Final requirements and state-of-the-art for next generation policies

Deliverable: D5.1.1

Editors:	Carine Bournez (W3C) Gregory Neven (IBM)
Reviewers:	Markulf Kohlweiss (KUL) Martin Pekárek (TILT)

Identifier:	D5.1.1
Type:	Deliverable
Class:	Public
Date:	31 August 2009
Status:	Final version

Abstract

One of the core activities in the PrimeLife project is the design and implementation of a versatile privacy policy language. Policy languages are a key element in any privacy-aware information infrastructure. Machine-interpretable languages have the major advantage over natural languages that, if designed properly, they allow automated negotiation, reasoning, composition, and enforcement of policies. This document provides the first step in the development of such a language. It describes a number of relevant use case scenarios involving privacy policies and derives from these a list of requirements that should be met by the PrimeLife policy language. It also presents a state of the art in policies languages.

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n^{o} 216483 for the project PrimeLife.

Members of the PrimeLife Consortium

1.	IBM Research GmbH	IBM	Switzerland
2.	Unabhängiges Landeszentrum für Datenschutz	ULD	Germany
3.	Technische Universität Dresden	TUD	Germany
4.	Karlstads Universitet	KAU	Sweden
5.	Università degli Studi di Milano	UNIMI	Italy
6.	Johann Wolfgang Goethe - Universität Frankfurt am Main	GUF	Germany
7.	Stichting Katholieke Universiteit Brabant	TILT	Netherlands
8.	GEIE ERCIM	W3C	France
9.	Katholieke Universiteit Leuven	K.U.Leuven	Belgium
10.	Università degli Studi di Bergamo	UNIBG	Italy
11.	Giesecke & Devrient GmbH	GD	Germany
12.	Center for Usability Research & Engineering	CURE	Austria
13.	Europäisches Microsoft Innovations Center GmbH	EMIC	Germany
14.	SAP AG	SAP	Germany
15.	Brown University	UBR	USA

List of Contributors

Contributions from several PrimeLife partners are contained in this document. Individual participants in the Activity 5 are (at the time of writing):

Claudio Ardagna (UNIMI), Carine Bournez (W3C), Laurent Bussard (EMIC), Michele Bezzi (SAP), Jan Camenisch (IBM), Aleksandra Kuczerawy (KUL), Sebastian Meissner (ULD), Gregory Neven (IBM), Stefano Paraboschi (UNIBG), Eros Pedrini (UNIMI), Ulrich Pinsdorf (EMIC), Franz-Stefan Preiss (IBM), Slim Trabelsi (SAP), Christina Tziviskou (UNIBG), Pierangela Samarati (UNIMI), Jan Schallaboeck (ULD), Stuart Short (SAP), Dieter Sommer (IBM), Thomas Roessler (W3C), Sabrina de Capitani di Vimercati (UNIMI), Mario Verdicchio (UNIBG), Rigo Wenning (W3C)

The authors would like to acknowledge other PrimeLife partners and external groups for their contributions to the scenarios and use cases: TILT, TUD, the W3C Policy Languages Interest Group (PLING).

This deliverable was rendered from HTML pages using Prince XML from YesLogic Pty Ltd. YesLogic has donated a license of Prince XML to W3C.

Table Of Contents

Chapter 1

Introduction

1.1 About this document

The design and implementation of a versatile privacy policy language is one of the core activities in the PrimeLife project. Policy languages are a crucial tool in any privacy-aware information infrastructure. Machine-interpretable languages have the major advantage over natural languages that, if designed properly, they allow automated negotiation, reasoning, composition, and enforcement of policies. This document provides the first step in the development of such a language. This document is the result of a joint effort of all partners involved in Activity 5 of the PrimeLife project to collect use case scenarios and derive concrete requirements from them. It also presents a state of the art in policy languages, focussing on PRIME language, XACML, P3P and SecPAL.

1.2 Liaisons to other groups

1.2.1 W3C PLING: The Policy Languages Interest Group

While collecting requirements for the PrimeLife policy language, a connection was made to the W3C Policy Languages Interest Group. This group was started following a W3C Workshop on Languages for Privacy Policy Negotiation and Semantics-Driven Enforcement. This workshop was organised with the help of the PRIME project. Participants found out that there is already a great variety of very specific policy languages but that those do not work together. PLING is supposed -among other things- to serve as a platform for exchanges about the different policy languages. Even after the end of the PRIME project, some PRIME actors, together with stakeholders from industry and academia, continued working in PLING. It has started to listen to the industry and to collect use cases, information on existing policy languages and remarkable cases. At the W3C Technical Plenary meeting in Mandelieu in 2008, members of PLING had a joint meeting with participants of PrimeLife Activity 5 to think about further cooperation.

1.2.2 ISO/IEC JTC 1/SC 27/WG 5 on Identity Management and Privacy Technologies

As an official liaison partner to ISO/IEC JTC 1/SC 27/WG 5 "Identity Management and Privacy Technologies", PrimeLife is actively contributing to the development of standards in this Working Group. Hans Hedbom from Karlstad University was appointed Liaison Officer. Kai Rannenberg, the Convener of SC 27/WG 5, is the PrimeLife coordinator for partner Johann Wolfgang Goethe University in Frankfurt/Main. Jan Schallaböck from Unabhängiges Landeszentrum für Datenschutz, a PrimeLife partner, is the Secretary of SC 27/WG 5.

These close relations ensure that requirements stemming from the work of ISO/IEC JTC1/SC 27/WG 5 on Privacy and Identity Management are taken into account by PrimeLife, and also that research results of PrimeLife find their way into the work on the standards. Of special interest is the work on ISO/IEC 24760 Identity Management Framework, ISO/IEC 29100 Privacy Framework and ISO/IEC 29101 Privacy Reference Architecture.

1.3 Legal requirements

The processing (including collection, storage, retrieval, transferral, and other means of handling) of any data that can be linked to a person (personal data, for a more thorough definition see Article 29 WP, Op. 136 [Art29Op136]) by another entity (a data processor) has to be legitimate. If processing takes place without obeying legitimacy those subjected to the processing (data subjects) would lose trust in markets and tend to not give away their data when acting in these markets. Data protection thus is a market enabler, but above all it is recognized as a fundamental right, and is acknowledged by many constitutions in the European Union, as well as in the Charter of Fundamental Rights of the European Union (cf. Art. 8 thereof) and the European Convention on Human Rights (Art. 8 thereof).

On the operational level the Directive 95/46/EC [Dir95/46/EC] on the protection of personal data (Data Protection Directive), and the Directive 2002/58/EC [Dir2002/58/EC] on Privacy and Electronic Communications (E-Privacy Directive) provide the baseline for compliance. In many areas sector specific regulation and national implementation of directives need to be taken into account. Both directives as well as most of the other regulation follow a set of well established principles, with the principle of fair and lawful processing, the purpose limitation principle, data minimization, and the transparency principle at their core to name a few.

