Sentry@Home - Leveraging the Smart Home for Privacy in Pervasive Computing

Susana Alcalde Bagüés
Siemens AG, Corporate Research and Technologies. Munich, Germany
Department of Electrical and Electronic Engineering
Public University of Navarra. Spain
susana.alcalde.ext@siemens.com

Andreas Zeidler
Siemens AG, Corporate Research and Technologies. Munich, Germany
a.zeidler@siemens.com

Fernandez Valdivielso
Department of Electrical and Electronic Engineering
Public University of Navarra. Spain
carlos.fernandez@unavarra.es

Ignacio R. Matias
Department of Electrical and Electronic Engineering
Public University of Navarra. Spain
natxo@unavarra.es

Abstract

This article introduces a privacy framework for Smart Homes, supporting individuals roaming freely in pervasive computing environments. Such environments typically are equipped with different kinds of sensors and tracking devices for context-aware service provisioning. While on the one hand, people want to take advantage of the comfort and added value of personalized context-aware services, privacy and traceability becomes a serious concern on the other hand. The question arises, how we can build up trust into inherently untrusted services in a potentially hostile environment? How can it be guaranteed that eventually all sensitive data is deleted or safely stored away? The Sentry@HOME concept, as part of our User-centric Privacy Framework, addresses these concerns. It is designed to become an integral part of the user's home environment; seamlessly embedded into the Smart Home software infrastructure.. The Smart Home itself then can be leveraged to act as a privacy proxy for a tracked individual. On behalf of the user it constitutes the central privacy enforcement point for all privacy-relevant accesses to private or sensitive data. We are confident that our contribution, the combination of Smart Homes and a privacy-aware infrastructure, substantially adds to the success of personalized pervasive computing systems.

1. Introduction

Embedding small computers into everyday objects is an essential ingredient for making Pervasive Computing a reality. Additionally, seamless interconnection of such smart things will create evolutionary environments - or smart spaces - characterized by a certain degree of ubiquitous intelligence perceived by the human user.

Numerous efforts are underway to apply the idea of Smart Spaces to the context of daily home-life [1]. Roughly from the year 2000 on, projects have been launched at universities as well as some companies to address this topic. Prominent examples include the Aware Home from Georgia Tech, Stanford's iRoom, Cisco's Internet Home, Microsoft's EasyLiving, Philips' Home of the Future, and UF's Assistive Smart House.

Figure1: The "T-Com Haus" in Berlin, Germany, in 2005. Siemens AG together with T-COM and additional partners set up a working smart home environment (from [3])

Siemens addressed this topic e.g. in the context of the Smart Home activities or the "T-COM Haus" project together with T-COM [2] [3].

From our point of view the vision of the Smart Home does not have to be limited to support individuals only within the boundaries of the domestic environment. It also may provide assistance for the inhabitants when they are outside, e.g. shopping or in the office. In our work we demonstrate how the available smart home infrastructure may be leveraged to tailor encountered third-party context-aware services to respect the user's privacy requirements. This is done based on stored preferences on the usage of contextual information. Using context information in service provisioning, such as identity, location, activity, habits, together with the applicable policies for handling such data, constitutes a new class of services that very well has the potential to generate an added value for many people.

Obviously, such support of people's daily routines by context-aware services has a serious impact on the requirements for privacy protection that should be addressed to preserve a person's privacy. In this article we detail how a framework for privacy protection, like our User-centric Privacy Framework (UCPF), deployed as part of the Smart Home infrastructure can be employed to monitor and control the disclosure of personal privacy-relevant information to inherently untrusted third parties, both inside and outside the home's boundaries. The UCPF provides users with the necessary tools to ensure that data is revealed only to the extent of their consent.

The remainder of this article is structured as follows: We motivate our approach by introducing simple but distinctive use cases in Section 2. Section 3 describes generic requirements on privacy that are common for residents of Smart Homes. The description of the main features of the User-centric Privacy Framework is given in Section 4. It is then followed by an extensive summary of the related work in the area of privacy enforcement in Section 5. Finally, Section 6 concludes this article and indicates the directions of future work.

2. Motivation

The goal of this section is to show the applicability of a privacy framework within a home environment. Therefore, three typical scenarios are introduced that make use of well-known context-aware services. Along these scenarios, we highlight the privacy implications involved and discuss the benefits of having a privacy enforcement framework deployed as part of the smart home infrastructure.

Scenario 1: Alice spontaneously decides to go jogging. But in the evening and still working, she is not in the mood of going alone. She uses her mobile device to check the information available about her neighborhood and discovers that Ana (a new neighbor next door she has never met before) has jogging as a hobby also and currently is at home. Alice is pleased because sharing common activities is a good "door opener" to introduce herself to her. Quickly, she sends her an invitation for going jogging together. Ana is also happy to accept the invitation and they go jogging together and later even arrange to go again soon.

Scenario 2: In Martha's family the youngest son Daniel is picked-up by the maid at the kindergarten every morning on her way to work. When both arrive home the Doorkeeper service [4] automatically recognizes Daniel and sends a message to Martha. She opens the door remotely after checking the picture sent by the Doorkeeper that avoids giving a key of the house to the maid. Later, during the day, Martha always can monitor the location and activity of her son from her mobile device and can be sure that her son is fine.

Scenario 3: Bob is sitting in the subway reading a personalized selection of news items uploaded by his home-based Follow-up news service. He decides to include topics from a new electronic magazine for a period of six months. Although he is not at home, he can establish the conditions under which the subscription is done. The service gets a profile with selected topics of interest as well as personal data, such as account number, and identity. The contract is valid for six months; afterwards the service is obliged to remove Bob's profile.

