as in densely populated areas where clear GPS signals may not be obtainable. Fast location collection.
Cons	<p>Relatively high power requirement, as a GPS receiver needs to operate.</p> <p>It can only be used outdoors where clear satellite signals can be obtained.⁶</p> <p>Depending on the device, it may take a long time (~30 seconds) to lock onto satellite signals.</p>	There is a dependency on the mobile network provider; it can only operate where mobile network reception is available.

Table 3-2 Summary of GPS-based location collection technologies

⁶ Specialised devices can make use of technologies such as differential GPS (D-GPS), but this requires D-GPS transmitters to be fitted in buildings and devices to have D-GPS receivers.

3.1.2 Mobile network-based solutions

	Cell of Origin	Time of Arrival	Angle of Arrival	Enhanced Observed Time Difference
Description	Location is approximated based on the cell that is servicing a user. The first location tracking systems for Enhanced 911 used this type of technology, but it has since been superseded by technologies that are more accurate.	Distance to base stations is calculated based on the signal propagation time between the mobile device and the station. Absolute timestamps (TOA) or measured differences between packets (TDOA) may be used. Calculations are performed by the mobile network systems rather than the device.	The angle of the signals received from a mobile device at a number of base stations is recorded and used to determine the device's location.	This is similar to Time of Arrival (TOA) except the calculations are performed on the device.
Accuracy	Depends on cell size. It could be on the order of 150m in urban areas, and up to 1km in rural areas.	50-150 m	50-150 m	50-150 m
Pros	No additional functionality required on mobile devices, as it makes use of existing mobile network infrastructure.	Greater accuracy than Cell of Origin. No need for a GPS receiver; standard mobile device hardware is sufficient.	Greater accuracy than Cell of Origin. No need for a GPS receiver; standard mobile device hardware is sufficient.	Greater accuracy than Cell of Origin. Less dependency on mobile network provider.
Cons	Poor accuracy.	Requires mobile network provider action to perform calculation, which may lead to privacy concerns.	Requires mobile network provider action to perform calculation, which may lead to privacy concerns. Multipath can cause issues.	Requires more sophisticated mobile devices to perform calculations.

Table 3-3 Summary of mobile network-based location collection technologies

3.1.3 Wi-Fi-based solution

Wi-Fi Positioning System	
Description	The identities and relative signal strengths (which correspond roughly to distance) to public Wi-Fi access points are recorded by the device, thereby allowing triangulation with respect to these access points. Based on a database of known access points and their physical location, an approximate location for the device can be calculated. This system was initially created by Skyhook. ⁷
Accuracy	10-20 m
Pros	Fast and relatively accurate location collection compared to mobile network techniques. Lower power requirements compared to GPS due to speed and no need for GPS receiver. Allows devices such as laptops to use location collection and hence interact with some LBS applications (such as those equipped with HTML 5 Geolocation ⁸).
Cons	Relies on access to Wi-Fi access points, which may not be available in certain locations.

Table 3-4 Summary of Wi-Fi-based location collection technologies

3.2 LBS MIDDLEWARE STANDARDS

As with all technologies, LBS is now moving towards common standards, allowing application developers to focus on their applications rather than core LBS components. The Open Geospatial Consortium (OGC) is one major group that is developing standards in the LBS space. It has a number of notable members such as ESRI, Oracle, Google, IBM and various US and European agencies. Three important standards (*OpenLS*, *GML* and *KML*) are described in the following subsections.

3.2.1 OpenLS

The Open Location Services (OpenLS) standard proposes an overall system architecture for LBS and interface specifications for various components.⁹ In particular, the system architecture separates (1) *location collection services* (i.e., systems that can find locations of specific mobile devices), (2) *LBS application providers* and (3) a “GeoMobility” server.

⁷ See <http://www.skyhookwireless.com>

⁸ See <http://dev.w3.org/geo/api/spec-source.html>

⁹ See <http://www.opengeospatial.org/standards/ols>

The GeoMobility server represents a Web Service-enabled middleware piece that handles common LBS functionality. It provides six services listed below as defined by OpenLS. The first five are considered core functionality:

1. **Directory service:** This represents “yellow pages”-style functionality and supports request such as “locate bank with name X” or “find me the closest bank.”
2. **Gateway service:** This supports a standard interface for LBS applications to determine the location of the device that it is running on. This is particularly useful where location collection involves mobile network providers.
3. **Location Utility service:** This supports conversion of coordinates into addresses and vice-versa (i.e., geocoding and reverse geocoding).
4. **Presentation service:** This supports map visualisation functionality, such as displaying maps and providing pan/zoom capabilities.
5. **Routing service:** This supports functionality to find routes between an origin and a destination, such as for turn-by-turn navigation systems. There are proposals to enhance this service so that routes can be correctly determined in near real-time when certain paths or areas need to be blocked out (e.g., due to temporary road works or traffic accidents preventing use of specific roads).
6. **Tracking service:** This provides functionality to record location trace of users, and allows other LBS components to be notified when a user moves. This service represents major functionality that would be useful for next-generation LBSs (and hence is represented by the “Location Tracking” component as shown in Figure 3-1).

So far, there are a few solutions that implement the OpenLS specification. For example, the Oracle Spatial extensions expose an OpenLS interface that can be used by applications.¹⁰ There is also an open source site (www.openrouteservice.org) that provides Google Map-like functionality while implementing OpenLS. Interestingly, while Google Maps does provide most of the above services, it does not outwardly expose an OpenLS interface. There are also commercial “pre-packaged LBS” solutions that implement the OpenLS core services such as from deCarta. At the time of writing, there were no publically advertised implementations of the OpenLS Tracking service.

3.2.2 XML (GML, KML) standards for location data

Two other important standards from OGC for transferring location data are as follows:

1. Geography Markup Language (GML):¹¹ This is an XML-based language for representing various geography data such as points of interest. It is used extensively in the above-mentioned OpenLS standard and in general for transferring geography data. A compressed form more suitable for lower-power devices, such as mobile devices, is presented in [Zhang and Xu, 2009].

¹⁰ See <http://www.oracle.com/technology/products/spatial/index.html>

¹¹ See <http://www.opengeospatial.org/standards/gml>

2. Keyhole Markup Language (KML):¹² This complements GML by providing information about annotations and markings on maps (visualisation). It was initially created by Google and is used in the Google Maps API. Google is now part of the OGC and has proposed KML as a standard.

The above standards may be useful in integrating with GIS systems, and as starting points for data models of LBS applications where detailed geography data is required to be stored or transferred.

3.3 TECHNOLOGIES FOR STORING LOCATION INFORMATION

Although much of the location information in an LBS is stored within the LBS core components, it is sometimes necessary to associate application-specific data with locations, such as geo-tagged photos and videos, and also query data based on location.

In the past, location information has been stored as individual latitude and longitude columns within database tables [Rubin, 2006]. User-defined functions and stored procedures can be written to perform necessary geo-spatial calculations such as determining distances and intersection of regions. The major issue with this type of implementation is that storage and indexing of data does not take into account the nature of spatial data.

Standard relational database indexes (B-tree and derivatives) store data “next to each other” based on greater than or less than functions. This works well with linear data such as numbers; however, coordinate data can be considered as two- or multi-dimensional, as it represents points on a map that can be divided into regions that contain points. Data points can then be grouped based on region. This concept is the basis behind R-trees [Wikipedia R-tree, 2010], which are implemented by commercial databases that provide spatial extensions such as Oracle and Postgres.¹³ Researchers have proposed even more sophisticated structures such as ZB-tree and D-tree, which provide even better performance for location-based queries [[Myllymaki and Kaufman, 2003] and [Vijayalakshmi and Kannan, 2008]].

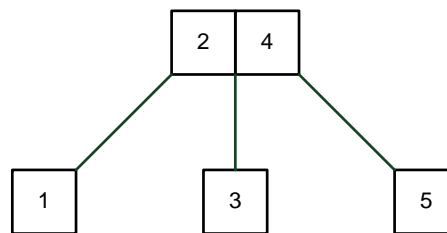


Figure 3-2 Diagram demonstrating a simple B-tree. Values are stored in nodes (boxes) and are ordered based on linear greater than/less than relationships.

¹² See <http://www.opengeospatial.org/standards/kml>

¹³ See <http://www.postgresql.org> and <http://postgis.refrations.net/>

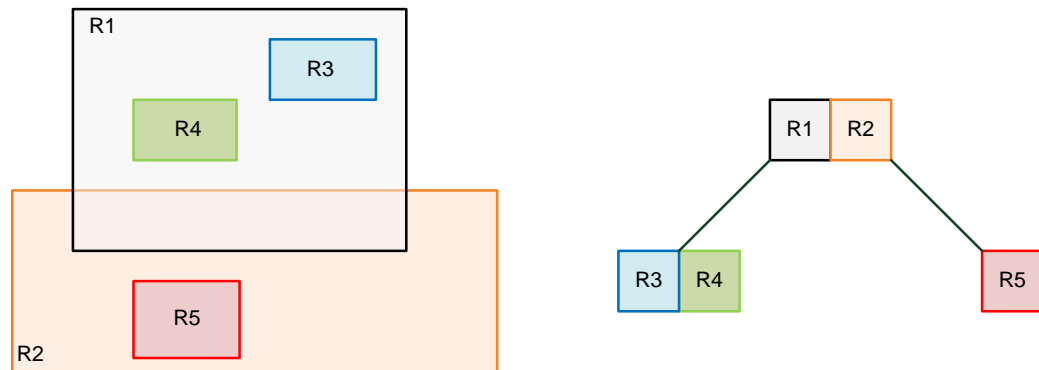


Figure 3-3 Diagrams demonstrating an R-tree. The left diagram shows regions where points may exist. The right diagram shows the equivalent R-tree. Sub-regions are considered sub-branches under parent regions (e.g., R3 and R4 are contained within R1, so are sub-branches under R1).

In order to ensure uniformity of interfaces provided by relational databases, the OGC has released a standard for extending SQL to support spatial data.¹⁴ This has been implemented by some common databases such as Oracle, Postgres and, to a lesser extent at the time of writing, MySQL.¹⁵ The standard defines a set of common functions to be provided, and support for specific spatial data types such as points, polylines and polygons. This means application developers no longer need to develop basic location querying functionality.

In terms of performance, benchmarks on low-cost hardware using currently available spatial indexing techniques have shown that even in the worst case, movement tracking for over 100,000 objects can be sustained [Mylymaki and Kaufman, 2003].

Moving beyond relational databases, many researchers have proposed storing location information using Semantic Web technologies such as Resource Description Framework (RDF).¹⁶ One of the major benefits of using Semantic Web technologies is that “merging” syntactically different data sources would be relatively easy – an effect of storing metadata about data structures, which allows necessary transformations to occur. As an example, an LBS that stores the current latitude and longitude and address of a user could integrate with other LBSs that take inputs as either latitude and longitude, or as an address. With Semantic Web, this kind of integration can occur with little effort from the developer; no custom coding to feed specific information data types to specific Web services is required. In addition to improved integration, Semantic Web also promises to allow simple rules to be developed that can infer new knowledge about users. These rules, for example, could define what and when tailored advertisements should be sent to the user based on his or her current location and past purchase trends.

¹⁴ See <http://www.opengeospatial.org/standards/sfs>

¹⁵ See <http://www.mysql.com>

¹⁶ There are a number of academic papers that present demonstrated LBS prototypes using Semantic Web technologies. Interested readers are referred to [Becker and Bizer, 2008], [Fang et al., 2007], [Liapis, Vassilaras and Yovanof, 2008] and [Weibenberg, Voisard and Gartmann, 2004].

In terms of practical implementation, geometry data type definitions for RDF are available, such as RDFMap and RDFGeom,¹⁷ which allows location information to be treated like any other data.

Other researchers have also suggested that coordinates can be translated into “locations” [Hoareau and Satoh 2009]. For example, one may define Office A has having a region with specified coordinates, so all coordinates within that region can be classified as being in Office A. This translation allows users to be “linked” with the matching location, which may allow rules to be defined more easily than when considering coordinate data alone. In addition, aggregation queries and rules can also be developed easily if locations are structured in hierarchical fashion. For example, a query to “show me all users in Office A” can aggregate query results from “show me all users on Level 1 in Office A,” “show me all users on Level 2 in Office A,” etc.

While relational databases are suitable for data models with a fixed schema (i.e., one knows what tables are required and what columns are required in each table), they are less suited to store more dynamic data or where flexibility in structure is required [Leavitt, 2010]. For example, where information stored as a user’s profile can vary greatly, non-relational databases may have an advantage.

Non-relational databases have been gaining momentum in recent times; Google and Amazon now provide public access to their BigTable and SimpleDB systems respectively, and Facebook has open-sourced the Cassandra database¹⁸ that underpins its application, with other social networking or Web 2.0 sites such as Twitter and Digg looking to move from relational databases to Cassandra. Non-relational databases may not be considered enterprise-ready now, but there is much research and development occurring on them.

