Privacy Sensitive Location Information Systems in Smart Buildings*

Jodie P. Boyer, Kaijun Tan, and Carl A. Gunter

Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801
{jpboyer, kaijunt,cgunter}@cs.uiuc.edu

Abstract. Increasing automation of buildings enables rich information streams about the activities of building users to reach networked computer systems. Privacy concerns typically cause this information to be accessible only by building managers and security personnel. However, if appropriate privacy mechanisms can be implemented, then it is possible to deploy location information systems that can contribute to the convenience and efficiency of users. This paper describes a three step approach to privacy-sensitive release of location information collected by building sensors. These steps entail defining an ownership model, defining environment events to be monitored, and creating a sharing model. These steps are described mathematically and then validated through a case study for a system called Janus's Map which provides a location information system for the card reader, door, and occupancy sensors of a modern smart building.

1 Introduction

Buildings are becoming increasingly intelligent through the deployment of Building Automation Systems (BASs). Advances in sensors, control systems, networking, and information technology enable greater automation to increase safety and comfort for users and convenience for building managers. BASs often also include enhancements to the security of buildings by providing more sophisticated entry mechanisms such as card reader systems and a variety of surveillance, audit, and remote control mechanisms. Such mechanisms create a rich stream of information about users. To address concerns about the privacy of users, it is typical to restrict this information to building managers and security officers, often with restrictions such as using surveillance data only to respond to specific types of security incidents. However, there are many cases in which the location information collected by the BAS can help building users as well as building managers. The aim of this paper is to present a model, architecture, and case study for distributing location data from a BAS infrastructure to users in a way that reasonably addresses privacy concerns.

* In: Security for Pervasive Computing, York England, April 2006.

For the purposes of this paper, let us define a *Location Information System (LIS)* to be a system that allows users and managers to control and learn information about tracked people and objects using information generated from a BAS. Mechanisms that can provide the needed raw data include networked card reader systems, video surveillance cameras, and motion detectors. These may be supplemented by Radio Frequency ID (RFID) tags, computerized scheduling and workflow information, communication activity, and other mechanisms. As a running case study for this paper we have focused on a subset of such mechanisms that can be found in the Siebel Center in Urbana, Illinois, a "smart building" opened in 2004. The Siebel Center BAS audits information about the use of card readers to gain access to shared and individual offices throughout the building and supplements this with information collected from occupancy sensors and sensors that detect the state of doors. This information is used only by building managers to improve building maintenance functions (such as diagnosing faulty card readers) and respond to security incidents. It provides a good case study for the challenges of creating an LIS that would open this information to building users to support applications like team communications and improved workflow.

There are two key challenges that must be addressed to create an LIS from the existing infrastructure provided by a BAS. First, it is necessary to convert the information provided by the BAS into the kind of information needed for a useful LIS. BAS sensor data usually provides only an estimate of the desired information about the environment that users care about. For instance, if a user was identified by a card swipe opening a door to a room with an inactive occupancy sensor and the sensor has been active ever since, then it is credible to conclude that that user is in the room. However, it is also possible that the user in question opened the door for a colleague and it is this colleague who is in the room, not the one who used the card reader. If the LIS is to be of any use it must be able to tolerate this type of inaccuracy or use supplementary information to disambiguate the sensor data. The second challenge is to create a privacy system that meets the constraints for enterprise purpose and user privacy norms while still serving a worthwhile function that all of the stakeholders can agree on. A particular observation is that there is no "one size fits all" solution for this, since these requirements will change between major types of users. For instance, the needs of a commercial enterprise are likely to differ from those of a university or government building and there could be great variation within these sectors such as the difference between a government public building and a military facility.

Our approach to building an LIS for a BAS is premised on the idea of delivering a limited degree of discretionary control to users. The primary contribution of this paper is to describe and validate a breakdown of this strategy into three fundamental steps:

1. Define an ownership model for system events.
2. Determine the environment events of interest and how to deduce them.
3. Develop a model for privacy-sensitive information sharing for these events.

The process begins by determining a way to associate system events such as card swipes or motion detector readings with users in a way that users can be said to

"own" the data because it is *about* them. Such events may be too low-level to be of direct interest so the next step is to determine environment events or state that users really care about, such as whether a user is in his office, and define ownership for these. These events must be inferred from system events with an acceptable threshold of error. The final step is to describe the rules that a user can employ to control the distribution of environment events that are owned by her.

