

# A Policy-Based Location Identification Architecture for Pervasive Systems

Sherif G. Aly  
The American University in Cairo  
Cairo, Egypt  
sgamal@aucegypt.edu

## Abstract

*In this paper, we present a policy-based architecture to be used in identifying the location of users within a pervasive system environment. Mobile users can define their own policies governing the granularity of their location identification. Such granularity is governed by a combination of policy predicates and premise topology information. Privacy can vary in a spectrum of levels from total obstruction to total transparency of location information. Sensors and agents coupled with such sensors continuously forward user locations and policies to a policy and state repository. The policy and state repository can be queried for user location information. Location information is filtered to comply with user defined policies at the agents closest to the source of the query. Historical information is eventually archived at the policy repository, and can also be used to query for possible predictions of futuristic location of users.*

**Keywords** Pervasive Systems, Location Identification, Policies, Topology, Privacy

## 1. Introduction

Intelligence about location identification is but one of many fields making up pervasive systems. In pervasive systems, environments rich in communication capability are gracefully integrated with human users [1]. Various communication-capable devices become more intelligently aware of their surrounding environments, and capable of interacting and blending in with the ever changing surroundings.

A key characteristic of pervasive system environments is the presence of a large number of sensors capable of capturing information about various devices. The more information there is about the activity of various devices, the more intelligent

such mobile devices can interact with one another and naturally blend with their surroundings. Such characteristic allows both the devices and the applications operating the sensors to become context aware. The essence of the vision of pervasive computing was to create environments rich in computation and communication capabilities, yet gracefully integrated with human users [2].

Location sensing devices can provide pervasive systems with information such as the location of people and other devices [3]. The proper identification of locations of various users existing within a pervasive system opens the door for a very wide range of applications capable of intelligently interacting to the mere presence of users within given locations. Examples of applications reacting to user locations could be as simple as both queries of user locations, and modifying the context of a physical area to match the preferences users entering such area, or even more complicated to include tracking and analysis of user locomotion patterns, and reacting to dynamic user locations relative to one another.

However, upon mentioning location identification of users within a pervasive system, one should never neglect user privacy needs, as well as the autonomy of users deciding whether it is appropriate for others to track their location or not.

Relevant research related to location context awareness as well as location identification has been undergone. In [4] the need for context awareness and context management was significantly emphasized, where pervasive systems must have some perception to track a person from location to the other. In [5], sources of location information have been categorized into either physical, virtual, or logical. Physical location information relates to location information gathered from physical sensors. Virtual

location information on the other hand is extracted from virtual entities such as networks, operating systems, and software applications. Finally, logical location information is inferred from both physical and virtual location information.

In [6], Chen developed a framework named BlueLocator for enabling enterprise location-based services, as well as locating and tracking enterprise users. In [7], a research effort was made at IBM research laboratories to develop an infrastructure that supports location-aware services. The infrastructure is based on a proposed location operating reference model.

Henricksen in [8] developed appropriate context modeling that can eventually be used in a context management infrastructure. The model also tackled issues related to wide variations in information quality. In [9], Myllyamaki proposed a methodology for aggregating location data from multiple sources, and in [10], a domain specific context awareness application was developed.

In this article, we present a policy-based architecture for user location identification within a pervasive system supporting privacy needs. Distributed sensors around the premise of an organization continuously send user location information to a repository through an agent coupled with the sensor itself. Users within the system can govern their own privacy through a set of user-defined policies also stored in the same repository. The location of users within the system can be queried, however, it is the user-defined policies that govern which information can and cannot be displayed as part of the submitted query using a query filtration process. Analysis could be performed on historical user data to conclude user location patterns.

This article describes the architecture, including the various sensors, the software agents, the policy definition and propagation, the policy and state repository, messaging, state filtration, and some proposed prediction techniques.

## 2. System Architecture

The policy-based pervasive architecture introduced in this article allows for a policy-based tracking and querying of the status of mobile users distributed across the premise of organizations. Through a collection of detection sensors, distributed software agents, and well defined policies, users are able to

query the status of each other within an organization to answer questions related to the both the current and historical locations of mobile users. Based upon historical tracking information, queries can also be performed to predict the futuristic motion track and activities of mobile users.

On the other hand, and in order to control and protect privacy, mobile users themselves can define policies that allow for their privacy protection. As an example, mobile users can set policies that allow other normal users to know only about their existence in a building, or within a specific premise, in an attempt to hide their exact location. Mobile users may also set policies to entirely obscure any access to their current status.

