PRECISE: Privacy-Aware Recommender Based on Context Information for Cloud Service Environments

Alberto Huertas Celdrán, Manuel Gil Pérez, Félix J. García Clemente, and Gregorio Martínez Pérez

The authors are with the University of Murcia.

This work has been partially funded with support from the Spanish MICINN (project RECLAMO, Virtual and Collaborative Honeynets based on Trust Management and Autonomous Systems applied to Intrusion Management, with code TIN2011-28287-C02-02, and project Mejora de Arquitectura de Servidores, Servicios y Aplicaciones, with code TIN2012-38341-C04-03) and the European Commission (FEDER/ERDF). Thanks also to the Funding Program for Research Groups of Excellence granted by the Séneca Foundation with code 04552/GERM/06.

ABSTRACT

Context-aware systems based on location open up new possibilities to users in terms of acquiring custom services by gathering context information, especially in systems where the high mobility of users increases their usability. In this context, this article presents a privacy-preserving solution offering context-aware services based on location in MCC. We propose a middleware, called PRECISE, which provides users with custom context-aware recommendations. These recommendations are given by considering the context information, and the users' locations, privacy policies, and previously visited places. MCC plays a key role in this solution, moving the data processing and storage needs to the cloud, as well as further advantages such as elasticity and load balancing. A thorough discussion when comparing PRECISE with other related works confirms that our solution improves the most relevant proposals so far.

INTRODUCTION

Context awareness is a concept that combines the environment, users' locations, identities of nearby people and objects, and changes in the previous terms [1]. The ability to determine users' and objects' locations at a particular time supplies valuable information to offer services from the elements of the environment where they are. These services can range from a simple location-based service (LBS) to emergency services, car navigation systems, or tour planning for tourists.

Nowadays, we can find some proposals that aim at offering context-aware services based on location, such as LOC8 [2], SOCAM [3], and CoCA [4]. LOC8 is a powerful framework that supports querying the environment in which users are located, whereas SOCAM infers information about a given context considering information from others. Instead, CoCA proposes a collaborative context-aware service platform where the users' location is inferred by considering information about the context and the location of the elements.

However, the ability to locate people raises serious privacy concerns, such as the right of users to determine when, how, and under what conditions their location can be released to other users or services. In this sense, other context-aware systems are focused on preserving location privacy by using policies. Among them, CoBrA [5] protects the privacy of its users by using rules, inferring whether they have the right permissions to share and/or receive information. Another example is CoPS [6], where users can control who can access their context data, when, and at what level of granularity. Finally, a more recent work, PPCS [7], proposes that the users' location and context information are shared (or not) depending on their privacy policies.

With previous solutions, users cannot protect their privacy against services, only against other users. Furthermore, they are oriented to offer local services in specific environments, setting aside distributed schemes. Providing context-aware services based on location is a complex task due to:

- The need to acquire the users' location and context-aware information from different sources
- The large amount of data required to shape the context-aware information
- The processing power requirements to analyze the information and provide the services
- The mobility of users when they make use of the services
- The management of the users' privacy according to their preferences
- The need to offer services that run on federated systems spanning different organizations

In attempting to meet these requirements, we figured out that the mobile cloud computing (MCC) paradigm can play a key role [8]. MCC provides mobile users with data processing and storage services in a cloud environment. Thus, mobile devices belonging to roaming users do not need large capacity for storage and processing. Furthermore, context-aware services based on location can share certain information to provide users with better and more personalized services, by using federation mechanisms as the one presented in [9]. To the best of our knowledge, there is not a wide spread of context-aware services based on location facing users' privacy concerns in MCC, this being a priority recently identified in [8].

In this article, we present an evolution of our work called *SeCoMan* [10]. Concretely, we propose a privacy-preserving and context-aware system that provides location-based services (LBS) in the MCC paradigm, which allows users to define privacy policies regarding services (although it is able to do it between users too, as an extension of the SeCoMan approach). This implies that the architectures and the benefits obtained from both solutions are rather different. With PRECISE, users can:

- *Release* their locations to specific services
- *Hide* their positions to specific users
- *Mask* their locations to other users by generating fictitious (fake) positions
- Establish the *granularity* and *closeness* at which they want to be located by services or users
- Preserve their *anonymity* to specific services

The central component of our solution is a middleware, called Privacy-Aware Recommender Based on Context Information for Cloud Service Environments (PRECISE), placed between users and context-aware services. PRECISE is allocated at the platform as a service (PaaS) layer of the MCC paradigm. This manages the context-aware users' information, their privacy policies, and their behavior patterns (i.e., the users' movements while staying in the environment). Context-aware services are in turn allocated at the software as a service (SaaS) layer of MCC, providing users with recommendations about context-aware information. Thus, PRECISE is a context-aware system that receives the users' location, context information, and changes in both from independent middleware, which can belong to the cloud environment or not.