We elaborate this approach in two stages, first by defining the general approach mathematically as a family of structures and functions that capture the key concepts at a high level and second by instantiating this model in a case study for the Siebel Center called "Janus's Map". It uses the electronic door lock system and occupancy sensors to implement an LIS that provides users information about the presence or whereabouts of other users in the building. Additionally, Janus's Map allows users to specify rules describing the information others are able to determine about them. These rules are fairly intricate but give users fine-grained control not only concerning who can see their data, but also how accurately their location is revealed to other users.

This paper is organized as follows. In Section 2 we discuss possible infrastructure that can be used for LISs. In Section 3 we present a general mathematical model for developing LISs. In Section 4 we present an instantiation of the general model for Janus's Map which is followed in Section 5 by a discussion of the architecture and implementation of Janus's Map. In Section 6 we discuss related work and then we conclude in Section 7.

2 Infrastructure and Applications for Location Information Systems

One of the key contributions of LISs is that they are able to determine high level information about an environment using existing infrastructure. There are many different pieces of infrastructure that could be used to aid in generating these environmental events, including ones outside of the typical BAS infrastructure. Here are a few examples:

Door Lock System Many buildings now use electronic door locks on some or most of the doors. Building occupants can use cards to unlock the doors to which they have access. As a result, the door lock system is able to (approximately) determine which users have opened doors in the building.

Occupancy Sensors Many BASs include occupancy sensors in many or all of the rooms in a building. These are typically used to control the lights in the rooms in order to save electricity, but may be linked into the BAS audit functions as well.

Network Jack Activity Because many buildings assign specific network jacks to specific users, network activity can be used to gain (partial) information about user location.

Application Software LISs could take advantage of certain types of application software, some of which provide their own activity data. In particular, most instant messenger programs provide status information about users.

`Valid_Access`	The user's swipe was accepted, and the door was opened
`ValidAccessNoEntry`	The user's swipe was accepted but the door was not opened
`InvalidAttempt`	The user's swipe was not accepted
`Door_Ajar`	The door was opened
`DoorAjarCleared`	The door was closed

Fig. 1. The events available in the door lock system

Video Surveillance Another piece of infrastructure that could be used for an LIS are the security cameras in common areas around the building. If a security camera is trained on a door, we would be able to keep a running count of users in a room by watching people enter and exit the room. If the security camera system is equipped with facial recognition software, the system would be able to locate people even in common areas, allowing the system to provide more accurate information. This source of data is considered particularly privacy-sensitive, however, and introduces significant processing challenges.

Wireless Network There are a variety of suggestions for using wireless networks to allow a laptop to determine its location. This process could be included in an LIS. This information would be most useful for an LIS if the MAC address or IP address of a wireless device could be easily associated with a specific user.

Global Positioning System (GPS) Despite the fact that GPS does not work well indoors, it could still be used to aid an LIS designed for a use on a campus or to tell if an individual has entered a building.

RFID Tags An LIS could work with either active badges, which transmit signals using battery power, or passive badges, like those used in inventory control. New technologies are making it possible to locate objects within rooms by reading these tags. Some advanced systems allow objects to be located in rooms to within a few centimeters.

Telephone If a user's office phone is in use, this provides evidence that they are in their office. This information is not easily integrated with BAS data in most installations but the rise of voice over IP could change this.

Our formalism, which we describe in the next section, is intended to extend to all of these possible sources of data and others we did not list or have not considered. Our case study is based on a realistic subset of these that can be found in a specific building. The Siebel Center is a 225,000 square feet facility opened in April of 2004. The BAS in Siebel Center is the Continuum system from Andover Controls. It uses both the an electronic door lock system and occupancy sensors. These two systems can be combined to estimate the locations of individuals in the building. The door lock system in the Siebel Center generates five events of interest which are shown in Figure 1. The occupancy sensors in the building can be queried to determine whether a room is currently occupied. The building uses card readers for access to sensitive areas including offices of individuals.

There are a variety of applications that an LIS could serve and we will not attempt to provide a list of these here. We focus mainly on the basic function of providing users with general information about the physical location of other users with a level of accuracy that makes the information useful. However, there is much more that one could hope to achieve in an ambitious program along these lines. For instance, an integration of calendar data, telephone and computer usage, and BAS sensor data might provide a sophisticated LIS that estimates availability of a user as well as physical location.

3 A General Model for Location Information Systems

As mentioned above, we envision that there is a three step process in developing an LIS. We will follow these steps as we present a general model for LISs.