Figure 1 shows a schematic of how a mobile user is detected by a location sensor, after which the detection is propagated to an agent coupled with the sensor itself. The agent then propagates the state of the mobile user to the policy and state repository. The mobile user can also set a policy, which is then transferred to the agent of the detection sensor, after which the policy is also propagated to the policy/state repository. The policy and state repository thus contains the latest up to date state of mobile users, along with all defined policies. Any user can then query its closest agent for the state of another mobile user. The query is propagated by the agent to the policy and state repository, which responds back with the raw result of the query itself, along with the relevant policy. It is up to the agent that received the query at first to then filter the result of the query in a manner that satisfies the given policy and the privileges of the querying user.

A topology repository containing information about the spatial hierarchy of the premise in which this system is utilized in is also used. For example the topology repository will contain information about the relationship between rooms within a building, and buildings within a campus. By storing such topology information in a tree like hierarchy, the system is capable of controlling the granularity of privacy information as set by the user defined policies.

The topology repository is queried by software agents to obtain information about the topology of the premise of an organization. As explained later in this article, topology information, along with user-defined policies determine how the status of queried

users will be filtered to comply with the user-defined policies.

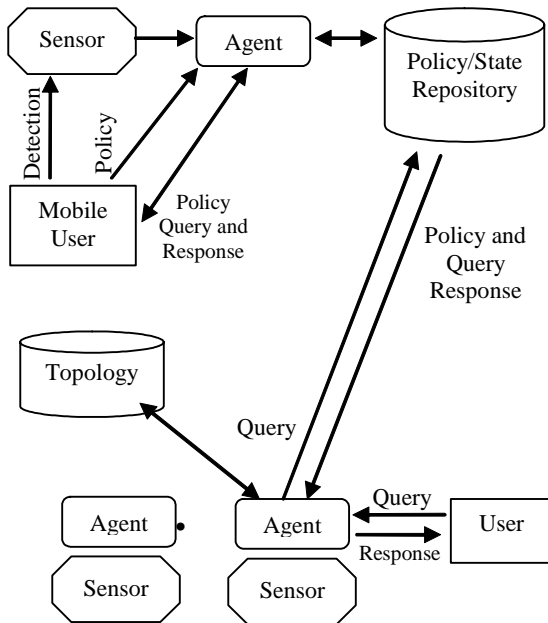


Figure 1 System Architecture

### 3. The Sensors and Agents

In order to have the ability to concisely determine the location of various mobile users across an organization, the usage of location sensors is required. Location sensors used for indoor environments must possess several important properties [1]. Location sensors should be able to provide very fine grain spatial information, and at a very high update rate. However, on the other hand, due to the need to operate in a very fine grain fashion, many locations sensors are required to precisely pinpoint the location of a given mobile user. For such reason, location sensors must also be cheap, and scalable.

Indoor environments are generally challenging in precise location determination [1]. Outdoor radio-based systems such as GPS, are usually successful in outdoor, wide areas, but are usually afflicted by serious multipath effects when used indoors. Other electro magnetic methods used indoors suffer lots of interference from electromagnetic emission devices such as monitors and televisions, and are also affected by metal structures. Optical systems on the

other hand suffer from line of sight problems, especially in the presence of opaque obstacles. Ultrasound-based location sensors on the other hand possess very desirable properties [1].

Within this framework, the assumption that cheap ultrasound location based sensors is distributed amongst the premise of an organization so as to provide a cheap and scalable mechanism of accurately pinpointing the location of various mobile users across an organization.

Each sensor is coupled with a software agent. Agents have five primary responsibilities:

1. Upon detection of the location of any mobile host, the sensor immediately forwards the detection information to the coupled agent. It is up to the agent to then forward the detection information to the policy/state repository so that there is always up to date information at the policy/state repository about the location of users.
2. Users attempting to query the status of other mobile hosts need only submit their request to the agent closest to them. The agent will then forward the query requests to the policy/state repository. The first responding agent to a query request is assumed to be the closest agent.
3. Mobile hosts defining privacy policies for themselves send their policies to the closest agent, which then forwards the updated policy along with the mobile host identification to the policy/state repository.
4. Mobile host requests for querying their own policies, and query responses are also sent through the closest agent.
5. Agents are also responsible for filtering query results to comply with policies defined for the mobile user.

The presence and activity of the agents is crucial for relieving mobile hosts and other querying users from unwanted long range communication, processing, and storage overheads required to implement this system, and eventually conserving very valuable power resources of the resource-limited mobile hosts. The first responding agent to

the mobile user is heuristically considered the closest agent.

As indicated earlier, a semi-dense network of cheap location sensors is required to provide fine-grain mobile host location information. The density of location sensors guarantees close proximity of mobile hosts to the sensor-coupled agent.

#### 4. The Policy/State Repository

The policy and state repository is responsible for storing mobile host policies, as well as current and historical information related to the location of mobile hosts across the organization.

