THE PLACE LAB PROJECT

Seminar

by

DANIEL KÖLSCH

presented to

Department of Computer Science IV
Prof. Dr.-Ing. Wolfgang Effelsberg
Faculty for Mathematics and Information Science
University of Mannheim

January, 2006

Tutor: Dipl.-Wirtsch.-Inf. Thomas King

CONTENTS

LIST OF ABBREVIATIONS

AP	Access Point
BSSID	Basic Service Set Identifier
CGI	Common Gateway Interface
FCC	Federal Communications Commission
GPS	Global Positioning System
GSM	Global System for Mobile Communications
HTML (XHTML)	(Extended) Hypertext Markup Language
ID	Identifier
NNSS	Nearest Neighbor in Signal Space
SS	Signal Strength
UMTS	Universal Mobile Telecommunications System
Wifi (also WiFi, Wi-fi, Wi-Fi, or wifi)	Wireless Fidelity
WiMAX	Worldwide Interoperability for Microwave Access

LIST OF FIGURES

LIST OF TABLES

1 Introduction

The ability to estimate and communicate a user's position is a key component in developing mobile computing applications. There has been a lot of research to build systems that allow estimating a user's position. A variety of sensing technologies were deployed, for example ultrasonic time of flight, infrared proximity, radio signal strength and time of flight, optical vision and electro-magnetic field strength, but existing location sensing approaches mostly lack of ubiquity and an easy-to-use architecture. Many approaches are too expensive to deploy, either in building the appropriate infrastructure or in buying special equipment for the user. Centralized systems require a special infrastructure that should be able to communicate with the outer world; in this case these are clients that are supposed to be tracked.

Existing location sensing systems are designed for either indoor or outdoor use. *GPS* for example only works outside buildings, because the receiver needs to have line-of-sight connection to the satellites. *GPS* was designed to maximize coverage and it is not applicable for indoor purposes. On the other hand, indoor systems like the infrared based *Active Badge Location System* [8] require expensive hardware infrastructure. Also designed as an indoor system is *RADAR's* [5] fingerprinting approach which is more sophisticated and promises better adaptability than those systems that need an extra infrastructure. With location fingerprinting one can take advantage of sensing technologies that already exist in most offices, which are mainly Wireless LAN access points. However, there are still some drawbacks, because *RADAR* needs a calibration step to train the system. This calibration step is in most cases a time-consuming procedure which boosts the barrier-to-entry. Second, deployment of these systems is only feasible in small environments, due to a costly calibration step.

For these reasons, Place Lab has the goal to build a location sensing system that lowers the barrier-to-entry by providing an easy to use architecture for users and developers, and to accomplish a maximal coverage of daily life [1].

2 Idea and Overview of the Place Lab Project

2.1 What is Place Lab?

Place Lab is a research project of the Intel Research Lab in Seattle, involving currently 25 core researchers and some universities (nearby the other research labs), which are University of Washington, University of California San Diego, University of California at Berkeley, Carnegie Mellon University and University of Cambridge to mention most of them.

The ultimate goal of institutions like Place Lab is a broad adoption of location-aware computing. Therefore it should be effortless, familiar and rewarding to use and develop these systems and applications (e.g. Google). The only way to achieve this ultimate goal is a truly ubiquitous deployment at a global scale, a strong community of developers and users, to form the basis for further developments and future applications. To make this happen, the Place Lab community has to make location-aware services valuable and readily accessible by a large user community in daily situations all over the world. The Place Lab software should also be ready for indoor and outdoor usage, which is crucial to accomplish a real-world system [4].

Place Lab tries to overcome the major barriers by

- designing low cost, highly convenient position-sensing technology
- making users comfortable in respect to their location privacy
- having existing web content easily customized to geographic locations

"We want widely available location-aware computing that people can use with today's technology," said Yatin Chawathe from Intel's Research Lab in Seattle. "The biggest problem with GPS is that it doesn't work indoors or in cities with tall buildings. More importantly any new service must be privacy observant: the user has to have control."

This point introduces another crucial fact - to develop a privacy-observant location system. The system will be designed estimating the location at user's side on the device itself. Therefore, there is no need for a centralized system or communication with the outside world or an infrastructure. The user remains anonymous to a certain controllable level [1].

2.2 Reasons for slow adoption – Place Lab initiative

In the following we can identify a cycle of technological and social barriers, reasons for slow adoption concatenated and depending on each other [4].

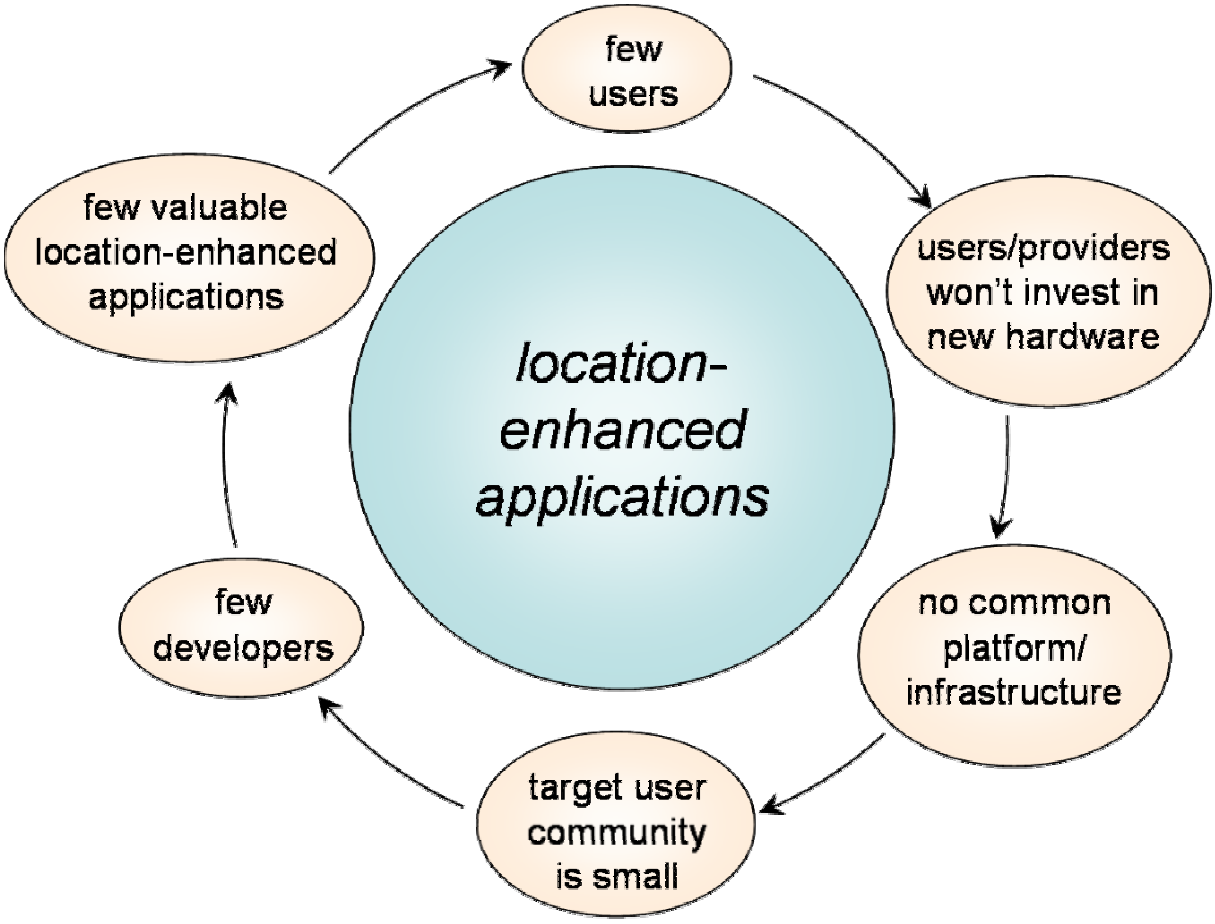


Figure 1 - General cycle of reasons for slow adoption of location-aware systems and applications