Ownership Model The first step in developing an LIS is to define an ownership model. From the ownership model derives a mechanism to understand how privacy preference can be applied an enforced. In this section we define the ownership model for a general LIS. Every LIS will have a set of users, U, a set of locations, L, and a linear ordered space, T which represents time. Additionally, every LIS has a set, S of system events, which are generated by the BAS, or other systems. Additionally, we define three functions that act on a system event $s \in S$. The function $user : S \rightarrow U \cup \{\perp\}$ is used to determine the users associated with a system event s. The function $loc : S \rightarrow L$ determines the location in which s occurred. The function $time : S \rightarrow T$ determines the time at which s occurred.

In an LIS, we use ownership to determine who has control over certain entities in the system. In general, there are two system entities to which we assign ownership, locations and system events. In an LIS, we define two functions that determine the owner of an entity. We use the function $o : L \rightarrow 2^U$ to determine the set of owners of a location. This ownership function will likely be a static mapping to a set of users that, for instance, have access to a room in the building. The function $\rho : S \rightarrow 2^U$ is used to determine the set of owners of a system event. We usually take the owner of the system event s to be $user(s)$, or the users associated with the event. If $user(s)$ returns $\perp$, then the owner of s will probably be $o(loc(s))$, the owners of the location in which s occurred. For example, if a system event is reading an RFID tag, the owner of the event might be the user that owns the tagged object. In contrast, if the event was the temperature reading for a room, the owner of the event might be the owner of the room. It is important to note that ρ should not be a static mapping as events are generated on the fly.

Definition An *ownership model*, $\mathcal{O}$ consists of:
- U, the set of users
- L, the set of locations
- S, the set of system events
- T, a set of values with a linear ordering, signifying time.

together with
- $time : S \rightarrow T$ which determines the time an event occurred

– $user : S \to U \cup \{\perp\}$ returns the user (if any) associated with a system event
– $loc : S \to L$ returns the location associated with a system event
– $o : L \to 2^U$ returns the owners of a location
– $\rho : S \to 2^U$ returns the owners of a system event □

Environment Events The main purpose of an LIS is to aggregate system events into information that can be easily understood by users. For example, it may take several RFID readings to determine the accurate location of an object. The average user would prefer to see the aggregated information then a list of low level events. We call such aggregated information environment events. The next step in developing an LIS is to determine an appropriate set of environment events. An environment event may be determined from many low level events, such as location, or deduces from a single event which may be determined by a single event. We call this set of environment events E.

We then define a function $induce : 2^S \to 2^E$ which determines the set of environment events that may be deduced from a set of system events. We choose to implement *induce* as a function that applies a set of deduction rules to the given set of system events and returns a set of environment events. These deduction rules are of the form

$$\frac{s_1, ..., s_n \in \Gamma \text{ such that } C}{e_1, ..., e_m \in \Sigma}$$

where $\Gamma \subseteq 2^S$, C is a set of conditions on $s_1, ..., s_n$ and $\Sigma \subseteq 2^E$. This rule means that the existence of a set of system events satisfying a set of conditions implies the a set of environment events. Additionally, the hypothesis may also contain some conditions on the set of system events.

Information Sharing The last and most important step of developing an LIS is to develop a method for information sharing. Currently, system events are heavily protected in order to respect user's privacy. In order to accommodate this, we define two families of functions $filter : U \times U \to (2^S \to 2^S)$ and $mask : U \times U \to (2^E \to 2^E)$. Each one of these functions returns a function that describes how elements should be altered or removed to project a user's privacy. We call $filter_u^v : 2^S \to 2^S$ u's filtering policy for v, which is applied when u is a target of v's search. We call $mask : 2^E \to 2^E$ u's masking policy for v, which is applied when u is a target of v's search. We apply two policies to each search because it gives he user not only the ability to control what system events are used to determine environment events, but also gives the user the ability to restrict what environmental events are returned. We sometimes refer to the tuple $(filter, mask)$ as the privacy policy for an LIS. The privacy policy is the final component that needs to be defined for an LIS.

Definition A *location information system*, $\mathcal{L}$, between an ownership model as defined above and a set, E, of environment events consists of the following 3 functions:
– $filter : U \times U \to (2^S \to 2^S)$ defines a user's filtering policy.
– $mask : U \times U \to (2^E \to 2^E)$ defines a user's masking policy.