Considering that next-generation LBSs will be looking to store vast amounts of information from advanced sensors and smartphones, and potentially integrate more with user profiles, a move towards these non-relational databases may be on the cards in the near future.

As can be seen from the above discussion, there are numerous options for storing location information suitable for today’s applications and tomorrow’s.

3.4 MOBILE DEVICE TECHNOLOGY FOR NEXT-GENERATION LBS

In Section 2.3, a list of desirable functionalities for the next-generation LBS was presented. To support these features, smartphones will need to provide the following:

1. Easy access to location information.
2. Ability to support push notifications.
3. Multiple sensors that can provide inputs into applications, such as still/video cameras, accelerometers and magnetometers.
4. Potential support for enterprise features, or other technologies such as Bluetooth peer-to-peer communications that may be of use to specific applications.

¹⁷ See <http://www.mapbureau.com/gml/>

¹⁸ See <http://incubator.apache.org/cassandra/>

Taking a snapshot of smartphones from the three major platforms currently available (Research In Motion BlackBerry, Apple iPhone, and Google Android), a brief review is done with respect to the above features. It is interesting to see how far these smartphone platforms are heading towards meeting the functionalities of next-generation LBS modeled in this report.

3.4.1 Easy access to location information

Although this may vary between models and make, there are certain series of smartphones from each platform that support A-GPS, mobile network-based and Wi-Fi-based triangulation.

In terms of developer's access to this location information, all three platforms provide a framework so that developers need not be concerned with the underlying technology. Developers only need to specify the accuracy required, and the framework automatically identifies the appropriate technology. The iPhone has the CoreLocation framework and Android has its own Location classes, while BlackBerry uses the Java standard Location API (which requires a Java Micro Edition application to be developed). iPhone and Android also support HTML 5 Geolocation through the browser.

The upcoming HTML 5 looks set to include a specification for "geolocation."¹⁹ This would allow Web applications running in a browser to access the user's current location, previously the domain of native applications only. Depending on the underlying implementation, it can tie into any available location collection technology. For example, on the iPhone, Web applications can access the same granularity of location information as is available to native applications.

One major benefit of HTML 5 Geolocation is that cross-platform LBS applications can now be written as Web applications. Obviously, other features such as push notifications would not be available.

At the time of writing, HTML 5 Geolocation is supported by Mozilla Firefox, Google Chrome and Apple Safari on the iPhone.

3.4.2 Push notifications

Research In Motion pioneered push technology with the BlackBerry,²⁰ and Apple now provides this service for the iPhone.²¹ These services are accessible via published APIs. Android does not support push notifications; however, since it supports multi-tasking, push notifications could be simulated with a polling background task.

The prototype application presented in Chapter 5.1 demonstrates how push notifications can be used for LBSs on the iPhone to support collaboration between employees (design details are presented in Appendix A).

¹⁹ See <http://dev.w3.org/geo/api/spec-source.html>

²⁰ See <http://na.blackberry.com/eng/developers/javaappdev/pushapi.jsp>

²¹ See <http://support.apple.com/kb/HT3576>

3.4.3 Availability of sensors

Devices for all three platforms now include still and video cameras, while iPhone and some Android devices include digital compasses and accelerometers. The prototype iPhone application presented in Chapter 5.1 and Appendix A demonstrates how information can be captured from these sensors to support data entry for an LBS.

3.4.4 Enterprise integration and other features

The original target market for the BlackBerry was enterprise users, and hence there is strong enterprise integration such as “push email.” Apple has also been developing the iPhone’s enterprise capabilities, with support for Microsoft Exchange integration, push email and native VPN.²² As such, these devices would be suitable for enterprise LBSs.

The Apple iPhone and Google Android platforms also now include support for Bluetooth-based peer-to-peer communications.²³ This may be useful for developing niche peer-to-peer LBSs, such as for location-based games or meeting collaboration tools.

Although this research focuses on smartphones, ultra lightweight devices with integrated GPS and 3G mobile network access are starting to appear. At the time of writing, some examples include Asus Eee T101, Viliv S5 or S10, Nokia Booklet 3G and Apple iPad.

Ultra lightweight devices have the benefit of larger touch screens, and may have keyboards to support more convenient text entry compared to smartphones. They can also make use of existing desktop applications, so that one LBS Web application with HTML 5 Geolocation could support multiple devices.

²² See <http://www.apple.com/support/iphone/enterprise>

²³ See http://developer.apple.com/iphone/library/documentation/NetworkingInternet/Conceptual/GameKit_Guide/Introduction/Introduction.html and <http://developer.android.com/guide/topics/wireless/bluetooth.html>

4 PRIVACY AND LBS

One major concern that arises when discussing LBS is how to maintain the privacy of users;²⁴ hence, finding privacy solutions has been an active area of research. Privacy is always a concern whenever users give out information about themselves. In this chapter, specific privacy concerns related to LBS, possible privacy solutions, and future trends in privacy research are discussed.

4.1 WHY IS PRIVACY A CONCERN WITH LBS?

The fundamental principle behind LBS is that users send requests to other parties containing their location information. This location information can be used to profile the person in terms of his or her movement, and can even be used to predict future movements and intents.²⁵

By combining a person's location with other aspects of his or her profile using information from different systems, such as recent purchases at a department store, much knowledge can be deduced about the user. For example, if it is noted that a person sends a number of LBS requests near a hospital over a long period of time, it could be inferred that this person may have a chronic illness, which may lead to some form of social discrimination such as in getting bank loans or purchasing health insurance. By knowing the purchase of a holiday package as well as noting that the purchaser's current location is not his or her home city, it can be deduced that the person is currently out of town. Planned thefts could take place.

On the other hand, deduced information about a person could be put to positive use. For example, a list of suggested items for purchase could be sent to users when they approach a department store. Also, deduced information could assist in finding missing persons. Anonymous information could also be used for traffic modeling. To allow these services, an "opt-in" legal policy could be enforced to ensure a person is aware of how his or her information is being used [van Loenen and Zevenbergen, 2007], at least by legally bound organisations. For example, iPhone users must acknowledge an application's activation of the device's location collection service.

However, there is still concern about unscrupulous use of a person's location information. Some possible situations where this may happen are:

- LBS providers could unscrupulously use the information without or beyond permission given by users.
- Other users of LBS providers could make use of the information to suit their own needs. For example, in "friend tracking" applications, the alleged friends of a person could make use of the person's location to carry out unwelcome or undesirable activities. While the LBS provider is acting legally, the final recipient of location information may not be.

²⁴ Cell phone triangulation that was mandated by the US government as part of the Enhanced 911 policy raised privacy concerns in 1998, long before the advent of LBS as we know it today. Users do not have a choice to opt into such a service, and hence there are concerns as to how their location information could be used.

²⁵ [Song et al., 2010] reported that even with anonymised location data it could be possible to predict user's locations up to 93% of the time.

- Attackers may use legal loopholes to obtain information from multiple systems, including an LBS provider, and combine the data to form a profile of users – not to mention that crackers can intercept data sent to LBS providers.

Much of the research in this area has been trying to address these issues by concealing exact location information from even LBS providers, so as to minimise the opportunity for misuse.

4.2 IDENTIFICATION REQUIREMENTS OF LBS

The severity of privacy breaches is highly dependent on the specific LBS application, and different severities would require different privacy solutions. To aid in identifying suitable solutions for different LBSs, Liu [2009] tagged LBSs in terms of the level of identification required:

1. **Anonymous LBSs:** These do not require any user identification in the request. For example, a “find me the closest restaurants” LBS request does not need to contain specific information about the user; only a request ID is needed. However, it may be possible to tie a request to a specific phone, and hence the user, by combining records from the LBS and phone companies – so privacy still is a concern.
2. **Identity-driven LBS:** These LBSs require user identity information to operate. For example, the application might notify friends when the person leaves the office in the evening. Also, internal enterprise LBSs are likely to fall into this category.
3. **Pseudonym-driven LBS:** This is in between the above types, and represents the case where a user needs to be identified but no personal user information is actually required. One example is an application that responds to a request to “turn on my Wi-Fi connectivity when I get to a McDonalds or Starbucks.” A particular device needs to be identified but details about the user are irrelevant.

4.3 PRIVACY SOLUTIONS

A number of technical solutions have been proposed to protect the privacy of LBS users. The general principles of some major solutions are described in the following subsections.

4.3.1 Anonymisation

The aim of anonymisation is to ensure that the LBS will not be able to link requests to specific users. The underlying principle is that a number of requests from multiple users are grouped together to be processed by an LBS provider at once. An intermediate trusted system called an “anonymiser” performs this aggregation and returns specific responses to users. An anonymiser provides *k-anonymity* to users by grouping together *k* users’ requests. The value of *k* can be configured to ensure sufficient anonymity [Liu, 2009]. Dummy requests could also be created to simulate additional users [Cho et al., 2009].

The above description is for a simple anonymisation. Extensions have been made to address any possible flaws, such as:

- With a simple anonymisation, by taking into account more context information around users’ requests, it may be possible to associate requests to users. In particular, requests may include “static” properties about users that may be known by LBS providers. For

example, although k users may request the location of a bank, far fewer may have accounts with that bank. This significantly reduces the set of users that would likely send such a request, thereby increasing the likelihood of linking a request to a specific user [Hasan, Ahamed and Tanviruzzaman, 2009].

- People typically follow roads or well-defined paths when travelling. Therefore, even with a level of uncertainty in the accuracy of a location (e.g., due to GPS accuracy), the possible locations where a user can send requests may be quite limited, thereby increasing the possibility of identifying users' exact locations [Liu, 2009].

The main downside to anonymisers is that they rely on the anonymiser being a trusted system, and the anonymiser can become a single point of failure, which is a common issue in standard client-server architectures.

Anonymisation is primarily used for ensuring privacy in anonymous LBS applications.

4.3.2 Cryptographic techniques

Cryptographic techniques such as encryption and secure hashes are commonly applied to conceal information. Researchers have applied some of these techniques to hide user's identities. For example, rather than passing a user Id with each request, a secure hash could be transferred instead. This allows parties to identify requests without seeing the user Id itself.

Protocols based on computational Private Information Retrieval (PIR) are presently popular in academic circles. These essentially prevent LBS providers from knowing specifically what data is requested from users by applying mathematical functions to the request parameters. Interested readers are referred to papers such as [Ghinita et al., 2008] and [Vishwanathan and Huang, 2009] for more details.

These techniques may require more computational power and sophisticated application code than other methods, and so may not yet be suitable for smartphone applications. However, protocols such as those based on PIR do not rely on centralised infrastructure as anonymisers do.

Cryptographic techniques would be useful for both identity-driven and pseudonym-driven LBS applications.

4.3.3 Accuracy filtering

Location information has a defined accuracy depending on the technology used. For example, GPS may be able to resolve location down to 5-10 metres, whereas triangulation may be on the order of 50 metres. While a high-accuracy technology may be used, applications may not need such high resolution, so a client application can reduce the accuracy of its location when sending requests to LBS providers, thereby reducing the possibility that the LBS provider can identify the user's exact location.

Accuracy filtering can be done by selecting the appropriate location collection technology, such as via iPhone SDK configuration parameters for accuracy, or by post-filtering of the location information. One example of post-filtering was a technique described in [Patrikakis, Voulodimos and Giannoulis, 2009], where the map of the world is divided into tiles, with an identifier based on its location in a Cartesian coordinate system. Each tile is then further divided into four subtiles, with their identifiers being the parent tile's identifier with suffixes representing more detail. This process

can be repeated until the required level of accuracy is achieved (e.g., around 20-23 times). A user's location is mapped to a tile and hence can be referenced by the tile's identifier. By reducing the length of the tile identifier that is submitted to an LBS provider, the accuracy of the location is reduced.

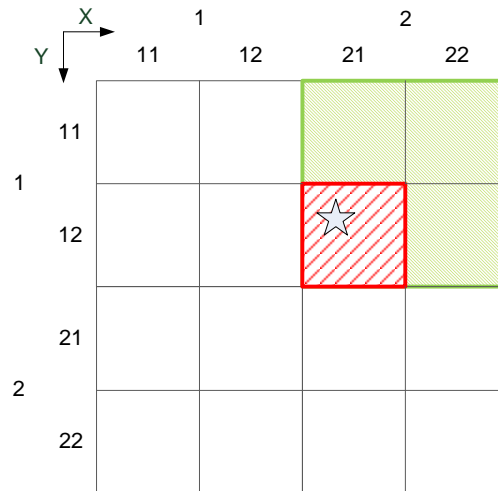


Figure 4-1 An example of post-filtering as suggested by [Patrikakis, Voulodimos and Giannoulis, 2009]. The star represents the user's location, which may be reported as the red square (21, 12) when higher accuracy is required, or the green square (2, 1) if only low accuracy is needed.

By choosing the appropriate location collection technology, application code can be simplified. The benefit of post-filtering is that different levels of filtering can be easily applied to different requests such as to different LBS providers.

Accuracy filtering techniques could be applied to any type of LBS. Apart from addressing privacy issues, selecting the most appropriate location collection technology is also considered a best practice for LBSs given the potential to reduce power drain on the battery.