Place Lab identified this cycle as a general pattern, leaving out important issues like privacy for example. However, in order to break through this cycle, they want to enable four things [4]:

- Place Lab uses technology standards that already exist. The aim is to lower the technical barrier, so the idea is to reuse or to "recycle" familiar technologies in a new context, namely in the location-aware domain. Web Developers for example are used to the common web "cgi" services model, and they probably would not have any problems to port their applications to a mobile device by adding a small amount of location-aware functionalities provided by some predefined libraries. A good example for this approach is the modification of the common HTML standard to XHTML-Basic which is a sort of miniature type of HTML 4.0, especially designed for small displays of mobile devices. HTML belongs to the standard repertoire of a web developer, so after a little familiarization it is absolutely convenient for the developer to transfer the common elements to a small display.

  In the case of location-aware systems, this new component or improvement of the already known is an additional piece of software that should be easily deployable and seamlessly connectible with the application (like the way you use new Java packages for example). Well defined interfaces are here an absolute must to keep it as simple as possible for the developer.

  The overall goal must be to remove or at least to reduce technological barriers of entry as much as possible. At least equally important is the need to keep technology problems away from the user's side. Thinking of the installation of a location-aware system on a mobile device, it shouldn't take more than one click to upgrade a standard browser with some location-awareness. A mature and well scalable privacy management is also an essential point speaking of convenience.

- Place Lab has the aim to develop a leading-edge developer and user community from a grass roots effort. University students, Wireless LAN clubs and enthusiasts should

build the basis, followed by stragglers and eventually the mass market. To establish such a community it is necessary to reduce the technological barriers, as we already figured out. A good example is DoCoMo's iMode because a lot of people could already write HTML code and web-services and it was easy to deploy this knowledge on the iMode system. A community of developers and users was formed and expanded.

- Speaking of real testing and real world data, you need a real world testing environment, a laboratory which will be used by researchers and developers from all over the world. One could implement his newly developed algorithm; another might devise a new user interaction model. All that could be done with live data and actual facts with real world circumstances.

- Location-awareness is a subset of context-awareness. There are a lot more attributes you can associate with a user than just knowing his mere location. For some applications it could be useful to know something about the current situation of the user, people or activities around the user. Place Lab tries to move the developer and user community in this direction and sensitize them as a next step for future developments.

Now four major goals of Place Lab can be identified [4]. Place Lab should enable the following:

- Inexpensive, easy-to-use technology standards for development and usage of location-enhanced systems
- A grown community of developers and users
- A real-world test environment
- Foundation of context-aware and proactive computing research

2.3 How does Place Lab work?

Technically the system is based on radio beacons, which are basically small periodically sent radio signals by Wireless LAN access points (IEEE 802.11) [16], fixes Bluetooth [17] stations and GSM [18] towers. In the next subsection reasons for using radio beacons and a list of advantages are given.

2.3.1 Radio Beacons

By collecting and using radio beacons that already exist in the wild Place Lab ensures to keep the user's privacy at a high level. Place Lab calls this method of gathering data "passive listening". By doing so, the system provides anonymous location estimation.

The first major advantage of using radio beacons is the fact that all beacons have a unique or semi-unique identifier (ID). There is no need for setting up a new standard to provide explicit identification of the received signals. This is crucial for making the system usable worldwide.
Place Lab doesn't have to install a new infrastructure because we only use that already exists. Place Lab also doesn't have to keep track of administration and maintenance of the infrastructure. It grows self-controlled, at least controlled by the community concerned about the specific technology. Let's take 802.11 as an example. The number of access points grows steadily, especially in large and dense areas like cities and towns. But also privately held access points increase in number. Place Lab takes advantage of this growth by making the system depending on 802.11 beacons. From this it follows that there is a high density of usable data which represents a major goal of the Place Lab initiative [4].
GSM beacons are also widely present since we have almost worldwide GSM coverage. This should promise good experimental results when exploring beacon coverage.
Bluetooth beacons are negligible because Place Lab couldn't find enough fixed Bluetooth stations currently deployed.

2.3.2 Architecture and Privacy

Figure 2 illustrates a usage scenario and indicates the mechanism of a client-based calculation of *position-without-communication*:

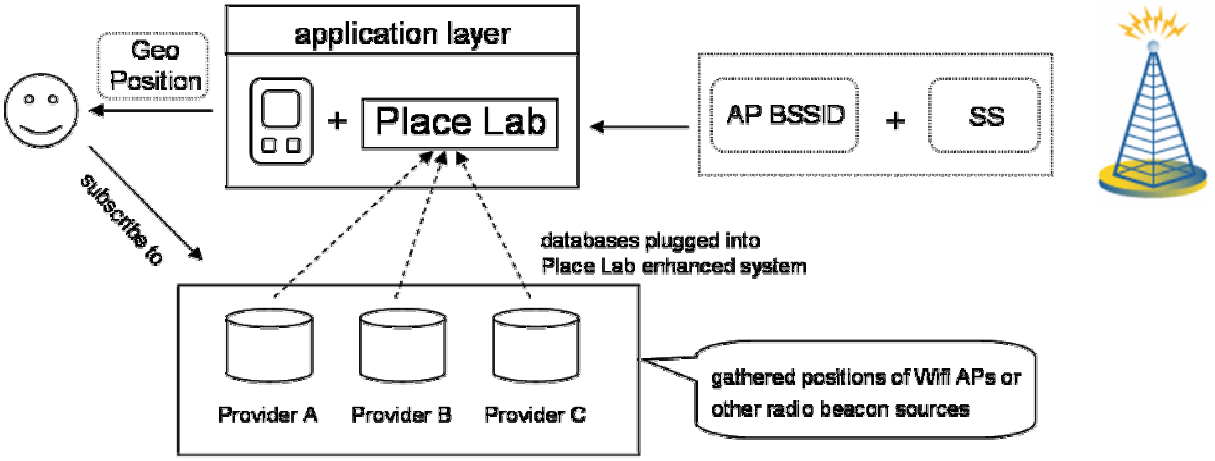


Figure 2 Fundamental principle of Place Lab's privacy mechanism

The Place Lab system consists of three main parts. In Figure 2 these parts are depicted with rectangles. On the upper right side you can see the radio sources sending out beacons. These beacons contain information like the signal strength of the beacon and the BSSID (Basic Service Set Identifier) of the access point, which is unique as mentioned in the previous subsection 2.3.1. This package is received by the mobile device with the Place Lab system installed. Place Lab then converts the received data to information that is used to map the beacon source to a location. For doing so the user must load some location information of known APs in advance. At this point let's take a look at the lower rectangle depicting the databases of the providers. The providers can either be institutional databases or databases of the war-driving community. War-driving will be explained in detail in subsection 3.2. Once the beacon information is gathered Place Lab can estimate the user's position and display it to the user. Instead of displaying the mere information of the current position to the user, it can be forwarded to an application that can use this position. You might think of a client mapping software like a navigation system, or location-enhanced web services (LEWS). One good

example for a LEWS-site is local.google.com [9]. Instead of typing the user's current position into a web form, the location information would be pushed directly to the service.
The picture clearly shows the protection of the user's privacy. The only arrow going out from the user is the one pointing at the databases. So the only interaction within the system is the one with the information providers. All the calculation is done on client side. If the result of these calculations remains there, the user remains anonymous and keeps his privacy.