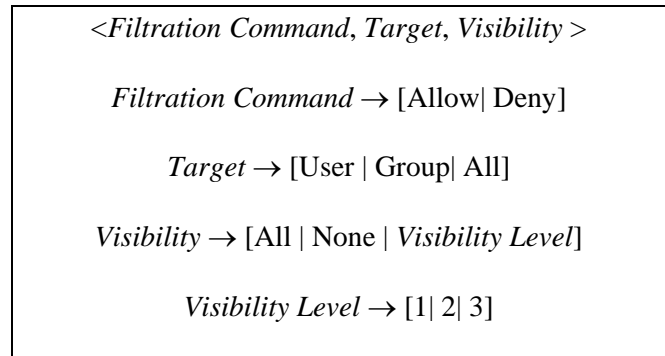
Every time a location sensor detects the presence of a given mobile host, the coupled agent to the location sensor immediately sends location information to the policy and state repository. Furthermore, such repository will also store various mobile host policies, and responds to mobile host queries sent through the agent to inquire about already defined policies.

When a new location detection of a mobile host is sent from a relevant agent, the policy and state repository will move the previously stored current state into a pool of historical data. Such historical data is very valuable in performing prediction of the futuristic location of users. Periodically, the policy and state repository will invalidate and archive historical data.

The policy and state repository can be queried through various agents for the current, historical, and possibly the futuristic state of other mobile hosts. The repository will then respond back to the querying agent with the result of the query, along with the policy set for the mobile host being queried. It is then up to the querying agent to filter the query result to be in consistency with the defined mobile host policy. Such filtration functionality is delegated to the agent in an attempt to offload processing at the policy/state repository.

#### 5. Policy Definition

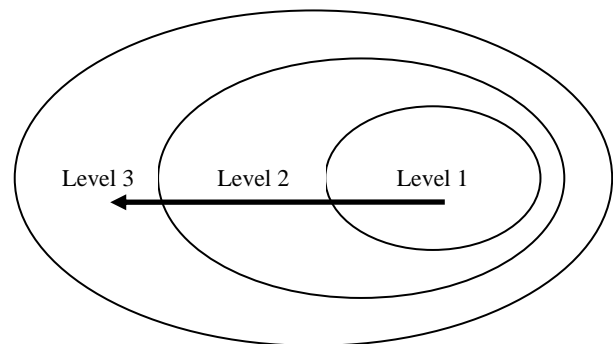
Mobile users can define their own privacy policies and register them at the policy/state repository via their closest agent. Such policies contribute in the query filtration process of various mobile users. Policy predicates are created and passed to the policy repository. Policy predicates are of the following form:



**Filtration Command:** Defines whether this predicate is made to allow or deny a given query.

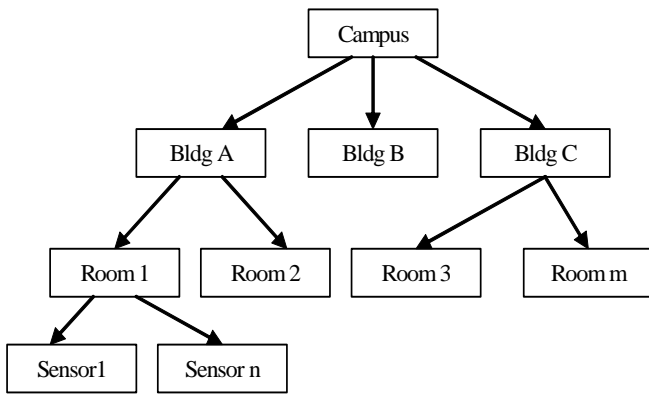
**Target:** The application of the filtration command can apply on a single user, a group of users, or all users.

**Visibility Level:** Either to allow full access, no access, or one of three predefined levels of access. Level 1 access implies the immediate area the user is in can be returned in the query. Level 2 implies a larger area that the user is in, and level 3 implies a larger area as shown in Figure 2 below.



**Figure 2: Visibility Levels**

In order to apply the concept of various visibility levels, an initial topology of the premise in the form of a tree hierarchy of the layout of the sensors is established as shown in Figure 3. The following is an example of a hierarchy:



**Figure 3: Premise Topology Example**

As an example, given the hierarchy above and:

- Given sensor 1 detecting the presence of a mobile user.
- Given a policy definition: <Allow, All, 3>

Then the result of a query made by any entity for the detected mobile user will only indicate that the user is on Campus. Alternatively, if the visibility level was 2, the query result will indicate that the queried user is Building A of the campus. Alternatively, if the visibility level was set to 1, then the result of the query will indicate that the user is in Room 1 of Building A of the campus. A visibility of “All” in the above example would also give the same result.

