# Location Based Services

**Gen Li**
4th Year Software Engineering
The University of Auckland
gli@aucklanduni.ac.nz

**ABSTRACT**

For most of us, technology is an indispensable aspect of daily life, governing ways in which we communicate with each other, means of transport, work, and entertainment. It would indeed be quite true to say in today's more modern societies that for most of us, almost every single minute spent is connected with some sort of technological grasp. Representative of this phenomenon, or perhaps more so iconic of a next step in its evolution is the branch known as ubiquitous computing. Ubiquitous is defined as "being everywhere", implying the likes of that mentioned above, which has already been established for decades. Ubiquitous computing on the other hand is a much more recent and specialized field, having a nature closely reminiscent of its name. As one can then imagine, such technologies would come with a strong focus on the aspect of location, and with that, related functionalities, of which a most prominent example would be the Location Based Service (which we will abbreviate to LBS). LBSs can be better defined as being services that incorporate the user's location, e.g. Global Positioning Systems (GPSs), local restaurant-finding applications, etc. Although LBSs have grown immensely prominent over the recent years, many problems can still be found with their use in the form of security issues, inefficient power consumption, and contextual-awareness deficiencies. The remainder of the article will be primarily focused on the discussion of such issues to provide some insights into the inner working and problems of related areas.

**Author Keywords**

Location based services, LBS, Context-Aware LBS architectures, Decentralised LBS architectures, K-Anonymity, Location Obfuscation Methods.

**ACM Classification Keywords**

Design; Human Factors; Measurement; Performance; Reliability; Security; Standardization

**INTRODUCTION**

Problems of recent LBSs include those of security, inefficient power consumption, and context-awareness deficiencies. Of these, the most prominent would be security, as the effect on end-users is by far the largest. LBSs expose users to the risks tied with having their locational information being accessible online. Problems arise when this accessibility leads to unwanted situations such as feelings of security breach, embarrassment or even exposure to physical harm. And so naturally, the area has drawn much focus and development. Methods derived to help enforce security within LBSs are primarily based on the anonymization of user information to prevent attackers from inferring relevant sensitive ones. Such information can be divided into two classes, sensitive information, and quasi-identifiers, as better explained in article [3]. Put into a scenario context, sensitive information is basically what its name implies (and so what the attacker is trying to get), and a quasi-identifier is information that the attacker can use to infer sensitive information, e.g. with knowledge of approximately where a user's home is situated (the quasi-identifier) the attacker can infer its actual location (sensitive information) if gained access to a comprehensive enough record of the user's locational data. A primary form of such anonymization is k-anonymity. K-anonymity comes in variations, but the essential principle behind it can be characterized by the following clause, 'with every quasi-identifier, there should be k-1 other quasi-identifiers in the same 'pool' with values that make each of the k instances indistinguishable from one another'.

Another issue related with security is the design of suitable interfaces that provide users with control of how much information they would like to disclose in the use of LBSs. Such designs are more prominent than one may think as the vessels are responsible for informing users of the risks they are exposed to at each level of disclosure and to help them make the right choices so as to better protect themselves. The issues will be touched upon in more depth with the help of results from surveys presented in articles [1] and [2] on user LBS security level preferences in regards to location anonymization methods and different means of displaying location information.

A simple but prominent issue with the use of LBSs is their general inefficient consumption of power. LBSs applications nowadays can be especially power taxing as

they can incorporate a good number of functionalities client side and in a parallel manner, such as GPS and Wi-fi. However, as shown by the study done in article [4], by using simple and practical means of conservation, power savings can be increased up to 85%. Such methods will be touched upon in detail.

The issue of the centralized anonymizer in LBS architectures and the lack of contextual awareness in LBS services in general will also be discussed, as well as their respective solutions from articles [5] and [6].

