Location Management in Pervasive Systems

Jadwiga Indulska and Peter Sutton

School of Information Technology and Electrical Engineering
The University of Queensland
Brisbane, Queensland 4072.

{jaga, peters}@itee.uq.edu.au

Abstract

Pervasive systems require context awareness in order to be able to provide "anywhere, anytime" computing to mobile users. One type of context information is location information: pervasive systems require knowledge of the location of users, devices and services. This paper reviews location management for pervasive systems. The review is based on our experience in the development of a location management system. Issues discussed include the types of location sensors (which are categorised as physical, virtual or logical); the processing of location information; and an architecture for a location management system. Three issues of ongoing importance are discussed in greater detail. These are location conflict resolution (i.e., how can the location of an entity be determined when multiple sensors provide conflicting information); the security and privacy of location information; and scalability issues in location management.

Keywords: pervasive systems, location-aware applications, context-aware applications, location sensors

1 Introduction

The growth in mobile computing has created increasing interest in utilising location information in mobile computing applications. There are already many approaches to define location information, to track the location of users and devices, to manage such location information, and to utilise location information to support mobile users.

Location management systems which gather and manage location information can be used for a variety of purposes and their complexity depends upon their purpose. Such systems are often used to provide users with information which is specific to user location (e.g. tourist guides) as evidenced in the work of Cheverst, Davies, Mitchell, Friday and Efstratiou (2000) or to support the delivery of personalised and location-sensitive information as in the BlueLocator project described by Chen, Chen, Ding, Rao and Liu (2002). Henricksen, Indulska and Rakotonirainy (2002) showed that location information can also be used as one of the attributes of complex context information which supports pervasive systems.

Recently, we have observed an increase in user demand for pervasive systems which provide "anywhere, anytime" computing: users often want access to the same computing applications and information on their workstation in the office, on their computing devices at home or on a variety of mobile devices when they are on the move. Pervasive systems need to be able to deal with the mobility of users and their devices and also, in the future, with users who may want to change their computing device whilst running some computing applications. For example, a user may change from using a PDA to using a workstation when the user moves to an area where this is possible. A pervasive computing infrastructure should allow users to move their computational tasks easily from one computing environment to another and should allow them to take advantage of the capabilities and resources of their current environment.

Pervasive systems have to be *context-aware*, i.e., aware of the state of the computing environment and also of the requirements and current state of computing applications. One type of context information is *location* information (e.g., the location of users, devices and services) which needs to be gathered from a variety of location sources, or *sensors*. The location information can be *physical* (e.g., location provided by a GPS device and a variety of other physical sensors), *virtual* (e.g., information provided by a calendar application), or *logical* (location information inferred from information provided by a physical or virtual sensor).

As pervasive systems are very complex and can use location information from a variety of sources and for a variety of purposes, gathering and managing location information in pervasive systems is a complex task. There are a variety of issues which need to be addressed in such systems, including: (i) types of location information (many sources and kinds of location information), (ii) location resolution and associated errors, (iii) resolution of conflicting location information, (iv) location information access and privacy (who can access location information and how can it be used), (v) architectures of location management systems, and (vi) integration of location information management with general context management in pervasive systems (cooperation of location management systems with the pervasive systems infrastructure) in a way which ensures scalability of the whole system.

In this paper we present a discussion of location management in pervasive systems. We describe the

requirements for such systems, characterise current solutions and show the issues which are yet to be properly addressed. The discussion presented in this paper is a result of a lesson we have learned whilst building a location management system and integrating it with a pervasive system infrastructure.

The structure of this paper is as follows. Section 2 characterises a variety of sources of location information. Section 3 presents an overview of issues related to processing location information like heterogeneity of location representations, architectures of location management systems, interfaces of location management systems and briefly describes problems which are encountered when location information is processed. The three most complex of these problems are discussed in the following sections 4, 5 and 6 which describe conflict resolution, access and privacy, and scalability issues, respectively. Section 7 concludes the paper.

2 Sources of Location Information

There are many sources of location information. We will consider the term *sensor* to mean a system which provides information. Location sensors can be *physical*, *virtual* or *logical*.

Physical location sensors provide information about the position of a *physical device*. Virtual location sensors extract location information from virtual space, i.e. software applications, operating systems and networks. Logical location sensors use information and events from physical or virtual sensors to *infer* physical location information.

We will use the term *entity* to refer to the person or piece of equipment whose location we are interested in. Physical, virtual and logical location sensors are described in more detail in the sections below.

2.1 Physical Location Sensors

Physical location sensors come in many shapes and sizes and use many different techniques for determining position. Physical sensor systems usually have two components which we will term *devices* and *infrastructure*. There is also an associated *communication medium* by which the devices and infrastructure communicate.