How to Preserve Users' Privacy in LBS

Users should be able to dynamically control their information by using rules that form policies, for example. Among the different sets of policies, we emphasize here the use of privacy-oriented policies, with which users can define their privacy preferences related to their location, identity, and personal information.

The set of rules forming the policies in PRECISE are composed of the following elements: *Type* is the kind of policy; the *target* is the person who defines the policy and whose information is being managed; the *requester* is the service or person who requests information about the target; the *place* is the region of the environment where the policy will be enforced; and the *result* determines the relationship the requester will have with the target's information. PRECISE is in charge of managing the context-aware information implicitly contained in each rule. Note that the context can be composed of some elements, such as *Profile*, *Date*, or *Time*.

$$Type \wedge Target \wedge Requester \wedge Place \wedge [Profile] \wedge [Date] \wedge [Time] \rightarrow \text{Result}$$

In the above context, PRECISE allows users to define two kinds of policies, which are introduced below.

Location Management Policies

These policies are defined by the users to specify their privacy preferences regarding their location. They are divided into five groups: *Release*, *Hiding*, *Cloaking*, *Granularity*, and *Closeness*. In any of the foregoing policies, the target can define different policies with the same requester regarding certain *Places*, *Profiles*, *Dates*, or *Times*, whether this requester is a user or a service, depending on the kind of policy.

As PRECISE hides the users' location to services by default, release policies are oriented to reveal the users' locations to specific services. This will allow a requester (service) to know the target's position (user). On the other hand, hiding and cloaking policies are oriented to users, as, unlike in the previous case, the users' location is available to other users by default. They can hide their location to a given set of users by defining hiding policies. Cloaking (or masking) policies are intended to generate one or more fake positions for a particular user, so other users cannot distinguish the real position in which the target is. Granularity policies can also be defined indicating the maximum accuracy at which users want to be located, may it be applied to users or services. With this kind of policy, it is possible to define several levels of granularity depending on the context in which the service is being provided. An example of these levels could be the section or building where the user is located.

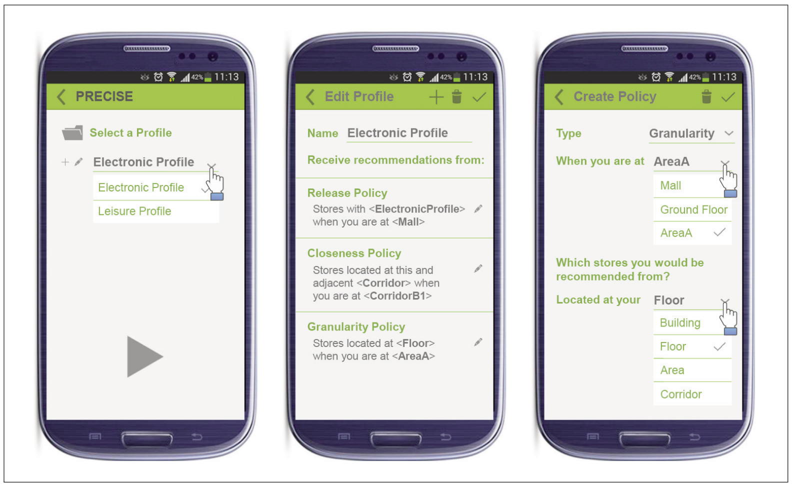
Finally, closeness policies can be applied to both users and services with the aim of indicating the minimum level of nearness at which the target (user) wants to be located. The nearness level is established with the same values as the ones defined for the granularity policies.

Anonymity Management Policies

PRECISE guarantees anonymous use of context-aware services, as users do not want a priori to reveal either their identity or personal information on what they consider sensitive data. To disclose such information, we define a new type of policy, called *Revealing*, oriented to services. This kind of policy can be specified by the users to receive custom recommendations from specific context-aware services. Otherwise, users will only receive general context-aware recommendations. In the literature, we can find several proposals in overcoming the anonymity of a user. For example, in [11], a solution for the anonymous use of services by using pseudonyms is proposed.

Depending on the users' needs and preferences, they can use some *profiles* defined by the system, or created by them, to reveal the right information (e.g., their location only) to the right requester (service), at the right place and time, in order to receive custom context-aware recommendations. A profile in our solution is related to a specific topic and is formed by a given set of policies. As an example, Fig. 1 shows how a user can select a given profile to start receiving recommendations, edit an already defined profile, or create a determined policy by using a mobile application. Note that the content of these three subfigures are subsequently explained in the next section.

