

Protecting the Privacy of Personal Smart Spaces

Kajetan Dolinar¹, Elizabeth Papadopoulou², Nicolas Liampotis³, Yussuf Abu-Shaaban² and Ioanna Roussaki³

1 INTRODUCTION

The purpose of privacy protection is dual, as it aims to both prevent, as well as cure privacy breaches. The starting point to develop such a privacy protection framework is the investigation and the definition of the privacy breach concept. A widely acceptable answer to this is provided by the privacy protection regulation, which however does not provide a precise enough definition of the privacy breach concept. The most common principles of privacy protection can be summarised in the following list [1]:

- **Principle of fair and lawful processing** (Article 6(1), letter a): “Any processing of personal data should be carried out in a fair and lawful way with respect to the data subjects.” (where a data subject is a legal or natural person to which the data refer)
- **Finality principle or Limitation principle** (Article 6(1), letter b): “Personal data must be collected for specified, explicit and legitimate purposes and may not be further processed in a way incompatible with those purposes.”
- **Data minimisation principle or Proportionality principle** (Article 6(1), letter c): “Processing of personal data should be limited to data that are adequate, relevant, and not excessive.”
- **Time minimization principle** (Article 6(1), letter e): “Data should be kept in a form which permits identification of data subjects for no longer than is necessary for the purposes for which the data were collected or for which they are further processed.”
- **Notification principle** (Articles 10, 11): “Data controller or his representative has to identify himself to the data subject and notify data subject about the personal data being processed, stored or further disclosed to any third party.”
- **Principle of data subject consent** (Article 2, letters a, h; Article 7, letter a; Article 8): “User consent is required for a legitimate data processing by any data controller. The user consent is defined as ‘any freely given specific and informed indication by which the data subject signifies his agreement to personal data relating to him being processed’.”
- **Principle of right to access personal data** (Article 12): “Data subject has right to access (and rectify) the data

collected about him, to get informed about the intended processing and the logic behind the intended processing of the data.”

- **Principle of right to object processing of personal data** (Article 14): “Data subject has right to object processing of personal data (subject to certain constraints: compare articles 7, letters e and f).”

Another approach to this problem is to start by studying the set of known contemporary problems regarding privacy protection. The problems in privacy protection have been thoroughly analysed by various research projects. For example, the following list of privacy protection problems has been identified by [2], which is the result of a quite exhaustive study:

- **Working from home**: monitoring of employees.
- **Digital rights management**: how much it interferes with the privacy of individuals?
- **ID theft and liability**: automated payments make it easy to spend money quickly, distance contracts do not offer the same guarantees, trust and confidence as in physical commerce.
- **Data laundering**: companies are paying a lot of money for personal and group profiles, while there are market actors in position to sell them.
- **Personal profiling**: a lot of people do not realise how much personal information they are constantly giving out when engaging in online transactions.
- **Inadequate profiling**: people are victims of an inadequate profiling based on false data or processing.
- **Disproportional request for personal information**: often data controllers require a volume information disproportionately large for the purposes of business.
- **Spyware and personal preferences**: spyware is a frequent way to steal data and intrude privacy.
- **Advertising and spam**: spam not only takes time and provokes irritation, but can also influence and infringe someone’s private world.

These are the real cases of privacy breaches and they cannot be mitigated unless a very sophisticated and efficient approach is taken, which should be based on state of the art technology and also consider the social models applicable. There is a sufficiently wide variety of technologies, tools and solutions to address those privacy breaches. Thus, a systemic protection should be possible.

In PERSIST we will not elaborate on each of these technologies, but will rather be interested in the overall framework of how those technologies and certain social models can fit together to prevent or cure those privacy breaches. Emphasis will be given on the abstract model and the required protocols and formal languages that need to be developed, in order to make the entire idea realizable, as well as on some specific solutions, as for example trust & reputation modelling.

¹ SETCCE (Security Technology Competence Centre), Slovenia. Email: kajetan.dolinar@setcce.si.

² Department of Computer Science, School of Mathematical and Computer Sciences, Heriot Watt University, UK. Email: {E.Papadopoulou, Y.Abu-Shaaban}@hw.ac.uk.

³ School of Electrical and Computer Engineering, National Technical University of Athens, Greece. Email: {nliam, nanario}@telecom.ntua.gr.

The rest of this paper is structured as follows. Section 2 elaborates on the overall architecture of the PERSIST privacy protection system. Subsequently, Section 3 discusses the concepts of privacy preferences and privacy policies, while it describes how these are exploited in order to conduct a privacy policy negotiation process. Then, Section 4 introduces the role of identities in PERSIST and presents the proactive identity selection process. Section 5 briefly exposes the trust management framework designed in PERSIST, investigating aspects regarding the trust management requirements, the trust representation, and the trust management architecture. Finally, in Section 6, conclusions are drawn over the entire privacy protection approach designed.