A *device* is the equipment associated with an entity. In some cases the device itself may be the entity of interest (e.g. we may be interested in where a mobile phone is rather than where the owner of the phone is) and in other cases the "device" may be a characteristic of the entity (e.g. biometric data). Examples of devices are GPS receivers, mobile phones, passive and active badges, people's faces, magnetic-stripe cards and even simple barcodes.

The *sensor infrastructure* is that equipment which needs to be installed and usually remains in a fixed position. Examples of infrastructure corresponding to the devices above are GPS satellites[1], mobile phone towers, badge proximity detectors, cameras, magnetic card readers and barcode readers. Infrastructure also includes supporting software which may be needed to manage the system and provide location information using some data format. Such software can be termed the *sensor agent*.

The *communication medium* is the mechanism by which the device and infrastructure communicate. This communication can be *unidirectional* or *bidirectional* and, if bidirectional, different media may be used for each direction. Communication may occur by transferring information via the medium (using some sort of modulation), or it may be that the presence or absence of the medium conveys information,[2] e.g. ultrasonic pulses. For the first three examples above, communication is via radio-frequency (RF) electromagnetic waves: GPS uses frequencies around 1575.42MHz and 1227.6MHz; GSM mobile phones use frequencies around 900/1800/1900 MHz (depending on the country or system); and badge proximity detectors are usually based on frequencies in one of the unlicensed ISM (Industrial, Scientific and Medical) frequency bands (e.g. 902 to 928MHz). Cameras receive visible light from their "devices" (faces); magnetic-stripe cards use data stored magnetically; and barcodes are usually read with visible or infra-red lasers. Other communication media include ultrasound, infrared light and, in the case of contact devices, electrical signals. It should also be noted that communication may occur *within* the infrastructure (e.g. between "base stations") and that this communication may use a different medium.

The sections below present a taxonomy of physical sensor technologies. An alternative taxonomy is presented by Hightower and Borriello (2001). Many of the categories presented in their work are subsumed into the four categories presented here.

2.1.1 Proximity vs. Position

Physical location sensors can provide either *position* or *proximity* information. *Position* sensors attempt to provide the coordinates of an entity (or more usually, a *device*) relative to some coordinate system. The coordinate system may be fixed and global (e.g., the latitude, longitude and altitude reported by a GPS receiver); or it may be mobile and local (e.g., "3 metres to my right"). Different sensors will have different *resolutions* and associated errors. These may range from centimetres (e.g. the ultrasound positioning of the Active Bat system described by Harter, Hopper, Steggles, Ward, and Webster (1999)) to tens of metres (e.g. raw GPS). Different sensors will also have different *ranges* over which they operate, ranging from zero (e.g. contact sensors and card readers) to global (e.g. GPS).

Proximity sensors locate an entity or device as being within a region but can not position the device within that region. Region sizes may range from tens of centimetres (e.g. RFID passive tags) to tens of metres (e.g. Bluetooth and 802.11 cells) to hundreds of metres (e.g. mobile phone cells).

[1] GPS satellites are not "fixed" in position, but they can be considered infrastructure.

[2] This is essentially an extreme form of amplitude modulation.

Proximity sensors with overlapping detection regions can form the basis of position sensors. Position information can be generated via triangulation or trilateration. The Active Bat system uses this approach with ultrasound time-of-flight. Many commercial RF-based real-time location systems (RTLS) use this approach with RF signal strength (e.g. WhereTags by WhereNet Corp. (2002) and SpiderTags provided by Wireless Mountain Lab. Inc. (2002)).

2.1.2 Line of Sight

A number of sensor technologies require there to be clear line-of-sight between a device and its associated infrastructure. Examples of devices requiring line-of-sight include GPS devices (clear line of sight is required from the receiver to multiple satellites) and most ultrasound or infrared based devices, e.g. Active Badges (as described by Want, Hopper, Falcão and Gibbons (1992)), and Active Bats. Some of these devices may work without clear line of sight, e.g. by reflecting signals off walls, but performance can be degraded. Communication, and therefore location sensing, is generally impossible if the device is completely obscured. For a device carried on or by a person, this is a particularly important consideration. It may be necessary for a device to be visible externally for the sensor to work, for example, the device may need to be pinned to clothing, or worn around the neck. Such requirements may prevent uptake of a device compared with one which can be carried in a pocket.

Systems built upon communication media which can penetrate clothing, people, and walls (e.g. various parts of the electromagnetic spectrum) have the advantage of fewer constraints on device and infrastructure placement. The disadvantage of these is the loss of information provided by natural barriers, e.g., because signals can travel through walls it may not be possible to determine which room a device is in.

2.1.3 Complexity Trade-off

There is a trade-off between the *complexity* of individual devices and the complexity of the infrastructure. Devices can range from simple, completely passive RFID tags at the low complexity end, to mobile phones or PDAs combined with GPS receivers at the high complexity end of the spectrum. Infrastructure component complexity can range from a simple barcode reader to a complex collection of directional antennae attached to a processing unit. Infrastructure and communication complexity usually increases as the required coverage range increases or as the required resolution distance decreases.