4.3.4 Transformation of location request data

The preceding solutions assume that the user is communicating with a trusted service – either an anonymiser or LBS provider. A “client only” solution that has been proposed is for the client application to transform the location information in the request, and then inversely translate the response that comes back. For example, a user may send a request for the distance to a nearby landmark. The user's current location may be translated by a random amount, and hence the distance would need to be mathematically transformed to account for the translation. Using this technique, the LBS provider would not receive the exact location of the user, and subsequent requests cannot be used to track a user due to the random offset.

This technique would only be useful for certain types of LBS applications, typically for anonymous LBS, where absolute location is not required.

4.3.5 Privacy policies

Privacy policies should contain a collection of several privacy solutions in a configurable way. For example, accuracy filtering may be used for some applications, and cryptographic techniques may be used where privacy is more important. Ribeiro and Zorzo [2009] suggest that a multi-level privacy policy is required, where higher levels add privacy solutions on top of the lower levels to achieve an overall higher level of protection.

Privacy policies would need to be flexible and configurable based on (1) individual LBS provider, (2) a user's profile, and (3) the ultimate end users of the location information.

4.4 ADDRESSING PRIVACY CONCERNS FOR ENTERPRISE LBS USERS

Opt-in policies can be easily applied to publically accessible LBSs. However, when enterprise LBSs are incorporated into business processes, there is no opt-in policy that users can agree to. Therefore, without acceptance from enterprise LBS users, employers may be seen as forcing employees to use an LBS. Since the employing organisation has access to all LBS information, and enterprise LBSs are typically identity-driven, it would be relatively easy for the organisation to associate location information with employees. These privacy concerns cannot be solved simply by the techniques described previously.

Van Loenen and Zevenbergen [2007] have shown that while privacy is a concern to users, they may be willing to give out their location information in exchange for benefits – e.g., a form of payment. DiMicco et al. [2008] also suggested that privacy (in particular for social networking applications) was not a large concern in enterprises compared with public systems. If the benefits of the LBS can be presented and demonstrated clearly to employees, buy-in and acceptance can be achieved.

4.4.1 Encouraging uptake of LBS via incentive schemes

In addition to gaining acceptance from employees, there may also be a need to encourage the uptake of LBS technology. Some approaches from deploying social networking tools in the enterprise could be used. For example, Farzan et al. [2008] presented the use of a points system, where users are awarded points for correctly using the tool, with rewards given out based on point count. For enterprise LBSs, points could be awarded for frequency of use or location updates.

As an extension to the points system, Garyfalos and Almeroth [2008] suggest a scheme based on coupons and mutual benefit. Consider an example of a discount offer. A coupon may be passed around to users' devices as they walk past a store. If they are not interested in the discount immediately, users can then pass around the coupon to other users that they come near (e.g., via peer-to-peer connectivity) and add their user Id to the coupon. This process continues until a user takes the coupon, at which point all users who have added their Id to the coupon get some sort of benefit (small payment, etc.). This small reward encourages intermediate users to pass around the coupon. The authors also proposed making a pyramid style scheme where the payment to intermediate users increases as more users pass around the coupon. Their scheme was designed to encourage passing of messages for an LBS that operates through peer-to-peer communications between users (a next-generation LBS concept), and could be applicable to an LBS that encourages collaboration between users.

4.5 FUTURE TRENDS IN PRIVACY RESEARCH

Although there has been much research in this area, there are still a large number of academic papers coming out on this topic relevant to next-generation LBS. Kulik [2009] identified a few areas for research including:

- **Privacy in distributed or decentralised LBSs:** Most work has been done in the area of centralised LBSs, where location and user data are stored in fewer places; it is easier to control access and simpler to enforce.
- **Protecting privacy in “continuous” queries to an LBS:** Continuous queries refer to regular requests to provide real-time updates of LBS information. Attackers may be able to intercept multiple queries from a user, which can be “glued” together to either form location trace information or to extract more accurate information about a user based on patterns of requests.
- **Configurable privacy policies:** There is a need to create simple interfaces to allow users to configure privacy policies themselves. For example, standard default “profiles” of privacy settings may be specified to assure that certain basic privacy levels are maintained. These policies could be aligned with features exposed in LBS standards such as the OpenLS specifications.

5 LBS CASE STUDIES

This chapter discusses some possible next-generation LBS applications including potential technical challenges and considerations, and suggestions on architectural and design decisions. These possible applications are for:

1. **Mobile insurance brokers.** This represents a typical enterprise LBS or publically accessible (niche market) LBS.

A prototype of this application was developed, and the key learnings are presented here. Detailed design decisions and methodology are documented in Appendix A.
2. Providing real-time updates for **public transport connection times.**
3. **Building induction management.**
4. **Building evacuation management.**

5.1 CASE STUDY 1 – MOBILE INSURANCE BROKER APPLICATION (MOBROKER)

This particular case study is targeted at mobile insurance brokers who operate in the field to assess risks as part of insurance applications and/or claims assessments. Their business process is typically paper-oriented; forms are filled out on site and then entered into a computer later in the office. Also, brokers may not have easy access to relevant electronic information while on site, such as past claims, statistics on weather and fire patterns and risk factors relating to crime rates in neighbourhoods.

The proposed application was designed to allow remote data entry directly into electronic forms that are synchronised with backend systems. Data entry would include simple text; photos tagged with location, direction and timestamp; and rudimentary maps using inertial navigation that does not rely on GPS. The form interface was also designed to facilitate collaboration by presenting past answers to form questions in a forum-style format.

The application can also access Web sites of related information, pre-populating URL fields with current location information where possible.

Finally, brokers may also find nearby “agents” and can request assistance as required by sending “push notifications” to other users.

A prototype of this application, called “MoBroker,” was developed to run on an Apple iPhone 3GS, with the server deployable to a cloud service such as Amazon EC2.²⁶ The following subsections describe the application in detail including use case scenarios, application architecture, major system components and potential beyond the prototype.

²⁶ See <http://aws.amazon.com/ec2>

5.1.1 Use case scenarios

Five main scenarios are described in the subsections below:

1. Viewing all premises to be visited at the start of the day
2. Performing risk assessment by answering form questions
3. Adding a photo as an answer to a form question
4. Creating a small map of an area as an answer to a form question
5. Requesting assistance from nearby agents or colleagues

5.1.1.1 Viewing all premises to be visited at the start of the day

A typical scenario of how MoBroker could be used at the start of the day is as follows:

1. A broker starts the MoBroker iPhone application and logs in.
2. The application presents a map of premises to be visited (with red markers) and a marker of where the broker is now (green marker). By tapping on a marker, information such as the address and customer name of a premise is displayed in a callout. This is shown in Figure 5-1.
3. The broker can also view the premises in list form, which is in order of approximate appointment time, by tapping on the list button on the top navigation bar.



Figure 5-1 Screenshot of the overall map view of premises to visit

4. The broker can drill down into the details of a customer's premises by clicking on the button in the callout on the map view. This brings up the screen shown in Figure 5-2. This screen enables the broker to:
 - a. View information about the appointment or customer's premises by clicking the "information" button.
 - b. View relevant information on Web sites for this particular premises.
 - c. Call the customer via the "phone" button on the toolbar (first button), such as to confirm the appointment time.
 - d. View a route to the premise via the Maps application (second button on toolbar). The broker may then follow the route and drive to the location.

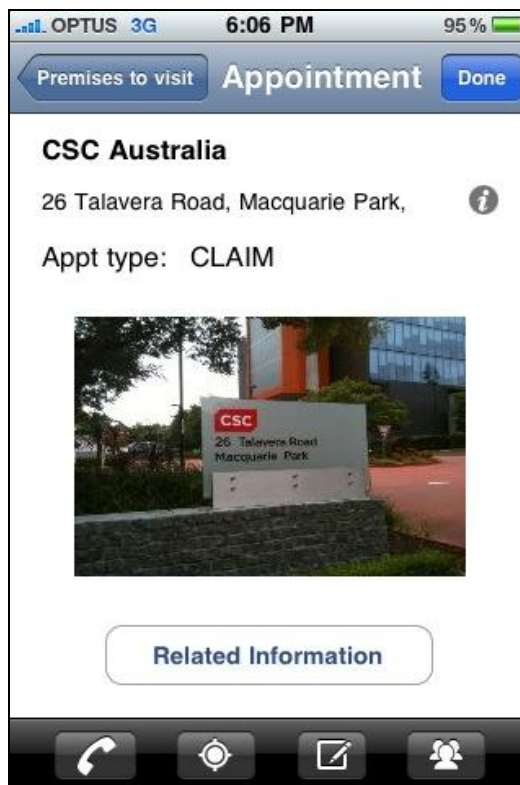


Figure 5-2 Screenshot of customer's premises details

5.1.1.2 Performing risk assessment by answering form questions

A typical scenario of how the prototype could be used to review and answer form questions once arriving at a customer's premises is as follows:

1. After arriving on site, the broker would log into the application.
2. The application would detect that the broker has arrived at the customer's premises and bring up the details immediately without having to navigate through the map view.
3. The broker would drill down into the questions to answer about the customer's premises by pressing the third button on the toolbar in Figure 5-2.
4. The list of questions would be displayed as shown in Figure 5-3. It is based on a forum-style interface and shows who provided the latest answer to a question and when.
5. When the broker taps a question, the details on the question can be seen, including a summary of past answers; these could be from brokers in the office, or by other brokers who may have previously visited the site (see Figure 5-4). The broker can scroll between questions using swipe left or right gestures.

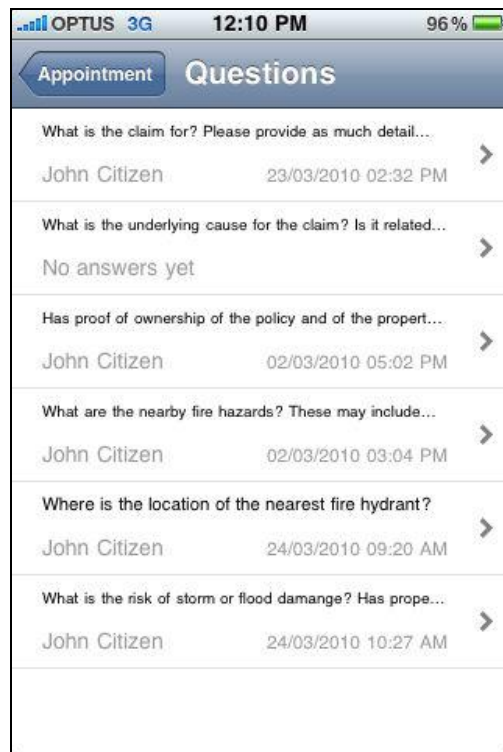


Figure 5-3 Screenshot showing the question list for the customer's premises with information about the most recent answers

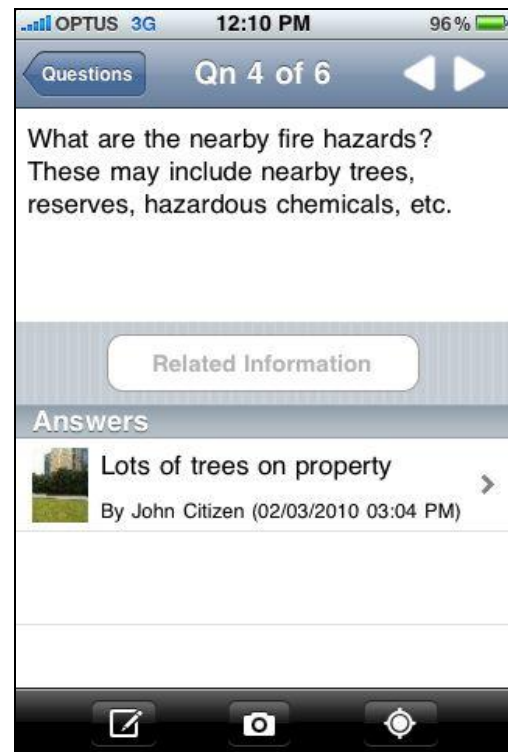


Figure 5-4 Screenshot showing a view of the question and summary of answers

6. If the broker wishes to view Web sites related to a particular question, such as crime statistics when answering a question on crime risk, he can tap the “Related Information” button, which brings up a list of relevant Web sites. The client application pre-populates the URL with the location information of the customer’s premises.
7. To view details of a previous answer, the broker can tap the answer to bring up a screen as shown in Figure 5-5.

