Universidade Nova de Lisboa
Faculdade de Ciências e Tecnologia
Departamento de Informática

Dissertação de Mestrado

Mestrado em Engenharia Informática

Unified Cooperative Location System

David Precatado Navalho (25987)

Lisboa
(2009)

Universidade Nova de Lisboa
Faculdade de Ciências e Tecnologia
Departamento de Informática

Dissertação de Mestrado

Unified Cooperative Location System

David Precatado Navalho (25987)

Orientador: Prof. Doutor Nuno Manuel Ribeiro Preguiça

Dissertação apresentada na Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa para a obtenção do Grau de Mestre em Engenharia Informática.

Lisboa
(2009)

Acknowledgements

Em primeiro lugar gostaria de agradecer ao Departamento de Informática (DI) da FCT-UNL, pela bolsa de investigação concedida.

Em particular, gostava de agradecer ao meu orientador, Nuno Manuel Preguiça, não só pela orientação, pelo apoio e pela disponibilidade, mas também pela enorme dedicação e pelas oportunidades proporcionadas ao longo da realização deste trabalho.

Queria também agradecer aos meus colegas e amigos de trabalho do DI, em especial ao Bruno Félix, Hélio Dolores, João Soares, Maria Café, Nuno Luis, Pedro Sousa e Simão Mata pela disponibilidade, apoio, ideiais, discussões, comentários, boa disposição e longas noitadas.

Ao professor João Lourenço, por todos os conselhos, discussões e partilhas.

Aos meus familiares, pela paciência, compreensão e incentivo que me deram durante todo o percurso académico.

Quero também deixar o meu agradecimento à Catarina, não só pelo incentivo, paciência e enorme compreensão demonstrados, mas acima de tudo por fazer parte da minha vida.

Finalmente, a todos aqueles que, directa ou indirectamente, contribuíram para a realização deste trabalho.

Summary

The widespread use of smaller, less expensive, and more capable mobile devices, has opened the door for more complex and varied mobile computing applications. Additionally, manufacturers are increasingly equipping these handheld devices with every type of wireless connectivity and sensors that can be explored for providing more complex services.

In recent years, several techniques for location estimation have been developed, providing different degrees of accuracy. Some of these solutions require the installation of specific hardware in the environment, while others explore the existing infrastructure. In particular, it is possible to explore the existing communication infrastructure to build a location system relying on signal strength measures.

However, while several systems already exist to locate users based on different approaches, there is no single one that is good for every situation while providing high accuracy, low cost and ubiquitous coverage. Not only that, but very few research has been made regarding on how a group of users can cooperate to improve accuracy or reduce energy consumption while using the location system.

This work presents the Unified Cooperative Location System, a modular and extensible location system. Its modular design can use every available technology on each device and different algorithms for location estimation. This approach allows to provide location services with high availability by relying on different technologies. It also allows to reduce the energy consumption on devices by sharing the responsibility of executing energy-heavy operations.

The system also includes an information exchange mechanism, allowing devices to gather location information from nearby users, like GPS or Wi-Fi, which would otherwise be unavailable for some. The results of our experiences show that the possibility of exchanging GSM information provides a practical solution for location estimation based on multiple GSM signals, thus significantly increasing location accuracy with this technology.

Keywords: location system, fingerprinting, information sharing, GPS, GSM, Wi-Fi, Bluetooth, mobile devices.

Sumário

O aumento do uso de dispositivos cada vez mais pequenos, mais baratos e com maiores capacidades, abriu a porta para variadas aplicações mais complexas. Além disso, os fabricantes estão constantemente a equipar estes dispositivos portáteis com todos os tipos de conectividade sem fios e sensores.

Tirando partido das novas características destes dispositivos, um grande número de técnicas e sistemas foram propostos para fornecer localização. Algumas destas soluções requerem a instalação de hardware específico nas proximidades, enquanto outras exploram a infra-estrutura existente de comunicações sem fios. Em particular, é possível explorar a infra-estrutura para construir um sistema de localização baseado em medidas de força de sinal.

No entanto, apesar de existirem vários sistemas para localizar pessoas, partilhar locais, ou mesmo fornecer serviços, não existe um único que seja bom para todas as situações, fornecendo boa precisão, baixo custo e cobertura ubíqua. Além disso, muito pouca pesquisa tem sido feita sobre como um grupo de utilizadores pode cooperar para melhorar a precisão ou reduzir o consumo de energia enquanto usam um sistema de localização.

Este trabalho apresenta o Unified Cooperative Location System, um sistema modular e extensivel. O seu desenho modular permite usar todas as tecnologias disponiveis para cada dispositivo, e usar diferentes algoritmos para estimar a localização. Esta abordagem permite disponibilizar serviços de localização com alta disponibilidade, ao usar diferentes tecnologias. Também permite reduzir o consumo de energia em dispositivos, ao partilhar a responsabilidade de usar operações com altos gastos de energia.

O sistema usa um mecanismo de troca de informação, que permite aos dispositivos partilhar informação de localização entre utilizadores próximos, tais como GPS ou Wi-Fi. Os resultados obtidos pelas nossas experiências demonstram que a possibilidade de trocar informação GSM fornece uma solução prática para recolher múltiplos sinais GSM, aumentando significativamente a precisão de localização usando esta tecnologia.

Palavras-chave: sistema de localização, fingerprinting, partilha de informação, GPS, GSM, Wi-Fi, Bluetooth, dispositivos móveis.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Context

Improvements in hardware and wireless technologies during the 90's have led to the development of multiple mobile devices, creating a new computer environment usually designated as mobile computing.

Recently, the widespread use of smaller, less expensive, and more capable devices, has opened the door for more complex applications, of which, location-aware applications are just an example. Some location-based applications are already becoming an integral part of our lives, GPS navigation probably being the most widely publicized and used of these systems, allowing for user location and navigation.

Handheld computers are now capable of running many different types of applications. Additionally, manufacturers are increasingly equipping these handheld devices with every type of wireless connectivity, including Bluetooth, WiFi and sensors, such as GPS.

Taking advantage of the new characteristics of these devices, and the fact that wireless technologies are becoming more and more widespread, several research and implementations are already underway, in an attempt to use all the available information a user can gather at any place, and use it to his advantage to locate himself or other individuals.

The ability to locate or track users can be further extended beyond simply locating the current position, however. Social applications can be developed which, based on the users current context, can allow for increased services, such as simple co-located instant messaging applications and location sharing amongst friends, gaming environments, children locators or even offering information or services based on the users current location [LdL08].

However, while several systems already exist to locate users, share locations, or even provide location-aware services, there is no single one that is good for every situation while providing high accuracy, low cost and ubiquitous coverage.

1.2 Motivation

Several systems have been built or prototyped, trying to provide a location system that is cost-efficient, highly accurate and that provides an ubiquitous coverage. Some require the installation of specific hardware on the environment [WFG92,PCB00,HHS+02]. Not only that, but many of these systems also require users to carry around a special device. These implementations usually impose additional costs to install and use the system. That fact, coupled with the need for users to carry extra devices, usually means these systems are only used on specific environments (e.g. for research at universities or organizations with enough money and a critical need of a location system), and usually indoors, since installing devices outdoors would mean also installing countermeasures against vandalism, etc. These systems tend to achieve very good accuracies.

Other implementations explore the existing infrastructure [RD07, BP00, VLHdL07, LCC+05], taking advantage of the fact that wireless communications are almost ubiquitous across the globe. For example, by reading the signal strength of the existing infrastructures, it is possible to create a map of the readings. The construction of these maps requires either manual input at the time of reading, stating the current location, or an automated method of storing the information with the current coordinates where the user is located at the time.The typical method used for this second process, is to have a second mechanism (usually a GPS receiver) to record the current location and store it together with the reading. When a user later tries to determine his location, the system applies an algorithm that tries to find the closest matches of the signals detected at the time with the data stored on the database. It then retrieves those closest matches, and infers the user is at the location that corresponds to those readings.

Usually, the implementations using the existing infrastructure can cover a much wider range than a specialized system, since most technologies are available almost anywhere, and a simple PDA or mobile phone can use these wireless technologies. Also, these solutions usually can be used both indoors and outdoors. However, not only are these maps based on estimates, which already reduce accuracy, but there usually is a training phase implied, which is very costly in terms of time.

Very few studies exist on how a group of users can cooperate to improve accuracy or reduce energy consumption for location systems on mobile phones. One solution [VPKE08] has been provided to share information to relatively distant users, though with very complex algorithms that require a nearby server to do the calculation, and no actual accuracy improvement is observed. Some attempts of using different location technologies have been proposed [RD07], but using only some of the technologies available in a single device, and only using one when the other fails.

Finally, there are very limited studies regarding Bluetooth [DDRC04], disregarded because the technology is usually associated with mobile environments, thus rendering it less useful for location. However, this wireless technology is present in most if not every new device that comes to the market. Not only that, but the fact that it uses a low-range, low-energy communication protocol may provide a useful support for location systems.

1.3 Implemented Solution

The Unified Cooperative Location System provides an unified location system that uses information obtained from multiple sources available to mobile phones, while providing the main requirements of good accuracy, ubiquitous coverage, low cost and user privacy.

There are several aspects involved in the location estimation process. Information can be obtained from multiple different technologies. Different technologies may require different algorithms for location estimation. The user may wish to use a method that consumes less energy at some time. Also, several algorithms require that data is gathered and stored, for later retrieval while using the location algorithm. The UCLS architecture has a modular approach, and the insertion of a new location estimation algorithm, information gathering technique, data retrieval or even a new technology, are very simple to add.

Taking advantage of the existing wireless infrastructures, the mobile devices using the system are capable of scanning the surrounding with all technologies available to them through the use of Sniffers. These Sniffers can regularly scan the mobile device's surroundings, gathering the available information for each wireless technology. The possibility of using all the available information, may allow more precise location estimation, as well as the possibility of conserving energy by using only a subset of the technologies.

To further improve the location information availability, the system uses an information exchanging mechanism. This method uses a ubiquitous communication protocol (Bluetooth) to contact nearby devices and exchange location information between them. This process allows one device to use information from a technology that it does not include (e.g.: GPS) by relying on information received from nearby devices.

The gathered location information can then be reported to the user, used in the process of location estimation or stored in the system's database. The database module provides the mechanisms to retrieve the necessary location information to estimate a device's current location. Although the sharing of information is anonymous, a user could still possibly detect the surrounding device's bluetooth address, although any regular bluetooth scan would achieve the same.

Since location information using current fingerprinting techniques requires an initial and time-consuming method of data-gathering, the system also provides methods to share information with an external server. This allows a community to contribute with information, allowing a faster creation of the necessary information for location estimation.

When locating a user, several algorithms may be available on the system. By using the available wireless information and the available location information in the database, the system will use its available algorithms, attempting to estimate the current location. To provide privacy, it is important that the system locates the client only with its available information, without resorting to external services for the location process.

To provide ease of use, the system provides an API, which allows a programmer to easily implement an application on top. The API provides the basic methods for location estimation, including the information gathering capabilities and location estimation. Additionally, configuration methods provide the ability to enable and disable the available technologies, algorithms, external databases access and sharing mechanisms.

1.4 Main Contributions

This work contributes with the design and implementation of a location system that merges various location techniques to provide the users not just with different accuracy/energy choices, but also to provide fall-back techniques for when one type of location technology is not available.

This research studied new methods of sharing location information between users to provide accuracy improvements and energy savings, as explained in the previous section. The results obtained show that the proposed system provides a practical solution for location estimation with its information exchange mechanisms, even when relying only in GSM technology.

1.5 Organization

After this chapter that introduces the problem and presents the direction of this work, the remainder of this document is organized as follows: Chapter 2 introduces location estimation solutions presents related work.

Chapter 3 describes the Unified Cooperative Location System Architecture, followed by its Implementation details in Chapter 4. Chapter 5 introduces the tests of the system's usability, power consumption and performance.

Finally, Chapter 6 presents the conclusions of this work as well as some possible improvements for the future.

2

Related Work

In this chapter we review related work on location systems. We start by presenting basic techniques used in the proposed approaches and then look with more detail to some of the most important solutions previously proposed in literature.

2.1 Basic Techniques

Location systems typically go through two phases: an offline phase, where a map of the area to be used with the system is built, and a second phase where the system scans the surrounding area, gathers the available information, and tries to infer the current location by using an algorithm. This section starts by presenting some basic definitions and algorithms used for determining location. It then proceeds for discussing several aspects related with the presented algorithms.

2.1.1 Performance of Location Systems

Location systems performance can be easily evaluated based on their accuracy, coverage, cost and privacy.

Accuracy is the most often-used metric of location systems and refers to the correctness of a system's location estimates. This measure is often expressed as an error distribution for coordinate-based location systems, typically in meters or centimeters, as well as a "median error" which indicates the amount of times the estimates are that accurate. Symbolic location accuracy is usually expressed as a percentage, since that measure is either correct or not.

Coverage refers to the physical area the system is able to work properly, if at all. It usually is a descriptive metric instead of a quantitative one. GPS, for example, is any place with a clear view of at least four GPS satellites.

Cost is related to not only time and money requirements to deploy and maintain a given location system infrastructure, but also to the incremental cost of adding one more device, person or object to be located on a system.

Finally, *Privacy* is also one of the key characteristics of a location system. The ability for a user to estimate his location by himself affords the best privacy measure. However, some systems perform some or all of the location estimation, thus leaving the user to have to trust the system's designers and managers to maintain his privacy.

2.1.2 Location Algorithms

Triangulation

Triangulation is the process of determining the location of a point by using the geometric properties of triangles [Bor01]. This technique is used for many purposes, including surveying, navigation, metrology, astronomy, binocular vision, model rocketry and gun direction of weapons [LdL08]. It can be divided into the sub-categories of *angulation* and *lateration*.

Angulation uses angles to determine the position of an object. Two dimensional angulation requires two angle measurements and one length measurement between two known points. In the example of figure 2.1, it is possible to calculate the location of X with the measures of the angles shown by using basic trigonometry rules.

For three dimensions, one length measurement, one azimuth measurement, and two angle measurements are needed to specify a precise position.

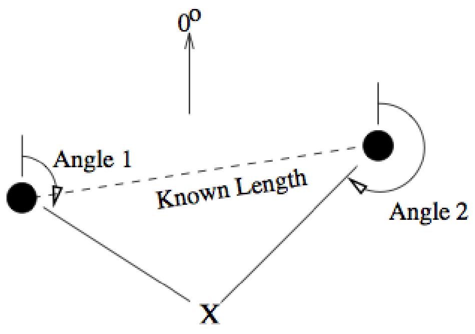


Figure 2.1: Example of locating object 'X' with a 2D angulation, using angles relative to a 0^o reference vector and the distance between the two reference points (from [Bor01]).

Lateration is used to calculate the position of an object by measuring its distance from multiple reference positions. Two dimensional lateration requires distance measurements from 3 non-collinear points, as shown in figure 2.2. In 3 dimensions, distance measurements from 4 non-coplanar points are required.

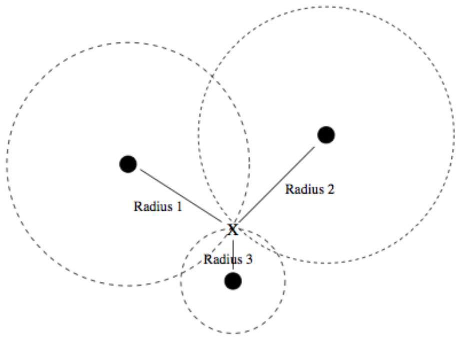


Figure 2.2: Example of locating object 'X' using a 2D lateration (from [Bor01]).

When using Lateration, there are three usual techniques to measure the required distances to the object being located: *Direct*, *Time-of-Flight* and *Attenuation*.

Direct measurements of distance use a physical action or movement to physically measure the distance between two points. Although simple to explain and understand, these types of measurements are very complex and difficult to obtain automatically. *Time-of-Flight* measurements consists in measuring the time it takes to travel between an object to some point P, at a known velocity. For example, the Cricket system [PCB00] takes advantage of the properties of ultrasound and Radio Frequencies (RF) to obtain the distance a user is from a beacon. Since the speed of sound in the air is much smaller than the speed of RF, if a beacon transmits both signals at the same time, the object being located simply has to record when the RF signal arrives, and count the time until the ultrasound arrives too. Then, it is a simple question of measuring the distance the ultrasound covered, by multiplying its speed by the time it took to arrive at the object (the difference between the time of arrival of the ultrasound and the RF, assuming the time of propagation of RF is 0, since it is much faster than an ultrasound).

Measuring times-of-flight has its issues, the first being reflections. Ignoring pulses arriving at the object via an indirect path caused by a reflection in the environment is a challenge, since direct and reflected signals look the same. Another issue in taking these measurements is agreement about the time. Systems like Cricket (see section 2.2.2) only need one measurement, since the object can calculate the time-of-flight of the ultrasound transmission by taking advantage of two different propagations models. However, systems like the Global Positioning System cannot make that kind of calculation (see section 2.2.1). In this case, the receiver is not synchronized with the satellite transmitters and thus cannot measure the time it took the signal to reach the ground from space. Therefore, an additional GPS reading is necessary for computing time.

Attenuation is the decrease of an emitted signal relative to the original intensity from an emission source. Some systems use functions or algorithms that correlate attenuation and distance for a type of emission, inferring the distance an object is from a point P by measuring the strength of the emission when it reaches point P (see sections 2.1.3 and 2.2.5).

Signal Strength Fingerprinting

Technologies like WiFi and Bluetooth have two inherent properties: spatial variability and temporal consistency [LdL08]. Due to the short range of these technologies, and the way signals are blocked and reflected by obstructions in the environment, the strength with which an emitter is observed varies considerably spatially, even over distances as small as a meter. Also, since the strength of a signal tends to be temporally consistent, a good spot to establish a connection to a particular emitter, will very likely still be a good place to connect in a few minutes, the next day, or the following month.

Location estimation using signal strength fingerprinting takes advantage of the aforementioned properties, involving two distinct phases: the mapping phase and the location estimation phase. The *mapping phase* consists in performing a site survey, in which the visible APs and their observed signal strengths are recorded along with the location in which the observation was taken. In order to provide good accuracy, the readings need to be of sufficient high density. Thus, a reading is usually collected every few square meters to allow for a good coverage.

After retrieving all the data, one can easily *estimate his location* by performing a radio scan, and matching with the previously gathered data whose scan most closely matches its observation.

Proximity

Proximity based location techniques are used to determine when an object is "near" a known location [Bor01]. Proximity techniques may often need to be combined with identification systems, if there is no subjacent technique already present.

Detecting physical contact with an object is the most basic technique of proximity sensing. Technologies for sensing physical contact include pressure sensors, touch sensors, and capacitive field detectors [HS99].

Another common technique relies in *monitoring wireless cellular access points*. Objects **X**, **Y** and **Z**, as depicted in figure 2.3, are located by monitoring their connectivity to one or more access point in a wireless technology. Objects belonging to the same network(s) can be inferred are in close proximity to each other.

