Personal Location Agents for Communicating Entities (PLACE)

by

Justin Lin

Submitted to the Department of Electrical Engineering
and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2002

Author .....
Department of Electrical Engineering
and Computer Science
May 20, 2002

Certified by .....
Robert Laddaga
Research Scientist, MIT Artificial Intelligence Laboratory
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Theses

Personal Location Agents for

Communicating Entities (PLACE)

by

Justin Lin

Submitted to the Department of Electrical Engineering
and Computer Science
on May 20, 2002, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Traditionally, location systems have been built bottom-up beginning with low-level sensors and adding layers up to high-level context. Consequently, they have focused on a single location-detection technology. With sharing of user location in mind, I created Personal Location Agents for Communicating Entities (PLACE), an infrastructure that incorporates multiple location technologies for the purpose of establishing user location with better coverage, at varying granularities, and with better accuracy. PLACE supports sensor fusion and access control using a common versatile language to describe user locations in a common universe. Its design provides an alternative approach towards location systems and insight into the general problem of sharing user location information.

Thesis Supervisor: Robert Laddaga
Title: Research Scientist, MIT Artificial Intelligence Laboratory

Acknowledgments

I am most grateful for suggestions, guidance, and supervision from Robert Laddaga; for suggestions and assistance from Hirohisa Naito; and for suggestions and support from Howie Shrobe. I would like to thank Krzysztof Gajos, Stephen Peters, and Andy Chang for their advice and assistance. I also would like to thank everyone in the Intelligent Room for providing for such an exciting and friendly work environment. Finally, I am grateful for support from Rod Brooks and the MIT Oxygen Alliance Partners: Acer, Delta, Hewlett-Packard, Nokia, Philips and NTT.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The rapid growth of power in computers over the past century has led to a popular notion that computation is becoming free. MIT Project Oxygen recognizes that computers have evolved from large expensive machines to small cheap desktops and portable devices [32]. The project asserts that this evolution will carry through into the future where computation will be pervasive, as free as the "oxygen in the air we breathe." Computers will be so tightly integrated into people's lives that they will be invisibly performing services for users at all times. Performing these services for the user, but without user input, will require the computer to be aware of the context surrounding the user at any given time. In this way, the pervasive computing movement calls for a wealth of contextual information in order to perform the appropriate tasks for each person in each of their possible situations [12, 2].

Of this context, location has become one of the most popular topics among research groups. This increase in location technologies has produced a large number of mobile (location) applications that fall under two categories: (1) users performing context-based tasks with his/her own location and (2) users performing context-based tasks with the locations of other users. The first category includes reminder services [13, 27], navigational services [36, 26], location-aware information delivery [27], environment customizations [37], location-based tour guides [15, 22, 31], and general context engines [14]. However, the second category remains largely unexplored. Some research groups have ventured into this domain (e.g. ActiveMap [28]), but possibly

due to the daunting privacy concerns, there has been relatively little research in this area. In this paper, we demonstrate the potential behind sharing one's location with others, given a location system infrastructure designed specifically for this purpose.

In addition to location applications, the exploding interest in location context has resulted in a multitude of location detection technologies developed by many different groups, each of which seems to contribute something unique and interesting to the field [21]. Individually, these systems are useful, yet each faces the limitations of their methods. Working together, they can obtain location information with better coverage, at varying granularities, and with better accuracy - three qualities particularly useful in sharing user locations. Consider how people generally solve such problems. When we need to locate some object or place, we refer to several sources of location information, by asking multiple people, referring to multiple maps, using GPS receivers, and other methods. In doing so we transparently interpret multiple coordinate systems in terms of each other, resolve disparate symbolic references, and gauge the probabilities associated with each bit of evidence. These things that we do with little thought or effort are incredibly difficult for our computer systems to do. In light of this complexity, we can see why there has been little effort to bring the unique and interesting features of various location systems together into one collective system.

One of the first and most difficult problems in creating a unifying location system lies in finding a common location representation that allows for collaboration between the location systems. Referring back to our human example, we see that the problem is really that of figuring out how to translate between differing underlying representations. Rather than trying to solve this problem for all location services, we concentrate our efforts in investigating the possibilities for a common location representation specifically for sharing *user-locations*.

A second and equally difficult task relates to access control. As computers become more integrated into our daily lives, people grow more concerned about privacy and security. Automated tasks are becoming increasingly popular, yet many of these tasks require users to provide personal information. Thus, one of the obvious challenges in building a system dealing with the exchanging of location information between users

is to provide users with full control over the distribution of their locations. Just as P3P [10] seeks to offer users complete control over their personal information on Web sites they visit, a location system must offer users complete control over their location information.

This thesis aims to provide insight into an alternative approach toward user location systems, namely building an infrastructure with location-sharing in mind from its conception. It describes Personal Location Agents for Communicating Entities (PLACE), an infrastructure that utilizes (1) a semantic representation of location as a means to attain fusion of sensors at varying granularities for better coverage and accuracy, and (2) an intermediary software agent between location devices and location services that performs sensor fusion and access control on behalf of the user. The thesis begins with a brief generalization of current location systems and the details and motivations behind our general design. It moves to a discussion on how to achieve a common universe of locations to allow for clear communication of location information between a world of entities. The thesis ends with suggestions for future work, brief overviews of applications that capitalize on our design, and some concluding thoughts.

1.1 Related Work

There exists a multitude of location systems today. Each system typically chooses a location representation that suits the technology used in determining one's location. For example, the Global Positioning System (GPS) uses multiple satellites that provide specially coded signals to a GPS receiver, which then can use these signals to compute it's location in latitude-longitude coordinates. Further, the Active Badges location system places a base per room that detects via infrared communication when badges enter the room, thereby providing a symbolic location representation, namely which room a user is in [16].

Each location system generally has both unique features and limitations in terms of technology, accuracy, scalability, and cost. Because these systems are usually

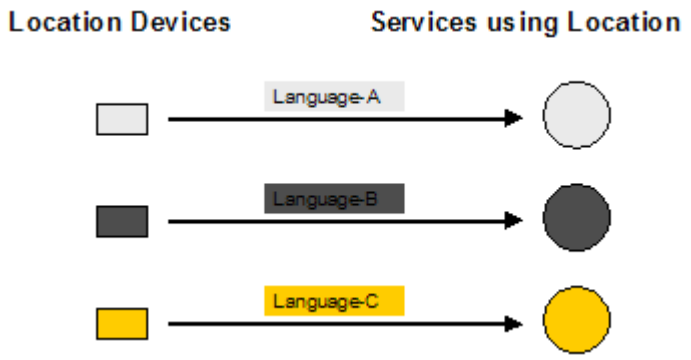

Figure 1-1: Conventional Method

developed in separate research facilities with different views on the ideal location system, there appears to be little effort to have two or more location systems coexist and collaborate with each other. Consequently, applications built to utilize location technologies have generally limited themselves to one location system, as shown in Figure 1-1. Thus, most applications that exist today face the limitations of their chosen location systems.

Korkea-aho and Tang [23] have made a similar observation that each location application seems to create its own representation. They aimed to "create a simple lowest common denominator data set that as many location information sources and applications in the Internet as possible could use." The result was called the "Common Spatial Location Data Set," consisting of geodetic latitude, longitude and altitude, accuracy and time of measurement, speed, direction, course, and orientation. This is a sensible solution for location systems in general; however, in the context of pervasive computing and describing locations of users, this geographic spatial location representation would not be ideal. People generally tend to reason about locations as semantic places rather than as latitude-longitude coordinates.

Furthermore, various specifications have been proposed. First, as specifications for describing geographical objects, maps, etc. G-XML [5] and GML [9] should be noted. They create a standard for encoding geospatial data into XML documents. Likewise, POIX [8] and NVML [36] are specifications for describing positions of moving and

static objects and encoding this information into XML. Both sets of specifications are based on common location expressions and add additional vocabulary depending upon their objective. There are some ongoing activities attempting to standardize coordinate representations for location expressions. However, they have not attempted to standardize symbolic representation for expressions of location information.

SignalSoft Corporation [11] has also attempted to utilize multiple location sources in their product suite. The Location Manager receives data from multiple location sources and delivers the data to location applications. They have successfully created a commercial product but restrict themselves to cell-phone-based location detection technologies. Furthermore, these cell-phone-based technologies use a spatial location representation. Again, we believe that this is not optimal for sharing user locations and instead utilize a semantic representation in our location system described below.

Multisensor data fusion is a field that has a huge amount of attention and numerous published results [19]. However, most work in the field is based on a signal processing, bottom up approach. We favor an approach that proceeds top down from a relevant decision problem, as an approach to filter away much of the irrelevant, ambiguous, or conflicting data.

Access control is another highly developed and explored area, with a great deal of literature. For specification of access control for purposes of maintaining military secrets, see [29]. For general discussion of security, confidentiality and information integrity issues, see [1, 24, 6, 18]. Our own efforts are more influenced by Role Based Access Control, access control based on the role of the user requesting the information [3, 33, 30].

Chapter 2

Sharing Location - Conceptual Requirements

In the context of this paper, location sharing is the sharing of information about locations of persons for the purpose of performing some task effectively, such as joining them for a meeting, delivering a package, predicting their near future behavior, and determining location dependent privileges. Sharing location information can be categorized into: (1) systems sharing location information with other systems, (2) systems sharing location information with humans, and (3) humans sharing location information with other humans.