Fair and lawful processing

> Conceptually European law effectively prohibits any processing except where there is a legal basis (Art. 6 of 95/46/EC). This means that in a professional context handling data without a legal basis is illegal. Noncompliance can result in penalties and may even lead to data protection authorities shutting down IT systems (see § 38 of the German "Bundesdatenschutzgesetz"). A legal basis can be derived either from legal regulation directly, e.g., when the law prescribes the storage of specific data for law enforcement purposes. In many cases however the lawfulness can be achieved by receiving a consent, in an unambiguous form, from the person whose data is concerned, i.e. from the data subject (Art. 7(a) of 95/46/EC).

Purpose limitation

The processing of personal data - even if acquired lawfully (which should result in legitimacy) - is still a subject to further regulation. It regularly needs to follow, amongst others, the principle of purpose limitation (Art. 6.1(b) of 95/46/EC). Put simply, the purpose limitation principle states, that data may only be collected, stored, processed or transferred for those purposes to which the data subject has given consent, or for which the law allows. No further processing that would be incompatible with the original purpose is allowed.

Data minimization

This includes, that if no purpose is at hand, the data has to be deleted or not even collected in the first place (Art. 6.1(c) 95/46/EC). But data minimization can be understood even in a broader sense to construct systems in such a way, that processing of personal data can be avoided (so called data avoidance). While the former is a legal requirement, the latter is not mandatory by law in all cases, but only where such technology is available under reasonable conditions (cf. Recital 46 of 95/46/EC). However, it can be sensible to develop and use such approaches even in an enterprise context when there is no legal necessity, as it may lower compliance costs.

Transparency and subject access rights

The principles regarding the processing itself are adjunct by specific, enforceable rights for the data subject (e.g. Art. 12 and 14 of 95/46/EC). The conceptual idea behind these rights is that the data subject should be able to find out what others know about him or her. In case this knowledge is illegitimate, the data subject should be able to stop the respective data controller from using this knowledge, by blocking, correction or deletion of the personal data.

In information systems the protection of any data does not always prove to be easy. This is especially true for the protection of personal data. A core difficulty lies in the diversity of the processing steps that this data may undergo, while at the same time being subjected to the above principles. For compliance it has to be ensured that any algorithm or any service of an IT system that processes a specific set or piece of personal data is within the limits of the legal foundation (e.g. the consent) for the processing, and that it does not violate the purpose limitation principle. At the same time, it needs to be ensured that the data subject is able to find out what happened to his or her data, who accessed it, and what it has been or will be used for.

This deliverable on policy requirements collects a thorough set of requirements from a number of use cases to facilitate compliance and enhance privacy protection of the user.

1.4 Definitions of policies

We first define three types of policies that, in our view, are important parts of any privacy policy: data handling, access control, and trust policies. This by no means implies that we consider these to be separate, independent policies that together form the privacy policy. Rather, we see them as three minimal aspects that have to be covered by any policy language. There may be other aspects, and the three aspects mentioned here may not be orthogonal. In fact, the exact relationship between these three policy types will be an important research topic in WP5.2.

1.4.1 Data Handling policies

A *data handling policy* (DHP) is a set of rules stating how a piece of sensitive data should be treated. In the context of privacy, we are mostly interested in the case where that piece of data is personally identifiable information (PII). The data handling policy specifies, amongst other things, for what purposes the data can be used (e.g., research, marketing), to which third parties the data can be disclosed (e.g., all, nobody, only auditors), and the obligations on data management (e.g., how long the data can be stored). Obligations define actions that must be executed by the party in charge of enforcing a policy. Those actions are triggered by events such as time or handling collected data.

We distinguish between three different types of DHPs, namely *data handling preferences* on the data subject's side, and *data handling policies* and *sticky policies* on the data controller's side. In the *data handling preferences* associated to a piece of PII, the data subject specifies its requirements on how it expects her data to be treated by the data controller. The data controller, before receiving the PII, describes his intentions on how he will treat the PII in his *data handling policy*. When an agreement is reached, the agreed-upon policy that the data controller has to adhere to is referred to as the *sticky policy*.

1.4.2 Access control policies

An *access control policy* (ACP) protects access to an object by specifying which recipients should be granted which type of access to it. The object being protected can be a piece of data like a file, a database record, or a webpage, but it can also be a more abstract functionality like a service or a remote procedure call. The subject can be specified by means of a unique identifier (e.g. user name), by roles (e.g. administrator), by groups that he belongs to (e.g. helpdesk), or by other attributes (e.g. age, reputation, ...). A subject can be any type of entity that is capable of making a request; it could be a natural person but could also be a running process or a device, or a combination of those. The possible types of access (e.g. read, write) depend on the resource that is being protected. Finally, the decision to allow or deny access can be based on the subject's properties, the content of the resource, details of the access request (e.g. parameter values passed in a remote procedure call), and secondary information like the current time, processor load, etc.

1.4.3 Trust policies

The concept of *trust* is almost inherently vague due to its close association to the subjective decisions made by humans in real life. Even within the technical community, there seems to be quite some confusion about the definition of trust policies. Here we clarify what we, for the scope of this document, understand under trust in privacy policies.

In general, a trust rule is a rule expressing that some entity is trusted to perform some action if some condition holds. The action can be either to certify a specific piece of content (e.g. write about technology or issue a passport) or to adhere to an agreed-upon data handling policy. The condition can be any condition that has to be satisfied for the rule to apply. We leave open what kind of statements the condition can contain; conditions can be based on the credentials held by X, privacy labels, reputation, environmental conditions, etc.

The two actions that we have in mind when talking about trust are trust in an entity to certify a specific type of content (content trust) and trust in an entity to adhere to an agreed-upon data handling policy (data handling trust). The main difference between both types is the direction

of the information flow. For content trust, it is the trusting entity who receives information (possibly indirectly) from the trusted entity; for data handling trust, it is the trusted entity who receives information from the trusting entity. Content trust is about the correctness of received information; data handling trust is about how information is treated after it is transmitted.

Examples of content trust are governments who are trusted to certify the identity of their citizens, banks who are trusted to issue credit cards, medical doctors who are trusted to make a diagnosis for a patient, websites who are trusted to issue reputation values to users, or certification authorities that are trusted to bind cryptographic public keys to real-world persons. Content trust includes the type of trust that a relying party needs to have in the issuer of a credential for the credential to have any meaning. Content trust may or may not be delegatable; certificate chains for example are a typical implementation of trust delegation. A relying party always needs at least one or more roots of trust from which trust relationships can be bootstrapped. Technically, these could be a set of top-level certificates that are hardcoded into a browser, a trusted directory server, or a piece of trusted hardware.

