A Privacy Service for Location-Based Collaboration among Mobile Users

Vagner Sacramento[1], Markus Endler[2] and Clarisse de Souza[2]

[1]Institute of Informatics, Federal University of Goiás – UFG,
Caixa Postal 131, CEP 74690-815, Goiânia, GO, Brazil
vagner@inf.ufg.br

[2]Department of Informatics, Pontifical Catholic University of Rio de Janeiro – PUC-Rio,
Rua Marquês de São Vicente, 225, RDC,
CEP 22453-900, Gávea, Rio de Janeiro, RJ, Brazil
{endler | clarisse}@inf.puc-rio.br

Received 9 September 2008; accepted 29 December 2008

Abstract

The potential loss of privacy due to the use of location-based applications may be one of the greatest obstacles to their wider acceptance. Nevertheless, most research about privacy management to-date has not taken into consideration the complexity in the utilization of collaborative applications that may require users' location sharing among participants. In this article, we propose a privacy service that helps users to maintain their privacy policy in a flexible and incremental way. We also carried out a qualitative evaluation study whose results illustrate several challenges that should be handled in the design of such a service.

Keywords: privacy, design of a privacy service, LBS applications, context-aware application, context-aware middleware.

1. Introduction

The dissemination of portable computing devices equipped with GPS and the existence of several middleware services that infer user location [31, 10, 16] have motivated the development of location-based applications (a.k.a. LBS – *Location Based Services*). These applications use the location information to offer customized services to the end users. For example, some of them allow users to share their location information with peers [38], make a search or send messages based on their location or proximity, which facilitates coordination and orientation in groups [30, 13], such as in search-and-rescue scenarios.

In spite of offering some benefits, such LBS applications introduce also new risks and threats to the user's privacy. These concerns call for location-based services and applications that consider privacy issues related to their usage from the beginning [36]. Nevertheless, the majority of related work [3, 5, 8, 9, 20, 33] does not take into account the high complexity of utilization of a LBS enabling privacy control. They do not take into account that the privacy level needs to be adjusted dynamically and that this control is highly dependent on both the situation, environment or task in which the user is currently engaged, as well as the levels of trust and the roles of the interacting users.

With the purpose of offering a privacy service with such a flexibility, we propose a service, called CoPS – *Context Privacy Service*, by which users can define and manage their privacy policy related to context information[1] in a gradual and interactive way. This service has been integrated into the MoCA architecture – *Mobile Collaboration Architecture* [42, 31], which allowed us to develop and experiment with some privacy-sensitive and location-aware application prototypes. The main contribution of our work is that we propose a larger, more comprehensible and effective set of privacy control mechanisms that assist the user in building and maintaining her privacy policy.

From the discussions in Westin [46] and Altman [1] we learn that the concept of privacy, and the way of handling and controlling it vary substantially from indi-

1 We consider location information a specific instance of context information.

vidual to individual. Therefore, we believe that only the affected user is able to properly decide which of his context information can, or cannot, be disclosed, because only himself is able to analyze, according to the situation, the risks vs. the benefits of this disclosure. Hence, the main challenge of the design of a privacy service is not simply to implement the access control to the user's information, but foremost to offer mechanisms that help the user to define and maintain his privacy policy in a gradual and flexible way. In this sense, the main question that guided our research was: How to provide a suitable set of privacy controls that offer flexibility and alleviate the complexity of the configuration and maintenance of the user's privacy policy? In order to deal with this question, we have carried out in-depth research about the challenges and approaches to design a flexible privacy service.

In this paper, we make a general discussion about privacy (Section 2) and derive some basic requirements about privacy management related to location-based applications. We also present our systematic study of related work concerning the design and utilization of technologies used to mediate user interactions, with the purpose of defining a conceptual model and identifying some privacy requirements that may be used as a basis of the development of a privacy service (Section 3). Based on this model and privacy requirements, we have designed and implemented the proposed privacy service (Section 4 and 5). Furthermore, we have carried out a performance evaluation to analyze the robustness of the proposed service, as well as a qualitative study with users to identify how they would utilize some of the privacy controls provided by CoPS (Section 6). The obtained results allowed us to compare our approach with other related work (Section 7) and derive some conclusions (Section 8) about the benefits and risks related to the utilization of location-based application for collaboration and communication.

2. Concepts and Discussion About Privacy

The notion of privacy is intrinsically associated with each person's individual perception of possible threats to her property, or her physical or moral integrity. This implies that privacy is an abstract and subjective notion, which materializes into a wide range of different, concrete concerns, according to the individual's specific demands in specific situations. Moreover, these demands are usually also related to cultural and social aspects of the user's community.

According to [1, 2], privacy control should neither be static, nor strictly based on rules. He also points out that the current technology's methodology for defining privacy preferences, e.g., users setting once some specific configuration parameters and then having to live conditioned on them, simply does not work. Also, in his theory of privacy, Westin [46] states that "individuals, groups and institutions claim the need of determining when, how and for which purpose information related to them are to be made available for other users".

These concerns raise some questions about how privacy control is being handled nowadays. On one hand, do users in fact have the means of precisely controlling access to their information? On the other hand, how much users really care about their privacy? In the meantime, many companies put a lot of effort into collecting personal information from their client base. They use their consumer profiles, the history of their credit card usage, the web access pattern, etc. in order to recommend new products, create focused marketing strategies, etc. which is a threat to the user's privacy. This is further amplified if the disclosed information is the user's location.

Some researchers [17, 4] have reported some results of polls where users said that they would not be concerned with disclosing their location information if they knew beforehand that the perceived advantages (of this disclosure) are greater than the risks. However, we believe that these users would have a different opinion about privacy if they were exposed to a scenario where they could suffer some form of aggression, such as an hold-up or kidnapping, because their location were disclosed. Therefore, user opinions are usually very dependent on the current perception of concrete risks and benefits which, of course, may change under different circumstances.

3. Conceptual Model and Privacy Requirements

In this section, we present the conceptual model underlying our privacy service, which primarily handles privacy issues related to access of user's location information. The model applies also to other sorts of context information, such as computational context, i.e., the state of the device's resources, or quality of the wireless connectivity. We focused our attention on privacy issues related to location information since this kind of information is the one that most stresses the privacy control issues. Therefore, in the remainder of the paper, we will use the terms context and location interchangeably.

In our model, each user is located at a well-defined symbolic place, which is represented by a name, and can change over time. The location information is inferred by a context service, i.e., a location/positioning service[2], that makes it available to the components of LBS applications. These applications, in turn, use this information to

2 Recall that most LBS require some symbolic or semantic information associated with a physical position, rather than just its geographic coordinates.

provide specific, collaboration-enhancing, services to the 'located' user or her peers, such as detection of user proximity, location-specific media/data access, etc.

The proposed model comprises several entities whose roles are explained as follows:

- *Requester* is a properly authenticated user that requests access to the context (i.e., location) of a *Subject*, which is produced and made available by a context service (or location service);
- *Subject* is the user whose context/location information is been requested;
- *PolicyMaker* is the user that is responsible for creating or modifying the privacy policy. She may be the Subject or another user, i.e., the model allows policy management both by the owner of the information and by the systems administrator of the organization (e.g., a company, or an institution)
- *Context/Location Service* is a computational entity responsible for processing the context access requests by the Requesters, publicizing the context information of the Subject, as long as the Privacy Service issues the corresponding authorization for the request.
- *LBS Application* is a distributed software system for communication or collaboration through which a Requester requests access to the Subject's context information;
- *Privacy Service* is the computational entity through which the Subject controls the access and sharing of her context/location information, according to the rules of the privacy policy defined by the PolicyMaker.

Figure 1 depicts the typical pattern of interaction among the entities of the above described model.

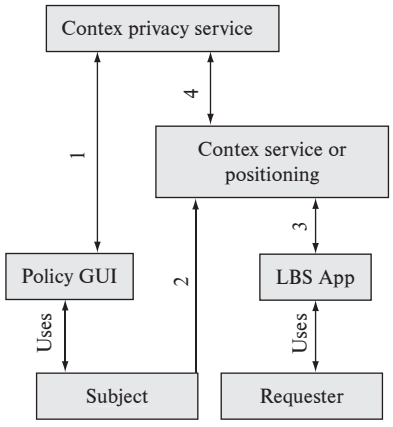

Figure 1. Interaction among the Entities.

Initially, the PolicyMaker (e.g., the Subject) defines the privacy policy by using the Policy Management Interface (1). In parallel, and contiguously, the Context/Location Service receives probed data from sensors and uses them to infer the Subject's location (2). This information, however, will be disclosed (to the Requester), depending on the restrictions specified by the privacy policy. The authenticity of the Requester will be guaranteed by an Authentication Service (not depicted). When the request is received by the Context Service (3), it sends an authorization request to the Privacy Service. According to the privacy policy, the Privacy Service then replies with a "Grant", a "Deny" or a "Not Available" message (4). Based on this reply, the Context Service then delivers - or not - the requested information.