Wherever machines (systems) are involved, we need to be concerned about interoperable data representations. Wherever humans are involved, we need to be concerned about issues of privacy, and the ability to present information in an understandable manner. Note that humans may well have privacy concerns about data residing physically (persistently) on certain machines. Further, even in machine to machine interactions, humans may need to diagnose and debug failures, and hence need to understand data involved in the interactions.

2.1 Descriptions of Locations

Locations are identified by descriptions. In order to accommodate the three modes of communication listed above, a location system must take heed of how both machines and humans describe locations. A location system must support a location representation that captures these descriptions so that both users and machines may participate in the commmunication.

Furthermore, supporting machine to human communication requires that they have a common understanding of locations. In the spirit of pervasiveness and usability, one cannot expect humans to learn the computer's reasoning about locations and to perform the processing behind this reasoning. Rather, the computer should be taught to understand a human's reasoning about locations. As shown in section 3.1, this essentially means teaching the computer about semantics.

2.2 Common Language

Ideally, all location devices and services would use a common, universal location representation and language. Unfortunately, this is not the reality. The vast contrast between location representations that exist today is not merely a consequence of philosophical differences, but rather of the inherent unique features that are specific to each location technology. GPS provides latitude-longitude coordinates, while Active Badges provide symbolic spaces, but both technologies prove to be useful in different scenarios.

Our goal is to allow for bidirectional translation to and from a common language that expresses location in a representation chosen and designed specifically for user-location services. This language serves as a standard, if you will, between communicating entities, but bidirectional translation provides versatility when required.

Because a universal language does not exist today, communication of location information is restricted to be from the location device to the location service (again, designed specific to the location device). With a common user-location language,

location information can be freely exchanged between all communicating entities, device to service and service to service.

2.3 Common Map of Locations

With a common language, entities can communicate location information, but cannot necessarily fully understand each other without having the same map of locations. Upon finding out that a user is in the "John Hancock Building," in order to be able to realize and use this location (e.g. find on a standard map, get directions, build higher-level context), one must have at least a similar understanding of "John Hancock Building" as the provider of the information. For the understandings to be similar enough, we require that the processes by which each user binds an object or location to the symbol, result in roughly the "same" object or location.

2.4 Sensor Fusion

Location is simply another piece of context. Deducing a user's location is an attempt to capture some information about the world. The multitude of location technologies that exist today provides many paths one can take to obtain this information. However, each location technology has its limitations; this motivates us to combine technologies and to "take all that we can get." With this approach comes the problem of sensor fusion. Just as context engines perform sensor fusion to establish context, location systems can perform sensor fusion to establish location.

By using more than one technology, we gain more information; with more information, we have a better understanding on the location of the user. This understanding potentially covers the varying types of location representations; location described in a relative positioning sense, in an absolute positioning sense, and in the logical sense. It also potentially covers more area, considering the indoor/outdoor limitations of location systems such as GPS and Active Badges. Last, understanding of a user's location increases because location information from different sources can reinforce

or conflict with each other and, therefore, addresses the inaccuracies of the location systems.

Perhaps an even more important purpose of sensor fusion and multiple location technologies is making location systems more robust to environmental conditions. Any given location system can fail for any number of reasons, and it will often be the case that backup systems require multiple sources to recover the coverage or accuracy of the primary system. Multiple systems for providing location information, and a flexible architecture that allows for arbitrary combinations of information from disparate sensor sources and other information, is central to a more robust and self-adaptive architecture [35, 25].

2.4.1 A Note on Coverage

A unifying location system infrastructure offers scalability and coverage to location applications. We assume that there are many interesting applications that utilize location information inside a building, requiring a location system other than GPS. Many research groups have already demonstrated possible uses of location information with successful prototypes of their applications built specific to a single location technology. In many situations, this single location technology suffices as the primary source of information inside a building. However, these applications are often constrained to one building, due to their dependency on the indoor location technology.

We do not claim that *all* applications apply to settings larger than a building. For those that do not apply to larger settings, it is still the case that they should eventually consider multiple sources of location information (such as cameras to supplement radio badges). So ideally all location systems should take account of such scalability constraints. A unifying location system *is* necessary, simply because a one cannot expect another user in a different building to have the same location technology at his/her disposal. That is, we cannot expect the same technology to be implemented in all buildings. Thus, a unifying location system offers coverage not only over indoor-outdoor boundaries, but also across different indoor environments. An application that desires to share location information at a larger scope than a single building

requires such a system. An application that makes use of multiple types of location information sources, even in a single building, or that expects to cooperate with external location systems, also needs such a system.

2.5 Access Control

The presence of an intermediary between location devices and location services provides us with a convenient place to perform access control. Traditionally, many location applications have not implemented access control because they do not involve sharing location information with other users. Those that do involve sharing of location information are often based on the user carrying a location device that the user can detach or disable, and thus address the access control issue (e.g. [28]). This sidestepping of access control issues does not adequately deal with common situations in which the user wants some people to have access to location information under certain circumstances. It also fails to deal adequately with non-personal location devices such as cameras, and with location information needed for emergency reasons. A less binary solution, and one that is less device dependent, is called for.

Chapter 3

Sharing Location - Design Issues

The following is a survey of the issues relevant to each of the conceptual requirements. This list is by no means comprehensive, but instead offers some insight into the vast spectrum of problems involved with building a location-sharing system.

3.1 Descriptions of Locations

The essence of information is its meaning. That is, the end goal of receiving information is to determine and interpret its meaning. Without the ability to extrapolate meaning, the information loses its value. Thus, when communicating location information, it is imperative to provide it using a representation understood by the recipient; otherwise, the communication is not successful.

3.1.1 Coordinates vs. Semantics

Coordinate representations (e.g. GPS, GIS) are particularly useful in determining the precise location of an object. In our three-dimensional world, every object occupies a specific space that can be represented with a set of coordinates unique to every other position in the world. Because of this, coordinates are easily understood and distinguished from other locations using computers with high processing power and large databases. On the other hand, humans unfortunately do not carry such resources

to interpret coordinates and extrapolate meaning from coordinates at will.

In contrast, semantic representations offer explicit contextual meaning. Whereas a set of coordinates is not immediately understood by most people, "45 State Street" informs the user of the street and the address number, two pieces of location context commonly understood by people. "Tanner's Bookstore" informs the user of the location's function and, if recognized by the user, the exact place that it refers to. Generally speaking, people do not normally memorize coordinates of places, but they do tend to remember descriptions of places.

One of the drawbacks to semantic representation is its complexity. Semantic names can be entirely redundant, represent overlapping regions of space, or be personal to a particular user or group of users. All of these result from the fact that semantic names are arbitrarily given to arbitrarily defined regions of space. Because of this, the reasoning that handles semantic names is far more complex than coordinate systems.

3.1.2 Human-to-Human Discourse

In normal human-to-human discourse, location is communicated using some sort of semantics, such as "Tanner's Bookstore," "45 State Street," or "my favorite bookstore," etc., as opposed to a set of coordinates. With this in mind, in regards to sharing user-location from user to user, it is reasonable to assume that the recipient would prefer to be presented with a semantic representation of the provider's location (although evidence of this is not in the scope of this paper). Given a semantic representation of a location, a human immediately realizes the meaning of the information (unless the semantics are foreign to the user). Given a coordinate representation of a location, in order to extrapolate meaning out of the information, the user most likely will use some type of service that offers translation from coordinates to a semantic representation (e.g. street address). In the end, the user desires semantics.

3.1.3 Location-Sharing Infrastructure

Thus, in building an infrastructure to support user-to-user communication of user-locations, the ultimate goal is to present the end-user with semantics capturing the meaning of the location. There are two immediate models that will achieve this purpose: provide the semantics themselves, or provide coordinates which the recipient translates to semantics. From the recipient's perspective, the differences between the two models largely coincide with the differences between the semantic and coordinate representations. As previously mentioned, the two representations simply contain different sets of information about the location.

However, from the provider's perspective, the two models have highly contrasting privacy implications. Because coordinates are inherently at a fine granularity, the provider always gives the recipient information about the user's physical location, and at this fine granularity. A possible remedy to this model would be to offer coordinates with less precision to help disguise the point location of a user. However, blunting the coordinates in this way would be restricted to simply providing location with a larger geographic approximation radius, giving no regards to the spacial boundaries (both physical and implicit) that are crossed while expanding the radius. Furthermore, this representation always gives explict clues to the physical location of a user.

In contrast, when using semantics, the provider has the power to variably withhold as much information as desired. In many scenarios, a user wishes to communicate enough information to get the point across, but not any more than necessary. For example, a user may wish to communicate to friends that he/she is in a "place where you can call me." This information is just enough to tell his friends that they can call him if desired. All clues to physical location are restricted to what may be inferred by the fact that the user is reachable by phone in his/her current location. Similarly, "Boston, MA" gives clue to the user's physical location, but only at the granularity of a city. All clues to the user's location at finer granularities are withheld. In this way, by offering only semantics about one's location, one can choose exactly how much information he/she wants to give.

In building an infrastructure for sharing user-location, privacy is of utmost importance due to the general sensitivity to giving out information about oneself. Because of this, the privacy features of a semantic representation make its presence in a location infrastructure essential. The ability to provide explicit contextual information also supports its presence in the system. At the same time, the advantages of a coordinate representation should also be noted. The precise granularity, uniqueness of each location, and minimal ambiguity of a coordinate representation make it more conducive to specific computational tasks that can be performed by the recipient's processing power. Thus, each representation has its unique features and therefore, should be supported by an all-purpose infrastructure for sharing user-location.