2 ARCHITECTURE

There are two parts of the PERSIST privacy protection system: the part concerned mostly about protection of privacy-sensitive information before the actual disclosure (referred to as *a-priori* protection) and the part intended to provide some protection or remedies in case of privacy breaches after the data have already been disclosed (referred to as *a-posteriori* protection).

A-priori protection is based entirely on technologies and among others includes the following techniques:

- define privacy preferences and induce privacy policy defined in a formal language suitable for machine processing;
- analyse trust in peers for different situations, or find out about reputation of peers;
- between two peers and based on their privacy policies negotiate over the privacy-sensitive information to be disclosed and the appropriate privacy protection;
- define appropriate digital identity for representation according to the agreement from privacy policy negotiation; and
- make up direct data protection: configure access control lists, credentials or other attributes, archive documents to preserve integrity and time of creation, encrypt or obfuscate data, buy insurance policy for protection against privacy breaches.

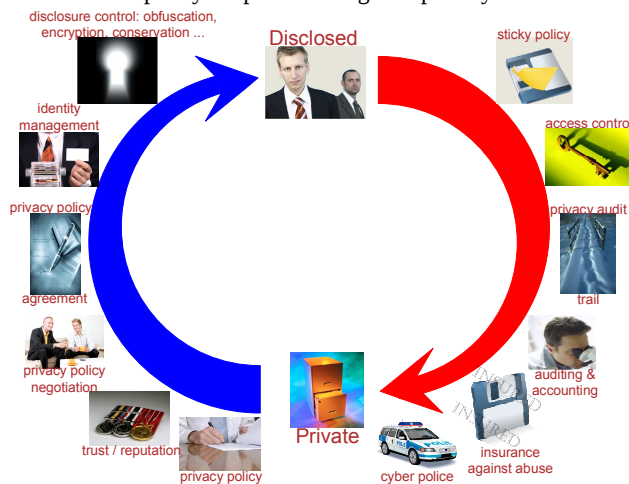


Figure 1. Privacy Protection Cycle

A-posteriori protection is a collection of techniques and social models which consist of the following:

- Attach a *sticky policy* to every data disclosed. The sticky policy is an excerpt from the agreement obtained after the privacy policy negotiation, which captures the essential privacy protection rules and parameters with respect to this particular data. The data with the sticky policy are cryptographically signed in order to preserve integrity, while data without any sticky policy are illegitimate.
- Entertain access control by credentials, and/or by access control lists, and/or by purpose (should match the purposes allowed in sticky policy).
- Use a privacy audit trail, i.e. record all actions on data (type of action, purpose, time, data and sticky policy) using a trusted hardware platform deployed on every node where processing of private identifying information goes on.
- In case of suspicion on a privacy breach, authorities can audit the privacy audit trail by investigating collective output of interesting trusted hardware platforms.
- If privacy breach has actually occurred, insurance compensates the curtailed person and penalizes the perpetrator.
- Finally, in case of severe privacy breaches, police and court take over.

This way the privacy protection is rounded into a cycle of systemic privacy protection, referred to as *privacy protection cycle*, which is illustrated in Figure 1. The *a-posteriori* part of the cycle is left mostly in a form of a concept and the majority of concrete design was made for the *a-priori* part.

We recognize several architectural components which bring about the functionalities required for a successful operation of the *a-priori* part of privacy protection cycle:

- **Policy Management:** provides interface for specifying privacy preferences, is able to produce privacy policies based on privacy preferences, and is able to negotiate agreements between two privacy policies.
- **Identity Management:** is able to produce digital identities that correspond to the extent of data and the degree of protection defined in the privacy policy negotiation agreement, it remembers the transactions where the digital identities are or were used, and it controls their lifecycle.
- **Trust Management:** uses appropriate model for collecting trust beliefs and for calculating reputation for each community member, is used in initial evaluation of peers during privacy policy negotiation, in making service recommendations, when deciding to join a group or when taking a group decision to accept a new member, etc.

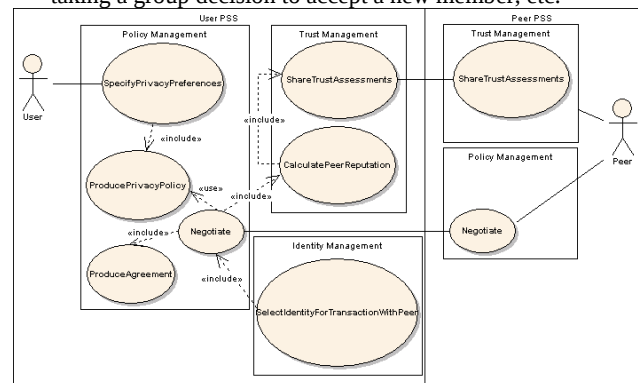


Figure 2. PERSIST privacy protection architecture

How these components interact and under which settings they are used can be seen in the diagram of Figure 2. The diagram is biased towards the User, however the situation is symmetric with respect to the Peer. It should be noticed that there are some less significant parts that are omitted in order to improve clarity. The `SelectIdentityForTransactionWithPeer` use case is invoked by a higher process of discovery of the Peer's resource of interest to the User and actuation of that resource. The `SpecifyPrivacyPreferences` use case is the only point of interaction for the User, as all the other use cases do not require the involvement of users.