By moving complexity into the device, the infrastructure can often be made simpler. For example, the Cricket location system shown by Priyantha, Chakraborty and Balakrishnan (2000) allows a simpler infrastructure (multiple uncoordinated base-stations or *beacons*) by moving the position computation into the device. Increasing device complexity, however, usually comes at the expense of increased device power consumption. For devices associated with people, this can reduce convenience through larger batteries (and hence larger device size) or more frequent charging or replacement of batteries.

The trade-off to be made will usually be cost-based and will depend upon the application. If the system is tracking many devices over a limited range, one might choose to have simple devices and a more complex but limited infrastructure. On the other hand, if the system is tracking few devices over a wide area, one might choose the opposite.

2.1.4 Identification

Many sensor systems incorporate the concept of identifying particular devices via some unique ID and centrally determining and monitoring the location of that device. This is ideal for inventory control and similar applications, but when the tracked entity is a person it raises privacy and ethical issues (discussed further in Section 5). Some systems compute location information on the device itself (e.g. GPS and Cricket), with Cricket allowing users to publish their location if they wish.

There are many physical sensors which can provide non-identifying information which can be used to generate or infer useful location information. Examples include the detection of movement in a room and door openings and closings. In a sole-occupant office, such information might be used to infer the location of the occupant. Such logical-sensors will be discussed further below.

2.2 Virtual Location Sensors

Events in virtual space can also provide location information. The sensors of such events are software processes rather than physical devices and infrastructure. Such sensor processes can monitor application events, operating system events or network events.

2.2.1 Application Event Sensors

Various software applications often hold information that can give location information about particular people or equipment. Examples include a networked calendar system which might list a person's appointments (with associated locations); a travel-booking system; even email may contain location information (although it may be difficult to extract automatically). An *application event sensor* can be considered a software process or module that is able to extract location information from some software application[3].

2.2.2 Operating System Event Sensors

A computer's operating system is aware of the activities occurring on the computer and is usually aware of the user(s) using the computer. Such virtual information can be used to provide physical location information. For example, console keyboard or mouse use by a logged-in user can, in most cases, be assumed to indicate that the user is using the computer. Similarly, information about remote logins and file server use can identify the computer which a particular person is using. This information, coupled with knowledge of computer locations, can be used to infer the location of

[3] The software application is *not* one associated with a physical location sensor.

users. If the computer is capable of being mobile (e.g. a laptop or PDA), the computer's location may be able to be determined by network event sensors, as described below.

Operating system event sensor processes may be run by any user (e.g., a sensor based on the UNIX "finger" command) or they may require special permission (e.g., administrative privileges or kernel modifications may be required to monitor keyboard activity).

2.2.3 Network Event sensors

A computer's IP address is a virtual location – a place on the network at which the computer can be found. An IP address is made up of a network number (and possibly a subnet number) along with a number which identifies the host on the network or subnet. Networks or subnets often correspond to particular physical locations; for example, an organisation may have an IPv4 class B network for a campus which is broken down into subnets for each building.

A mobile computing device may use a particular IP address on one network, and another address on a second network. Determination of which address is active provides a virtual location for the device, which, when coupled with additional information, can provide a physical location. Such additional information may include, for example, the fact that a particular person has an office in each building and can only be physically networked in those offices.

A sensor process able to determine such virtual locations could take the form of an instrumented DHCP (Dynamic Host Configuration Protocol) server, or it could be a process which regularly "pings" a series of IP addresses to determine which are "alive". Similar sensors can operate at lower network levels, e.g., looking for the presence of particular network interface cards (NICs) by watching Address Resolution Protocol (ARP) packets for particular MAC addresses.

In some cases, a virtual location can not easily be converted to a physical location. For example, an IP address assigned for a dial-in connection provides no physical information unless it can be coupled with other knowledge, e.g., caller-ID information or the fact that a particular user only ever dials-in from home. In other cases, proxies or gateways may obscure a user's true virtual location and hence their physical location.

2.3 Logical Location Sensors

Many of the physical and all of the virtual location sensors described above require some form of computation or association in order to provide *physical* location information. *Logical location sensors* combine information from a physical or virtual sensor with information from some other source in order to infer the physical location of some entity[4]. The additional information can be as simple as a database of the locations of fixed equipment (e.g., magnetic card readers or desktop computers) which, when coupled with an event from a sensor (e.g., card swiped or user logged in), can give a physical location.

Sensor agents (i.e., controlling software) associated with physical location sensors may, under this definition, be considered logical location sensors. Given the role of software in most physical sensor systems, the boundary between logical sensors and physical sensors is somewhat blurred. The easiest way to specify the boundary is based on system interfaces. A physical sensor system provided by some manufacturer may be able to infer location based on information stored within the system. If the interface provided by the manufacturer directly provides physical location information, then this complete system can be considered a physical sensor. If, however, a separate (perhaps third-party) hardware or software module is required to perform the inference, then this module can be considered a logical sensor which works with a particular physical sensor system.