3.1.4 Semantic Representation of Locations and Relations

The key motivation to using a semantic representation for user-locations and the relations between user-locations is the desire to match the computer's reasoning about locations to the user's reasoning about locations. Creating a semantic representation that actually achieves this goal has been historically difficult in the field of artificial intelligence - put simply, human reasoning is vastly complex. As demonstrated below, the same difficulty in capturing this complexity applies to location representation.

Semantic Places

Whereas coordinates uniquely identify a point in space, semantic names for places are arbitrarily ambiguous, arbitrarily redundant, and arbitrarily personal.

People refer to places in numerous ways. For example, a bookstore could be referred to by the following:

1. By standard names - "Tanner's Bookstore"
2. By arbitrary names - "my favorite bookstore"
3. By reference - "place we went to last week"
4. By description - "that red brick building with the bright red door"

5. By relation to other places - “the bookstore next to the coffee shop”

In addition, people’s interpretation of what constitutes a place depends on the context and on their personal views. For example, the “Prudential” (in Boston, MA) can refer to :

1. The skyscraper
2. The mall
3. The plaza surrounding the mall
4. Any combination of the above

Semantic Relations

Semantic relations face even greater complexity because they deal with the problems (noted above) of representing each location involved and with the problems of relating two locations to each other.

Containment. The most popular relation used by location systems today is containment. Though a seemingly simple concept, the inherent ambiguity problems of semantics still comes into play here. At first glance, everything can be organized in a strict hierarchy of containment; a room exists inside a building, a building exists inside a city, a city exists inside a state, and so on. However, this model does not take into consideration the following:

1. Ambiguous boundaries
2. Individual perspectives
3. Overlapping spaces and partial containment

Kashmir exists in a region whose boundary has been disputed between Pakistan and India for over fifty years. Because of the ambiguous boundary, the city cannot be objectively placed under either Pakistan of India alone. Furthermore, Pakistan

believes the city remains on their side of the border, whereas India believes otherwise. From their individual perspectives, each country would believe that it contains Kashmir. Last, school zones serve as an example of overlapping spaces because they are often spread across city boundaries. If more than one city partially contains the school zone, then the school zone cannot be placed strictly under one particular city. Additional examples of overlapping spaces include mountain ranges and rivers, each of which can span many countries and/or finer granularity boundaries (states, provinces, etc.).

Nearness. Nearness represents one of the many useful pieces of context relevant to locations. Given his/her location, a user may desire to find nearby restaurants, the nearest gas station, or the nearest printer.

Although people's use of "near" in everyday conversation is regular and casual, the term carries a loaded meaning. That is, the interpretation of "near" in a sentence requires context evaluation in addition to the meaning of the word alone, simply because people use "near" in numerous ways. Upon determining whether two places are near each other, the following three questions must be answered:

1. Near by what metric?

2. How close is near?

First, nearness depends on the metric. "Massachusetts" can be said to be near "New York" by plane or car, but not by walking. It should be noted that each metric requires its own information base. That is, in the same way coordinates can determine geographic distance between two locations, flight paths, car paths, and pedestrian paths must be available to determine nearness for their respective metrics.

Furthermore, nearness depends on what is considered near. Driving in New York City to a destination that is one mile away takes a considerable amount of time. Driving in Dallas to a destination one mile away takes a considerably shorter amount of time and will more often be deemed near than in NYC.

The examples thus far involve two places at the same granularity. However, nearness is even more ambiguous when comparing two places of different granularities,

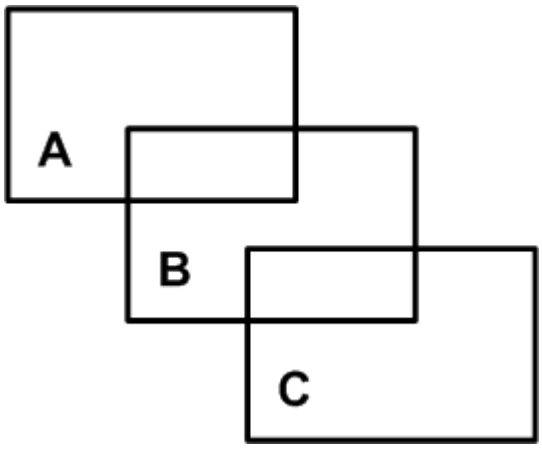

Figure 3-1: Partial Containment

such as "Empire State Building" and "Massachusetts." When someone is in the "Empire State Building" and asks if "Massachusetts" is nearby, there is first the question of what metric should be used. If the pedestrian metric is chosen, the two places most likely would not be considered near. On the state level, if the metric is the number of states separating two locations, they most likely would be considered near since New York and Massachusetts are neighboring states. However, if the automobile metric is chosen, then to what point in Massachusetts should nearness be evaluated? In this way, relating places at different granularities makes nearness even more ambiguous.

Making Inferences from Semantic Information. In addition to the problem of how to represent the data, semantics also introduce significant problems in how to use the data.

For containment, a straight-forward rule can be used to traverse the data. If there is full containment in a hierarchy of places, where A contains B, and B contains C, we can safely infer that A contains C. However, overlapping spaces make inferring containment a much more difficult problem. If there is only partial containment on a hierarchy of places, where A partially contains B, and B partially contains C, we cannot assume that A partially contains C (as shown in Figure 3-1). Therefore, in the case of partial containment, one cannot safely make further inferences via traversing the data.

Making inferences about nearness has its own set of rules. Given that A is near B, and B is near C, one cannot assume that A is near C. Nearness must always be evaluated relative to the original location and cannot be inferred by transitivity.

Thus, for every semantic relation supported by a location system, there must be an associated set of rules for traversing the data.

3.2 Common Language

The first step to bringing all entities into the same universe of locations is to have them understand each others' languages. There are a spectrum of possible solutions between two extreme solutions to the language problem. First, all entities can speak their own language as long as they offer translation to and from all other languages. This represents the more cumbersome (but also more flexible) of the two solutions because it places the responsibility of deciphering all incoming information on each entity. Alternatively, all entities could use the same location representation and avoid translation altogether. However, forcing one location representation onto all entities would entail losing some valuable information that perhaps can only be encapsulated with a different representation. Thus, unless a location system supports both translation when possible to allow for sensor fusion ıand the ability to use information specific to each entity, the system is not fully utilizing the information available. A possible "middle" solution would be to provide a common language to allow for sensor fusion and also translation mechanisms between each of these languages.

Translation in itself is a difficult problem. Because each representation offers unique information about a location, translation often is approximate, leading to ambiguous mapping and loss of information. Despite these drawbacks, translation is worthwhile and necessary, for it is the only means of relating two clues with different representations.

3.2.1 Translation - Ambiguous Mapping

Consider the problem of translation between a coordinate system, such as in GPS, and a semantic representation. A coordinate can belong to many semantic notions of locations. The natural translation scheme is to map a coordinate to all places that contain the coordinate. This scheme then is subjective to all the issues of containment, ambiguous boundaries, individual perspectives, and partial containment, discussed in Section 3.1.4.

Mapping between spaces defined by Cricket and a semantic representation runs into similar ambiguity. Cricket is a location system that uses RF and ultrasound to determine distances between receivers carried by users and strategically placed beacons [34]. Although coordinate representation of locations is supported, Cricket primarily has been used to deduce the *space* holding the user. Spaces in Cricket are arbitrarily defined and not necessarily consistent with people's perceptions of places. Thus, they may not coincide with the semantic notions of places. A possible translation scheme is to map to all semantic places that overlap the given Cricket space, adjusting the certainty to the proportion of overlapping space over the entire space. The scheme, however, is inherently ambiguous.

3.2.2 Translation - Loss of Information

Consequently, once a mapping has occurred, the new representation often loses a particular piece of information that was specific to the former representation. In the case of the translation from coordinates to semantics, consider the mapping from a point located in the Mississippi River to the semantic name, "Mississippi River." Although the original location might be in the Louisiana portion of the river, "Mississippi River" does not contain this information. Loss of information similarly occurs in translation between Cricket spaces and semantic represention.

3.2.3 The Importance of a Good Translation Scheme

A poor translation scheme could do more harm than good. As an extreme scenario, if all the locations in the world given by a particular sensor were translated to Boston, Massachusetts, then this sensor would be misleading for all locations oustide of the city. A more realistic example might be when a coordinate system represented in the translation is misaligned with the GPS receivers by twenty meters. In this case, the coordinates would always be reducing the accuracy of the sensor fusion results.

Because translation is approximate, certainties of every mapping should be represented in every result. Defining these certainties requires much consideration and tuning. Mappings with overly optimistic certainties can provide misleading support for a particular location. Mappings with overly pessimistic certainties can provide less support for a particular location than it deserves.

3.3 Sensor Fusion

3.3.1 Recognition of Conflicting or Supporting Clues

In location systems supporting one location sensor, every new piece of data usually takes precedence over the old data . The most recent location clue essentially determines the user's current location, and the old clues become obsolete. If the new reading has the same value as the previous reading, then the location is assumed to not have changed. Otherwise, if the sensor reading has changed over time, then the user is assumed to have changed locations. In the former case, the clues support each other, while in the latter case, the clues conflict with each other.

With multiple sensors, recognizing conflicting and supporting clues is not as straightforward. Upon having more than one sensor contribute to location detection, there is the problem of relating the clues from each sensor.