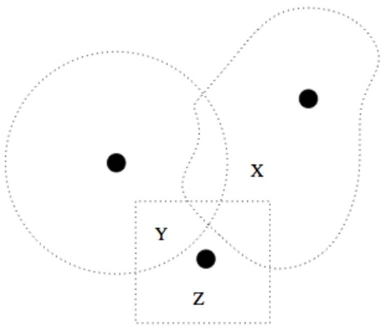


Figure 2.3: Example of a proximity technique based on wireless access points. In this figure, a radio network might have the shape of the region containing object **X**, while the square shape may belong to an infrared technology, contained inside a room (from [Bor01]).

Observing automatic ID systems is another proximity location-sensing technique. This includes systems like credit-card point-of-sale terminals, computer login histories, land-line telephone records, electronic card logs, identification tags, etc. If the device scanning or identifying the object has a known location, the location of the object can be inferred.

Another possible approach is to use cameras to locate nearby objects.

2.1.3 Signal Strength Modeling

Taking advantage of the fact that wireless signals propagate in predictable ways, a user can predict the most likely distance to an emitter by modeling the propagation of the radio signals emitted by such devices [LdL08]. These models can range from simple heuristics like simply assuming an 802.11 can be heard for 100 meters in all directions from the access point, to complex ones that consider antenna types and the attenuation of various building materials.

In order to infer positioning a positioning system, this type of modeling also requires the construction of maps, albeit this modeling is fairly simpler and faster to build than the ones needed for Signal Strength Fingerprinting. The radio maps for these models usually contain an estimation of the position of each known radio beacon, as well as other useful characteristics when available [LCC+05] (e.g., antenna height and direction, transmission strength, etc.). Of course, in case of a small area to map (e.g., Universities, an office, etc.) the location of APs can be manually determined, which helps in determining the users current location later on.

2.1.4 MinMean and MinMode Algorithms

Some systems listen for surrounding communications, and try to choose the nearest beacon among the detected ones in order to make its calculations based on the available information. The Cricket system is such an example, depending on these algorithms to detect the nearest beacon, which allows it to determine the nearest place to the receiver [PCB00]. The MinMean and MinMode simple algorithms allow for a quick and easy estimation of the closest beacon.

The MinMean algorithm calculates the mean distance for each unique beacon detected on the current position. Then, it selects the beacon with the minimum mean as the closest one. While the MinMean algorithm is very simple, it does not take into consideration multi-path effects. The MinMode is a heavier algorithm, but achieves higher accuracy by calculating a highest likelihood estimate by computing the per-beacon statistical modes over the past n samples. After assigning the estimated mode for each unique beacon, the system then selects the beacon with the lowest value as the nearest.

2.1.5 Wardriving

Wardriving is the act of searching for Wi-fi wireless networks by a person in a moving vehicle, using a portable device capable of detecting such networks. While searching for these networks, it is a common practice to store the signal strength of the access point located, as well as using a GPS device to store the location where the access point was located. The information gathered is then usually logged on community websites, effectively providing maps of known networks.

These databases are becoming increasingly popular, providing users with free, readily available maps for several areas spread throughout the world, which can then be used with Signal Strength Modeling methods for building location applications [LCC$^+$05].

Although less accurate than an human introduced database, the fact that these map databases are not only fast and easy to generate, but also increasingly widespread among communities online, makes it a feasible solution to obtain information about wide regions, thus providing a less accurate but faster beacon location map, thus bypassing one of the main problems in building maps for location systems, which usually require human interaction throughout the whole mapping phase.

2.1.6 Self-Mapping in 802.11 Location Systems

Location systems based on scanning nearby radio sources (e.g. 802.11) can estimate the position of a mobile device with reasonable accuracy and high coverage. However, all these systems also require a previously built map based on Signal Fingerprinting (section 2.1.2), so they can infer the position based on previous knowledge. Building these maps is very time intensive, and even when no physical human input is required, like in war-driving where the scans and data gathering are done automatically, it is still very time-consuming, and the database needs to be refreshed on a regular basis, in order to have a good database. Self Mapping algorithms [LHSC05] are a way of trying to avoid constant calibration or new data gathering processes, by allowing a system to automatically build and improve its radio map while being used.

Graph Algorithm for Self-Mapping

This algorithm uses two types of input data: a *seed set* and *radio traces*. The seed set contains the known or estimated locations, represented in latitude and longitude, of a set of beacons. Each beacon on the set must therefore have a unique identifier. The radio traces are made up of a collection of time-stamped radio scans. Each of these radio scans contain a timestamp as well as the group of detected beacons ids and their signal strength.

After building the map with the original seed set, a user can start using the self-mapping system. As the users moves about, new APs will be detected, and for each pair of beacons, b1 and b2, detected under a fixed sliding window (e.g. 1 second), the system estimates their maximum distance based on Seidel's Model for propagation of signals in the wireless networking band [SR92](see section 2.1.3 for signal strength modelling). If no edge exists between a pair of beacons *b1* and *b2*, one will be added with the weight of the calculated distance. If an edge already exists, but the weight is higher, then it will be replaced with the new estimated distance. After assigning these weights, we still have to assign a location to any new node introduced, since only the initial seed set had coordinates assigned.

Nodes that belong to the seed set are special nodes in that their location does not change. In order to assign locations to the new nodes, one must make sure to minimize violations of the constraint that for all nodes *i* and *j* that are connected by an edge, the distance between them should be less than the weight of the edge. Iteratively, each new node will be assigned the location of the nearest seed node in the graph (if no node is available, no location is assigned). Then, since this will most probably violate several constraints, a random new node is chosen and its location is alternated randomly, in an attempt to reduce the error. If a reduction can be achieved, the node is moved to that new location and its neighbors are updated with the new information. This continues until either the error is zero (there are no violations) or the algorithm cannot find any new improvement. Once the algorithm completes, a new radio map is complete.

2.2 Systems

2.2.1 The Global Positioning System

GPS is by far the most widely used location technology [LdL08]. GPS is designed as a passive one-way location system, where all signals are transmitted by orbiting satellites and the receivers simply determine the position based on the information received. This technique allows for a complete worldwide coverage and an unlimited ability to scale. Its accuracy ranges from several meters to a few centimeters, with the later requiring much more expensive equipment (tens of thousands of dollars).

The Global Positioning System has a major limitation, however. GPS signals do not penetrate well through water, soil or walls, and thus it cannot be used indoors, underwater and it even can fail to work in urban areas where very tall buildings surround a user. Accurate location requires an unobstructed view of at least four orbiting satellites.

Architecture/Implementation

The basic GPS-positioning algorithm is supported by all GPS receivers and allows the receiver to estimate its position in three dimensions (latitude, longitude and altitude) [LdL08] by tracking four or more satellites.

To calculate the GPS receiver's location, the estimated positions of the satellites are used, together with the distance from the receiver to the satellites. In order to estimate the distance, both the receiver and the satellite need to be synchronized. Since the receiver needs to be a low cost solution for the end-user, it cannot have an atomic clock, which would be too expensive.

To solve this issue, the receiver computes time, using the incoming signals from four or more satellites [Bra06]. By solving the equation presented in figure 2.4 for the four satellites, the receiver calculates his cartesian coordinates and the current time.

Strong aspects

This system provides one of the best solutions for outdoors mapping, achieving a very high accuracy. It already provides an near ubiquitous infrastructure when outdoors, and many high-end mobile devices already come equipped with GPS chips. Our system uses GPS, when available, to achieve the best available accuracy outdoors, needing only to activate it and use the location information provided. Averaging GPS information with close devices may allow to improve accuracy.

$$R_i = \sqrt{(x_i - x)^2 + (y_i - y)^2 + (z_i - z)^2} - b$$

Figure 2.4: The basic GPS positioning algorithm. Given a distance R_i to a satellite located at (x_i,y_i,z_i), the receiver calculates its location in three-dimensional space(x,y,z) and the clock bias b

2.2.2 The Cricket Location Support System

Cricket [PCB00] is a location-support system for in-building, mobile, location-dependent applications. It achieves a room-sized granularity to within one or two square feet, effectively locating users in specific portions of rooms.

The Cricket system is composed by a set of beacons and listeners. A beacon is a small device attached to some location that continuously disseminates information about a geographic space to listeners. A listener is a small device attached to a mobile or static node, that listens to the messages disseminated by the beacons, using these same messages to infer location. The system provides an API to programs running on the node, thus allowing them to learn where they are and act accordingly, like advertising their location to a resource discovery service, or even an application present on the node itself.

Architecture/Implementation

The Cricket system includes three fundamental mechanisms: location determination, the listener algorithms and techniques for handling beacon interference and beacon configuration, and positioning.

Determining the Location A combination of RF and ultrasound is used on the beacons and listeners to determine the distance to beacons. Beacons concurrently send information about the space over RF, together with an ultrasonic pulse, thus enabling the listeners to use the time difference between the arrival of both signals to determine the distance to the beacon. These calculation takes advantage of the fact that the speed of sound (ultrasound) in the air is much smaller than the speed of light (RF) in the air. Since a listener will most likely hear several beacons at the same time, it uses either the MinMode or MinMean algorithms, as described in section 2.1.4, to select the nearest beacon - both algorithms yielded similar results.

Reducing interference Interference is to be expected, since every beacon is independent from each other. Moreover there is no explicit scheduling or coordination between the transmissions of different beacons in close proximity. In order to maintain simplicity and reduce overall energy consumption, the problem of collisions was handled by having the beacons transmit their messages in a randomly uniform distribution within an interval [150, 350]ms - a smaller frequency will increase the amount of time between location inferences and a bigger one will increase the probability of collisions.

Beacon positioning and configuration The detection of the nearest node may not suffice to correctly identify the room where one is located. To address this problem, whenever one wishes to demarcate a physical or virtual boundary corresponding to a different space, it must be placed at a fixed distance away from the boundary demarcating the intersection between two spaces. We can see an example of the beacons placement in figure 2.5

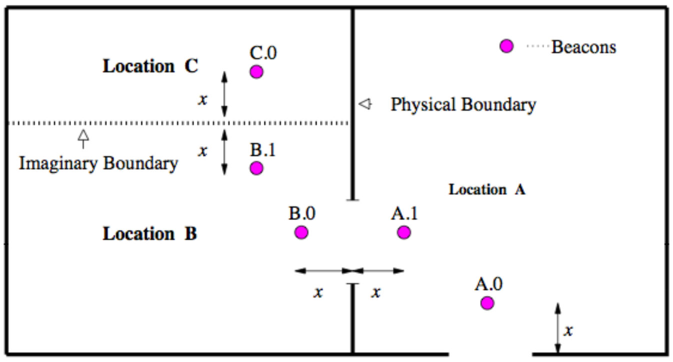


Figure 2.5: Illustration of a correct beacon positioning according to Cricket (from [PCB00]).

The stated solution managed to provide a location system which manages to locate a static and mobile user with a precision of up to one feet. Furthermore, it also provides an API enabling the use of applications that use virtual spaces to communicate with each other, enabling the use of services on the work place. Also, the ability to provide floorplans and effectively locate users through the use of the API was achieved.

Strong aspects

Cricket demonstrated that space location instead of absolute locations can achieve a higher degree of accuracy in determining the room, or section of a room, a person is currently in. Furthermore, Cricket also manages to demonstrate that careful planning of the positioning of beacons can influence results and improve location estimation.

Symbolic (space) location proves to be a solution that is sometimes better then absolute, not just because one can achieve a higher rate of success in identifying the location, but because the name of a location can have more meaning to a end-user. Our system provides a method of obtaining Symbolic location using the existing infrastructure by using fingerprinting techniques. However, unlike cricket, our solution does not require any additional hardware for location estimation.

2.2.3 RADAR: In-Building RF-based User Location and Tracking System

RADAR [BP00] is a system developed to locate and track a user inside a building through the use of commonly available radio-frequency transmitters. By providing a floor map, based on cartesian coordinates, as well as an initial database of gathered data, a wifi base station can infer a user location with a median accuracy in the range of 2 to 3 meters.

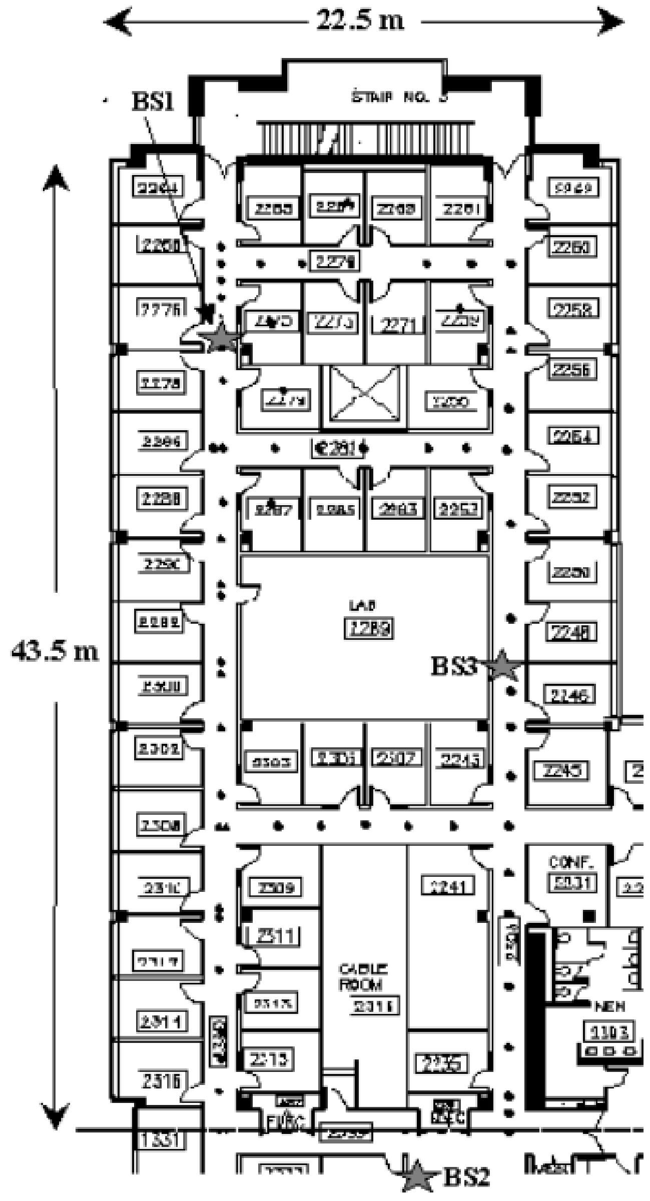


Figure 2.6: Floor map where the RADAR system was tested (from [BP00]).

Architecture/Implementation

To use this system, it is necessary to measure signal strengths in several distinct locations and orientations. For example, 2.6 is the map of the floor where the system was tested. The dots represent the places where the data was collected, and the stars represent the three base stations.

To locate a user, a simple triangulation method is used, where given a set of signal strength measurements at each of the base stations, the users location is determined by finding the best match in the database, as explained in 2.1.2.

To track mobile users, new Signal Strength data was gathered while randomly walking through the building floor at a uniform pace. Then, a sliding window containing the last 10 measurements was used to track a user. The system uses the values contained in the sliding window to calculate the mean signal strength, and then uses the value with the basic signal strength fingerprinting method in 2.1.2.

Strong aspects

This system was the first to prove that it was possible to use available base stations to provide a location system to an end-user. With a simple triangulation approach based on the nearest nodes, and calculating the euclidean distance, one can achieve location estimations ranging from 2 to 3 meters.

The RADAR system also tested the use of Radio Propagation Models, as explained in 2.1.3. Furthermore, the study also provides important insight into building a tracking system based on Radio Propagation Models. A mathematical model of indoor signal propagation was used, in order to generate a set of theoretically-computed signal strength data, allowing for a precision of 4.5 meters. Though not as good as the precision obtained by the previous method, this is cost effective in the sense that there is no need to obtain a big amount of data before being able to use the system. Not only that, but a base station can be moved from one place to another, with no need to regather all the new data.

The UCLS location algorithm is based on fingerprinting proposed in RADAR [BP00]. Our system extends this approach by scanning not just the available Wi-Fi base stations, but every available technology.

2.2.4 Place Lab: Device Positioning Using Radio Beacons in the Wild

Place Lab [LCC+05] is a location system that focuses on using the available resources throughout our surroundings, allowing users to discover and communicate their positions. By listening for surrounding radio beacons, Place Lab can determine a users location in latitude and longitude coordinates, if it knows some previous information of the scanned information.

Place Lab's main goal is to achieve maximum coverage in our daily lives, providing a service that is always working, even when indoors. Furthermore, it is also their aim to provide a service that is affordable to the average user, which is achieved by using regular devices already present on a person's daily travels: Laptops, PDAs, cellphones.

Architecture/Implementation

The system's architecture is mainly divided in three components: The Mapper, the Tracker and the Spotter(s) (figure 2.7).

The mapper is used to provide the location of known beacons to other components, always containing at least a latitude and longitude. This information data can be obtained either directly from a mapping database or from a previously cached portion of a database.

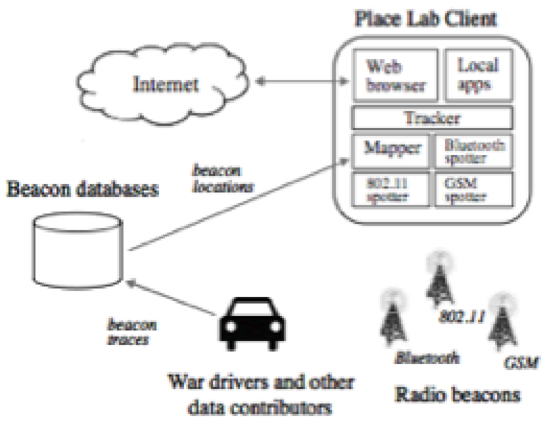


Figure 2.7: Place Lab's Architecture (from [LCC+05]).

There is one spotter for each radio protocol the client can handle, and each one gathers information about the surrounding beacons. The IDs and signal strengths of the located beacons are then transmitted to the other components of the system (mainly the Tracker).

Finally, the Tracker is used to gather the information provided by the available spotters and, based on the available information on the mapper, estimates the user's current position. The tracker can also encapsulate additional methods to provide a higher accuracy, enabling access to external data like road maps, or even algorithms that use propagation models, allowing it to have an understanding of how various types of radio signals propagate, and how that data relates to distance, the physical environment and the current location.

Place Lab's functionality depends on the availability of public databases like wigle.net, containing more then 2 million 802.11 access points. In order to provide a big enough cover, war-driving data was used to build the clients databases (see 2.1.5). Through the use of own-made and publicly available war-driving databases, the Mapper component can have access to the gathered information.

Strong aspects

Place Lab brings a solution to privacy requirements, since users locate themselves and are totally independent from an external source to help with location. Furthermore and probably more interesting, it also provides a solution for the construction of very large databases based on Fingerprinting, by taking advantage of publicly available databases with the needed information.

The architecture of the UCLS has some similarities with this system, using the concept of providing an API for the end-user, as well as using several modules that scan for available information. The use of external databases to gather higher amounts of updated information is also invaluable, as is the wardriving method to collect data automatically for absolute location estimation.