Privacy seals are the most prominent example of data handling trust. For example, a customer may only be willing to give away her contact information to websites approved by the Better Business Bureau, because those are the only websites that she expects to treat her information correctly.

Chapter 2

Scenarios

Note: Several of these scenarios refer to the PrimeLife personas: Hannes, Frank, Mariangela, Inga, Eugene, Josha, Ines and Florence. You will find more information about them in D4.1.1 appendix B.

2.1 UC1. Wiki scenario

2.1.1 UC1.1. Register to the Wiki

Issue a credential for a nickname. The credential issuing is protected by a policy which grants access if the nickname is not already in use and if some administrator has approved the issuing (e.g. the existence of another credential).

One possible chain of activity:

1. Hannes clicks on the "Register new user" button
2. PRIME-AskSendData pops up asking for a nickname
3. After sending "hann" (stored as PII in the server), the server calls its policy evaluation for issuing credentials.
4. The policy evaluation finds a policy stating that a nickname has to be present, which is
    - unique among all nicknames in the PRIME PII database
    - unique among all nicknames in the external wiki database (using a custom condition function?) and
    - certified by an Identity Provider.
5. Since the nickname is not certified, deny is returned.
6. Hannes gets a page stating that the nickname has to be approved.
7. He contacts an Identity Provider to get his nickname signed.
8. After the administrator at the Identity Provider checks Hannes' data, he signs the nickname.
9. Hannes tries to fetch the login again and now the credential is presented.

2.1.2 UC1.2. Edit policy for number of sites

Create/edit a policy for existing or not-yet existing wiki pages. Access can be set for:

- all sites,
- all sites that are "owned" (=created) by some user (this is a list of sites that can be queried at runtime, but is unknown at deployment),
- one specific or a static list of specific sites (which could be implemented by inserting separate policies for all those sites).

One possible chain of activity:

1. The admin sets up all wiki sites to be readable by default.
2. Some actions as viewing all sites starting with "User:" are only readable by the user with the appropriate nickname (the word following "User:"). Note: This should not be done by using the current policy-tag "subject", but by the presence of a correct value for the category "nickname".
3. All sites created by a user are - by default - only editable by this user.
4. All these settings can be overridden by more specialised settings.

2.1.3 UC1.3. Create a new site with policy

Restrict the access to a wiki page to users belonging to a group. (Same problem as above: the list cannot be known at deployment and can change during runtime.) It may be possible to implement credentials (list members get credentials and the presence of a credential is necessary).

One possible chain of activity:

1. Mariangela creates a new site in the wiki. Directly above the "Submit" button, there is a selection combo field with the caption "Who else can access this site?" and the following values
    - nobody
    - everyone
    - users of group...
    - the following users...
2. When choosing either of the last two options, she can select a group or a specific list of users.
3. After submitting, the server creates a new policy in the PRIME server containing her selection for the created site.
4. She can also change this setting later by clicking "Edit Policies".
5. The policy gets evaluated when someone tries to access this specific site. All other policies specified by other sources are evaluated as well.
6. Mariangela chooses to restrict access to group 'Friends on MyPrime'.

It is left open how this use case would actually be implemented. It could either be implemented by having the policy engine query the actual member identities of the group from an external source (over RequestContext or somehow specify together with the policy evaluation request), or by means of a credential stating the membership to the "Friends on MyPrime" group in the policy evaluation.

2.1.4 UC1.4. Access Control based on reputation

Restrict the access to a wiki page to pseudonymous users with some reputation value, whose value is higher than a specific number. The number may be written in the policy or come from an external source (e.g. the reputation of the creator).

1. Josha wants to edit a wiki about recommendations of music bands. The site requires membership of an anonymous reputation system.
2. Josha can only edit sites, where he can prove a higher reputation than some values set.
3. Josha does not want to reveal his exact reputation value (to remain anonymous). Thus the created claim request by the policy engine asks for a proof of "higher than value X".

2.2 UC2. Blog scenario

The main policy here is a trust policy specifying who I trust to provide or certify a certain type of content. For this to work, we need at least a taxonomy of content types and fields of expertise. For example, if I trust someone to be knowledgeable about football, I don't necessarily trust his comments on basketball. Someone trusted in the whole field of sports however could comment on both.

The blog should also support (dis)approval comments or scores on blog entries, so that users can base their opinion about an entry on the opinions of other users that they trust. A user's policy can then for example state that only entries are trusted that received 5-star scores from at least 3 trusted users.

Additionally, blog entries should be accompanied with a DHP describing the allowed usage of the information (read, quote, modify,...).

2.2.1 UC2.1. Creating a blog

1. Inga logs in under a previously created pseudonym, thereby making her photoblog to be created linkable to the other blogs under her pseudonym (another user would create a new pseudonym).
2. Inga optionally adds pseudonyms of co-authors who should also be given write access to her photoblog.
3. She reveals to the server those of his attributes that he deems relevant for the credibility of the blog and is willing to reveal, e.g. his profession, diploma, group membership,... The Server may also require her to have some attributes in order to be allowed to start a blog on a certain topic. This information might be evaluated by the server administrator or it may also be revealed to other users.

2.2.2 UC2.2. Posting an entry

1. Mariangela logs into the server (possibly under a pseudonym) and requests write access to the blog she wants to post to. The server authenticates her and checks that she is registered as one of the authorized authors of this blog.
2. She writes her entry and authenticates it, e.g. by signing the content. She also attaches a data handling policy specifying for what purposes it can be used, which parts can be quoted, etc.

3. The server makes the new entry publicly available on the website, together with the origin authentication and DHP data.

2.2.3 UC2.3. Posting a blog comment

1. Inga logs into the server under an existing pseudonym that she previously created, allowing this comment to be linked to her previous comments under this pseudonym, or she could create a fresh pseudonym for this comment.
2. If this is a new pseudonym, she reveals (and proves) those of his attributes that she thinks are relevant for the credibility of this comment.
3. She enters her comments, possibly including an appreciation score for the article. This score information might be revealed to the public or restricted to given parties (e.g. the author).
4. The server makes the comment publicly available on the blog, together with (proofs of) its author's attributes.

2.2.4 UC2.4. Reading an entry

1. Frank specifies in his policy under what conditions he trusts the blog entries and comments, and hence under what conditions they will be shown on his screen. These conditions can involve statements about the content of the entry, the author's attributes, the contents and/or scores from the commenters, and the commenters' attributes.
2. Frank requests to read a blog entry. The server sends the entire entry with all comments to him, but only those that satisfy his policy (as a reader) will show up on his screen.
3. Optionally, there may be an "overrule" option by which a reader can also read untrusted entries and comments. The lower level of trust should be made clear by the interface though, e.g. by displaying the text in a different color.