Time	Location	User	Type
1/1/2006 07:45	SC3405	Alice	InvalidAccess
1/1/2006 10:00	SC4105	Alice	ValidAccessNoEntry
1/1/2006 10:01	SC4309	Alice	ValidAccess
1/1/2006 10:01	SC4309	$\perp$	DoorAjar
1/1/2006 10:03	SC4309	$\perp$	OccupancySensorTrue

Fig. 2. An example stream of system events in Janus's Map

– $induce : 2^S \to 2^E$ is a function that maps a set of system events to a set of environmental events. □

Additionally, we define a family of functions $reveal : U \times U \to (2^S \to 2^E)$. We define the function $reveal_u^v : 2^S \to 2^E$ as the composition of *filter*, *mask*, and *induce* as follows:

$$\begin{array}{ccc} & \xrightarrow{filter_u^v} & \\ 2^S & & 2^S \\ {\scriptstyle reveal_u^v}\downarrow & & \downarrow{\scriptstyle induce} \\ 2^E & \xleftarrow{mask_u^v} & 2^E \end{array}$$

The function $reveal_u^v$ is called when v is performing an action for which u is the target.

4 An Example Instantiation: Janus's Map

As described in the previous section, there are three steps in defining an LIS. In this section, we present an insanitation of an LIS for the Siebel Center called Janus's Map. The Siebel Center is equipped with an electronic door lock system and occupancy sensors. The electronic door lock system allows doors to be opened with a person's university ID and the occupancy sensors are designed to shut lights off in rooms in order to save electricity. We propose using these systems to build an LIS.

Ownership We begin with the ownership model of Janus, which requires us to define spaces of users, locations, system events, and times. Users can be modeled with user IDs and times as real numbers. Locations include such things as offices (represented as numbers) but the space required is more subtle than that. To accommodate our masking policy, we introduce the set $\mathcal{G} = \{floor, wing, room\}$ which defines the possible granularities of locations in the building. We also define the sets $L_{floor} \subset L$, and $L_{wing} \subset L$, and $L_{room} \subset L$, which (strictly) contain the floor locations, wing locations, and room locations, respectively. Therefore any location l in the building could be defined as a tuple of type $L_{floor} \times (L_{wing} \cup \{\perp\}) \times (L_{room} \cup \{\perp\})$. System events, S, are tuples of type $(U \cup \{\perp\}) \times L \times T \times \tau$ where τ is a set of types of system events comprising the following values: ValidAccess, ValidAccessNoEntry, InvalidAccess, DoorAjar, DoorAjarCleared, OccupancySensorTrue, and OccupancySensorFalse. These correspond to the event types in Figure 1.

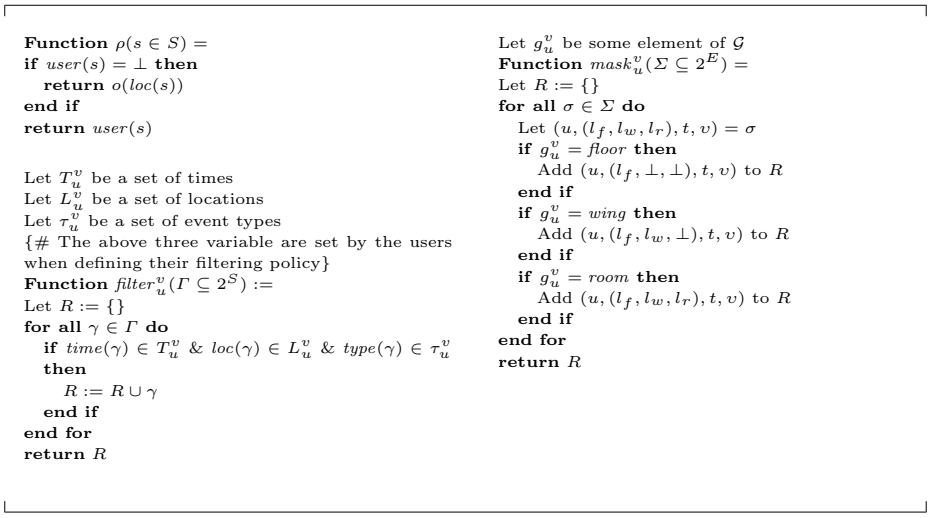

Fig. 3. Algorithms for ρ, *filter*, and *mask*

To complete the definition of the ownership model we need to define functions *time*, *user*, *loc*, o, and ρ. The function *time* returns the times at which a system event occurred; such as the time at which a door is opened. The function *user* returns the user field of an event. The function *loc* returns the location in which an event occurred. Additionally, we define the function $type : S \to \tau$ which returns the event type field of a system event. For the formal model we use a static policy describing the ownership o of locations.[1] Generally, any person who has a desk in a room is assigned ownership rights over a room. For public spaces, we assign ownership to a system administrator. Finally we define the ρ function. By default, we assign event ownership to the user referred to in the event, for example the user who swiped their card into the door. If the user field of an event is $\perp$, we assign the owner of the event to be the owner of the location at which an event occurred. A formal definition of ρ is in Figure 3.