### Location Obfuscation

As we all understand, security is fundamentally one of the most prominent issues in all areas of life, for it represents the condition that if taken for granted may lead to the compromising of whatever the security belongs to, which in the worst case can mean a discontinuance of its existence. No different is it in the realm of computing, and especially so when people are involved. And so it's no wonder that such a large amount of effort have been put in the area of enforcing security in the use of LBSs. People need privacy, as it is strongly tied with security. When privacy is breached, so is one's sense of security. Coupled with man's instinctive need to fulfil this need of security, it could cause havoc in a person's life. And so it brings to us the fundamental issue, which is to preserve man's sense of security, during the use of LBSs. One's sense of security is more or less dependent on what he or she needs to feel secure, and so furthering the definition to the preservation of what one needs to feel secure. So essentially, all you need to do is identify what that is which a person needs to feel secure, protect it against from being compromised, and voila. However, it's never that simple in the real world, for everything is subjective after all. Hence the next step would be to find a best fit solution to cater the collective population, and intuitively, that pretty much is the best that one can do, in an ultimatum sense.

Before we delve anymore into the ethereal aspects of the matter, let us have a look at some practical means of enforcing security in the use of LBSs. Perhaps the most prominent concept in which most methods of location obfuscation are based on is the idea of making anonymous the need-to-be-anonymous information. Such a method is the satisfaction of k-anonymity among the relevant information. To help explain k-anonymity, it is helpful to divide the information within record instances to two types, sensitive values, and quasi-identifiers. As described in the introduction, sensitive values are those that the attacker wants to gain, and quasi-identifiers are the information that attackers can use to infer the sensitive values. The inference method used would be to discern between the quasi-identifiers of the records. Hence the satisfaction of k-anonymity helps to protect against such inferences by generalizing quasi-identifiers of the records, so that they'd be equivalently indistinguishable.

Going a level above, let's look at the 3 general LBS architectures presented in article [3], Trusted-Server, Untrusted Server, and Untrusted Server & user to user communication, which is presented in Figure 1 extracted from the article. Due to the large number of methods portrayed in the article, we will concentrate only on some basic ones surrounding the 3 architecture types.

**Figure 1: Different architectures for the communication with LBS providers [1].**

The Trusted-Server architecture involves an intermediate anonymizer server that does the anonymizing for the users, forwards the anonymized requests to the LBS provider, receives the LBS response and then forwards that back to the users. The most prominent role the anonymizer has in the architecture is the vessel in which different user requests can come together and then be used to help anonymize each other. The pros and cons of the architecture are both in its centralized architectural paradigm, in that the anonymizer takes care of a bulk of the processing from the phones as well as allowing an easy means of creating equivalence classes (a pool of requests that satisfy k-anonymity), but would mean that the trusted server really must be trustworthy, otherwise increasing the chances for an attack on all users whose data is stored on the anonymizer. Its centralized structure is also inherently simpler to attack than say a distributed one. A variation of the above architecture is one in which the anonymizer doesn't communicate with the LBS provider and instead forwards the anonymized requests back to the users after which they would start communicating with the LBS provider themselves. An advantage of this scheme would be the disconnection between the anonymizer and the LBS provider, hence reducing the likelihood of the anonymizer sharing sensitive information to the LBS provider.

The Untrusted Server architecture differs greatly from the Trusted Server architecture in that it doesn't contain an

anonymizer and so results in an inability to satisfy k-anonymity. The architecture instead usually adopts communication protocols that are based on a cryptographic method called Private Information Retrieval (PIR). The contents of the user requests become unknown to the LBS but their identity and the fact that they sent a request cannot be hidden. The architecture is probably more computationally expensive on both the client and LBS server compared with the Trusted Server architecture due to likely increases in the systems' structural complexity and nor is it as eloquent. It does however have the advantage of not having the collective security threat from the use of an anonymizer. Other obfuscation methods used in the architecture include the sending of a dummy location that is close to the original location or a cloaked region that contains it to the LBS.