3 PRIVACY POLICIES & PREFERENCES

This section describes the process of Privacy Policy Negotiation, along with the privacy policies and privacy preferences that are used to conduct the negotiation process.

3.1 Privacy Policy for Services

One general approach to protect the user privacy involves checking with a service what access to the personal user data it requires, before it is executed. Thus, when a user requests a service, the user's PSS communicates with the service and negotiates with it over the access it needs, subject to the user's requirements.

In order to accomplish this, the system uses the concept of privacy policies. These are complex data structures that define the user's requirements regarding the negotiation process. This process is referred to as *Privacy Policy Negotiation*. However, because of their complexity, they are difficult to create and maintain for the non-expert end-users. As a result, the concept of privacy preferences has been introduced. These are much simpler to understand and can be maintained via a user-friendly process. These are defined by the end-user and are used to generate the corresponding privacy policies automatically, which are far more complex.

3.2 Privacy Policy Negotiation

Privacy Policy Negotiation is the process that takes place when a user initiates the use of a service. Before the user actually starts using the service, the PSS of the user and of the service provider have to agree on the terms of usage. This requires a negotiation based on the service's privacy policies and those derived from the user's privacy preferences.

During a privacy policy negotiation, the two parties involved (the user's PSS and the Service Provider's PSS) exchange the privacy policies. Based on these, they negotiate the terms where the two policies do not match. The policies that dictate the actions to be taken during a negotiation are created on the fly after evaluating the corresponding privacy preferences. Figure 3 roughly describes the Privacy Policy Negotiation process. Assume that Abby's PSS advertises a Travel Agent Service. Jack is currently seeking to book his next holiday when his PSS receives Abby's Travel Agent Service Advertisement and requests to use it (Step 1 in the diagram). Abby's PSS responds to Jack's PSS with its Privacy Policy statement (Step 2 in the diagram). Upon receiving Abby's Travel Agent service Policy statement, Jack's Policy Negotiator creates a list of Jack's personal data, to which Abby's PSS is requesting access, and

passes this list to the Policy Manager along with Abby's PSS identifier, requesting that a privacy policy is generated for this purpose. The Policy Manager retrieves all the preferences related to the attributes that Abby's PSS is requesting access to and evaluates them. The evaluation process includes requesting a trust evaluation of Abby's PSS and its Travel Agent service. The outcome of the trust evaluation is fed into the preference evaluation, where it appears as a condition. After the evaluation process is complete, the outcomes of the preferences are merged into a single privacy policy document expressed in XACML (<http://xml.coverpages.org/xacml.html>) and returned to the Policy Negotiator (Step 3 in the diagram).

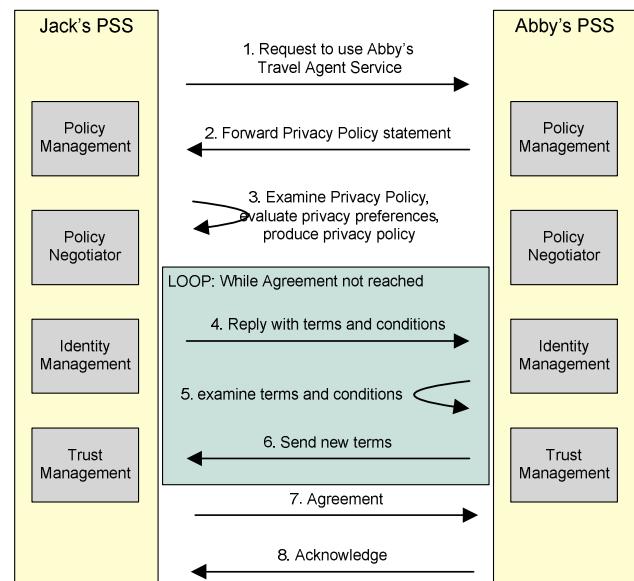


Figure 3. Privacy Policy Negotiation

Based on the customised privacy policy document, the Policy Negotiator edits Abby's Policy document under each statement indicating an agreement or a disagreement. Where a disagreement is stated, the reason is also included and terms and conditions Abby's PSS has to agree to so that access can be granted. The revised policy document is sent back to Abby's PSS, where the same process is repeated until an agreement is reached (Steps 4, 5 and 6 in the diagram). After the agreement is reached, a copy is kept by both parties for their records (Steps 7 and 8 in the diagram).