Figure 2 shows an example stream of system events in the BAS for the Siebel Center. The owner of the event in first line is Alice and the owner of the event in the last line is the owner of the room SC4309.

Environment Events Now that our ownership model has been defined, we can define our environment events for for Janus's Map. The main goal of Janus's Map is to present location information to users in the building. As a result we define an environment event that describes the location of an individual. We define E as set of tuples $U \times L \times T \times P$ where $P := \{\text{In}, \text{Near}\}$. The P field of an event describes whether we are certain that a user was actually in a location or only near it, for example, if a user has an invalid swipe to a door, we can only say they were near that location at the time of the swipe but they never

[1] In the implementation this needs to vary over time since users will periodically get new offices.

actually entered it. In Janus's Map we define other environmental events, but for simplicity we will focus on user locations.

As described above, we choose to define *induce* as a function that applies a set of deduction rules to a set of system events and determines a set of environmental events that can be inferred from the given system events. One such deduction rule states that if a ValidAccess event occurs followed by a DoorAjar followed by a OccupancySensorTrue event all in the same location we can deduce that the users who performed the ValidAccess was in the room at the time of the OccupancySensorTrue event as well as that there were near the room at the time of the ValidAccess event. Given this induction rule, the events listed in Figure 2 induce two environment events: (Alice, SC4309, 1/1/2006 10:01, Near) and (Alice, SC4309, 1/1/2006 10:03, In). In the interest of space, we omit the other deduction rules for Janus's Map.

Information Sharing We now define the privacy policy for Janus's Map. Users define a privacy policy by specifying two functions that describe how system and environment events should be revealed in order to preserve a subject's privacy. We define both the *filter* and *mask* functions inductively by defining a general version of the functions filter_u^v and mask_u^v[2]. These functions are shown in Figure 3 and described below. In order to define filter_u^v, the user, u first defines the sets T_u^v, L_u^v, and τ_u^v which specify requirements on system events that may be released to v. The set T_u^v specifies the set of times during which the system events must occur in order to be released to v, the set L_u^v specifies the locations in which a system event must occur in order to be released to v, and the set τ_u^v is the types of system events that can be released to v. The function then proceeds to filter out events that do not meet all of these requirements and returns only the events that occur during a time specified in T_u^v, in a location specified in L_u^v and have an event type specified in τ_u^v. In order to define a masking policy, mask_u^v, user u must first select a value for g_u^v, which is an element of the set $\mathcal{G}$. The value specifies the granularity at which locations can be returned to v, for example, Alice might specify that Bob is only allowed to know her location by floor. The mask_u^v function the proceeds to mask a location based on the value of g specified by u. By default, filter_u^v and mask_u^v always return the empty set.

For example, Alice could define a filtering policy, $\mathit{filter}_{alice}^{bob}$, in which T_{alice}^{bob} is defined as the set of times defined by the phrase "Any day between 08:00 and 17:00", $L_{alice}^{bob} = \text{Ł}$, and $\tau_{alice}^{bob} = \{\text{ValidAccess}, \text{DoorAjar}, \text{OccupancySensorTrue}\}$. Similarly, Alice could define a masking policy, $\mathit{mask}_{alice}^{bob}$, such that $g_{alice}^{bob} = \mathit{floor}$.

We now define the function *reveal*. The function reveal_u^v is a composition of filter_u^v, *induce*, and mask_u^v. In Janus's Map, *reveal* returns a set of environment events. Because Janus's Map is mainly a search system, our reveal function roughly translates into a search function. The most recent environment event returned by reveal_u^v would be the system's best guess of u's location as returned to v. If we were to apply $\mathit{reveal}_{alice}^{bob}$ to the set of system events listed in Figure 2, the following environment events would be revealed to us: (Alice, SC4, 1/1/2006

[2] The filtering policy and masking policy are simplified here for readability. A full treatment of these policies is given in Section 5

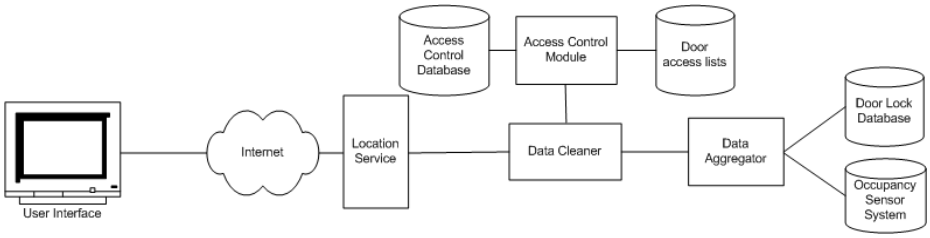

Fig. 4. The architecture of Janus's Map

10:01, Near) and (Alice, SC4, 1/1/2006 10:03, In). We now present a discussion of the architecture and implementation of Janus's Map for the Siebel Center.