The Untrusted server & user to user communication architecture is the same as the Untrusted Server architecture with the exception of the users being able to communicate with each other and work collaboratively to obfuscate location from the LBS provider. A variation to this is the scenario in which users do not trust each other. In this case, the system structure incorporates a proximity paradigm in which services are only available when relevant users are in proximity of each other, e.g. an application that only shows when friends are nearby is only available when those friends are near each other.

Now that we've had a little look at the lower level algorithms and higher level regimes used to enforce location obfuscation, it's time to delve into the user preferences of such via the surveys that were conducted in articles [1] and [2].

In article [1], the main problems posed included such problems as the ability to infer when a user is away from home via the use of a combination of Foursquare and Twitter, which was highlighted by the website pleaserobme.com. Extra emphasis is also put on the greater security risks involved with more continuous forms of LBSs due to the increased presence of the temporal property of the data, allowing for greater abilities of inference on information regarding the user. The survey itself consisted of 32 participants that were required to carry GPS data logging devices over a period of 2 months wherein loggings had to be sent back to the test team on a biweekly basis.

The data was then used in the visual modelling of the various obfuscation methods which consisted of; deleting, where a non-regular polygon that covered sensitive locations were deleted; randomizing, where each GPS point was moved randomly by a certain distance; discretizing, where the map was segregated into squares, and only squares and combination of squares containing the actual location is returned instead; subsampling, where data at certain points of time could be seen, e.g. only data from every 10min was visible, i.e. only 6:10pm, 6:20pm, 6:30pm, etc. is available; and mixing, where the location data of other users were added to the map. Primary information that the test-team wanted to elicit consists of the participants' willingness to share data, their obfuscation method preferences, their level of appeal for LBSs, and how valuable they regarded their own privacy data.

21/32 of the participants were willing to share the data publicly with GPS points removed from a regular polygon that contained sensitive locations. Mixing was the most popular obfuscation method with 15 participants, followed by deleting with 8, and randomizing with 7. Most users were willing to trade their privacy data for location based services. And the average bid for selling their privacy data was $150 for 1 month and $1400 for 12 months.

The amount of participants willing to share the data publicly really offers no insights in the current context as it is much too subjective, as with the results of the LBS appeal levels and the monetary value on privacy data. The results for obfuscation preferences seems sensible to a degree in that the mixing, deleting and randomizing do yield better promises of security upon impression.

Although the results were either unsurprising in that they could have been deduced via common sense, or practically of no use due to a lack of control in the test itself, the experiment does present areas that could be improved in the privacy control interfaces used to elicit the user preferences, hence taking a step forward in the progress of automation within the related area.

The survey conducted in article [2] attempts to elicit user preferences in the area of data visualisation and insight into the correlations between the choices and security concerns. 12 users participated and were given Nokia N95s to log location data for 2 weeks. Visualisations used in the survey consist of the map-based, text-based and time-based models which are presented in Figure 2 extracted from the article.

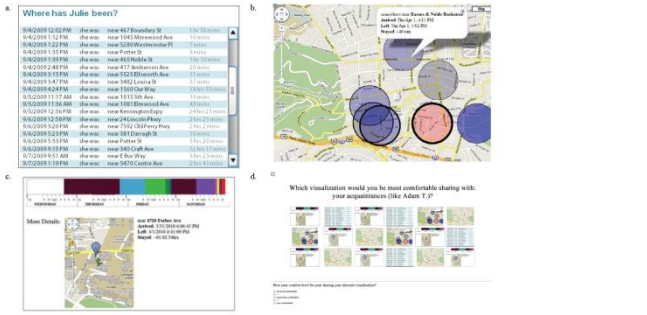


**Figure 2: (a) Text-based shows arrival, labels, & duration. (b) Map-based shows arrival, departure, duration, labels, spatiality, frequency, & sequence. (c) Time-based shows same features as (b). Visualizations made to be isomorphic. (d) Prompt shown to participants for choosing and evaluating visualizations [2].**

The map-based model uses halos to mark the location of the user. The centres of the halos are not necessarily the actual location of the user, and their size is dependent on the location of the user, e.g. if the user specifies that he is on Queen Street, then that halo will be larger than say if the user specified he was at Starbucks on Queen. The transparency level of each halo depends on the ratio in which the location was visited in regards to the other locations on the map. The text-based model contains the arrival time, location and duration of a location record. And the time-based model contains a timeline in which each coloured block is the period spent at a location, as well as a map and textual information that pops up when the corresponding coloured block is clicked on. The three visualisations are isomorphic, in that they contain essentially the same information.