Varying Granularities

Two clues can be at varying granularities. If one location sensor provides information at the coordinate granularity, another sensor provides information at the room granularity, and a third sensor provides information at the building granularity, then a location system must contain information regarding the relationships between these places in order for any sensor fusion to happen.

Proximity

Similarly, two clues representing locations close to each other have an ambiguous contribution. The fact that the two locations are near each other reinforces the fact that the user is in the vicinity. At the same time, the fact that the two locations are two distinct places suggests that the user may have moved from the location of the older clue to the location of the newer clue. Again, information about the proximity of the two places must be present in the location system to achieve either interpretation.

3.3.2 Resolving Conflicting or Supporting Clues

Once the clues are determined to be conflicting or supporting, the clues must be resolved in some fashion in order to come to a best approximation of a user's location. Factors such as time, certainty, and granularity offer complexity to such resolution.

Time

Location information becomes stale over time. On one extreme, knowing a user was in a bookstore fifty years ago does not offer much to the user's current location. On the other hand, knowing a user was in a bookstore within the last five minutes gives a strong indication that the user is probably still in the bookstore or vicinity of the bookstore. Context further complicates the situation because knowing a user was in Boston in the last five minutes suggests that the user is still in Boston in most circumstances, but not if the user is on a plane.

Certainty

In contrast to single device location systems, a newer clue does not immediately render an old clue obsolete because the two clues could be from different sensors with different certainties. If two clues arrive close to one another, the older clue comes from a sensor with certainty 0.9, and the newer clue comes from a sensor with certainty 0.1, then there is a good chance that the older clue is the more valuable clue.

Granularity

Last, given two clues that support each other but that are of different granularities, one must decide how to resolve the reinforcement. For example, assume a user is detected to be in the "Empire State Building" and in "New York City." The fact that the user is detected in the Empire State Building should be strong support that the user is in New York City. Furthermore, the fact that the user is detected in New York City should be much weaker support for Empire State Building, but supportive nevertheless. The reasoning here is that given that a user is in New York City, the user is less likely to be in any building outside of the city.

3.4 Common Map of Locations

In the above examples, the semantic names for the places are casually assumed to be unique. Uniqueness is essential when communicating location because entities must not confuse two places with identical names. For example, more than one building can have a room called "Room 800." More than one institution can have a building called "Bldg. NE43." In the relation-map shown in Fig 4-2, we do not explicitly address the potential ambiguities.

One solution would be to include a specification in each name, such as "Cambridge.MIT.Bldg-NE43.Room150." However, this approach has two problems. First, the specification must address all granularities in the universe of locations in order to ensure that there is not a duplicate place specification. That is, we must ensure that the universe of locations does not include more than one room with the name "Cambridge.MIT.Bldg-

NE43.Room800." Second, the specifications imply a strict hierarchy of containment, an unrealistic perception of the world.

The only solution likely to work in the long run, solving most problems, and scaling to very large systems, is one that has the ability to actively translate symbolic information from an outside representation into an internal one, and vice versa. In order for this ability to work, it is necessary for systems to conduct investigatory dialogues to resolve ambiguities, remove conflicts and derive new concepts and associations. This is of course far too difficult, except in very crude approximation, for today's technology.

3.5 Access Control

Location tracking is a delicate issue. The ability to find other people at any time is a very powerful utility; however, the ability for other people to find you at any time is a rather uncomfortable notion. People do not want to be stalked by strangers, suffocated by friends and family, or always locatable by business associates. Therefore, when using a location service, it is important for users to have the functionality to precisely fine-tune the access control to their location information. Though widely used as a standard access control mechanism, Access Control Lists (ACLs) alone do not suffice because they do not take advantage of contextual information that users may find pertinent in their distribution of location information [20].

3.5.1 Context Applied to Access Control

Many types of context can potentially contribute to access control. The following highlights potential uses for access control based on who, why, when, how, what, and where:

- Who is asking?
  - no strangers - keeps undesired people from tracking you.

  - family only - offers more security to parents when they know the locations of their children.
  - travel groups - prevents individual members from getting lost.

- Why does someone want to know?
  - emergencies - be locatable by anyone during emergency situations that are relevant to you.
  - technical support - IT employees only want to be found if there is a legitimate technical support issue.

- When does someone want to know?
  - during scheduled meeting times - let other members of meetings know how far you are from the meeting.
  - during business hours (9am-5pm) only - the ability to quickly find resources is conducive to productivity and efficiency.

- How are they asking, or how is the location information provided?
  - give abstract location information, but not output of a camera.
  - give location information if they are asking over a secure network path.

- What am I doing?
  - sleeping/taking a vacation - regardless of where, do not disclose location.
  - giving lecture - allow all who are interested in your lecture to find out where you are speaking.

- Where am I?
  - not in library - prevent people from distracting you while studying.
  - at home - know when friends get home from school.

- Combination

- no business contacts outside business hours - provide increased productivity at work while not allowing work to travel with you after business hours.
- friends during designated social hours - allow friends to socialize with you only when desired.

The information of who, why, etc. simply contributes to user context, and a location-sharing system can utilize this context to deduce access permissions to a user's location information. If additional context is available, users can add further precision to the access control of their information. Thus, as shown in the medical database domain [38], users can enhance their access control by incorporating context as additional variables and constraints.

3.5.2 Covert Channel Problem

Regardless of the access control scheme that a user utilizes to protect his/her information, one must still be wary of the covert channel problem [29]. A covert channel is a communication path that can be exploited by someone to transfer or receive information in a manner that violates a system's security policy. That is, there may be alternate methods to obtaining one's location. For example, if a user tends to spend the evenings with a friend, then one can obtain clues to the user's location by learning of the friend's location. Also, surveillance cameras gain location information for those who walk in its view. Users have no control over this information. In sum, there does not exist a method for conducting seal-proof access control; rather, the problem is determining what is good enough.

Chapter 4

General Design of PLACE

We propose a design and implementation exploring the possibilities of a unifying location system infrastructure that allows user-location to be commonly understood among communicating entities. Our design goal is to create a user-location infrastructure supporting *sensor fusion* and *access control* using a *common versatile language* to describe locations in a *common universe*.

The ultimate goal behind our unifying location system is to allow for all communicating entities to fully understand each other with respect to location information. In order to accomplish this, these entities must all share a common universe of locations. Sensor fusion cannot operate unless sensors are able to collaborate. Users cannot perform access control based on location context without precisely matching their perceptions of the locations. Communicated location information carries meaning exactly to the extent that there is a common universe of locations shared by the communicating parties.

In the next few sections is a description of the software environment housing PLACE, and of the current design for language, representation, access control, and a common universe of locations.

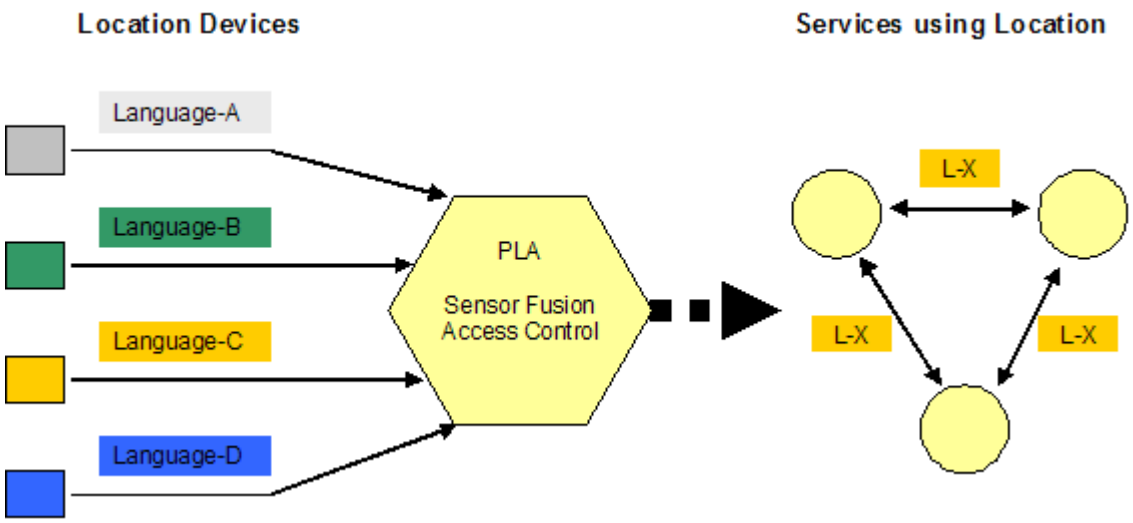

Figure 4-1: Personal Location Agent

4.1 Software Platform

The ubiquitous computing environment within which our location service is designed to function in the MIT AI Lab Intelligent Room [20]. The Intelligent Room is based on a distributed agent infrastructure called Metaglue [7], which provides directory and brokering services to agents. Metaglue supports lightweight agents, and provides an architecture that organizes agents into societies, in such a fashion as to allow intersociety communication. Because the service of sensor fusion occurs on behalf of each user, a distributed, agent-based software platform is quite appropriate, even if the Metaglue architecture did not dictate such a solution.

In the Metaglue environment, agents work within a society, and each user has a society of agents. Each user's society represents the computing environment devoted to performing services for the user. As seen in Figure 4-1, we place the responsibility of sensor fusion and access control (described below) on an agent called the Personal Location Agent (PLA) that acts as an intermediary between location detection devices and location services.