3 Processing of Location Information

The research on pervasive computing systems is a response to the emergence of new types of mobile and embedded computing devices and developments in wireless networking. The domain of computing began to spread from the workplace and home office to other areas of everyday life. In the future, cheap, interconnected computing devices will be ubiquitous and capable of supporting users in a wide range of tasks as predicted by Weiser (1994) and Norman (1998). Computing devices will take many forms, from traditional mobile devices such as mobile phones and handheld computers, to networked home appliances and wearable computers. In addition, research is being carried out on intelligent objects ("smart items") – objects with embedded storage, computing and communication capabilities, as shown in the work of Beigl, Gellersen, and Schmidt (2001). Such objects can interact and create communities of smart items. The objects or the communities can also interact with users through users' computing devices.

Pervasive systems are highly complex systems. The environment in such systems is heterogeneous and makes constructing location management systems a complex task. The previous section clearly showed that sources of location information are heterogenous. Location sensor agents produce fragments of location information which differ in type (physical, virtual, or logical), spatiotemporal characteristic (location, orientation, motion, time), resolution (i.e. level of accuracy), and granularity of the information (i.e. how often it is provided) - to list the most important differences.

The location information has to be gathered from sensor agents and reconciled, i.e., various types of location information about particular entities (e.g. users, devices, applications, and smart items) have to be compared and evaluated in order to compute the location of the entities. Some of the problems which have to be solved during such reconciliation are reasonably straightforward. For example, location readings from two physical sensors, where one of the sensors provides a location region (e.g. a phone cell) and

[4] A distinction is made between logical sensors and the fusion of sensor data (to be discussed further in Sections 3 and 4). Logical sensors work with data from particular sensor systems and, e.g., do not try to resolve conflicts between sensor data.

another the exact coordinates, can be easily resolved if the exact coordinates are within the specified region. There are however a number of problems which are quite difficult to resolve based on the location information only. If a user has many devices (e.g. mobile phone, PDA, laptop) which can provide physical location information and in addition the system is able to gather some virtual location information (e.g., from software agents interacting with the user) the location management system may need to deal with conflicting information and has to provide a means for resolution of such conflicts.

To compare and evaluate location information aggregated from a variety of sensors, a location management system has to be able to transform the location information into a common format. In addition, as location information should be made available to a variety of clients (e.g., users, computing applications and the pervasive system infrastructure), the selected format should be suitable for all of these clients of the location management system. There are already several approaches to define the format of location information including the OpenLS initiative (as described by Hecht (2002)) with the OpenGIS location specification language (from Open GIS Consortium (2002)) which uses the Geography Markup Language (GML) location representation format, and approaches like the Mobile Location Protocol developed by Location interoperability Forum (2002) which uses XML for location representation and defines location formats as DTDs. The drawback of format definitions like GML is that they are defined to describe spatial physical location and it would take a considerable effort to extend them to incorporate virtual location information. As XML is commonly used in current distributed systems and e-commerce systems, it provides a more open, flexible and scalable solution for a common location representation format in pervasive systems.

Location management systems gather location information from many sources and process it to create the final representation available for a variety of clients. The location management systems developed so far for mobile distributed computing and location-based applications usually have a hierarchical architecture. As discussed by Leonhardt (1998), the architecture of these systems can include the following layers: Application/Presentation Layer, Fusion Layer, Abstraction Layer, Reception Layer, and Sensor Layer (see Figure 1). The hierarchical architecture is also suitable for location management in pervasive systems and the architecture of our location management system follows the hierarchical approach. However, it has to allow scalable distribution of the location management system, and the functionality of particular layers needs to be refined to fully address the requirements of pervasive systems, as described below.

Sensor Layer. The lowest level of the location management architecture is the Sensor Layer which represents the variety of logical, virtual and physical location sensor agents producing sensor-specific location information (e.g., GPS coordinates, mobile cell, information that a particular mobile entity was recognised by a camera, or IP address).

Reception Layer. The fragments of location information produced by sensor agents are delivered by the Reception Layer to the Abstraction Layer. It is necessary that the solutions used in the Reception layer are open (can easily accommodate new types of sensors) and scalable (can easily accommodate a larger number of sensors).

Abstraction Layer. This layer takes sensor-specific location information delivered by the Reception Layer and transforms it into a standard format. This layer needs to capture relationships between locations and their identifying features as it needs to associate attributes (eg. mobile cell identifiers) with physical places (for example, suburbs). Relationships between different places should also be managed.