At the first glance, the three scenarios we introduced above seem to very appealing to the idea of living in houses that allow its residents actively the use of context-aware services. But clearly, unrestricted access to personal information is out of the question because it would have serious and unwanted side effects. For instance, in Scenario 1 without restricting the access to the location of Ana and her private profile, unsolicited advertising services, targeted at people who like jogging, surely would flood her inbox with offers for running equipment, in the best case. In Scenario 2 Daniel's mother has access to his location and activity, but what would happen if it is also reachable by people with bad intentions?

In general the knowledge about location and activity of residents can greatly improve service provisioning in many cases, for instance in child-care and elderly support. But the same information can have a fundamental impact on a person's privacy when disclosed to the wrong recipient.

Another dangerous side-effect of an unregulated disclosure of personal data is the possibility to extract a detailed user profile out of collected data. A user in a pervasive computing environment inherently always is leaving a trace of her activity. Using this data someone might compile a very detailed profile of a person by using correlation and data mining techniques offline. However, from the privacy enforcement point of view, the problem stems from the simple fact that between the user and the environment there is no legally binding relationship that forces it to delete data after use. One desirable solution therefore is a mechanism for automatically removing stored data once the information is not valid or needed anymore. As an example, in Scenario 3 Bob's profile has to be removed after six months when the subscription is invalidated. Obviously, this mechanism implies that the policy for handling stored data is known and, even more importantly, that the user and the environment are constantly checking the compliance to the policy's content.

All these unwanted side effects are examples that highlight the need for a sentry of personal information: an entity that takes care of a user's privacy requirements, enforces access control, protects data from falling into untrusted hands, and tracks the usage of information already disclosed in the past. Without it, concerns about privacy issues may very well prevent users from actually using personalized context-aware services inside and outside a smart home environment.

Unfortunately, there is no "automagical" solution that provides perfect privacy and complete comfort at the same time. People do want to share information with others in many situations in everyday life. But this also means that people have to reveal personal data. Our approach to balance this situation is to try to provide users with the needed tools to specify under which circumstances they are willing to share information in an easy and comfortable way. Afterwards, we use our privacy framework for enforcing the chosen privacy settings based on the smart home infrastructure as the central privacy proxy.

3. Sentry@HOME – The Privacy Proxy

The scenarios presented above show that the nature of interactions and relationships between real and virtual entities will change radically. The evolution of domestic environments towards resident-aware homes is enabling homes to adapt to the user's context, preferences, tasks, and needs. Additionally, it will provide a virtual open door for enhancing the interaction with the "outside", such as the neighborhood, relatives, friends, colleagues, or visitors. Ultimately, we expect to see the frictionless interlocking of "inside" and "outside" with respect of context-aware services, i.e. the concept of the Smart Home goes one step further by providing assistance for the residents when they are on the move. Smart applications then are seamlessly switching from an "inside" to an "outside" mode. For this scenario, we propose to leverage the existing smart home infrastructure by making it the main sink for privacy related information and let it control the access to it. However, having the smart home acting as a context broker for context-aware services is only feasible if privacy functionality is provided as integral part of the home infrastructure.

In this article we present a new infrastructure component for smart homes: A privacy proxy, named Sentry@HOME. Its main task and responsibility is to take care of privacy-related data when accessed from the outside. Based on a set of privacy policies defined by the user it controls the dissemination of data within the context-aware service interaction chain as shown in Figure 2.

3.1 A Sentry in the Interaction Chain

To illustrate the concept of Sentry@HOME we first consider a typical interaction chain between a user and some service (without Sentry@HOME component), like shown in Figure 2. The data flow along the interaction chain involves different autonomous entities like people, companies or organizations [5] who typically adopt one specific role at a time.

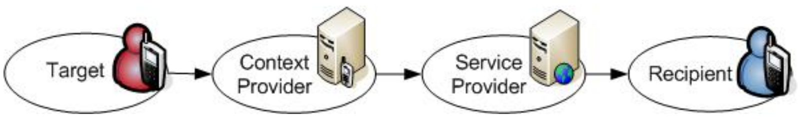

Figure 2. General Service Interaction Chain

We distinguish the following general roles: i) A “Target” is the tracked individual; ii) The “Service Provider” compiles context information for its users; iii) The “Recipient” (one or many) is the user of the service; iv) The “Context Provider” acts as an intermediate entity, responsible for collecting, caching and managing context information and for disseminating it, accordingly. As an example, Table 1 shows the relation between the roles described here and the entities involved in our scenarios in Section 2.

Table 1: Mapping Example Scenarios to Roles in our Model

Roles	**Scenario 1**	**Scenario 2**	**Scenario 3**
Target	Ana	Daniel	Bob
Service	Friend-Finder	Tracking Service	Follow-up News service
Context Provider	Location System	Location System	Profile Manager
Recipient	Alice	Martha	Bob

It is obvious that potentially malicious third parties can take advantage of this completely open interaction setting. Except for the target itself, the rest of the entities along the processing chain have to be classified as untrusted, since they are not under the direct control of the user. Even if a contractual relationship exists between the entities we consider this not to be enough for completely trusting such third-party services. Therefore, information disclosure has to be reduced to the least possible, potentially reducing the experienced quality of service for the user.

As part of our privacy framework we introduce a new important role: The “Sentry” (and later its implementation Sentry@HOME). It is designed to take over a new role in the interaction chain: the so-called user privacy enforcement point, see Figure 3, which controls all accesses to privacy-relevant data. A Sentry’s main purpose is to relief a user from manually-only consenting or dissenting to privacy statements as they commonly can be found in today’s Internet services, using e.g. P3P [6].

