

# LOCATION INFORMATION SYSTEM – A DESCRIPTION OF A POSITIONING MIDDLEWARE SYSTEM

Kimmo Koskinen, Radek Spáčil, Jouni Ikonen  
Lappeenranta University of Technology  
Department of Information Technology  
P.O.Box 20, 53851 Lappeenranta, Finland  
E-mail: kimmo.m.koskinen@lut.fi

## KEYWORDS

Location information, positioning, WLAN, service architecture

## ABSTRACT

The spreading of wireless communication networks has brought a possibility to use the location information of the users in various applications. To make this possible, applications need APIs (Application Programming Interface) for accessing the location information. The location information of a user is private information, which needs to be protected from misuse. For this reason, the user should have complete control over the usage of his own location information. There is a need for middleware systems which gather location information, provide an API for the services and allow users to authorize the use of their location information. This paper presents Location Information System (LIS), a middleware system for location based services in Wireless Local Area Networks (WLAN).

## INTRODUCTION

The development of Location Based Services (LBS) presents a wide application area. This area includes activities such as gathering, categorizing storing, accessing and authorizing the usage of location information. This paper discusses one example implementation of a platform for providing LBS services. The solution provided is not complete system for covering all possible usage cases, but a proof of concept for gathering position information from a wireless local area network and allowing services a limited access to the gathered information.

Positioning system forms the active part of the location service, usually by gathering sensor information and combining this information with a location model. A positioning system itself may consist of sensors and a position calculation system which evaluates sensor data to calculate the position of an object.

Survey and categorization of positioning systems can be found from [Hightower 2001]. Generally current positioning systems are application and/or sensor system dependent. Hightower et. al [Hightower 2002] introduced an idea of location stack, to make position data generating systems and applications independent of each other. Idea is very good, but real implementation is quite complex (of course depending on level of detail the stack provides, e.g. how security and access to data is controlled).

Some of well known positioning systems are:

- Active badge system [Want 2001], where the locatable objects carry an “Active badge”, a small tag-device which transmit an infrared signal with unique code in short intervals. This signal is observed by a network of sensors which are placed in an area where the tag’s are desired to be locatable. A master station polls the sensors for tag sightings and this data is then available for applications.
- Microsoft RADAR [Bahl 2000] and commercial Eka-hau Positioning Engine (EPE) [Ekahau 2005], which both use WLAN (802.11b/g) networks for positioning. The locatable devices are a WLAN network interface cards, which are identified by their hardware address (MAC). Location based services can retrieve the location of a devices from the manager station through a specified APIs.
- Global Positioning System (GPS) differs from previous type of systems as object to be located calculates their own location by themselves. In GPS, the locatable device has a receiver which collects the signal which is transmitted by satellites orbiting the earth. The receiver device can calculate it’s own position from at least three individual satellite signals. Still, the GPS receiver itself does not know the location of other GPS receivers. If the location information of one device needs to be accessible by many other parties, there must be a system distributing this information.

Different positioning systems offer different APIs for the location based applications. Application use these interfaces

to retrieve the location information of the terminals. Depending on the communication network, terminals can be identified with, for example, IP addresses or MAC addresses (e.g. in 802.11b WLAN networks). Identifying different terminals is necessary to be able request the location of a specific terminal. For the users to be able to authorize the usage of their location information, the terminal has to be associated with a user. Then it depends on the policy of the user to allow or disallow the usage of the location information.

There may also be some system specific policies, such as “a user may always get his own location information”, which can be implicitly applied. Also, there may exist policies for emergency situations in which the location information of the user can be given to authorities such as the police [FCC 2005]. The user may also have some straightforward methods to disallow the usage of the location information. For example, if there is a software agent gathering and forwarding sensor data from the terminal of the user, the user can just “switch off” the terminal or the agent. However, when multiple applications can use the location information of a user in various ways, there is a need for centralized management of the usage of location information. Still, forcing a system level access policy is not straightforward.