An important aspect of the survey is that the semantic labels of location data were automatically generated (and further refined by the users themselves). This was done by using GPS, or the Skyhook API when GPS wasn't available, to get the location, and then get the location label using the location and mapping it with online databases. Geographic labels were extracted from the public databases using means of reverse geocoding, and the semantic labels via the Google Maps API. And so like the survey conducted in article [1], it too conducts an interesting side quest of testing effectiveness of a particular security interface.

At the end of the 2 week period, each user had to choose whom they were willing to share their privacy data with using the different visualisations out of 4 groups of people, Families, Close Friends, Acquaintances and Supervisors. The results were as expected as users selected by theory the safest options when asked about sharing their data, and the least safe when asked about viewing other people's data.

The results, as with those of the survey in article [1] were once again quite redundant, as they could have been inferred by through common sense, however the automatic generation of location semantic labels proved to be effective with a final statistic of approximately 72% success rate in accurately generating the label, which is quite a good result, showing prospects in using such a method in other applications.

### Power Consumption

The publication brings to attention the deficiency of LBS technologies' power management systems. Two issues are implied as being the main culprits of such an ordeal; the weakness of current battery, and perhaps hardware synergy properties, and the often wasteful implementations of the LBSs themselves.

Using means of power profiling, the author was able to quantify the average power expenditure of the many different LBSs as well as non-LBSs on his phone. The analysis was very detailed and probed down to a relatively atomic level, hence allowing for precise enough calculations for the exact determining of what and where the decompositions of the average power expenditures were.

Results show that there was much room for the minimization of LBS use over periods of time as there were often times when LBSs or their modules could be turned off or switched with the operation of a more conservative process. Furthermore, there were also many prospects in the optimization of location determination as the inter-changing between different location-tracking hardware, i.e. Wifi, GPS, etc. could be utilized. An example of this is the contrast between Wifi and GPS, where Wifi is less power consuming but less accurate, and so when less accuracy is in need, the use of Wifi in place of GPS could be implemented to reduce power consumption, which in the experiments yielded up to 85% in power savings. Using an error-model $emodel = ugps + (t - tgps) \times vest$, where ugps is the estimated accuracy of the last positional fix, tgps is the time of the last positional fix, t is the current time, and vest is the estimated velocity. The emodel variable is assigned a maximum value, which only when exceeded would prompt for the use of GPS or Wi-Fi to get a new locational fix. Using this method, results of up to 62.3% and 69.7% in expenditure savings for 100 and 200 metres intervals respectively were recorded on pedestrians walking in a residential area.

The area of mobile device battery life enhancement is an important aspect of development. The publication introduces readers to the epidemic of in-efficient LBS power consumption and presents some effective solutions that would appear simple and practical.

### Contextual Awareness

Article [5] presents the problem of "inherent rigidness" of current LBS technologies, in that they are not very lively or context aware. To address this, a context aware architecture for LBS systems is presented.

The architecture consists of the Task Manager, User Interface, Trigger Manager, Service Manager, and Context Manager. The Task manager is in charge of coordinating the tasks of the system; the Trigger Manager is in charge of determining when a task will run; the Service Manager handles the rawer and lower-level logic of the service; and the context manager is in charge of determining contexts via the sensing of both context-related information from both external and internal sources. Figure 3 extracted from the article presents this model.