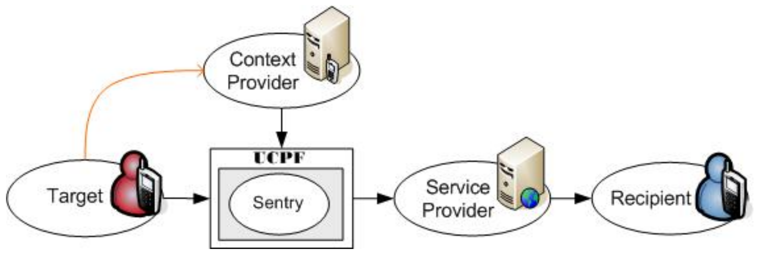

Figure 3. Service Interaction Chain with Sentry.

The main problem addressed by the Sentry concept is that today users usually have to enforce their privacy preferences manually, e.g. by reading a service's privacy statements or by manually selecting the data to be disclosed. Naturally this does not scale well. In the future the amount of information collected more or less automatically will increase by several magnitudes. Manual interaction easily would overwhelm users then. We even believe that the vision of context-aware infrastructures and services ultimately will fail without automatic mechanisms that avoid requesting a user's consent for each transaction. In this regard, our Sentry implementation provides means to automate this task, safely based on privacy preferences to be enforced by the Sentry (ref. Figure 3). It ensures that the constraints defined by the target are respected at all times. This is a main advantage especially if the interaction of different context sources is considered. Then a single *user privacy enforcement point* is beneficial as it acts as a "concentrator" for privacy-related issues. It guarantees that no information is disclosed outside the specified preferences.

3.2. Design requirements

In the design of our privacy framework we follow a user-centric approach to privacy. Several publications in the area of privacy protection, such as the well-known P3P [6] are concerned with how to define service-side privacy practices. The user there is limited to reviewing the offered service privacy criteria and to either accept or reject the terms of service. Our approach, like the Confab system [7], tries to give as much control as possible to the tracked individual. Additionally, in our framework we are explicitly not assuming that the sensing technology has to be attached to the user's computing device.

As the top requirement, a privacy framework has to provide explicit control over which data is disclosed when, how, to whom, and with which constraints. For designing the general architecture of our privacy framework, we partly followed "The Five Pitfalls for Designers" by Lederer et al. [8]. However, in our work we focus on providing flexibility, expressiveness, non traceability, interconnectivity, and user-friendly interfaces. Since we address users in their domestic environment, interfaces should be intuitive, easy to use and should not require excessive configuration to manage and modify privacy preferences. Basically, the user should find the same usability as with "regular" smart home services.

Another basic design requirement is flexibility and adaptability. Privacy preferences naturally change over time as the environment is not static and situations change. For instance in Scenario 2, assume Daniel has grown up and probably doesn't want his parents to track him without restrictions on when and how. Another example is in Scenario 3 when Bob might decide to subscribe to a different News service. We address flexibility as a key feature of our privacy framework and it is reflected in a policy-based scheme that allows for the control of behavior without modifying internal code or the implemented program logic. Policy matching between user's privacy preferences and that of the context-aware service is the most common mechanism used to enforce access control. Moreover, when the specification of privacy policies is embedded into a central privacy framework, users in general have more flexibility for expressing their preferences than using tools of a service provider. Also, they are relieved of the tiresome task of specifying access rules for each service separately.

The policy specification language should provide expressiveness to specify constraints on any piece of context. However, the use of traditional policy-based schemas [9] does not fulfill all of our requirements for expressiveness. In [10] we introduced and have shown the usefulness of Foreign Constraints and Transformations for allowing users to specify their privacy preferences. Without going into detail here (see Section 4) the use of Foreign Constraints and Transformation can greatly enhance the expressiveness of the policy specification.

Foreign Constraints and Transformations are also part of another requirement for implementing privacy protection apart from policy matching. Techniques of this category are usually summarized as "Noise" [11] and group together all technologies that try to hide or disguise a user's context or identity. Prominent examples are obfuscation, cloaking, or anonymization. Later in this article, we show how Transformations can be used to implement obfuscation.

Additionally, Lederer et al. state as an important design requirement: "Designs should not inhibit users from transferring established social practice to emerging technologies". Applied to the evolution of information technology it implies that the user needs mechanisms to exploit "plausible deniability" [7] i.e., lying about his context. Pervasive sensing environments by nature tend to violate this requirement by their ubiquitous nature. Therefore, we added a special functionality to our framework using Noise for injecting Ambiguity into any context information if needed or wanted.

One more requirement for our design guidelines is Interconnectivity. The system acts as context broker for the user when using context services. In order to respect the privacy preferences, all third party services are required to retrieve a target's context only from the appropriate Sentry. The system makes it also possible that two instances of a Sentry@home connect directly. This becomes mandatory especially for enabling the evaluation of Foreign Constraints.

4. User-centric Privacy Framework

The User-centric Privacy Framework (UCPF) presented here tries to fulfill the requirements stated before. It has been developed to help users to protect their privacy-related information. We believe that most users will be less concerned with the consequences of actively using context-aware services when they know that such data is managed and protected by their own privacy framework, without depending on context providers or other third parties.

We also have the vision of a community privacy proxy that protects the privacy of a set of "similar" individuals, i.e. with similar interests and privacy requirements, and is deployed in a communally available infrastructure. This is suitable particularly in the private home area, although being applicable to small companies or offices, as well.