Users should be able to dynamically control their information, by using rules that form policies, for example. Among the different sets of policies, we emphasize here the use of the privacy-oriented policies, with which users can define their privacy preferences related to their location, identity, and personal information.


11:13
PRECISE
Select a Profile
Edit Profile
Receive recommendations from:
Create Policy
Which stores you would be recommended from?

Electronic Profile ✓
Electronic Profile
Leisure Profile

Name: Electronic Profile

Release Policy
Stores with <**ElectronicProfile**> when you are at <**Mall**>

Closeness Policy
Stores located at this and adjacent <**Corridor**> when you are at <**CorridorB1**>

Granularity Policy
Stores located at <**Floor**> when you are at <**AreaA**>

Type: Granularity
When you are at: AreaA — Mall, Ground Floor, AreaA ✓
Located at your: Floor — Building, Floor ✓, Area, Corridor

Figure 1. How a user can choose a given profile and how the policies can be created or edited through a mobile application: a) selecting a given profile to start receiving custom context-aware recommendations; b) editing a profile to receive recommendations when meeting a number of policies; c) defining a granularity policy about where receiving recommendations.

A Motivating Example

We present in this section an example to show how our solution manages the users' privacy in context-aware services based on their location. We define a scenario based on a mall where users receive recommendations about products and offers on their mobile devices. In this scenario, the main entity of PRECISE acts as the mall's middleware itself, giving recommendations according to the users' location, their profiles, and the context-aware information related to the space description of the environment, the description of the elements close to the user, and their relevance and meaning.

In our scenario, we have a mall with different stores and places to buy products and enjoy leisure time. This mall has the infrastructure to know the location of its users and a service to provide custom context-aware recommendations. Users' location is registered by the mall's middleware, that is, by PRECISE, to record their patterns of behavior when interacting with all services and stores provided in the mall; for example, time visiting a given store or routines of the users after staying in a given space. Using this information, PRECISE (the mall's middleware) is able to give useful recommendations to its users about products and offers provided by the stores located at the mall. These recommendations take into consideration the context-aware information, the users' location, some profiles defining typical customs of the users, and patterns about their behavior. Changes in the context are managed by PRECISE receiving the context-aware information from the infrastructure deployed at the mall. The map of the mall is graphically depicted in Fig. 2.

As an example to show how the previous mall's middleware (PRECISE) gives recommendations to its users, suppose a user (*Andy*) moving around the shopping center who is interested in electronic products. To receive recommendations, he uses his mobile device, access to the mall's middleware, and selects *ElectronicProfile* (Fig. 1a).

Electronic Profile

This profile contains policies that form the Andy's privacy preferences. The first rule consists of a *release* policy, indicating that his position has to be released to the services with *ElectronicProfile* when he is at the mall (Fig. 1b). Thus, he will only receive recommendations of electronic stores.

Person(*#Andy*)∧*isLocated*(*#Andy*, *#Mall*)∧ *Service*(*?Requester*)∧*hasProfile*(*?Requester*, *#ElectronicProfile*) → *hasRelease*(*#Andy*, *?Requester*)

The next rule is a *closeness* policy. It indicates that Andy only wants to receive recommendations of stores located in the same and adjacent corridors when he is located in *CorridorB1* (recommendation point 1 in Fig. 2). As a result, Andy will only receive recommendations of ElectronicStoreB.

Person(*#Andy*)∧*isLocated*(*#Andy*, *#CorridorB1*) → *hasGranularity*(*#Andy*, *#Floor*)

The last rule is a *granularity* policy (Fig. 1c), indicating that Andy only wants to receive recommendations of the stores placed on the same floor when he is at *AreaA* (recommendation point 2, Fig. 2). Thus, Andy will receive recommendations of ElectronicStoreA and ElectronicStoreB.

Person(*#Andy*)∧*isLocated*(*#Andy*, *#AreaA*) → *hasGranulariaty*(*#Andy*, *#Floor*)

Suppose that Andy is on the *UndergroundFloor* (recommendation point 3, Fig. 2), still using the ElectronicProfile set. The mall's middleware will recommend some products of ElectronicStoreA, ElectronicStoreB, and ElectronicStoreC, taking into account that Andy visited ElectronicStoreA after receiving the second recommendation. Moreover, Andy will receive no recommendation of CinemaA, as CinemaA's service is not contained in ElectronicProfile. After (maybe) visiting some electronic stores, Andy decides to change his profile for another (Fig. 1a): *LeisureProfile*.