2.2.5 ARIADNE: A Dynamic Indoor Signal Map Construction and Localization System

ARIADNE [JBPA06] is an indoor Location system, which only needs to make one signal strength measurement per available Access Point. By feeding the system with a floor map (e.g., a CAD or floor plan image file), together with a couple of measurements for each Base Station, ARIADNE builds a signal strength map, which can then be used to estimate a user location based on the information provided by it. This algorithm uses the same principle of fingerprint mapping, but without the time costly process of retrieving measures all over the building.

Architecture/Implementation

ARIADNE is composed of two Modules: the Map Generation Module, which builds the Signal Strength Map, and the Search Module, which uses a cluster-based search algorithm to locate users. The map Generation module extracts structural parameters from the floor plan picture using basic image processing techniques. Basically, what is done in this phase is the construction of a map representing walls present on the

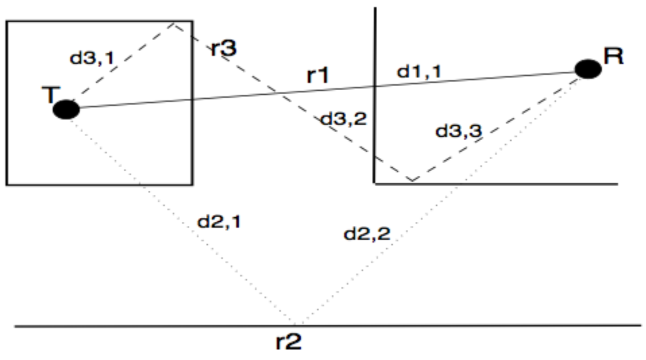


Figure 2.8: Radio Propagation with ray tracing (from [JBPA06]).

current floor. This will lead to the second map building phase, which uses a Radio Propagation Model to insert into the map the necessary data to compute the strength emitted by an antenna, taking into consideration not only the usual propagation models, but also the exact walls the radio signal will traverse. Not only direct paths are considered, but up to two reflections are also accounted for the map built. This means each Base Point will be present on the map, together with a combination of the trace of several possible radio propagations. Figure 2.8 gives an example of how the system calculates the radio propagations. Being T the transmitter, and R the Receiver, each ray emitted from the transmitter, either by a direct path to the receiver or by reflection, can be exactly determined, by using a ray imaging technique [Val94, FPY96].

After having built the map, the system is ready to use, and locates users by using its Search Module, which is composed of several phases. First, the user scans his surrounding area, and obtains the visible APs, together with their Signal Strength. Then, the system searches its map, and obtains not one but several likely candidates for the location match based on a predetermined mean square error threshold. The system then chooses a random group of those candidates, and uses a K-clustering method [Mac67] to obtain the most likely location of the user.

Strong aspects

This system not only presents an alternative and faster way to map a whole building than the ones presented with usual fingerprinting systems, but it also presents an algorithm that effectively detects a user not only at a certain coordinate indoors, but also through a multi-story building.

It is valuable insight to understand one can build information maps to provide to mobile devices. The fact that the process is computationally heavy suggested that external help could be needed for the construction of more complex maps. The UCLS builds more complex data structures for mapping by using the same principle (relying on computational rich computers), allowing for faster scans of the available information.

2.2.6 Always Best Located, a Pervasive Positioning System

Always Best Located [RD07] is an indoor-outdoor pervasive positioning system. Using different existing technologies, this system aims to provide a combination of good coverage and accuracy by taking advantage of the best characteristics of each, using whichever gives the best results at the user's current position.

Architecture/Implementation

The Always Best Located system is built using a combination of GPS and WLAN. GPS is a proved system for outdoors location, providing an excellent accuracy and availability for outdoor environments.

To provide indoor tracking, location data was gathered by means of a war-driving solution around the tested area. Then, a location map was built using the Signal Strength Fingerprints obtained by the war-driving solution. Then, a self-mapping algorithm is continuously used while the user is indoors, so he can not only locate himself, but also add new AP locations to the database easily, with no need for further configuration.

After the required training phase, the system is ready to use, using GPS when possible to locate the user, and switching to WLAN whenever GPS connectivity is lost. This yielded results of 3.7 meter accuracy for GPS and between 8.5 and 14.1 meters for WLAN positioning. Furthermore, an additional method was implemented to locate places indoors, instead of estimating global positions, in order to refine indoors accuracy. This is only done with a training phase, where a user uses the traditional fingerprinting method and inserts the room where each measurement is taken. A probabilistic approach was used to locate users when under these conditions (the APs have associated places), successfully identifying the correct room 88% of the time.

Strong aspects

This study effectively merges two distinct technologies in order to locate a user, providing the user with a fallback when a technology fails (GPS indoors). Furthermore, it also effectively merges absolute positioning with symbolic (place) positioning, by providing place information to a user whenever absolute positioning is unavailable.

Our system merges every available technology, although it does go one step further by providing every location type possible, since both an absolute and symbolic location may be desired. By sharing information with close users, our system has the potential to improve location results.

2.2.7 Calibree: Calibration-free Localization using Relative Distance Estimations

Calibree [VPKE08] is a user location system that uses no previous off-line calibration or map construction in order to locate a user. This system uses an algorithm that computes the relative distance between pairs of mobile phones using the application, based on the signatures of their surrounding GSM radio environment. Then, the system combines these distances with the known location of GPS-equipped phones that belong to the created network, in order to estimate absolute locations (Figure 2.9). If no known locations are known (e.g., no phones with available GPS coordinates belong to the system), relative positions are still available, having been previously computed as part of the algorithm.

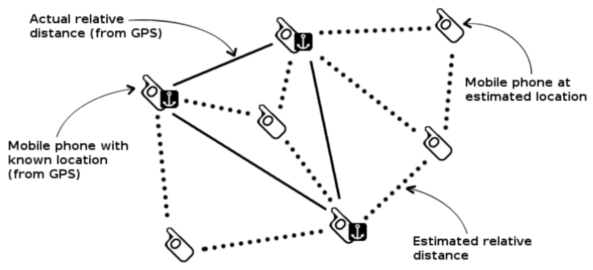

Actual relative distance (from GPS)
Mobile phone at estimated location
Mobile phone with known location (from GPS)
Estimated relative distance

Figure 2.9: Absolute and Relative positions of mobile phones using Calibree (from [VPKE08]).

Architecture/Implementation

As stated above, Calibree has two stages, and the first one consists in estimating the relative distance between each pair of phones. This is achieved by comparing the GSM signatures that each phone detects. A Spearman coefficient [PFTV92] between rankings of common cell towers by signal strengths is used to compute the relative distance between each pair of mobile phones.

For the second phase, estimating absolute location, Calibree uses a self-mapping graph algorithm (see section 2.1.6), where the phones with available GPS coordinates are placed as a Seed, and the edges weight is the estimated relative distance between the phones. After the graph is built, a user will be able to estimate his position simply based on his location in the graph.

Strong aspects

This system brings a new approach to studied systems, where no calibration is needed to locate a user, as long as there are other users using the same system. The fact that a user can share information with neighbors brings a new perspective to location systems, mainly that users can cooperate amongst themselves to help locate each other.

Although the information sharing mechanism of this proposed study seemed too heavy to implement on an actual system, the fact remains that information sharing may help not only on location estimation, but also on providing information to users who wouldn't be able to otherwise.

The Unified Cooperative Location System uses a simplified communication protocol, by simply identifying close users via bluetooth, and sharing information directly between them. Instead of using a complex graph approach, a simple, direct contact is made between each two users, allowing for a cost-effective solution for communication.

An interesting question that is worth investigating in the future is the possibility of adopting some of the ideas proposed in Calibree in UCLS.

2.2.8 The Skyloc Floor Localization System

Skyloc [VLHdL07] is a GSM-based floor location system, which determines the floor a user is located in a specific place (building). By listening to the signal strength of the available GSM towers, the system can correctly identify the floor in up to 73% of the cases, and within 2 floors accuracy in 97% of the cases. This system also introduces a new algorithm to improve the accuracy of systems based on signal strength fingerprinting matching, called feature selection.

Architecture/Implementation

This system follows the usual implementation of data retrieval in an initial training phase. Signal strength fingerprinting (see 2.1.2) is obtained in several locations. Every time a recording is obtained, the program prompts the user to input the current floor.

To avoid the limitation of current mobile phones, which only provide information about the cell they are currently connected with, the program was built for a specific phone, that can retrieve information about every cell the phone is able to detect.

After the collecting phase, a user can use the system to estimate his location (the floor he is located at), by matching the current measurement with the set of measurements collected previously.

To improve accuracy, the use of feature selection techniques is introduced for the fingerprint matching process [VLHdL07]. SkyLoc's feature selection algorithm produces a ranking of features for each floor, where each feature on a ranking is assigned a weight based on its accuracy at identifying the floor correctly.

Additionally, a second algorithm was used to further improve accuracy, based on a sliding window. A fixed-size sliding window was defined, where the last N measurements are taken into account. By counting every floor achieved for each measurement, it is determined that the one that was counted more often is more likely to be the current floor.

The implementation of both algorithms nearly doubled the achieved accuracy, compared to a simple Euclidean Distance measurement between the obtained data and all the stored information.

Strong aspects

One can learn from this study that GSM can be used for location systems fairly well if a user could somehow obtain the data from surrounding cells (against only being able to detect the one his mobile phone is associated with). This can be achieved by using our system's information exchange mechanism, which allows the gather of information from more then one GSM tower when another user of the system is nearby.

UCLS uses the idea of Skyloc by relying on information from multiple devices, instead of relying on special hardware.

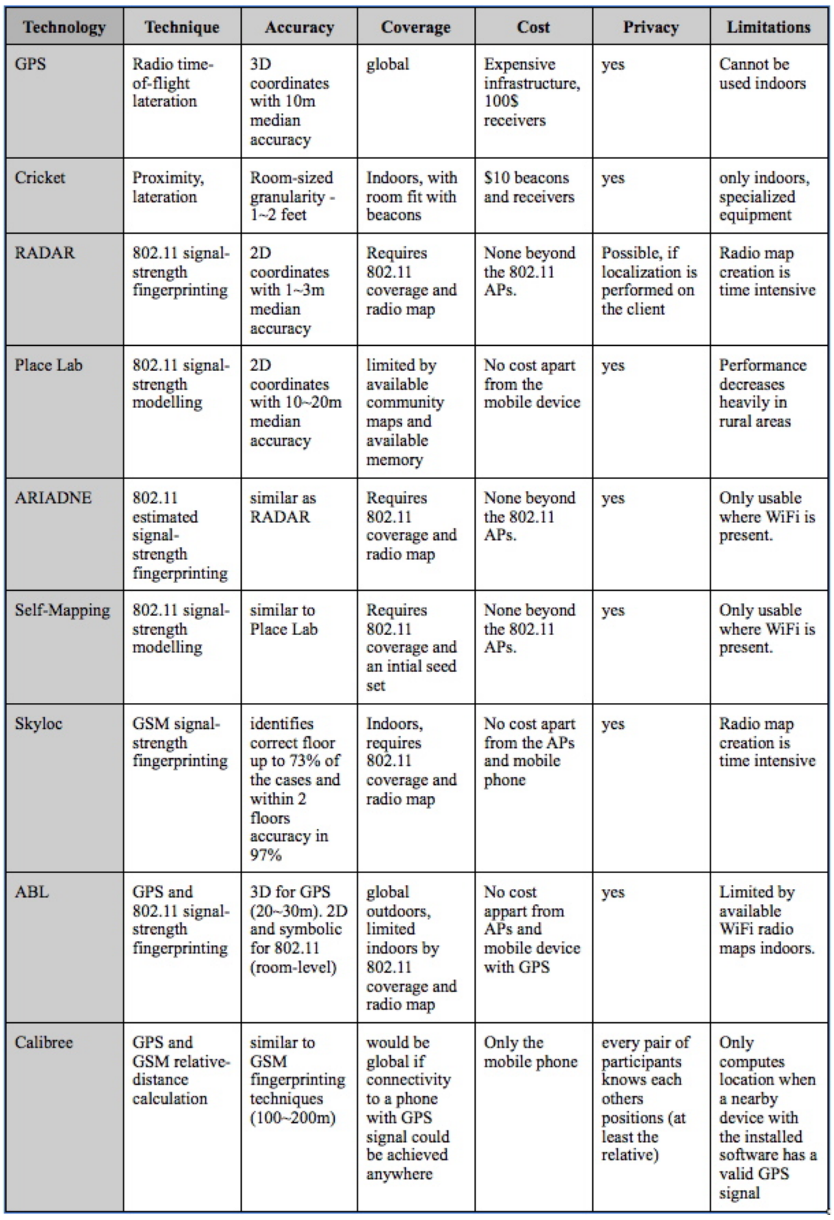
2.3 Summary

The following table summarizes the main characteristics of the different systems presented (figure 2.10).

Besides its characteristics, the table presents the systems limitations. A quick analysis of the table reveals that most of the systems use already existing infrastructures, which is also our approach to the system. The fact that these technologies are widely available, and can be explored at no extra cost to the users is a very appealing fact.

Our work combines several of the solutions of the various systems to obtain location estimates, as discussed in the previous sections. While most works only focused on one or two technologies at a time, our solution tries to use all available technologies. Not only that, but by implementing several algorithms, it is possible to achieve adequate accuracy with possibly less energy consumption.

As discussed in section 1.3, we allow information sharing amongst users to improve accuracy, by applying Skyloc's ideas of using multiple readings to obtain better



Technology	Technique	Accuracy	Coverage	Cost	Privacy	Limitations
GPS	Radio time-of-flight lateration	3D coordinates with 10m median accuracy	global	Expensive infrastructure, 100$ receivers	yes	Cannot be used indoors
Cricket	Proximity, lateration	Room-sized granularity - 1~2 feet	Indoors, with room fit with beacons	$10 beacons and receivers	yes	only indoors, specialized equipment
RADAR	802.11 signal-strength fingerprinting	2D coordinates with 1~3m median accuracy	Requires 802.11 coverage and radio map	None beyond the 802.11 APs.	Possible, if localization is performed on the client	Radio map creation is time intensive
Place Lab	802.11 signal-strength modelling	2D coordinates with 10~20m median accuracy	limited by available community maps and available memory	No cost apart from the mobile device	yes	Performance decreases heavily in rural areas
ARIADNE	802.11 estimated signal-strength fingerprinting	similar as RADAR	Requires 802.11 coverage and radio map	None beyond the 802.11 APs.	yes	Only usable where WiFi is present.
Self-Mapping	802.11 signal-strength modelling	similar to Place Lab	Requires 802.11 coverage and an intial seed set	None beyond the 802.11 APs.	yes	Only usable where WiFi is present.
Skyloc	GSM signal-strength fingerprinting	identifies correct floor up to 73% of the cases and within 2 floors accuracy in 97%	Indoors, requires 802.11 coverage and radio map	No cost apart from the APs and mobile phone	yes	Radio map creation is time intensive
ABL	GPS and 802.11 signal-strength fingerprinting	3D for GPS (20~30m). 2D and symbolic for 802.11 (room-level)	global outdoors, limited indoors by 802.11 coverage and radio map	No cost appart from APs and mobile device with GPS	yes	Limited by available WiFi radio maps indoors.
Calibree	GPS and GSM relative-distance calculation	similar to GSM fingerprinting techniques (100~200m)	would be global if connectivity to a phone with GPS signal could be achieved anywhere	Only the mobile phone	every pair of participants knows each others positions (at least the relative)	Only computes location when a nearby device with the installed software has a valid GPS signal

Figure 2.10: Comparison between the different technologies.

estimates and extending the information sharing mechanisms presented by Calibree.

3 System Architecture

The Unified Cooperative Location System (UCLS) is a modular location system that can use any wireless technology available in a mobile device to estimate both symbolic and absolute location. UCLS includes an information sharing mechanism among mobile devices, allowing mobile devices to share location-related information. This mechanism allows a mobile device to obtain information from hardware it does not contain (e.g. GPS), or additional measurements for hardware it contains (e.g. GSM signal strength). This approach helps in improving location information accuracy.

The system promotes user privacy by establishing the location estimation on the device, without querying external entities every time a location estimation request is made. Since location sharing is made among nearby devices (usually up to 10 meters on mobile devices), and information exchanged is anonymous, location privacy is not significantly impacted by this mechanism.

The system has a very low entry cost, as it requires no additional hardware. A user simply needs to install the software on his mobile device, and start using it. UCLS was designed with the goal of being extensible. Thus, its modular approach allows to easily add support for new technologies and algorithms. Further control is achieved with this implementation, by deciding which technologies the user wishes to use at a time, controlling not only the energy consumption, as well as the required estimation precision. Algorithms can easily be added to the system, as well as additional location information storage and retrieval methods.

Since the construction of fingerprinting databases is somewhat time consuming, the Location System provides methods to share information with external databases in bulk. Gathered readings can be sent to the database, as well as retrieved, at any point in time the user wishes to do any of those.

3.1 System Overview

The Unified Cooperative Location System (UCLS) has been designed to provide the programmer with a simple yet powerful location API. To achieve this goal, the UCLS Architecture consists of five modules and a collection of Sniffers, as depicted in figure 3.1. The Sniffers are simple background services, whose purpose is to obtain information that can be used for location (e.g. GPS information and wireless communications signal strength). This information is reported to the Communications Module. The Communications Module is responsible for aggregating the information from the sniffers and delivering it to the other modules. This module also provides basic bluetooth communication primitives.

The Information Exchange Module uses the bluetooth service search and communications primitives provided by the Communications Module, to find nearby devices wiling to share location information. This allows for an easy method to share not only gathered location information, as well as more complex information such as location estimations, or relative distance estimations between the devices.

The Database Module provides primitives to store and read location information. It also allows the mobile device to connect to external servers or services. This is used mainly to share location information. In this case, the information stored locally acts as a cache of the more complete information stored in the server. The server can also be used to provide efficient location information data structures.

The core of the system is represented by the Location Module, whose function is to estimate the location of the device. To this end, this module controls the other modules of the system, selecting which information to store or retrieve from the Database Module, to use the available location algorithms. This selection is based on the current configurations of the system.

Finally, the Location API Module allows the system to be used by a developer, integrating it easily on his developed location-aware application. The remainder of this chapter introduces each module of the system and how they relate to each other.

3.2 Location API Module

The Location API Module provides an API (listing 3.1) for a developer to use a location system on his own location-aware application. Operations can be divided in three

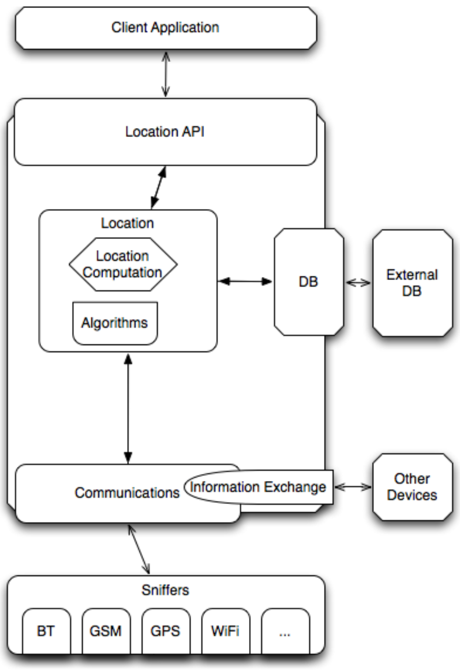


Figure 3.1: Unified Cooperative Location System Architecture

types: configuration, information acquisition and management and location estimation.