A first prototype of the UCPF will be part of the residential gateway within the Siemens Smart Home Lab. The residential gateway is an entity that provides access to home-based services from the inside and outside of the house. The gateway has as major functional blocks: connectivity, QoS control, and security provisioning. The incorporation of the UCPF adds privacy as separate functionality and let users directly interact with their privacy preferences.t.

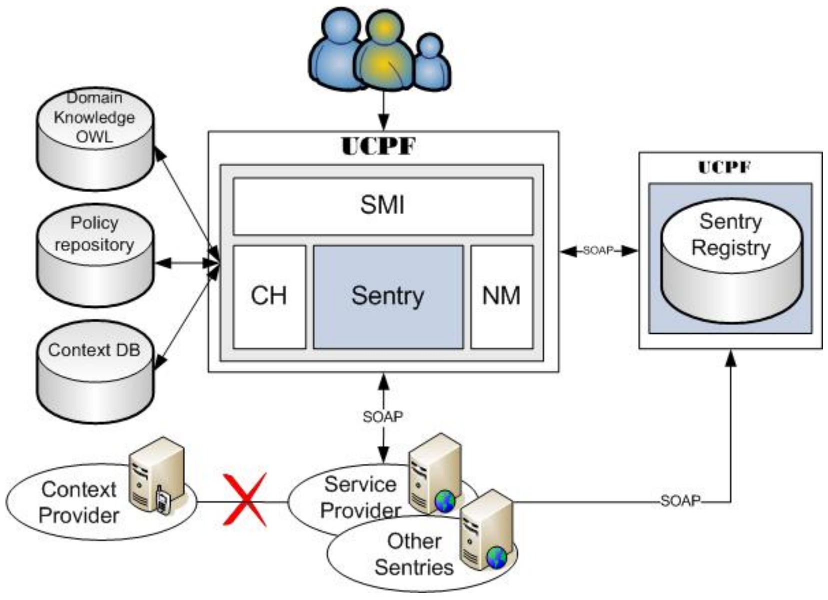

Figure 4. UCPF's Components Overview.

The UCPF consists of five functional elements, the Sentry Registry, the Sentry implementation, the Context Handler (CH), the Sentry Manager Interface (SMI), and the Noise Module (NM). In Figure 4 we give an overview over those components.

The Sentry Registry is the only component that is not integrated in the gateway. This entity is shared among sentries and located in the Internet offering verified access to the residential gateway running the framework. It allows a particular instance of the Sentry@HOME to connect to other instances of the same kind or with other context-aware services. The Sentry Registry is a directory where third-parties can search whether a particular person has made available some context, needed for a particular task. Additionally, it can be used for performing a more general search, e.g. filtering on persons living in a concrete zone and that have made their location information available. It tracks the availability of people's context and provides the pointer to the appropriate Sentry@HOME service instance.

Sentry is a major architectural building element. A Sentry instance, as it was mentioned before, manages the context disclosure of a target to third parties. To do so, it is build around the core elements of a policy distribution architecture as defined in RFC2753 [12] acting as policy decision point (PDP), and policy enforcement point (PET). Furthermore, the Sentry component has the capability of being the policy obligation point to track the obligations agreed on by third parties [13]. It acts also as the system's policy engine and is used for reasoning on policies formulated in our SeT language (Section 4.2). Policies, meta-policies, and domain knowledge are used to find and enforce privacy decisions about access rights, transformations, and obligations, respectively.

The interface between Sentry and Service Registry is facilitated by the Context Handler (CH). It provides the means to publish the Sentry as a service offering a target's context information in a Sentry registry. It supports also the identification of external sources of context e.g. for the evaluation of Foreign Constraints. We define Foreign Constraints as restrictions that a user might establish based on the context of other individuals, for instance in Scenario 1, Ana may want to formulate a constraint that limits when her location is available to persons that are in the same quarter. As long as Alice stays at her office in another quarter of the city, she is not allowed to see Ana's location; On the other hand, back home she gets eligible and can see Ana's location. To our knowledge, today's existing policy schemas do not allow for the definition of Foreign Constraints; however, they are essential for privacy in pervasive environments as they are rather natural for users to use. The integration of Foreign Constraints into the UCPF opens a wide field for further application scenarios and therefore is part of our ongoing research work.

Additionally, the CH acts as a mediator between Sentry and context providers. The exchange of information is done via XML machine readable language. We designed an "open interface" that is able to collect and understand data from different context management platforms. We assume that in the near future a standard context model will emerge, which could be shared by context providers as the base for a common context model for interoperability. The current trend is to use context ontologies for the exchange of data. In order to address our special needs we have developed four basic ontologies, based on the Web Ontology Language (OWL). We model location with the classes Coordinate, Address and PoI, activity with calendar objects, profiles by adapting a small part of the FOAF ontology to OWL, and we also created a simple ontology for describing time. The context model, together with the SenTry language (SeT), constitutes the domain knowledge of the UCPF.

The interaction of the user with the UCPF is possible through the Sentry Manager Interface (SMI). It is implemented as an API used in the development of user applications and interfaces. Through this API it is possible to generate, upgrade or delete privacy policies, receive information about the current applicable policies, or getting feedback on potential privacy risk. Other functionality of the SMI API is to facilitate the methods for the generation of user's profiles that are linked with the adequate policies. Profiles are used to store static data, like preferences and topics of interest.