Typically, relationships between different places are modelled as a "contains" relationship. For example, there exists a direct "contains" relationship between Lower Manhattan and Manhattan; there also exists a transitive "contains" relationship between Lower Manhattan and the city of New York. Other (non-hierarchical) approaches to defining relationships between different places are also possible. The Abstraction Layer has to provide these transformations for both physical and virtual location information[5] and needs to provide mappings between virtual (conceptual) and physical location information, where this has not been performed by the sensor.

The Abstraction Layer, like the Reception Layer, needs to be open and scalable; it should easily accommodate translations of new types of physical or virtual location information and use a scalable method for persistent storage of received location information. This location information is then available for processing by the Fusion Layer (see Figure 1). In our location management system we implemented this persistent storage as a relational database.

The Fusion Layer. The Fusion Layer retrieves location information from the persistent storage and collates currently stored location updates to return a single, coherent location of an entity. If there are conflicts in the processed location information they should be resolved at this layer.

In our system, the processing at the Fusion Layer is performed on demand to provide scalability of the location management system. This is because there could be a great difference between the frequency of location updates from sensors and the frequency of location information requests from the clients of the location management system. By introducing persistent storage at the Abstraction Layer, it is possible to perform costly fusion algorithms only when they are required, not for every location information update. Once a location has been determined, it should be cached as a complete location description which will be valid until another update pertaining to that entity is received at the Abstraction Layer. As a result, many requests for a single entity's location can be served without significantly increasing the cost of generating the location from individual location fragments if location information updates are not very frequent.

[5] Logical location information can be considered to be physical – i.e. physical coordinates are generated.

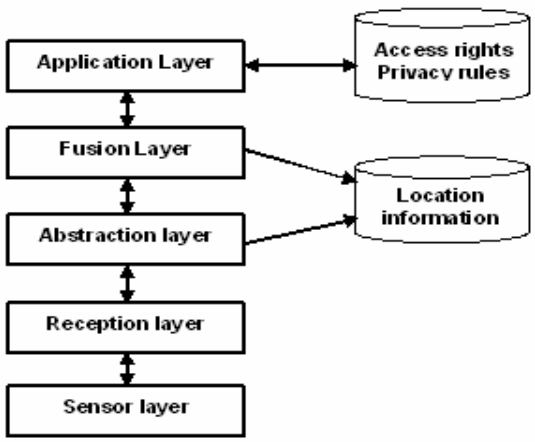

Figure 1. Architecture of a location management system

The Application/Presentation layer. This layer interacts with the variety of clients of the location management system and therefore needs to address several issues including access rights to location information (who can access the information and to what degree of accuracy), privacy of location information (how the location information can be used), security of interactions between clients and the location management system, and scalable mechanisms for interactions between the system and its clients.

Due to strict privacy laws in many countries it is necessary that appropriate privacy policies are incorporated into the location management system, i.e., the system provides an appropriate language to define privacy policies, provides an interface to define, modify, and delete privacy policies, manages the policies in a persistent store and applies them when requests for location information are processed. The management of privacy policies has to be scalable as both the number of clients of the location management system and the number of entities for which the location management system keeps location information can be large.

Another important issue for this layer is the provision of flexible and scalable interaction mechanisms (and appropriate user interfaces for them) for interactions between the system and its clients. Different clients (users, applications, infrastructure of the pervasive system) can have different requirements with regard to mode of interaction (pull/push) and granularity of location information. For some clients, the pull mode (client/server) of interactions may be suitable as they need location information only when they request it. However, for other clients the push model (information about location is "pushed" to the client when the location information changes) is more appropriate.

The need for a push model can be easily explained taking the infrastructure of pervasive systems into account. This infrastructure has to manage context information and location information is one of the attributes of the context information. As the context management part of the pervasive system has to evaluate context changes of particular entities to decide whether any adaptability actions have to be applied, it requires notification of the location changes in order to perform the evaluation. To provide a scalable solution, the interface to the location management system needs to provide a way for its clients to express their requirements for notification about location changes and the required granularity of this notification. The granularity of notification depends upon the need of a particular client (user, application, context management) and may be different even for the same tracked entity (e.g., one client may need to be notified if a user leaves a building, when another has to know user movements with a granularity of 1 m).

As can be seen, there is a plethora of issues which need to be addressed in location management systems in order to make them suitable for pervasive systems. In the following sections we discuss in more detail three of these issues: *conflict resolution* of location information, *security and privacy* of location information, and *scalability* of location management systems.

4 Conflicting Location Information

As shown in section 2, location information in pervasive systems can have many sources. This is particularly true for users as their location may be indicated by a variety of physical, virtual and logical sensors, and their devices' locations may be indicated by a similar variety or sensors. An example of such conflicting location information is the following set of location readings: GPS coordinates from a user's car (corresponding to a company parking lot), cell identification from the user's mobile phone (corresponding to the area around the user's home), virtual information from a location tracking agent in the user's workstation which observed that the user typed on the keyboard 10 minutes ago, and a current camera sighting which spotted the user in the corridor. The Fusion Layer has to evaluate the location entries and compute one single location. Some of these conflicts may not be real conflicts because the differences in resolution and errors in various location technologies can create different location readings. For example if two current location readings are a mobile phone cell and GPS coordinates, and the GPS coordinates are inside the cell area, the difference is due to different technologies used for location tracking.