## CASE WLPR.NET AND RELATED WORK

The wireless network built by Laboratory of Communication Engineering at Lappeenranta University of Technology—WLPR.NET—covers the campus of the Lappeenranta University of Technology and part of the Lappeenranta city. The WLPR.NET is built according to *Lappeenranta model* [Juutilainen 2002], which allows the access network (built with devices conforming to 802.11b/g standards) to be centrally managed (as opposed to *community networks*) and still have an option for the user to choose independently an Internet Service Provider (ISP). This network model emphasizes the possibility to create local services. One type of local service are Location Based Services (LBS). The creation of a unified positioning platform in the local access network will benefit the possible service providers and increase the attractiveness of the local network.

The current cell based positioning system in the WLPR.NET network has minimal requirements from the client devices. No additional software is needed from the client device to be able to use the positioning system. Location data of the terminal devices of the users is gathered from the access points in the network mainly with RADIUS (Remote Authentication Dial In User Service) service.

At the first stage, the location data was stored in a database to which every application had access rights. This however created problems for the management of the application, as they had to be separately registered by the database administrator. The functionality of the applications was tied to the database structure which could be subject to change. Also the users in the network did not have sufficient control over

the usage of their location information. Moreover, this model restricted the possibilities of the applications by allowing only request-response type of operation. If the applications needed to react to changes in location information, they had to poll the database for updates which causes unnecessary traffic. It was seen practical to provide an alternate way to make use of location information in the communication protocol level and to provide a model for creating LBSs.

## Applications for Location Information Systems

Use of location information in different applications has been studied in, for example [Hightower 2001, Hightower 2002], the area of ubiquitous computing. Location information in ubiquitous computing has been used for example in the Aura project at the Carnegie Mellon University [Aura 2005]. An example of context-aware application where location information has been used is the *Portable Help Desk* [Galan 2002] application developed in the Aura project.

Another application of positioning enabled network was demonstrated by Lancaster GUIDE project [Davies 2002], a hand-held computer based tourist guide for visitors to Lancaster. The location information of hand-held computers in a WLAN network was used in creating this system. However, the technical solution depends on non-standard changes of 802.11b network protocols. It is necessary to note that GUIDE project started in time where 802.11 standard was not widespread and therefore special solution was needed.

## APIs for Location Information Systems

General APIs for location systems have been defined by many organizations. The Location Working Group (LOC) [OMG 2005] (formerly known as Location Interoperability Forum, LIF) of the Open Mobile Alliance (OMA) has developed a specification for Mobile Location Protocol (MLP). This protocol is an application level XML (eXtensible Markup Language) protocol for getting the position of mobile stations. The XML messages can be transported by different transport level protocols such as HTTP (Hyper Text Transfer Protocol) or SOAP (Simple Object Access Protocol) [W3C 2005]. The protocol itself is independent of the network technology: the bearer and location derivation technology.

The Parlay Group [Parlay 2005] has defined Open Service Access (OSA) API which consists of 14 parts. This definition enables service providers to use the services offered by mobile networks. Part 6, Mobility Service Capability Feature (SCF), provides a definition for accessing the location information of the user in mobile networks. There exist mappings for at least Java, CORBA (Common Object Request Broker Architecture) IDL (Interface Description Language) and WSDL (Web Service Description Language) of the Mobility SCF.

All the above mentioned API's strive to provide some common way to retrieve location information of users in the

network. The problems is that they are defined primarily for mobile telecommunication networks and not for wireless data transfer networks such as WLAN. Implementing these interfaces for use in this work would be a too cumbersome task. Secondly a common model of authorizing the usage of location information by the services must be taken into account.