Leisure Profile

As before, the first rule consists of a *release* policy. This indicates that Andy wants to release his position to the services with *LeisureProfile* when he is at the mall. He will then receive general recommendations about movies in theaters.

Person(*#Andy*)∧*isLocated*(*#Andy*, *#Mall*)∧*Service*(*?Requester*)∧*hasProfile*(*?Requester*, *#LeisureProfile*)→*hasRelease*(*#Andy*, *?Requester*)

The next rule is a *revealing* policy, and indicates that Andy wants to reveal his pseudonym to *CinemaA* just when he is on the *FirstFloor* (recommendation point 4, Fig. 2). Andy will start receiving custom recommendations about movies when he is on the first floor, considering

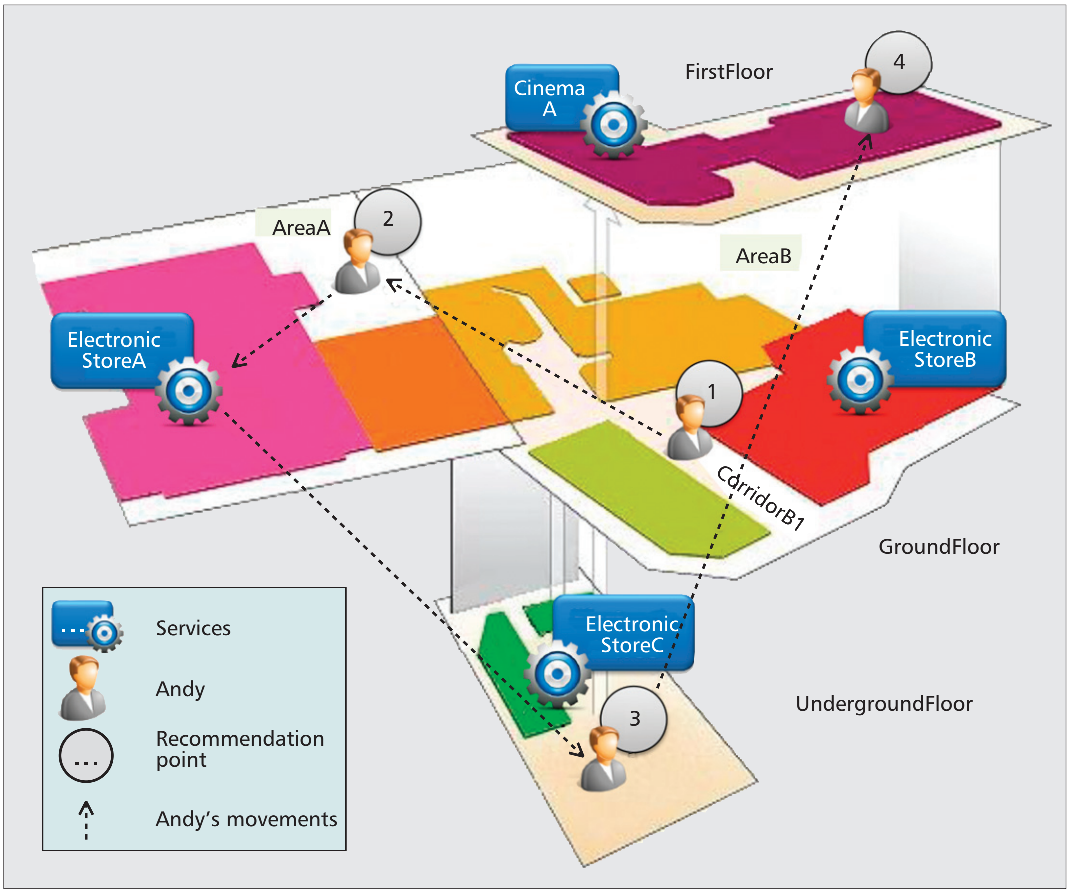


Figure 2. Overview of the scenario.

The storage and computing requirements are allocated at the IaaS layer of the MCC paradigm, in order to manage the large amount of data needed to shape the contextual information and the processing power to provide the services.

his records and preferences. This information is known by *CinemaA*'s service because Andy provided it during a previous registration process.

$Person(?Andy) \wedge isLocated(\#Andy, \#FirstFloor) \wedge \rightarrow hasRevealing(\#Andy, \#CinemaA)$

Finally, suppose that half an hour before the movie starts, Andy receives a notification of cancellation and recommendations about other interesting movies. This implies that when there are context changes, the mall's middleware will receive these changes and report such information to users who have the previous policies set, like Andy.

ARCHITECTURE

This section describes our architecture to deploy a privacy-preserving and context-aware system that provides services based on location. Figure 3 depicts the proposed architecture.