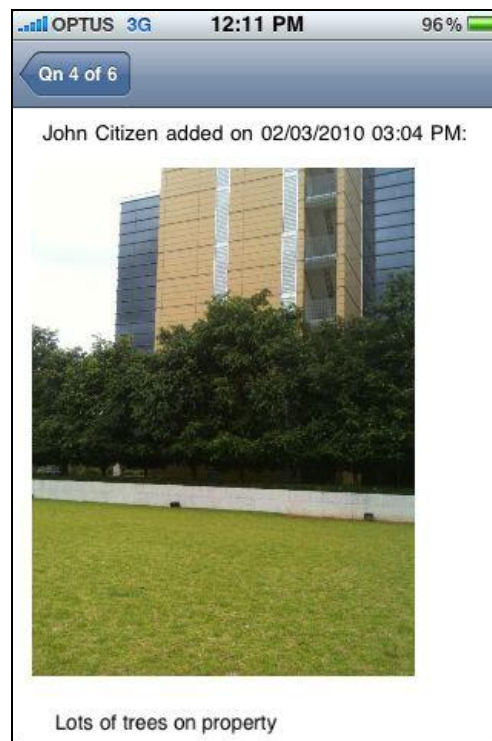


Figure 5-5 Screenshot showing details of a previous answer. This particular answer included a picture and a caption.

8. To add an answer to the question, there are three options corresponding to the three buttons on the bottom toolbar as shown in Figure 5-4:
 - a. Add a text answer (first button).
 - b. Take a photo with location, direction, timestamp and a caption (second button). This is described further in Section 5.1.1.3.
 - c. Create a small map of an area using a tool based on inertial navigation (third button). This is described further in Section 5.1.1.4.
9. When the broker has completed all the questions, he can confirm completion of the form on the Premises Details screen (Figure 5-2) by pressing the “Done” button.

This could trigger a backend process to submit the answers to various insurance agencies or other systems for processing.
10. Upon completion, the broker continues on to the next scheduled property.

5.1.1.3 Adding a photo as an answer to a form question

Taking a photo uses the inbuilt camera functionality on the smartphone, along with location collection and digital compass. This is triggered by pressing the camera button as shown in Figure 5-4.

When taking a photo, the broker has an option to use an existing photo as a template (see Figure 5-6). For example, the broker may be reviewing a claim for a broken window, so he may choose to overlay a photo of the window from the original insurance application form to allow a before-and-after comparison to be made. The application would search for photos that were taken near the broker's current location and present a list as shown in Figure 5-7.

When taking a picture with a template photo, the template is shown as a faded image overlaid on top of the actual camera preview. The application, using the inbuilt digital compass, directs the user to turn left or right as required to align with the template photo's direction as shown in Figure 5-8. This rudimentary "augmented reality" feature assists the user in lining up the photo. If no template is selected, no overlay is shown.

After taking the photo, a text entry screen is presented to allow the user to enter a caption. After entering the caption, the photo and caption are submitted to the server.



Figure 5-6 Screenshot showing option to use a template photo when taking a picture

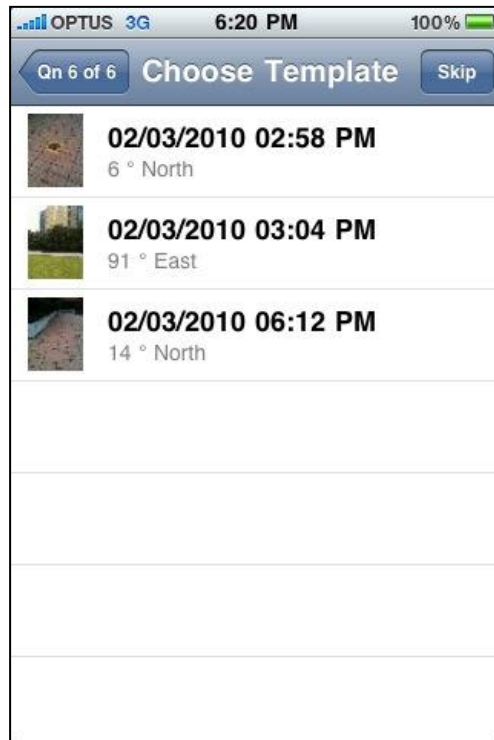


Figure 5-7 Screenshot showing list for selecting a template photo to use. Photos are filtered to be close to the current location and are sorted by date.

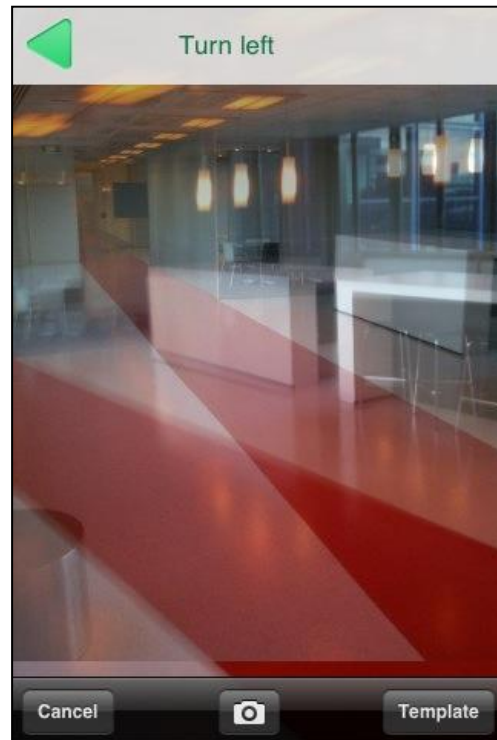


Figure 5-8 Screenshot showing the camera view with a template image overlaid. The overlay also indicates what direction the broker needs to turn to align with the direction of the template photo.

5.1.1.4 Creating a small map of an area as an answer to a form question

MoBroker can draw a simple map of an area by recording the number of steps taken in a direction as recorded from the digital compass (inertial navigation). It is a simple demonstration of navigation techniques that could be used where GPS or mobile network triangulation are not sufficiently accurate or available.

This functionality may be used to perform the following:

- Assist with estimating fire risk by measuring the distance and path from a building to the nearest fire hydrant.
- Record the approximate size of a room or building to confirm accuracy of insurance application forms and premium calculations.

The steps to using this functionality are as follows:

1. The broker would select to create a map. This brings up the map entry screen similar to Figure 5-9.
2. The broker would press the "Start Mapping" button.

3. The broker takes steps in the desired direction. To change direction, the broker would stop, rotate and wait for the compass readings shown on the screen to stabilise before walking in the new direction.
4. When mapping is complete, the broker presses the “Stop Mapping” button. The broker may also choose to delete map segments that were created in error.
5. To view the map, the broker presses the “Done” button to bring up the map display as shown in Figure 5-10. The map display draws the recorded map segments to scale, and orientates the diagram to match the actual direction as recorded by the compass.
6. The broker can confirm to save the map by pressing the “Use” button.



Figure 5-9 Screenshot of the map entry screen while in mapping mode

Figure 5-10 Screenshot of the map display screen. The recorded map segments are drawn to scale on the screen, and rotate in accordance with the orientation of the phone.

5.1.1.5 Requesting assistance from nearby agents

MoBroker also supports a use case where one broker can request assistance from another nearby broker or agent. For example, a trainee broker may need help from a senior broker. This is accessed from the Premise Details screen (Figure 5-2) by pressing the last button on the toolbar.

The screen would then display a list of nearby brokers or agents within 10 km of the current location. The user has the option to view a broker's contact details (integrated with the Address Book on the phone), or can request assistance at the current site as shown in Figure 5-11. This triggers a notification to be sent to the target broker as shown in Figure 5-12.



Figure 5-11 Screenshot showing the process of requesting assistance from another broker. The user taps the target broker's name and confirms the request.



Figure 5-12 Screenshot showing a push notification being received by the target broker

When the target broker arrives at the requested premises and logs in, the system detects the target broker's proximity to the premises and sends a notification back to the requesting broker to indicate that assistance has arrived. This demonstrates simple collaboration and location-based push notification.

5.1.2 Application architecture

This LBS is suited to a client-server architecture. The application architecture implemented in the prototype is presented in Figure 5-13, with the descriptions of the modules in Table 5-1 that follows.

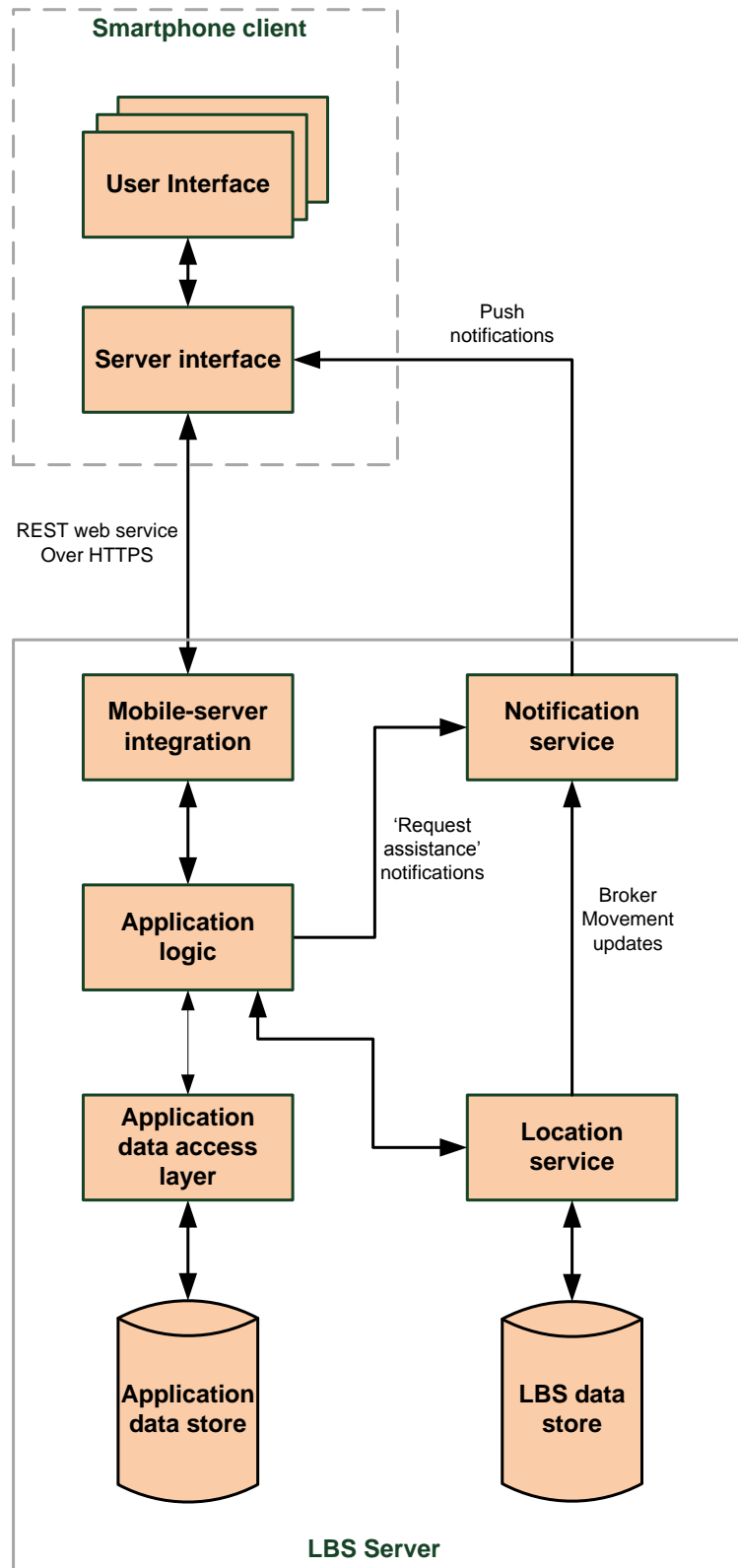


Figure 5-13 Application architecture for the Mobile Insurance Broker LBS

Module	Description
User Interface	This represents the various screens in the prototype application. This module also typically integrates with the location collection capabilities on the smartphone.
Server interface	This module handles communications with the server component of the LBS.
Mobile-server integration	This module handles client-server communications between the Mobile Application and the Application and LBS data stores. It exposes functions provided by the Application logic module in a secure manner to the mobile application.
Application logic	This module contains the business logic for the user interface of the application such as for retrieving premises to visit, retrieving questions, saving answers and sending requests for assistance.
Notification service	This module is responsible for sending notifications to brokers. The notification for requesting assistance is triggered via user action through the Application logic module. A location update of the target broker when he arrives on site triggers a notification to be sent to the requesting broker.
Application data access layer / Application data store	These modules are responsible for accessing and storing application-specific information such as question and answer data. The Application data access layer provides a database-independent wrapper around the data store per standard enterprise application architecture.
Location service / LBS data store	These modules are responsible for accessing and storing location trace information about premises and brokers. This is separate from the Application data store in order to allow the Location service to be reused between multiple LBSs. In the prototype, these modules were custom developed. It is expected, however, that a commercial product could be used in the future. Such a product would implement the OpenLS Tracking specification.

Table 5-1 Description of the modules in the application architecture for the Mobile Insurance Broker LBS example

5.1.3 Challenges and possible solutions

There were many challenges that were encountered while developing the prototype LBS. Some of these are listed below. Detailed descriptions of technical challenges and implemented solutions are presented in Appendix A.