## WLPR LOCATION INFORMATION SYSTEM ARCHITECTURE

The architecture of the LIS is a collection of services joined by a database and interfaces for:

- collecting and updating location information
- querying location information
- registration of services and users
- management of user rights and service attributes

### Requirements for the LIS

The API for LIS should be available in many programming languages to not limit possible applications. Also the problem of managing service authorization and user rights must be solved. Open standards have to be used to guarantee free usage of the LIS API. Extensibility allows fast development and the ability to make modifications easily.

Implementation based on XML ensures good extensibility. CORBA [Corba 2005] was one option for the API but it was seen to be too heavy for LIS. The possibility of using SOAP [W3C 2005] for the LIS API was investigated. This lead into studying the Web Services framework [Webservices 2002]. Web Service technologies provide a way to describe the interface of a service with WSDL [WSDL 1.1]. With WSDL, the LBS API can be specified in a platform and programming language independent way. SOAP is a XML -based messaging framework for exchanging XML -messages. The WSDL definition of a service can be mapped to SOAP. SOAP can then be mapped to many transport protocols. HTTP mapping for SOAP was chosen in this work. The use of WSDL and SOAP provided a way to both describe the LIS service interface and to define a protocol for communication between the LIS and LBSs.

Users have to be able to control the usage of their location information. This can be done by specifying access control to the location information. This is problematic to implement in the service API, because this interface is meant only for communication between the LIS and the LBS. This was solved by storing the access policies to LIS. These policies define which services are able to request the location of the user. The user can edit these policies with a HTML based interface. This is done through a secure HTTPS session.

## Overview of the LIS

An overview of the LIS and the WLAN positioning system is presented in Figure 1. The user uses his terminal to interact with the location aware service. The location aware service uses a SOAP interface to query the location information of the users in the WLAN network. As the user is using the service in the WLAN network, the location information of his terminal gets recorded with the use of RADIUS service. WLAN access points are configured to used 802.11 Open Systems authentication. This causes the access points to make authentication requests to the RADIUS service whenever the terminal of the user is activated (i.e. turned on) or when the terminal changes the access point where it is connected to. The cell based location data from RADIUS access reports is stored into a *Location Database* with the help of a *taildaemon* program. The *taildaemon* parses the RADIUS log files and extracts the cell (identified by access point IP address) of the terminals (identified by MAC address) in the network. The LIS then uses the *Location Database* when location information of a user is requested.

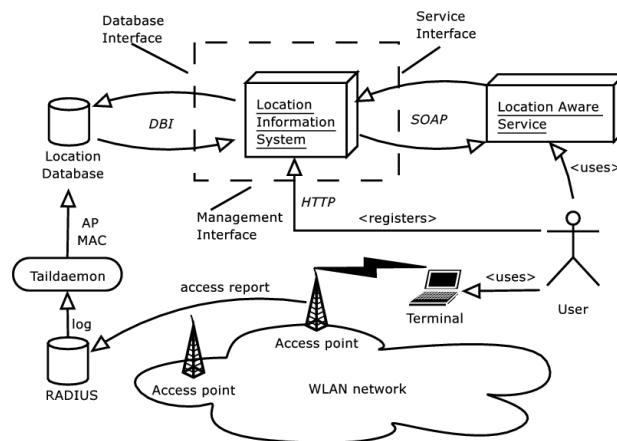


Figure 1. LIS overview

### User and service management interface

In the LIS, both users and LBSs have accounts. Both account types have username/password authentication. Users have a *nickname* and LBSs a *LBS name*. Both account types have attributes related to them. These attributes are managed through a HTML web page interface, which is the *Management Interface* in Figure 1. To get an account, users register to the LIS. For LBSs, the registration is done by the service provider. Both service providers and users can log into the system with their *user name/password* pair to make changes to the account attributes.

User accounts contain the MAC address of the user's WLAN device which is detected when the user registers for the first time and also on subsequent logins. The user can use multiple devices, therefore there is always one *active* device, which has the same location as the user. Users can see a list

of LBSs registered to the LIS and can select which LBSs are able to use the location information of the user. The accounts can also be removed by the corresponding user or service provider who owns the account.