Listing 3.1: Location API Module

```
public interface LocationAPI{
    //configuration
    public boolean activateTechnology(int TechId);
    public void deactivateTechnology(int TechId);
    public boolean activateAlgorithm(int AlgorithmId);
    public void deactivateAlgorithm();
    public boolean activateInfoExchange();
    public void deactivateInfoExchange();
    public void setExternalServer(String serverAddress);
    //information acquisition and management
    public boolean activateLocationRecording(int seconds);
    public void deactivateLocationRecording();
    public void sendReadingsToServer();
    public void gatherReadingsFromServer();
    public RawData recordLocation(Location loc);
    //location estimation
    public RawData getLocalInformation();
    public LocationInformation locateDevice();
}
```

The API provides a set of simple configuration options to control what location mechanisms the system should use. An activation/deactivation call is available for every technology (GPS, GSM, Wi-Fi). If the associated hardware is locally available, the correspondent sniffer is activated. Otherwise, this technology can only be used if the information exchange mechanism is active and a nearby device willing to share this type of information is available.

To provide further configuration, a call is available to activate the system's available algorithms. These algorithms are used for estimating the device location based on the currently available location information.

In addition to the aforementioned sharing mechanisms, and similar to the available calls to activate and deactivate certain wireless technologies, an activation and deactivation call is available for information sharing between similar devices. This should allow a device to obtain location information from another nearby device, including GPS and fingerprints information. Whenever this option is active, information gathering between nearby devices will occur for both mapping mechanisms (manual and automatic), as well as to estimate the current location if the used algorithm supports this feature.

To provide the user with the ability to map new locations using fingerprinting approach, the API provides two simple methods for fingerprint acquisition: an automatic and a manual one. The automatic acquisition mechanism has an activation and a deactivation call. The activation call is initialized with a timer in seconds, corresponding to the interval between reads. While active, the system will gather the surrounding wireless signal fingerprints whenever a GPS lock is available, and store that information in the DB Module.

The manual gathering mechanism is used mainly for symbolic or relative mapping construction, although if no local information is provided but a GPS is still available, the system will still record the observed data, since it will be interpreted as an absolute location reading. The API thus provides just a simple, manual call, which receives the location information (symbolic and/or relative), and gathers and stores one fingerprint.

The Location API Module provides methods to upload or download location-related information to/from external devices or servers. For this purpose, two methods are available to either share the last gathered information with an external device/server or request new location information from an external device/server.

A call is available to request location information. This call only returns information regarding the current estimated location of the mobile device and which technologies and algorithm were used. The location estimation accuracy will depend on the available technologies, location information databases and algorithms. An additional call is also available to simply scan a fingerprint and return it to the user.

3.3 Database Module

The Database Module (listing 3.2) is responsible for providing mechanisms for managing and storing location information. This location information can include Wireless fingerprints, antenna locations and mapping information. In the current prototype we are only using a solution based on fingerprints.

Listing 3.2: Database Module

```java
public interface DBController{
    //configuration
    public void setServer(String serverAddress);
    //information acquisition and management
    public void storeLocationInformation(LocationInformation reading);
    public void sendData();
    public boolean gatherServerReadings();
    //location estimation
    public Iterator<LocationInformation> getReadings(int TechId,
        String id);
    public Iterator<LocationInformation> getReadings();
}
```

The Database Module can be configured to connect to an external server, providing extended functionalities to the information storage and retrieval mechanisms. By connecting to an external server, the location information previously recorded on the device can be shared with multiple users of the external databases. The device can also request the shared location information from the database server. This simplifies the process of obtaining the needed information for location estimation.

The fact that the device connects with an external server also gives the device two additional functionalities. First, readings or queries of certain data can be requested by the device to an external database/entity on a per-request basis. Thus, instead of downloading the full database of gathered information, the device can make specific information requests, either to acquire information of a given area in case the device has low storage capacities, to obtain information online from the database or even from location services like the ones provided by Skyhook Wireless [Sky09].

The second added functionality is the ability to use external resources to build and prepare efficient information for the device: instead of providing a list of location information, more complex data structures can be created for the device, to simplify location estimation. This will allow the device to save time and energy, whenever it is necessary to search the stored information.

To gather the stored information, this Module supplies methods to access location information, providing an Iterator over the available information. The Iterator can either be over all of the available data, or a subset of the stored location information data. The iterator implementation will depend on the available Data Structures. Currently, an iterator is returned when a technology identifier and the identifier of the AP/Cell is shared. Then, every stored Location Information that contains the specified AP/-Cell is gathered and an Iterator to that collection is returned. This is only usable if a data structure is present to quickly return the needed results. Otherwise, the simpler method is used, which provides an Iterator for all the available location information in the Database.

3.3.1 Database server

A simple database server, accessed through web services interface (listing 3.3), was created, allowing users to store their location information. This service allows the insertion and retrieval of location data. Insertions are made by simply sending the information to the database, together with a user identification and mobile device used to obtain the information. To retrieve location information, a call is available with several optional variables. These variables help the user to define if they want information from a specific user, mobile device or location.

Listing 3.3: Web Service Interface

```java
public interface GathererImplWS{
    public LinkedList<LocationInfo> getReadings(String user,
        String device, LocationInfo location);
    public void addLocation(LinkedList<LocationInfo> locations,
        UserInfo user);
}
```

3.4 Communications Module

The Communications Module (listing 3.4) provides communication functionalities for accessing the sniffer and perform basic communication between mobile devices. In the context of communicating with the sniffer, this module provides a function to open/-close connections and to send/receive typed data. Since each technology requires different levels of control as well as different kinds of information gathering, each technology module must implement the getLocationInfo functionality call, which will return a gathered LocationInfo for that specific technology.

The Communications Module also implements a simple bluetooth services search call, used to find devices providing a requested service. This is used mainly for sharing information among mobile devices.

Listing 3.4: Communications Module

```
public interface ModuleCommunicator{
    //configuration
    //information acquisition and management
    public boolean openCommunications(int TechnologyId);
    public void closeCommunications(int TechnologyId);
    public int getInt(int TechnologyId);
    public void sendInt(int TechnologyId, int intToSend);
    public String getString(int TechnologyId);
    public void sendString(int TechnologyId, String stringToSend);
    //location estimation
    public LocationInformation getLocationInfo(int technologyId);
    public List<DeviceInformation> searchServices(String service);
}
```

3.4.1 Sniffers Communication protocol

To obtain location-related information, the Communications Module must contact the device's available Sniffers. Each technology provides different types of information (e.g.: GPS provides longitude, latitude and altitude information, while Wi-Fi provides the SSID and Signal Strength information), as well as different methods of operation (e.g.: GSM location and signal information is always and instantly available to the device, whereas a GPS device needs an activation phase, followed by a satellite lock in order to start providing location information).

Due to these facts, there needs to be one Sniffer per wireless technology, since the information returned and the necessary control mechanisms may differ from Sniffer to Sniffer. A standard protocol is used for every one, including to open and close communications, and to send and receive information. The specifics of the communication protocol, including if an activation or disable call are necessary, are left for each Sniffer and Communication Module implementation.

3.4.2 Bluetooth Communications

The Communications Module also includes a function to search for a specific service on the nearby bluetooth devices. Once the search is complete, the call returns the information needed to access the detected services. Additionally, methods for basic communication are available to the system, mainly the connection and disconnection methods, and the sending and retrieval of information.

3.5 Information Exchange Module

The Information Exchange Module (listing 3.5) is used by the Location Module to obtain location information from nearby devices. This module is only active when the mechanism is enabled. To share information, this module uses the available methods for service search and information exchange present in the Communications Module.

Listing 3.5: Information Exchange Module

```java
public interface GroupShare{
    public void searchDevices();
    public List<LocationInfo> requestData(List<int> technologies);
    public LocationInfo requestLocation();
    public List<LocationInfo> requestData(List<int> technologies,
        int timestamp);
}
```

When active, the device will share wireless information with nearby devices that request that information. To this end, the Communications Module retrieves the needed information, and supplies it to the device that requested the information. In the future, it would be interesting to add an option to select with which devices to communicate with.

Optionally, when a request for data sharing is made, a timestamp can also be shared, requesting the device to make the readings at a specified time. This allows the devices to perform a near-synchronous reading of all the available wireless information.

This last option is important when sharing information while moving (e.g.: during war driving). In this case, it is important that the information gathered is obtained as much as possible at the same time. The difference between the real location of several devices while in movement may vary greatly with just a couple of seconds difference. To be able to obtain synchronized information, the devices need to have some sort of time synchronization capability. Since we are using Mobile phones, we assume the devices are synchronized with the time of their respective Mobile Operators. Other more complex time synchronization mechanisms could be used in the future.

To request information from nearby devices, this Module uses the available bluetooth service discovery in the Communications Module, to locate nearby devices available for sharing. Then, the device simply needs to request the detected devices for their information. After receiving all the available data, including its own, it can return the received data to the Location module.

Since the communication between the devices allows for a simple information sharing, in the communication protocol it is possible to specify which information to obtain, including a location estimation.

3.6 Location Module

The Location Module (listing 3.6) is the core of the system. It is used by the Location API Module to provide the service to the Programmer. This module controls the System's configurations and functionality. It uses the Database Module to store and retrieve information, and uses the Communications Module to gather and share real-time location information.

Listing 3.6: Location Module

```
public interface ULSController{
    //configuration
    public boolean activateTec(int TechId);
    public void deactivateTech(int TechId);
    public boolean activateAlgorithm(int AlgorithmId);
    public void deactivateAlgorithm();
    public boolean activateInfoExchange();
    public void deactivateInfoExchange();
    public void setExternalServer(String serverAddress);
    //information acquisition and management
    public boolean activateLocationRecording();
    public void deactivateLocationRecording();
    public RawData recordLocation(Location loc);
    public void storeReading(Reading reading);
    public Iterator<Reading> getReadings(Reading reading);
    public void sendLocationInfoToServer();
    public void gatherLocationInfoFromServer();
    public void gatherStructuresFromServer();
    //location estimation
    public RawData getLocalInformation();
    public LocationInformation locateDevice();
}
```

This Module's methods are basically similar to the ones provided in the Location API, implementing the main functionalities, controlling the system through the selected preferences, available algorithms and communicating with the Modules of the UCLS.

3.6.1 Location Algorithms

Location algorithms can be added to the system. These algorithms can access the other modules in the system to obtain information to estimate location. In the current prototype we have implemented two algorithms that will be detailed in section 4.5.1.

First, a fingerprinting-based location estimation [BP00] is available, which uses the Communications Module to gather location information readings, and stores them on the Database Module. To locate the device, the Location Module first acquires local information with the Communications Module's activated technologies. Then, it searches the stored readings in the Database Module with the use of the Iterator, in order to find the stored reading that most closely resembles the one acquired via the Communications Module, thus inferring the current position.

The second available algorithm is used with the Information Exchange mechanism. When more then one GPS reading is gathered, the average of the reads is calculated.

4 Implementation

This chapter presents the implementation of the UCLS. After presenting the target environment, the implementation of the various modules is described. The description is not intended to be complete but only to describe the non-trivial decisions and solutions implemented.

4.1 Target Environment

In order to provide a system supported by most current mobile devices, the Java 2 Micro Edition (J2ME) programming language was used for most of the implementation. Some portions of the system, however, had to be implemented in a native language, to access functionalities that are not available to the Java system. In this case, since mobile phones using the Symbian OS were used, the small amount of extra code was implemented in Python for S60 (PyS60). This language provides the necessary support to gather the necessary location information (e.g. signal strength) required by our fingerprint-based algorithm, which the java environment does not provide.

4.2 Database Module

The Database Module provides several storage and retrieval mechanisms for location data. Additionally, it provides functionalities which allow it to communicate with an external server. The external server can be used to store and retrieve data.

For storing data in the mobile devices, it was decided to use the native J2ME storing mechanisms, based on Record Stores. A Record Store is basically an array of byte arrays, allowing direct access to a specific position by providing the position (an integer) in the Array.

4.2.1 Storing gathered Readings

In the current prototype, the Database Module manages location information fingerprinting structures as Grouped Readings. A Grouped Reading consists on the collection of all the available wireless location information fingerprints the device is able to gather. Currently, a Grouped Reading can contain GSM, Wi-Fi and GPS location information, as well as an additional Symbolic or Relative location information and a reading time.

Each type of location information contained in a Grouped Reading is represented by a specific class object in the Database Module, to provide a better modularity and extensibility. Each of these objects contain methods to populate its specific information from the sniffer, as well as two additional encoding and decoding methods. These two additional methods are nothing more then simple methods to pass their information to a byte array, and a byte array back to its represented object, respectively. This allows a simpler storage on a device's Record Stores, as well as for communication transmissions.

Each Object contained in a Grouped Reading (GSM, Wi-Fi, GPS and Symbolic and Relative locations) is stored on a different Record Store. An additional index Record Store maintains the location information, the reading time and links to the other information (see figure 4.1). It was chosen to store each data type Object in a different Record Store to simplify implementation and indexing.

4.2.2 Retrieving stored Grouped Readings

The information retrieval process is fairly simple, returning an Iterator for the stored Readings on the device. Each subsequent request for the next reading returns a new stored location information, which includes all objects stored.

The Iterator implementation simply iterates through the index Record Store. For returning a Grouped Reading, the Iterator internally gets all the referenced information from the Record Stores.

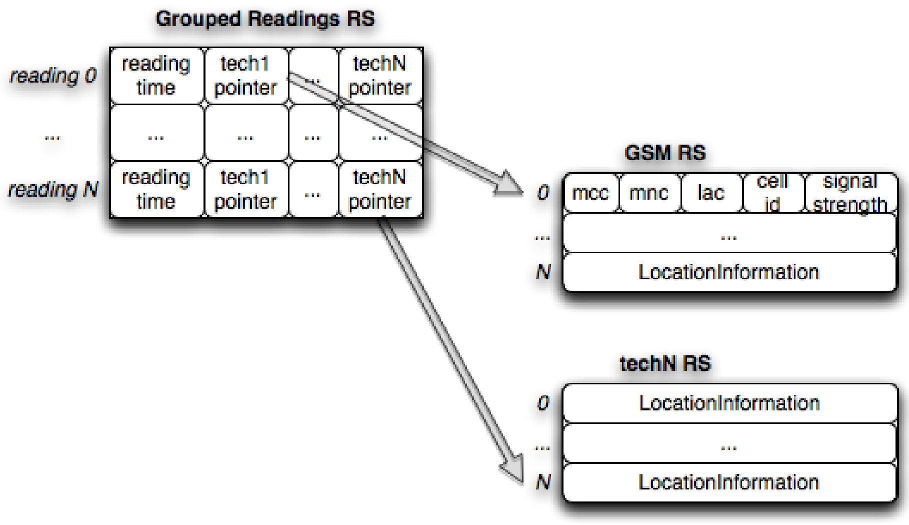


Figure 4.1: UCLS Record Stores

4.2.3 Uploading and Downloading location information to an external server

To facilitate the sharing of information among users, a system has been implemented, composed by two components, as can be seen in figure 4.2. First, a server accessed by a Web Service. This server maintains a community shared location database, used to store location information contributed by mobile devices. The second component of the system consists on an Intermediary service, used to accomplish several goals. The first goal is to serve as an intermediate between mobile devices and the Web Server. The second goal is to provide added functionality, which could be computationally intensive to the mobile device.

The Intermediary service (written in java) can be made available either through an IP or Bluetooth connection. This allows a device with no Internet connection to still be able to connect to the Server through the Intermediary. The Intermediary service can run on any system capable of running java.

The Database Module provides a simple method to set the intermediary's address. When sending new data to the server, the Database Module connects to the defined Intermediary. The Mobile Device then sends all its latest obtained information to the Intermediary, which in turn propagates this information to the Database.

The process of sending the gathered readings to the Web Server is fairly simple, sending each Grouped Reading at a time, together with a username and a device identification. The username needs to be set by the user, but the device identification (as well as the current OS version) is automatically retrieved through a J2ME operation.

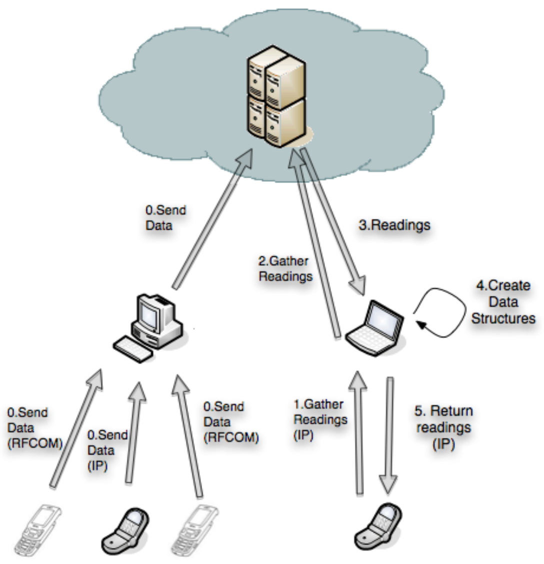


Figure 4.2: Web Service and Intermediaries communication

The retrieval process is somewhat similar to the storing procedure. The mobile device simply requests all the available data to the Intermediary, which in turn will obtain the necessary information from the server. The communication protocol is similar to the one available in the Communications Module, implementing simple functions to open and close communications, and send typed data back and forth.

Additional Improvements

For improving the storage and retrieval process, the mobile device can simply send all its current readings to the Intermediary, without any further processing on the part of the mobile device. The intermediary will then decode the byte arrays, and send them to the database correctly parsed.

Similarly, when the Intermediary retrieves data from the database, it will re-encode the data immediately to byte arrays. This data is then propagated to the mobile device, where it is simply stored into Record Stores.

Another improvement is performed at the time of retrieving location information from the Web Server back to the Mobile Device. Searching through all the available readings on the device can be both time consuming and computationally intensive, specially after gathering readings from multiple devices or locations. To address this problem, the intermediary can build indexing structures, which allow the device to search its Record Stores faster, retrieving only a subset of its complete collection.

To accomplish this, the Intermediary parses all the gathered information, and builds an hashtable for each location information type. The key of the hashtable is an identifier related with the technology (e.g.: the BSSID for Wi-Fi Access Points) and the data contained is a list, pointing to the Grouped Readings where that specific identifier appears in the Grouped Readings Array.

Retrieving information from these hashtables is easy (see figure 4.3). Given a technology identifier, the list contained in its hashtable is returned. Then, each Grouped Reading linked by that list is returned, effectively gathering all the Grouped Readings that contain information relative to the requested technology identifier.