5.1.3.1 Selection of a suitable smartphone platform

The iPhone was selected as the target platform for the following reasons:

1. The device included the latest sensors and features available on mobile phones at the time of writing, including digital compass, accelerometers, GPS and camera with access to an overlay view for augmented-reality-type features.

2. The large screen was suitable for features such as map and photo displays.
3. Ruggedised cases can be easily purchased, such as from Otterbox.²⁷
4. The device had broad availability including Australia, where development was done.

Android would also be a valid platform, and the support for background tasks would be ideal for supporting regular location updates to the server. However, at the time of writing, Android devices with comparable features to the iPhone were not readily available in Australia.

In order to support both platforms simultaneously, a tool such as Appcelerator Titanium²⁸ could have been used. However, some features, such as camera overlay views to support augmented reality features, were not inbuilt in the tool at the time of writing, and hence a native application was developed. This does mean that the client part of the system would need to be ported to different platforms if so desired.

One possible concern with using a smartphone for this application is the relatively small screen and keyboard size in comparison to an ultra-lightweight netbook, and hence a netbook may be easier to use. There are two responses to this. Firstly, a smartphone is sufficiently small to be clipped onto a belt and hence taken on site, where manoeuvring through tight pathways would be easier. Secondly, the mobile application should be designed such that only a minimal amount of work needs to be done on site with the smartphone; it complements a desktop-based application or intelligent server-based logic that can be used to pre-populate form answers.

5.1.3.2 Dealing with mobile network outages

At certain premises, mobile network coverage may not be available, or the network speed may be slow (e.g., GPRS versus 3G). This could affect all smartphone-based remote applications. A possible remedy is to design the mobile application such that necessary data is stored on the device itself through an on-board SQLite database or similar database. The application also needs to re-synchronise with the server component whenever sufficiently fast network access is available again; platforms such as iPhone and Android provide capabilities to detect this.

Obviously, data stored on the mobile device must be secured. Most platforms prevent cross application access of on-board databases, and access to an application should be secured at least via authentication. It is suggested that security software development kits be specially developed for smartphone platforms.²⁹

5.1.3.3 Providing secure access to backend data

The Mobile Insurance Broker application requires access to backend data containing potentially sensitive information such as insurance policy premiums, application forms and customers' personal details. This data needs to be accessed over the Internet to be available on smartphones. While this is not a new problem to many organisations, it is an important issue that must be considered when designing integration with backend systems.

²⁷ See <http://www.otterbox.com/iphone-3g-3gs-cases/>

²⁸ See <http://www.appcelerator.com/>. The CSC iPhone application is written using this tool.

²⁹ [Collins, 2010] discuss the view of Kaspersky on this issue.

5.1.4 Possible extensions to the prototype

The prototype has demonstrated the implementation of some next-generation LBS features such as collaboration and location-based push to enterprise applications. A more elaborate system could include features of the prototype and other additions such as:

1. Reports on users' movements such as number of sites visited, distances travelled, or a visualisation of premises covered over a financial quarter. These reports would allow an organisation to objectively measure user performance or possibly streamline costs (such as carbon emissions from car travel).
2. Additional user interfaces to handle the data, especially a browser-based interface that would be more ideally suited to a laptop or PC for the less mobile users. HTML 5 Geolocation may be an option to provide basic location information.
3. True integration with backend systems, such as through ACORD³⁰ interface standards or implementation of a suitable ACORD data model. Since the server component could be deployed to a publically accessible cloud service, there are security considerations related to communication between public and private systems.
4. Integration with scheduling and route optimisation algorithms to minimise broker's travel costs in terms of distance and time.

The concepts behind the prototype can also be applied to many other industries and business processes that follow similar patterns. The "Equipment inspection log" and "Doctor's patient list" LBS examples described in Section 1.2 could make use of most of the functionality in the prototype. Two more examples are as follows:

- A real-estate agent would typically perform property inspections as part of a rental service, or in preparation for property sales. A system similar to this prototype could be used to allow notes and photos to be captured and immediately used by staff at the office, and to check backgrounds of potential tenants. The template photo facility could be used for rental inspections to determine if faults were pre-existing before a renter moved in.
- The prototype could also be used for a school excursion system at a national park. Points of interest could replace premises, where students may need to answer questions when they arrive at each site. For further reading, they could access related information on Web pages that are pre-defined by teachers. In addition, students could collaborate to share answers or thoughts at each site and could request assistance from nearby teachers.

5.2 CASE STUDY 2 – PUBLIC TRANSPORT CONNECTION TIMES APPLICATION

A daily trip for many people may involve several means of public transport, requiring coordination of arrival and departure times at interchanges. Transport planners are available to develop plans for commuters based on schedules. However, these do not take into account real-time running; trains, buses and ferries may run early or late or be cancelled. It would be a great comfort to commuters to

³⁰ See <http://www.acord.org/>

know whether they would make their connecting service or would need to arrange for other alternatives given delays.

A possible solution would be to provide commuters with an LBS that tracks the person's movements along his or her trip and, based on actual transport running times, estimates whether the person will make connecting services. This "prediction" can be *pushed out* to commuters as they are approaching the interchange point. A possible scenario for this application would be:

1. Users plan a trip using the LBS application on their smartphone, possibly using existing trip planners provided by transport authorities.
2. When a user boards a vehicle (e.g., train or bus), the smartphone application could detect the user's location and velocity. With a series of samples, it could be possible to record a route travelled by the user and match this with a tracked vehicle. For example, if the user was within the vicinity of a physical train at a point in time, and is moving in the same direction as that train, then the user is likely to be a passenger on that train. Alternatively, users may manually enter information about the vehicle they are on.
3. As vehicles move, the LBS would receive real-time updates on vehicle locations, estimated arrival and departure times. Since users can be associated with the vehicle they are on, the user's estimated arrival time (ETA) at interchange points can also be inferred. Based on this, the system could estimate whether the user will make his or her connecting service at the interchange point (i.e., whether the user's ETA is before the connecting service's estimated departure time).
4. As users are approaching their interchange point, LBS could push a notification out to users informing them of whether they will make their connecting service.
5. Users may also manually query the LBS via their smartphone to find out if they will make their connecting service, and to obtain possible alternatives.

This kind of application could also provide the transport authorities with useful planning information beyond a single mode of transport; it can capture common trips for commuters and overall quality of service to commuters (i.e., on-time running across the entire trip, not just a single leg on a bus or train).

5.2.1 Possible application architecture

This public transport connection time LBS application is suited to a client-server architecture. A possible application architecture is presented in the Figure 5-14, with the descriptions of the modules in Table 5-2 that follows.

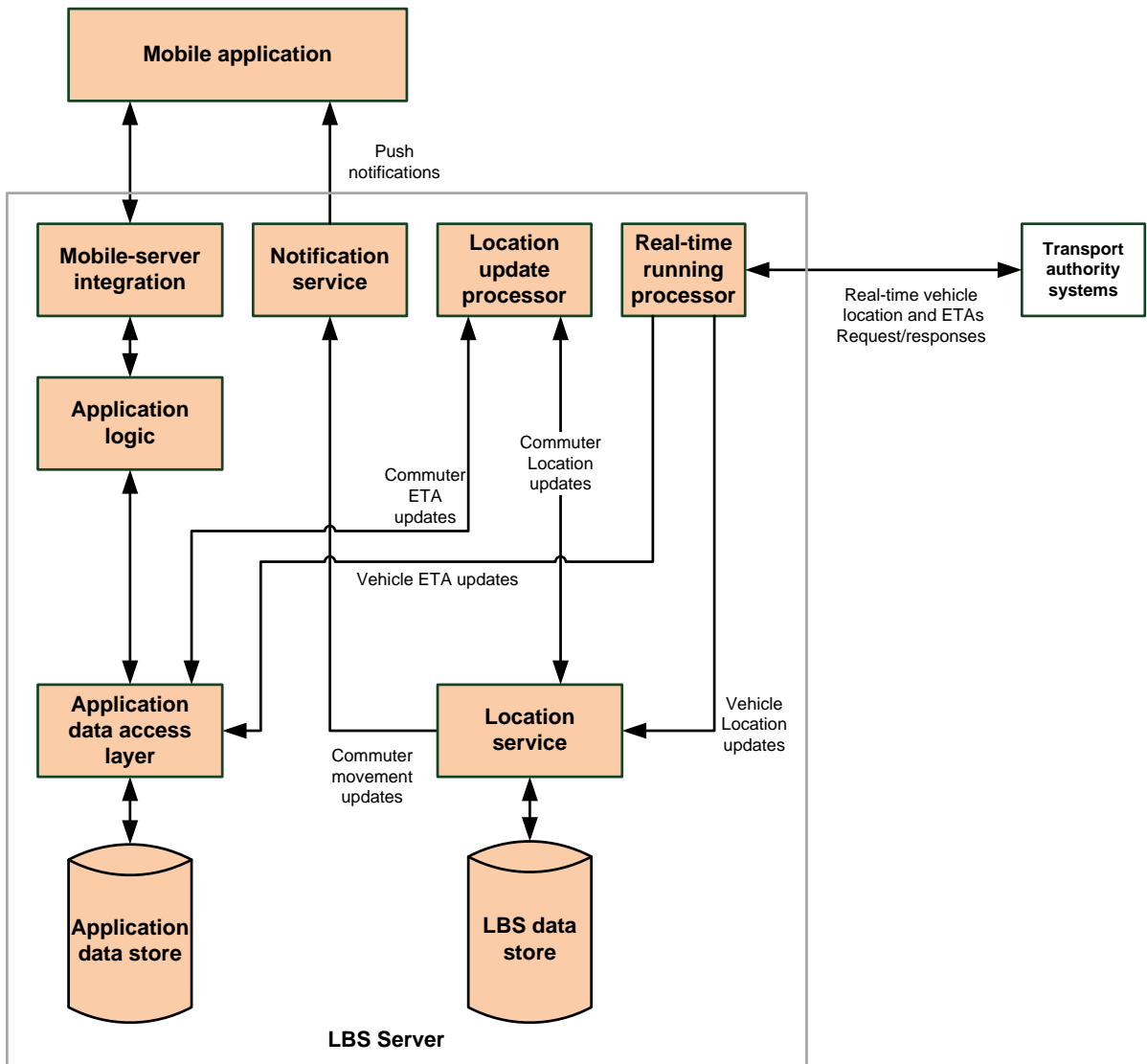


Figure 5-14 A possible application architecture for public transport connection times LBS. The tan blocks represent modules that are part of the LBS.

Module	Description
Mobile application	This is the application that runs on the smartphone. This consists of a user interface for entering trip information, a module for managing location and movement information, a client-server communications module, and a push notification handler.
Mobile-server integration	This module handles client-server communications between the Mobile Application and the Application and LBS data stores. It exposes functions provided by the Application logic module in a secure manner to the mobile application.

Application logic	This module contains the business logic for the user interface of the application such as for creating a trip, viewing real-time running information on vehicles, and updating a user's location when boarding a vehicle.
Notification service	This module is responsible for sending notifications to commuters about connecting services when they are approaching an interchange point. This could be triggered on a regular basis (e.g., once a minute), or via an update notification on tracked commuters from the Location service.
Location update processor	This module updates the location of commuters based on associated vehicles, calculates estimated time of arrival/departure at interchange points, and determines whether commuters will make connecting services. It can also clean up specific user information, such as mobile phone identifiers, once a user has completed a trip. Since this is a data-intensive process, a production rules engine such as JBoss Drools or IBM iLog could be considered for the implementation. This could be triggered on a regular basis (e.g., once a minute) or via an update notification on vehicle movements from the Location service.
Real-time running processor	This module requests real-time updates on vehicle locations and estimated arrival/departure times if available, and updates relevant data points in the data stores. This could be triggered to run on a regular basis (e.g., once a minute). Updates may also be pushed to it from the transport authority systems, but this may be considered an unnecessary system coupling.
Application data access layer / Application data store	These modules are responsible for accessing and storing application-specific information such as user trips and estimated arrival/departure times. The Application data access layer provides a database-independent wrapper around the data store as per standard enterprise application architecture.
Location service / LBS data store	These modules are responsible for accessing and storing location trace information about commuters and vehicles. Ideally, this would implement the OpenLS Tracking specification, as other modules in this system can make use of location update and notification features. It is expected that a commercial product could be used to provide the functionality of these modules. These modules could also be reusable between LBSs.

Table 5-2 Description of modules in the application architecture of the public transport connection times LBS

5.2.2 Challenges and possible solutions

Developing the proposed LBS presents many challenges as listed in Table 5-3. Alongside is a discussion of possible solutions.