The conflict resolution at the Fusion Layer has to take into account several factors including location resolution differences, time of location readings, and relevance of various location readings to a particular entity. Initial approaches to the resolution of conflicts in location information are starting to appear in the literature.

The system presented by Myllymaki and Edlund (2002), which was developed to support location-based computing, receives location information from a variety of devices and computes a single user location by applying the aggregation function to these location readings. The function compares timestamps, resolution and "associative confidence" to compute user location. The associative confidence is a probability that the device whose location is being reported is actually at the same location as the user being tracked. Each device can have two associated confidences depending on whether the device is moving or is stationary (if a car is

moving there is a high probability that its owner is in the same location as the car, if the car is stationary the probability that its owner is in the same place is low).

The algorithms which we applied in the Fusion Layer for location conflict resolution are very similar in basic concepts to the method described above, i.e., we use time, resolution and relevance of location readings, but in addition, we also take the history of conflict resolution into account. We have also found that, due to a large variety of virtual, logical and physical sources of location information in pervasive systems, it is not enough to have two values of confidences related to objects. There are, for example, many types of sensors which provide very accurate location reading but the reading is only accurate for a very short time. Sensors like cameras or swipe cards provide accurate location readings but after some time the information value of such readings disappear. This feature of such sensors has to be captured in the conflict resolution algorithm. To deal with this problem, our location management system allows defining confidences for various types of sensors. The Fusion Layer provides a means for both easy modification/addition of confidences for particular types of location sources and modification of the aggregation function for particular users. We have improved on the existing approach, however, in general, the problem of reliably resolving location conflicts in pervasive systems is still an open research issue.

5 Security and Privacy of Location Information

The issue of protection of location information is a very important one. This importance is illustrated by the difficulties that many telecommunication companies currently have with providing location-based applications. The provision of these applications is very limited due to strict privacy laws in many countries. Some of the telecommunication companies limit the provision of user location information to the user's devices only. The problem is even greater for future pervasive systems which could gather location information from many sensors and can infer the user's location even based on user's actions. It has to be noted that to achieve protection of location information several aspects of the problem have to be addressed:

- *security* of location information exchange (which can be achieved through authentication and encryption);
- *access rights* to location information (which can be enforced by the definition of access rights, the authentication of clients accessing location information, and the checking of access rights); and
- *privacy rules* for location information (which define the purposes for which location information can be used if accessed).

There are no comprehensive solutions for access rights to location information and privacy of location information as yet. Research has been carried out on access rights and privacy of information in variety of areas (access control, policy based management, privacy rules) which, when integrated and enhanced, can create a basis for location information protection in pervasive systems. Two of the current approaches which are closely related to research on location privacy are characterised below.

Cranor, Langheinrich, Marchiori, Presler-Marshall, and Reagle (2002) describe the W3C P3P (Platform for Privacy Preferences) initiative which is an attempt to standardise ways to provide privacy of users accessing web resources. P3P addresses the issues of (i) how users define access rights to their location information; (ii) how they express their privacy preferences (i.e. how the location information can be used); and (iii) how the system manages and exchanges privacy rules. When users access Web resources, their user privacy preferences are compared to the browser's preferences and depending on the result, an action is taken. The P3P specification defines how the comparison is made and defines methods for resolving conflicts. P3P privacy preferences are exchanged as an XML representation of the privacy data over the HTTP protocol. Although still a work-in-progress, the P3P is a promising approach for the definition and management of access rights and privacy preferences, and its application to location information is feasible. The OpenLS initiative which uses the OpenGIS location specification language advocates the use of P3P in the future.

BlueLocator is an IBM project whose goal is to build an enterprise location service, i.e., provide location information about employees. It uses the OpenGIS GML specification for communicating location information. BlueLocator uses Access Control Lists to define who can access employees' location information and in which circumstances. It leverages the enterprise organisation structure and uses a report-to-chain tree as a basis for defining Access Control Lists. Only peers (employees who report to the same manager) and the manager can access the location information of a particular employee. The system uses physical location information only but it will be extended in the future.

In our location management system we are in the process of incorporating a P3P based approach to access rights and privacy preferences. Our early experience with using P3P for defining privacy of location information created mixed results:

- It was reasonably easy to extend our location management system to validate and compare P3P rules as the validator and comparison engine are available for P3P;
- It was feasible to apply P3P to define location privacy rules but the process is cumbersome and creates very lengthy rule definitions as it is impossible to aggregate many rules in one P3P definition.