A much less obvious, yet perhaps more important motivation to using an agent-based platform for location is the vision of user-centric computation. Traditionally,

location applications have simply obtained location information directly from the users' location devices, giving essentially no regard to the user as an entity, or the ultimate consumer of location information. This approach runs counter to the new trend away from application-centric computation towards user-centric computation. We propose that by performing location detection and treating location as a property of a user entity, regardless of what services actually use the information, we achieve the notion of computation acting on behalf of the user, as opposed to the application. In doing so, we separate location detection from location utilization.

Indeed, the Metaglue environment in which we have embedded the location services is itself a component of a larger ubiquitous computing environment: The MIT Oxygen prototype system, within which the Intelligent Room functions as a prototype of Oxygen's Enviro21 [17].

4.2 Language

PLACE implements a common default language with which all communicating entities communicate, but offers translation when necessary. The translation is carried out within the Personal Location Agent, in order to localize the translation burden. Even when using translation, it helps to reduce the problem to translating to and from a common representation in the PLA, rather than use a cross-product translation approach. Though a universal location representation for all entities and for all situations would inevitably be inadequate, a common language for all entities but for only communicating user-location seems much more reasonable. We choose to represent locations with semantic names for places and the various relations between places, simply because people seem to usually reason about locations in this way. Reasoning about location information is central to our purpose.

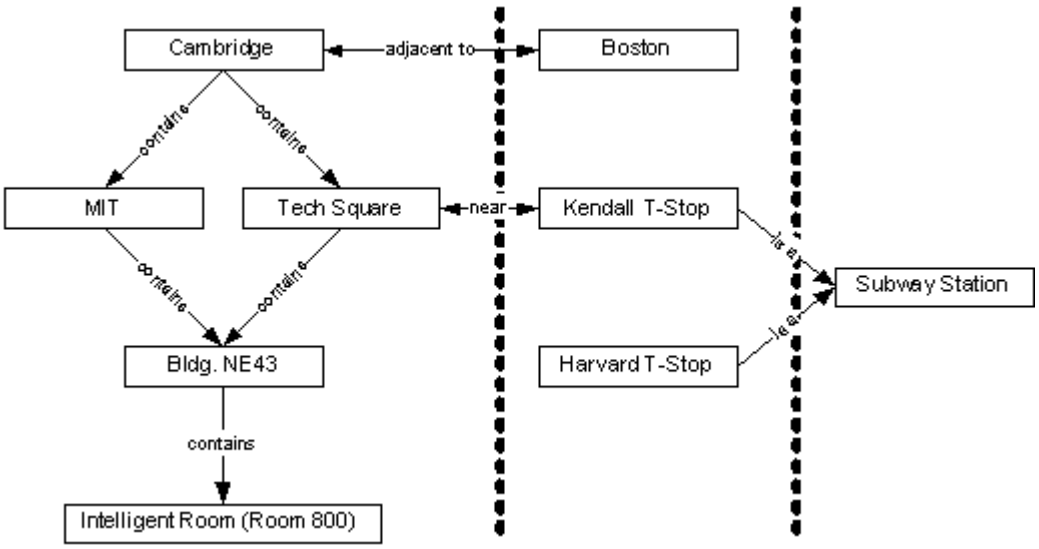

Figure 4-2: Example of a Relation-Map

4.3 Relation-Map

Semantic names alone offer little with respect to a unifying location system. It is the semantic (e.g. geographical, physical) relations existing between the places that offers great value to our system. This knowledge representation for locations comprised of semantic names connected by semantic relations is called a *relation-map*.

4.3.1 Collaborating Sources at Varying Granularities

Without relations, a unifying location system cannot interpret location information at varying granularities as supporting or conflicting information. Consider Figure 4-2 showing a portion of a relation-map. If two location devices return "Intelligent Room" and "Bldg. NE43," the system can determine that the clues support each other only if it knows that "Bldg. NE43" contains "Intelligent Room." Otherwise, "Bldg. NE43" and "Intelligent Room" compete with each other, and the user will be concluded to be either in one or the other.

4.3.2 Building Context

Furthermore, PLACE uses a relation map not only to handle multiple collaborating sources at varying granularities, but also to serve as a database of information with which further inferencing can take place. Given that a user is in "Intelligent Room," we can use the relation-map shown in Figure 4-2 to make the following inferences:

- The user is at Bldg. NE43.
- The user is at MIT.
- The user is at Tech Square.
- The user is at Cambridge.
- The user is adjacent to Boston.
- The user is near Kendall T-Stop.
- The user is near a Subway Station.

In this way, we combine sensor information with general knowledge about the world to build rich location context. At first glance, some may view this context as common sense and therefore not useful because users would not learn anything new; however, computers that acquire this small piece of common sense (one of the classic problems in artificial intelligence) can provide a wealth of additional services. For example, when a user is near a subway stop, he/she would most likely want to go to a restaurant either in the vicinity or close to other subway stops. Also, when in the vicinity of Boston, one might want to notify friends in Boston that he/she is in the area.

4.3.3 A Note on Context in a Relation-Map

In Figure 4-2, the dominant relation verb is "contains." Containment is the most common relation used among the various location representations that exist today. The primary advantage of using a containment hierarchy is that it provides the most

desired organization for a world of places. However, many location representations either require a strict containment hierarchy of places or make it very difficult to represent multiple parent places. Brumitt and Shafer [4] recognize that the real world cannot be neatly "partitioned into unambiguous, non-overlapping pieces that exhaustively cover the area in question." The structure of a relation-map allows for easy representation of these multiple hierarchies.

While most location representations tend to focus on containment alone, a relation-map supports other location verbs such as "near" and "adjacent to." As shown above, these verbs are very useful in establishing further context by allowing for proximity and adjacency representations along with containment. Furthermore, a relation-map admits the possibility of annotation with additional entities and relations (such as "is a"). Such annotation would typically be done by and for applications that need the additional relational information. This could very easily extend to other context analysis such as behavior or activity deduction (i.e. context building not relevant to location) by simply adding relational verbs. We are not suggesting that all context information should be stored and evaluated together, but we are saying that different forms of context information need to be related to each other in flexible ways.

We leave this as an open issue but offer a suggested solution. The purpose of the relation-map is to provide a location representation that is conducive to communication of location information, to collaboration between multiple location sources, to sensor fusion at varying granularities, and to access control. We propose that context be added to a relation-map on a need-basis. That is, if in order to provide one of these features, we need to add a particular verb, the appropriate application will add it.

4.3.4 Access Control

Within the Intelligent Room and E21 projects, access control is treated as a separate and separable service, orthogonal to services like location awareness. Consequently, our design does not solve any access control problems per se, but simply uses access control mechanisms supplied as a service by the intelligent environment. However,

there still remains significant design work on the location side, because we must specify the expressive power that we require of the access control system in order to allow the flexible access control that we need.

With our software platform having an agent society per user, we can allow users to constantly obtain location information but control distribution of this information to services and other users. More specifically, the Personal Location Agent (PLA) working in a user's society can keep track of the user's location at all times, but control distribution of this information to those outside of the user's society. The primary benefit to this design is that the user can allow services working on his/her behalf to access his/her location at all times, while restricting access to foreign services.

Semantic Expressions

The ability to establish location at varying granularities with a relation-map enables a user to perform such access control with greater ease. Using the convenient organization of places portrayed by a relation map, users can create more intuitive distribution rules. For example, instead of creating a rule stating,

"If a user is in room-1, room-2, ... , or room-n, then tell those in room-1, room-2, ... , or room-n that the user is at work."

we can instead create a rule stating,

"If a user is in his work-building, then tell those in his work-building that the user is at work."

Furthermore, by customizing a relation map and inserting one's own perception of places and relations between places, a user can create rules that more adequately portrays his/her personal life. For example, with the customized relation map shown in Figure 4-3, a user can create access control rules using semantic terms like "social place," "work place," "lunch place," etc.

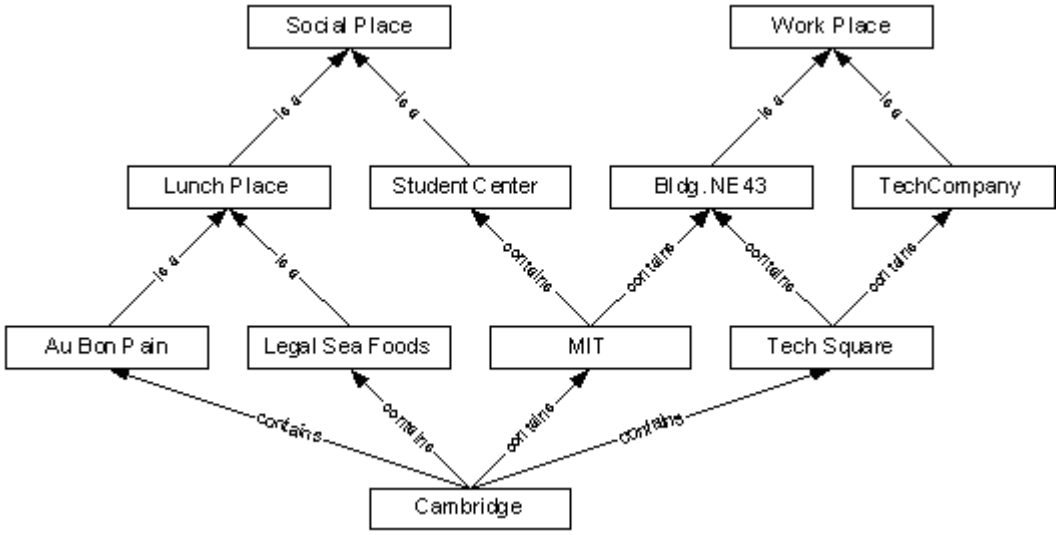

Figure 4-3: Example of a Customized Relation-Map

Distribution at Varying Granularities

Users may also use the additional context available in a relation map to distribute location information at varying granularities. For example, a user may wish to have the following distribution:

- Let family members know the user's exact location.
- Let friends know the user's location at building granularity.
- Let acquaintances know the user's location at city granularity.
- Let strangers know the user's location at planet granularity.