The last component of the UCPF is the Noise Module (NM). It is a modular component that incorporates additional tools for the policy matching mechanism and can be extended gradually. In a first phase, Transformations and our White Lies noise generator module were implemented. Basically, we define Transformations as any process that the target may define over a piece of context information, e.g. to decrease the accuracy of the data. In our framework Transformations are embedded in the policy definition and are applied if the policy's constraints evaluate to true, which is always validated before transmitting any data. The White Lies module gives users the possibility to "disconnect" logically in a pervasive environment. "Plausible deniability" obviously is hard to achieve in an environment saturated with sensors and other tracking devices. The only way to "switch-off", like switching-off ones cell phone, appears to be by lying about location and activity whenever it is accessed by the system. However, for this article we consider the implications of While Lies to be out-of-scope and it will be discussed elsewhere

4.1. Architecture sketch and implementation

Figure 5 shows a sketch of the UCPF architecture. The uppermost layer is the Privacy Layer. This layer is responsible for: i) enforcing a resident's privacy preferences, ii) being the context provisioning proxy, and iii) providing the API to interact with users. The Sentry, the Context Handler, the Noise Module and the Sentry Manager Interface are components located in the Privacy layer.

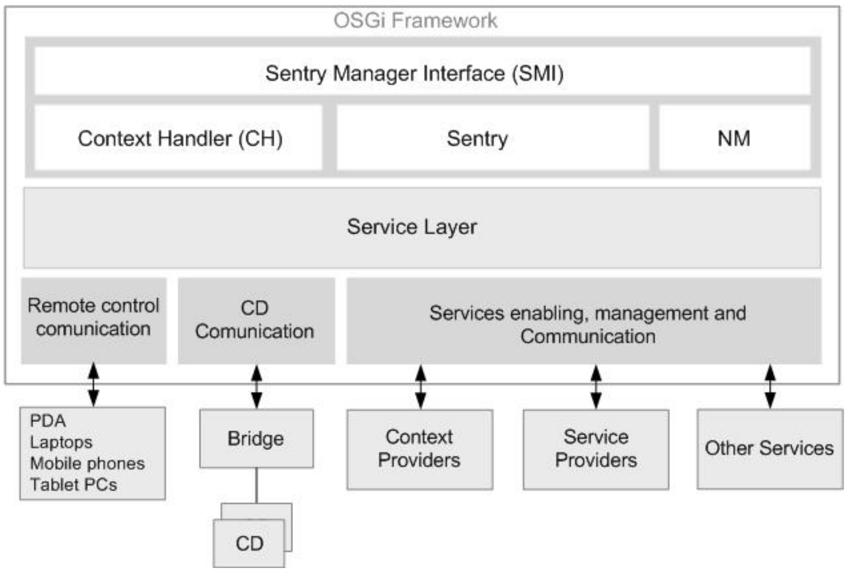

Figure 5. Architecture Based on the OSGi Framework.

In the first step of a two phase implementation of our UCPF architecture the Sentry@HOME prototype together with parts of the SMI API and the NM have been developed. Current activities include the implementation of the CH, the Sentry registry, and an intuitive graphical user interface for creating, maintaining, and monitoring privacy preferences.

Sentry@HOME has been developed in JAVA on top of the Java Expert System Shell, call Jess. The Sentry receives queries for user context from the Service layer. Such queries trigger the execution of the Jess rule engine, which then accesses the policy repository to retrieve all policies applicable for the current situation. They are evaluated based on the provided facts and enforced afterwards. Along with policies in the Jess language, the policy engine also accepts policies formulated in the Semantic Web Rule Language (SWRL) and are based on the existing SeT specification language.

The Service layer is the interface between services and the Privacy Layer. One of the features of he UCPF is that it must be able to interact with a wide variety of service types. Entities can be service providers, context providers, Sentry services, as well as other services not making use of any context information. Obviously, such an infrastructure requires the flexibility to add and remove services dynamically without interrupting the overall operation and the flow of data along the context processing chain; those functionalities are delegated to the underlying OSGi framework.

The home gateway itself is implemented on top of the Open Service Gateway Initiative (OSGi) framework. OSGi provides a rich environment for multiple Java-based components to work in a single Java Virtual Machine (JVM). OSGi also adds the ability to manage the life-cycle of the software components from anywhere in the network.

Both the Service layer and the Communication Layer are part of the home gateway's architecture.

The bottom layer is the Communication layer which provides wired and wireless access to the broadband network and the home network. It is subdivided into three groups: the Remote Control Communication, the Service Management and Communication and the Controlled Device Communication. All-IP communication is assumed among the home gateway and all devices and systems connected, such as controlled devices (CD), remote controls (inside and outside the house) and with the servers which store domain-based knowledge, policies and historic context data

4.2 The Sentry Language (SeT)

Besides the set of ontologies created to model context data (location, activity, time, and profile) in OWL, we have been working on the definition of a policy language which specifically encourages end-users to take advance of the expressiveness and flexibility of the UCPF. So far, there is no other language that allows the specification of Foreign Constraints and Transformations while using ontologies to model context.

The Sentry specification language (SeT) is composed of a global ontology with its own unique XML namespace. The SeT ontology describes the classes and properties associated with the policy domain in OWL-DL. Figure 6 shows the most important classes and properties: Policy, Rule, Entity, Resource, and Action. Conditions within our specification of SeT are expressed as rule statements in the Semantic Web Rule Language (SWRL), which combines OWL-DL and RuleML dialects.

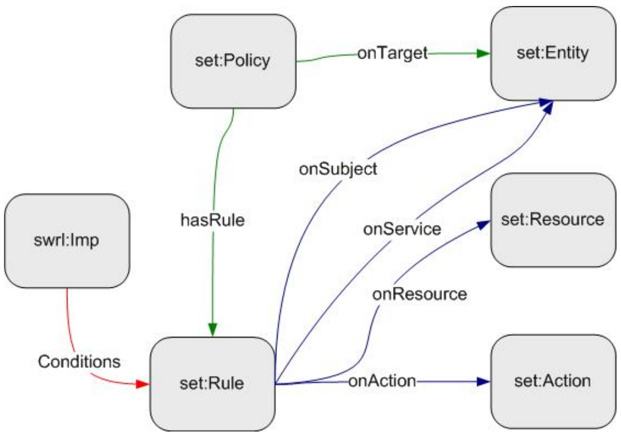

Figure 6. SeT main Classes and Properties.