2.3 UC3. Enterprise workflow scenarios

2.3.1 UC3a. Computer part workflow

!Eugene had over-clocked his computer. At some point in time, suspicious smoke came out of the box. Since then the computer did strange things. It required !Eugene to click an additional OK-box for every new application he wanted to start. It also turned itself off after 3 hours of uninterrupted web browsing. It would also let !Eugene download only 3 emails per hour and let him make only one blog entry per day. !Eugene concluded that over-clocking had turned his computer insane and brought it to customer service.

At the customer service of ninja-computers, they recorded the name, address, phone number, region and birth date of !Eugene as well as the password for the encrypted file system and credit card payment information. The computer was disassembled and the parts were sent to certain departments. The customer service took care to only send the very needed information to the respective repair stations. Thus the password for the encrypted file system was only given to the hardware repair. The phone number would only be given to those mechanics who would need further information from !Eugene.

It turned out that some mechanics thought the CPU got severely damaged and would have to be replaced. The power unit needed some external repair and the hard disk had to be sanitized to rectify remains of viruses stored. A mechanic found that the strange melting produced by over-clocking the CPU put it in an unstable state. They tried to fix the CPU by introducing different statements about the correctness of CPU calculations and moved the spurious remains on the hard disk into hidden areas.

Each mechanic should be able to specify which type of contents he trusts. Trust can be based on the type and content of an item, and on the attributes of the mechanic who entered the entry. There is also some "recursion" in trust; for example, if one trust mechanic Y but does not trust the analysis and tests made by repair unit X, and mechanic Y based his diagnosis on analysis and tests made by repair unit X, then one may not trust this diagnosis by mechanic Y.

In addition to trust policies, there is also a need for access control policies as not all suppliers will need to know the password for the encrypted filesystem. Additionally, as !Eugene has little confidence, he wanted to specify who is allowed to see what. For example, the disk repair should get the password but not the payment information.

It should also be possible to specify what happens to the data, e.g. whether it can be used for research purposes, whether it can be merged with other data, etc.

2.3.2 UC3b. Company HR workflow

A US-based company has numerous subsidiaries worldwide. Alice and Bob, two engineers respectively based in German and Swiss subsidiaries apply to the same internal job offer at the American headquarter.

Each subsidiary maintains an employee database containing employees' names, addresses, bank account numbers, skill sets, salaries, and previous evaluation results. Since databases are respectively hosted in Germany and Switzerland, they are subject to stringent and different privacy rules. The full databases can never be transmitted to the US.

The final decisions who will be hired is made by the US board based on a blinded database containing only skill sets, salary ranges, and evaluation results. The evaluation results are signed by the manager who was in charge of the employee when the evaluation was made.

Once the decision is made, the legal department of the concerned subsidiary receives details regarding the position change in order to inform the current manager, start the relocation process, and so on.

2.4 UC4. Social Network Sites

This section lists a number of use cases focused on the use of Social Network Sites (SNSs). We have limited the description of the use cases to the actual interaction of the user with the system, which will normally take place using an Internet connection. The use case describes the interaction only, and does not go into details concerning the technical implementation of a certain use case.

One of the key privacy issues in SNSs is the differentiated access to content posted on the SNS. So far, we have identified two general ways in which this access can be managed:

- On the level of connections: each (new) connection is granted certain access rights. For instance, friends could have different access rights than colleagues.
- On the level of content: each new piece of content is supplied with an overview of the (classes of) connections that may access this content.

We have not resolved which one of the two is preferable from the point of view of effectiveness, efficiency, and user friendliness.

2.4.1 UC4.1. Register to an SNS

1. When registering, Mariangela supplies a selected username and password.
2. The SNS checks whether the username is still available. If not, she must select another username.
3. She gets first time access to an empty profile page.

2.4.2 UC4.2. Establish connections

1. Mariangela searches the SNS for potential connections. The means through which she finds these connections is less relevant: based on exact e-mail addresses, joint groups, via existing connections, particular search terms, etc. The result is the presentation of one or more potential candidates for connecting.
2. She selects the name of the potential connection, selects the types of information the potential new contact will have access to (see key issue above) and clicks: “Connect!”
3. The invited person receives an invitation to connect with a link to the profile of the inviting person, which can be opened before acknowledging the invitation. The profile preview (that can be checked before accepting the invitation) reflects the exact view that the invitee will have based on the settings the inviter selected when drafting the invitation.
4. The invited person can click “Accept!”, “Decline!”, or first visit the preview.
5. In the case the invitee accepts, he/she has to select the types of information the new contact will have access to (see above).
6. The inviter gets updated on the decision of the invitee, and the connection is registered in the SNS system, viewable from the profile pages from both the inviter and the invitee.

2.4.3 UC4.3. Post content to own profile

1. Josha selects a piece of content that he wants to post to his own profile (e.g. a photograph or a blog entry).
2. He needs to login, to ascertain that he is the owner of the profile.
3. He is presented with a number of connection(s) (classes), that potentially have access to the photograph.
4. He selects which connection(s) (classes) will have access.
5. The content gets posted on Josha's profile page.
6. Depending on the preferences of the profile owner, a message containing information of the upload is sent to selected connections.
7. Depending on their preferences, the selected connections receive a message that new content has been added.

Concerning points 3. and 4. Of this use case, we refer to the key issue about differentiated access described in the introduction of this scenario.

2.4.4 UC4.4. Post content to another profile

1. Inga selects a piece of content that she wants to post to another profile (e.g. a photograph or a comment).
2. She needs to login, to ascertain that she is a legitimate SNS member.
3. The settings of the destination profile owner are checked whether he allows the uploading of content by this particular uploader to his profile.
4. If yes, the content gets posted on the profile page.
5. Depending on the preferences of the profile owner, a message containing information of the upload is sent to selected connections.
6. Depending on their preferences, the selected connections receive this message that new content has been added.

In principle, the profile owner to whose profile information is being uploaded, should be in control who will be able to access the newly posted information. This would be in line with points 3 and 4 of the last use case titled Post content to own profile. In order to keep the use case manageable, we decided not to include this option here.

2.4.5 UC4.5. Control access to and use of SNS content (profile information or other content)

The goal of this use case is to differentiate access to and use of SNS content between different parties:

- (Different categories of) other SNS members
- The SNS provider
- Third parties (such as automated mash-ups using APIs).

The use case follows the following steps:

1. Content is posted on the profile (see use case above).
2. Access to the content is controlled.
3. Another SNS member, either a connection or not, the SNS provider or a third party wants to use the content (copy, modify, add comments). Obviously, the involved party needs to have obtained access to the content based on step 2 first.
4. The profile (and content) owner wants to define the allowed use of the content and differentiate this between parties (different connections, other SNS members, SNS provider, third parties).
5. Depending on the rights given to the party involved he can or cannot perform the desired action.