5 Janus's Map

This section focuses on the architecture of our prototype LIS that has been developed for the Siebel Center. Janus's Map can be broken down into 9 separate components. The interaction between these components is show in Figure 4. Breaking down Janus's Map into multiple components allows each component to be replaced if the infrastructure is changed with minimal effect on the other components.

User Interface The user interface for Janus's Map is a web page. By using a web page, all users are able to use the system regardless of their preferred platform.

Location Service The location service is the interface on the server side that services requests to the information service. The function *reveal* is defined in the location service

Data Cleaner The data cleaner sanitizes the location data that will be returned to the users. It will be discussed in greater detail below. The data cleaner contains an implementation of *filter* and *mask*.

Access Control Module The access control module is used to interface the the access control databases. The access control module contains an implementation of the function ρ.

Access Control Database The access control database stores the users rules that grant the right of other users to see their data. These rules are discussed in greater detail below. Rules in the implementation of Janus's Map encapsulate both the filtering policies and the masking policies.

Door Access List The door access list stores the unlock rights for users for specific doors.

Data Aggregator The data aggregator component collects data from the data sources and packages the data into a common data structure used throughout the rest of the system.

Door Lock Database The door lock database contains a log of all events in the door lock system.

Occupancy Sensor System The occupancy sensor system can be queried to determine the current occupancy state of a specific room.

Target: Bob, Carol
Visible fields: Event Type, Event Time, Room
Granularity: Wing
Number of past entries: 5
Event types: Valid Access only
Event time: Between 8am and 5pm
Event date: From Jan. 1, 2006 to Jan 1, 2007
Event days: Monday - Friday
Rooms: All

Fig. 5. Alice's example rule

Access Control The access control system in Janus's Map is of key importance because it is a privacy sensitive system. As discussed in Section 4 users are able to define filtering policies and masking policies which control access to their owned events. In order to facilitate the delegation of access rights, Janus's Map allows users to define rules that encapsulate the definition of both the filtering policy and the masking policy. We give an example of a rule in Figure 5. Rules for Janus's Map can be split into 3 basic parts.

Targets The targets of the rule are the users to whom the rule applies. For example, Alice has a rule with the targets Bob and Carol. This means that the rule will concern Bob and Carol's ability to access Alice's location data. This essentially allows users to define a single filtering and masking policy that applies to many users.

Data Access The data access portion of a rule describes the kind of data that the targets can view about a specific user. Users are able to limit the quantity, the event type, the time, the date, and the room number. For example, Alice's rule states that Bob and Carol are only able to see events that occurred between 8am and 5pm on weekdays. This part of the rule defines a filtering policy.

Visibility The visibility portion allows users to force certain fields to be hidden in the results returned to the rule targets. Additionally, the visibility portion of the rule also allows the users to change the granularity of the room data returned. Users can define the granularity to be one of 4 levels, room, wing, floor, or building. This way users are able to prevent the targets of rules from finding too much information about their location in the building. For example, Alice's rule states that Bob and Carol are only able to see the Event Type, Event Time, and the wing for events returned to them. This part of the rule describes a masking policy.

Rules Application In order to fully understand how Janus's Map works, it is important to know the order each of the filtering rules are applied. The very first filter that is applied to the data is the number of past entries that can be used for deduction. For example, only the latest 5 events are allowed to continue forward. Once this happens, each event is then considered separately. The system determines if the date and time are within in the constraints, and

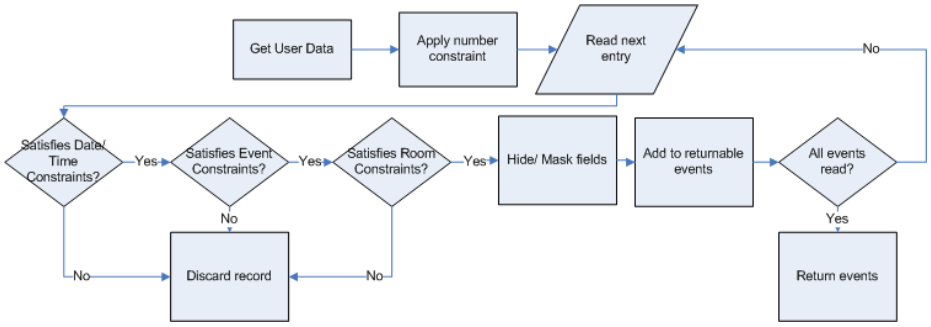

Fig. 6. The rule application process

whether the event type and event door are allowed by the rule that applies to the particular user. The last part of the rule that is applied is removing invisible fields and masking the room number as appropriate. This process is illustrated in Figure 6. You will notice that this process encapsulate both *filter* and *mask*.