In this model, the user does not need to define her privacy policy before she begins to utilize the LBS application, but can use some pre-defined privacy rules that are stored in the Context Privacy Service. Then, while she is using the LBS application, she can gradually define her own rules through a Privacy Policy Configuration Application and by such she will naturally create her own privacy policy, customized to her current need.

In the model, we assume that the Privacy Service cannot be made responsible for the possible consequences that may arise if the Requester forwards the Subject's context information to other users. As discussed by Grudin [14], the risks of a privacy invasion are not only determined by the means of its access, but are dependent also on the way that the information is stored (where, for how long, etc.) and distributed.

Moreover, we are making the following assumptions in the model: a) the Requester, the Subject and the PolicyMaker have each an unique identity which cannot be forged; b) the Privacy Service is used within a community where the users know each other and have a certain basic level of trust in each other, e.g., employees within a same organization, class-mates, friends, etc.

The use of the privacy service within such a community benefits from the implicit relation of trust and the individual's desire to maintain her reputation, which are essential elements of any social environment. It also helps to identify the possible types of requesters which need to be considered when devising the privacy policies [29].

It is worth mentioning, however, that the definition - and user's perception - of a community is an individual notion, and hence may vary from person to person. Therefore, even within a well-defined - or informally perceived - community, a user may feel the need to keep her privacy according to her personal preferences. For example, the Subject may wish to deny or grant access to her location information depending on the identity of the Requester (e.g., her boss, a colleague, etc.), the circum-

stance (e.g., at lunch time, after work, etc.), with different precisions of the location information, etc. Several such needs for fine-tuned privacy control are taken into account in our proposed conceptual model.

3.1. General Requirements of a Privacy Service

From the reults of a user survey [32] and results reported by other research groups - and discussed in [40] - we found that the main goal should be to support easy customization and adaptation of the user's privacy preferences, i.e., the users should be able to gradually refine their privacy policies while they use the service, according to their actual needs. In fact, this main requirement drove several design and implementation decisions of CoPS [40]. In the following we will discuss only the main requirements that we have identified. A more detailed description of the conceptual model and the whole set of requirements can be found in [40].

The main function of a privacy service is not just to implement basic access control for user's context information, but also to provide the user with appropriate means of managing her privacy policy gradually and interactively. In particular, it should help the user to easily define new rules, evaluate how they work in different situations, and allow her to modify and refine them, in order to properly maintain the desired level of privacy. This main requirement can be translated into the following properties of a Privacy Service, some of which have also been advocated by other researchers [22, 33]:

- Hierarchy of privacy policies;
- Different types of access control policies;
- Time-limited Rules;
- Ability to define groups of users;
- Ability to notify Subjects when their context information is queried;
- Detailed record of context queries;
- Temporal and spatial restrictions to context access;
- User-defined privacy profiles for common situations;
- Fine-grained access control.

In order to deal with the privacy needs of users within an organization and individually, the Privacy Service should provide means of organizing the privacy policies in a *hierarchical structure of several levels*. However, based on our observations, we found that the following three levels are sufficient for most LBS applications: a level of organizational policy, a level of user policy, and the default policy level. The organizational policy is defined by the system administrator, and should have precedence over the other levels. Policies at this level are used by the organization to enforce some general privacy control that is stronger than the specific user policies, for example, to ensure that each person's location is always available for applications required in emergency situations, such as guidance to exits during fire-alarms, etc. The user-level policy is defined by the Subject, and should have precedence over the default policy. The latter consists of a *template* of rules (also defined by the system administrator), which are applied to each new user until she creates her own policy. For some users, which do not care to define their own privacy policy, this default policy may be good enough for their daily tasks within the organization. Results presented by other authors [35] indicate that, in fact, most users do not care to modify the default system configurations of LBS applications. Therefore, the existence of a default policy is very important, specially in the early phase of a LBS deployment, since the users are not impelled to immediately set up their own policy, just to have minimal privacy control.

The privacy service should also offer different *types of access control policies*, since there are always users who are either very much or very little concerned with privacy issues, and who may want to adopt a more conservative, a more liberal or a moderate approach concerning the disclosure of their personal information. Our approach was to define three types of policies: Reserved (i.e., conservative), Liberal, and On-Demand. In *Reserved* type, by definition all the requests are denied, except the ones that match any rule which grants the access. In the *Liberal* policy type, all the requests are, by definition, granted except when the request matches a rule that explicitly denies the access. In type *On-Demand*, the final decision (Deny or Grant) for each request is determined interactively, by demanding an explicit input (e.g., Yes/No) from the Subject.

For each type of access control policy, the Policymaker can define a set of rules that determine under which circumstances the Subject's context information will be disclosed, and he can choose any of the following outcomes: "Grant" or "Deny" (but not both), "Not Available" or "Ask Me". In addition, the Policymaker (i.e., the Subject) may choose not to define any rule initially, and by such allow the Subject to use the On-Demand policy type, to gradually and interactively define her privacy control.

During the creation of a rule, the Policymaker should also be able to specify the *form of notification* (e.g., e-mail, SMS) that is to be sent to the Subject each time that the Privacy Service receives a request. This functionality also establishes an implicit social protocol among the users, which may help to prevent some malicious actions [23]. The simple fact that the Requesters are aware that the Subject *may* receive notifications of any access attempt can be a strong inhibitor of privacy violation actions.

As a means of supporting flexible access control, the Privacy Service should also allow the Policymaker to adjust the *temporal and spatial granularity*, and the *precision* of the disclosed information. For example, consider a scenario where user John would like to share his location information with his classmates, so that the group members can coordinate their activities using a LBS application such as *Friend Finder*. But maybe John does not feel very comfortable of sharing his exact location. In this case, he could adjust the spatial granularity of his location information: instead of disclosing the information at the granularity of rooms (e.g., inside room RDC512), it would be disclosed at the granularity of buildings (e.g. inside the RDC building). John could also set the temporal granularity (i.e., a temporal restriction), by specifying that his location information is to be disclosed only during a specific time period (e.g., Mondays to Fridays, nine-to-five), and only to a specific group of users. Moreover, he could specify also the precision (*freshness*) of the information to be disclosed, e.g., instead of his current position, the service should disclose the location where he was 30 minutes ago.

According to Goffman [12], users assume different roles in their social interactions. This makes us believe that some users may wish to define their privacy control policy according to their current role, task or activity. Some of these modes, e.g., "relaxing", "in a meeting", are possible values of an activity context, which only the user himself can precisely define. Moreover, it is likely that the user might want to define a specific privacy control policy for each such role/context. In order to fulfill this requirement, the Privacy service should allow the Policymaker to create *Privacy Profiles*, where each profile defines the access control rules for a specific role or activity context. For example, when the Subject would manually select the profile "in a meeting", this would enable only the rules of this profile, i.e., the privacy policy that the user wants for this specific situation or activity.

In order to increase flexibility, the Privacy Service should provide *fine-grained access control* where more specific rules have precedence over more generic rules. Hence, the service should implement a deterministic criteria for identifying the most specific rule to be applied for a given request. Through this fine-grained access control, the user could, in each rule, specify a restriction or an action to be executed when evaluating a request from a specific group or individual. For example, for each rule, the user can specify a different form of notification, adjust the granularity of the information to be disclosed, restrict the access for a specific group of users and for a specific pre-defined time interval.

4. CoPS Architecture

In this section, we briefly describe the architecture of the developed service for privacy control, (CoPS – Context Privacy Service), which addresses the privacy requirements presented in Section 3.1. The architecture of CoPS, illustrated in the Figure 2, offers a fine-grained access control, which enables the user to create and manage her privacy rules gradually, while using the LBS application.

In general lines, CoPS consists of a server and two client APIs. The server manages the access to the context information and the APIs implement the protocols for interaction with the CoPS server. The API *Context Access Authorization (CAA)* is used by the context (or location) service to send authorization requests (for access to the user's context/location) to CoPS . The API *User and Policy Management (UPM)* is used by the application clients of the Subject and the Requester to access and analyze logs, verify the consistency of the privacy rules, among others.

The interaction between these elements, illustrated in Figure 3, is similar to the one described in Section 3. The main differences are the use of the APIs *CAA* and *UPM* and the implementation of the user authentication as part of the privacy service. In the current implementation, the Requester authenticates himself at CoPS (3) to validate his/her identity. This authentication is performed through component *DUMAC* (Dynamic User Management and Access Control), shown in Figure 2, which generates a session token that is used to create a User Identification Token (UIT) for future requests. The UIT is a hash of the session token, and is carried on every request sent by the LBS application in order to certify that the user has already been authenticated. Both the request arguments and the token are relayed to CoPS by the context service in order to solicit access to a Subject's location (5). The implementation details of the generation and distribution of the user's session tokens are explained elsewhere [41].