## 6. Messaging

The following table shows some message formats for:

- Host detection by the location sensor
- Querying the status of a given mobile host (current, historical, and prediction). This is sent from a requesting user to the agent, and then to the repository.
- Status Query responses (current, historical, and prediction) sent from the repository to the agent, and back to the requesting user.

- Policy registration performed by a mobile host to register a policy. This message is sent from the mobile host to the agent, and then to the repository.
- Policy deletion performed by a mobile host to delete a policy, and is also sent from the mobile host to the agent, and finally to the repository.
- Policy query performed by a mobile host to query for its currently registered policy status. This is sent from the mobile host to the agent, and then to the repository.
- The response for the policy query sent by the repository to the agent and eventually back to the mobile host.

**Table 1 Messages**

Message Type	Format
Host Detection	Detect(HostID, Location, TimeStamp)
Current Status Query	CurrentQuery(SourceID, HostID)
Historic Status Query	HistoricQuery(SourceID, HostID, From, To)
Prediction Status Query	PredictionQuery(SourceID, HostID)
Current Query Response	Response(HostID, Current Location, Policy)
Historic Status Query	HistoricResponse(HostID, [] Historic Locations, Policy)
Prediction Status Query	PredictionResponse(HostID, [] Predicted Locations, Policy)
Policy Registration	Register(HostID, Policy Definition)

Policy Deletion	Delete(HostID, Policy Definition)
Policy Query	Query(SourceID, HostID, POLICY)

## 7. Path Prediction

Two primary techniques are considered for path prediction. Prediction of a location path could either be done based upon historical data points representing the location of the user during a similar time, or upon extrapolation of the spatial alignment of various data sensors.

The former technique namely, the prediction of location path based upon historical data points representing the location of the user during similar times proves to be more accurate. If a given user has a history of entering the building every day in the morning to go to their office, it is very highly probable that the same user will take the same route the next morning, and hence, the path of the user during that time of each morning can be predicted with some degree of accuracy.

The latter technique relies on preexisting knowledge of the spatial distribution of various location sensors, and the spatial organization of the building itself. This technique proves to be very sensitive to modifications of the locations of such sensors, as well as modifications in spatial layouts of the building.

Prediction need not provide a single definite result, but rather, may also provide multiple predictions of path with various probabilities of occurrence. Such prediction is subject to future work.

## 8. Conclusion

In this paper we presented a policy-based architecture for the purpose of identifying the location of users within a pervasive system environment. Sensors continuously detect the location of users within the premise of an organization. A software agent coupled with each sensor forwards such location information to a policy and state repository. The policy and state repository always contains the latest up-to-date information about the location of users.

In an attempt to protect privacy, users can define policies that govern the granularity of their location identification. Privacy can vary in a spectrum of levels ranging from total obstruction to total transparency of location information.

## 9. References

- [1] G. K. Mostefaoui et. Al , “Context Aware Computing: A Guide for the Pervasive Computing Community”, The IEEE/ACS International Conference on Pervasive Services (ICPS’04), Lebanon, 2004.
- [2] M. Satyanarayanan, “Pervasive Computing: Vision and Challenges”, *IEEE Personal Communications*, pp 10-17, 2001.
- [3] A. Harter et. Al, The Anatomy of a Context-Aware Application, In *Mobile Computing and Networking*, pp. 59-68, 1999.
- [4] D. Saha and A. Mukherjee, “Pervasive Computing: A Paradigm for the 21<sup>st</sup> Century” *IEEE Computer*. Pp. 25-31, March 2003.
- [5] J. Indulska and P. Sutton, “Location Management in Pervasive Systems”, The Workshop on Wearable, Invisible, Context-Aware, Ambient, Pervasive, and Ubiquitous Computing, Australia, 2003.
- [6] Y. Chen, et. Al. “BlueLocator: Enabling Enterprise Location-Based Services”. IEEE International Conference on Mobile Data Management, Singapore, 2002.
- [7] Y. Chen et. Al, “LORE: An Infrastructure to Support Location-Aware Services”, *IBM Journal of Research and Development*, Vol. 48 No. 5/6, September/November, 2004.
- [8] K. Henriksen et. Al., “Modeling Context Information in Pervasive Computing Systems”. First International Conference on Pervasive Computing, Switzerland, 2002.
- [9] J. Myllymaki and S. Edlund, “Location Aggregation from Multiple Sources.”, IEEE Third International Conference on Mobile Data Management, Singapore, 2002.
- [10] K. Cheverst et. Al., “Developing a Context Aware Electronic Tourist Guide: Some Issues and Experiences”. Conference on Human Factors and Computing Systems, Netherlands, 2000.