A policy in the UCPF is an XML document that groups together the privacy preferences of an individual (identified with the onTaget (property), which are collected on a set of rules (with the property hasRule). The target's privacy preferences are modeled by using one out of four rule categories for each rule, namely the positive authorization rule (PAR), the negative authorization rule (NAR), the positive transformation rules (PTR) and the positive obligation rules (POR). Each rule explicitly delimits the entities, which can access private data (target's resource) with the properties onSubject, onService, onResource and onAction. Figure 7 shows an example of a simple policy defined with the SeT language. The policy has only a PAR rule that limits the release of Ana's location (coordinates) to the Friend-finder Service and only when Alice is the recipient. As an example of how to define conditions that may include foreign constraints within SeT, see the next Figure 8.

It details the SWRL statements needed to constraint the disclosure of Ana's coordinates on the rule "AnaPAR_015": only when both are in the same city and only when Ana is in "free time" situation. The first condition on the SWRL rule is SeT:MatchedRequest(AnaPAR_015, true), this property is true when all the values defined in the rule, Figure 7, are matching a service request.

```
<set:Policy rdf:ID="AnaPolicy">

  <set:hasTarget rdf:resource="#Ana"/>

   <set:hasRule>
    <set:PAR rdf:ID="AnaPAR_015">
         <set:onSubject rdf:resource="#Alice"/>
         <set:onService rdf:resource="#Friend-finder"/>
         <set:onResource rdf:resource="#Ana.Coordinates"/>
         <set:onAction rdf:resource="#Disclosure"/>
         <set:hasEvaluation rdf:resource="#Grant"/>
    </set:PAR>
   </set:hasRule>

</set:Policy>
```

Figure 7. SeT Policy example

```
SeT:MatchedRequest(AnaPAR_015, true) ∧
SeT:ServiceRequest(?varRequest) ∧
SeT:RequestedTarget(?varRequest, Ana) ∧
SeT:RequestingSubject(?varRequest, Alice) ∧
SeT:inCity(Ana.location, ?varCityTarget) ∧
SeT:inCity(Alice.location, ?varCitySubject) ∧
swrlb:stringEqualIgnoreCase(?varCityTarget, ?varCitySubject) ∧
SeT:Situation(Ana.situation) ∧
SeT:hasReasonedSituation(Ana.situation, "Free_time") ∧
  →
SeT:hasEvaluation(?varRequest, "Grant")
```

Figure 8. SeT Constraint example

Nevertheless before a Sentry can evaluate a rule the OWL instances and SWRL statements have to be translated into Jess rules with the Java API SWRL factory.

The Sentry component is by default configured not to disclose any data from any individual. At least one authorization rule must exist that allows the transmission of a specific piece of information and which is evaluated positively and allows this action.

The action requested identified in the rule with onAction is executed if and only if all facts that are part of the condition are true. In case of the NAR the action would lead to explicitly reject the query from the service. However, for the PAR, PTR, and POR rules, when an action is executed, the requested context is transmitted, although with different nuances. When a PAR is enforced, the requested data is directly transmitted, whereas a PTR involves a previous step that applies the adequate Transformation first, selected with the hasTransformation property, to modify the data before the action is executed. And, finally, in the case of a POR if the data is transmitted then the entity that receives it, accepts an obligation with the Sentry at the same time, e.g. to remove the received subscription data after six months (cf. Scenario 3).

The Sentry monitors the state of any obligation accepted by third parties. The state of an obligation is classified, following the proposal of **Error! Reference source not found**. as: unfulfilled, fulfilled and violated. If an obligation is in state unfulfilled it means that the entity, which still holds an obligation with the Sentry, has not performed the contracted action. The Sentry is usually triggered by the hasTimeLimit property associated with a POR rule. Another property that is checked is hasEndCondition. It indicates the moment in which the obligation must be fulfilled. If it evaluates to true the obligation is changing state from unfulfilled to violated. Sanctions describe the penalties of deviating from the policy. These sanctions usually represent an action that may be performed, e.g. to remove the authorization to get any new data from the Sentry.

SeT currently does not support delegation policies and does not have elaborated tools for policy analysis. At the moment we are more focusing on providing flexibility and expressiveness.

5. Related work

Previous privacy-related work in the field of pervasive computing roughly can be divided into two groups. The first group is focusing on policy matching techniques and the second group on techniques to guarantee that only the minimum amount of information is disclosed for satisfying a service's need to fulfill a user's request. Anonymization of users, obfuscation of the revealed context, or even dissemination of intentionally false data is summarized in [11] and generally is called Noise in the literature.

Policy matching and Noise injection commonly cannot be found integrated in the same framework. There are exceptions, though, e.g. in the Internet-Draft [16],released by the Internet Engineering Task Force Working Group on Geoprivacy (GEOPRIV) an authorization policy language for controlling access to location information is detailed. The current draft supports policy matching by introducing a rich set of rules that allow users to grant or deny access to their location. The schema also supports Noise in the form of obfuscation, e.g. defining the accuracy with which the location information is revealed. However, other forms of Noise currently are not considered.

According to [17] a privacy policy is an assertion that a certain amount of information can be accessed by a defined entity under a certain set of constraints. We classify privacy policies from the point of view of defining service privacy practices or user privacy preferences.