This architecture is oriented to the cloud in order to allow actors to use and manage the resources more efficiently. Context-aware services, *ElectronicStores* and *CinemaA* in the previous section, are located at the SaaS layer to provide recommendations according to their internal context. Instead, PRECISE (the mall's middleware) is a privacy-preserving middleware allocated at the PaaS layer to manage the global system context as well as the users' information.

The context awareness and space information are provided by independent middleware (the mall's infrastructure), which can belong to the cloud or not. The storage and computing requirements are allocated at the IaaS layer of the MCC paradigm in order to manage the large amount of data needed to shape the contextual information and the processing power to provide the services.

ACTORS

The PRECISE administrator (the mall's administrator in the previous section) manages the information related to users' locations and registration of the context-aware services. This actor also indicates to the *Communication manager* the different middleware required to receive location and context-aware information. On the other hand, the Service administrators (the administrators of *ElectronicStores* and *CinemaA*) are in charge of managing the context-aware information of their services, so each service has its own administrator.

The last type of actor in our solution is the user (Andy). Users are people who use PRECISE to obtain recommendations about the context in which they are located. They define their policies to manage their privacy directly in PRECISE, without having to rely on context-aware services.

COMPONENTS

The layers composing our system architecture provide the necessary resources so that the actors can use and manage the system. The *ElectronicStores* and *CinemaA* of the previous section are privacy-preserving services because they do not know Andy's identity, location, or personal

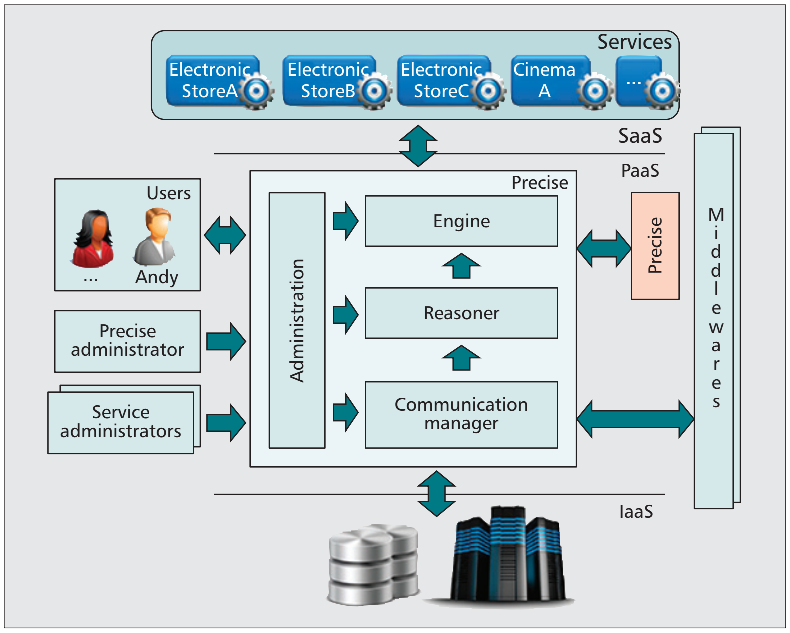


Figure 3. The PRECISE architecture.

information. As Andy is registered with *CinemaA* and defined a revealing policy for such a service, PRECISE will only reveal to *CinemaA* a pseudonym for him. Using that pseudonym, *CinemaA* will access Andy's information (provided by Andy during the registration process) and will be able to provide him with more personalized context-aware recommendations. Instead, the *ElectronicStores* will only provide Andy general context-aware recommendations.

As shown, PRECISE is a middleware that manages the context and preserves the users' privacy, without having to undergo a registration process. This implies that Andy, for example, will have different pseudonyms in different sessions, so PRECISE will not be able to track him. Thus, Andy's behavior pattern will just contain information about the current session. In another case, when Andy is registered in PRECISE, it will be able to access Andy's pseudonym and link his behavior patterns of different sessions, thereby providing him better suited context-aware recommendations.

In order to manage the context, PRECISE uses ontologies to shape the contextual and user information, semantic rules to define the policies, and semantic reasoning to infer new knowledge. The complete definition of the PRECISE ontologies can be accessed and downloaded from [12].

To perform all tasks, PRECISE has different components. The *Engine* component is in charge of requesting recommendations to the context-aware services, and provides users with recommendations based on previous ones. The *Reasoner* component makes the decision whether or not to request recommendations to specific services. This component infers the decision making as input for the updated ontological model formed by the union of the context-aware information and the users' information, and the semantic rules defined by the users.

The *Communication manager* component is in charge of receiving the context-aware and space information from the middlewares. This provides independence for PRECISE with regard to the sources of information.