### Service API

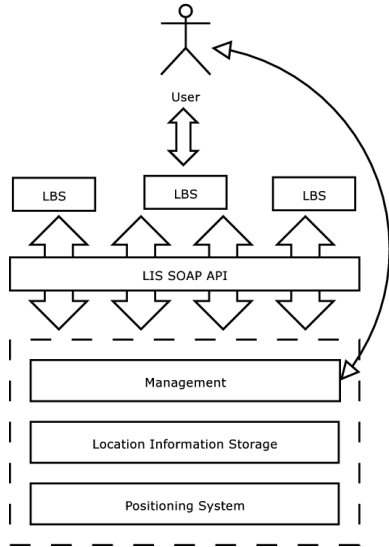


Figure 2. LIS layer architecture

The LIS can be divided into a layer architecture with interfaces between each layer. The layer architecture of LIS is presented on Figure 2.

The lower most level is the *Positioning System*. Positioning system produces location data which is stored in the *Location Information Storage* layer. *Management* layer contains functionality for user and LBS account management. The SOAP API, specified with WSDL, provides two types of operations for the LBSs, *request-response* and *trigger* operations.

The following *request-response* operations provide immediate answers. LBSs can use *GetUserLocation* operation for fetching the location of a specific user. The LBS will get in return a location identifier from which the LBS can fetch more attributes with *GetLocationInfo* operation. *GetLocationList* operation returns a list of all cell id's present in the cell positioning system.

Every operation specifies the *LBS name* and the *password* in the request. When the operation is executed, the authenticity of the LBS is checked first. After this the operation itself is authorized. Figure 3 presents the authorization procedure. Currently, the *GetUserLocation* specifies also *caller* parameter, person who is making the positioning request. However, user-to-user authorization is not implemented yet and this parameter is not used. The current functions and their parameters are defined in Table 1.

*Trigger* operations provide the LBS a conditional way of reacting to the change in location information. *SetTrigger*

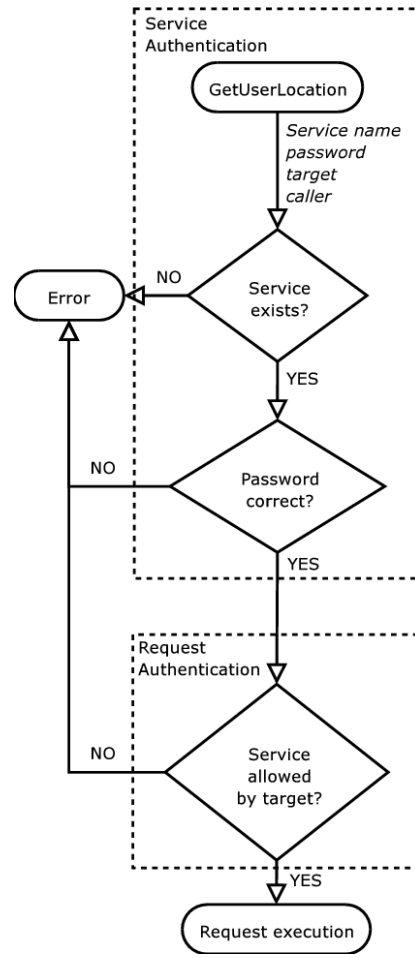


Figure 3. Request authorization

operation is used to place a trigger on a specified user. The LBS will get in return an identification of the trigger. Whenever the user moves, a SOAP callback function *FireTrigger* is executed by the LIS. This callback function is implemented by the LBS. *FireTrigger* function has the following parameters: new location of the user and the identification of the trigger which had been fired. *DeleteTrigger* can be used by the LBS to remove the triggers which have been set.

The trigger notification should be authorized before information is actually sent to the LBS. This is because the user targeted by the trigger might have disallowed the LBS to use his location information after successful *SetTrigger* operation.