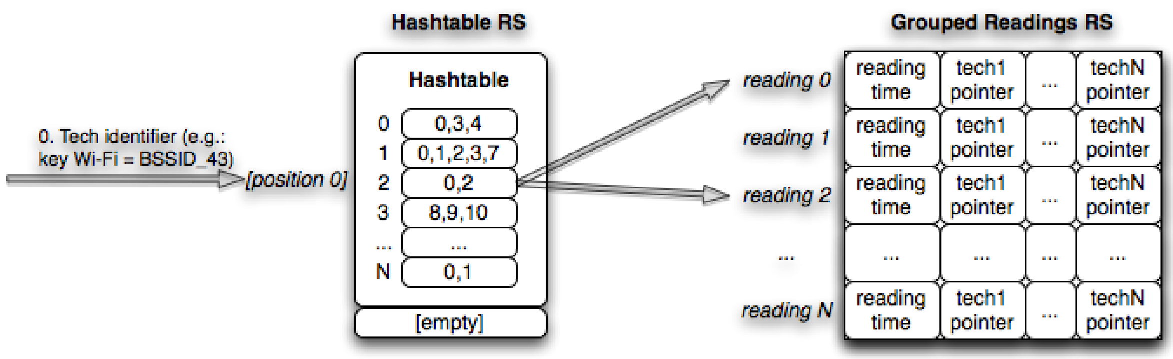


Figure 4.3: UCLS Hash Tables indexing

A second level of abstraction was added to these structures, in order to provide fewer results when using signal strength information. Instead of returning all the available information of a given technology identifier, the hashtables can be split into several smaller ones, each one containing a list of only a range of signal strengths detected for the specific identifier. Although the functionality is available and implemented, we have not yet researched what impact in location estimation this method might have.

4.2.4 Obtaining Readings for Location Estimation

The Database Module includes a method that returns an Iterator over the available Readings on the System, so that the Location Module can use these Readings to Estimate the device's current location. This method can be called with no additional information, which will return a simple Iterator over all the available Readings. It can also accept a recently gathered Grouped Reading.

When a Grouped Reading is given, the Database Module's method will return an Iterator for the subset of stored Readings to the Location Module. Specifically, it will return any Grouped Readings that include information from the technologies detected on the scanned Grouped Reading. It can only do this if it has available the indexes mentioned in section 4.2.3. Otherwise, the default Iterator will have to be used.

4.3 Communications Module

To provide a better modularity and allow for an easier implementation of new Sniffers, as well as the modification of existing ones, a basic Communication Interface is available in the Communications Module, providing several communication primitives. This module also provides the System with a set of Bluetooth Communication Primitives for exchanging typed data.

4.3.1 Sniffer Modules

Although the Java language is available for most of the existing mobile devices, it does not provide support for accessing the necessary wireless information for location estimation. This makes it necessary to implement the Sniffer Modules in another language, one that has native access to function calls of the Operating System. On the current implementation, the Sniffers were implemented using the Python programming language available for Symbian S60 platform, PyS60. This language provides a simple working environment and it has access to all the information required by our system.

As explained in section 3.4.1, for modularity and extensibility, each technology is implemented by an independent Sniffer. Each wireless technology has a Sniffer TCP Stream Server listening to a designated port. Basic generic methods are available to open and close a socket communication, and to send and receive integers and strings. These methods simplify the implementation of the communication process. Each objective relative to each technology is charged with implementing the code for implementing the communication protocol with the sniffer.

Three Sniffers were deployed , one for each available technology in our test devices: Wi-Fi, GSM and GPS. A Bluetooth Sniffer was not implemented because the Symbian OS does not implement Signal Strength measurements for this technology, although it could, nevertheless, be interesting to scan for nearby fixed location bluetooth access points.

GSM Sniffer

Available GSM information is currently very limited, with most Operating Systems only providing information regarding the tower they are currently connected to and associated signal strength. This also means that, to get any information at all, the mobile device must have an active operator card on the phone (which is usually the case) to successfully connect to a tower.

The GSM Sniffer implementation listens for incoming requests on port 10001 and returns the current information (figure 4.4). This information consists of five integers, the first four representing the cell location, and the last representing the current strength of the signal received from the tower. The 4 integers representing location include: the Mobile Country Code (MCC), which identifies the country where the tower is; the Mobile Network Code (MNC) which identifies the network within the country; the Location Area Code (LAC) which identifies an area covered by a group of radio cells; and finally the Cell ID which identifies the single tower the mobile device is connected to.

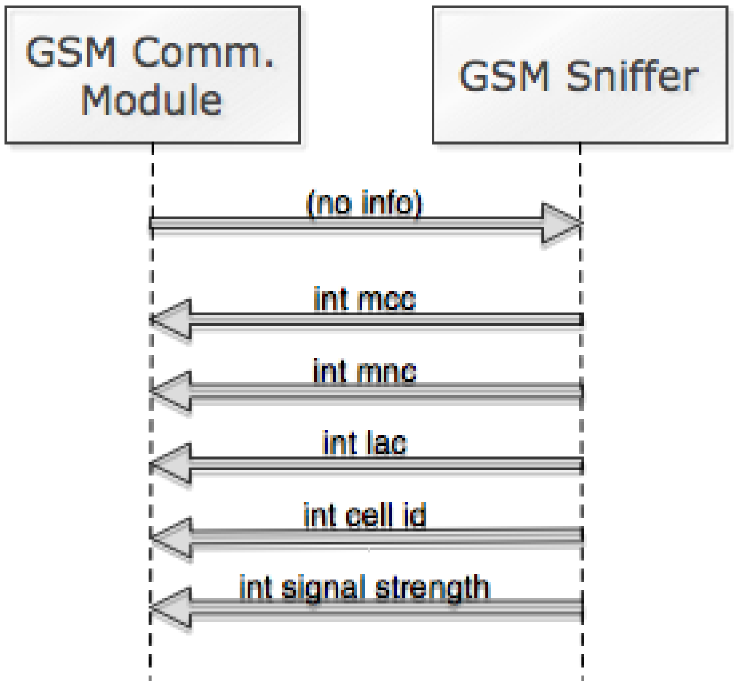


Figure 4.4: GSM Communication Protocol

Wi-Fi Sniffer

Listening on port 10002, the Wi-Fi Sniffer gathers the surrounding wireless information, and returns the gathered information back to the Communications Module (figure 4.5). In our implementation, as Symbian's python default implementation does not provide the necessary API to scan and obtain information of the surrounding Access Points, the wlantools[1] are used. This is an open-source module for PyS60 that allows to scan the environment and obtain information regarding the surrounding Wi-Fi Access Points, and associated signal strengths.

[1]http://chris.berger.cx/PyS60/PyS60

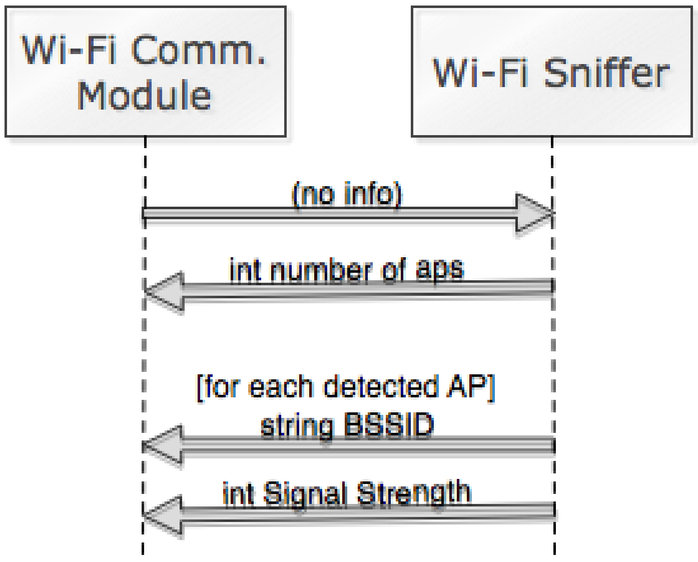


Figure 4.5: Wi-Fi Communication Protocol

The returned information is organized as follows. An initial integer reports the number of detected Access Points (APs). Then, for each of the obtained APs, the Sniffer sends the Basic Service Set Identifier (BSSID), and its received Signal Strength (Rx Level).

GPS Sniffer

The GPS Sniffer, listening on port 10010, requires some additional control compared to the other Sniffers. In order to use GPS effectively, a device needs to first acquire a lock on the available Satellites which, in some cases, can take several minutes. Whenever the GPS is deactivated and reactivated, it needs to go through the same satellite locking process. In order to provide a good, continuous, location information using Absolute Positioning, the GPS needs to be kept active. As keeping the GPS active incurs in energy usage, it should be deactivated when it is not needed. Figure 4.6 represents the communication protocol used for the GPS Sniffer.

Given the stated premises, the GPS Sniffer TCP Stream Server needs additional control other than simply waiting for connections and returning the observed values. Whenever a connection is opened, the client can send one of three commands: enable, disable and gather.

The enable command serves the purpose of initializing the GPS device. This command has a parameter (integer) stating the period between each location estimation, allowing for a finer control over when each scan should be made.

The gather command is used to request the current location information. Since GPS

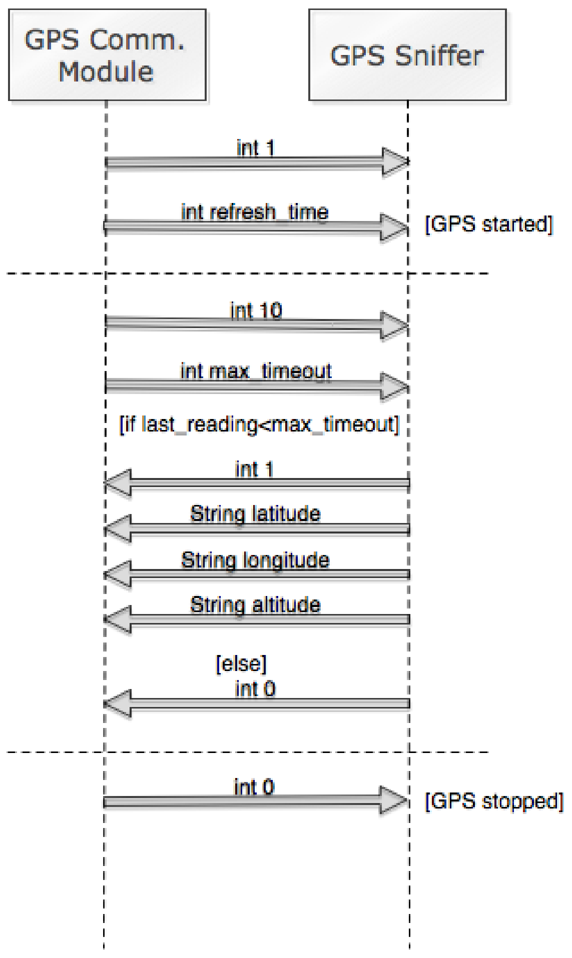


Figure 4.6: GPS Communication Protocol

locks can be constantly acquired and lost, it was found easier to control the current location estimation validity on the Sniffer. To this end, whenever a new location is acquired, the time of the location acquisition is also stored, in milliseconds. Thus, the gather command must include an integer, stating the maximum time for a reading to be considered valid. If the difference between the current time and the last reading is greater than the allowed time, the Sniffer returns zero, indicating an error. Otherwise, the Sniffer returns the integer one, indicating that one value is available, followed by the location information coded as a string for the latitude, longitude and altitude.

The disable command is used to stop the GPS from continuously gathering data, allowing the device to save energy when a GPS location detection mechanism is not required. If a reading is requested while the device has not been initialized, the connection is simply closed.

4.3.2 Bluetooth Communications

Although there is no available wireless information regarding Bluetooth Technology, it still provides the most cost-effective way of communication between nearby peers for small amounts if data. Taking advantage of this fact, the Communications Module includes support for basic bluetooth primitives.

These primitives consist in methods for searching for nearby bluetooth devices given a specific service identifier. To this end, the method searches for every available bluetooth device in the vicinity, and then searches for all the available services within each device. If the required service identifier is found, the device is stored on the Communications Module. To speed the searching process, this method can be supplied an additional integer indicating the class of devices to be scanned.

An additional method is available, returning all the previously found devices with the given service identifier successfully detected. Primitives for sending and receiving strings and integers are also available. The implementation of this methods rely on the use of JSR-82.

4.4 Information Exchange Module

The Information Exchange Module provides a mechanism for nearby devices to share location information. To this end, it uses the available bluetooth communication primitives present in the Communications Module. When this module is activated, if Bluetooth is available to the device, a bluetooth RFCOM service is initiated, waiting for location information requests from other devices. The service implements a request/reply protocol.

Currently, only two operations are available, but it is fairly simple to extend the Module to include additional operations. The first operation gathers all the available wireless information from the device and returns it to the requester. Since the Communications Module already provides all the necessary primitives to gather wireless information, it is a simple process of using it to gather the information and return it to the requester. Adding new operations should be fairly simple, since all the information gathering primitives are available, as well as the bluetooth communication primitives.

The second operation allows to obtain the location estimation of the device. When receiving this request, this module calls the location estimation primitives available in the Location Module.

In order to request information from a nearby device, the Exchange Module supplies a method to search for nearby devices. This method uses the available bluetooth services search available in the Communications Module, automatically searching and storing internally the latest scanned devices. Since a Bluetooth scan may not find every available device with one sweep, previously scanned devices using this method are kept until a communication attempt fails multiple times. It was decided to supply an explicit method for scanning instead of an automatic, because with our test hardware, a device could not search for a service at the same time it communicates with another, and communications are more frequent than searches for available devices.

After a successful search, if at least one device providing a sharing service is detected, whenever an information request to the Information Exchange Module is made, the device will communicate with all the available surrounding devices providing the service. In the future, we plan to implement a method to limit the devices to contact.

4.5 Location Module

The Location Module, at the core of the UCLS, controls the other Modules of the system. This module maintains the system's configurations, including which Sniffers are to be used (Communications Module), if the sharing service is enabled (Information Exchange Module), the external location server (DB Module) and what algorithm should be used to estimate the device's current location (Location Algorithms).

This module is used by the Location API, allowing access to the system's functionalities and location-based capabilities.

4.5.1 Fingerprinting

The current Wi-Fi and GSM location algorithm included in the Location Module uses fingerprints and it is based on the proposed in RADAR [BP00]. The algorithm works as follows: First, it starts by gathering the required available information to the device, by requesting that information to the Communications Module. If the Information Exchange Module is active, it also requests the same location information from nearby devices to that module.

Second, the algorithm will search the most close fingerprint in the stored data using a 1-kk search algorithm. To this end, the stored fingerprints are obtained from the Database Module. For each fingerprint, the algorithm computes a distance or scoring. Once the iteration is complete, the Module will return the readings's location information containing the best score.

Scoring

To obtain the most similar reading, a scoring method is performed. Several definitions for closest match have been experimented in other location systems [LdL08]. In our current prototype, scoring is done by computing the distance between two sets of fingerprints: one containing the currently scanned fingerprint and another a fingerprint stored in the Database Module. We currently have two approaches for the distance calculation, Manhattan and Euclidean distances (radar [BP00] used the Euclidean approach).

The distance, $d(u, v)$, between two fingerprints vectors v and u, is computed as follows for the Manhattan distance: $d(v, u) = \sum_i |v_i - u_i|$. In the vectors, each position corresponds to a different base station/tower. The Euclidean distance is computed as follows: $sqrt((ss_1 - ss'_1)^2 + (ss_2 - ss'_2)^2 + (ss_3 - ss'_3)^2)$.

An additional parameter is available to the algorithm, in order to address the situations where the observed base stations/towers are different. When there are multiple IDs available (e.g.: 6 detected APs) but only one is matched with a stored reading, if the final score is 10, even if there is another match, with the 6 detected APs, but the sum of the differences is higher (e.g.: 2 for each), then the match with one AP is chosen over the second. This leads to higher errors, not only because we are only matching one AP, in this case, thus the possible area the user is in will be much bigger, but in fact each available AP on the second reading is probably nearer to the gathered reading. To solve this problem, a penalty value can be set, so that for each unmatched ID, the weight is added to the final score.

4.5.2 GPS averaging

The GPS averaging algorithm intends to improve the current location estimation process. When multiple GPS readings are gathered (using the information exchange mechanism), the average of the gathered values is returned, thus increasing location accuracy.

4.6 Events Dissemination

The simple API of UCLS makes using the system simple. However, it hides, from the application, information that could be otherwise useful, even if only for providing awareness to the user. To this end, we have included a mechanism that allows an application to subscribe events produced in the system. The idea is that, each component, publishes an event any time some relevant action occurs (e.g.: starting to search for devices, number of devices detected, searches completed, gathers completed, location information obtained, etc.)..

To this end, an additional call was added to the System Location API, allowing a programmer to optionally subscribe to an events notification procedure. This is a simple Interface, currently providing simple methods, that allow the reception of events that would otherwise be inaccessible to the end-user.

4.7 Adding or Modifying new Modules to the UCLS

One of the key attributes of the system is its modularity. This design allows to easily add or modify new functionalities with little effort or impact to the other components, making such changes as transparent as possible to the end user (the programmer using the System's API).

4.7.1 Technologies Information

The addition of a new technology for supporting location may imply several new characteristics, that the system much be able to support. First, it is very likely the provided information is very different from the currently available technologies. The fact that Wi-Fi, GSM and GPS wireless location information varies so much is proof enough that the obtained information can easily be very different for new implemented technologies. Thus, to support a new technology, a new Sniffer must be implemented, as well as the corresponding protocol in the Communications Module. Since each technology includes an object containing this protocol's implementation, it is only necessary to define this object. As mentioned earlier, this object is also used by the Database Module, to store the associated data by using its decoding and encoding methods.

4.7.2 Location Algorithms and Storage Mechanisms

In order to add a new Location Algorithm, the Programmer will have to add the algorithm implementation to the Location Module, in order to provide access to that specific Algorithm. Additionally, a specific retrieval mechanism for the needed information may also be added, to support the new algorithm.

Not all the available algorithms for location use Signal Strength fingerprinting [SK98, BKK96,Gut84], and the Intermediary can be used to provide a new method of creating data structures for data retrieval, to support the new location algorithm.

Further experiments would be necessary to verify if the proposed API for the modules is sufficient to support additional algorithms and location-related information.

5

Evaluation

This chapter presents the evaluation of the Unified Cooperative Location System prototype. Usability, performance and accuracy tests were performed during this period. The usability of the system was tested by using the provided Location API, implementing a simple application that allows a user to build location information maps, as well as locate himself.

The performance tests are related to energy consumptions of a device using the UCLS with different technologies. To test the accuracy of the system, an offline method was used, since it provided a faster and easier method to measure the system's accuracy.