3.3 Privacy Preferences

The Privacy Preference lies in the heart of the privacy architecture of a PSS and expresses the privacy-related actions that need to be taken in certain situations. The privacy preference model consists of two main classes; the Privacy Preference Condition and the Privacy Preference Outcome. The Privacy Preference Condition describes the situation under which the Privacy Preference Outcome should be applied. Conditions are built up from elementary comparisons that check the values of attributes that describe the current context of the user at any particular point in time. Apart from the obvious and most commonly used attributes such as location, day of week and time, other parameters include features of the services that are being used at a given point in time by the user. Any parameter

that can be represented as a key-value pair can be used in a condition in the Privacy Preference Condition.

In the context of privacy, the trustworthiness of a service, with which a user is interacting, can be very important. The issue of trust in a PSS is further explored in Section 5.

There are two types of Privacy Preferences: the Privacy Policy Negotiation Preference (PPN preference) and the Identity Selection Preference (IS preference). The format of the Condition part of both types of preferences is the same. The difference between them lies in the format of the Outcome. The Outcome part of the PPN preference is used to produce the Privacy Policy that is used for conducting the PPN process described in the previous section. The Outcome of the IS preferences is used to select an identity from the pool of digital identities available to the user for a specific purpose. The IS preference is discussed in the next section. An introduction on the Identity Management (IDM) of PSSs is given in Section 4.

The PPN preference refers to one specific piece of personal information that has been requested. The Outcome of this type of preference suggests either to allow or to deny access to this piece of information. Each piece of information held in the user's PSS can have one or more Privacy Policy Preferences associated with it. Every time a piece of information is requested, the Policy Management component evaluates all the privacy policy preferences associated with it. It is possible that the outcomes of the privacy preferences conflict with each other. In this case, the system selects the "strictest" preference outcome.

4 IDENTITIES

4.1 Identity Lifecycle

Identities in PERSIST are used to group certain attributes of the user data and permit services to access them. The structure of the identity is never revealed to the services that use them. They are structures used internally in the Privacy Architecture of PSSs that maintain policies mapped to them and record historic transactions exploited for learning privacy policy preferences and identity selection preferences. The following diagram illustrates the Identity lifecycle in PERSIST.

An identity is created when the user requires a service execution, but the privacy components do not have a valid identity to map to the requirements of the services as specified in the privacy policy agreement. The Identity Management then creates an Identity to be used during the usage of this service. Once the Identity is created, the IDM attaches restrictions to the usage of this identity. The restrictions are extracted from the agreement reached at the end of the privacy policy negotiation process. At this point, an Identity reaches the "Valid" state. An Identity becomes "Active" when the service is given permission to start accessing its attributes and becomes "Inactive" when it is not used by any of the currently running services. The "Invalid" state applies to an Identity temporarily during the Identity Selection process, when the IDM tries to match an identity from the pool of identities to the requirements specified in the policy agreement and identifies as "Invalid" all identities that do not match these requirements. An Identity can take the state of "Unavailable" when certain circumstances such as context conditions render it unusable. The users themselves are also provided with the option to select a specific identity that should only be used in a specific place (e.g. home or office environment)

and nowhere else. Finally, an Identity is "Revoked" when the user deletes it from the system. In this case, the Identity is not deleted completely but exists in a "Revoked" state. This is due to the fact that the system needs to have a record of the attributes of that identity to map it to the historic transactions that are kept for processing.