2.4.6 UC4.6. Report on access to and use of SNS content (profile information or other content)

The goal of this use case is to generate an overview of accessed profile information (logging), once again differentiated per user, such as other SNS members, the SNS provider and third parties (including APIs).

1. After his profile information (or other content) has been accessed or used, according to the steps in the use case above, Hannes (the profile owner) wants to have an overview of the accessed information and what has been done with it. This enables the profile owner to check what happens with his data/content.
2. Either each time information is accessed or used a report is sent to the profile owner, or this is done on a daily/weekly/monthly basis. A distinction between access and use is desirable.

2.4.7 UC4.7. Transfer profile to another SNS (data portability)

Mariangela decides to use more than one SNS or to switch to another SNS (to follow her friends' advice). In order to save time she wants to transfer (parts of her) profile information to the new SNS.

1. She first creates an account on the new SNS.
2. Her identity (as a profile owner) is checked and the (old) SNS allows the transfer of the profile information.
3. A connection between the two SNS services is established.
4. The profile information is transferred or copied, preferably immediately into the new profile.

2.4.8 UC4.8. Integration of mobile SNS applications

1. Hannes has an application on his SNS profile that uses (location) data from a mobile device (for instance a map that shows his location).
2. The location data from the mobile device are sent to the SNS. This may be a continuous process, or only be occurring at several (user-triggered) intervals.
3. The data is verified and connected to the correct SNS profile.
4. The data are transferred to the application.
5. The SNS processes and displays the data.

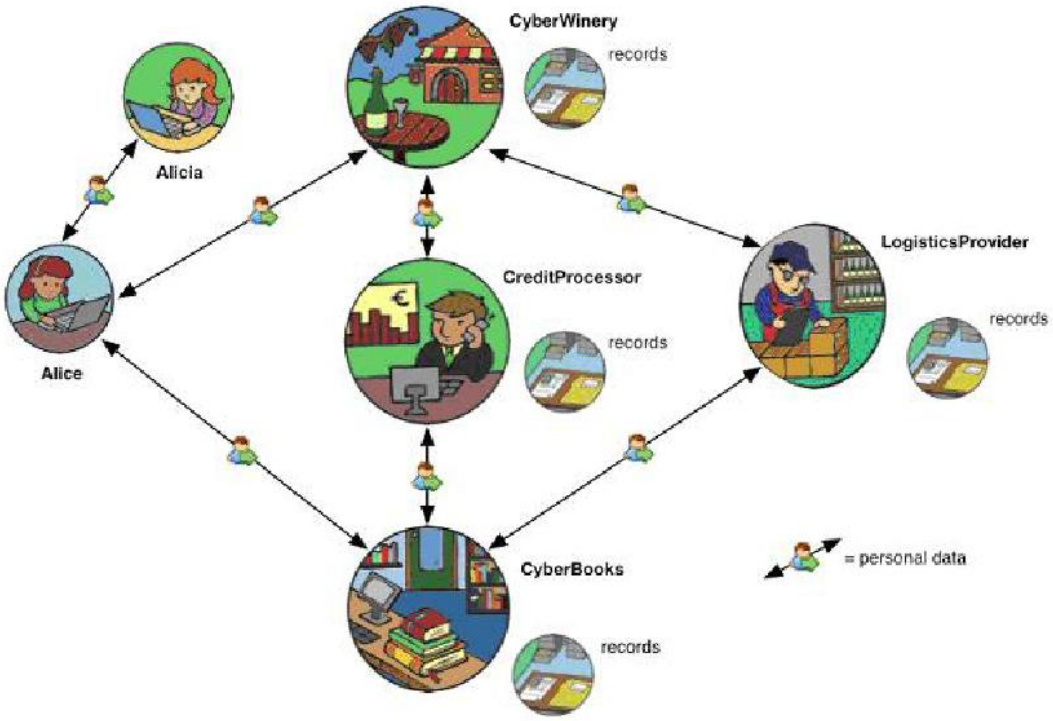
2.4.9 UC4.9. Close an SNS profile

1. Florence decides to stop using the SNS.
2. She selects the option to close her profile.
3. All profile information is erased. (Important: all information that originated from her profile page, and/or has been distributed throughout the SNS and beyond, should be erased as well).

2.5 UC5. Anonymous credentials and identity management

After a recommendation from her sister Alicia, Alice considers to purchase a box of white wine at ‘CyberWinery.com‘. The figure below shows the main entities involved in the scenario and the data flows in a classical, non-privacy-preserving setting. For example, prior to the purchase Alice is likely to first create an account at CyberWinery, thereby disclosing personal data. The account will store purchase data, personal preferences, and possibly even credit card data. CyberWinery has outsourced warehousing and delivery to ‘LogisticsProvider’, which requires data from CyberWinery (like a delivery address). For the payment by credit card, CyberWinery checks Alice's credit card details for authorization at

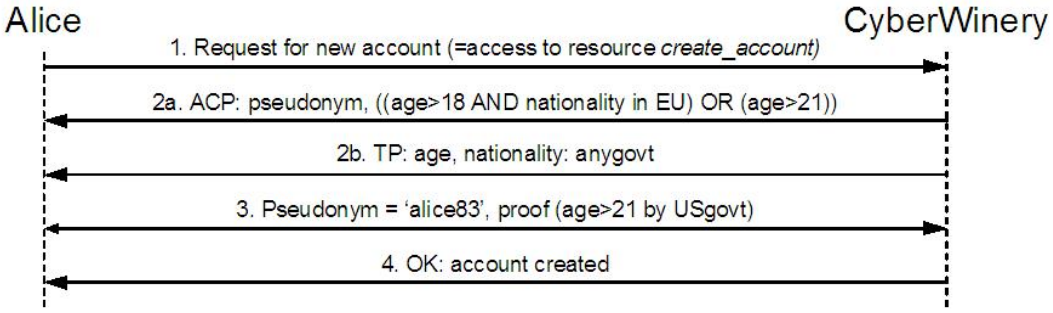
'CreditProcessor', which stores the transaction details for their own business and accounting purposes. Other services may also be present, such as CyberBooks which is recommended by Alicia for the purchase of the Good Wine Guide. This purchase again requires Alice to register with her personal data and CreditProcessor and LogisticsProvider are also possibly involved in this transaction.



Alice, a vigilant and sensitive 'Netizen', is aware of the risks involved in online transactions and knows that the loss of personal information can cause severe financial and reputational damages which are difficult to repair, and she has heard that the use of personal data by others may lead to discrimination, exclusion, and social sorting. Because of this, she adheres to the principle to share a minimum amount of personal data on the Internet. Fortunately for Alice, CyberWinery uses anonymous credential technology, which assures her that she can make use of a secure privacy-enhancing identity infrastructure that complies with current data protection legislation. CyberWinery also has respected trust marks and provides clear information about the buying process.