4.4 Uniquely Identifiable Places

PLACE's current solution for a common map of locations is to have all users share a common store holding the locations, and then to distribute this store across many databases. This allows for unique identification numbers to be attached to each location. All entities may communicate using these unique IDs and understand each

other because they all possess the identical mapping from the IDs to the symbolic locations. At the same time, this common map of locations also solves the problem of having communicating entities possess the same vocabulary of locations.

We want to be clear that this solution is only a stopgap. It won't scale well, and carries with it enormous problems about generating, distributing, and coordinating information. We are only interested in solving those problems as much as is required to support our stopgap. Furthermore, it simply sweeps under the rug the fact that existing and new applications that depend on other representations for object and locations will still need to be semantically coordinated with the unique identifiers.

4.5 Subscriptions

At times, it is useful to know a user's location on demand; other times, it is useful to know when a user arrives at a location. PLACE supports *subscriptions*, characterized by a user, a place, and a time range of interest. When the user arrives at the place within the time range, the subscriber is notified.

Chapter 5

Implementation of PLACE

PLACE is essentially a collection of agents working together on behalf of the user. The following is a description of these agents and the language used to convey location.

5.1 Relation-Map

The relation-map is built on top of the semantic network developed in the Intelligent Room. The semantic network consists of arbitrary nodes and links developed in Java that interfaces with an underlying database store. The fundamental units of the semantic network are the `Node` class and the `Link` class. Each of these extends `UniqueID`, which contains one property of type long called `uniqueID` and a set of methods for executing basic queries to the database. The semantic network maintains the invariant that no two objects stored in the database have the same `uniqueID`. Thus, all nodes and links are uniquely identified by these `uniqueID`s.

5.1.1 `LocationNode` and `LocationLink`

The `LocationNode` class is an extension of the generic `Node` class. It contains the following properties:

- `globalID` - (`GlobalID`) the unique identifier consisting of its `storeID` and `uniqueID`

  - `storeID` - (`long`) the ID of the store in which the `LocationNode` resides
  - `uniqueID` - (`long`) the ID of the `LocationNode` within the store

- `owner` - (`AgentID`) null if the `LocationNode` is public; the user's LIA ID if the `LocationNode` is private or specific to the user

The `LocationLink` class is an extension of the generic `Link` class. It contains the following properties:

- `globalID` - (`GlobalID`) the unique identifier consisting of its `storeID` and `uniqueID`
  - `storeID` - (`long`) the ID of the store in which the `LocationLink` resides
  - `uniqueID` - (`long`) the ID of the `LocationLink` within the store
- `owner` - (`AgentID`) null if the `LocationLink` is public; the user's LIA ID if the `LocationLink` is private or specific to the user
- `fromNode` - (`Place`) the node from which the link points
- `toNode` - (`Place`) the place to which the link points

`GlobalID` as the Unique Identifier

Note that each `LocationNode` and `LocationLink` has a `globalID` that couples the `storeID` and the `uniqueID`. Because each store's invariant only maintains unique identifiers locally, the unique identifier for the common map of locations shared by all users spanning across many stores is the `globalID`.

Customized Relation-Maps and Accessibility

In order to allow for customization of relation-maps, each `LocationNode` and `LocationLink` has an `owner`. A `LocationNode` or `LocationLink` is considered be *accessible* to a user if and only if `owner` is set to the user or to `null` implying public accessibility.

Notice that up to this point, there is no commitment to a semantic representation of locations. The abstraction layer consisting of `LocationNode` and `LocationLink` merely provides for unique identifiers and ownership.

5.1.2 `Place`

The `Place` class, an extension of the `LocationNode` class, holds a semantic representation of a location. It has the following additional properties:

- `equivalents` - (`HashSet` of `GlobalID`) the `GlobalID`s of places declared to be equivalents of this place
- `name` - (`String`) the descriptive name by which people identify the place
- `granularity` - (`int`) the granularity of the place

Equivalence

As mentioned in Section 3.1.4, each location has many descriptive names by which people refer to them. Each descriptive name is represented by its own instantiation of the `Place` class. Thus, in order to insert the reasoning that two names can describe the same place, there is the concept of `equivalence`. Equivalence is manually inserted into the relation-map as a property of a `Place`. The `equals` method is modified to return true if and only if two places are in the same equivalence group. An invariant is maintained to require each member of an equivalence group to include all other members in its `equivalents` set.

5.1.3 `ContainsLink`

The `ContainsLink` class, an extension of the `LocationLink` class, is the relation portraying containment. `ContainsLink` has the following static methods that execute queries specific to containment:

- `getParents(AgentID aid, Place child)` - retrieves the `fromNode` of each `ContainsLink` with *child* as the `toNode` and with either `aid` or `null` as the `owner`

- `getChildren(AgentID aid, Place parent)` - retreives the `toNode` of each `ContainsLink` with `parent` as the `fromNode` and with either `aid` or `null` as the `owner`

- `getAncestors(AgentID aid , Place child)` - recursively calls `getParents` to retrieve all places at all granularities that contain `child` and that are owned by `aid` or by no one

- `getDescendants(AgentID aid , Place parent)` - recursively calls `getChildren` to retrieve all places at all granularities that are contained by `parent` and that are owned by `owner` or by no one

5.1.4 `PedestrianLink`

The `PedestrianLink` class, also an extension of the `LocationLink` class, is the relation portraying pedestrian paths between two places. `PedestrianLink` has the following static methods that execute queries specific to pedestrian transit:

- `getShortestDistance(AgentID aid, Place from, Place to)` - gets the distance of the shortest path from `from` to `to` using `PedestrianLink` accessible by `aid`

- `isWithin(AgentID aid, Place from, Place to, long distance)` - returns true if

5.2 Location Clues

All location clues extend an abstract class called `LocationClue`, which consists of the following elements:

- `source` - (`String`) the identifier of the source
- `name` - (`String`) the name of the user that has been detected
- `time` - (`long`) the time at which the user has been detected
- `certainty` - (`long`) the certainty at which the souce has made the detection

5.2.1 `PlaceLocationClue`

The `PlaceLocationClue` class, an extension of a `LocationClue`, is a location clue specific to PLACE's semantic representation for locations and thus, contains two additional properties:

- `place` - (`Place`) a semantic name for the location
- `granularity` - (`int`) the granularity of the location

The clue's granularity is set to the granularity of the `place`.

5.2.2 `CricketClue`

The `CricketClue` class, also an extension of a `LocationClue`, is a location clue specific to Cricket's representation for locations and thus, contains a CricketPlace, as opposed to a `Place` in `PlaceLocationClue`:

- `space` - (`CricketSpace`) the ID for the Cricket space

5.3 Location Information Agent

The Location Information Agent (LIA) is responsible for the following tasks:

- Retrieve any location information requested by another agent.
- Perform translation between each supported location representation and the user's relation-map.

5.3.1 Single Store

Although the infrastructure allows for easy extensibility to multiple database stores, PLACE currently supports a single store. The underlying semantic network is still under development and has yet to support multiple stores.

5.3.2 Information Retrieval

Each element of the relation-map contains retrieval methods specific to the element. However, these methods are only for retrieval from the local store. The LIA is responsible for finding information outside of the local store and for essentially providing for a distributed map of locations (not currently implemented). Consequently, there is a necessary layer of abstraction for information retrieval between the low-level methods defined by each location element and the agents who want to know about all locations in the universe. Thus, the LIA is the sole source of location information for all other agents working in a user's society. The following query methods have been implemented for the currently implemented elements in the system:

- `getPlaces(ConditionCollection cc)` - retrieves all places owned by the user or public to all users, that match the specified conditions
- `getMyPlaces(ConditionCollection cc)` - retrieves all places owned by the user that match the specified conditions
- `getPublicPlaces(ConditionCollection cc)` - retrieves all places public to all users that match the specified conditions
- `getContainsAncestors(Place p)` - retrieves all places containing `p`
- `getContainsDescendants(Place p)` - retrieves all places contained by `p`

Note that a `ConditionCollection` is simply the semantic network's means of specifying conditions for data retrieval.

In addition to providing for a multiple-store universe of locations, the LIA serves to choose only the places and relations relevant to the user. For all given queries, by

default, the agent returns all relevant objects that are either owned by the user or public to all users. Otherwise, methods such as `getMyPlaces` and `getPublicPlaces` allow for agents to either retrieve only private places or only public places.

5.3.3 Translation

The current implementation of PLACE only supports Cricket locations in addition to its own semantic representation. Generally, translation will be left to the developers integrating particular location systems into Metaglue. However, the LIA offers the following methods for translation to and from Cricket spaces:

- `convertToPlace(CricketSpace cs)` - converts a `CricketSpace` to a `Place`
- `convertToCricket(Place p)` - converts a `Place` to a `CricketSpace`

These methods simply perform a hash table lookup on the `CricketSpaces` that are registered with the LIA.