The triggers are implemented in a PostgreSQL database with PL/pgSQL. PL/pgSQL is a loadable procedural language for the PostgreSQL database system. It allows to extend the database functionality with SQL like procedural language. The functionality of the trigger mechanism is presented on Figure 4.

| Function                 | Parameters                                  |
|--------------------------|---------------------------------------------|
| request-response methods |                                             |
| GetUserLocation          | target,caller,service, password             |
| GetLocationInfo          | location_id,location type, service,password |
| GetLocationList          | service,password                            |
| trigger methods          |                                             |
| SetTrigger               | target,URI,proxy,service, password          |
| DeleteTrigger            | trigger_id,service,password                 |
| FireTrigger              | trigger_id,location                         |

Table 1. LIS API methods

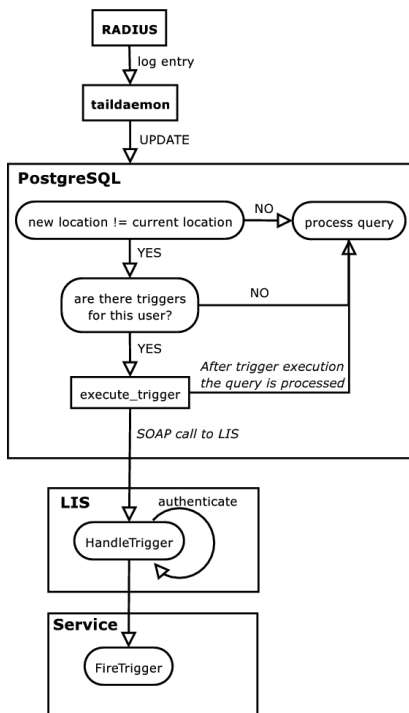


Figure 4. Trigger functionality

## LOCATION INFORMATION SYSTEM APPLICATIONS

Currently there exists one application which uses the LIS system. This is the Floating Note application described in [Multaharju 2004]. The idea in this application is to present a new way to use the location information provided in a WLAN network in the communication between the users of the network. Users can post *notes* in their current location on a forum in which message threads are arranged by the cell locations of the network. Different locations contain different threads of messages.

The floating note uses the *request-response* interface provided by LIS for fetching location information of the user. The user has an account in the Floating Note application and this account contains the user name of the user's LIS account.

When the Floating Note requires to know the location of the user, the SOAP request is authorized based on the allowance rules of the targeted user.

Other possible applications would include tracking applications or context aware applications which would react to the change in the location information of the user. For example, a notification application where a user could set a condition such as "remind me to borrow a book from library when leaving from work". The notification would contain the work area as one parameter, and when the user leaves the work area, a notification message is displayed to the user.

## DISCUSSION

The LIS layer architecture gives a possibility to combine location data from different positioning systems. This would include at least GPS data. Applications could decide on what single positioning system to use but combining data from different sources could also be beneficial. This leads into the discussion of combining the different reference systems used by different positioning systems.

Positioning systems use a reference model in which to express location information. These *coordinate reference systems* may vary from positioning system to another. For example, cell positioning systems provide a discrete one-dimensional reference space (a list of cell id's) for the location data. Other positioning system may use multidimensional, continuous coordinate reference systems. An example of this kind of system is GPS (Global Positioning System) which can offer location information expressed in *latitude*, *longitude* and *altitude*. There exists also "hybrid" positioning systems which offer a mixed discrete and continuous presentation of location information. For example Ekahau EPE [Ekahau 2005] presents location information with *floor-maps* (discrete set of maps) and continuous *x* and *y* coordinates inside the floor-maps.

Despite of the different reference models, mappings from one reference model to another can be established. Discrete cell locations can be mapped to a continuous reference model by defining cell coverage areas in the continuous reference model coordinate space. Mappings from multidimensional reference models to another can also be established. It should be noted that coordinate transformation operations are usual in Geographic Information Systems (GIS). Coordinate transformation operations are needed because reference systems may differ, for example, from country to country.