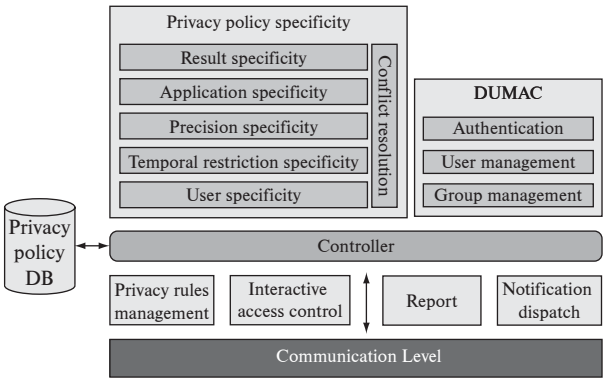

Figure 2. General CoPS Architecture.

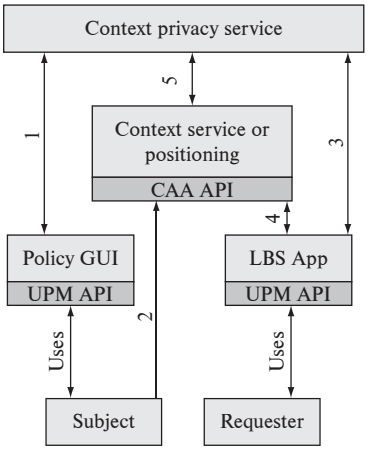

Figure 3. Interaction among client and server.

The component *Communication Level*, depicted in Figure 2, provides interfaces for synchronous and asynchronous communication of either plaintext or encrypted data using the SSL (*Secure Socket Layer*). All requests that reach the *Communication Level* are relayed to component *Controller* which identifies the type of request and interacts with the other components in order to execute the corresponding action.

The components *Notification Dispatch* and *Interactive Access Control* are used by component *Privacy Policy Specificity* to send to the Subject, respectively, access notifications and authorization inquiries, the latter asking for a Subject's final decision regarding a request. This inquiry happens if the user has adopted an On-Demand access control policy, or when the action associated with a selected rule is "Ask Me". In response to this inquiry, the Subject shall reply with a *Grant* or *Deny*, and may configure CoPS to store this decision as part of his/her privacy policy, so as to avoid repeated inquiries in future requests.

The central components of the architecture, however, are *Privacy Rules Management* and *Privacy Policy Specificity*. They are used for processing the requests regarding rule management and modifications in the configuration of the user's privacy policies. Some of their main responsibilities are: management of the hierarchical privacy policies, management of user groups, handling of temporary privacy rules, perform the basic operations of rule management (e.g., inclusion, removal, etc.), among others.

4.1 Structure of the Privacy Rules

The structure of a CoPS privacy rule is composed of several fields. Any privacy rule is associated with a default access policy type (Liberal, Reserved or On-Demand) chosen beforehand by the PolicyMaker. The proposed rule fields and their semantics are described as follows:

- *Policy Maker, Subject and Requester*: as described in Section 3.
- *Context Variable*: The specific type of context data being requested from the Subject or its device (e.g., symbolic location, IP Address, MAC-address, energy level, etc.).
- *Application*: List of application names that can be used by the Requester to access the context variable. The wildcard '*' represents any application.
- *Precision*: Specifies the value precision of the context variable (e.g., for the symbolic location information, this could be the spatial precision like state, city, ZIP code, building, room, etc.).
- *Temporal Restriction*: Date and time interval restrictions for disclosing the context information (e.g., weekdays, from 9 AM to 6 PM).
- *Freshness*: Specifies the freshness of the disclosed context information (e.g., location 15 minutes ago, or current location). The default value is 0 minute.
- *Timestamp*: Specifies the time in which the privacy rule has been created. This field is used to resolve possible conflicts among the rules.
- *AccessPolicy*: Represents the type of access policy (e.g., Reserved, Liberal or On-Demand) that this privacy rule is associated with.
- *Policy Level*: One of the three possible hierarchical levels of a rule: "organization", "individual" or "default".
- *Result*: Outcome of applying this rule to a request. Possible values are: "Not Available", "Ask Me", "Grant" and "Deny".
- *Notify Me*: The means by which the PolicyMaker wants to be notified when the rule is applied. The options available are "NoNotification", "E-Mail", "ICQ", "MSN" or "SMS".

4.2. Group Definitions

Group(s) definitions provide an additional facility for the management of privacy rules and also decrease the processing effort during evaluation of the requests. The *Subject* or *Requester* fields of a privacy rule can either hold a userID or a groupID.

There are two general categories of groups: *administrator* and *user-defined* groups. The first ones are structured groups, which reflect the hierarchical structure of the organization, and define the corresponding user roles, similar to RBAC [43]. Groups at a higher hierarchical level include all groups at a lower level, e.g., at University PUC, the group "puc.employee" comprises the groups "puc.employee.staff" and "puc.employee.prof", of which the latter in turn contains the group "puc.employee.prof.cs". The second category, *User-defined* groups, are arbitrary lists of userIDs without any structure, similar to *buddy lists*.

4.3. Privacy Policy Evaluation

During the evaluation process, more than one rule may match the request, for many reasons. For instance, when the requester belongs to several groups mentioned in field "Requester" of some rules (e.g., "Alice" belongs to groups "Coworker"and "MyFriend"), then all these rules match the request. CoPS' specificity algorithm aims to determine the *most specific* privacy rule that applies to a request and, if necessary, resolve possible conflicts among the rules.

The specificity algorithm works as follows: Given a set of rules previously selected (by the engine) to evaluate a request, the algorithm identifies the most specific rule of the set by comparing their structure fields in the following order of priority: *Subject*, *Requester*, *Context*, *Temporal Restriction*, *Precision*, *Application* and *Result*. When comparing rules with respect to a field, only the ones with the most specific value in this field are selected for the further specificity analysis, while all other rules are not considered for selection. This way, even if two or more rules have different relative specificity (i.e., they differ in two or more fields) the algorithm can identify the most specific rule analyzing these fields according to their priorities. For all fields, the wildcard symbol "*" may be used, with the meaning "least Specific".

For the specificity of the *Subject* and *Requester* fields, privacy rules mentioning an individual user (e.g., "Alice") are more specific than rules containing a user-defined group (e.g., "MyFriend"), which in turn is more specific than the ones mentioning an administrator-defined group. The administrator-defined group specificity follows the usual interpretation of a hierarchy: groups at a lower hierarchy level are more specific than groups at a higher level (e.g., "puc.employee.prof.cs" is more specific than group "puc.employee.prof").

The same hierarchy-induced specificity applied to the administrator-defined group is used also for the *Precision* field. For example, when comparing rules concerning location information, the most specific ones are those where field *Precision* mentions the lowest level in the location-hierarchy, e.g., "country.state.city.zip" (level 4) is more specific than "country.state.city" (level 3). Two or more privacy rules can be at the highest level of specificity with regard to their *Precision* field if they have the most specific value, and are at the same level in the hierarchy. When this happens, the next field (according to the priority) of these rules is compared to identify the most specific rule. In order to allow for such specificity analysis the developer of the Context Service has to define the syntax (e.g., campus. building.floor.room) of the name hierarchy for this specific field, which is a configuration parameter of CoPS .

The field *Temporal Restriction* represents the time interval and date at which the Requester is granted or denied access to the context information, depending on the access policy approach used (Liberal or Reserved). This field is very useful when the user wants to restrict the access in some special situations (e.g., at lunchtime or at working hours). The specificity for this field is evaluated in three phases: (1) select the rule(s) that match the time and date of the request; (2) identify the rule with the largest time interval and check whether the time interval of the other rules are its proper subsets (e.g. Temporal Restriction "Feb 5, 10:30 AM-2:00 PM" is a proper subset of restriction "Feb 5, 10:00 AM-6:00 PM"). Rules are considered to be at the same level of specificity either if they have identical time intervals, or if the time interval is not a proper subset of the largest time interval; (3) select the rule with the smallest time interval, when they are not at the same level of specificity.

With regard to the field *Application*, the specificity criterion uses only two possible levels: any application (represented by "*") and a list of applications. Finally, if all previously considered fields are at the same level of specificity, the *Result* field is the one used to select the most specific rule to evaluate the request. The possible values for this field are: "Not Available", "Ask Me" and "Grant" (or "Deny"). The "Not Available" result has precedence over "Ask Me", which in turn has precedence over the others (i.e., result "Not Available" is more specific than "Ask Me", which in turn is more specific than "Grant" and "Deny"). The reason is that "Not Available" implicitly means "Deny" and "don't let Requester know it", while "Ask Me" may be interpreted as "Deny" or "Grant", depending on the Subject's mood. A conflict is detected when there is more than one rule with a result "Not Available" or "Ask Me", or when all rules have either a "Grant" or "Deny" result. In this case, the last rule with highest specificity created by the PolicyMaker will be selected. It is necessary to define a deterministic choice for these situations because the conflicting rules may have different notification methods and only a single rule must be chosen to evaluate the request.