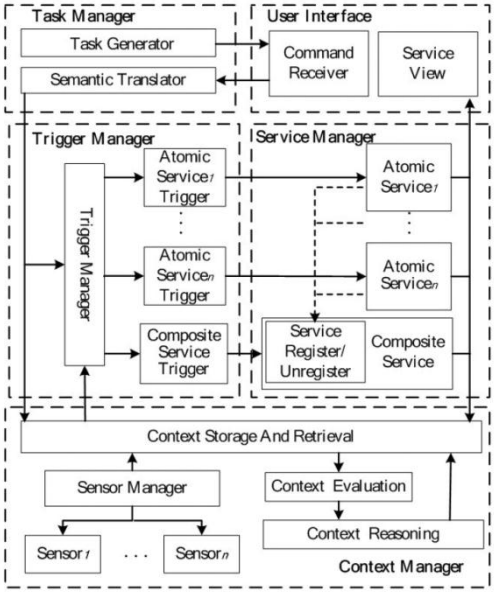


Figure 3: The generic context aware LBS architecture

The specifics of the publication were mostly of a theoretical type and contained no apparent information in regards to the practicality of the system. The intuition behind the design does seem doable, though I personally could not judge.

Context-awareness is perhaps one of the most important recipes in recreating "life" within computers. The architecture presented by the publication does present readers with some good insight into the inner workings of a contextual aware system.

### Decentralised Anonymization

The architecture presented in article [6] represents a form of the Untrusted Server architecture presented above. It strives to rid of the epidemic of the security issues posed by the presence of an anonymizer whilst keeping the performance levels as high as possible.

The idea behind the design is to use the mobile service operator as the keeper of the service locations, and the LBS as the keeper of location identifiers. Together, the two groups of information can be converted into an abstract space (Matching service) wherein only the relative distances between the locations can arise, and so even if an attacker is to compromise the service, he cannot get the actual locations. And because the mobile service operator and LBS both can only possess one of the sets of information, they cannot infer the location either, hence making a very safe system.

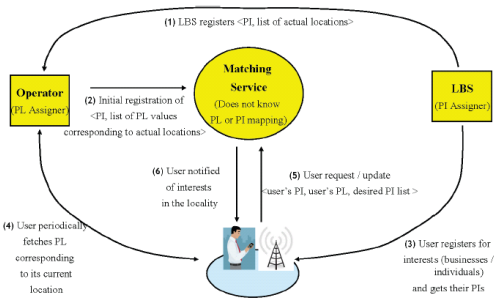


Figure 4: The decentralized LBS architecture

### Conclusion

So as one can see, though LBSs have become very prominent over the recent years, there are still many problems associated with their use, which is somewhat surprising.

### Future Work

There is still much left to do to make LBSs a safe enough service that can be used without having to put any thought in the matter. A relatively certain direction in which developmental effort should be put in is the design of quality privacy control interfaces, but other than that, there would appear to be much room for improvement in most of the areas portrayed in this article.

## REFERENCES


1. Brush, A.J.B., Krumm, J., Scott, J. Exploring end user preferences for location obfuscation, location-based services, and the value of location. In *Proc. UbiComp 2010*, ACM Press (2010), 95-104.
2. Tang, K.P., Hong, J.I., Siewiorek, D.P. Understanding How Visual Representations of Location Feeds Affect End-User Privacy Concerns. In *Proc. UbiComp 2011*, ACM Press (2011), 206-216.
3. Terrovitis, M. Privacy Preservation in the dissemination of location data. In *Proc. SIGKDD 2011*, ACM Press (2011), 6-18.
4. Kjærgaard, M.B., Location-Based Services on Mobile Phones: Minimizing Power Consumption. *Ext. Pervasive Computing, IEEE 2012*, IEEE Xplore 2012, 67-73.
5. Zhu, T., Wang, C., Jia, G., Huang, J. Toward Context-Aware Location Based Services. In *Electronics and Information Engineering (ICEIE 2010)*, IEEE Xplore 2010, V1-409 – V1-413.
6. Jaiswal, S., Nandi, A. Trust No One: A Decentralized Matching Service for Privacy in Location Based Services. In *Proc. MobiHeld 2010*, ACM Press (2010), 51-56.