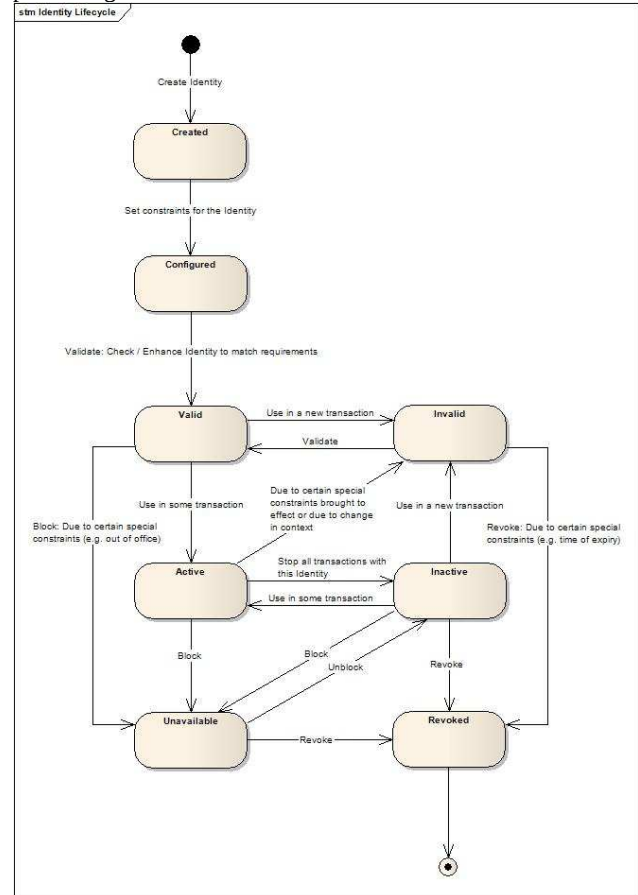


Figure 4. Identity Lifecycle in PERSIST

4.2 Identity Selection

The Identity Manager is the component that manages the Identities of the user. The management of the Identities includes maintaining both offline and run time information about the identities at all times. This includes the status of the identities as specified in the lifecycle diagram, the current transactions in which each of the identities is used, the attributes that are assigned to each identity and the restrictions that apply to each identity. It is also responsible for selecting the Identity to be used in each transaction based on the Identity Selection preferences. The Identity Selection process occurs right after the Privacy Policy Negotiation process. The Identity Selection process involves finding the list of valid identities from the pool of identities and evaluating the Identity Selection preferences to determine the exact identity to use. The process of Identity Selection is shown in Figure 5. The "Validation" process is the process that finds all the identities in the set of identities that match a set of requirements. These requirements are extracted from the Agreement that was reached during the policy negotiation. This agreement and information about the requestor

are provided to the Identity Manager by the Privacy Manager when it requests that an Identity is selected to be used in a transaction (Step 1 in the diagram). The Validation process returns a list of “Valid” identities. If the list of “Valid” identities returned is empty, the Identity Manager creates a new Identity that satisfies all the requirements specified in the agreement (Step 3b in the diagram). If one or more “Valid” identities are found, the Identity Manager requests the evaluation of Identity Selection preferences from the Policy Manager that determines which of these Identities should be used (Step 3a in the diagram). The Policy Manager evaluates the identity selection preferences that apply to all the identities in the list and determine which one is the best identity in this case (Step 4a in the diagram).

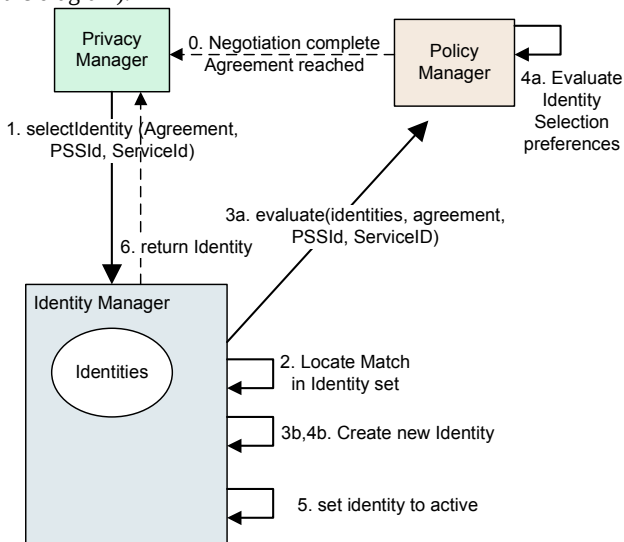


Figure 5. Identity Selection in PERSIST

It is also possible that the Policy Manager suggests that a new Identity needs to be created, due to the fact that one or more conditions do not allow any of the valid identities to be used. In this case, the Identity Manager creates a new Identity (Step 4b in the diagram). At this point, an Identity is selected to be used and the Identity Manager marks this Identity as “Active” (Step 5 in the diagram). The Identity is returned to the Privacy Manager and the service is allowed to execute on the PSS (Step 6 in the diagram).

4.3 Identity Selection Preferences

The Identity Selection Preferences are used to select the most appropriate Identity from the pool of existing Identities by evaluating a number of conditions that describe the environment of the user. As explained in section 3.3, the format of the Identity Selection preferences is almost the same as the Privacy Policy preferences. Both have the Preference Condition and the Preference Outcome parts. The Preference Condition part has exactly the same format. However, the Preference Outcome is quite different. The Outcome of an Identity Selection preference specifies in a given situation whether or not a specific identity can be used or not. For every Identity of the user, it is possible to have more than one preferences that dictates the conditions under which this identity should be used or not.

As described in section 4.2, when the IDM requests the evaluation of the Identity Selection preferences, it provides the list of “Valid” Identities in this particular instance, the requestor details and the agreement reached during the negotiation process. The Identity Selection Preference is associated with the requestor and the set of attributes the requestor needs to access. The Outcome suggests one or more Identities that would be suitable in any situation given the details of the requestor and the list of data that will be accessed.

5 TRUST MANAGEMENT

The PERSIST architecture specification was based on a scenario-driven approach, as outlined in [3]. The need for a trust management framework in this architecture became evident after analysing the identified use cases and the interaction patterns that appear in these scenarios. This section elaborates on the trust management framework that has been designed for the needs of Personal Smart Spaces.

5.1 Trust Management Requirements in PERSIST

This subsection briefly describes the requirements that should be addressed by the trust management framework in PERSIST.