The privacy service was designed to be largely independent of the set of relevant context types. So, it would be possible include new rules which mention new/different parameters in all fields of a privacy rule, except for those that have explicit value restrictions such as "Result", "Privacy Level". The specificity algorithm implements the matching between the registered privacy rules and a request considering only the values of the Rule's parameters and Request structure, independent of their semantics.

4.4. Privacy Policy Evaluation Example

In this section, we show an example of possible privacy rules for user Bob, assuming that the Reserved policy has been chosen as default, i.e., whenever a request does not match any rule, it will be denied. These rules (shown

Table 1. Example rules.

Rules	Access policy	Subject	Requester	Context variable	Temporal restriction	Precision	Application	Result	Freshness	Policy Level	Notify Me
R1	Reserved	Puc. Student	Puc. Manager	Location	*	puc	Ap1	G	0	O	e-mail
R2	Reserved	Bob	Puc. Student	Energy	09:00 AM to 06:00 PM	*	*	G	5	U	ICQ
R3	Reserved	Bob	My Friend	Energy	09:30 AM to 12:30 AM	*	*	G	0	U	ICQ
R4	Reserved	Bob	Coworker	Energy	11:00 AM to 02:00 PM	*	*	NA	0	U	No Notify
R5	Reserved	Bob	Coworker	Location	09:00 AM to 12:00 AM	*	*	G	0	U	No Notify
R6	Reserved	Bob	Alice	Location	09:00 AM to 11:00 AM	campus. building	*	G	0	U	MSN
R7	Reserved	Bob	Alice	Location	10:00 AM to 04:00 PM	campus. building. floor. room.	*	G	15	U	e-mail

in Table 1) determine how and when Bob's location and energy context variables will be disclosed. In this example, we also assume the existence of some *user-* and *administrator-defined* groups (Bob's and PUC's groups are shown in Table 2), which are mentioned in some of the rules.

Table 2. Assumptions about User Groups

	Assumptions	
	Group	**Members**
user-defined	Bob.MyFriend	Bob, Alice, John
	Bob.Coworker	Alice, Jane, John
administrator-defined	Puc.Student	Bob, Alice, Jane, John
	Puc.Manager	Jane, Paul

Through some scenarios, we will now explain how the privacy rules are selected and used to evaluate a request, using the algorithm explained in Section 4.3.

As already mentioned, the rule to be applied to the request is always the most specific one, and comparison of the rule's specificity takes into account the fields *Subject*, *Requester*, *Context*, *Temporal Restriction*, *Precision*, *Application* and *Result*, in this order. Thus, the algorithm compares the values in the corresponding columns (from left to right), and as soon as one (or several) rules have a more specific value in one of the columns, they are candidates for further comparison.

Scenario1: If Jane makes a request for Bob's location, both R1 and R5 would apply. However, the request would be granted by R1, because this rule belongs to a higher level than rule R5 and, consequently, the first rule overrides the others.

Scenario2: Consider a request from John to get the energy level of Bob's device. In this case, R2, R3 and R4 are the related rules. But among those, rules R3 and R4 are selected because the user-defined groups mentioned in these rules are more specific than the administrator-defined group of R2. Finally, the request will be evaluated by R4 because, despite their fields *Requester*, *Temporal Restriction*, *Precision* and *Application* having the same level of specificity, their *Result* value differs, and "Not Available" has precedence over "Granted".

Scenario3: For Alice's request to get Bob's location rules R5, R6 and R7 should be examined. Among those, R6 and R7 take precedence over R5 because they apply to an individual user, "Alice", rather than to a group, as specified by R5. Although the R6 and R7 are at the same level of specificity in the *Temporal Restriction* field, R7 is more specific than R6 in the *Precision* field, and therefore will be applied to grant the request.

5. Access Authorizations Caching Implementation

CoPS has been implemented in Java and used MoCA's communication API and Event service for the interaction between the CoPS server and the client APIs. We carefully structured and coded the Privacy Policy Engine so as to maximize the efficiency of the privacy rules evaluation.

In its usual mode of operation, the Context Service forwards the request to CoPS whenever it receives an access request for the Subject's context information. If the Requester is successfully authenticated and the request is granted, CoPS replies with a "Grant", otherwise with a "Deny" or "Not Available" result. In order to reduce the response time of a context access request, we have implemented a cache holding CoPS' results of recent requests in

the CAA API. This way, once a request from a Requester *R*, related to a given Subject *S*, a specific context variable *C*, an application *A* and a precision *P* has been evaluated, the Context Service can evaluate subsequent queries concerning (*R,S,C,A,P*) from the local cache.

From the results of our tests we could perceive that caching significantly reduces the number of queries to CoPS and also reduces the response time of the context access authorization.

The local cache managed by the CAA is completely transparent to the Context Service, and the developer of this service can decide whether he wants to use it, or not. When the CAA processes a new access authorization request from a context service that uses cache, it forwards the request to CoPS and subscribes itself at CoPS' event server as interested in being notified whenever the result of the evaluated request changes. This way, whenever there is a change of a privacy rule, the event service will evaluate if this modification invalidates the result of any of the subscriber's request. The event server will only analyze the requests that match the Subject of the updated privacy rule. If the result value changed, the server will immediately notify the subscriber(s), such that the corresponding CAA's can update the cache with the new result of a given request.

After the Subject has defined his privacy policy the privacy rules will not be updated quite frequently, and consequently, the number of notifications of cache updates will be low.

The main problem of using a cache for the access authorization results is that there may be a short time interval between the update of a specific privacy rule at CoPS and the corresponding delivery of a cache update notification at the context service leading to a potential breach in the Subject's privacy control. The context service developer has to analyze if the increase of performance out-weights an eventual risk of an incorrect result.

6. Performance Evaluations and User Experiments

In order to evaluate the impact and the benefits of the proposed privacy service, we carried out two kinds of experiments, a performance evaluation and qualitative user studies. The first experiment was done to evaluate how much CoPS adds to the response time of a location service. The second one aimed at a qualitative assessment of how the test users would utilize the privacy controls offered by CoPS .

6.1. Performance Evaluation

In this section, we describe some performance tests that we did with the purpose of measuring CoPS' scalability and throughput. The goal was to measure how much the request processing by CoPS contributes to the total time spent for the context access request. In these tests, we used two machines, on which we executed the CoPS server and the clients, respectively. Both were 2.4 Mhz Pentium IV, 512 MB of RAM, running WindowsXP Professional connected to a fast-ethernet local network. In our experiments, we did not use CAA caching, and measured the response times of the rules' evaluation process with and without network latency.

In essence, we wanted to evaluate three issues. First, we wanted to measure how the response time increases as a function of the number of applicable privacy rules at all specificity phases. Second, we wanted to identify how response time increases as a function of the number of concurrent clients with a pre-defined amount of privacy rules analyzed at all specificity phases. Third, we wanted to measure the latency of the specificity algorithm disregarding the network latency.

In our first experiment, we populated the CoPS' rule database with a carefully selected set of privacy rules, in such a way that the same amount of privacy rules would be selected at each specificity phase. This experiment aimed to identify how the increase of the most-specific rule set (i.e., applicable privacy rules analyzed at each specificity phase, including temporal restriction, precision, application and result) contributes to the latency. Figure 4 outlines the results of this experiment. In this test, we ran one client that made 100 consecutive requests and measured the average elapsed response time to evaluate the requests. We carefully set up the rules' fields so that each different test could select a specific amount of most-specific rules to be analyzed at each specificity phase.