The difficulty in using P3P to support privacy of location information will be exacerbated by the requirement that in pervasive systems there are many clients of the location management system. It is therefore necessary to provide access rules for the variety of these clients, and user privacy preferences need to address the variety of ways in which the location information can be used by these clients. This

implies that the location management system has to have an interface for users to define access rights and privacy preferences for their (and their artefacts) location information. It also needs well developed support for users to be able to evaluate their definitions, discover conflicts in their definitions and see the consequences of the defined access rights and privacy preferences. The issue is complex and is an open research problem.

6 Scalability Issues

To achieve a scalable location management system it is necessary to provide scalable solutions for the system's components. The solutions in the following areas will have high impact on the whole system's scalability:

- reception of location information from sensors;
- interaction with clients;
- updates of location information in persistent repositories; and
- distribution of location management.

The impact of interaction methods (interactions of the Reception Layer and interactions between the Application Layer and its clients) on the scalability of the location management system has been partially discussed in section 3. In our location management system we used a distributed notification service, Elvin, designed by Segall, Arnold, Boot, Henderson, and Phelps (2000), that provides notifications and causal ordering of notifications. It provides the functionality of the Reception Layer as it forwards notifications between the Sensor Layer and the Abstraction Layer. The same notification system is used for delivering notifications about location changes between the Application Layer and the clients of the location management system. Elvin has a sophisticated mechanism for the registration of interest in events for which notifications are to be delivered, and therefore it allows clients to specify the granularity of location information notifications. Figure 2 illustrates both the Reception Layer and the clients' interactions with the system.

As location information is stored in a persistent repository in the Abstraction Layer, a scalable method for updating this repository needs to be developed for entities with continuous movements (e.g., GPS coordinates of a fast moving car). This is because the organisation of the repository (e.g., a database) is not usually designed for very frequent updates. There are many existing approaches (spatial indexing techniques) for this problem. For example, each entity's location can be described as a linear function and the database is only updated when some parameters change, e.g., the speed or direction of the entity. Another approach is the Lazy Update R-Tree which updates an entity's position only if the entity moves out of its bounding rectangle as shown by Kwon, Lee and Lee (2002). Any of the known techniques can be used in the location management system designed for pervasive infrastructure, but the technique should be applied in the Sensor Layer in order to limit the number of notifications sent to the Abstraction Layer.

As we use the Elvin notification system and Elvin allows registration for a specified granularity of notifications, we achieved a scalable approach in the following way: the Abstraction Manager registers with the sensor agents for notifications about changes in particular location parameters (e.g., current location, speed, direction) and the required granularity of this information (e.g., `notify if the difference between the current location and the previous notification is greater than 100m). The clients may need different granularities of location information. For example, another application may need notifications about location changes with a granularity of 1m. Therefore, to meet the needs of all the clients, the value of the granularity with which a given sensor agent sends location information to the Abstraction Layer could be a highest common divisor of the granularities of location notifications requested by the clients.

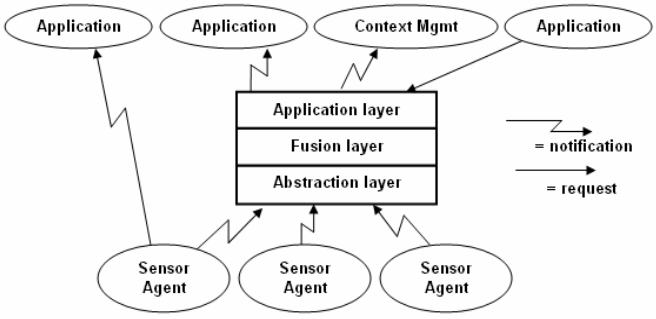

Figure 2. Interactions between sensor agents and clients

As pervasive systems may be geographically large and have a heterogeneous computing and networking infrastructure, one of the scalability requirements is distribution of location managers in this infrastructure. The location management system with the architecture and functionality described in section 3 can be easily distributed. Wide-area coverage can be achieved by deploying a number of managers implementing the Abstraction Layer which will gather location information from geographically close sensors. Multiple Abstraction Managers can update a single Fusion Layer manager. The solution allows distribution of not only the Abstraction Layer but also the Fusion Layer and the Application/Presentation Layer. On the other hand, if a tracked entity moves a large distance from its home location management system, provision can be made for the creation of a visitor location profile in the Abstraction, Fusion and Application Managers at the new destination.

Not all applications require the sophisticated services provided by the location management systems. There are some applications, like simple tourist guides, which may work on one type of location information, e.g. GPS coordinates, in a limited geographical area. The proposed architecture allows such applications to directly register with the required sensor agent to provide notifications about location changes.

7 Conclusions

The proliferation of mobile devices has created a demand for location-based services. There is a large body of research on the basic issues of location-based computing: technologies for location sensors, location determination for mobile devices, location representation formats, early approaches to aggregation of location information (conflict resolution), and privacy of location information.