2.3.3 Databases

The function of a database is to supply the system with a set of positions of beacon sources. After that, the system can map a received signal to the location of the beacon source.
For 802.11 and Bluetooth signals we have

- **Companies, universities and departments** (Figure 3)
  Mostly tens or hundreds of APs, but it requires a format-translation step to use the imported databases, because it is likely that the provided information is fixed in different formats.
- **War-driving community** (see Excursion in subsection 2.2)
  Their databases contain estimated locations for millions of beacon sources.

For GSM signals you can use imported databases of the **FCC** (Federal Communications Commission).

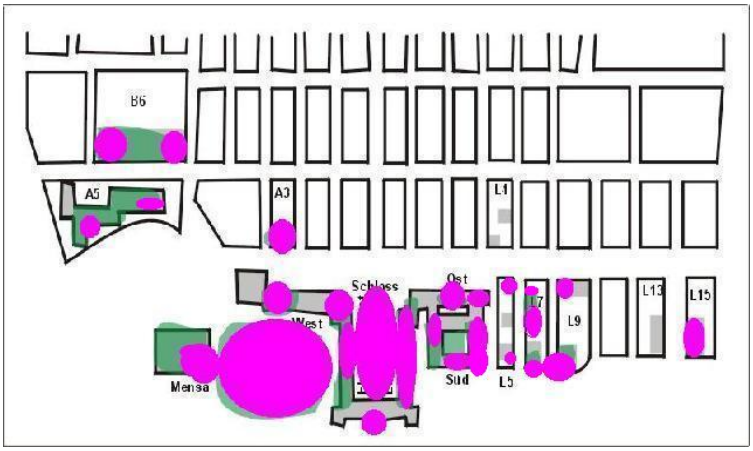


Figure 3 Wifi coverage of the University of Mannheim Campus

3 Results of research and experiments

3.1 Coverage

"Why don't we use GPS for positioning?" This question is worth asking, because Place Lab follows the maxim to use an infrastructure that already exists. Using GPS should be convenient, shouldn't it? For answering this question let's check the coverage of GPS in a real-world testing environment with people in their daily lives: Place Lab selected an immunologist, a home maker and a retail clerk as test subjects, each equipped with a GPS device carrying around in their daily routine. The results are shown in the table below.

Test Subject	**GPS**	
	coverage	avg. gap
Immunologist	12.8%	68 min
Home maker	0.6%	78 min
Retail clerk	0%	171 min
Average	**4.5%**	105 min

Table 1 - GPS coverage and average gap

Obviously GPS is not applicable for a positioning system that should be used in people's daily lives. Coverage of 4.5% is not acceptable at all if you want to have the ability to estimate the current position at *any time*, even if the estimation itself is not perfectly correct.
The other metric in this table is the average gap. This number indicates the time between two usable signals or in other words the average time period that a user would have to wait until positioning is possible. Again, the results show that GPS is not a good choice for having anytime-positioning.

The test subjects were also equipped with a Wireless LAN enabled Laptop and a GSM device, in this case a Nokia 6600 mobile phone. The same metrics were applied and the results are shown in the following table.

Test Subject	**GSM**		**802.11**	
	coverage	avg. gap	coverage	avg. gap
Immunologist	100%	-	87.7%	1.6 min
Home maker	98.7%	2 min	95.8%	1 min
Retail clerk	100%	-	100%	-
Average	**99.6%**	1 min	**94.5%**	1.3 min

Table 2 - GSM and 802.11 coverage and average gap

GSM/802.11 and GPS results differ highly in coverage and average gap. The test subjects have seen almost 100 % coverage of GSM and almost 95% coverage of 802.11. An average gap of about 1 min in each case is acceptable in respect to anytime-positioning. Hence, GSM and 802.11 are adequate technologies for setting up a positioning system for usage in daily situations.

Bluetooth beacons are not distributed enough to gain adequate data for positioning. Fixed Bluetooth stations are not yet deployed in a fair number, so the scarce distribution affects a low usability.

To proof the convenience of 802.11, Place Lab checked the density of 802.11 access points in three different areas around Seattle. The goal was to analyze the impact of different neighborhoods and different landscape scenarios on the coverage and the accuracy of positioning. The first test scenario was Seattle Downtown, which is an urban environment. The density of access points is expected to be high. The second scenario was Ravenna, a residential area and the third scenario was Kirkland, a suburban area, where density is expected to be rather low.

The sample areas and the results of the experiment are shown in Figure 4 [1].

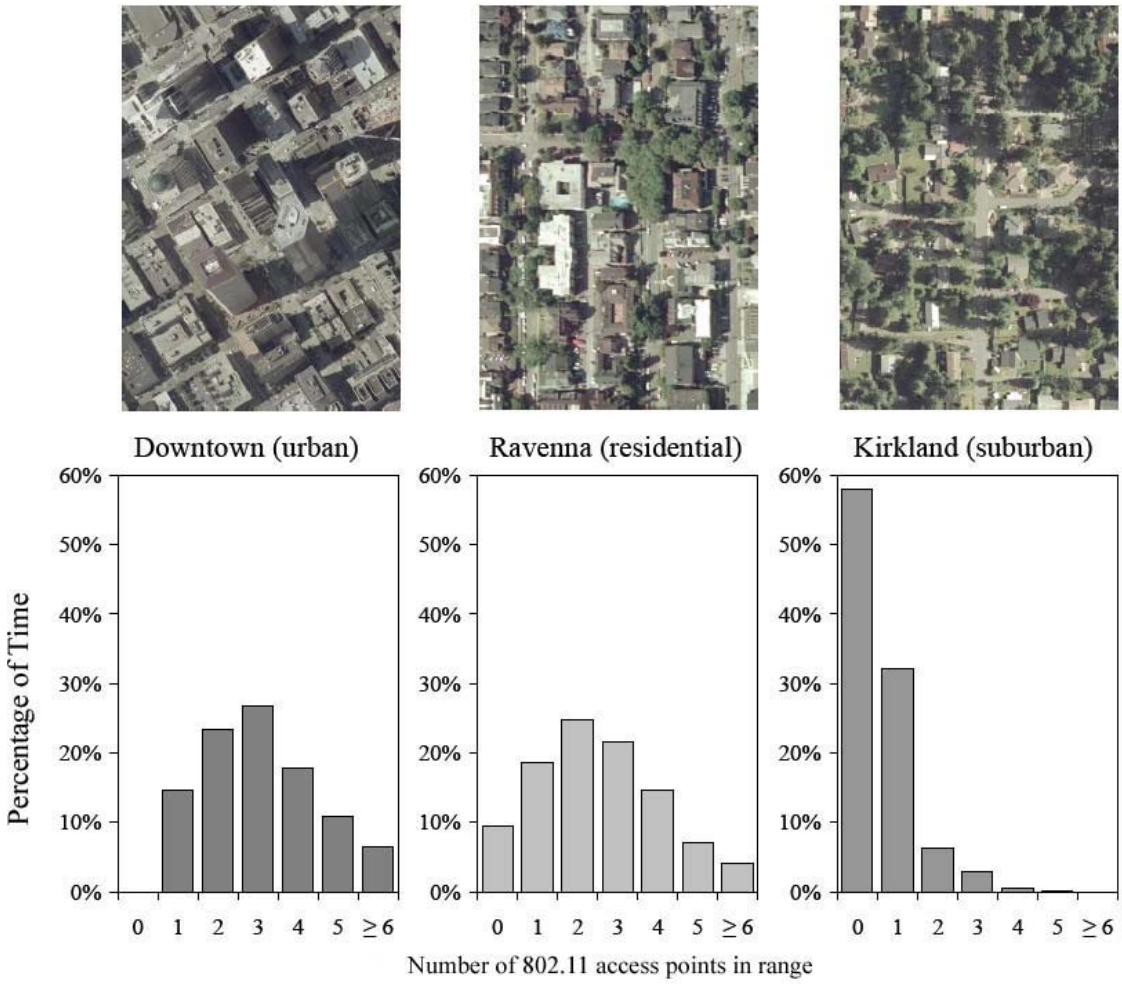


Figure 4 Density of 802.11 Access Points in three different areas around Seattle. The bar diagrams depict the percentage of time while a number of 802.11 access points were in range [1].