The probably best-known approach to privacy policies stems from the World Wide Web Consortium (W3C), which standardizes the Platform for Privacy Preferences (P3P) [6] P3P enables web sites to express their privacy policies and to compare them with the user's privacy preferences, which in turn can be specified by using A P3P Preference Exchange Language (APPEL) [18]. The policies are transferred to the user's browser and then matched to his personal preferences locally. However, as stated in [17], P3P has not been tailored to the specific requirements of pervasive applications. PawS, a privacy awareness system for ubiquitous computing [19], extends P3P to cover aspects of pervasive applications. In PawS, when a user enters an environment in which services are collecting data, a privacy beacon announces privacy policies of each service. A user's privacy proxy then compares these policies against the user's own privacy preferences. If the policies match, services are allowed to collect information and users can utilize the services in return. If the policies do not match, the system notifies the user, who then may choose not to use the service in question or, in some cases, simply physically can leave the area in which the collection of information occurs. Both define privacy practices for services, which are not within the scope of this work.

While APPEL [18] provides a good starting point for expressing user privacy preferences, it cannot support the richness of expressions needed for the evaluation of user criteria in real-world application domains. In [20] such requirements are implemented as system components called validators. The features of validators are described without defining a concrete implementation language and they need a centralized location provider to enforce them. Another approach is the Confab system [7] where a complex data structure is used to represent contextual information, the basic context atom called a context tuple that is equivalent to a web page. Information is captured, stored, and processed on the end-user's computer. This gives end-users high degree of control and choice over what personal information is disclosed. But it falls short in its flexibility of sharing context information in general. In [11] a model for user control of privacy is presented. There the Confab architecture was extended by a personal proxy that acts on behalf of the users and permits the integration of Noise. The work emphasizes the need of a privacy entity to manage different privacy techniques beside policy matching in order to cover a user’s need for privacy control.

Anonymization mechanisms technically hide the identity of a tracked user with respect to emitted context data so that she is not identifiable within a set of other tracked subjects, constituting the anonymity set [21].We can distinguish between techniques of data- and identifier abstraction. In a data abstraction, anonymity can be accomplished by cloaking data, e.g., by reducing temporal and/or spatial accuracy, so that data of different targets cannot be distinguished. In [22] cloaking is based on the formal model of k-anonymity [23]. For enforcing k-anonymity, a trusted context provider is needed, which has global knowledge about a group of targets. In identifier abstraction, pseudonyms are associated with context data. However, this approach suffers from the obvious problem that pseudonyms can be uncovered by statistical attacks. For this reason, in [24], pseudonyms are dynamically changed into mix zones to avoid linking different pseudonyms of a target together.

In [25], a formal model for obfuscating location information is given. In contrast to anonymization techniques that have the goal to hide a target’s identity, the the obfuscated identity in principle is known. Instead, position accuracy is reduced as far as application requirements can still be adhered to.

6. Conclusions and outlook

Pervasive computing may deeply affect many aspects of our daily lives. The envisioned environments typically are equipped with different kinds of sensors and tracking devices for context-aware service provisioning. Obviously, such support of people's daily routines by context-aware services has a serious impact on the requirements for privacy protection that must be addressed to preserve a person's privacy. We propose to leverage existing smart home infrastructures by making it the main storage for privacy related information of the residents and let it control the access to it. However, having the smart home acting as a context broker for context-aware services is only feasible if privacy functionality is provided as integral part of the home environment.

In this article we presented a new infrastructure component for smart homes: A privacy proxy, named Sentry@HOME, as part of our User-centric Privacy Framework (UCPF). Its main task and responsibility is to take care of privacy-related data when accessed from the outside. Based on a set of privacy policies defined by the user it controls and enforces privacy for individuals roaming freely in pervasive computing environments.

To our believes, the use of a trusted infrastructure in the smart home that is able to prevent privacy infringements by untrusted third-party services is the best way to foster the further spread of personalized context services especially outside the walls of the own home. We are confident that our approach of embedding privacy enforcement components into the existing smart home infrastructure greatly adds to realization of this believes. We showed how the smart home can act as a safe harbor for privacy sensitive data and our Sentry@HOME can act as a guardian sentry.

We are aware that the work still is not finished, though. Therefore, part of the ongoing and future work is to complete the implementation of our UCPF prototype and provide interconnectivity with other instances of Sentry@HOME. Furthermore, currently we are working on the development of the Sentry Registry that will offer authorized access to the residential gateway running the framework.

Another interesting and very active line of research is the application of the concept of Foreign Constraint. Foreign Constraints are an important extension to existing approached mainly because they are intuitively understood and instantaneously usable by human users. Also they greatly add to the expressiveness of any policy language for privacy protection.

Lastly, our plans include using the UCPF to demonstrate the concept of White Lies as important privacy mechanism. The White Lies module gives users the possibility to "disconnect" logically in a pervasive environment, which is considered a necessary requirement for socially accepted technology.

In summary, we believe that the approaches presented in this article add significantly to the field of privacy protection of individuals in pervasive computing environments especially in combination with and based on Smart Homes and a privacy-aware infrastructure.

7. References

[1] Jianhua Ma, et al. “A Walkthrough from Smart Spaces to Smart Hyperspaces towards a Smart World with Ubiquitous Intelligence”. In the IEEE Computer Society proceedings of the 11th International Conference on Parallel and Distributed Systems, pages 370-376, 2005.