Trust-based PSS negotiation: PSS negotiation is the process where two PSSs decide whether to interact with each other and is thus considered as the starting point for any collaboration between PSSs. This decision should be based on the trustworthiness of each PSS owner, i.e. the person or legal entity on whose behalf the smart space operates, as it determines the services -if any- to be made available by each party.

Trust-based privacy policy negotiation: Upon successful completion of the PSS negotiation, privacy policy negotiation takes place before allowing one party to use a service provided by the other, i.e. the visited PSS. More specifically, the privacy policy associated with the service of the visited PSS is first evaluated against the privacy preferences of the requestor and then the two parties negotiate on the personally identifying information that needs to be made available to the service in question. This process should not rely solely on the trustworthiness of the owner of the visited PSS, but the trustworthiness of the provided service itself should also be considered.

Trust-based service recommendations: The recommender system of a PSS can provide its owner with suggestions on a particular type of service based on his/her previous choices or those of other PSSs. The problem of trust is inherent in collecting and evaluating these suggestions. Therefore, the trust management framework should support the assignment of a confidence level to each service recommendation based on the trustworthiness of the party that provides the specific service.

Trust-based PSS grouping: One of the key features of PSSs is their ability to form groups for sharing context information, services and other resources. This raises substantial new privacy challenges as a member of a PSS group does not only share personal information with a service, but, at the same time, provides access to this information to other members of the group. From the point of view of a non-member, its decision whether to join a group should rely on the trustworthiness of group members, as well as, that of the services that are shared among PSS group members. On the other hand, the

trustworthiness of a PSS requesting to gain membership should also be considered prior to acceptance. It is therefore evident that the trust aspect is of key importance in PSS grouping.

Decentralised trust management: PSSs are capable of operating in both infrastructure and ad-hoc networks. This introduces additional requirements on the trust management framework of a PSS. More specifically, in order to support the ad-hoc environment, PSSs should neither rely on a centralised trust management system nor require maintaining global knowledge on the trustworthiness of every PSS. Thus, an efficient distributed trust management framework needs to be established.

Collaborative trust evaluation: The trust evaluation performed by a PSS regarding other PSSs or services should be based not only on the experience and evaluation of its own interactions, but also on those of other trustworthy PSSs. In order to support this, the trust management framework should employ mechanisms to monitor PSS interactions (direct experiences) and collect trust calculations (recommendations) from other PSSs. Effectively, trust will increase through successful interactions and degrade otherwise.

Automatic trust (re)assessment: Given the dynamic nature of trust, its assessment and reassessment should be performed in an automatic fashion allowing PSSs to constantly update their trust relationships in accordance with the behaviour of the trust objects. This would protect PSSs from parties that used to be trustworthy but have become less trustworthy or even malicious over time.

5.2 Trust representation

Trust is a very broad concept and the lack of consensus on a representation scheme is evident in the literature. In fact, the interpretation of trust has been compared by [4] to the story of the six blind men who perceived an elephant differently depending on the part of its body that they touched (e.g. as a rope because of its tail, etc.) [5]. Nevertheless, in this section, we focus on the PERSIST perspective on trust representation. The basic domains of trust relationships in personal smart spaces are defined, while the trust measurement scale is discussed. Finally, a high-level view of the trust management framework architecture is presented.

5.2.1 Trust domains

The trust management framework should allow an entity, i.e. a PSS, to express the trustworthiness of other entities within a domain. Based on the requirement analysis presented in section 5.1, five basic trust domains have been identified:

1. Trust domain D_1 expresses the trustworthiness $t(A,B)$ assigned to **PSS B** by **PSS A**.
2. Trust domain D_2 expresses the trustworthiness $t(A, S)$ assigned to **service S** by **PSS A**, where S is provided by **PSS B**.
3. Trust domain D_3 expresses the trustworthiness $t(A,G)$ assigned to **Group PSS G** by **PSS A**.
4. Trust domain D_4 expresses the trustworthiness $t(A,P)$ assigned to a **non-PSS party P** (e.g. provider, operator, user, developer, administrator) by **PSS A**.
5. Trust domain D_5 expresses the trustworthiness $t(A,R)$ assigned to **resource R** (e.g. service, network, content) by **PSS A**, where R is provided by a **non-PSS party P**.

It should be pointed out that the above domains express one-way trust relationships. We also assume that trust is not symmetric, thus, $t(A,B)$ is in principle not equal to $t(B,A)$. Moreover, supporting trust domains on a service level, allows for a more fine-grained level of PSS trustworthiness. So, for example, the weather forecast service may be assigned with a different trust value than the restaurant recommendation service that are both provided by the same PSS.

5.2.2 Trust scale

Trust, in its simplest form, can be binary, i.e. a given entity may be either fully trusted or not trusted at all [6]. However, most trust models use a single value over a specific range in order to represent trust. In fact, a variety of different trust value ranges has been proposed. In [7], the trust value is taken from $[0, 1]$, while in [8] a scale with discrete trust levels in the form $\{-n, \dots, -1, 0, +1, \dots, +n\}$ is considered.