From this test, we have identified that the total number of rules in CoPS' database do not have direct impact on the

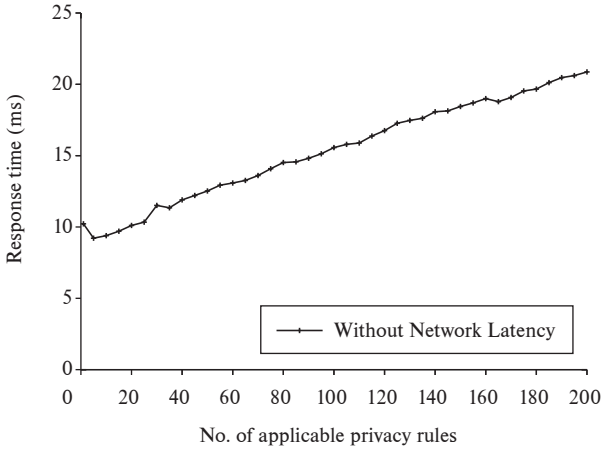

Figure 4. Response Time vs. Number of Applicable Privacy Rules.

response time latency, because the SQL queries used for retrieving the rules (depending on the Subject, Requester, context variable and access policy) already discard all the non-applicable privacy rules with low delay. In addition, the algorithm described in Section 4.3 shows that each phase of the specificity analysis may eliminate some rules for further analysis. Hence, it is important to note that the main bottleneck of the specificity evaluation algorithm is not the amount of applicable privacy rules selected through the SQL query, but the number of privacy rules processed at each specificity phase. As shown in Figure 4 the response latency has linear increase with the number of most-specific privacy rules. From this experiment, we can see that when the most-specific set is large (about 200 most-specific rules at all specificity phases for a single request) the response time is about 20ms. However, we believe this to be unlikely scenario, and that in practice in the worst-case, the specificity algorithm will select and evaluate, at each specificity phase, no more than 15 privacy rules for each request. Hence, we realized the following tests (shown in Figure 5) using a pre-defined set of 15 most-specific applicable rules.

Figure 5 shows the results of the second and third experiments, where we analyzed the average response time, varying the number of concurrent clients. For these tests, we populated CoPS' user database with 301 users: 300 possible Requesters and one Subject 'S1'. In order to reduce the amount of privacy rules, we grouped the 300 Requesters into 'MyFriend' group and created 15 privacy rules with the Subject and Requester fields holding 'S1' and 'MyFriend' values respectively. All these rules were at the INDIVIDUAL level and we assumed that the Liberal default access policy had been chosen. We also carefully set up the rules' fields so that all of them would be always selected/analyzed in all specificity phases for each request. We then ran an increasing number of concurrent requesting clients, where each client made the same request 100 times. Next, we measured the response times with and without network latency.

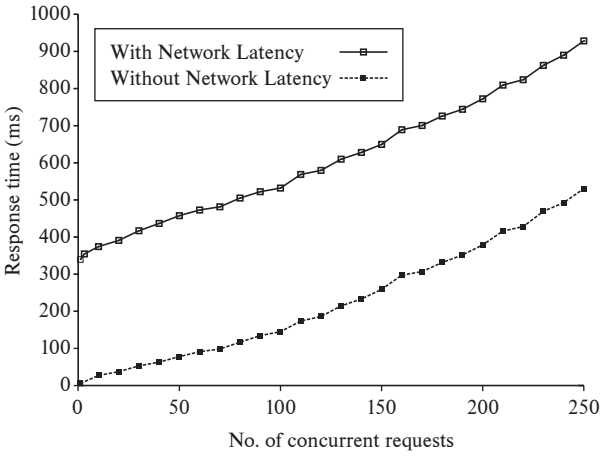

Figure 5. Response Time vs. No. of Concurrent Requests.

The only difference between these two experiments is that the second one evaluates the specificity algorithm's latency, but it does not take into account the network latency.

The results (Figure 5) show a linear increase of both the response time including network delay and the algorithm latency when the number of concurrent clients is increased. Furthermore, the results show that the specificity algorithm's latency has little influence on the total response times of a context access requests. These tests indicate that the overhead caused by the privacy control processing by CoPS represents only a small portion of the total round-trip-delay of context data access and processing by a Context Service.

6.2 Qualitative User Studies

Because privacy is a subjective concept with a wide range of individual variations in interpretations and preferences, we have carried out a study with typical users (in their particular social and cultural context). They were exposed to simulated situations where the level of location information disclosure was an important issue, and asked to use CoPS' privacy-control functions to adjust their privacy rules. The goal of the study was to gain a deeper understanding of the ways that the user configures and modifies her privacy settings, and not to generalize, or even predict, how *most* users would deal with privacy problems using CoPS .

The participants were interviewed and later played a game using a CoPS simulation. We used a simplified indoor-location-based multi-player treasure hunt game *LoMC* (Figure 6). The game provided instant messaging (chat) capability and means of querying the other participant's locations, as well as controlling disclosure of the own location. We then exposed the user to some hypothetic scenarios of a two-team treasury hunt competition (through a game instructions chat room), where she would have to interact and coordinate with their hypothetic team-mates their movements and the best location disclosure strategy to win the opponent team. Through a server interface (Figure 7) we monitored and logged all the user's actions, such as requests to access the other user's location, (simulated) movement to another location, changes of her own privacy settings, etc.

An analysis of the interviews and test behavior allowed us to appreciate whether the participants' perceptions converged to, or diverged from, some design assumptions related to the usability of CoPS. Specifically, the focus of the study was to find out how participants interpreted CoPS privacy control mechanisms. We wanted to know if the

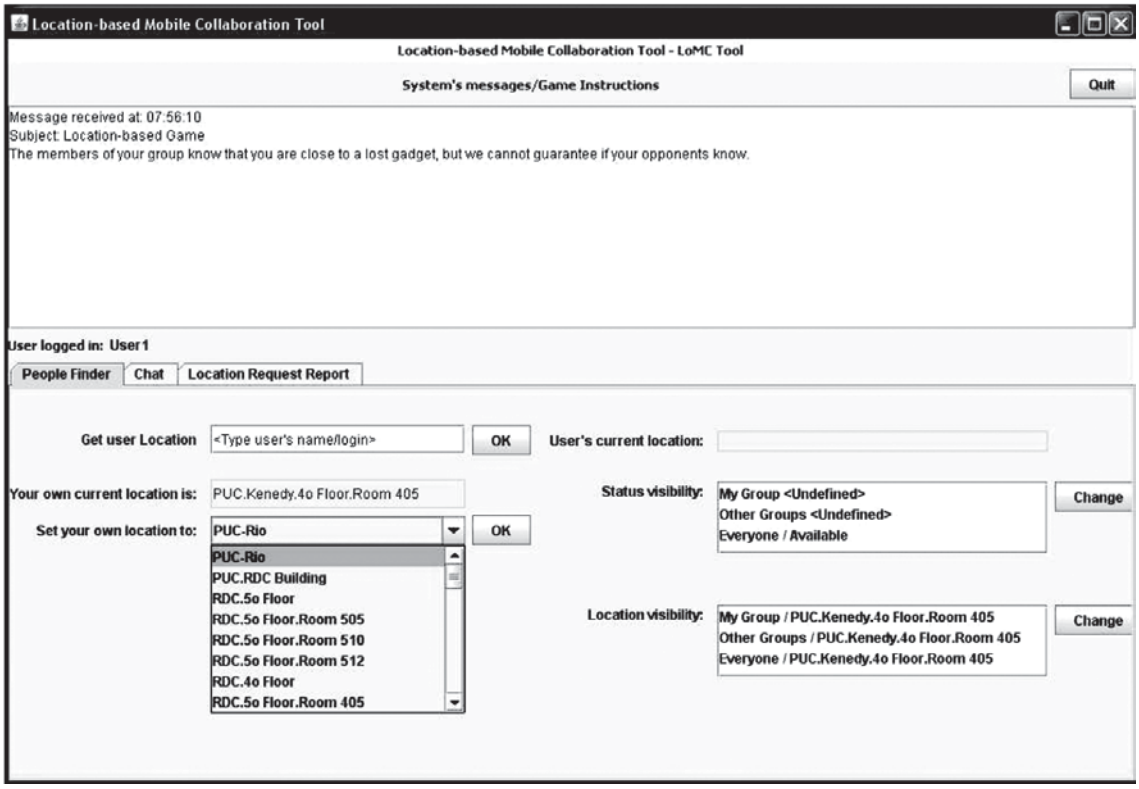

Figure 6. LoMC Client.