Finally, the *Administration* component is responsible for supporting administrative tasks of the actors, including policy management, registration of the context-aware services, and management of the *Communication manager* component.

Sequence Diagram

The interaction between the PRECISE components is formed by four main blocks of steps, which are represented on the right of Fig. 4. This figure shows the sequence diagram, remarked on below, when Andy is at *AreaA* (recommendation point 2, Fig. 2). These blocks, from the perspective of PRECISE, are:

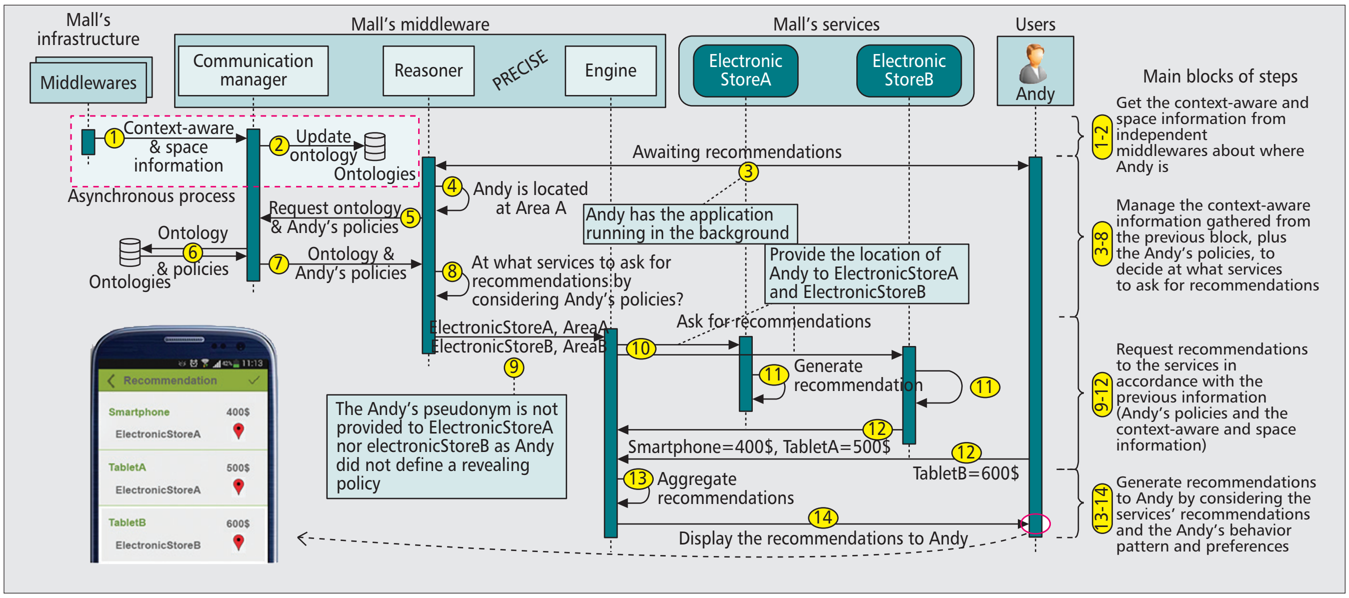
- Obtain the contextual and space information from independent middleware solutions and store it.
- Manage the context-aware information gathered earlier, and use Andy's location and his policies to decide with which services to ask for recommendations in a privacy preserving fashion.
- Ask for recommendations to the corresponding services and receive their products and offers.
- Provide Andy with recommendations by considering the services' recommendations, Andy's preferences, and his patterns of behavior.

Discussion

We present here a thorough comparison between our solution and the main related work introduced in the first section, which is shown in Table 1. First, LOC8 provides users with space and context information, although it does not allow users to define rules to dynamically manage their information or protect their privacy. In this sense, PRECISE lets users manage their information and privacy preferences by using rules to protect their identities, personal information, and locations against specific context-aware services or users.

SOCAM uses rules for inferring specific context information by considering other contexts and facts. CoCA is another solution that uses rules to dynamically manage context information. It infers users' locations taking into account the context and space information. Instead, PRECISE makes use of reasoners to infer new knowledge considering the context and space information. In spite of SOCAM's and CoCA's use rules, they do not allow users to define policies to manage their privacy preferences.