In such trust representation schemes, the highest value usually denotes full trust, while the lowest one stands for full distrust. The problem in this approach is the representation of distrust as a result of negative experiences, as opposed to distrust based on lack of previous experience or knowledge of it.

The above issue can be addressed by maintaining two distinct order structures on trust values, i.e. a trust ordering and an information ordering, as proposed by [9]. The former represents the degree of trustworthiness, as previously described, while the latter expresses the degree of precision of trust information. More specifically, the first (lowest) element in the information ordering represents no knowledge and the last (greatest) element represents certainty. A similar solution is proposed by [10]. In their work, trust is represented as a 3-dimensional metric that includes values for belief (trust), disbelief (distrust), as well as, uncertainty. The problem of differentiating between distrust and simple uncertainty is further aggravated in ad-hoc environments where interacting with entities not known beforehand is very common. Thus, the uncertainty factor is also considered in the trust representation scheme of the PERSIST trust framework.

Furthermore, another aspect that is considered by this framework is the freshness/outdatedness of collected data that contribute to the trust evaluation. Thus, the trust model designed is capable of distinguishing between the same level of trust, evaluated based on the same type and number of interactions, when these interactions have taken place in different points in time (e.g. the last month or one year ago). This feature addressed the last of the presented requirements, thus protecting PSSs from parties that used to be trustworthy but have become less trustworthy over time or vice versa.

5.3 Trust Management Architecture

The components of the Trust Management framework are illustrated in Figure 6. The Trust Manager component controls various operations in the framework and provides an interface to other PERSIST components, such as the Policy Management and the Recommender System. Direct trust information is maintained in the Trust Repository. The automatic (re)assessment of direct trust between pairs of PSSs (or between a PSS and an external actor in general) is handled by the Direct Trust Evaluator, which uses the Risk Engine and the Learning Management for this purpose. Inferring trust when there is no direct trust relation between PSSs is supported by the Trust Inference component. Inferring the trust relationship between a group of PSSs and an

individual PSS is handled by the Group Trust Evaluator component.

The rest of this section is structured as follows. Initially, a description of the issues that should be considered during the process of direct trust assessment is provided, which is followed by a brief overview of the issues regarding the inference of trust in PERSIST.

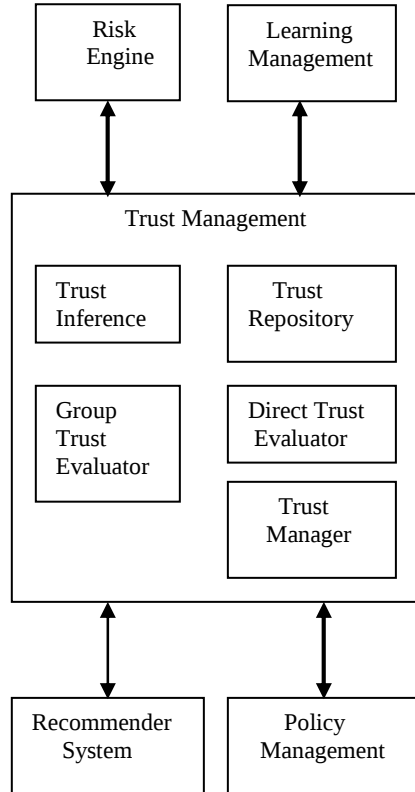


Figure 6. Trust Management Architecture

5.3.1 Direct trust assessment

A trust relation between two PSSs is initially created after their first interaction. As described in Section 5.1, the automatic reassessment of trust is a requirement of the Trust Management framework. When a PSS interacts with an entity, a number of factors could influence the (re)evaluation of the PSS's direct trust in that entity after the interaction has been completed, such as:

- Factors that could be learnt from the history of interaction including the number of previous interactions between the PSS and the entity, the frequency of interaction and interaction duration.
- Another factor that could influence the (re)evaluation of trust after interaction completion is the interaction risk. A high risk interaction with an entity that was completed successfully should lead to a high trust value allocated to that entity. Similarly, the trust in an entity should be significantly reduced if a high risk interaction fails.
- Trust ageing is another factor that could be considered when assessing direct trust between PERSIST entities. A trust value should be decreased if a long time has passed since the last interaction (the last trust update).

5.3.2 Trust inference

The use of trust inference is required to estimate PSS A's trust in PSS B, for a specific domain, when there is no direct trust relation from A to B in this domain. A number of trust inference algorithms have been devised based on aggregating the trust values along the trust path(s) between A and B. The algorithm proposed by [11] includes a set of trust propagation schemes based on the eigenvalue propagation and weighted linear combination. The EigenTrust algorithm presented in [12] calculates the reputation of peers in a peer-to-peer file sharing network. Each peer reputation is given by the local trust values assigned directly to it, weighted by the reputation of the assigning peers. An approach is proposed in [13] to determine trust by randomly selecting a path in a decision tree that is constructed based on trust information.