2.5.1 UC5.1 Login use case

CyberWinery lets Alice control the amount of identity information that she gives away by allowing purchases via pseudonymous accounts that are created using anonymous credentials. CyberWinery demands Alice to prove that she is either over 18 and an EU national, or over 21, but this is possible even within Alice's choice to be pseudonymous. Alice only needs to attribute a number of anonymous credentials (issued by Trusted Third Parties, such as her bank or the State) to her chosen pseudonym.



Alice starts the interaction by issuing a request (1) to create a new account. The CyberWinery expresses its requirements to create a new account as an access control policy (ACP) with the account creation service as the resource. CyberWinery's ACP specifies (2a) that it needs a pseudonym and a proof that either the user is over 18 and of EU nationality, or over 21. CyberWinery's trust policy (TP) specifies (2b) that it trusts any official government to certify the attributes age and nationality of users. Alice chooses (3) to use pseudonym 'alice83' and proves to the CyberWinery that she has a valid credential from the US government stating that she is over 21. The CyberWinery verifies the proof and creates (4) the requested account.

Alternatively, for some reason Alice may in (3) prefer to prove the statement (yearofbirth=1971). For example, Alice may not be using standard certificates rather than anonymous credentials and happens to have a certificate of that statement lying around. The CyberWinery has to be intelligent enough to realize that (yearofbirth=1971) implies ((age>18 AND nationality in EU) OR (age>21)), and should grant access.

2.5.2 UC5.2 Buying use case

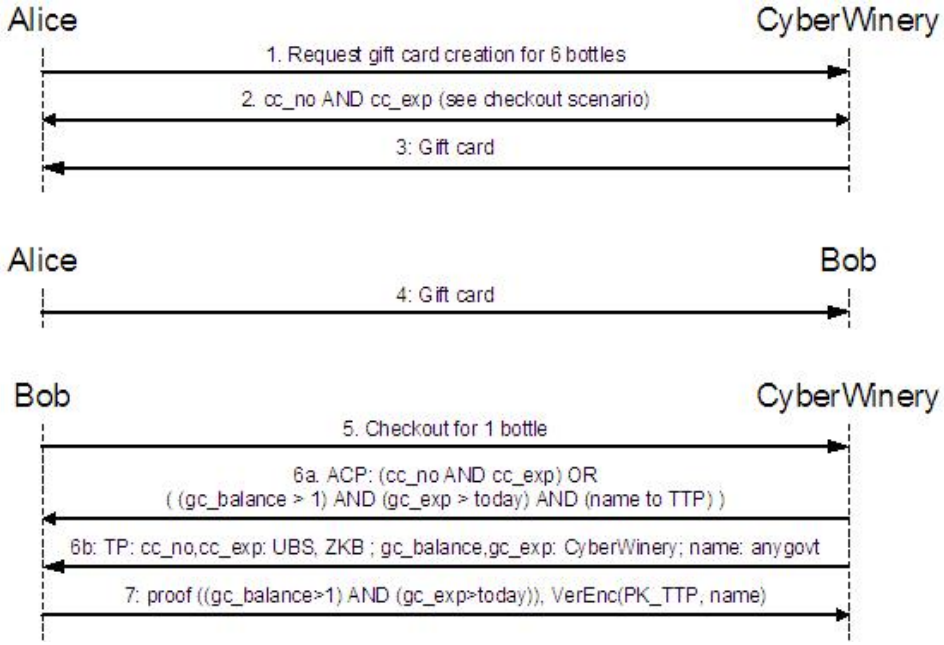
Alice brows the catalogue, chooses a box of white wine, puts it in her virtual shopping cart and proceeds to the checkout. She now has to give her credit card information to CyberWinery to pay for the wine, and her address to complete the shipping.

By clicking on the checkout button Alice initiates an access request (1) to CyberWinery's buying service. CyberWinerys' access control policy (2a) states that in order to gain access to this service, Alice has to reveal her address to the the Logistics Provider (LP) , reveal her

credit card number and expiration date to the merchant, and use her credit card to sign the total amount of money due. The trust policy (2b) states that any government is trusted to certify Alice's address, and that either UBS or CreditSuisse (both are Swiss banks) are trusted to certify Alice's credit card info. Alice's privacy policy however (3a) states that she's only willing to reveal her credit card info to "good businesses", which in her trust policy (3b) is defined as businesses with either a gold label from the BetterBusinessBureau (BBB) or a reputation more than 5 stars from PrivacyOrg. Finally, her data handling policy (3c) states that her credit card information can only be used for the purchase and has to be deleted immediately afterwards. Similarly, her address can only be used for shipping and has to be deleted as soon as the item has been shipped. CyberWinery accepts the DHP (4) and proves that it has a BBB gold label. Alice sends over (5) her address, verifiably encrypted under the Logistic Provider's public key, and proves the correctness of her credit card information.

2.5.3 UC5.3 Gift certificate use case

Bob's birthday is coming up. Alice would like to buy him a box of 6 bottles of wine, but since she isn't sure about his taste, she prefers to give him a gift card. CyberWinery has gift cards that allow the recipient to buy 6 bottles of wine, probably within a certain price range. The bottles can be different ones, though, and the 6 bottles do not have to be bought in a single order. If the gift card is one (or more) anonymous credential issued by CyberWinery, then CyberWinery cannot link Alice's purchase of the gift card to Bob's spending of it, and also cannot find out whether Bob is buying the different bottles with the same gift card or with different ones. Still, it can detect overspending of a gift card, and refuse to proceed with the transaction if this occurs.



In steps 1-3, Alice buys a gift card for 6 bottles. Payment is done similarly to the buying use case above. In step 4, Alice sends the gift card (in the form of one or more anonymous credential issued by CyberWinery) to Bob. When Bob checks out one bottle of wine (5), CyberWinery requires him to give either his own credit card information or show a valid gift card that has not expired and has not been overspent (6a). In the latter case, Bob also has to submit his real name to a trusted third party TTP, so that he can be held accountable if overspending is detected afterwards. CyberWinery's trust policy (TP) thereby specifies that (6b) either UBS or ZKB (both are Swiss banks) are trusted to certify Bob's credit card info,

that only gift cards certified by CyberWinery itself are trusted, and that it trusts any official government to certify Bob's name. Bob chooses to pay with a gift card and proofs this fact to CyberWinery (7). Additionally he sends over his name, verifiably encrypted under the trusted third party's public key. CyberWinery checks the proof and proceeds with the transaction if everything is okay.