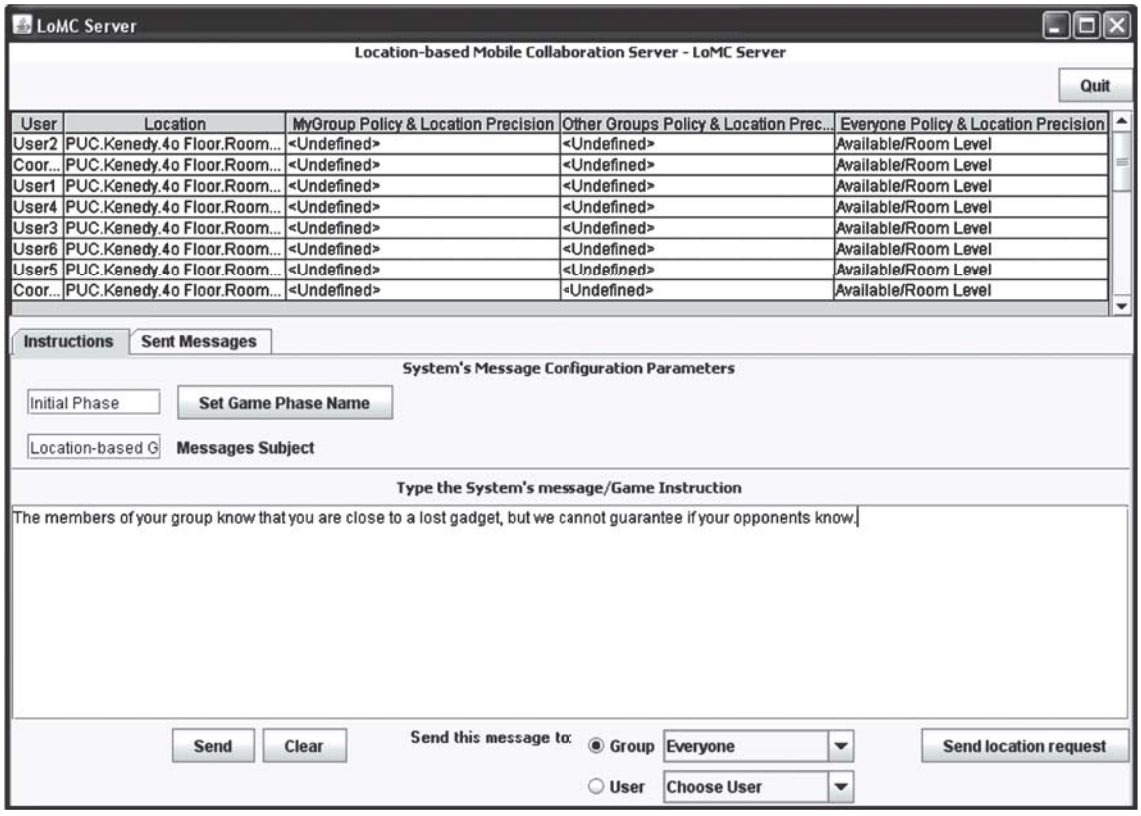

Figure 7. LoMC Server.

users perceived them to be effective (i.e., "if they worked") and useful (i.e., if they met their expectations) with respect to handling privacy issues in location-based services.

Qualitative studies usually require long preparation and laborious execution [45]. Our experiment required the preparation of the hypothetical game scenarios, the development of the simulator of the game, and the selection of methods and techniques for the interviews and the simulation. We made audio recordings of the interviews and video recordings of the participants' interaction with the simulator. During the interviews, participants were asked open-ended questions, whose answers were compared, during the analysis and interpretation phase, with their attitudes and decisions while running the game simulation.

In the experiment, only a subset of the controls presented in Section 3.1 was used (e.g., controls for location visibility and notifications). We have decided not to run the experiment with all CoPS controls for mainly two reasons: the complexity of simulating comprehensible and plausible scenarios for each and every control dimension; and the influence of interface design decisions on the users' interpretation of privacy control features of the simulator. In other words, simulating the use of all controls would require the design and implementation of a very game-specific and intuitive user interface, whose implementation would not be cost-effective and justifiable for the purposes of our qualitative studies. The critical question in selecting a subset of controls was, however, to know whether they were sufficiently representative of the kinds of privacy-related issues that using CoPS is likely to entail. Our choice was informed by previous studies reported in the literature, and, as will be shown, the subset of controls in our experiments gave rise to a rich collection of findings.

6.2.1. Methodology

Our studies were designed to collect feedback from participants with respect to a number of CoPS design hypotheses, briefly described below (for a detailed description, see [40]). In particular, we were interested in finding out whether there were reasons for rejecting them.

Design Hypotheses involving Users' Needs and Preferences:

Hypothesis 1: There are situations when users want to balance privacy against sociability, disclosing their location information in different levels of detail, depending on such factors as the time of day and the potential requesters.

Hypothesis 2: Some users don't want to go into the details of configuring privacy-protection controls when using LBS. However, they want to know what information is being broadcast about them, to whom, and how.

Hypothesis 3: Some users don't care whether their location is known by other users or not.

Hypothesis 4: Some users are willing to create different privacy profiles to match different social roles that they play (e.g., teacher, supervisor, project coordinator).

Hypothesis 5: There are certain social situations where users want to deny access to their location information, but they don't want requesters to know they are doing it.

In order to assess the plausibility of the above design hypotheses, we elaborated a series of plausible privacy-sensitive social scenarios, and cast them in the context of a game. We also elaborated a set of open-ended questions that participants should answer before and after they played the simulated game. Such questions addressed aspects of the game scenario, both related to specific design hypotheses. Thus, we could triangulate conclusions about the value of CoPS designed features.

The procedures of the experiment started with the interviews, whose questions were divided into two groups. The first group of questions aimed at finding out how respondents dealt with privacy issued while using chat, instant messaging, and cell phones. The second, aimed at finding what they rationally think they would do while using LBS to play a `quest' game (competitors should find the location of various objects, based on cues they were given individually). Two examples of pre-test interview questions are:

- *Do you have personal privacy criteria to answer a cell phone call? What about for making a call yourself?*
- *Do you see a problem if a player decides to disclose his location to his group or to his opponents?*

The next procedure was the game. First, participants were given a short tutorial where we explained the game itself and how to use the simulator. The quest game was played by two competing teams. Each player's goal was to help his or her team find a number of hidden objects around campus before the other team did it. The quest was to take place in their habitual campus life context, i.e., some of their team members might be in class or in a meeting with their supervisors, while others might be free to look for the objects. All communication among team members was done via the simulator. The game simulator worked like a text-based MUD or RPG system (e.g., Adventures Unlimited - http://www.tharel.net/), and location information was signified by the names of places around campus (like buildings, cafeterias, lounges, etc). The simulator was controlled by one of the authors, who was the game manager, capable of sending different scenario parts to the players, inducing different situations and unfolding of the game strategy. For example, the simulator could send a player a text message shown in Figure 7: "The members of your group know that you are close to a lost

gadget, but we cannot know for sure if your opponents know it or not."

Through the simulator's interface (Figures 6 and 7) participants could control the broadcast conditions of his location information, as well as to request information from all other participants (in his or her team, and in the other team). Commands issued through the interface were actually executed by the human game manager in a 'Wizard of Oz' setting [27, 26].

The final experiment procedure was a post-test interview, where we tried to elicit the participants' experience with the simulator, trying to find out if they could use the simulator's interface, if the controls were felt to be useful, what were the main difficulties, how difficult it would be to manage those privacy controls in real life situations, etc. Altogether, participants were involved in the study for approximately 90 minutes.

6.2.2. Findings

In pre-test interviews participants reported that they do care about privacy online if personal data is involved or if others can 'draw conclusions about them', like inferring that they are working, or just passing time, or with their friend, and so on. As one participant said: *"I care that people may have access to my personal information, or that they can find out that I am online and what I'm doing."*

Nevertheless, we found cases where participants who said they cared about their privacy did not even bother to block their competitors' access to their location information. Such access did allow competitors to draw relevant conclusions about what the participant's whereabouts, and thus there was an apparent contradiction between what the participants said and what they did. Engagement may have played a factor. Perhaps the simulated game was not engaging enough for some participants, but in all recorded interaction shows that most participants did control other people's access to their location information. This reinforced our first design hypothesis.

We also found that participants disclosed their exact location information for members of the same team, and imposed some kind of restriction to members of the competing team. We took this to mean that in real life situations they have strategies for deciding the level of disclosure they grant to different people in different contexts.

Participants declared that they believed it would be too difficult to update their privacy disclosure preferences in various contexts. There are too many controls involved and the diversity of privacy-sensitive situations is wide. Most participants felt that pre-configured profiles (e.g., 'In a Meeting', 'Taking a Break', etc.) could be used to alleviate this complexity, especially if there was a way to automatically trigger such profiles and if exceptions were easy to handle. One participant said:

"I would like to create profiles to control privacy-sensitive information. If the user has to configure each and every parameter every time, he is likely to make mistakes and give access to his location when he does not mean to."

This reinforces our fourth design hypothesis.

However, our experiments showed evidence that weakens the fifth design hypothesis. When they found themselves in situations where we thought they would choose to 'Deny Access', they did not do it. In post-test interviews most users said that they did not remember this option or that they did not understand it. We think this may have happened for different reasons. One is that the interface design made this option more complicated to find than it should be. The other is that in the game scenario itself the cues for using this functionality were not properly motivated. Therefore, further studies must be carried out in order to assess our fifth design hypothesis.

We also found that our design hypotheses may be reinforced or weakened as different social circumstances arise. For example, if users think that there is no particular risk in letting others know where they are, they may just let their location be informed, taking an occasional look at access request reports and notification. This goes against our first design hypotheses, and reinforces the second and third. In perceived risk situations, however, users block access to all location information, and carefully control how/if they are seen by others. Here our first design hypothesis is stronger than the second and third. Therefore, the attitude of users towards our design hypotheses is not one of definitive accept or reject. It is contingent to the social and psychological circumstances of the users.