Challenge	Possible solution and comments
<p>How to gather sufficiently accurate location information about commuters to associate their movements with physical vehicles?</p> <p>This could be a major issue where good mobile phone reception or GPS reception is not available, such as in underground train stations.</p>	<p>There are two sub-issues associated with this challenge. Firstly, access to accurate location collection technologies such as GPS may not be available. Secondly, there may be no access to the mobile network to allow real-time transmission of location information to the LBS provider.</p> <p>For both sub-issues, transport providers could deploy or make use of existing Wi-Fi access points. Commuters' smartphones can connect to these access points and make use of triangulation to accurately identify their location.</p> <p>For the second sub-issue, it may also be necessary to buffer location information on the smartphone until access to the mobile network is restored. This would need to be a consideration in the development of the smartphone application.</p> <p>As a worst case, manual entry of the vehicle that the commuter is on should be supported. Given the potential benefits of the LBS to commuters, some data entry may be acceptable.</p>
<p>How to handle privacy implications of storing a commuter's trip information?</p> <p>This information represents both planned and actual location trace.</p>	<p>This application could be considered as a pseudonym-driven LBS and hence actual user identification such as names need not be recorded. Therefore, the information in the LBS would need to be combined with other systems in order to associate trips with particular persons. Standard security practices to protect the information, along with enforced legal restrictions on LBS and transport providers' use of the information, would be a major part of protecting user's privacy.</p> <p>In order to deliver push notifications to users, an identifier for their device would need to be recorded. Access to this information must be tightly controlled, and once a trip is complete, the identifier should be destroyed; the trip information should be anonymised if it is to be used by authorities for future planning purposes.</p>

<p>How to handle scaling of the LBS?</p> <p>There may be millions of commuters using the system during peak times in large cities.</p>	<p>The centralised component of the LBS would ideally be deployed on a publically accessible cloud service that would allow dynamic scaling of the individual subcomponents as required. For example, more front-end servers could be deployed during peak times.</p> <p>In addition, the servers may implement intelligent caching of data for common trips. This would reduce repetitive processing when users request information manually.</p>
--	--

Table 5-3 Technical challenges and possible solutions for the public transport connection time LBS

5.3 CASE STUDY 3 – BUILDING INDUCTION MANAGER

In many organisations, employees, contractors and visitors entering a facility need to be inducted, during which time evacuation procedures, locations of amenities, and other site-specific knowledge are presented. A visitor's induction may be valid for a fixed period such as one year, after which another induction is required. Some possible areas of improvement with this process are:

1. A lot of information may be presented to a visitor during an induction, and it is likely that not all the information would be retained immediately.
2. During the life of an induction, irregular visitors may not recall all important aspects of the induction but would not be required to go through the induction process.
3. Running an induction process requires resources possibly in terms of personnel and/or login stations.

A possible solution would be to develop an application that can display relevant induction information for a particular facility that a person is visiting and record when inductions have taken place. The basic use case for such a solution would be:

1. When a visitor or new employee arrives at a facility, they would start the application. The application detects the current location and sends a sign-in request to a central LBS server. Since no induction has been recorded by the user at the facility, he or she would be required to review induction material on his or her smartphone and digitally sign off on completion of the material.
2. During the visit, the user can bring up the induction material and review as necessary. Floor plans to locate amenities or nearby printers may be useful.
3. Upon leaving the facility, visitors would sign off using the application. The sign-off could also happen after the visitor has left. For example, if the visitor needs to leave urgently to catch a flight, he or she can sign-off while in the taxi to the airport.

Some additional features could be:

1. The pedestrian navigation system described in the Mobile Insurance Broker case study could be applied to this application to give turn-by-turn navigation to the nearest kitchen or printer.
2. Visitors may be inducted for certain parts of a facility but not others. If the visitor's location can be tracked, notification messages could be sent to warn the visitor if he is moving near a restricted area.
3. The application could manage inductions for multiple facilities that may be used by an organisation.

5.3.1 Challenges and possible solutions

The application architecture for this example should be quite similar to the application architecture presented in Section 5.1.2, and so it is not discussed further here. However, there are some challenges that would need to be addressed:

Challenge	Possible Solutions and Discussion
How to ensure that the system is used for sign-in and sign-out of visitors, and is not skipped?	Many organisations already have policies to enforce induction, sign-in and sign-out of visitors. These policies would still need to be enforced.
How to handle different smartphones?	The application may be considered as an extension or shortcut to an existing induction process, rather than a replacement. Therefore, users without a supported smartphone would go through the standard induction process. Multiple versions of the application could be developed for different smartphones.
How to provide accurate indoor movement tracking and navigation for maps or notifications of restricted areas?	The pedestrian navigation system described in the first case study provides relative navigation from a starting point and requires care when changing direction, as the compass has a relatively slow response time. This system could be useful for simple turn-by-turn navigation but not for more sophisticated movement tracking. For absolute positioning and movement tracking, the navigation system may need to be supplemented with Wi-Fi access points that can be configured with GPS coordinates to allow accurate triangulation.

Table 5-4 Challenges and possible solutions for the building induction manager LBS

5.4 CASE STUDY 4 – BUILDING EVACUATION ASSISTANT

During evacuations, such as for fire alarms, accounting for all personnel at a facility is important. Many industrial sites have systems in place to generate a manifest upon evacuation based on using access passes to enter and exit the site. However, in an office block scenario, this may be more difficult to achieve given the large number of people and that there often is no policy to force individuals to “swipe in” and “swipe out,” especially when visitors are not given access passes.

A possible solution would be to add to the building induction management LBS described in the previous section. Upon entering and exiting a facility, all personnel could sign in and sign out, respectively, using their smartphone. When an evacuation occurs, a manifest can be generated by retrieving the list of signed-in personnel.

The real benefit of using a smartphone-based LBS would be during a roll-call process to determine who is still unaccounted for. All personnel can report their current location via the LBS application, and wardens can access a list of personnel who have not reported in, or have reported in from within the evacuated facility. Wardens could push notifications out to users who have not reported in, or call them directly.

While not all personnel may have access to a smartphone, this type of system could supplement existing setups.

5.4.1 Challenges and possible solutions

A possible challenge is dealing with a potentially large number of messages that would be communicated during an evacuation, especially of a large facility. Standard mechanisms of buffering and retry can be used.

It may also be possible to create a temporary peer-to-peer network between personnel's smartphones while they are in close proximity to each other at evacuation sites. This is being made possible by peer-to-peer networking APIs now added to smartphone SDKs such as iPhone, as well as Android's use of Bluetooth connectivity. The general principle would be the desire to send registration messages to other smartphones, and the requirement that the smartphones forward collected registration messages along with their own registration message. These messages would eventually arrive at a warden's smartphone and update the manifest accordingly. The benefit of such a setup is that there is little dependence on mobile network infrastructure.

5.5 SUMMARY

In this chapter, four examples of LBSs that implement some next-generation features and functionalities were presented. The targets for these LBSs are quite varied, and the potential challenges that each LBS faces are quite different. However, the underlying application architecture remains relatively similar. These case studies have demonstrated the utility of next-generation LBS beyond consumer applications into the enterprise arena.

6 FUTURE TRENDS BEYOND NEXT-GENERATION LBS

LBS is a rapidly moving field, with many players and huge market potential, especially in mobile advertising. Three possible future trends for LBS are presented below: a move towards (1) **LBS middleware service providers**, (2) **improvements in visualisation** in LBS, and (3) **context-aware computing**.

6.1 LBS MIDDLEWARE SERVICE PROVIDERS

The LBS architecture presented in Chapter 3 suggests a separation between the location store and the application data store. This allows the location store to be reused between LBS applications within an organisation, or replaced with another commercial implementation when available. The OpenLS standards (especially the Tracking service) provide a defined interface for the location store.

The current situation is that users would submit their location information to each application (or potentially each organisation) that he or she wishes to use. This means duplication of data between applications and/or between organisations. Each organisation would need to maintain infrastructure to store location information. In addition, the application of privacy policies to the location data would need to exist on the device. If multiple devices are used, such as a laptop while in the office but a mobile phone while travelling, then policies may need to be configured and implemented multiple times.

It could be much simpler and more effective to have “LBS middleware service providers” that provide a location store for application developers (Estrin [2009] describes a similar concept as a “personal data vault” that contains user profile and captured sensor information). End users would subscribe to these service providers and submit their location information to the providers. LBS applications would also need to register with the service provider in order to request or be notified of location information. The service provider would need to provide privacy policy configuration to the end user. This would be applied to the location information sent to the LBS application.

Google Latitude demonstrates a step in this direction by providing a platform for user location information for a number of Google applications such as Google Talk and Gmail Chat.

If a middleware service provider were to provide this type of service, a number of issues need to be resolved as follows:

1. Effective privacy policy management and enforcement needs to be developed. Not only is it necessary to create a simple-to-use interface for users to configure privacy policies, the processing of these policies must be efficient enough to handle large data volumes.
2. A business model for the service providers would need to be developed. This may be on a usage basis for LBS applications and/or end users (e.g., number of requests per month). A free market between multiple service providers, much like the financial sector, may also be appropriate.
3. Legal frameworks may need to be put in place to ensure the service provider does not violate the end user's privacy.

6.2 IMPROVEMENTS IN VISUALISATION

Visualisation of information presented by LBS is well suited to maps. However, there is a challenge of showing maps on small form-factor devices such as smartphones. Even with relatively large and high-resolution displays available on current mobile devices such as the iPhone or Google Nexus One, it is still quite difficult to read maps on hand-held devices compared to a laptop or desktop computer. Harun et al. [2009] suggest a few techniques that may help including:

1. Using views such as “fish eyes,” where important areas are in focus and surrounding areas (context) are out of focus.
2. Using “halos” around points of interest, instead of just simple pin annotations, to show when they are off-screen and hence require panning to view.
3. Generalising unimportant features of a map such as by not showing them. For example, a navigation application may show only roads and no building information.
4. Using schematic maps rather than accurate maps to show points of interest.

Another area that has received much attention of late is augmented reality,³¹ where real images are mixed or augmented with artificially created views. These extra views may provide additional information about a location (e.g., an archived photo to be overlaid with a current image, as per the Mobile Insurance Broker case study) or could display something completely artificial such as for an immersive game. There is no need for users to interpret data on a screen or map because it is presented alongside what they can physically see.

Currently, most augmented reality for LBS on smartphones has been implemented based on digital compass readings and current location (e.g., the Layar browser³²); there is no interpretation of objects as seen in the current view. For example, when a user is pointing at a building of interest, the application measures the direction but would not detect the physical presence of the building. With improvements in processing power, it can be expected that these LBSs will be integrated with real-time image processing to improve the granularity of context information.³³ One may see an evolution in how extra information is displayed so as not to overwhelm users.

³¹ See http://en.wikipedia.org/wiki/Augmented_reality

³² See <http://layar.com>

³³ We can see the trend towards integrating real-time image processing: Google Goggles (<http://www.google.com/mobile/goggles/>) and Microsoft Photosynth (http://www.ted.com/talks/blaise_aguera_y_arcas_demos_photosynth.html). The Android platform appears to provide developers with access to the camera preview images and hence may support real-time image processing; the iPhone platform currently does not. New phones such as from Qderopateo (<http://www.qderopateo.com>) are also being developed with a focus on augmented reality.

Another possible form of visualisation is video streaming controlled by a user's location.³⁴ This could be very useful for tourism or education applications. With improving network bandwidth, commercial LBSs may appear with this functionality in the near future.

6.3 CONTEXT-AWARE COMPUTING

Location trace represents one important piece of context about a user. Adding this to other context such as current date and time and a person's role in an organisation could provide valuable information that LBS providers can use to make decisions or inferences about a user. Much like how relevant information can be pushed to users based on their locations, these inferences can further filter information flow to the most pertinent data. For example, based on the user's current location, current time and calendar, a mobile digital dashboard application may bring up relevant graphs and statistics about a department in preparation for a meeting.

Much research around this "context-aware computing" is centred on Semantic Web technologies (i.e., storage of information using RDF and OWL ontologies, and reasoning through descriptive logic). Some researchers have suggested that location data can be stored much like other sensor data in RDF and fed into descriptive logic reasoners [Evans, 2007]. Others have also demonstrated combining spatial extensions in relational databases with RDF data stores in location-based Semantic Web applications [Becker and Bizer, 2008].

As Semantic Web technologies take hold, LBSs may be seen as a subset of these more advanced context-aware applications.

While a significant amount of contextual information is already available, more data can be captured in the future. This may be through new sensors on smartphones, through short-range communications between smartphones or with sensors in the user's surroundings. For example, LBS applications may extract air quality measurements from nearby environmental sensors and suggest to users to wear masks in high pollution areas.

³⁴ Jeong and Kim [2006] present a demonstration of an LBS that shows broadcasted video on mobile devices when users are in certain locations.

7 CONCLUSION

Location-based services on smartphones have had great success in the consumer market, providing useful functions such as finding nearby points of interest. Next-generation LBSs promise to deliver even more interactive services to users and create a huge knowledgebase of location-tagged information. The major technological drivers of this are push notifications; better mobile network access through 3G and Wi-Fi; integration of advanced sensors on smartphones into applications such as accelerometers, digital compasses and still/video cameras; and Web 2.0 collaboration. As a result, analysts have predicted massive growth in the LBS market over the next few years.

The LBS phenomenon need not be limited to consumer applications. The case studies and MoBroker prototype presented in Chapter 5 demonstrate that enterprises and industry can make use of next-generation LBS based on smartphones to streamline internal business processes and tap into a rich network of location-based knowledge that can be used for performance modelling.