The most important side effect of this ordering is that users are actually able to limit data sources. For example, if Alice says her friends cannot see when she is in the kitchen in wing 3B but also her friends are only allowed to know the wing of the room and not the actually room, her friends would never see if she was in 3B if the only room she ever used in the wing was the kitchen. One could imagine giving users the ability to limit types of data sources if many different types of data sources are used for the location detection, thereby giving the user direct control not only over the information other people see, but also the accuracy of the information.

Back channels One of the key challenges we encountered was preventing back channels. In information systems, users are often able to infer more information than they are explicitly given. An example of such a back channel involves the release of a rule. If Bob knew Alice had a rule that prevented him from knowing she was on the 4th floor, and he was returned no information when querying for Alice's location, Bob could guess with reasonable certainty that Alice was on the 4th floor. For this reason, users of Janus's Map are cautioned to keep their rules private. It may, however, still be possible for Bob to determine Alice's rules over time by aggregating personal observations with the information he was given by Janus's Map.

An additional back channel we face with Janus's Map is what we call "The Digital Rights Management Problem." One major problem with digital media is file sharing. As a result, many companies have made an effort to use technical means to prevent files from being shared, such as watermarking. However, once someone can execute the media there are many ways to circumvent this protection. One such attack would be hooking the speaker port into the microphone port and recording a new copy of the music that can be shared. Industry has, in effect,"raised the bar" on media sharing so that it requires more effort on the part of the media pirate. A location information system would be faced with a similar problem. For example, Alice states that Bob can only see her events on

the day that they occurred. There is nothing to prevent Bob from using other means to create his own log of Alice's events over time. Therefore, in the system design it is important to "raise the bar" for this kind of attack.

We now discuss work that is related to Location Information Systems.

6 Related Work

We are not aware of any prior work that aims to provide a general formal system, methodology, and case study for exploiting a BAS to implement an LIS. The vast majority of work concerning BASs focus on administration as opposed to access by average building users. However, there is a substantial literature associated with each of the key components of this endeavor. We therefore survey some of the work on ownership, location detection, and privacy for location systems.

Ownership The concept of ownership in the formalism here is technical not legal, but legal ownership does provide some guidelines for what the technical definitions might need to be, so it is worth making a very brief survey of some of these. In 1995, The European Union passed legislation concerning the use of personal information [1]. One of the most important parts of this law was that users had the right to object to the use of their information. In essence, the EU elected to give users ownership over their personal information, in that entities are required to ask users how to use their personal information. E-mail is assigned ownership in many jurisdictions including the U.S. Most service providers have a policy that states that they will not release a user's e-mails to anyone, giving the user ownership over their e-mails. If they would like this information to be read by other people, it is the user that is in charge of delegating rights to the e-mail, by forwarding the e-mail to others. Since the The Health Insurance Portability and Accountability Act (HIPPA) of 1996 was passed by the U.S. Congress passed, users have also been given some amount of control of their medical information such as the right to ask what sort of information is being stored about them and correct errors in this information. Additionally, the importance of ownership for location information has also been widely accepted. For example, the Wireless Privacy Projection Act of 2003 [2] requires cell phone providers to receive approval from users before their location information can be used.

Location Detection Global Position System (GPS) [3] is one of the most common location detection systems. It uses satellites to determine location and works best outdoors when there is line of site to the satellites. Because of this, GPS is not effective indoors and cannot be used for location tracking buildings. AT&T Labs Cambridge developed the Active Badge system [4] and Bats [5] for indoor location detection systems. Active Badges transmit an identifier to building sensors using infrared and Bats perform a similar function but use ultrasonic pings. Both systems attach sensors to things that are to be located and also include infrastructure sensors. Active Badges provide room level accuracy and bats provide accuracy as good as four centimeters. These systems are in an experimental stage of development. In a separate project, we have been exploring the use of Ubisense (`www.ubisense.net`) as an LIS that handles both people and tagged objects and supports workflow monitoring. Nibble [6] is a location service that

uses an 802.11 wireless infrastructure to determine room level locations. By measuring the signal strength from different access points, Nibble is able to estimate the location of the laptop on which it is running. This provides interesting potential as a foundation for an LIS, although it does require users to carry laptops and advertise the location information they collect.