Remarkable is the similarity of the first two diagrams. One could have expected a high drop of access points in range in Ravenna, but the dispersion is almost the same, although a light shift to the left is observable. The strong decrease of access points in range in Kirkland is striking since more than 50% of the time tested there, not one access points was seen. If a signal was received, mostly only one beacon was received at the same time. This promises not very good results in accuracy of the positioning when 802.11 is the only beacon technique used.

The following experiment tries to measure density and accuracy in each of the three test scenarios for 802.11, GSM and both at the same time, which is called *sensor fusion*.

	802.11		**GSM**		**802.11 + GSM**	
	accuracy	**coverage**	**accuracy**	**coverage**	**accuracy**	**coverage**
Downtown Seattle (urban)	20.5 m	100.0 %	107.2 m	100.0 %	21.8 m	100.0 %
Ravenna (Residential)	13.5 m	90.6 %	161.4 m	100.0 %	13.4 m	100.0 %
Kirkland (Suburban)	22.6 m	42.0 %	216.2 m	99.7 %	31.3 m	100.0 %

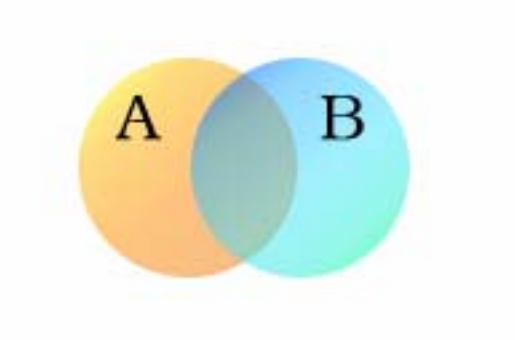
Table 3 - 802.11, GSM and 802.11+GSM accuracy and coverage

To remind the bad GPS coverage of 4.5%, we observed 100.0% coverage in all three scenarios if a combination of 802.11 and GSM was applied. This is a great leap forward because now Place Lab has the proven fact that 802.11 and GSM together are applicable for a convenient positioning system. An accuracy of 13.4 to 31.3 m, which is the error of positioning, is also acceptable for most of the applications.

3.2 Accuracy

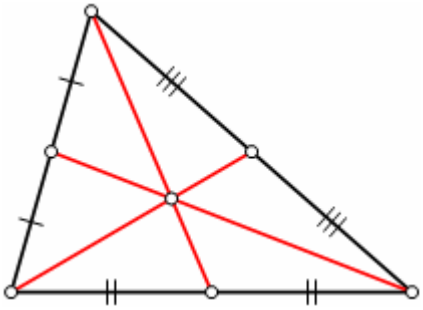
Place Lab doesn't imply a specific way to perform the actual positioning, it rather implements different positioning algorithms that are already known and tested by other institutions. That is why I will not go too much into detail here, although the algorithms that are used by Place Lab should be mentioned and briefly described at least. References are given for some algorithms to provide further and detailed information.

- **Venn Diagram**



This approach is the easiest way to locate a device by using radio signals. The received signals are mapped to locations. The centers of the circles in the diagram depict the observed locations of the beacon sources. The radius of the circles is the maximum signal range of the beacon sources. The intersection of all circles depicts all the possible positions of the device to be tracked.

- **Centroid**



The Centroid algorithm is a very easy approach, too. The algorithm positions the user at the center of all of the access points heard during a scan by computing an average of the estimated positions of each of the heard access points. One modification of the algorithm is to use weighted positions by using the received signal strength as weight to the position.

- **RADAR (Fingerprinting)**

  This algorithm is based on an indoor positioning mechanism proposed by RADAR [5]. The idea is the following: A user receives signals as sets of the access points' IDs and the corresponding signal strengths. These sets of received signals build the fingerprint of the user's current position. Now, the algorithm works in two phases. In the offline phase (training phase) the system has to be trained by observing these fingerprints and assigning them to the position where the fingerprints were received. In the online phase a fingerprint is observed and looked up in the previously built fingerprint database. To be able to compare two fingerprints, RADAR proposed a metric called *NNSS (Nearest Neighbor in Signal Space).* This metric computes the Euclidean Distance between two given sets of signal strengths. The best match of the received sample to a stored fingerprint is declared the best position estimate of the user. Modifications of this algorithm is the so-called *k-nearest-neighbor* approach, which takes not only the best match but the k best matches into account by applying a simple average over them. Based on preliminary experiments with varying values of k, Place Lab discovered that k = 4 provides good accuracy.

- **Particle Filter with Sensor Fusion**

  Sensor Fusion means the combination of different sensing techniques; in this case these are 802.11 and GSM. A particle filter is a probabilistic approximation algorithm that implements a Bayes filter [12]. It represents the location estimate of a user at time *t* using a collection of weighted *particles* p_t^i , w_t^i, *(i = 1,…,n).* Each p_t^i is a distinct hypothesis about the user's current position. "Each particle has a weight w_t^i that represents the likelihood that this hypothesis is true, that is, the probability that the user's device would hear the observed scan if it were indeed at the position of the particle." [13]
  A detailed description of the particle filter algorithm can be found in [13].

With the radio techniques, the proof of concept that 802.11 and GSM are applicable for anywhere and anytime device positioning and the positioning algorithms that should ensure accurate position estimation, Place Lab has almost everything it needs to perform real-time location estimation. The only thing that is left is the data of the fixed radio senders. For simplification Place Lab restricts its test environment to 802.11 beacons since they only want to show the differences of the above mentioned algorithms in accuracy. Large sources for 802.11 access points are the War-Driving databases, that I want to explain shortly in detail. For now, let's just assume a large database that contains a lot of estimated positions of access points and received beacons.

To discover how well you can estimate a position with 802.11 beacons and War-Driving databases, Place Lab arranged their own "War-Drive" in the areas explained in subsection 3.1. Using these data, an accuracy testing with three of the positioning algorithms was conducted [1]. The results are shown in the following table.

Algorithm	**Downtown (meters)**	**Ravenna (meters)**	**Kirkland (meters)**
Centroid	24.4	14.8	37.0
Fingerprint	18.5	15.3	30.0
Particle Filter	18.0	14.4	29.7

Table 4 - Centroid, Fingerprint ad Particle Filter algorithm in three scenarios

The *Fingerprint* and *Particle Filter* approaches result in almost the same positioning errors while *Centroid* as a very simple algorithm performed poorly as expected in most cases. Seeing that a fingerprint approach can almost outperform a sophisticated algorithm like Particle Filter, it should be promising to use fingerprinting in the real system to reduce complexity since mobile devices often lack of computing power and a long battery lifetime.