5.1 Location API Usability

In this section, we describe our initial evaluation of the usability of the Location API. To this end, a simple J2ME menu-based application was built. Figure 5.1 presents the main menu of the application. The application was made as simple as possible, but still executing all actions that are common in location aware environments, providing a location information gathering method (Gather Local Data), a request to send gathered data to our external Web Server(Send Readings), a request for a map (Get Map), an option to estimate the device's location (locate me!), and a simple configurations menu (Configure).

Only the data gathering operation required additional input, implemented using an additional menu. In this case, Symbolic location requires manual input for the current location, as can be observed in figure 5.2. The interface allowed to define the number of readings at a time, and the location according to different types of location. Any of these boxes could receive input, and if none did, if a GPS lock was available, the location would still be stored.

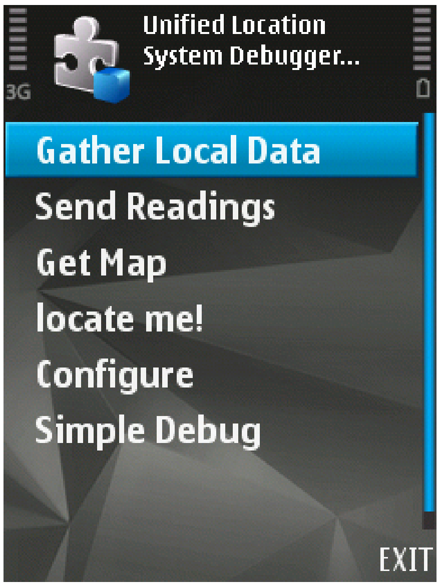
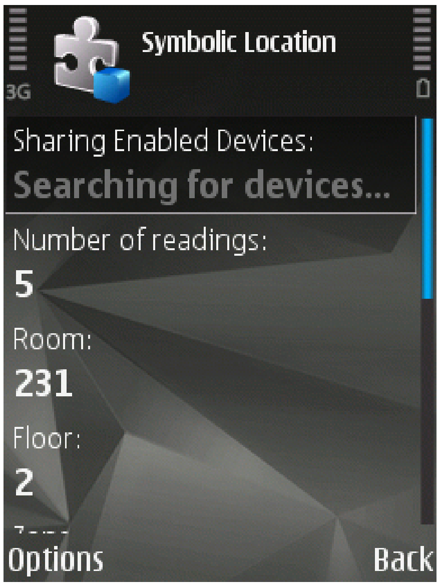


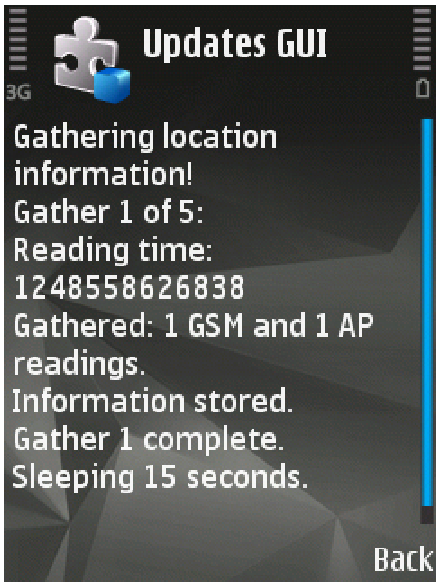
Figure 5.1: UCLS main menu



Sharing Enabled Devices:
Searching for devices...
Number of readings:
5
Room:
231
Floor:
2

Figure 5.2: Information gathering menu

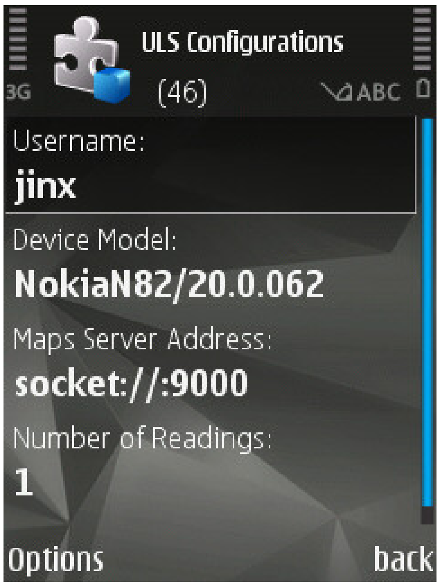
In the example of figure 5.2 it can be observed that the sharing mechanism was active and looking for available devices for information sharing. After all configuration was completed, the options menu provided an 'Add' option. This operation requested the UCLS to gather the available data and store it, including the Symbolic Information provided by the user. Since the system provides an interface for internal events notifications, this was used to supply the user with simple information during the data gathering process, as can be observed in figure 5.3.



Gathering location information!
Gather 1 of 5:
Reading time:
1248558626838
Gathered: 1 GSM and 1 AP readings.
Information stored.
Gather 1 complete.
Sleeping 15 seconds.

Figure 5.3: Gathering process feedback

The configuration menu provided the necessary configurations for the UCLS System, which also proved very simple, following the same type of menu-based configuration as the data gathering menu, as seen in figure 5.4.



Username:
jinx
Device Model:
NokiaN82/20.0.062
Maps Server Address:
socket://:9000
Number of Readings:
1

Figure 5.4: Configuration menu

The application proved to be very simple to implement, simply needing to call the available API methods in order to provide location information gathering capabilities, communicating with external servers to share and gather location information, using the several configuration options, including setting the external server address and activating and disabling the available technologies and sharing mechanism.

Although the developed application is very simple, it executes all operations usually performed in location-aware applications. This makes us confident that the API is complete and very simple to use, providing an intuitive interface to build applications with. For a more complete evaluation, it would be necessary to provide the system to third-party users and receive their feedback. Due to time constraints, it was not possible to execute such tests.

5.2 Battery tests

To better understand the energy usage of different technologies used for location estimation, and the potential impact of the sharing approach proposed in UCLS, several consumption tests were performed. Initial tests were performed to the system, in order to measure how long it would take for a mobile device to become fully discharged under certain operations. These tests were performed with a N82 Nokia device, which has an announced stand-by time of around 210 to 225 hours. The tests were performed with the screen deactivated, while running a script that measured its uptime every 10 minutes, storing the value and sleeping in the 10 minutes intervals.

Several GPS battery tests were performed under different situations, in order to test the different energy consumption. GPS was activated to continuously acquire readings at 10 seconds intervals. The measured tests included the scenarios where the device was able to acquire a clear signal; an unstable signal acquisition, where the signal was constantly being reacquired and lost; and no signal at all. The results have shown no major difference with a battery time between 11.0 and 11.5 hours and an average of 11.2 hours.

Wi-Fi battery tests were performed, scanning the surroundings at 10 seconds intervals. The tests were in different locations, where there are a large number of Base Stations that are always detectable, locations were only a few Base Stations were sometimes detected, and a location where almost no Base Station was ever detected. Similar to the GPS results, the Wi-Fi tests revealed that the energy consumption remains similar under different situations, ranging from 20.9 to 21.8 hours. The average time to deplete the battery was of 21.3 hours. Note that these tests only included scanning for Wi-Fi base-stations, with no additional communication.

We later tested the UCLS system, using a second N82 device, scanning the environment for both Wi-Fi and GSM signals, storing the acquired data. The first tests all

Table 5.1: Energy consumption using one device

Technology scanned	Energy consumption
GSM	11.5mA (97.6h)
Wi-Fi	49.2mA (22.8h)
GPS	117.2mA (9.6h)

Table 5.2: Energy consumption of the mobile device requesting data

Number of partners	Energy consumption
one	47.5mA (23.6h)
two	50.5mA (22.2h)

yielded results ranging between 14 and 15 hours, which is significantly shorter than the initial tests. The main difference here was that, despite the fact that the screen was off, we were using the dissemination system to print several messages to the screen every time an event happened (signals acquired, readings stored, etc). Leaving the dissemination system still working, but deactivating the screen updates on the GUI, we achieved up to 22h of Wi-Fi information gathering, similar to our initial tests. This shows that the energy consumption was the result of updating the GUI with fresh information.

We were unable to perform full battery consumption tests when using the Bluetooth technology. We could not run our tests until the battery was exhausted since the mobile phones we have tried always crashed after 2 hours. This test is very simple, periodically sending one packet to a server and receiving one packet in reply. We believe the problem should be related with hardware or the OS support for the Bluetooth stack.

Using a Profiler, we obtained estimates of what the energy consumption would be when using the system. We measured each situation for rounds of at least 10 minutes, in order to obtain a good estimate of the current (mA). Taking into consideration a battery of 1122mAh, we have obtained initial data when using one mobile device, scanning every 10 seconds, with Wi-Fi, GPS and GSM. Table 5.1 summarizes the results. We can see the obtained results are approximate to what the consumptions were with the full battery tests when using Wi-Fi and GPS.

Taking these results into account, we measured what the energy consumption was when using two and three devices, sharing GSM data. We measured the energy consumption of the mobile device requesting the data to the multiple devices. Table 5.2 presents the obtained results.

These results show that the energy consumption related with data sharing is smaller than the energy used for obtaining GPS, and at worse consumes as much as using Wi-

Fi signal strength information. This shows that a group of users can cooperate in a location, by sharing the responsibility of obtaining the position among them over time. It also shows that the energy consumed by the information sharing mechanism is only a fraction of the energy used for estimating position.

5.3 Accuracy tests

In this section, we present an evaluation of the location mechanism implemented in the Unified Cooperative Location System prototype. In particular, we were interested in evaluating the impact of the information sharing mechanism proposed. To this end, we have measured the accuracy of location estimation with data gathered during and after the complete implementation of the system. This data was gathered at distinct days and times in the FCT-UNL campus and stored on a database available through a webservice. The data was gathered using the UCLS, some times with shared readings, sometimes without. Most of the accuracy results presented next are only relative to the Manhattan distance, unless otherwise stated, because we achieved better results using the Manhattan algorithm. We present the final GSM accuracy results (section 5.3.1) with both the Manhattan and Euclidean approaches.

In order to simplify evaluation, as well as experiment with different algorithms and small adjustments to algorithm parameters, we have performed tests off-line. To this end, we have created a program that gathers the available information from the database, and experiments with the algorithms implemented in the system. This approach effectively simplifies the testing process while still relying on the algorithm code implemented in the system.

The simulator can receive several parameters, including the penalty value for the fingerprinting algorithm. For each iteration of the algorithm, the simulator removes a reading at random from the database. The removed reading then acts as if it was just acquired, thus simulating an acquired reading. The simulator then uses the fingerprinting algorithm to estimate location without considering the removed reading. All results presented were achieved by repeating this process 10.000 times, in order to obtain a better accuracy estimation.

5.3.1 GSM-based location accuracy

In this section, we describe the evaluation of the implemented fingerprinting-based algorithm for GSM. The tests with GSM allowed us to evaluate the effectiveness of using the information sharing mechanism for improving location estimation.

We started by mapping some areas of the FCT-UNL campus using the Information Exchange mechanism. All the information obtained by this method contained only GSM fingerprints, which were then uploaded to our Web Server. Initial GSM data was gathered from the buildings identified as EI, EII, EVII and EX in the map present in figure 5.5.

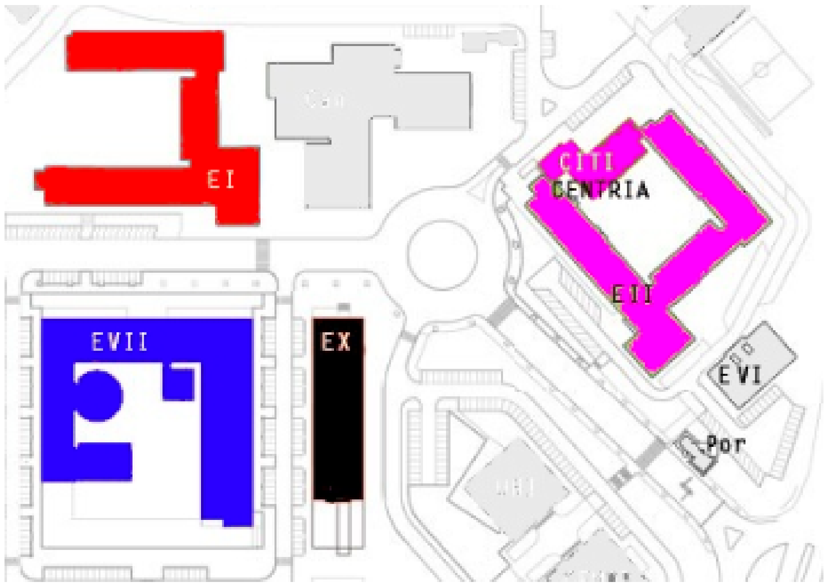


Figure 5.5: Portion of the FCT-UNL campus map

Four Nokia E70 devices and two Nokia N82 devices were used to obtain the GSM information, each one connected to one of the three available network operators in the area. For each two mobile devices connected to the same operator, the first was connected to a GSM cell, and the second to a UMTS cell, since different connections types to the same operator provide not only different cell ids, but also different signal strengths. A total of 434 grouped readings were obtained in this initial phase, each consisting of at least four identified cells. Readings were obtained throughout two different days. A total of 2213 unique readings were gathered through this method, containing 23 distinct cell ids.

We did not always obtain 6 signals at each information exchange, due to the fact that the device's bluetooth communications sometimes fail, and some mobile phones readings are sometimes lost. Out of the 434 grouped readings, only 78 contained 6 simultaneous readings. 321 contained 5 simultaneous readings, and the 35 remaining contained 4 simultaneous readings.

The readings in some buildings include unique cell ids that makes determining the location easier. We have started by focusing on the readings with identical cells for an initial evaluation of the feasibility of location estimation. To this end, we have considered three distinct cell identifiers, which were present on all of the buildings, totaling 315 readings out of the 2213.

A graphical representation of the gathered readings is presented in figure 5.6. Each axis represents the signal strength in one of the three selected cell ids. Each identified point in the graphic represents a grouped reading, containing the three selected cell identifiers. Each point belonging to a different building is clearly identified with both a different color and symbol.

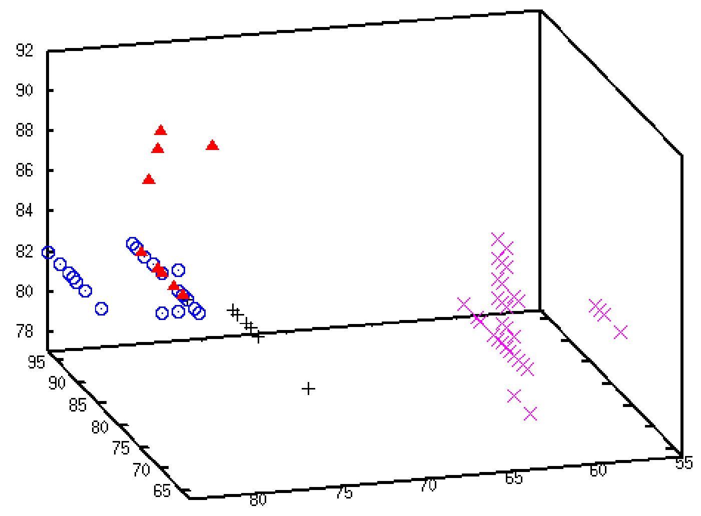


Figure 5.6: 3D visualization of the three cells signal strengths. Legend: black plus signs building I, magenta crosses building II, blue circles building VII and the red triangles building X

Although we chose the most challenging subset of our readings, with three of the most predominant cell ids available, a quick look at the figure allows us to clearly distinguish both buildings I and II from the remaining two. Some of the readings for buildings VII and X are very close, while others are clearly apart. An explanation of this fact may come from the fact that these buildings have very different characteristics, with areas ranging from no available signal, to areas surrounded by glass, with higher signal reception. As expected, the points for buildings closer to each other are also closer in the graphic.

The overall results suggest that these values can be used to identify the building the user was at with good accuracy.

After this initial observation, the basic UCLS location algorithm based on fingerprinting was used to test the actual location accuracy. This was done offline, as explained before.

To test the accuracy of the algorithm under different situations, every execution of

Table 5.3: Symbolic location accuracy using GSM

Number of Cells	Hit Rate(%)
1	86.8
2	88.7
3	92.5
4	95.6
5	88.3
6	89.3

the algorithm over the removed grouped reading was actually applied up to six times, since each grouped reading could contain at least 1 and at most 6 unique readings. Choosing one of the readings at random from the grouped reading, the algorithm was applied, assuming only one reading was available to the mobile device. A second reading was added after trying to locate with only one, applying the location algorithm again. This was repeated until all the available unique readings were considered.

Table 5.3 summarizes the results obtained. The first column represents the number of cells considered for location estimation. The second column indicates the location accuracy rate using the algorithm with no modifications (penalty value was set as 0). These results simulate the accuracy rate using information sharing up to 6 devices with differently identified cells (line one represents the accuracy with a single cell id).

From an initial observation, we can see that using only a single cell gives approximately an 87% accuracy rate. Using up to three more cells increases accuracy, as expected. However, the accuracy drops when using a total of 5 or 6 cells for location estimation. This can be easily explained, and additional parameters can set to improve these initial results.

Accuracy drops with more cells, because the more cells one adds, the more the total score is, even when more cells would obviously mean a better accuracy. For example, if 5 cells are matched, each one with a difference of 2 dBm, scoring a total difference of 10, but a second match is found, with only 1 cell, where the total difference is 9, then the single cell wins because it has a lower total score then the sum of the 5 previously matched cells. The 5 matched cells should have won, because they all have much more identical values to the compared reading then the single cell, which is distanced by a full 9 dBm score.

In order to further explore how the algorithm can be improved, three improvements to the algorithm were explored. The first improvement is based on adding an additional penalty, discussed in section 4.5.1. This penalty is added for each cell that didn't find a match while comparing with a stored group of cells. E.g.: If the current reading contains 6 cells, but only 1 was a successful match, then a penalty is added

Table 5.4: GSM accuracy using penalties

Number of Cells	% (penalty=0)	% (penalty=1)	% (penalty=10)	% (penalty=10000)
1	86.8	86.7	86.6	87.2
2	88.7	96.9	98.0	97.8
3	92.5	98.2	99.1	98.5
4	95.6	99.1	99.9	98.5
5	88.3	97.4	99.9	95.9
6	89.3	97.0	100.0	96.3

for each unmatched cell (5 penalties are added). Table 5.4 summarizes the results obtained by varying the penalty values. In this case, 1 value in penalty is equivalent to 1 dBm. The more penalty is added, the more relevance is given to results containing more successful hits. The results already improve dramatically just by adding a value of 1 dBm as a penalty, reaching even better results with penalties starting at around 4 dBms, as can be seen by quickly analyzing the graph in figure 5.7. It is also important to notice that too much penalty can affect negatively the algorithm, as can be seen when applying a penalty of 10000.