Given the potential in the LBS market, it is no wonder that there has been much research into the area. In particular, much attention in academic research has been focused on ensuring the privacy of users. Many techniques, such as anonymisation, accuracy filtering and cryptographic algorithms, have been developed so that only minimal quality location data needs to be given to LBS providers. With a combination of legal enforcement such as to force opt-in for application usage, and improvements in related privacy technologies and policies, privacy issues can be addressed and hence should not hinder provision of location-based services.

In addition, standards such as OpenLS have evolved to allow interoperability between players and the provision of basic services such as map visualisation and directories. In the future, more standardisation and provision of services such as for user tracking will allow LBS developers to focus on delivering new functionality instead of re-developing basic LBS infrastructure.

Further advances in visualisation technologies are also expected, from basic improvements on how information is displayed on maps, to new augmented reality displays whereby location-based information is made visible alongside point of interests through a camera view.

The future of LBS in the both consumer and enterprise arenas promises to be very exciting; achieving the ultimate goal of true context-aware computing may not be far away.

BIBLIOGRAPHY

- Ahn, J., Heo, J., Lim, S., & Kim, W. (2008). A Study on the Application of Patient Location Data for Ubiquitous Healthcare System based on LBS. *10th International Conference on Advanced Communication Technology* (pp. 2140-2143). IEEE.
- Becker, C., & Bizer, C. (2008). DBpedia Mobile: A Location-Aware Semantic Web Client. *Proceedings of the Semantic Web Challenge at the 7th International Semantic Web Conference*. Karlsruhe.
- Brilingaite, A., Jensen, C. S., & Zokaite, N. (2004). Enabling Routes as Context in Mobile Services. *12th International Symposium of ACM GIS* (pp. 127-136). Washington DC: ACM.
- Cho, E.-A., Moon, C.-J., Im, H.-S., & Baik, D.-K. (2009). An Anonymous Communication Model for Privacy-Enhanced Location Based Service Using an Echo Agent. *3rd International Conference On Ubiquitous Information Management And Communication* (pp. 290-297). Suwon: ACM.
- Collins, B. (2010, March 11). *Kaspersky: Apple is blocking iPhone security software*. Retrieved March 12, 2010, from PC Pro: <http://www.pcpro.co.uk/news/security/356344/kaspersky-apple-is-blocking-iphone-security-software>
- DiMicco, J., Millen, D. R., Geyer, W., Dugan, C., Brownholtz, B., & Muller, M. (2008). Motivations for Social Networking at Work. *Computer Supported Cooperative Work* (pp. 711-720). San Diego: ACM.
- Estrin, D. (2009, October 16). *Participatory Sensing: from ecosystems to human systems*. Retrieved March 1, 2010, from http://senseable.mit.edu/engagingdata/presentations/ED_Plenary_Estrin.pdf
- Evans, C. (2007). Intelligent Retail Business: Location Based Services for Mobile Customers. *2nd International Conference on In Pervasive Computing and Applications* (pp. 354-359). IEEE.
- Fang, L., Xiaolei, L., & Bian, F. (2007). Autonomic LBS based on context: Preview. *International Conference on Wireless Communications, Networking and Mobile Computing* (pp. 3266-3269). Shanghai: IEEE.
- Farzan, R., DiMicco, J., Millen, D. R., Brownholtz, B., Geyer, W., & Dugan, C. (2008). Results from Deploying a Participation Incentive Mechanism within the Enterprise. *Conference on Human Factors in Computing Systems* (pp. 563-572). Florence, Italy: ACM.
- Fitzpatrick, M. (2010, March 10). *Mobile that allows bosses to snoop on staff developed*. Retrieved March 11, 2010, from BBC News: <http://news.bbc.co.uk/2/hi/technology/8559683.stm>
- Garyfalos, A., & Almeroth, K. C. (2008). Coupons: A Multilevel Incentive Scheme for Information Dissemination in Mobile Networks. *IEEE Transactions on Mobile Computing*, 7 (6), 792-804.
- Ghinita, G., Kalnis, P., Khoshgozaran, A., Shahabi, C., & Tan, K.-L. (2008). Private Queries in Location Based Services: Anonymizers are not Necessary. *SIGMOD* (pp. 121-132). Vancouver: ACM.
- Google. (2010). *Corporate Information - Google Milestones*. Retrieved March 5, 2010, from Google: <http://www.google.com/corporate/history.html>

- Harun, H., Jailani, N., Bakar, M. A., Zakaria, M. S., & Abdullah, S. (2009). A Generic Framework for Developing Map-Based Mobile Application. *International Conference on Electrical Engineering and Informatics* (pp. 434-440). Selangor: IEEE.
- Hasan, C. S., Ahamed, S. I., & Tanviruzzaman, M. (2009). A Privacy Enhancing Approach for Identity Inference Protection in Location-Based Services. *33rd Annual IEEE International Computer Software and Applications Conference* (pp. 1-10). Seattle: IEEE.
- Hoareau, C., & Satoh, I. (2009). From Model Checking to Data Management in Pervasive Computing: A Location-based Query-processing Framework. *International Conference on Pervasive Services* (pp. 41-48). 2009: ACM.
- Humerfelt, S. (2009, January 7). *Brief History of GPS*. Retrieved March 5, 2010, from http://home.online.no/~sigurdhu/GPS_history.htm
- Java Community Process. (2010). *JSR 179: Location API for J2ME*. Retrieved March 5, 2010, from The Java Community Process(SM) Program: <http://www.jcp.org/en/jsr/detail?id=179>
- Java Community Process. (2010). *JSR 68: J2ME Platform Specification*. Retrieved March 5, 2010, from The Java Community Process(SM) Program: <http://www.jcp.org/en/jsr/detail?id=68>
- Jeong, Y., & Kim, W. (2006). A Novel TPEG Application for Location Based Service using Terrestrial-DMB. *IEEE Transactions on Consumer Electronics* , 52 (1), 281-286.
- Juniper Research. (2010, February 23). *Mobile Location-Based Services Market to exceed \$12bn by 2014 driven by Increased Apps Store Usage, Smartphone Adoption and New Hybrid Positioning Technologies, According to Juniper Research*. Retrieved March 1, 2010, from Juniper Research: <http://juniperresearch.com/shop/viewpressrelease.php?pr=177>
- Kim, J. W., Jang, H. J., Hwang, D.-H., & Park, C. (2004). A Step, Stride and Heading Determination for the Pedestrian Navigation System. *Journal of Global Positioning Systems* , 3 (1-2), 273-279.
- Kim, S., Jung, Y., Lee, S., Lee, E., & An, S. (2008). Efficient Geocast utilizing Topology Information Database. *IEEE 8th International Conference on Computer and Information Technology Workshops* (pp. 210-215). Sydney: IEEE.
- Kulik, L. (2009). Privacy for Real-time Location-based Services. *SIGSPATIAL* , 9-14.
- Kupper, A., Treu, G., & Linnhoff-Popien, C. (2006, September). TraX: A Device-Centric Middleware Framework for Location-Based Services. *IEEE Communications Magazine* , pp. 114-136.
- Leavitt, N. (2010, February). Will NoSQL Databases Live Up to Their Promise? *IEEE Computer* , pp. 12-14.
- Liapis, D., Vassilaras, S., & Yovanof, G. S. (2008). Implementing a Low-Cost, Personalized and Location Based Service for Delivery Advertisements to Mobile Users. *3rd International Symposium on Wireless Pervasive Computing* (pp. 133-137). Santorini: IEEE.
- Liu, L. (2009). Privacy and Location Anonymization in Location-based services. *SIGSPATIAL* , 15-22.

- Mladenov, M., & Mock, M. (2009). A Step Counter Service for Java-Enabled Devices Using a Built-In Accelerometer. *The 1st International Workshop on Context-Aware Middleware and Services* (pp. 1-5). Dublin: ACM.
- Mobilizy. (2009, 23). *IBM and Ogilvy develop IBM Seer for Wimbledon together with Mobilizy*. Retrieved March 10, 2010, from Mobilizy: <http://www.mobilizy.com/deibm-und-ogilvy-entwickeln-ibm-seer-fr-wimbledon-2009-gemeinsam-mit-mobilizyenibm-ogilvy-launch-ibm-seer-wimbledon-2009-mobilizy>
- Myllymaki, J., & Kaufman, J. (2003). High-Performance Spatial Indexing for Location-Based Services. *Proceedings of the 12th International Conference on World Wide Web* (pp. 112-117). Budapest, Hungary: ACM.
- Oakes, C. (1998, January 6). 'E911' Turns Cell Phones into Tracking Devices. Retrieved March 5, 2010, from Wired: <http://www.wired.com/science/discoveries/news/1998/01/9502>
- Patrikakis, C., Voulodimos, A., & Giannoulis, G. (2009). Personalized Location Based Services with respect to privacy: A user oriented approach. *2nd International Conference on Pervasive Technologies Related to Assistive Environments*. Corfu: ACM.
- Rao, B., & Minakakis, L. (2004). Assessing the Business Impact of Location Based Services. *37th International Conference on System Sciences* (pp. 1-8). Hawaii: IEEE.
- Ribeiro, F. N., & Zorzo, S. D. (2009). LPBS - Location Privacy Based System. *IEEE Symposium on Computers and Communications* (pp. 374-379). Sousse: IEEE.
- Rubin, A. (2006). *Geo/Spatial Search with MySQL*. Retrieved February 15, 2010, from Scribd: <http://www.scribd.com/doc/2569355/Geo-Distance-Search-with-MySQL>
- Scarlett, J. (2010). *Enhancing the Performance of Pedometers Using a Single Accelerometer*. Retrieved February 10, 2010, from Analog Dialogue: <http://www.analog.com/library/analogdialogue/archives/41-03/pedometer.html>
- Song, C., Qu, Z., Blumm, N., & Barabasi, A.-L. (2010). Limits of Predictability in Human Mobility. *Science*, 327 (5968), 1018-1021.
- van Loenen, B., & Zevenbergen, J. (2007). Privacy (regimes) Do not Threaten Location Technology Development. *International Conference on Mobile Data Management* (pp. 238-242). Mannheim: IEEE.
- Vaughan-Nichols, S. J. (2009, February). Will Mobile Computing's Future Be Location, Location, Location? *IEEE Computer*, pp. 14-17.
- Vijayalakshmi, M., & Kannan, A. (2008). Enhanced D-Tree - An Index Structure For Window Queries In Location Based Services. *IEEE International Networking and Communications Conference, 2008. INCC 2008*. (pp. 124-131). Lahore: IEEE.
- Vishwanathan, R., & Huang, Y. (2009). A Two-level Protocol to Answer Private Location-based Queries. *Intelligence and Security Informatics Conference* (pp. 149-154). Richardson: IEEE.
- Weibenberg, N., Voisard, A., & Gartmann, R. (2004). Using Ontologies in Personalized Mobile Applications. *12th International Symposium of ACM GIS* (pp. 2-11). Washington, DC, USA: IEEE.

- Wikipedia. (2010, March 11). *App Store*. Retrieved March 16, 2010, from Wikipedia: http://en.wikipedia.org/wiki/App_Store
- Wikipedia. (2010, March 16). *Global Positioning System*. Retrieved March 16, 2010, from Wikipedia: http://en.wikipedia.org/wiki/Global_Positioning_System
- Wikipedia. (2010, March 15). *Google Maps*. Retrieved March 16, 2010, from Wikipedia: http://en.wikipedia.org/wiki/Google_Maps
- Wikipedia. (2010, March 14). *History of the iPhone*. Retrieved March 16, 2010, from Wikipedia: http://en.wikipedia.org/wiki/History_of_the_iPhone
- Wikipedia. (2010, March 14). *HTC Dream*. Retrieved March 17, 2010, from Wikipedia: http://en.wikipedia.org/wiki/HTC_Dream
- Wikipedia. (2010, January 19). *R-tree*. Retrieved February 20, 2010, from Wikipedia: <http://en.wikipedia.org/wiki/R-tree>
- www.3G.co.uk. (2004, November 10). *Assisted-GPS Test Calls for 3G WCDMA*. Retrieved March 5, 2010, from www.3G.co.uk: <http://www.3g.co.uk/PR/November2004/8641.htm>
- Xin, C. (2009). Location Based Service Application in Mobile Phone Serious Game. *International Joint Conference on Artificial Intelligence* (pp. 50-52). Pasadena: IEEE.
- Zhang, A., & Xu, Q. (2009). The Strategy Design of Compression and Transmission of cGML Spatial Data and Its Application in LBS. *5th International Conference on Wireless Communications, Networking and Mobile Computing*. Beijing: IEEE.

APPENDIX A – DISCUSSION OF DESIGN DECISIONS IN THE MOBILE INSURANCE BROKER PROTOTYPE

This appendix presents details on various design decisions and technologies or frameworks that were used in the development of the prototype (MoBroker).