Overspending can be checked online or offline. If it is done online, then Bob doesn't really need to send his name to the TTP, because CyberWinery can simply abort the transaction as soon as overspending is detected. If it is done offline however, and it is impossible to cancel the transaction (for example because the wine has already been shipped), then CyberWinery should contact the TTP to obtain Bob's real name to make him pay for the wine.

2.5.4 System overview

The figures in the scenarios given above illustrate the sequence of messages that are exchanged between the parties performing online transactions. The messages exchanged are access requests, different types of policies, confirmations, proofs, etc. For describing the scenarios, these kinds of illustrations of the message flow are sufficient to give an idea of what the policy language(s) should be capable of. However, with the intention that such systems shall eventually be working and deployed, a closer look at the individual system components is necessary. Although a detailed description of the design of such systems is too early at this stage, we want to give a high level description of the system. This might produce some useful ideas for the development of the policy language(s) and possibly new requirements arise by thinking about the whole system in which the policy language(s) shall finally be embedded.

Client components

Users have a credential repository that contains all the credentials they hold. For example, Alice has a number of anonymous credentials in her credential repository. One is her digital ID card issued by the U.S. government, mentioning her name, address, birthdate, etc. Another is a credit card issued by UBS (a Swiss bank), mentioning her name, the credit card number, and the expiration date.

Associated with each attribute contained in the credentials, a user has a release policy (release policy is the name of an access control policy for PII) specifying under what conditions she is willing to reveal this attribute A user's trust policy states the requirements on issuers of certified attributes, i.e. who is, from Alice's point of view, allowed to certify which attributes. For example, Alice's release policy, together with her trust policy states that she reveals her credit card data only to sites approved with a gold label from the BetterBusinessBureau or a 5-star reputation from PrivacyOrg.

A specific data handling policy might also be associated with the attributes of the possessed credentials. These policies state what the obligations on the data receiver are, once she received the data. For example, she has to use the data only for specific purposes and must delete the data after a defined period of time.

Server components

A credential repository contains all the credentials that a service provider holds. CyberWinery has for example an anonymous credential which was issued by the BetterBusinessBureau to state that CyberWinery has gold status.

A service provider offers a number of services to which access is restricted. Associated with each resource there is an access control policy specifying what the requirements are in order to get access to the resource. For example, CyberWinery offers the services create_account and buy_service. Associated with the create_account resource, there is a policy specifying that the user needs to provide an uncertified pseudonym, and has to prove that she is either over 18 and national of an EU country, or over 21.

Like users, service providers maintain trust policies that state who is, from their point of view, allowed to certify data. For example, for the age and the nationality attribute, CyberWinery states that any national government is trusted to certify this kind of data. Service providers might as well have release policies associated with the attributes contained in their credentials.

Request processing

Each party that is involved in an online transaction has to perform a number of steps in order to process an incoming request. The following describes some of these steps to get an impression of how users and service providers handle transactions.

When a user requests access to some resource, the service provider looks up the policy associated with this service. This lookup may include deriving the specific policy from a more general policy for all of its resources. The service provider also determines the trust policies that are associated with the attributes contained in the access control policy that is going to be sent to the user. Depending on the policy, additional steps might have to be performed before the policy can be sent to the other party. Such steps could include resolving special ontologies that were used when creating the policy, etc. As soon as the access control policy for the requested resource is fulfilled, the user gets access to the resource.

A service provider maintains session information to be able to know whether a user already proved to satisfy the requirements contained in the access control policy. In case the user already proved to satisfy parts of the requirements, only the parts she still needs to prove are sent. A user who has already proved to satisfy all requirements gets direct access to the requested resource.

In case a party is requested to reveal or prove some predicate over an attribute, the party checks the corresponding release policy to see whether she is willing to do so and what the associated requirements would be. The other party could then for example be required to show certain credentials. Thereby, also the relevant trust policies have to be determined in order to let the other party know which issuers are allowed to certify the required attributes. In addition to the release policy, also the data handling policy applicable to the corresponding attribute has to be determined and sent. Again, the trust policies relevant for the content of the data handling policy need to be determined and finally sent to the other party. If a party is willing to reveal or prove some predicate over an attribute without further requirements, or the requirements have already been fulfilled by the other party, then the attributes of the relevant proofs can be sent.

2.6 UC6. User Interface perspectives

2.6.1 UC6.1. Trust and assurance HCI

Trust policies should include statements about trust factors that can influence the trust of end users, such as statements of privacy seals on the service side, reputation metrics, statements about the system configuration, etc. Input for trust negotiation should not only be claims by the service side/communication partner, but also statements by third parties, such as statements from consumer organisations on whether a service is "blacklisted".

2.6.2 UC6.2. User interfaces for policy display and administration

For simplifying the management of data release policies (preferences) for the user, our UI proposals have built on providing a set of predefined data release policies, which can be customised "on the fly". The user's data release policies should be able to state

- what data categories or what concrete data values may be released for what purposes and to whom or to what types of recipients (e.g., only "trusted" partners)
- conditions under which data may be forwarded to other parties
- type of pseudonyms that should be used under this release policy (e.g., transaction pseudonyms, role-relationship pseudonyms, etc.)
- the expected obligations and reputation of the potential recipient.

Work package 4.3 needs a policy language to describe all the attributes needed by its interfaces. Research is currently heading towards a fine granularity of purposes, so if a transaction with a service is conducted, and this transaction includes the ordering of a product, and at the same time the subscription of an advertisement mailing by the service, this could be expressed by two seperate policy statements for the different purposes "order" and "advertisement".

Such a policy would need to be able to express:

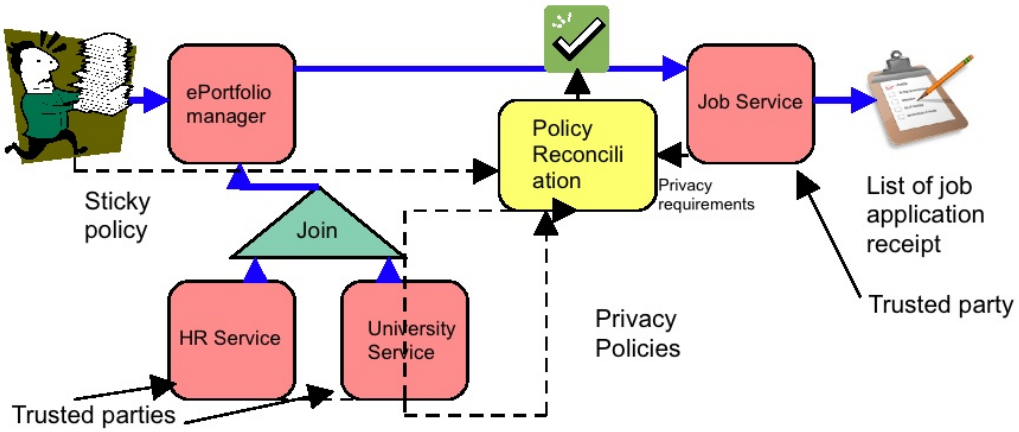
- the types of data
- the purpose for which this data is collected and processed
- the processor of the data, following a specific defined structure, including address data
- a field for a full version of the legal policy, possibly with further substructures
- a field for the condensed version of the legal policy, possibly with further substructures
- a field for a short version of the legal policy, possibly with further substructures such as: the identity of the controller and the purposes of processing (w/exceptions), any additional information which in view of the particular circumstances of the case must be provided beforehand to ensure fair processing, and a clear indication must be given as to how the individual can access additional information
- a reference to an icon illustrating the policy
- the recipients or categories of recipients of the data and how the data is handled after it has been transferred to them
- a point of contact must be given for questions and information on redress mechanisms either within the company itself or details of the nearest data protection agency