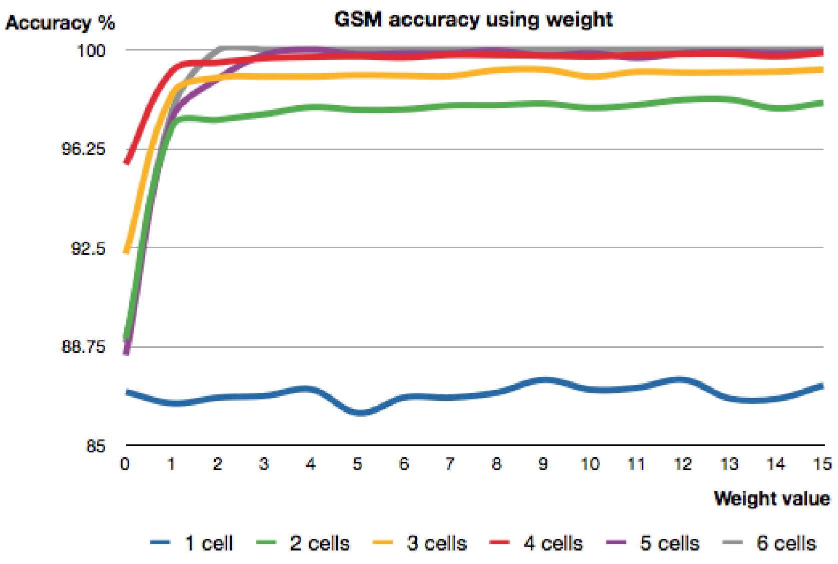


Figure 5.7: GSM accuracy using penalties

The second devised improvement is based on attempting to distribute the total score obtained when calculating a similar result, by dividing the total absolute difference by the number of successful matching cells. Using this method, if a match contains 5 cells, then the total score (the absolute difference between the matched cells) is divided by the number of matched cells (in this case, by 5), effectively lowering the final score to 1. Inversely, if a match contains only 1 cell, with a total score value of

Table 5.5: GSM accuracy using division improvement

Number of Cells	Hit Rate (normal)	Hit Rate with division
1	86.8	86.8
2	88.7	89.0
3	92.5	92.7
4	95.6	96.4
5	88.3	93.4
6	89.3	93.7

Table 5.6: GSM accuracy using cell-choosing improvement

Number of Cells	Hit Rate (normal)	Hit Rate with cell-choosing
1	86.8	86.7
2	88.7	96.7
3	92.5	97.9
4	95.6	98.7
5	88.3	95.0
6	89.3	95.0

4, the final score is divided by 1, thus the reading containing more cells wins, since it now has a lower value. Table 5.5 summarizes the results obtained using this second improvement. This method does not improve accuracy by much, except when using more cells.

The third developed method was devised to choose which reading to return, when the score used to determine a location estimation coincides with another one. Since the database contains many readings, sometimes the best computed scores are equal. The base algorithm just selects the first, lowest score, that was found on the database. However, that decision can be improved: if more matching cells were used to calculate the position, then it is more likely that their reported location is more accurate than a group of cells with the same result but fewer matching cells. What this method does is very simple: if a group of matching results have the same score, the group with the highest amount of matching cells is chosen. As can be seen in Table 5.6, this cell-choosing method improves the results when using more then one cell.

Finally, table 5.7 summarizes the result obtained when using the three improvements in conjunction (with a penalty set at 25). The improvements can clearly be seen. It is also interesting to point that, although the results have been significantly improved using multiple cells, using only one cell is significantly worse than using only 2. This is very encouraging, suggesting that good results can be obtained using the system with just two mobile phones, that can be from the same operator.

After obtaining the previous results at identifying buildings correctly, the next phase of testing was initiated, by mapping the wings, floors and rooms of building II. The

Table 5.7: GSM accuracy using the three improvements

Number of Cells	Hit Rate (normal)	Hit Rate (improved)
1	86.8	86.7
2	88.7	98.0
3	92.5	99.1
4	95.6	99.8
5	88.3	100.0
6	89.3	100.0

Table 5.8: GSM accuracy using improved algorithm

Number of Cells	% Room	% Floor	% Wing
1	73.16	85.3	86.6
2	94.9	96.8	97.23
3	98.15	98.9	98.68
4	98.9	99.1	99.5
5	99.5	99.7	100.0
6	98.9	100.0	100.0

following table 5.8 summarizes the obtained results, using the improved algorithm, identifying rooms, floors and wings.

The obtained results were surprisingly good when using multiple cells, even for locating the room the user was at. Once again, using only one cell gives a considerably lower hit rate than when using two or more cells.

For a final phase of testings, we made two additional adjustments. First, we merged the data obtained when mapping the campus (containing the four buildings) with the additional GSM data gathered at building I which included symbolic information for the wings, floors and rooms. This allowed us to obtain a more realistic view of a bigger database. The second adjustment, was to simulate an information exchange between nearby devices that were not in the same position. Instead of randomly selecting a random reading from the database, we started by randomly selecting a room. Then, we randomly selected up to 6 random fingerprints from all the available Grouped Readings for the room. When testing the location, we now use from 1 to 6 signal strengths from different fingerprints.

Tables 5.9 presents the accuracy results for the first adjustment, with the merged databases. For these final GSM accuracy tests, we included the accuracy when using both the Euclidean and the Manhattan distance. The second column contains the results obtained by simply merging the databases and using our current algorithm (Manhattan distance), while the third represents the accuracy when using the Euclidean distance algorithm.

Finally, table 5.10 presents the results using the Manhattan and Euclidean distance

Table 5.9: Location accuracy using GSM inside a building using merged databases

Number of Cells	% Manhattan Merged	% Euclidean Merged
1	73.6	74.0
2	94.6	94.4
3	97.7	96.7
4	99.0	97.9
5	99.3	98.0
6	99.5	93.2

Table 5.10: Location accuracy using GSM inside a building using merged databases and the exchanging adjustment

Number of Cells	% Manhattan exchanged	% Euclidean exchanged
1	76.3	75.8
2	84.7	80.9
3	88.0	83.2
4	91.0	84.0
5	93.2	87.2
6	95.3	87.3

algorithms, this time using our approach of simulating an exchange between devices in the same room. Although the accuracy drops when considering randomly distanced readings, the results are still good - with an accuracy over 84% when using information from only two cells. These results also suggest that it is important to investigate ways to consider the distance between the devices that share information when estimating location.

Overall, the results obtained in our experiments suggest that the information sharing mechanism can be used to build a practical GSM-based location system.

5.3.2 Wi-Fi-based location accuracy

In this section, we describe the evaluation of the system when using Wi-Fi location estimation. To execute these studies, we have mapped building II again, this time with Wi-Fi readings. A Nokia N82 device was used for the duration of these tests, gathering data at different times and days. The device gathered and stored the Wi-Fi signals it detected. The sharing mechanism was disabled for these tests.

A total of 268 readings were gathered for these tests, containing 31 distinct Base Station identifiers. The average number of identified Wi-Fi Base stations per reading was of 8, and the average number of readings per room was of 13.

Table 5.11 contains the accuracy results obtained when using Wi-Fi estimation, using both the normal and the improved location-estimation algorithm. The results ob-

Table 5.11: Wi-Fi accuracy

Algorithm	% Room	% Floor	% Wing
Normal	68.9	82.8	75.6
Improved	96.2	98.6	100.0

Table 5.12: Wi-Fi signal variation over different days

Days	% AP1	% AP2
Day 1	-68.5+/-4.9	-84.2+/-1.8
Day 2	-62.9+/-3.7	-83.9+/-3.2
Day 3	-66.0+/-5.3	-83.5+/-3.5
Day 4	-70.0+/-3.1	-82.6+/-3.2
Day 5	-68.0+/-2.9	-80.6+/-8.3

tained were surprisingly worse then expected and are even worse then those obtained with GSM. This surprise came from the fact that previous research, like RADAR [BP00], reported very good accuracies when using relative location estimation, with a median resolution in the range of 2 to 3 meters. Our symbolic locations are larger areas than a coordinate in a map, thus better results were to be expected.

To understand the obtained results, two additional studies were performed. First, we have studied the stability of signal strength in our environment. To this end, we left an N82 and E70 device, previously used to gather Wi-Fi readings, on the same place for a long period of time, making readings every 5 seconds. This allows us to study how the Wi-Fi signal strength varied over time, as well as how it varied from one device to the other. This was done over the course of 5 days.

A total of 13442 readings were collected over the course of the 5 days, with an average of 4.4 detected APs per reading and a total of 7 distinct APs identified. Table 5.12 shows the results obtained on two of the detected APs on the N82 device, showing how signal varied over time. Additionally, figure 5.8 is a representation of how the signal varied over time on the same identified Base Station, when comparing the N82 and E70 device.

The results for other base stations were consistent, showing two properties: the signal strength can vary over time and different devices obtain different values in the same place (one mobile phone consistently read lower values). This can be due to a number of reasons, ranging from the current number of users on the network, its instability (APs may adjust the signal due to the interference), and the device's hardware. In order to provide a system usable with multiple mobile devices, given a shared database, these results show that it is necessary to study how to calibrate a device to the available data, which would allow for better location estimations. This seems less important with GSM, where the signal presented itself more stable and reliable during

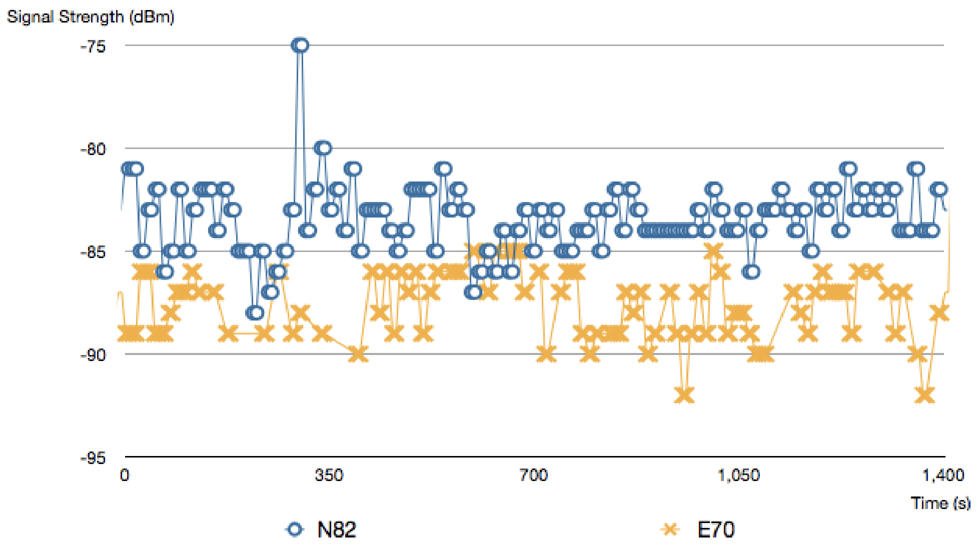


Figure 5.8: Signal Strength variation on an N82 and E70 device

our experiences.

We then proceeded to an additional information gathering, again with the available devices (N82 and E70 devices), in order to map the department again. This time, we have recorded the device used for each reading. This information was gathered over the course of two days, using both devices to gather data at the same symbolic locations. The N82 recorded 268 readings on the first day and an additional 63 on the second. 30 distinct APs were detected, with an average of 8.23 APs per reading. The E70 recorded 234 readings on the first day and an additional 61 on the second. It only detected 18 distinct APs, averaging 4.84 AP detections per reading.

Four additional tests were performed with the gathered data, studying how the room level accuracy varied with the different tests. The first test, represented by the first line in table 5.13, is the accuracy obtained when using the dataset acquired with the N82 device on the first day. The second line corresponds to a larger database, consisting of the readings acquired from the N82 over the course of the two days, containing the additional 61 readings. The third line corresponds to the readings obtained over the two days with the E70 device. Finally, the fourth line corresponds to a database containing readings of both the E70 and N82 devices over the two days.

Although the E70 device gathered slightly less readings because it sometimes crashed, it still contained data gathered from all the symbolic locations, at the same times, that the N82 also obtained. However, it did not gather as many APS on average, nor did it detect as many unique fingerprints as the N82 device did. That can probably explain why the E70 device did not manage to obtain as good an accuracy as when using the N82 device. Another interesting note, is how the accuracy for the N82 de-

Table 5.13: Wi-Fi floor accuracy with different readings

Data set	% Normal	% Improved
N82: day 1	69.0	92.6
N82: day 1 and 2	82.8	97.7
E70: day 1 and 2	75.7	93.3
N82 and E70: day 1 and 2	81.9	96.4

vice improved, when we added another 63 readings. it would be interesting to further test how the accuracy would increase or decrease with added wireless data. The RADAR [BP00] research also provided results where better accuracies were obtained when a higher number of fingerprints were available.

6 Conclusions

The Unified Cooperative Location System is a location system designed to explore the multiple information available in a mobile device for performing location estimation. The system design is modular and promotes extensibility. This approach simplifies the addition of new technologies, algorithms or storage mechanisms used in the location estimation process. The UCLS's architecture provides the necessary flexibility to make these modifications, without breaking the functionality of the remaining Modules.

By taking advantage of the available wireless communication infrastructure, the system provides a low-entry cost solution. The use of J2ME allows the solution to be used in most of the currently available devices on the market. The only modules that must be implemented in other language are the module for obtaining raw information used in the location estimation process. In our prototype, they were implemented in PyS60 for Symbian OS. However, we believe that they are simple enough to be easily implemented in any other mobile operating system.

The ability to use different technologies and algorithms for location estimation allows to use the one that is more adequate in each circumstance. Our initial studies regarding energy usage show that different technologies consume different amounts of energy. Thus, it can be possible to balance the accuracy of the system with the energy consumption by using different approaches.

When compared with similar systems, UCLS presents a novel mechanism for information sharing among mobile devices. The ability to share location information can provide several improvements to the location system. First, mobile devices can obtain location information that otherwise could not be available to them. Second, the ability to share location information allows for improved location estimation accuracy by obtaining additional measurements. The results presented in section 5.3.1 show that by sharing GSM signal strength among nearby users, it is possible to estimate user's location with good accuracy (over 95% for symbolic room-granularity). This makes our approach the first to present a practical GSM-based location that works in common mobile phones that can only return information about the single tower they are connected to.

Finally, it allows devices to conserve energy by cooperating for obtaining the needed information and sharing them via bluetooth. Our initial energy consumption studies seem to corroborate this.

UCLS includes a server that promotes information sharing among users. Sharing readings databases with a community allows a faster construction of the information needed for location estimation, which can take some time to build initially, when no data is available. Additionally, the system can take advantage of more computationally rich components to build more advanced databases or services for the mobile device.

UCLS provides a Location API that is very simple to use and configure. A programmer does not necessarily need to know how the system works to provide the end user with a location estimation. The availability of the event dissemination allows the user to get some feedback of the current operations.

The application developed allowed to test the system's API. Although the API revealed itself easy to use in an application that performed most operations of a location application, additional experience with additional programs is needed in the future.

6.1 Future Work

During the research and implementation of the Unified Cooperative Location System, some aspects that could help improve the system were identified.

Relative location estimation through the use of bluetooth signal strength information could be used to improve location estimation when exchanging information. Although we have considered this direction, it was not possible to further address it because the Symbian OS based mobile phones we were using did not provide the needed information.

Some mobile devices, such as some using the Windows Mobile Operating System provide Bluetooth Signal Strength, and could be used to calculate and share that information. However, these devices do not usually contain bluetooth communication access to J2ME, so it would be necessary to add that functionality at a lower level, eventually providing a similar communication protocol with the main system as the one used for the Sniffers. Another option for relative location estimation would be to use computers available in the environment, which would estimate the relative distances between them and the devices, and share that information.

Location solutions based on signal propagation modeling solutions would also be interesting to research. The ability to quickly map an indoors location through the use of floor plan is interesting, even though it usually provides worse results then fingerprinting [BP00,JBPA06]. It presents a bigger advantage, in the fact that it is fairly faster to build a signal propagation map than fingerprinting.

Signal Propagation Modeling could also be very interesting when used with GSM Towers and our information exchange mechanism. If a database containing Cell Towers absolute positions was available, either stored on the mobile phone or directly available through the Mobile Operators, we could use our exchanging mechanism to triangulate our position, based on the shared signal strengths and distance estimations to the surrounding Cells.

Finally, further studies on energy consumptions and automatic algorithm selection should be done, in order to provide a system that is able to choose the required algorithm and technologies depending on the user's preferences and/or needs. Also, further studies on bluetooth energy consumption would be interesting, in an attempt to further decrease the energy consumption when sharing data.

Bibliography

[BKK96] Stefan Berchtold, Daniel A. Keim, and Hans-Peter Kriegel. The x-tree: An index structure for high-dimensional data. In *VLDB '96: Proceedings of the 22th International Conference on Very Large Data Bases*, pages 28–39, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers Inc.

[Bor01] Gaetano Borriello. A survey and taxonomy of location systems for ubiquitous computing. Technical report, IEEE Computer, 2001.

[BP00] Paramvir Bahl and Venkata N. Padmanabhan. Radar: an in-building rf-based user location and tracking system. In *Proceedings of IEEE INFOCOM 2000*, volume 2, pages 775–784, 2000.

[Bra06] Brain, Marshall and Harris, Tom. How GPS Receivers Work, 25 September 2006. `http://adventure.howstuffworks.com/gps3.htm`.

[DDRC04] J. C. Danado, António Eduardo Dias, Teresa Romão, and N. Correia. Bposs - a bluetooth positioning system. In *Proc. of IST Mobile & Wireless Communications Summit 2004*, pages 985–989. Elsevier, 2004.

[FPY96] A. Falsafi, K. Pahlavan, and Ganning Yang. Transmission techniques for radio lan's-a comparative performance evaluation using ray tracing. *Selected Areas in Communications, IEEE Journal on*, 14(3):477–491, 1996.

[Gut84] Antonin Guttman. R-trees: a dynamic index structure for spatial searching. In *SIGMOD '84: Proceedings of the 1984 ACM SIGMOD international conference on Management of data*, pages 47–57, New York, NY, USA, 1984. ACM.

[HHS$^+$02] Andy Harter, Andy Hopper, Pete Steggles, Andy Ward, and Paul Webster. The anatomy of a context-aware application. *Wirel. Netw.*, 8(2/3):59–68, 2002.

[HS99] Ken Hinckley and Mike Sinclair. Touch-sensing input devices. In *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 223–230, New York, NY, USA, 1999. ACM.

[JBPA06] Yiming Ji, Saâd Biaz, Santosh Pandey, and Prathima Agrawal. Ariadne: a dynamic indoor signal map construction and localization system. In *MobiSys 2006: Proceedings of the 4th international conference on Mobile systems, applications and services*, pages 151–164, New York, NY, USA, 2006. ACM Press.

[LCC$^+$05] Anthony LaMarca, Yatin Chawathe, Sunny Consolvo, Jeffrey Hightower, Ian Smith, James Scott, Timothy Sohn, James Howard, Jeff Hughes, Fred Potter, Jason Tabert, Pauline Powledge, Gaetano Borriello, and Bill Schilit. *Place Lab: Device Positioning Using Radio Beacons in the Wild*. 2005.

[LdL08] Anthony LaMarca and Eyal de Lara. *Location Systems: An Introduction to the Technology Behind Location Awareness*. Morgan & Claypool Publishers, 2008.