A.1 IMPLEMENTATION OF LOCATION SERVICE AND DATA STORE

It was mentioned in Chapter 5 that location service and data store modules of the prototype LBS could be implemented as an OpenLS Tracking standard-based system. At the time of writing, no such open source system was available, so one had to be quickly developed. The developed system provided the necessary functions as specified by the OpenLS Tracking standard. However, for the purpose of this demonstration, the full specification including XML payloads was not implemented. Even though separate databases were not implemented for the application and location data stores, the data model was actually designed for separation between the two, so that the location data store can be moved easily at a later date.

There were three design decisions of note from the implementation of the location service and data store: (1) **Database provider selection**, (2) **Accessing spatial database features via Object-Relational Mapping** and (3) **Considering the use of component-based application architecture**. These are discussed in the subsections below.

A.1.1 Database provider selection

There are many options for database systems, including relational versus non-relational and those with or without spatial extensions, such as OpenLS SQL extensions. Four key criteria were used to evaluate and select a database provider:

1. Support for spatial calculations including distance, spherical distance, and intersection of regions.
2. Support for high scalability such as spatial indexing features.
3. Support for potential deployment platforms. In particular, LBS applications are suitable for deployment onto cloud services such as Amazon EC2.
4. Low or no cost to the project.

As discussed in Chapter 3, there have been significant advances in relational databases to support indexing of spatial data. These make use of spatial coordinates (e.g., R-Trees) rather than indexing only on one ordinate at a time. In addition, spatial extensions in databases provide native support for some spatial calculations. These would ensure higher scalability over non-spatial databases and hence only spatial database providers were considered, including MySQL, Postgres/PostGIS and Oracle.

Oracle and Postgres/PostGIS could provide the most complete implementations of the OpenLS SQL extension standard. Of particular notice, MySQL was missing a spherical distance calculation function that is needed for distance calculations taking into account the curvature of the earth. Oracle is a commercial product and the licence cost was beyond the budget of this project. Both

Postgres/PostGIS and MySQL were free to use; hence Postgres/PostGIS and MySQL were first evaluated.

Given Postgres/PostGIS's more complete and advanced spatial features, it was initially used during the early development phase. However, once Amazon EC2 was selected to be the deployment platform for this demonstration, the database was then migrated to MySQL. As an added benefit, there was more documentation on setting up MySQL on the platform, and so it was considered a safer option.

A.1.2 Accessing spatial database features via Object-Relational Mapping

It has been standard practice over the past few years to make use of an Object-Relational Mapping framework to decouple an application from its underlying database rather than using direct database access such as through JDBC. In line with this practice, there is Hibernate Spatial,³⁵ which is an add-on to the popular Hibernate framework. Hibernate Spatial provided support for MySQL, Postgres/PostGIS and Oracle GIS features, specifically OpenLS SQL extensions. Some minor extensions were required to cater for distance calculation function in the database.

Using Hibernate Spatial proved to be extremely useful when migrating from Postgres/PostGIS to MySQL. Only a minor extension to the provided MySQL dialect was required and the application code did not need modification.

A.1.3 Considering the use of component-based application architecture

Following the concept that the location service is a reusable module between LBS applications, this lends itself to the possibility of implementing the location service as a module for a component-based platform such as OSGi.³⁶ However, after a brief investigation into using OSGi, it became apparent that at the time of writing, the technology was not quite mature. In particular, OSGi-specific libraries need to be created for each library to be used in the application. While in general that should not be too difficult, it would require a fair amount of time for trial and error. Hence it was decided to stay with standard enterprise application architecture.³⁷

A.2 CLIENT-SERVER COMMUNICATIONS

The two key considerations for client-server communications were:

1. The implementation should be as simple as possible in terms of coding and also less demanding on processing power of mobile devices.
2. The design would need to take into account the nature of relatively low bandwidth and high latency of mobile data networks in comparison to fixed or wireless Ethernet.

³⁵ See <http://www.hibernate.org>

³⁶ Liapis, Vassilaras and Yovanof [2008] also suggested this approach.

³⁷ Shortly after this decision was made, SpringSource moved their dmServer project over to Eclipse, which indicated that perhaps OSGi was not quite ready for mainstream use.

A Representational State Transfer (REST)-based Web service was more appropriate than a SOAP-based Web Service for the first consideration. The server in this prototype was primarily an intelligent data store, so using a “resource”-based approach was more suitable than an RPC approach that typifies standard Web Services. For simple processing and minimal data transfer, Javascript Object Notation (JSON) data format was selected over XML.

To minimise the effect of latency, the interface was designed to require as few calls as possible. In addition, pre-loading of data was used, such as to load question and answer data for previous and successive questions.

The largest transfers were sending and receiving images, and hence these had significant impacts on the apparent responsiveness of the application. To minimise the size of image data sent from the mobile device, JPEG images were compressed to a point where pixelisation was just not visible. This reduced file sizes down from ~1MB to ~200kB. Since the mobile device has a fixed resolution of 320 x 480, there is no need to send a full 1600 x 1200 image. Images were scaled to an appropriate resolution on the server. This reduced download sizes from ~200kB to ~10kB per image. As a result of these optimisations, the application can be used on 3G networks comfortably, and would work on GPRS networks as well.

As discussed in Section 5.1.3.2, mobile applications would also need to accommodate network outages. For simplicity, the current prototype does not handle this. However, the iPhoneOS CoreData³⁸ framework provides a simple means of storing persistent data using SQLite³⁹ and an ORM-style syntax. By using this framework to temporarily store information from the existing client data model during network outages, answers can be submitted to the server when the network is available again.

A.2.1 Server-side implementation

Two options were tried for the server-side implementation. Firstly, a Grails⁴⁰ application was developed (based on version 1.2). This is a Java-compatible Groovy-based framework for developing Web applications with a core principle of “convention over configuration” (similar to Ruby on Rails). It theoretically could generate code for HTML-based data viewing and data entry screens based on human coded data model classes. In addition to HTML handling, it should only require a handful of lines of code to turn on JSON handling in order to provide REST Web services.

Unfortunately, parsing of JSON did not work as documented and so payloads could not be submitted to the server. While the Grails framework is promising, especially for developing Web applications, it was only used to provide data entry screens for testing. Given the tight timeframes for prototype development, a second option was sought.

This was to develop a Java-based JAX-RS service,⁴¹ which was the final solution used. JAX-RS is a standard where developers need only annotate service classes in order to expose REST Web

³⁸ See

<http://developer.apple.com/mac/library/documentation/cocoa/conceptual/CoreData/CoreData.pdf>

³⁹ See <http://www.sqlite.org>

⁴⁰ See <http://www.grails.org>

⁴¹ See <https://jsr311.dev.java.net/>

services; JAX-RS implementations would detect the annotations and provide the necessary HTTP request handling. There are a few implementations of JAX-RS now available including Jersey (the reference implementation), RESTlet⁴², RESTEasy⁴³ and Apache CXF⁴⁴. RESTEasy was selected as it had good Spring framework⁴⁵ integration, which was used in the application for dependency injection and transaction management, and provided JSON marshalling and unmarshalling without requiring any JAXB XML annotations by using the Jackson⁴⁶ library.

A.2.2 Client-side implementation

On the client, communicating with a REST Web service only needs an HTTP client. To wrap up handling of errors, as well as put together HTTP payloads, the ASIHTTPRequest⁴⁷ library was used. To assist with JSON marshalling and unmarshalling, the json-framework⁴⁸ library was used.

One important design decision was to ensure that the client data model, which is used to support the user interface, should closely match either the data model used for the interface between the client and server, or one that closely follows the server data model. This made model transformations relatively simple without necessary coupling to the server.

A.3 INERTIAL NAVIGATION SYSTEM

The mapping functionality in the prototype was based on the concept of inertial navigation or “dead reckoning,” whereby map segments were recorded based on the distance travelled for a given direction. The distance was measured by implementing a pedometer function using the iPhone’s accelerometer, and the direction was easily obtained from the device’s digital compass.

The general algorithm for recording the segments was to monitor for changes in direction. Once the change in direction reached a certain threshold, the current count of steps and distance walked was recorded.

The main challenge for this component was developing the pedometer. While it is possible to purchase pedometer add-on modules for the iPhone such as Nike+iPod, several academic papers, patents and application notes suggested that inbuilt accelerometers could provide fairly accurate step counts and distance calculations especially over a longer distance (on the order of +/- 5% accuracy).

There were many different algorithms proposed for determining when a step had taken place, and calculating the distance travelled in each step based on the magnitude of acceleration. The general

⁴² See <http://www.restlet.org>

⁴³ See <http://www.jboss.org/resteasy>

⁴⁴ See <http://cxf.apache.org>

⁴⁵ See <http://www.springframework.org>

⁴⁶ See <http://jackson.codehaus.org/>

⁴⁷ See <http://www.allseeing-i.org>

⁴⁸ See <http://code.google.com/p/json-framework>

approach for step detection appeared to use peak detection on the magnitude of acceleration, either a combined measurement between all accelerometers, or the largest value, with some logic to take into account changes in acceleration due to step up, swinging and step down phases.

In order to determine the distance travelled, some methods used lookup tables based on the user's height, while others used empirically determined equations.

Another common feature of these algorithms was the need to filter the accelerometer signals to remove noise. Most suggested a first order low pass filter, and the DC component had to be removed to eliminate the effect of constant gravity on calculations.

For this prototype, two algorithms were implemented. Firstly, one was implemented from Mladenov and Mock [2009], with distance determination from Kim, et al. [2004]. This worked fairly well in terms of accuracy in both step counting and distance with some minor algorithm adjustments. However, the step detection relied on a large buffer of samples that were batch processed every two seconds, which made the application difficult to use.

A second algorithm implemented was from Scarlett [2010], with some necessary modifications to constant values, and improved low pass and DC filtering. The benefit of this algorithm was that steps could be determined after eight samples (approximately 0.3 of a second) and so appeared more responsive to users. The accuracy reported by Analog Devices for this step counter and distance algorithm was +6/-4%, which proved to be approximately correct for the prototype.

The prototype demonstrated that rudimentary navigation without GPS or other absolute location collection technologies is possible. However, there are limitations. Firstly, the inertial navigation system is limited to relative navigation; the starting point would still need to be defined either manually or through location collection. Secondly, the digital compass, like standard magnetic compasses, has a lag. This may be acceptable for the prototype application or turn-by-turn navigation on foot; it may not be suitable for other high-speed tracking applications.

A.4 CAMERA OVERLAY WITH TEMPLATE PHOTOS – “AUGMENTED REALITY”

iPhone OS version 3.1 included access to an “overlay” view on the camera preview. This has been used in “augmented reality” applications such as the Layar browser. This allows potentially useful information to be presented about what the user is currently seeing. For iPhone applications this information is typically based on the current location and direction.

For the prototype, once a template photo is selected that was originally filtered on location, a faded image created by reducing its opacity is placed in the overlay view. Currently, the prototype only informs the user about changing direction in order to align with the direction of the original photo. This can be easily extended to provide navigation instructions based on the difference in location, such as “move 10 metres northeast.”

In the future it may be possible to process live camera images in real-time on smartphones to identify physical objects such as people, buildings or equipment. The Android platform provides access to live camera images, and some companies have started demonstrating the technology.



Worldwide CSC Headquarters

The Americas

3170 Fairview Park Drive
Falls Church, Virginia 22042
United States
+1.703.876.1000

Europe, Middle East, Africa

Royal Pavilion
Wellesley Road
Aldershot, Hampshire GU11 1PZ
United Kingdom
+44(0)1252.534000

Australia

26 Talavera Road
Macquarie Park, NSW 2113
Australia
+61(0)29034.3000

Asia

139 Cecil Street
#06-00 Cecil House
Singapore 069539
Republic of Singapore
+65.6221.9095

About CSC

The mission of CSC is to be a global leader in providing technology enabled business solutions and services.

With the broadest range of capabilities, CSC offers clients the solutions they need to manage complexity, focus on core businesses, collaborate with partners and clients, and improve operations.

CSC makes a special point of understanding its clients and provides experts with real-world experience to work with them. CSC is vendor-independent, delivering solutions that best meet each client's unique requirements.

For 50 years, clients in industries and governments worldwide have trusted CSC with their business process and information systems outsourcing, systems integration and consulting needs.

The company trades on the New York Stock Exchange under the symbol "CSC."

The information, views and opinions expressed in this paper constitute solely the author's views and opinions and do not represent in any way CSC's official corporate views and opinions. The author has made every attempt to ensure that the information contained in this paper has been obtained from reliable sources. CSC is not responsible for any errors or omissions or for the results obtained from the use of this information. All information in this paper is provided "as is," with no guarantee by CSC of completeness, accuracy, timeliness or the results obtained from the use of this information, and without warranty of any kind, express or implied, including but not limited to warranties of performance, merchantability and fitness for a particular purpose.

In no event will CSC, its related partnerships or corporations, or the partners, agents or employees thereof be liable to you or anyone else for any decision made or action taken in reliance on the information in this paper or for any consequential, special or similar damages, even if advised of the possibility of such damages.