As long as applications are using the coordinate system of only one positioning system, they can not benefit from the possibilities offered by other positioning systems. This differentiates the applications by the use of a particular positioning system. There exists a need to have a uniform way to refer to location data. A location model with a computable location identifier has been presented in [Changhao 2002]. The authors define a location identifier (Aura Location Identifier, ALI) which is used for referring to location data. In ALI different location representations (coordinate based, hi-

erarchical) are combined and expressed with a URI (Universal Resource Identifier) -like presentation. This method allows to refer to the location data produced by different positioning systems in a uniformed way. It also allows to execute operations for different ALIs, such as “what is the distance between two ALI references”.

A method of combining location data, which is described in [Angermannr 2001], uses probability density functions of each location from different sensing systems to calculate resulting position. By super-positioning data from several sensors it is possible to reach higher accuracy of the resulting location.

OpenGIS has taken a different approach for the combination of different coordination reference systems by defining a coordinate transformation service [Opengis 2005]. This service can be used to conduct coordinate transformation calculation and thus to ease the task of the location application.

Another item of development would be more complex triggers. Some services would benefit from the possibility to define area targets for the triggers which would fire when a user enters or leaves the area. Eventually the trigger mechanism could contain a logic execution environment which would allow to construct more complex rules for firing the triggers.

The procedure currently allows only coarse selection of which service is allowed to use the location information of specific users in the operations provided by LIS. Future development would include specification and implementation of user-to-user authorization. This would rely on the service to provide the correct caller and target pair in the operations (for example, “User A wants to know the location of user B”). This is needed for handling the user-to-user authorization policies in the LIS (as opposed to letting the LBS itself create user-to-user policies and decide whether “user A is allowed to know the location of user B”). However, this issue can be problematic because there is a need to form trust relationships between users, LBSs and the LIS. A system for providing location policies has been presented in [Hengartner 2003]. The authors discuss the problems on defining location policies and an implementation based on public key infrastructure.

## CONCLUSION

This paper presented a platform for creating Location Based Services (LBS). The LIS platform provides an interface for accessing the location information of users in an open WLAN network. The LIS provides also a way for the user to control the usage of their location information. Division to several independent modules with pre-defined interfaces ensures good modularity and scalability. LIS uses a cell based WLAN positioning system but the architecture allows to acquire location information from other positioning systems.

The API provides two different methods for the services to access the location data. *Request-response* and *trigger* methods. Request-response method provides immediate answers

for the queries. Trigger method provides the service a way to react to the change in location information (event based).

Privacy is ensured by service and user rights management system, both services and users have accounts in the LIS. Management interface allows the users to define which service can access their location information. On protocol level, every method call is authenticated by service name and password.

It is necessary to say the current implementation is not suitable for applications which need very small response times. At this stage of development, the emphasis lays on modular architecture, rather than speed.