5.4 Personal Location Agent

The Personal Location Agent (PLA) is responsible for the following tasks:

- Receive and store locations of all representations.
- Given a set of location clues, perform sensor fusion and create a resulting location clue with determined location and computed certainty at each granularity.
- Notify Subscription Manager Agent of newly computed location clues.

5.4.1 Receiving Location Clues

The PLA has the following methods to receive the clues of each supported representation, which currently consists of `Places` and `CricketSpaces`:

- `receiveClue(PlaceLocationClue plc)` - stores `plc` and recomputes user's location

- `receiveCricketClue(CricketClue cc)` - stores `cc` and recomputes user's location

5.4.2 Sensor Fusion

Clues with Varying Granularities

Given a location clue, the PLA duplicates the clues for each equivalent and ancestor for the given place. The PLA assumes that if a clue suggests that a user is in a particular place X, then it also suggests that the user is in all equivalent places and all places containing X. Thus, for each of these places, the PLA creates a new location clue containing the same source, name, time, and certainty. Although certainty could arguably increase for inferring at higher granularities, the current implementation does not support this reasoning.

Clue Decay over Time

The certainty of each location clue decays linearly over sixty seconds. More explicitly, for a given clue C:

$C.certainty = C.certainty * ((1 - (currenttime - C.time))/60seconds)$;

Reinforcement

During sensor fusion, a clues that suggests a place that has already been suggested is considered to be reinforcement (or support). The PLA uses the same approach as that used in rule-based systems. For a clue D reinforcing a clue C:

$C.certainty = C.certainty + (D.certainty * (1 - C.certainty))$

Resulting Location Clues

Given that the given location clues contain a variable degree of uncertainties, and that the reasoning behind its sensor fusion also has a variable degree of uncertainty, the PLA produces results with an associated uncertainty. Using the mechanisms above, the PLA merges the given locations clues into a set of location clues, one per

granularity. Conflicts at each granularity are resolved by simply taking the location clue with the higher certainty. If two clues have equal certainty, then both are taken.

5.5 Subscription Manager Agent

The Subscription Manager Agent (SMA) is responsible for the following tasks:

- Receive and maintain active subscriptions.
- When a subscription has been realized, notify the subscriber.

5.5.1 Subscription

A Subscription is described by the following properties:

- `aid` - (`AgentID`) the agent interested in the user's location
- `from` - (`long`) the time from which the subscription is active
- `to` - (`long`) the time to which the subscription is active
- `place` - (`Place`) the `aid` desires to know when the user reaches this place

Realized Subscriptions

When the user arrives at `place` after `from` and before `to`, the subscription is *realized*. At this point, the SMA sends a notification to `aid`. A notification is the standard messaging protocol between agents in Metaglue.

Active Subscriptions

A subscription is *active* when the current time is within the time range specified by `from` and `to`. Otherwise, the subscription is *inactive*.

5.5.2 Managing Subscriptions

In order to manage subscriptions, the SMA must collaborate with the PLA to keep up-to-date on the user's location so that it knows immediately when a subscription is realized. There are a few scenarios that must be accounted for in subscription management.

The intuitive case is when a user moves into the location of interest and realizes an existing subscription. The PLA informs the SMA when there is newly updated location information. At this time, the SMA checks its subscriptions to see if any of them are met. When a subscription is realized, the SMA informs the subscriber.

The alternative case is when a subscription is realized not because the user has newly updated location information, but rather, because the current time has entered the specified time range of the subscription. Thus, the SMA must check its subscriptions not only in response to updated information, but also in response to the activation of a subscription whose place of interest is already the current location of the user. Upon checking subscriptions, the SMA finds all inactive subscriptions that match the user's current location. The agent computes the time until activation for each of these inactive subscriptions. Finally, the agent sets a timer for the shortest of these times, after which the SMA rechecks the subscriptions.

Chapter 6

Future Work

PLACE consists of many components, many of which deserve individual research attention. Building a system that is extensible has been a primary design goal during development. The following is a short list of features in PLACE that are strong candidates for future work on the system.

6.1 Support More Relations

The relation-map can be easily extended to support more types of relations. For each additional relation added, the following tasks must be completed:

1. Determine and develop useful retrieval methods for a local store. These methods should embody the rules for traversing the data, as discussed in Section 3.1.4.

2. Teach the LIA about these methods so that agents may use the new relation. Whether the LIA returns private and/or public places is dependent on the relation being implemented.

6.2 Support More Representations

Another possible extension in the language portion of PLACE would be to support more location representations, thereby increasing the potential benefits of sensor fu-

sion. In order for a new representation to be supported, the following tasks must be completed:

1. Design a translation scheme by which the new representation can be converted to and from places in a relation-map.

2. Teach this scheme to the LIA so that it may perform translation service for other agents.

3. Create an extension class of LocationClue to support the new representation.

4. Modify the PLA to accept and store these clues supporting the new representation.

For example, a likely candidate would be a coordinate-based location representation such as the longitude-latitude coordinates used in GPS. The following classes and methods must be developed:

- Classes
  - `GPSCoordinate`
  - `GPSClue`
- Methods
  - `convertToPlace(GPSCoordinate gps)` in Location Information Agent
  - `convertToGPS(Place p)` in Location Information Agent
  - `receiveGPSClue(GPSClue gpsc)` in Personal Location Agent

6.3 Clue Decay Algorithm

All clues currently decay linearly over a fixed amount of time. However, as mentioned in Section 3.3.2, the decay function of the clues should be dependent on many factors, including the granularity, the sensor type, and even the rate at which the user is moving. Determining a method of creating decay functions for each clue based on

such factors and modifying the infrastructure to support this feature would provide for more accurate sensor fusion.

6.4 Granularity Convention

Although PLACE supports the notion of varying granularities throughout the system, the actual granularities are not defined. One could imagine creating a convention in determining the granularities, currently implemented as integers, of all places. As a simple example, all points could have a granularity of 1, all small (defined by a range of square footage, say between 0 and 100 square feet) rooms and spaces could have a granularity of 2, all medium (between 100-200 square feet) rooms and spaces could have a granularity of 3, etc. Conventions can be established to as large of a granularity as needed. If a large number of granularites are desired, the PLA can be modified to determine the best location for each range of granularities, as opposed to each granularity.

6.5 Multiple Stores

As mentioned in Section 5.3.1, PLACE currently supports a single store. However, GlobalIDs were created precisely for handling multiple stores and a distributed universe of locations. As Metaglue's presence becomes more widespread, multiple stores will be a requirement for scalability reasons. Thus, such an expansion is a worthwhile investment.

6.6 Applications

The general design of PLACE was inspired by the issues that are inherent in the problem of sharing user locations. The following describes a series of applications that demonstrate the motivations behind the features of the design.

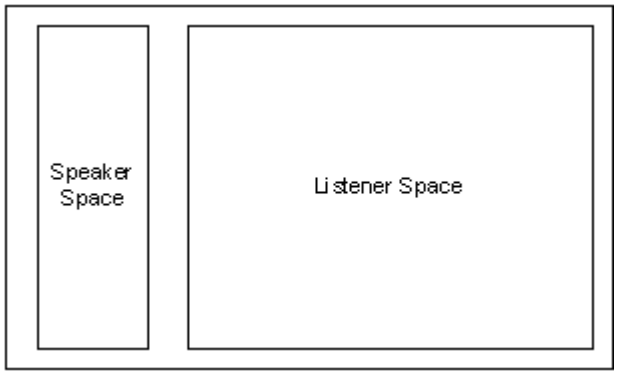

Figure 6-1: Example of a Relation-Map for Symposium Progress Tracker

6.6.1 Symposium Progress Tracker

The Symposium Progress Tracker (SPT) is a simple application that demonstrates the convenience of a semantic location representation. The uses location to determine how far a symposium has progressed with respect to its agenda. For example, consider a symposium that consists of three speakers. Once the first speaker reaches the speaker space (see Figure 6-1), the SPT assumes that the agenda has proceeded to the first speech. When the first speaker leaves the speaker-space and the second speaker enters the speaker-space, the SPT assumes that the agenda has proceeded to the second speech. In this way, the SPT tracks the progress of the symposium. This information can be used to automate audio/video programming (that is, display the portion of the symposium that the viewer cares about, using cameras, and microphones local to the symposium, and available to the viewer by network) or to allow users to attend only specific portions of the symposium.