If such contingencies were found in a small group of participants, we can expect a much more intricate and diverse set of contingencies to be found as we expand the group of observed users. Consequently, we believe we cannot make useful generalizations with respect to the users' expectations and needs. What can be learned from our experiment, however, is that effective and agile communication of design features and system functionality probably constitutes the prime challenge of this kind of technology.

Along these lines, we collected evidence that sometimes users 'waste' the design effort because they 'forget' that some feature can be used. Yet, at the same time, participants expressed their belief that maybe the technology should have even 'more [sophisticated] control features'. This observation echoes an early finding by [6] about the 'paradox of the active user'. The authors found that there are conflicting cognitive strategies at work when users interact with computer technology. On the one hand, their social context offers high payoff for 'getting things done'

(productivity) and low payoff for 'learning how to do things better'. On the other, users' performance quickly reaches an asymptote at mediocre levels of efficiency, where users can 'get things done' (one way or another). Together, these facts suggest that users want technology to do smart things, but they don't want to go into the trouble of having to learn it. For designers and developers of computer technology, this attitude may be very disorienting. Carroll & Rosson suggest that:

"There are two ways we might reduce the motivational 'cost' of learning: make the learning safer and more risk-free, or make the relevant information easier to find. If trying out a new function is perceived as risk-free, a learner may be more willing to try it; it is less likely to interfere with the goal of producing something. Several design approaches have been taken in promoting the "safety" of systems during training. These fall into two classes - controlling the consequences of any given action by the user, and controlling the actions available to the user"[6].

Since 'controlling the consequences of any given action by the user' when privacy is involved is not a realistic goal, we might try to control 'the actions available to the user'. But, it is not clear if this kind of design direction would not cause more (instead of less) social blunders to users. Hence, further studies are clearly needed.

7. Related Work

There are several research work that deal with privacy issues related to anonymity [21, 15], data confidentiality (through cryptography mechanisms) [20, 19], access control [11], privacy architecture to the Web [24], among others. However, we cared to make a comparison with works that propose an approach, a service or an architecture to handle privacy issues related specifically to location- or context-aware applications.

Context Fabric (Confab) [22] is an architecture that offers privacy control mechanisms tailored to location information. The design of Confab's architecture is based on several privacy requirements that stress flexibility in the use of privacy-aware applications. Compared to this work, we incorporated functionalities to CoPS that offer more flexibility for the user's and organization's privacy policy management. For example, we defined mechanisms for fine grained access control, user-defined privacy profiles, hierarchy of privacy policies, access control policies, access reports, rule templates, privacy profiles, among others.

Unlike CoPS, CoBrA (*Context Broker Architecture*) [7, 8] and *pawS* (*Privacy Awareness System*) [28] use the Requester's location as a key information to determine which access restrictions should be applied for disclosure of each kind of context information. However, the authors of these works do not discuss how their systems would behave when the required location information is not available.

Myles [33] proposed a *component framework* for privacy control which has been integrated to the location service *LocServ (Location Service)*. Through *Validator* components, *LocServ* evaluates if the user's location requested by a third party may be disclosed or not. The system presumes that a trustworthy organization will define a set of *Validators*, in the attempt not to overwhelm the user with this task. For atypical situations in which the default privacy policy implemented by *Validators* is not appropriate, the users may create new *Validators* by using tools that the authors considered "simple" and "intuitive". CoPS has some similarities with this project. For example, both support time-related access restrictions, user groups and rule templates. However, in CoPS 's design we have considered a more practical and feasible approach for dealing with the challenge of the continuous maintenance of the users' privacy policy, instead of just assuming that the system administrator is able to foresee the users' privacy needs or that some "simple tools" will effectively help the user to manage her privacy policies.

The IETF (*Internet Engineering Task Force*) Geopriv [44] identified the need of processing and transferring the location information among location-based services and applications preserving the privacy of the involved users. Geopriv creates a location object that encapsulates the users' location information and the requirements of privacy associated to the information. Some of the privacy requirements of CoPS are similar to the ones defined by IETF Geopriv, for example, the disclosure of the location information at different granularities and under temporal restrictions. However, because of the amplitude of the proposal and the complexity of dealing with privacy issues in a general setting, Geopriv is silent about several issues related to the management and maintenance of the users' privacy. As discussed by Harper [18], the rejection or adoption of a technology may be the consequence of an ideological attitude which, a priori, bears no direct relation to the purposes of the technology. This makes us believe that general proposals tend to be unsuccessful because of the difficulty to take into consideration the very diverse ways that people in different communities react to intrusiveness by technologies.

Neisse et al. [34] have proposed the concept of a *Context-Aware Management Domain* as an abstraction that allows to enforce more generic, context-aware policies (of different types and purposes) on dynamic sets of entities, i.e., Requesters and Subjects, according to their association with a common context situation, such as "persons nearby", which holds for persons that are closer than a given distance. For example, such a context-aware privacy

obligation policy could be "when Bob's identity is requested by nearby persons it should not be anonymized". While this abstract means of defining policies may be helpful for some kinds of policies, we don't believe that it is suitable for privacy control. In our point of view, a user's privacy preference largely depends on who the requester is, and hence cannot be formulated just in terms of a situation context. Moreover, unlike CoPS, this work does not take into account that privacy rules usually need to be continuously updated by the user, not only because it is difficult to get the initial preferences wright, but also because the user privacy requirements may change according to his current situation, activity or mood, all of which can hardly be precisely specified through a context situation in advance. Yet another difference is that the proposed policy provider does not support fine-grained control of context disclosure, such as selection of a specific (context) data precision or freshness level, which gives the user with more flexibility for personalizing his privacy policies.

Compared to these works [9, 3, 36, 25, 22, 8, 33], the main contribution of our work is that CoPS offers a larger, more comprehensible and effective set of privacy control mechanisms that assist the user in building and maintaining her privacy policy. For example, we have defined new design principles (e.g., fine-grained access control, user-defined privacy profiles, hierarchy of privacy policies) and incorporated new functionalities for privacy control into CoPS (e.g., Policy Hierarchies, the Specificity Algorithm, Access Policies, Privacy Profiles, among others) that offer more flexibility for the management of privacy control by users and organizations. We believe that these functionalities are of fundamental importance to the usefulness of a privacy service.

8. Conclusion

From the qualitative user studies we could identify some users' opinions and attitudes that reinforce the benefits and usability of CoPS 's privacy controls. Several users expressed the wish, or need, to find a compromise between sociability and privacy, e.g., by disclosing their location information with varying precision to different groups of Requesters. Users also stated that LBS applications for collaboration and communication may be very useful, but that without a proper mechanism that support fine-grained and customizable control of location information disclosure, these applications would present high risks to their privacy, and would compromise the benefits.

The notion of privacy is very personal, and there are substantial variations of the perceptions and attitudes that people have in relation to their privacy, e.g., we always find users that are very little, or very worried with their privacy. Consequently, we believe that in order to provide support for most users' needs, the privacy service should feature a significant set of diverse privacy controls, giving the user a maximum of flexibility (to shape his individual privacy policy) and ease of managing it. In particular, these controls should support both conservative and liberal (aka. moderate) approaches toward privacy. For example, they should allow users to, *a priori*, deny all access to the own location, or enable users to disclose their location but monitor the accesses in parallel in order to identify and inhibit possible abuses.

As discussed in the results of the interviews and experiments, *the users should have much flexibility, but should not be overwhelmed with the configuration of their privacy preferences*. This conclusion has pointed to us one of the main problems related to the design of a privacy service, which can be summarized by the dichotomy Flexibility *versus* Complexity of privacy policy management. With CoPS the user does not need to define her privacy policy before she begins to utilize the LBS application, because she can use the *Default* policy rules. Then, while she is using the LBS application, she can gradually define her rules, by interacting with the system (by choosing the On-Demand policy, or creating rules with an "Ask Me" result), and by such she will naturally create own her privacy policy, customized to her current need. Furthermore, the user can select the "Liberal" policy, where she is able to identify eventual abuses/intrusions through notifications and reports about access attempts, or adopt a conservative approach through the "Reserved" policy. The user may also create temporary rules, or set up privacy profiles, that will help her to change the current policy according to the circumstances.

It's worthwhile to emphasize that, as discussed by [37, 39], there is neither a unique nor complete solution that ensures the users' privacy. In order to achieve the privacy level closest to the desired one, we should take into account the combination of different resources such as the legislation with its well-defined penalties to the possible violations, the social and/or corporative norms, the possible technological solutions and, finally also the belief in the appropriate social conduct of users and companies, that should follow the social protocol and ethic guidelines established in the society.