A number of issues could be tackled in devising an algorithm for the trust inference problem in PERSIST. This includes:

- **Trust grouping:** As described in section 5.1, grouping of PSSs is supported in PERSIST. For a PSS to join/interact with a PSS group, the trust relation should be inferred. Trust inference becomes a one-to-many problem where the collective trust of the PSS group members needs to be estimated.
- **Aggregating trust across domains:** A wide range of services with different domains could be supported in the PERSIST platform. Thus, a large set of domains will be associated with the trust relations between PSSs. The trust inference algorithm could consider aggregating trust values from related domains.

6 CONCLUSIONS

This paper elaborated on aspects of the privacy management framework designed by the PERSIST project Consortium to address the requirements of Personal Self Improving Smart Spaces. More specifically, the concept and the various types of privacy breach in computing/communication environments have been explored before the brief presentation of the overall architecture of the PERSIST privacy protection system. Additionally, the privacy policy negotiation process has been exposed, as well as the role of privacy preferences, privacy policies, and identities. Finally, the trust management framework designed is presented, where the trust management requirements, the trust representation, and the trust management architecture have been briefly described.

The design of the privacy protection framework in PERSIST has been finalised in March 2009. Up to this point, detailed interface specification has been produced, while all necessary UML diagrams have been generated (i.e. class, component, sequence and state diagrams). According to the PERSIST workplan, the prototype implementation of the privacy protection framework will be complete by May 2010. Thorough testing and evaluation will follow, in order to validate this proof-of-concept implementation. Of course, there are various other aspects that need to be addressed before this privacy protection framework can be widely used, such as protection from spyware, pushed advertising and spam, theft of identity, data laundering and abuse of personal information provided by individuals during online transactions. However, it is a firm belief of the authors that the research work described in this paper will make

a small step towards the introduction of privacy-aware personal smart spaces in the wide market.

ACKNOWLEDGEMENTS

The research leading to these results has received funding from the European Community's Seventh Framework Programme [FP7/2007-2013] under grant agreement n° 215098 of the Persist (*PERS*onal *Self-Improving SmarT* spaces) Collaborative Project.

REFERENCES

- [1] DIRECTIVE 95/46/EC OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 24 October 1995 on the protection of individuals with regards to the processing of personal data and on the free movement of such data, Official Journal L No. 281, (1995).
- [2] Y. Punie, S. Delaitre, I. Maghiros and D. Wright, (eds.) “SWAMI Deliverable D2: Dark scenarios in ambient intelligence - Highlighting risks and vulnerabilities”, Technical report, PERSIST consortium, (2005).
- [3] K. Frank et al., “PERSIST Deliverable D2.1: Scenario Description and Requirements Specification”, Technical report, PERSIST consortium, (2008).
- [4] R.J. Lewicki and B.B. Bunker, “Trust in relationships: A Model of trust development and decline”, in *Conflict, Cooperation and Justice*, pp. 133–173, Jossey-Bass Publishers, (1995).
- [5] Wikipedia. Blind men and an elephant.
http://en.wikipedia.org/wiki/Blind_men_and_an_elephant, (2008).
- [6] K. Aberer and Z. Despotovic, “Managing trust in a peer-2-peer information system”, in *Procs of the 10th ACM International Conference on Information and Knowledge Management (CIKM’01)*, pp. 310–317, New York, USA, (2001).
- [7] M. Richardson, R. Agrawal, and P. Domingos, “Trust management for the semantic web”, in *Procs of the 2nd International Semantic Web Conference*, pp. 351–368, (2003).
- [8] H. Prade, “A Qualitative Bipolar Argumentative View of Trust”, in *Procs of the 1st International Conference on Scalable Uncertainty Management (SUM’07)*, pp. 268–276, Berlin, Heidelberg, (2007).
- [9] V. Cahill et al., “Using Trust for Secure Collaboration in Uncertain Environments”, *IEEE Pervasive Computing*, Vol. 2, No. 3, pp. 52–61, (2003).
- [10] X. Li, M.R. Lyu, and J. Liu, “A Trust Model Based Routing Protocol for Secure Ad Hoc Networks”, in *Procs of IEEE Aerospace Conference*, pp. 1286–1295, Big Sky, Montana, USA, (2004).
- [11] R. Guha and R. Kumar, “Propagation of Trust and Distrust”, in *Procs of the Inter national World Wide Web Conference*, New York, USA, pp. 403–412, (2004).
- [12] S. Kamvar, M. Schlosser and H. Garcia-Molina, “The EigenTrust Algorithm for Reputation Management in P2P Networks”, in *Procs of the ACM World Wide Web Conference*, Budapest, Hungary, pp. 640–651, (2003).
- [13] U. Roth and V. Fusenig, “How Certain is Recommended Trust-Information”, in *Procs of the ACM World Wide Web conference*, Edinburgh, UK, (2006).