CoBrA allows users to protect their privacy by using policies, indicating the personal information that they want to reveal to other users. Instead, PRECISE is a privacy-preserving solution that, by default, protects users' identities, not revealing personal information to context-aware services. PRECISE allows users to define policies to specific services and users, not just to users as CoBrA does. On the other hand, the policies defined in CoBrA take into account the users' location and the context in which they are located, whereas PRECISE allows users to define richer policies than CoBrA. Users of CoBrA cannot define policies to manage their



Mall's infrastructure
Middlewares
Mall's middleware
Communication manager
Reasoner
PRECISE
Engine
Mall's services
Electronic StoreA
Electronic StoreB
Users
Andy
Main blocks of steps
1 Context-aware & space information
2 Update ontology
Ontologies
Asynchronous process
Awaiting recommendations
3
4 Andy is located at Area A
Andy has the application running in the background
Request ontology & Andy's policies 5
Ontology & policies 6
Ontologies
7 Ontology & Andy's policies
8 At what services to ask for recommendations by considering Andy's policies?
Provide the location of Andy to ElectronicStoreA and ElectronicStoreB
ElectronicStoreA, AreaA
ElectronicStoreB, AreaB
9
Ask for recommendations
10
11 Generate recommendation
11
12
Smartphone=400$, TabletA=500$
12
TabletB=600$
13 Aggregate recommendations
14
Display the recommendations to Andy
The Andy's pseudonym is not provided to ElectronicStoreA nor electronicStoreB as Andy did not define a revealing policy
Recommendation
Smartphone 400$ ElectronicStoreA
TabletA 500$ ElectronicStoreA
TabletB 600$ ElectronicStoreB
1-2 Get the context-aware and space information from independent middlewares about where Andy is
3-8 Manage the context-aware information gathered from the previous block, plus the Andy's policies, to decide at what services to ask for recommendations
9-12 Request recommendations to the services in accordance with the previous information (Andy's policies and the context-aware and space information)
13-14 Generate recommendations to Andy by considering the services' recommendations and the Andy's behavior pattern and preferences

Figure 4. Sequence diagram showing the interactions between the PRECISE components when Andy is located at AreaA.

location privacy, which is considered an important requirement in context-aware systems.

CoPS addresses the inability of CoBrA to manage users' location privacy, although CoPS does not consider privacy preferences about personal information. Using policies, CoPS allows users to decide to whom and at which precision they want to share their location and context information with other users. Instead, PRECISE's users can define polices to specific users, services, or a group of them depending on their location, profiles, context, and time. Therefore, PRECISE covers the users' location privacy of CoPS through hiding and granularity policies. Furthermore, PRECISE allows users to generate fictitious positions for specific users and manage the level of closeness at which they want to be located.

PPCS takes into account the shortcomings of the two previous systems. Specifically, PPCS protects users' information and locations by allowing users to decide the granularity at which they want to share their information, to whom, and where; and the time during which they want to reveal information. However, PPCS is not able to generate fictitious positions or establish the level of closeness at which users want to reveal their information.

Finally, SeCoMan addresses the drawbacks of PPCS, allowing users to define cloaking and close-

	LOC8	SOCAM	CoCA	CoBrA	CoPS	PPCS	SeCoMan	PRECISE
Context awareness	✓	✓	✓	✓	✓	✓	✓	✓
Cloud paradigm								✓
Rules		✓	✓	✓	✓	✓	✓	✓
User privacy				✓	✓	✓	✓	✓
Service privacy								✓
Anonymity policies				✓		✓		✓
Release policies								✓
Revealing policies								✓
Hiding policies					✓	✓	✓	✓
Cloaking policies							✓	✓
Granularity policies					✓	✓	✓	✓
Closeness policies							✓	✓

Table 1. Comparison of context-aware systems.

	LOC8	SOCAM	CoCA	CoBrA	CoPS	PPCS	SeCoMan	PRECISE
Context modeling	Ontologies OWL	Ontologies OWL	Ontologies RDF/OWL	Ontologies OWL	Database	Ontologies OWL-DL N3	Ontologies OWL 2	Ontologies OWL 2
Policies modeling	Not supported	Jena rules	Jena rules	Rei policy	RBAC	Jena rules N3 rules	SWRL	SWRL
Reasoner	Not defined	Jena	Jena	Flora-2	Not supported	Jena	Jena/Pellet	Jena/Pellet

Table 2. Comparison between context-aware systems in terms of the technologies they are using.

ness policies, as well as to stay in certain locations depending on their privileges and manage the context-aware information through operational policies. Furthermore, SeCoMan's users can share their location to the right person at the right place and at the right time. In addition, PRECISE allows users to manage their privacy regarding services, although it is able to do it for users too as PRECISE is an extension of SeCoMan. This implies that their architectures, use cases, and the benefits obtained from both solutions are rather different.

Additionally, all previous solutions are not oriented to the cloud paradigm, as PRECISE is, so they cannot obtain its profits such as distributed processing and storage, scalability, load balancing, and monitoring.