[2] Pictures of the Future. “Communications, Always–on Society”, PoF Fall 2004, Siemens AG. Available at: http://www.siemens.com/pof

[3] Pictures of the Future. “T-Com House: At Home in the Future”, PoF Fall 2005, Siemens AG. Available at: http://www.siemens.com/pof

[4] ePerSpace-IST Integrated Project, . “D1.1 Service Scenarios and Specifications”. 2004.

[5] Axel Küpper. “Location-based Services Fundamentals and Operation”. John Wiley & Sons. August, 2005.

[6] L. Cranor, M. Langheinrich, M. Marchiori and J. Reagle. “The Platform for Privacy Preferences 1.0 P3P Specification”. W3C Recommendation. 2002.

[7] Jason I. Hong and James A. Landay. “An architecture for privacy-sensitive ubiquitous computing”. In the ACM proceedings of the 2nd international conference on Mobile systems, applications, and services, pages 177-189, 2004.

[8] S. Lederer, J. I. Hong, K. Dey and A. Landay. “Personal privacy through understanding and action: five pitfalls for designers”. Personal Ubiquitous Computing, volume 8, pages 440 – 454, 2004.

[9] M. Sloman. “Policy driven management for distributed systems”. Journal of Network and Systems Management, volume 2, pages 333-360, 1994.

[10] S. Alcalde-Bagüés, A. Zeidler, C. Fernandez-Valdivielso, and Ignacio R. Matias. “A User-centric Privacy Framework for Pervasive Environments”. In Proceedings of the OTM Workshops 2006, pages 1347-1356, LNCS 4278, Springer-Verlag, Heidelberg.

[11] B. A. Price, K. Adam, and B. Nuseibeh. “Keeping ubiquitous computing to yourself: a practical model for user control of privacy”. International Journal of Human-Computer Studies, volume 63, pages 228-253, 2005.

[12]R. Yavatkar, D. Pendarakis and R. Guerin. “RFC2753 – A Framework for Policy-based Admission Control”. 2000.

[13] N. Damianou, N. Dulay, E. Lupu and Morris Sloman. “Ponder: A Language for Specifying Security and Management Policies for Distributed Systems”. Imperial College Research Report DoC 2000/1.

[14] L. Kagal, T. Finin and A. Joshi. “A policy language for a pervasive computing environment”. Proceedings of the 4th International Workshop on Policies for Distributed Systems and Networks, 2003.

[15] Lalana Kagal. “A Policy-Based Approach to Governing Autonomous Behavior in Distributed Environments”. PhD Thesis, University of Maryland Baltimore County, 2004.

[16] H. Schulzrinne, H. Tschofenig, J. Morris, J. Cuellar and J. Polk. “Geolocation Policy: A Document Format for Expressing Privacy Preferences for Location Information”. IETF Internet Draft. January 2007.

Available at: http://www.ietf.org/internet-drafts/draft-ietf-geopriv-policy-10.txt

[17] Jorge R. Cuellar. “Location Information Privacy”. Geographic Location in the Internet, Kluwer Academic Publishers, pages 179-212, 2002.

[18] M. Langheinrich, L. Cranor and Massimo Marchiori. "APPEL: A P3P Preference Exchange Language". W3C Working Draft, 2002.

[19] Marc Langheinrich. “A Privacy Awareness System for Ubiquitous Computing Environments”. In the LNCS Proceedings of the 4th International Conference on Ubiquitous Computing, pages 237-245, 2002.

[20] G. Myles, A. Friday and Nigel Davies. “Preserving Privacy in Environments with Location-Based Applications”. IEEE Pervasive Computing, volume 2, pages 56-64, 2003.

[21] A. Pfitzmann and M. Köhntopp. “Anonymity, Unobservability, and Pseudonymity – A Proposal for Terminology”. Proceedings of the International Workshop on Design Issues in Anonymity and Unobservability, 2001.

[22] Marco Gruteser and Dirk Grunwald. “Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking”. Proceedings of the First International Conference on Mobile Systems, Applications, and Services, 2003.

[23] Latanya Sweeney. “k-Anonymity: a model for protecting privacy”. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, volume 10, pages 557-570, 2002.

[24] Alastair R. Beresford and Frank Stajano. "Location Privacy in Pervasive Computing". IEEE Pervasive Computing, volume 2, pages 46-55, 2003.

[25] Matt Duckham and Lars Kulik. " A Formal Model of Obfuscation and Negotiation for Location Privacy". Pervasive, pages 152-170, 2005.

Authors

Susana Alcalde Bagüés

Received her MS in Electrical and Electronic Engineering from the Public University of Navarra, Pamplona (Spain). She is currently working on her PhD thesis in the area of Privacy in Pervasive Computing at Siemens AG, Corporate Research and Technologies, Software & Engineering. Germany.

Andreas Zeidler

Received his PhD in Computer Science from Technische Universität Darmstadt and currently is working as a Software Architect and Researcher in the field of Pervasive Computing in the Software Architecture Department at Siemens AG, Corporate Research and Technologies, Software & Engineering, Germany.

Carlos Fernandez Valdivielso

Received his MS (1998) and PhD (2003) in Electrical and Electronic Engineering from the Public University of Navarra, Pamplona (Spain). He is currently an Associate Professor at the Electrical and Electronic Engineering of the Public University of Navarra (Spain) and also the Managing Director of Domotic Engineering. His research interest is in the areas of sensor networks and home automation

Ignacio R. Matías

Received his MS (1992) and PhD (1996) in Electrical and Electronic Engineering from the Polytechnic University of Madrid (Spain). He is a Professor in the Electrical and Electronic Engineering of the Public University of Navarra, Spain. He has coauthored more than 250 book chapters, journal and conference papers related to sensors and home automation.

INTENTIONAL BLANK