3.3 War-Driving

Without making a digression, I want to introduce the basic ideas behind War-Driving, the War-Driving community and how Place Lab applies War-Driving to their system and location-aware systems in general.

“War-Driving (also known as LAN Jacking) is searching for Wireless LANs by automobile. It involves using a car and a Wireless LAN equipped computer, such as a laptop or a PDA, to detect the networks. It is also known as "WiLDing" (Wireless Lan Driving), originating in the San Francisco Bay Area with the Bay Area Wireless Users Group (BAWUG). It is similar to using a scanner for radio.” [14]
Many War-Drivers use GPS devices to measure the location of the network found and log it on a website. For better range, antennas are built or bought, and vary from omnidirectional to highly directional. Software for War-Driving is freely available on the Internet, notably, NetStumbler [11] for Windows, KisMac [19] for Macintosh, and Kismet [10] for Linux.

War-Driving is a worldwide activity, with a large community of people. The largest War-Driving database is *WIGLE.net* (www.wigle.net):
“...four and a half million observed networks.” Wed. Nov 9, 2005. Wigle.net is a great contributor to the yearly *WorldWide WarDrives* which are organized War-Driving events (www.worldwidewardrive.org).
War-Driving databases are used to find networks, given a user’s position. People who want to have Internet access wherever they are or hackers who want to use foreign networks in order to fake their identity are interested in such databases. Place Lab uses War-Driving databases in reverse by querying a location, given a set of networks.

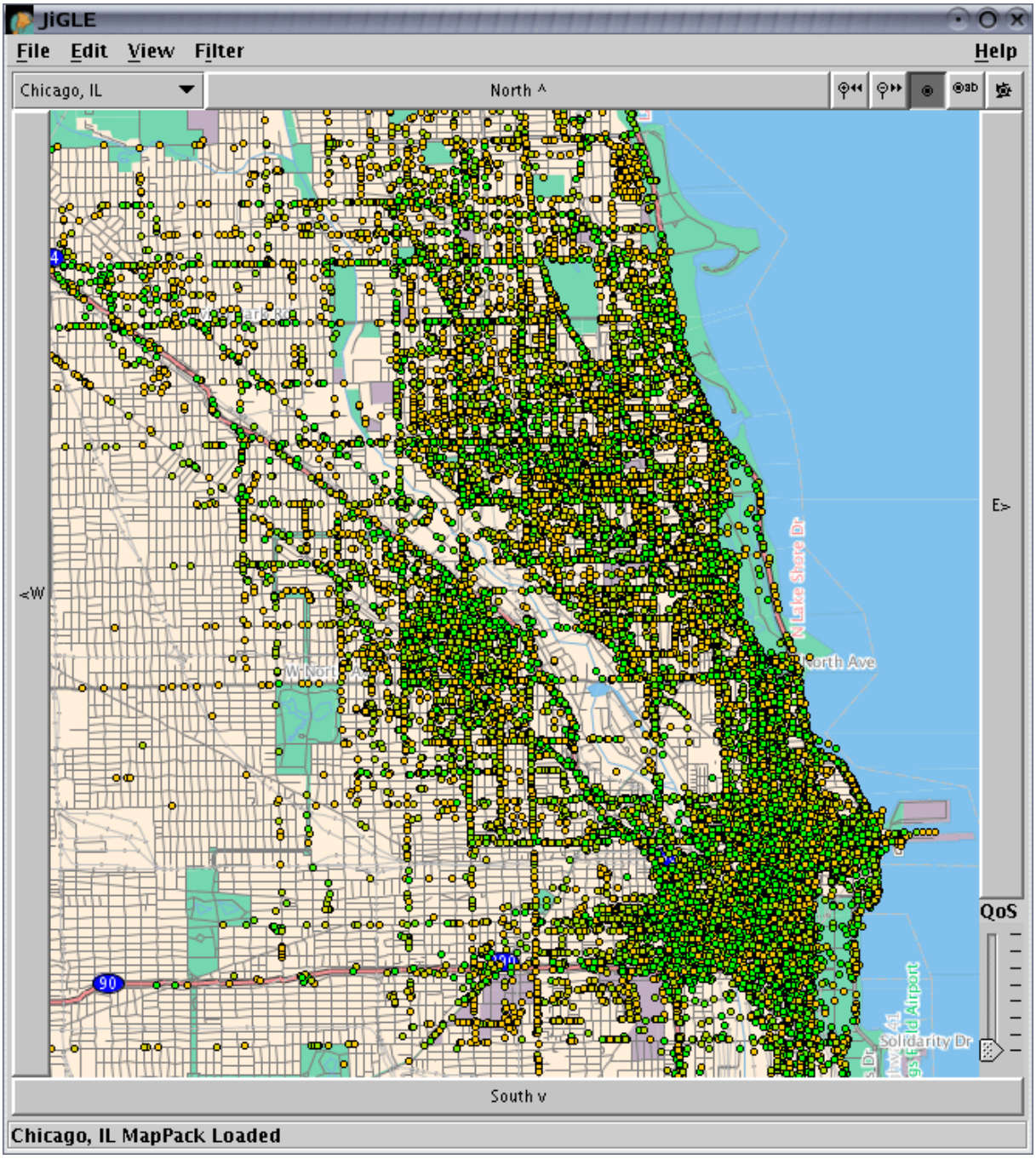


Figure 5 Radio Map of Downtown Chicago, IL (WIGLE.net). One point depicts a received radio signal.

3.4 Self-Mapping

As described in subsection 2.3.3 the system needs databases containing information about locations of beacon sources. To do so, a user has to plug those databases into the system before the system can be used, which is called *calibration step*.
The goal of Self-Mapping is to minimize or even to eliminate explicit system calibration by allowing the system to build the radio map as the system is used.
To build a learning and self-optimizing system, Place Lab needs some initial data, the so-called *seed-set*. Experimental results show that this set does not need to be very large, which promises real-time convenience. Another advantage is that there is no more need for GPS dependence. New beacon sources and never seen networks can be mapped into the system's radio map on-the-fly while already existing beacon sources can be re-mapped and their position estimations can be optimized. No more database updates are necessary, because the system is now self-dependent.

The Self-Mapping approach implements a graph algorithm to map received beacon sources as correct as possible to their coordinates in the real world. The following example and Figure 6 explain the algorithm in detail. Let b_1 and b_2 be two access points that are sending beacons continuously. The stars depict three different positions where the two radio signals of b_1 and b_2 are observed including their signal strengths. Each set of received beacons contains a timestamp *t*, too. For conveying the two beacon sources to coordinates, the algorithm first tries to compute the maximum possible distance between the access points. This is, leaving out the received signal strengths, double the maximum range of each access point. The estimated location would be exactly in the middle between the two access points, because that is the only possible position where both beacons can be received with the maximum range. Considering the signal strengths, this distance will decrease the stronger the signals are, because there is a high correlation between the distance to an access point and the signal strength of the received beacon of this access point. Thus it makes sense to use the set of the strongest received signals to compute the maximum distance between the beacon sources. In Figure 6, position number three is the set with the strongest signals.