In this paper we discussed future location management systems, which, in addition to providing location information to single applications, will need to provide location information to the infrastructure of pervasive systems. These systems will be inherently more complex, due to a large variety of location sensors, the variety of entities supported by the pervasive system infrastructure, the high probability of conflicts in location information, complex location access and privacy rules, and scalability problems (related to the heterogeneity of sensors, a large number of sensors, a large number of clients, and the geographical distribution of sensors). This paper has presented a categorization of location sources, an architecture for location management systems, and a discussion of three important aspects of such systems: conflict resolution in location information, access rights and privacy rules, and scalability issues. The presented discussion is based on the authors' experience in developing a location management system and integrating it with the infrastructure for pervasive systems. This development is ongoing.

8 References

CHEVERST, K., DAVIES, N., MITCHELL, K., FRIDAY, A. and EFSTRATIOU, C. (2000): Developing a context-aware electronic tourist guide: some issues and experiences. *Proc. Conference on Human Factors and Computing Systems*, The Hague, Netherlands.

CHEN, Y., CHEN, X., DING, X., RAO, F. and LIU, D. (2002): BlueLocator: Enabling Enterprise Location-Based Services. *Proc. Third International Conference on Mobile Data Management*, Singapore. IEEE Computer Society.

HENRICKSEN, K., INDULSKA,J., RAKOTONIRAINY, A. (2002): Modeling Context Information in Pervasive Computing Systems. *Proc. First International Conference on Pervasive Computing,* Zurich, Switzerland, LNCS **2414**: 167-180, Springer Verlag.

HIGHTOWER, J. and BORRIELLO, G. (2001): Location Systems for Ubiquitous Computing. *Computer* **34**(8): 57-66.

HARTER, A., HOPPER, A., STEGGLES, P., WARD, A. and WEBSTER. P. (1999): The anatomy of a context-aware application. *Proc. Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, Seattle, USA.

WhereNet Corp. (2002): WhereNet: WhereTag II. http://www.wherenet.com/download/WhereTag.pdf, last accessed 5 September 2002.

Wireless Mountain Laboratories Inc. (2002): Spider Tags RFID Data and Asset Tracking. http://www.wirelessmountain.com/spidertags.html, last accessed 5 September 2002.

WANT, R., HOPPER, A., FALCÃO, V. and GIBBONS, J. (1992): The active badge location system. *ACM Transactions on Information Systems* **10**(1):91-102.

PRIYANTHA, N.B., CHAKRABORTY, A. and BALAKRISHNAN, H.(2000): The Cricket location-support system. *Proc. Sixth Annual International Conference on Mobile Computing and Networking*, Boston, MA, USA.

WEISER, M. (1994): The world is not a desktop. *Interactions* **1**(1): 7-8.

NORMAN, D. (1998): *The Invisible Computer*. Cambridge, Massachusetts, USA, MIT Press.

BEIGL, M., GELLERSEN, H.W. and SCHMIDT A. (2001): MediaCups: Experience with Design and Use of Computer-Augmented Everyday Artefacts. *Computer Networks: Special Issue on Pervasive Computing* **35**(4): 401-409.

HECHT, L. (2002): Location Services: Remarks, Considerations, Challenges. First Location Interoperability Forum Meeting, http://www.openls.org/docs/LIF.htm.

OPEN GIS CONSORTIUM, (2002): OpenGIS Abstract Specification. http://www.opengis.org/techno/abstract.htm last accessed 28 May 2002.

LOCATION INTEROPERABILITY FORUM, (2001): Mobile Location Protocol. Approved Specification LIF TS 101 v2.0.0, Nov. 2001. http://www.locationforum.org/publicdownload/LIF-TS-101-v2.0.0.zip.

LEONHARDT, U. (1998): Supporting Location-Awareness in Open Distributed Systems, PhD Thesis. Imperial College, London.

MYLLYMAKI, J. and EDLUND, S. (2002): Location Aggregation from Multiple Sources. *Proc. Third International Conference on Mobile Data Management*, Singapore. IEEE Computer Society.

CRANOR, L., LANGHEINRICH, M., MARCHIORI, M., PRESLER-MARSHALL, M. and REAGLE, J. (2002): The Platform for Privacy Preferences 1.0 (P3P1.0) Specification. W3C Recom., April 2002 http://www.w3.org/TR/P3P/

CRANOR, L., LANGHEINRICH, M., MARCHIORI, M. (2002): A P3P Preference Exchange Language 1.0. W3C Working Draft,2002.http://www.w3.org/TR/P3P-preferences.

SEGALL, B., ARNOLD, D., BOOT, J., HENDERSON, M. and PHELPS, T. (2000): Content Based Routing with Elvin4. *Proc. AUUG2K*, Canberra, Australia.

D. KWON, D., LEE, S. and LEE, S. (2002): Indexing the Current Positions of Moving Objects Using the Lazy Update R-tree. *Proc. Third International Conference on Mobile Data Management*, Singapore. IEEE Computer Society.