[LHSC05] Anthony LaMarca, Jeff Hightower, Ian Smith, and Sunny Consolvo. Self-mapping in 802.11 location systems. In *Proceedings of the Seventh International Conference on Ubiquitous Computing (Ubicomp 2005), Lecture Notes in Computer Science*, pages 87–104. Springer-Verlag, 2005.

[Mac67] J. B. Macqueen. Some methods for classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symposium on Math, Statistics, and Probability*, volume 1, pages 281–297. University of California Press, 1967.

[PCB00] Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan. The cricket location-support system. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 32–43, New York, NY, USA, 2000. ACM Press.

[PFTV92] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C : The Art of Scientific Computing*. Cambridge University Press, October 1992.

[RD07] L. Reyero and G. Y. Delisle. Always best located, a pervasive positioning system. In *Wireless Pervasive Computing, 2007. ISWPC '07. 2nd International Symposium on*, 2007.

[SK98] Thomas Seidl and Hans-Peter Kriegel. Optimal multi-step k-nearest neighbor search. In *SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 154–165, New York, NY, USA, 1998. ACM.

[Sky09] Skyhook. Skyhook Wireless, 12 June 2009. `http://www.skyhookwireless.com/howitworks/`.

[SR92] S.Y. Seidel and T.S. Rappaport. 914 mhz path loss prediction models for indoor wireless communications in multifloored buildings. *Antennas and Propagation, IEEE Transactions on*, 40(2):207–217, Feb 1992.

[Val94] R.A. Valenzuela. Ray tracing prediction of indoor radio propagation. *Personal, Indoor and Mobile Radio Communications, 1994. Wireless Networks - Catching the Mobile Future., 5th IEEE International Symposium on*, 1:140–144 vol.1, Sep 1994.

[VLHdL07] Alex Varshavsky, Anthony LaMarca, Jeffrey Hightower, and Eyal de Lara. The skyloc floor localization system. pages 125–134, 2007.

[VPKE08] Alex Varshavsky, Denis Pankratov, John Krumm, and Eyal de Lara. Calibree: Calibration-free localization using relative distance estimations. In *Proceedings of the 6th International Conference on Pervasive Computing (Pervasive)*, volume 5013 of *LNCS*, pages 146–161. Springer, 2008.

[WFG92] Roy Want, Veronica Falcao, and Jon Gibbons. The active badge location system. *ACM Transactions on Information Systems*, 10:91–102, 1992.

A

Wi-Fi Gathered Readings

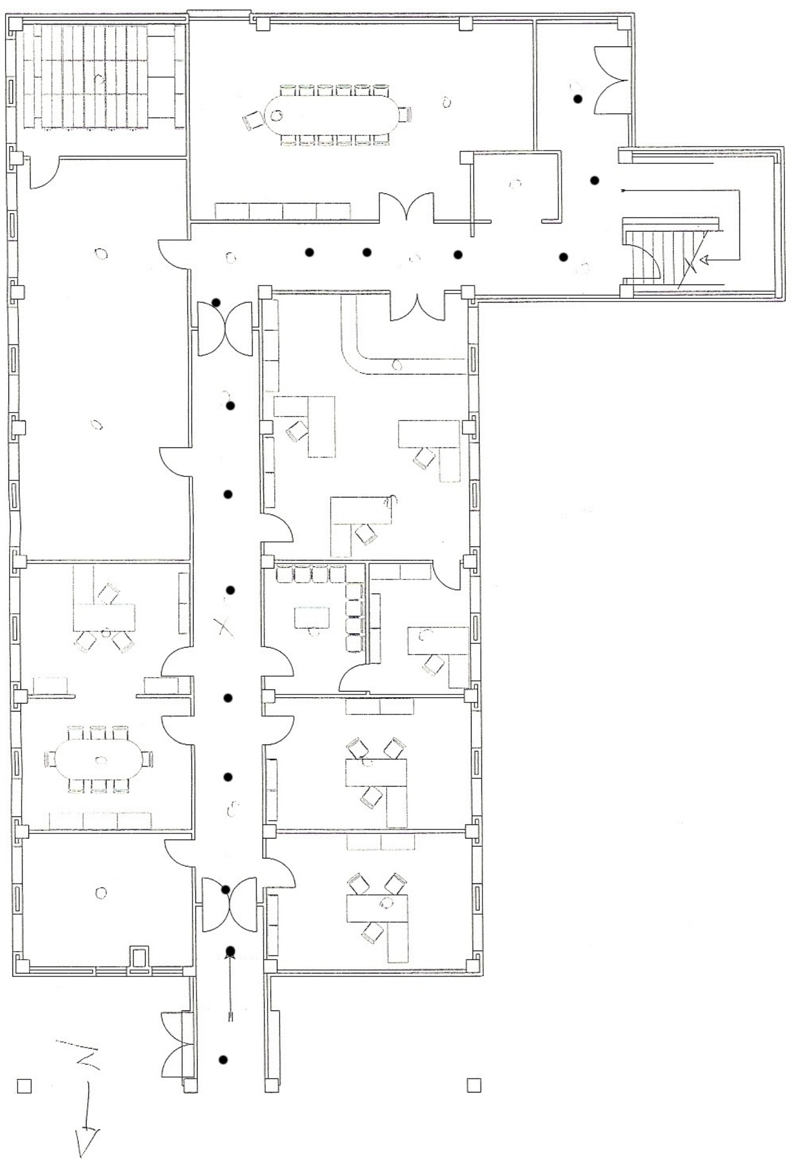


Figure A.1: Building II, CITI section, floor 1

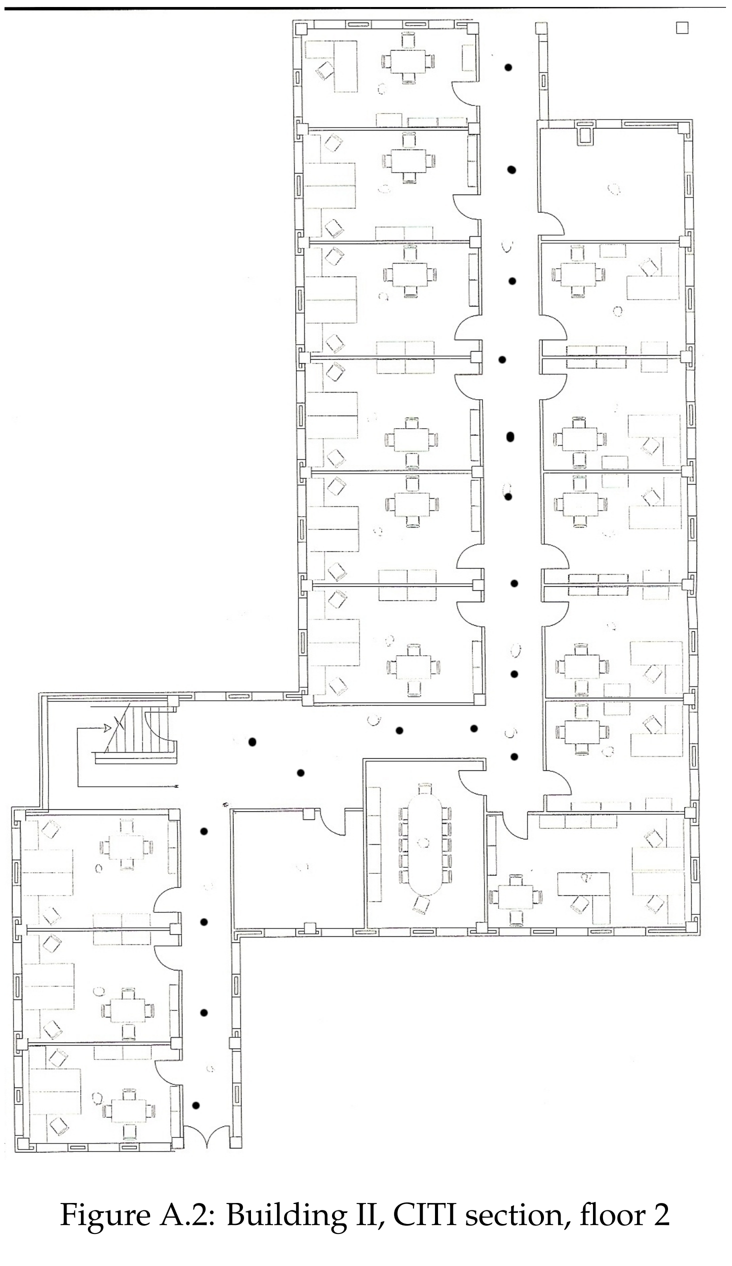


Figure A.2: Building II, CITI section, floor 2

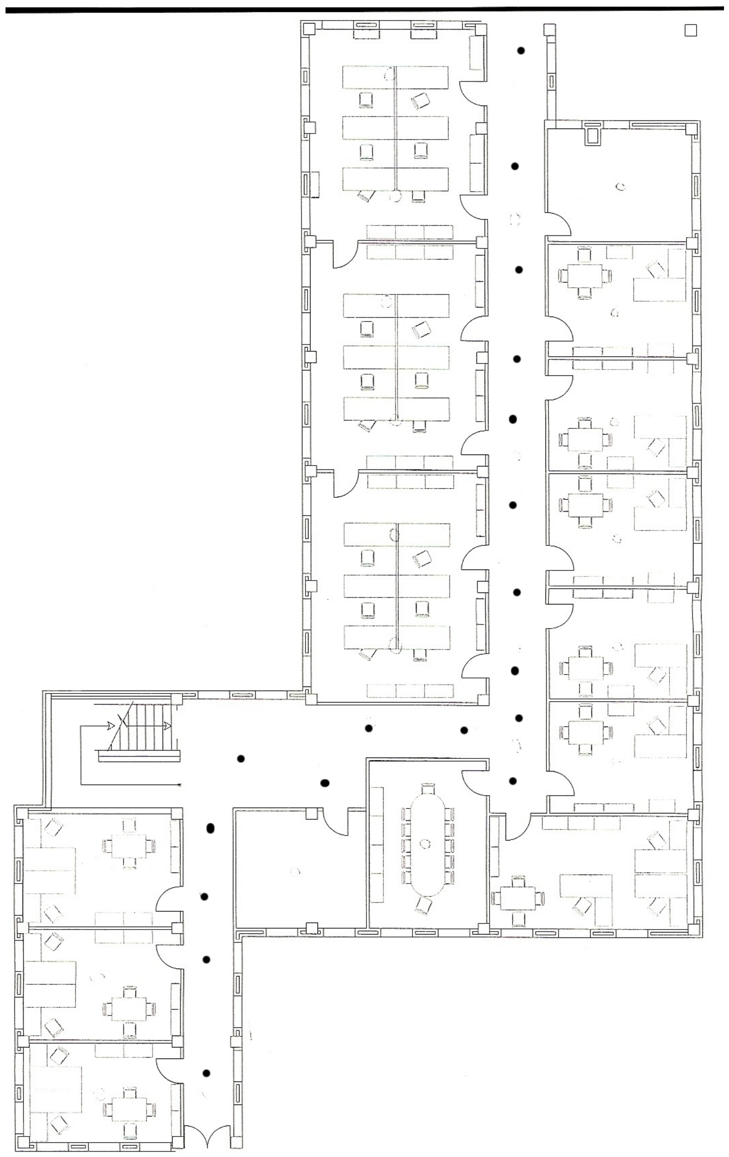


Figure A.3: Building II, CITI section, floor 3

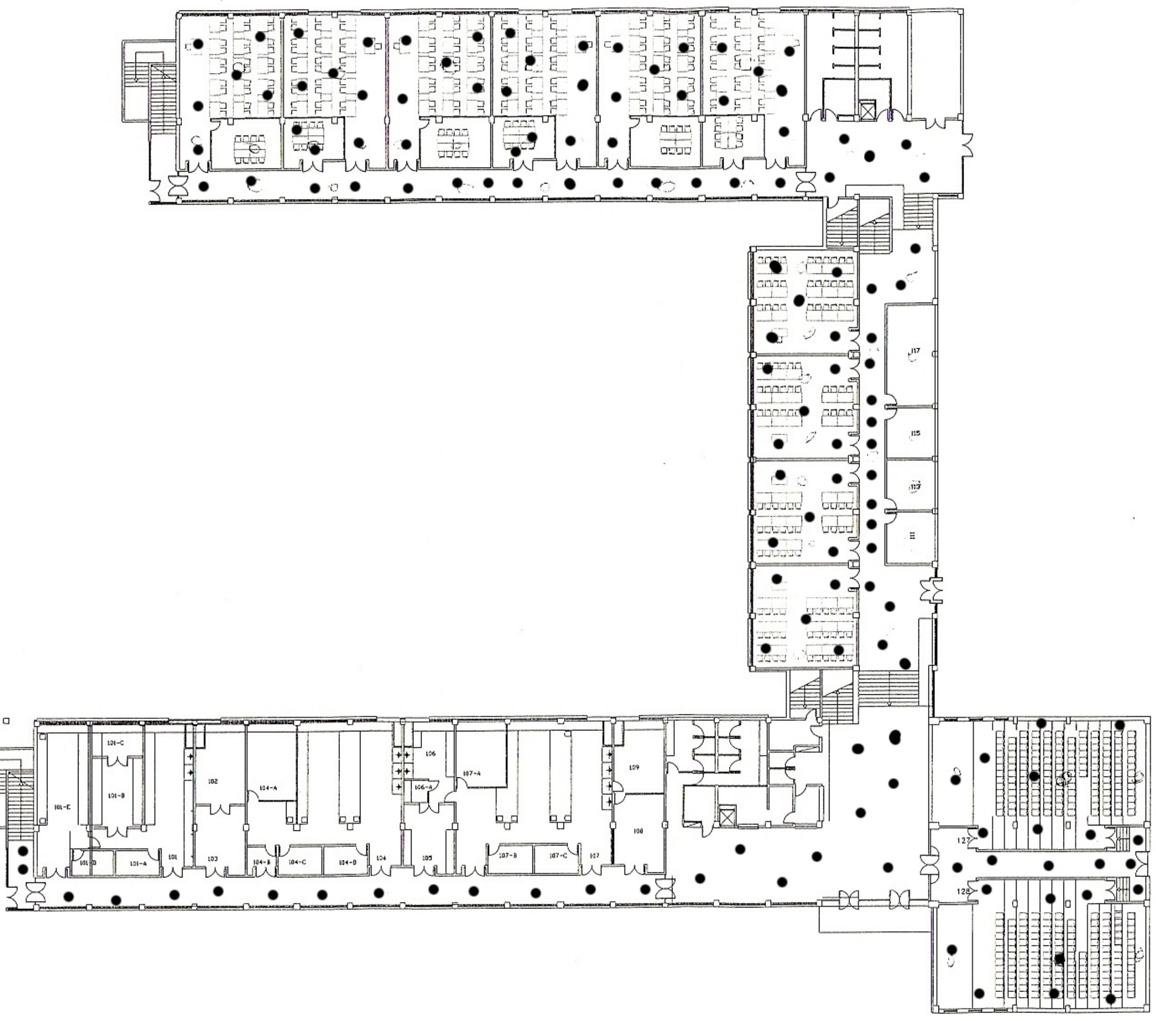


Figure A.4: Building II, remaining corridors, floor 1

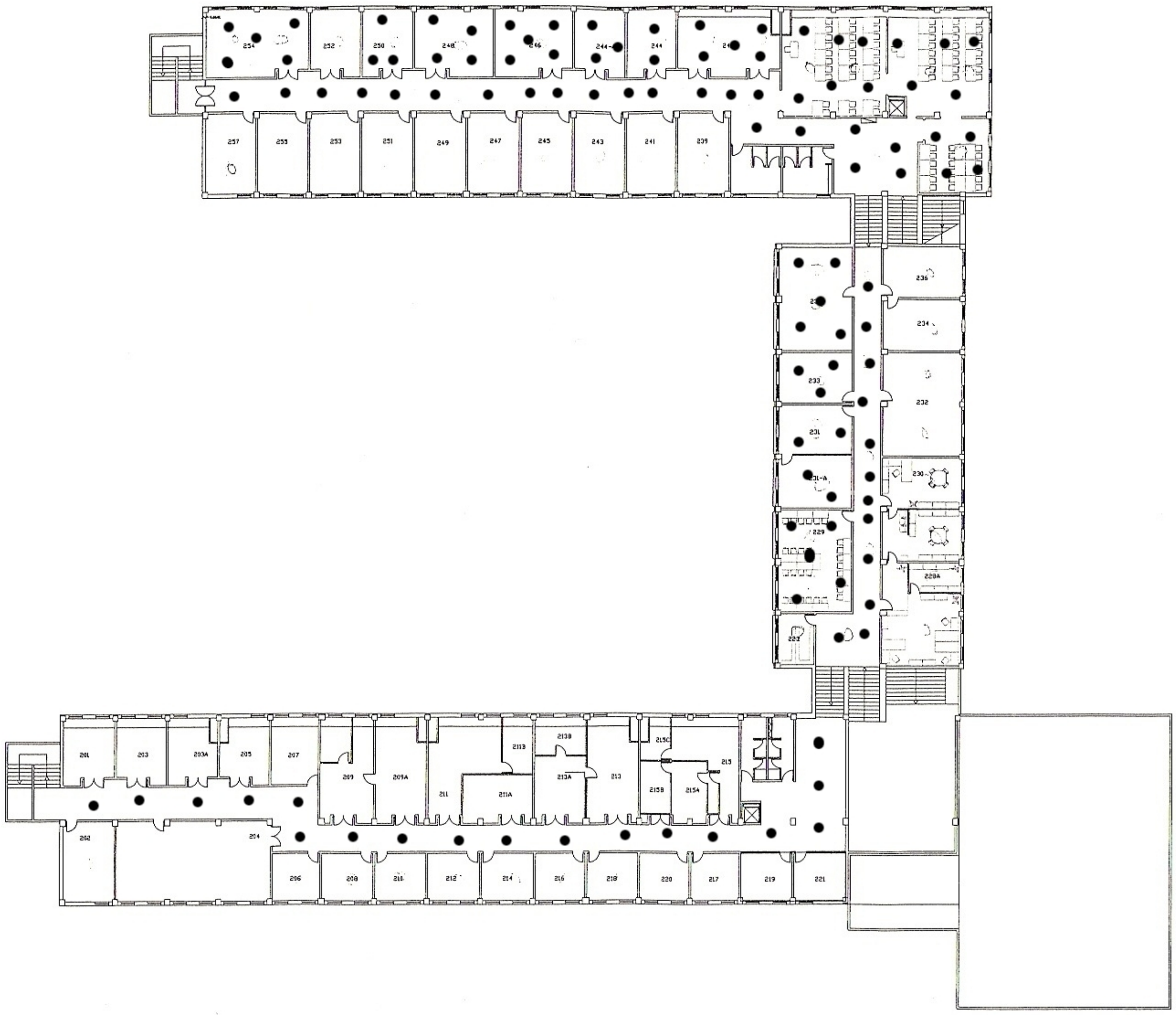


Figure A.5: Building II, remaining corridors, floor 2