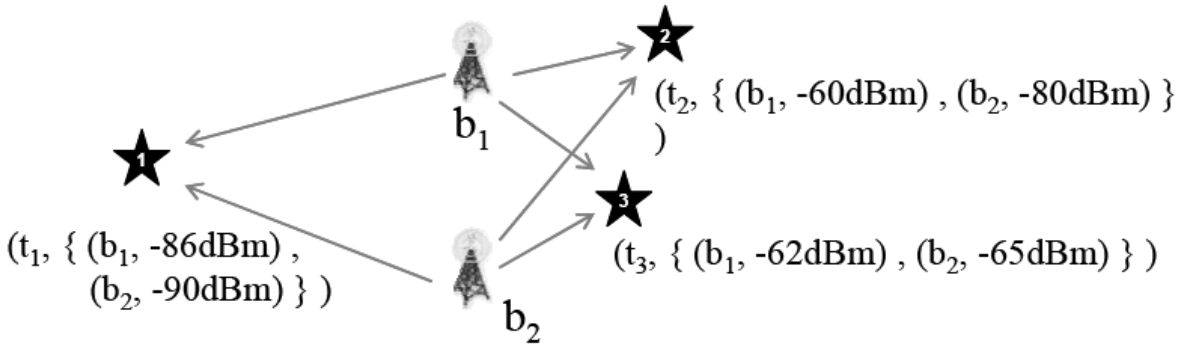


Figure 6 Two APs, three positions with their corresponding received signal strengths

The following lines concerning the distance quantification are an excerpt of *"Self-Mapping in 802.11 Location Systems"* [3]: To quantify the distance, Self-Mapping uses Seidel's model for propagation of signals in the wireless networking band [15]. According to the model, expected signal strength (*ss*) at distance *d* is: $ss = ss_0 - 10 \cdot n \cdot \log_{10}(d/d_0)$ where ss_0 is the signal strength that would be observed in free space at reference distance d_0 from the transmitter. The constant *n* is based on characteristics of the particular radio and the physical environment (density of obstacles, etc), with *n* typically varying between 2 and 5. At 1m, 802.11 access points do not typically return signal strength greater than -32dBm, so we use $d_0 = 1$ and $ss_0 =$ -32dBm. Since we operated in a city with primarily wood and glass structures, and since we can assume nothing about the radios being modeled, we chose n to be on the low side: n = 2.5. Solving for d with these values, we get:

$$ss = -32 - 10 \cdot 2{,}5 \cdot \log_{10}(d/1)$$

$$d = 10^{(-32-ss)/25}$$

This establishes an estimated distance between the user and the observed beacons.

This distance calculation step is performed for every pair of beacons that the user observed during the usage of the system. Once the maximum distance of each beacon pair was computed, the algorithm draws a graph with beacons as nodes and edges carrying the distance between the beacons as weight. If no edge exists between two nodes, one is added with the maximum distance. If there is an existing edge and the user observes a maximum distance that is lower than the current weight, it is replaced by an edge with the lower distance. In the graph below, there are two kinds of nodes, the anchor nodes representing the initial seed-set and non-anchor nodes representing all beacons that are supposed to be mapped (see Figure 7). In order to find a solution for this graph, positions of a non-anchor node are iteratively assigned to its corresponding anchor node, which will certainly violate a large number of edge constraints.

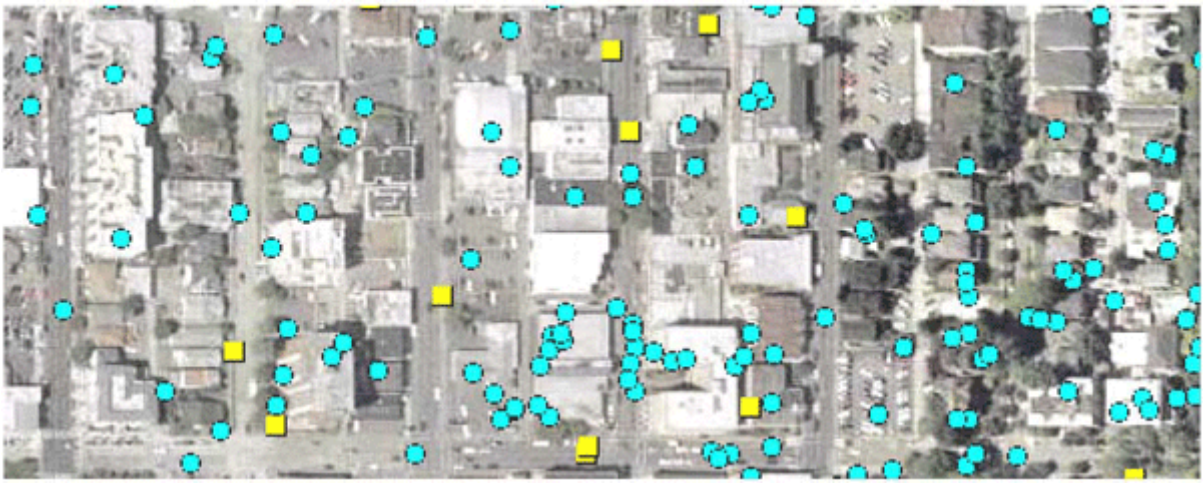


Figure 7 Yellow rectangles depict all anchor nodes; light blue circles depict all non-anchor nodes.

If the distance between two nodes is larger than their shared edge, there is an error of the node that sums up to the graph error with each node error. An error is reduced by repeatedly choosing a random non-anchor node and trying to reduce its error. This is done by trying alternative positions of the node whose error is currently intended to be reduced. If there is a position that reduces the overall graph error, this new position will be assigned to the node. When the algorithm completes, new positions are used to build the radio map.

The errors produced by Self-Mapping compared to errors of other access point placement techniques like War-Driving, Self-Mapping as a self-dependent system performs quite well considering the fact that War-Driving uses GPS to locate the position where the beacons were received. Without GPS, Self-Mapping is able to map a beacon with an average error of 31m according to the table given below. In contrast to that, War-Driving performs slightly better with an error of 26m [3].

Error in AP placement (meters)		
Access Point	**Self-Mapping**	**War-Driving**
`00:09:d7:c4:3c:81`	35	27
`00:09:5b:99:a9:c0`	54	37
`00:04:5a:0e:6e:fc`	24	32
`00:02:3a:9e:a3:d7`	11	14
`00:0f:3d:4f:84:a0`	31	18
Average	**31**	**26**

Table 5 - Self-Mapping and War-Driving error results

Accepting an increased average error in access point placement of 5m and thus a less accurate device positioning, Place Lab gains the advantage of being self-dependent without GPS and to have a system that learns and improves itself during its regular usage.

4 Projects implementing Place Lab

In the following section I want to outline some of the current projects that use Place Lab as positioning system. Location-aware computing is becoming popular at universities as well, that is why the least two examples come from students at University of Washington.
To find a complete list of all projects, please visit the Place Lab Homepage [20].

The first example is **ActiveCampus** [6]. It was developed and tested at University of California, San Diego. The ActiveCampus project aims to provide location-based services for educational networks and understand how such systems are used. Figure 8 shows how ActiveCampus can be used. It allows configuration of ICQ, AIM, MSN, and Yahoo accounts to use Instant Messaging. It also enables the student in its campus life by using its context, e.g. the location.