Approaches for Location Privacy A survey on this topic can be found in [7]. We augment this survey with some additional references. Although it does not concern location privacy *per se*, Graubart's Originator Controlled Access Control (OrCon) [8] is similar to the concepts used in Janus's Map. OrCon requires the data collector to request the originator's permission before data can be disseminated. OrCon seems to especially applicable to privacy systems because the originator and collector of data are often different. This is the case in the Janus's Map system where the originator can be consider the person who caused an event and the collector is the back end door lock system. Gunter *et al.* [9] present a formal privacy system inspired by and applied to location based services. This formal system allows users to issue licenses to allow subscribers to use the location information of the user. This approach is similar to the system presented here because both systems put the power in the users hands. The main difference between [9] and the current work is the level of detail at which the formalism describes how to create a license of this type. While [9] is quite general and aimed at many types of privacy-constrained applications, the model we have provided in this paper is focused on a specific application (LISs based on BASs) and provides somewhat more detail. Snekkens [10], takes similar approach to [9] to privacy policies; he recommends a central server stores information concerning user rules about the release of their location information and allows for fairly complex rules for users to specify the accuracy and detail included in released information. Like [9], Snekkens presents a fairly general system. Snekkens' focus is on the development of a language to define privacy policies while the current work focuses on mechanisms for enforcing privacy policies, in fact, one could imagine using the language presented by Snekkens to aid users in defining their privacy policies for an LIS.

7 Conclusion and Future Work

In this paper we investigate the use of Building Automation System sensors to build Location Information Services. Currently, BAS data is kept secret because of privacy concerns. For this reason, we consider privacy of paramount importance when developing an LIS. To this end, we presented a general mathematical model for developing privacy sensitive LISs. This model follows the following three basic steps: Define an ownership model, determine environmental events, and develop a system for privacy-sensitive information sharing. The most important focus of this model is to put users in control of data they own, because it is about them. The model allows users to define filtering and masking policies that control the access to their data. Additionally, because BAS events are generally low level, the LIS must also aggregate these events into environment events.

We then present an instantiation of this model for the prototype LIS, Janus's Map, to show the feasibility of creating an LIS for a building. Janus's Map uses the electronic door lock systems and the occupancy sensors to give users an approximate location of others in a building. As per the model, Janus's Map allows users to specify access permissions for events that they own. Users are not only able to specify who can find them, but are also able to limit the accuracy of any information released about them. In addition to discussing how this model could be instantiated for Janus's Map, we also present a discussion of the software architecture as well as a discussion of the challenges of preventing back channels in an LIS.

In the future, we hope to integrate more systems into Janus's Map. By doing this, we feel we could significantly improve the accuracy of system. Additionally, we would like to get feedback from users of the system as to its usefulness and accuracy. Additionally, because rooms have multiple owners, we would like to explore policy merging to resolve conflicting rules.

Acknowledgements

The authors would like to thank Adam Lee, Erin Chambers, Chuck Thompson and anonymous reviewers for their help and insightful comments. The work was sponsored by a grant from the MacArthur foundation and NSF grant CCR02-08996.

References

1. The European Parliment and the Council of the European Union: Directive 95/46/ec of the european parliament and of the council of 24 october 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data. (1995)
2. 108th Congress: HR 71: The wireless privacy protection act. In: United States House of Representatives. (2003-4)
3. Getting, I.A.: The global positioning system. IEEE Spectrum **30** (1993) 36–47
4. Want, R., Hopper, A., Falcão, V., Gibbons, J.: The active badge location system. ACM Trans. Inf. Syst. **10** (1992) 91–102
5. Ward, A., Jones, A., Hopper, A.: A new location technique for the active office. IEEE Personnel Communications **4** (1997) 42–47
6. Castro, P., Chiu, P., Kremenek, T., Muntz, R.R.: A probabilistic room location service for wireless networked environments. In: UbiComp '01: Proceedings of the 3rd international conference on Ubiquitous Computing, London, UK, Springer-Verlag (2001) 18–34
7. Görlach, A., Heinemann, A., Terpstra, W.W.: Survey on location privacy in pervasive computing. In Robinson, P., Vogt, H., Wagealla, W., eds.: Privacy, Security and Trust within the Context of Pervasive Computing. (2004)
8. Graubart, R.: On the need for a third form of access control. In: Proceedings of the 12th National Computing Security Conference. (1989) 296–303
9. Gunter, C.A., May, M.J., Stubblebine, S.: A formal privacy system and its application to location based services. In: Privacy Enhancing Technologies (PET). (2004)
10. Snekkenes, E.: Concepts for personal location privacy policies. In: ACM Conference on Electronic Commerce. (2001)