6.6.2 Patient Tracker

Patient Tracker uses location information of patients to track a patient in a medical facility and to provide doctors with appropriate and relevant information when approaching a patient. Whereas doctors in the ER might want to immediately know the patient's vital signs (including temperature, blood pressure, heart rate, respiratory

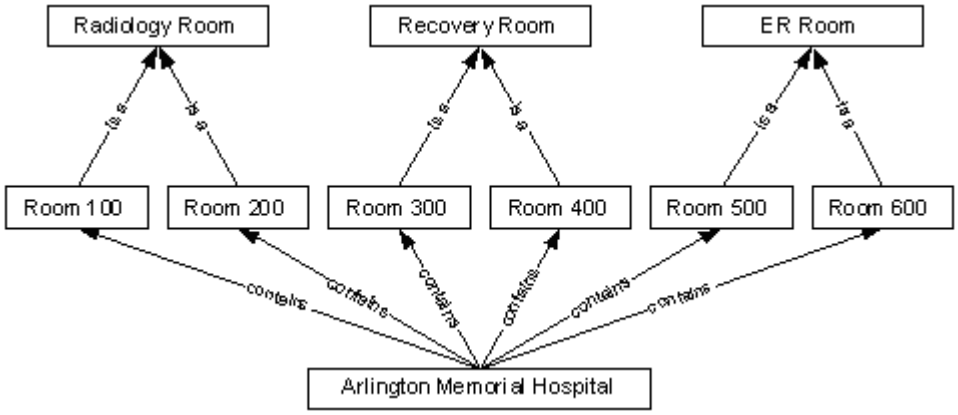

Figure 6-2: Example of a Relation-Map for Patient Tracker

rate, etc.), those in the recovery department might want to immediately know the patient's laboratory values (cell blood count, electrolytes, etc.) or study results (x-rays, CAT scans, etc.). In this way, Patient Tracker not only utilizes symbolic representation of physical location but also a relation-map semantic relation that categorizes the rooms in the hospital (see Figure 6-2). In any case, whether given in certain departments or all departments, patient information given to doctors before or upon arrival in preparation for treatment can save doctors time and energy. Furthermore, in terms of privacy issues, patients do not want to show their location to persons other than doctors and nurses responsible for their care, and appropriate visitors.

6.6.3 Expert Finder

Expert Finder uses location information to find people declared as experts of a specified topic. In some cases, it is much easier to ask someone for help rather than search through manuals and documents. Unlike the Patient Tracker, Expert Finder potentially deals with people outside the scope of a building. Upon medical emergency, one might want to find the nearest doctor in the area. Thus, in addition to semantic location representation and relation maps utilized by the previous applications, the ability to interpret multiple location technologies is necessary to ensure that one is able to understand the doctor's location. On a smaller scale, one might want to find

the nearest IT employee for immediate on-site technical support.

Even experts do not want to show their location while they are in a private location. However, in a significant enough emergency, that privacy concern may need to be over-ridden. Flexible, Role Based Access Control can solve this type of problem. In this case, when an expert comes close to a user in public space, the system will tell that to the user.

Furthermore, we may need to cast a very wide net for an expert (consider for example obtaining experts to deal with an impending terrorist attack). If an expert is outside, GPS (or cell phone) may detect their position, while if an expert is inside building, certain indoor location systems detect that. To treat multiple sensor inputs and information sources, we need the sensor fusion function.

Chapter 7

Contributions

My research essentially exposes the Intelligent Room to the problem of sharing locations between users. It examines the issues behind the problem, offers a general infrastructure to support the sharing of user-locations, and makes suggestions about how to make further improvements to the system.

7.1 Survey of Issues Relevant to Sharing Locations

The research community studying location has produced much literature describing the inherent issues to varying aspects of location. My research focuses on the problem of location sharing and attacks the general issues behind location in top-down fashion. That is, the issues were always approached with location sharing in mind. This study has resulted in a list of considerations one must take in building a location sharing infrastructure.

7.2 Location Infrastructure

Given these considerations, I designed and developed a prototype of a location sharing infrastructure, integrated with the Metaglue architecture. PLACE offers a set of solutions to several location issues, basic functionality, and a modular design conducive to future extensions or improvements. The design implements sensor fusion,

supports access control, and uses a semantic representation of locations in the attempt to mimic how humans reason about locations.

7.3 Extensibility

Although I offer an implementation of PLACE, there is much room for extensions and improvements to the system. Chapter 6 describes several candidates for future work on the system. The survey of issues, many of which are very difficult problems unto themselves, in combination with these suggested improvements aim to provide direction for implementation extensions and improvements to PLACE and for future research on the problem of sharing locations.

Bibliography

[1] E. Amoroso. *Fundamentals of Computer Security Technology.* Prentice Hall, 1994.

[2] S. Ark and T. Selker. A look at human interaction with pervasive computers. *IBM Systems Journal*, 38(4), 1999.

[3] R. W. Baldwin. Naming and grouping privileges to simplify security management in large databases. In *In IEEE Symposium on Computer Security and Privacy*, 1990.

[4] B. Brumitt and S. Shafer. Topological world modeling using semantic spaces. In *Workshop Proceedings of Ubicomp 2001: Location Modeling for Ubiquitous Computing*, September 2001.

[5] Database Promotion Center. G-xml (geospatial-extensible markup language) protocol 2.0. http://gisclh.dpc.or.jp/gxml/contents/index.htm, March 2001.

[6] D. Clark and D. Wilson. A comparison of commercial and military computer security policies. In *IEEE Symposium of Security and Privacy*, pages 184–194, 1987.

[7] M. Coen, B. Phillips, N. Warshawsky, L. Weisman, S. Peters, and P. Finin. Meeting the computational needs of intelligent environments: The metaglue system. In *Proceedings of MANSE'99*, Dublin, Ireland, 1999.

[8] MOSTEC: MObile Information Standard TEchnical Committee. Poix: Point of interest exchange language, version 2.0 document revision 1. http://www.w3.org/TR/poix/, July 1999.

[9] Open GIS Consortium. Geography markup language (gml) 1.0. http://www.opengis.org/techno/rfc11info.htm, December 1999.

[10] World Wide Web Consortium. Platform for privacy preferences project. http://www.w3.org/P3P/.

[11] SignalSoft Corporation. Wireless location services. http://www.signalsoftcorp.com.

[12] M. Dertouzos. The future of computing. *Scientific American*, July 1999.

[13] A. Dey and G. Abowd. Cybreminder: A context-aware system for supporting reminders. In *Proceedings of International Symposium on Handheld and Ubiquitous Computing*, 1999.

[14] A. Schmidt et al. Advanced interaction in context. In *In Proceedings of International Symposium on Handheld and Ubiquitous Computing (HUC 99)*, 1999.

[15] N. Davies et al. Using and determining location in a context-sensitive tour guide. *Computer*, August 2001.

[16] R. Want et al. The active badge location system. *ACM Trans. Information Systems*, pages 91–102, January 1992.

[17] K. Gajos and A. Kulkrani. Fire: An information retrieval interface for intelligent environments. In *Proceedings of the International Workshop on Information Presnetations and Natural Multimodal Dialogue (IPNMD 2001)*, Verona, Italy, 2001.

[18] J. Goguen and J. Meseguer. Security policies and security models. In *Proceedings of the 1982 Berkeley Conference on Computer Security*, pages 11–20, Berkeley, CA, 1982.

[19] D. Hall and J. Llinas. *Handbook of Multisensor Data Fusion.* CRC Press, Boca Raton, June 2001.

[20] N. Hanssens, A. Kulkarni, R. Tuchinda, and T. Horton. Building agent-based intelligent workspaces. In submission.

[21] J. Hightower and G. Borriello. Location systems for ubiquitous computing. *Computer*, August 2001.

[22] R. Jose and R. Davies. Scalable and flexible location-based services for ubiquitous information access. In *Proceedings of International Symposium on Handheld and Ubiquitous Computing (HUC 99)*, 1999.

[23] M. Korkea-aho and H. Tang. Experiences of expressing location information for applications in the internet. In *Workshop Proceedings of Ubicomp 2001: Location Modeling for Ubiquitous Computing*, September 2001.

[24] M. Krause and H. Tipton. *Handbook of Information Security Management.* CRC Press LLC, June 1998.

[25] Robert Laddaga. Creating robust software through self-adaptation. *IEEE Intelligent Systems*, 14, May/June 1999.

[26] S. Long, D. Aust, G. Abowd, and C. Atkeson. Cyberguide: Prototyping context-aware mobile applications. In *Proceedings of ACM CHI'96 Project Note*, 1996.

[27] N. Marmasse and C. Schmandt. Location-aware information delivery with commotion. In *Proceedings of International Symposium on Handheld and Ubiquitous Computing*, 1999.

[28] J. McCarthy and E. Meidel. Activemap: A visualization tool for location awareness to support informal interactions. In *Proceedings of International Symposium on Handheld and Ubiquitous Computing (HUC 99)*, 1999.

[29] Department of Defense. Trusted computer security evaluation criteria, December 1985. DOD 5200.28-STD.

[30] National Institute of Standards and Technology. Security requirements for cryptographic modules. *Federal Information Processing Standard 140-1*, 1992.

[31] R. Oppermann and M. Specht. A context-sensitive nomadic exhibition guide. In *Proceedings of International Symposium on Handheld and Ubiquitous Computing (HUC 99)*, 1999.

[32] MIT Project Oxygen. http://oxygen.lcs.mit.edu.

[33] K. Poland and M. Nash. Some conundrums concerning separation of duty. In *In IEEE Symposium on Computer Security and Privacy*, 1990.

[34] N. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Proceedings of the ACM MOBICOM Conference*, Boston, MA, August 2000.

[35] P. Robertson, R. Laddaga, and H. Shrobe. *Proceedings of the First International Workshop on Self-Adaptive Software; Lecture Notes in Computer Science 1936*. Springer-Verlag, 2001.

[36] M. Sekiguchi, K. Takayama, H. Naito, Y. Maeda, H. Horai, and M. Toriumi. Navigation markup language (nvml). *World Wide Web Consortium (W3C) Note*, August 1999.

[37] A. Snoeren and H. Balakrishnan. An end-to-end approach to host mobility. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MOBICOM 2000)*, Boston, MA, August 2000.

[38] S. Tzelepi, D. Koukopoulos, and G. Pangalos. A flexible content and context-based access control model for multimedia medical image database systems. In *Proceedings of ACM MMSig'01*, 2001.