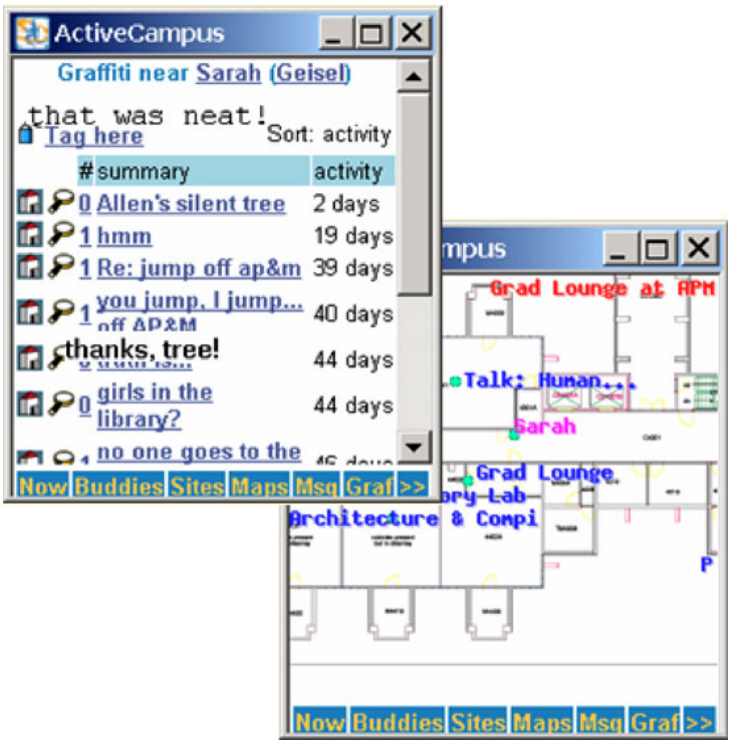


Figure 8 The ActiveCampus project at UCSD

Topiary is a tool created by the Group for User Interface Research at University of California, Berkeley. It was made for prototyping location-enhanced applications that allow designers to quickly design, prototype, and test a location-enhanced application without requiring them to implement the application or deploy a supporting infrastructure, enabling them to get early feedback about their design from real end users. Figure 9 shows how several clients are simulated (Alice, Bob and Carol), as well as some places (Café, Book Store etc.). On the left side you some *Scenarios* (Bob enters Gym etc.) that can be simulated without actually testing these scenarios in real life. Several scenarios can be combined in a storyboard which can be created by clicking on *Storyboard* at the top of the window.

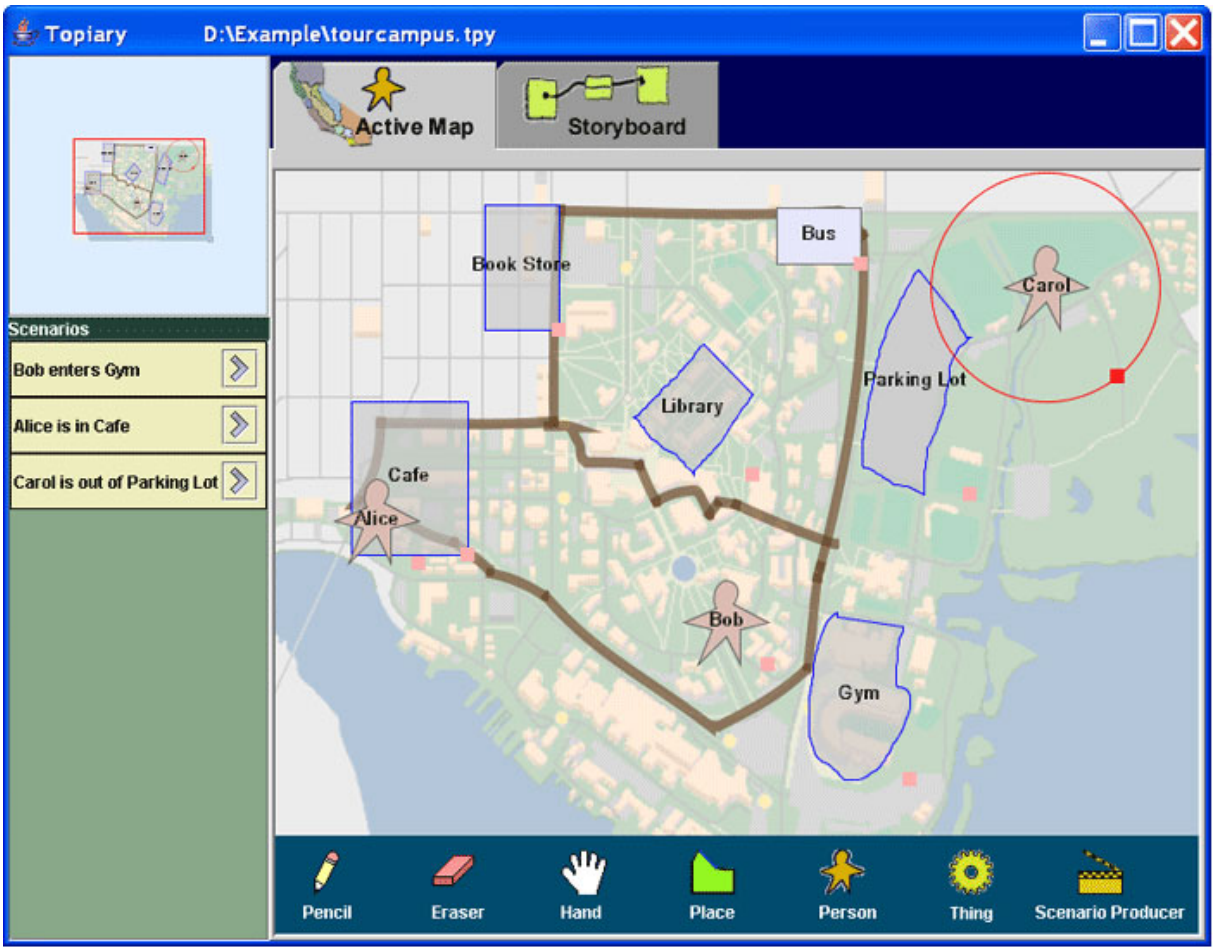


Figure 9 Topiary by the Group for User Interface Research at University of California, Berkeley

Privacy Control for Location-Enhanced IM

“A central obstacle for privacy-observant location systems remains in the difficulty of getting users to fluently understand and precisely control their privacy settings. In this project, we explore potential strategies for overcoming this problem in the context of a specific application: a location-enhanced instant messenger (IM).” [21]

Figure 10 shows how these privacy settings could be realized. The bars on the right side of each user are called *"privacy slider"* and have different colors, “which indicates the number of times that buddy has checked your location (length of color), the type of information that buddy is getting (the color: blue = none, yellow=fuzzed, orange=exact, red=history), and allows the user to easily limit the rate at which that buddy can query (by clicking a limit-spot on the bar and thereby "locking" that level - as has been done for "welbourne")” [21]. Each user can manage its own privacy settings in an intuitive way by clicking on the tab *Privacy*.

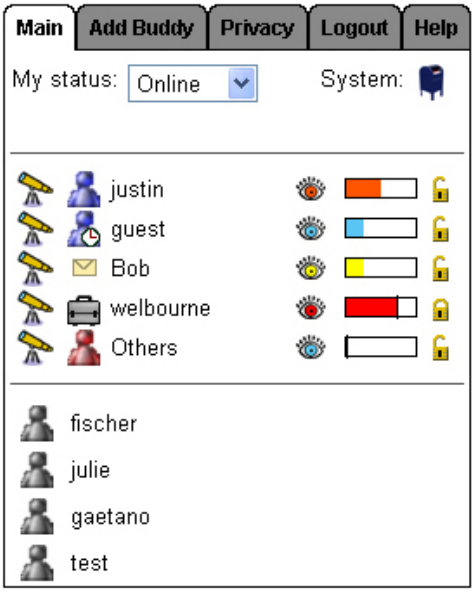


Figure 10 Privacy Control for Location-Enhanced IM, at University of Washington

The Place Lab framework generates geometric positions in terms of longitude and latitude coordinates. However, many applications require a more symbolic notion of locations, as suggested by Hightower et. al. in [7]. The project **Place Extractor** builds an abstraction layer that provides a more symbolic notion of locations from the coordinates generated by the Place Lab framework.