References

[1] I. Altman. The environment and social behavior: Privacy, personal space, territory and crowding. In *Proceedings of Monterey, CA: Brooks/Cole Pub. Co., Inc.*, 1975.

[2] I. Altman. Privacy regulation: Culturally universal or culturally specific? *Journal of Social Issues*, 33(3): 66-84, 1977.

[3] D. Anthony, T. Henderson, D. Kotz. Privacy in location-aware computing environments. *IEEE Pervasive Computing*, 6(4):64-72, 2007.

[4] L. Barkhuus, A. K. Dey. Location-based services for mobile telephony: a study of users' privacy concerns. In *Proceedings of INTERACT*, 2003.

[5] A. R. Beresford, F. Stajano. Location privacy in pervasive computing. *IEEE Pervasive Computing*, 2(1): 46-55, 2003.

[6] J. M. Carroll, M. B. Rosson. Paradox of the active user. In *Proceedings of Interfacing Thought: Cognitive Aspects of Human-Computer Interaction*, Cambridge, pages 80-111, 1987.

[7] H. Chen, T. Finin, A. Joshi. A context broker for building smart meeting rooms. In *Proceedings of the Knowledge Representation and Ontology for Autonomous Systems Symposium, AAAI Spring Symposium*, Stanford, California, pages 53-60, 2004.

[8] H. L. Chen. An intelligent broker architecture for context-aware systems. Unpublished Phd thesis, University of Maryland, 2005.

[9] C. Cornelius, A. Kapadia, D. Kotz, D. Peebles, M. Shin, N. Triandopoulos. AnonySense: Privacy-aware people-centric sensing. In *Proceedings of the 2008 International Conference on Mobile Systems, Applications, and Services (MobiSys)*, pages 211-224, 2008.

[10] Inc. Ekahau. http://www.ekahau.com/, Jan. 2005.

[11] D. Ferraiolo, R. Kuhn. Role-based access controls. In *Proceedings of 15th NIST National Institute of Standards and Technology-NCSC National Computer Security Conference*, pages 554-563, 1992.

[12] E. Goffman. The Presentation of Self in Everyday Life. Doubleday, New York, 1956.

[13] K. Gonηalves, H. K. Rubinsztejn, M. Endler, B. Silva, S. D. J. Barbosa. Um aplicativo para comunicaηγo baseada em localizaηγo. In *Proceedings of VI Workshop de Comunicaηγo sem Fio e Computaηγo Mσvel*, pages 224-231, 2004.

[14] J. Grudin. Desituating action: Digital representation of context. In *Proceedings of HCI '01: Human-Computer Interaction*, pages 269-286, 2001.

[15] M. Gruteser, D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proceedings of MobiSys '03: Proc. First International Conference on Mobile Systems, Applications, and Services*, pages 31-42, 2003.

[16] A. Haeberlen, E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, L. E. Kavraki. Practical robust localization over large-scale 802.11 wireless networks. In *Proceedings of MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*, New York, pages 70-84, 2004.

[17] R. K. Harle, A. Hopper. Deploying and evaluating a location-aware system. In *Proceedings of MobiSys '05: 3rd international conference on Mobile systems, applications, and services*, New York, pages 219-232, 2005.

[18] R. H. R. Harper. Why people do and don't wear active badges: a case study. *Comput. Supported Coop. Work*, 4(4):297-318, 1996.

[19] U. Hengartner. Access Control to Information in Pervasive Computing Environments. Phd Thesis, Carnegie Mellon University, School of Computer Science, Aug 2005.

[20] U. Hengartner, P. Steenkiste. Implementing access control to people location information. In *Proceedings of SACMAT '04: 9th ACM symposium on Access control models and technologies*, New York, pages 11-20, 2004.

[21] B. Hoh, M. Gruteser. Location privacy through path confusion. In *Proceedings of SecureComm '2005: 1st IEEE/CreatNet International Conference on Security and Privacy for Emerging Areas in Communication Networks*, Sept 2005.

[22] J. I. Hong. An Architecture for Privacy-Sensitive Ubiquitous Computing. Phd Thesis, University of California at Berkeley, Computer Science Division, 2005.

[23] J. I. Hong, J. A. Landay. An architecture for privacy-sensitive ubiquitous computing. In *Proceedings of MobiSYS '04: 2nd international conference on Mobile systems, applications, and services*, New York, pages 177-189, 2004.

[24] L. Ishitani. Uma Arquitetura para Controle de Privacidade na Web. Phd Thesis, Universidade Federal de Minas Gerais, 2003.

[25] A. Kapadia, T. Henderson, J. J. Fielding, D. Kotz. Virtual walls: Protecting digital privacy in pervasive environments. In *Proceedings of the Fifth International Conference on Pervasive Computing (Pervasive)*, volume 4480 of *LNCS*, London, pages 162-179, May 2007.

[26] J. F. Kelley. An empirical methodology for writing user-friendly natural language computer applications. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, New York, pages 193-196, 1983.

[27] J. F. Kelley. An iterative design methodology for user-friendly natural language office information applications. *ACM Trans. Inf. Syst.*, 2(1):26-41, 1984.

[28] M. Langheinrich. A privacy awareness system for ubiquitous computing environments. In *Proceedings of the 4th international conference on Ubiquitous Computing*, London, pages 237-245, 2002.

[29] S. Lederer, J. Mankoff, A. K. Dey. Who wants to know what when? Privacy preference determinants in ubiquitous computing. In *Proceedings of CHI '03: CHI '03 extended abstracts on Human factors in computing systems*, New York, pages 724-725, 2003.

[30] MoCATeam. Moca applications home page. http://www.lac.inf.pucrio.br/moca/applications.html/, Apr 2007.

[31] MoCATeam. Moca home page. http://www.lac.inf.puc-rio.br/moca/, Apr 2007.

[32] MoCATeam. Results of the user survey about privacy and spontaneous collaboration. http://www-di.inf.puc-rio.br/endler/pub/SurveyPrivacy-Results.htm/, Apr 2007.

[33] G. Myles, A. Friday, N. Davies. Preserving privacy in environments with location-based applications. *IEEE Pervasive Computing*, 2(1):56-64, 2003.

[34] R. Neisse, M. Wegdam, P. D. Costa, M. J. van Sinderen. Context-aware management domains. In *Proceedings of* B. Hulsebosch, G. Lenzini, M. Wegdam, editors, *Context Awareness and Trust 2007 (CAT07), First International Workshop on Combining Context with Trust, Security and Privacy, Moncton, Canada*, volume 269 of *CEUR Workshop Proceedings*, pages 42-47, July 2007.

[35] S. Patil, J. Lai. Who gets to know what when: configuring privacy permissions in an awareness application. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, New York, pages 101-110, 2005.

[36] C. Patrikakis, P. Karamolegkos, A. Voulodimos, et al. Security and privacy in pervasive computing. *IEEE Pervasive Computing*, 6(4):73-75, 2007.

[37] L. Perusco, K. Michael. Control, trust, privacy, and security: evaluating location-based services. *IEEE Technology and Society Magazine*, 26(1):4-16, 2007.

[38] M. Raento, A. Oulasvirta, R. Petit, H. Toivonen. Contextphone: A prototyping platform for context-aware mobile applications. *IEEE Pervasive Computing*, 4(2):51-59, 2005.

[39] M. Rotenberg, C. Laurant. Privacy for human rights 2004: An internation survey of privacy laws and development. Electronic Privacy Information Center, Washington D.C. 2004.

[40] V. Sacramento. Privacy Management for Context-Aware Applications in Mobile Networks. Phd Thesis, Pontifical Catholic University of Rio de Janeiro/PUC-Rio, Sept. 2006.

[41] V. Sacramento, M. Endler, F. N. Nascimento. A privacy service for context-aware mobile services. In *Proceedings of SecureComm '2005: First IEEE/CreatNet International Conference on Security and Privacy for Emerging Areas in Communication Networks*, pages 182-193. Sept. 2005.

[42] V. Sacramento, M. Endler, H. K. Rubinsztejn, L. S. Lima, K. Goncalves, F. N. Nascimento, G. A. Bueno. Moca: A middleware for developing collaborative applications for mobile users. *IEEE Distributed Systems Online*, 5(10):2, 2004.

[43] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, C. E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38-47, 1996.

[44] H. Schulzrinne, H. Tschofenig, J. B. Morris, J. R. Cuellar, J. Polk, J. Rosenberg. Geolocation policy: A document format for expressing privacy preferences for location information, June 2008.

[45] E. R. Turato. Tratado Da Metodologia Da Pesquisa Clinico Qualitativa. Vozes, 2003.

[46] A. F. Westin. Privacy and freedom. In *Proceedings of New York*, 1967.