Table 2 shows the technologies used by the previous systems. For shaping the context, OWL 2 is more expressive than the rest because it was designed as an ontological language, besides being an open standard language. Furthermore, we think that Pellet is the appropriate reasoner because it supports semantic rules expressed in the SWRL language. SWRL is widely used in semantic web, which includes a type of axiom of the form *if* … *then* …, called Horn clause logic. Finally, the Jena application programming interface is used in our solution to generate the ontological models considering the ontologies and the semantic rules.

Conclusion and Future Work

We have presented a solution offering context-aware recommendations that takes into account context information, and users' locations, privacy, and behavior patterns. Context-aware services are allocated at the SaaS layer of the MCC paradigm, providing users with recommendations about context-aware information. The central element of our system is a middleware allocated at the PaaS layer, called PRECISE, which manages the context, preserves the users' information, and is independent of the middleware providing the space and context information.

As the next steps of this research, we plan to deploy a federation of services where they can share some information according to users' privacy and preferences. It will allow PRECISE to offer better custom context-aware recommendations to users.

References

[1] G. Adomavicius and A. Tuzhilin, "Context-Aware Recommender Systems," *Recommender Sys. Handbook*, Sept. 2011, pp. 217–53.
[2] G. Stevenson *et al.*, "LOC8: A Location Model and eXtensible Framework for Programming with Location," *IEEE Pervasive Computing*, vol. 9, no. 1, Jan.–Mar. 2010, pp. 28–37.
[3] T. Gu *et al.*, "An Ontology-Based Context Model in Intelligent Environments," *Proc. Commun. Net. and Distrib. Sys. Modeling and Simulation Conf.*, Jan. 2004, pp. 270–75.
[4] D. Ejigu, M. Scuturici, and L. Brunie, "CoCA: A Collaborative Context-Aware Service Platform for Pervasive Computing," *Proc. 4th Int'l. Conf. Info. Tech.*, Apr. 2007, pp. 297–302.
[5] H. Chen, T. Finin, and A. Joshi, "An Ontology for Context-Aware Pervasive Computing Environments," *Knowl. Eng. Rev.*, vol. 18, no. 03, Sept. 2003, pp. 197–207.
[6] V. Sacramento, M. Endler, and F. N. Nascimento, "A Privacy Service for Context-Aware Mobile Computing," *Proc. 1st Int'l. Conf. Security and Privacy for Emerging Areas in Commun. Networks*, Sept. 2005, pp. 182–93.
[7] P. Jagtap *et al.*, "Preserving Privacy in Context-Aware Systems," *Proc. 5th IEEE Int'l. Conf. Semantic Computing*, Sept. 2011, pp. 149–53.
[8] H. T. Dinh *et al.*, "A Survey of Mobile Cloud Computing: Architecture, Applications, and Approaches," *Wireless Commun. and Mobile Computing*, vol. 13, no. 18, Dec. 2013, pp. 1587–1611.
[9] P. Falcarin *et al.*, "Context Data Management: An Architectural Framework for Context-Aware Services," *Service Oriented Computing Applications*, vol. 7, no. 2, June 2013, pp. 151–68.
[10] A. Huertas Celdrán *et al.*, "SeCoMan: A Semantic-Aware Policy Framework for Developing Privacy-Preserving and Context-Aware Smart Applications," *IEEE Sys. J.*, to appear.
[11] A. Pfitzmann and M. Köhntopp, "Anonymity, Unobservability, and Pseudonymity — A Proposal for Terminology," *Designing Privacy Enhancing Technology*, Feb. 2001, pp. 1–9.
[12] University of Murcia, "Complete Definition of the PRECISE Ontologies," http://reclamo.inf.um.es/precise.

Biographies

Alberto Huertas Celdrán is a research associate in the Department of Information and Communications Engineering at the University of Murcia, Spain. His scientific interests include security, semantic technology, and policy-based context-aware systems. He received an M.Sc. in computer science from the University of Murcia.

Manuel Gil Pérez is a research associate in the Department of Information and Communications Engineering at the University of Murcia. His scientific activity is mainly devoted to security infrastructures, trust management, and intrusion detection systems. He received an M.Sc. in computer science from the University of Murcia.

Félix J. García Clemente is an associate professor in the Department of Computer Engineering at the University of Murcia. His research interests include security and management of distributed communication networks. He received a Ph.D. in computer science from the University of Murcia.

Gregorio Martínez Pérez is an associate professor in the Department of Information and Communications Engineering at the University of Murcia, Spain. His research interests include security and management of distributed communication networks. He received a Ph.D. in computer science from the University of Murcia.