Figure 11 shows graphically the usage of the system. All locations the user has been during usage, say one day, are split into clusters. Each cluster has a different color and a center which is a plus in the screenshot below. Each cluster is assigned a name (Library, Book Store, Café etc.). Once a user comes back to such a place, the system recognizes the place and the assigned name.

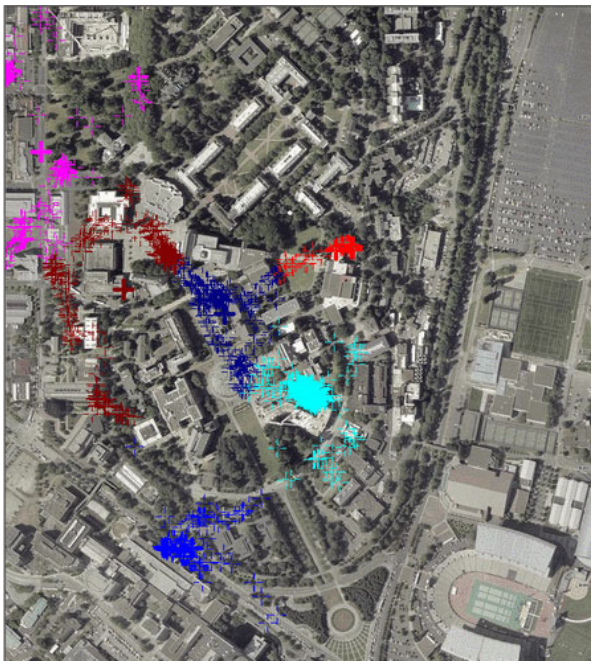


Figure 11 Place Extractor: Translating Coordinates into Places

5 Future Work and Prospects

One way to improve Place Lab is to utilize symbolic place names like “Bank”, “Office” or “at home”. This is for most of the future location-aware applications more convenient than mere geo-coordinates such as (48.43456, -122.45678). Some applications might need altitude information too. Again, a pure geo-coordinate in a longitude and latitude fashion is not applicable for most of them. So Place Lab introduced their “2.5”-dimensions, where the last half dimension is a symbolic name for measuring the altitude. Examples are “parking garage, level B” or “University, A5, 1st floor, part C”.

“Place Lab introduced an algorithm called BeaconPrint that uses WiFi and GSM radio fingerprints collected by someone’s personal mobile device to automatically learn the places they go and then detect when they return to those places. BeaconPrint does not automatically assign names or semantics to places. Rather, it provides the technological foundation to support this task.” [7].

Future developments are important, especially developments of self-controlled systems and algorithms like Self-Mapping. One could find a smarter graph algorithm to reduce the error in the graph. A random pick of alternative access point positions is probably not feasible if there is a lot of data.

Industry will come up with new technologies that must be analyzed for possible usage with Place Lab. Examples for that are the upcoming WiMAX standard (IEEE 802.16 [23]) for metropolitan areas, fixed Bluetooth stations which are currently available in small numbers but not widely distributed, and UMTS [22], which is one of the third-generation (3G) mobile phone technologies, is very interesting in respect to current developments in the mobile multimedia sector. All these beacons can probably be used with Place Lab, but they have to be analyzed and adopted first.

6 Conclusions

Experimental results showed that a combined usage of 802.11 and GSM provides perfect coverage in our daily life, and an accuracy of about 15 – 30 m is sufficient for most of the applications. It also turned out that Bluetooth beacons are not distributed enough to gain adequate data for positioning. Fixed Bluetooth stations are not yet deployed in a fair number, so the scarce distribution affects in a low usability.

Nonetheless, Place Lab will grow steadily due to the fact that the infrastructure will grow. Technical barriers will be lowered, because users and developers don't need to buy new equipment, they already carry all they need for using the system. In the course of that a worldwide user- and developer-community will be formed and will grow, which promises that there will be enough space for new ideas and innovations.

LIST OF REFERENCES

1. LAMARCA, A., CHAWATHE, Y., CONSOLVO, S., HIGHTOWER, J., SMITH, I., SCOTT, J., SOHN, T., HOWARD, J., HUGHES, J., POTTER, F., TABERT, J., POWLEDGE, P., BORRIELLO, G. & SCHILIT, B. Place Lab: Device Positioning Using Radio Beacons in the Wild. *Pervasive 2005*, Munich, 2005.
2. CHENG, Y., CHAWATHE, Y., LAMARCA, A., KRUMM, J. Accuracy Characterization for Metropolitan-scale Wi-Fi Localization. *In Proceedings of Mobisys 2005*.
3. LAMARCA, A., HIGHTOWER, J., SMITH, I., CONSOLVO, S. Self-Mapping in 802.11 Location Systems. *In Proceedings of Ubicomp 2005*, Tokyo, Japan.
4. SCHILIT, B., LAMARCA, A., BORRIELLO, G., GRISWOLD, W., MCDONALD, D., LAZOWSKA, E., BALACHANDRAN, A., HONIG, J. & IVERSON, V. Challenge: Ubiquitous Location-Aware Computing and the Place Lab Initiative. *In Proceedings of WMASH 03*.
5. BAHL, P. & PADMANABHAN, V. RADAR: An In-Building RF-Based User Location and Tracking System. *In Proceedings of IEEE INFOCOM*, pp. 775-784 (2000)
6. GRISWOLD, W. G., SHANAHAN, P., BROWN, S. W., BOYER, R., RATTO, M., SHAPIRO, R. B. & TRUONG, T. M. ActiveCampus - Experiments in Community-Oriented Ubiquitous Computing. *To Appear: IEEE Computer*
7. HIGHTOWER, J., CONSOLVO, S., LAMARCA, A., SMITH, I., HUGHES, J. Learning and Recognizing the Places We Go. *In Proceedings of Ubicomp 2005*, Tokyo, Japan.
8. WANT, R., HOPPER, A., FALCAO, V. & GIBBONS, J. The Active Badge Location System. *ACM Transactions on Information Systems*, 10, 91-102. (1992)
9. Google.com. Google local: `http://local.google.com`
10. Kismet. `http://www.kismetwireless.net`
11. NetStumbler. `http://www.netstumbler.com`
12. FOX, D., HIGHTOWER, J., LIAO, L., SCHULZ, D. & BORRIELLO, G. Bayesian filtering for location estimation. *IEEE Pervasive Computing*, 2(3):24–33, 2003.
13. HIGHTOWER, J. & BORRIELLO, G. Particle filters for location estimation in ubiquitous computing: A case study. *In Proceedings of Ubicomp* 2004.
14. Wikipedia. `http://en.wikipedia.org/wiki/War_Driving`
15. SEIDEL, S. Y. & RAPPORT, T. S. 914 Mhz Path Loss Prediction Model for Indoor Wireless Communications in Multifloored Buildings, *IEEE Transactions on Antennas and Propagation*, 40, 207-217. (1992)
16 IEEE 802.11. `http://grouper.ieee.org/groups/802/11/main.html`
17 Bluetooth. `http://www.bluetooth.com/bluetooth/`
18 GSM Wikipedia. `http://en.wikipedia.org/wiki/GSM`
19 kismac. `http://kismac.binaervarianz.de`
20 Place Lab. `http://www.placelab.org`
21 `http://www.cs.washington.edu/education/courses/590gb/04wi/projects/goshi-welbourne`
22 `http://en.wikipedia.org/wiki/UMTS`
23 `http://www.ieee802.org/16/`