## References

- [Hightower 2001] Jeffrey Hightower, Gaetano Borriello, “Location Systems for Ubiquitous Computing”, *Computer*, August 2001, Vol. 34, no.8, pp.56-66.
- [Hightower 2002] Jeffrey Hightower, Barry Brummit, Gaetano Borriello, “The Location Stack: A Layered Model for Location in Ubiquitous Computing”, *Proceedings of the 4<sup>th</sup> IEEE Workshop on Mobile Computing Systems & Applications*, Callicoon, New York, June 2002, pp. 22-28.
- [Want 2001] Want, R., Hopper, A., Falcao, V., Gibbons, J., “The Active Badge Location System”, *ACM Transactions on Information Systems*, January 1992, Vol. 10, No. 1, pp 91-102.
- [Bahl 2000] P. Bahl and V. N. Padmanabhan, “RADAR: An In-Building RF-Based User Location and Tracking System”, *Proceedings of IEEE Infocom 2000*, Tel-Aviv, Israel, March 2000, pp. 775-784.
- [Ekahau 2005] Ekahau, *Ekahau website*, <http://www.ekahau.com>, visited: February 13, 2005
- [FCC 2005] Federal Communications Commission, *FCC: Enhanced 911 - Wireless Services*, <http://www.fcc.gov/911/enhanced/>, visited: February 14, 2005
- [Juutilainen 2002] Juutilainen, M., Ikonen, J. and Porras, J., “Evaluation of a Next Generation Public Wireless Multi-ISP Network”, *Proceedings of 27<sup>th</sup> conference on Local Computer Networks*, IEEE, Tampa, Florida, 6-8 November 2002, pp 405-414.
- [Aura 2005] Carnegie Mellon University, *Project Aura, Distraction-free Ubiquitous Computing*, [http://www-2.cs.cmu.edu/~sim\\$aura](http://www-2.cs.cmu.edu/~sim$aura), visited: February 13, 2005
- [Galan 2002] Garlan, D., Siewiorek, D., Smailagic, A., Steenkiste, P., *Project Aura: Toward Distraction-Free Pervasive Computing*, *IEEE Pervasive Computing, April-June 2002*, [http://www-2.cs.cmu.edu/~sim\\$aura/docdir/aura-pervasive02.pdf](http://www-2.cs.cmu.edu/~sim$aura/docdir/aura-pervasive02.pdf), visited: February 13, 2005

- [Davies 2002] Davies, N., Cheverst, K., Friday, A., Mitchel, K., "Future Wireless Applications For A Networked City: Services for Visitors and Residents", *IEEE Wireless Communications*, February 2002, Vol.9, Issue 1, pp 8-16.
- [OMG 2005] Open Mobile Alliance, *Location Working Group Home Page*, [http://www.openmobilealliance.org/tech/wg\\_committees/loc.html](http://www.openmobilealliance.org/tech/wg_committees/loc.html), visited: February 13, 2005
- [Parlay 2005] The Parlay group, *Home page*, <http://www.parlay.org>, visited: February 13, 2005
- [Corba 2005] Object Management Group, *OMG's CORBA Website*, <http://www.corba.org>, visited: February 13, 2005
- [W3C 2005] W3C, *SOAP Version 1.2 Part 1: Messaging Framework*, <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>, visited: February 13, 2005
- [Webservices 2002] W3C, *Web Services Activity*, <http://www.w3.org/2002/ws>, visited: February 13, 2005
- [WSDL 1.1] W3C, *Web Services Description Language (WSDL) 1.1*, <http://www.w3.org/TR/wsdl>, visited: February 13, 2005
- [Multaharju 2004] Multaharju, M., Koskinen, K., Ikonen, J., "Floating Note—A Location Based Messaging Application", *Proceedings of 2nd Workshop on Applications of Wireless Communications*, Lappeenranta, Finland, August 2004, pp. 9-16.
- [Changhao 2002] Changhao, J., Steenkiste, P., "A Hybrid Location Model with Computable Location Identifier for Ubiquitous Computing", *UbiComp'02*, Goteborg, Sweden, 2002.
- [Angermannr 2001] Angermann, M., Kammann, J., Robertson, P., Steinga, A., Strang, T., "Software representation for heterogeneous location data sources within a probabilistic framework", *International Symposium on Location Based Services for Cellular Users*, Locellus, February 2001, pp. 107-118.
- [Opengis 2005] OpenGIS consortium, *Web Coordinate Transformation Service*, <http://www.opengis.org/docs/02-061r1.pdf>, visited: February 13, 2005
- [Hengartner 2003] Hengartner, U., and Steenkiste, P., "Protecting Access to People Location Information", *Proceedings of First International Conference on Security in Pervasive Computing*, SPC 2003, Boppard, Germany March 2003.