2.7 UC7. Service Composition Scenarios

This section describes some use cases related to service composition. For additional information on those use cases, look at PrimeLife's work package 6.3, which focuses on service composition. More details can be found in the D6.2.1 and upcoming H6.3.1 documents.

2.7.1 UC7.1. ePortfolio scenario

This use cases sketches an ePortfolio scenario as an illustration for service composition.



Overview of a service composition

Florence wants to apply for a job using a composition of services that includes the following (see figure above): an ePortfolio manager that contains details concerning her education, employment history, and personal details; a HR service (provided by the employee's employer) that can send a verification to the ePortfolio manager that Florence was/is an employee and can also confirm that she has taken training; the University Service can provide proof of qualifications/certifications; the Job Service provides offers based on Florence's preferences. We assume in this case that the services within the composition are trusted.

Privacy-enabled Policies

The different services within the composition enable policies to be attached to data; they outline the way the data should be handled within a service composition.

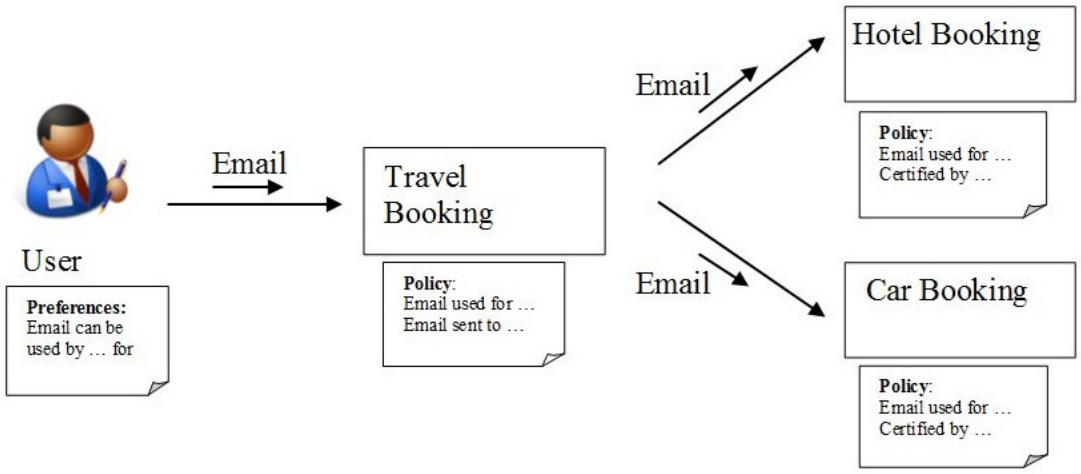
Use Case

- Florence would like to use the web service composition to search for new jobs.
- The ePortfolio combines existing data and aggregated data from different sources (HR and university).
- The privacy-enabled ePortfolio is provided to a service that offers jobs (i.e. Job Service).
- The Policy Reconciliation service is the intermediary between the ePortfolio and the Job Service.

- The Job Service agrees to enforce part of the preferences of the ePortfolio (e.g. do not send data to a third party if the third party plans to use data for statistics and delete data after 10 days.

2.7.2 UC7.2. Data Flow in Travel Booking

In composite services, it is frequent that personal data collected by the composition be shared with different services. For instance, in the figure below, the "Travel Booking" service collects Inga's e-mail address (Email) to contact her to confirm the reservation. Third-party services may be subsequently called in order to book the travel, e.g. "Hotel Booking" and "Car Booking". Both may also require Inga's e-mail address to contact her in case of an emergency or an unexpected change in the reservation.



The data handling policy of each service getting specific personal data has to match the user's preferences. For instance, if the user only provides his e-mail address for booking or statistical purposes and with the obligation to delete it within six months, involved services can only use this e-mail address for the specified purposes (i.e. cannot use it for advertisement) and must delete this e-mail address within six months.

The user also needs a way to specify whether secondary use (i.e. Travel Booking sharing user's e-mail address with third parties) is authorized.

The policies of all services are matched with the user's preferences.

- Option 1: the user gets a global policy that combines all third-party policies. This requires an appropriate language and tools to merge and generalize policies.
- Option 2: the user "delegates" to the front-end service the right to select third-party services and hand user's personal data to those services. The policy and preference language must support conditions on third parties. For instance, "my personal data can only be used for statistics and can only be shared with third parties in Europe".

2.7.3 UC7.3 Data Flow in Human Resources

The previous use case describes a common data handling scenario with secondary use. A user (i.e. a human being) is in front of a service that consumes his personal data. The user has data

handling preferences and the service has a data handling policy. This use case shows that this client-server approach is too restrictive and insufficient.

Let's illustrate this with a concrete example. An employee provides PII (e.g. nationality, number of children, etc.) to his employer. The employee and employer agree on how this PII will be handled. For instance, a sticky data handling policy could define that PII can be provided to specific third parties for a specific purpose.

In well-defined cases, the employer can provide to a subcontractor (e.g. financial auditor or tax advisor) the PII provided by the employee. The employer has to make sure that providing the employee PII to this subcontractor does not violate the sticky data handling policy.

In such scenarios, a human being (e.g. subcontractor) may have to expose and enforce privacy policies. The border between data handling, access control, and rights management may disappear.

- Any party can act as data source and data consumer. In other words, privacy preferences are not restricted to human beings (users) and privacy policies are not specific to services.
- Access control and data handling are not independent. For instance, the sticky data handling policy associated with PII can directly impact the access control of the employer's HR service, e.g. any subcontractor can access the employee's PII if it commits to fulfill some obligations.

2.7.4 UC7.4. Privacy tradeoff in search/discovery

When searching for services (e.g. an available room in closest hotels), there may be a tradeoff between quality of service (e.g. functionality) and privacy. For instance, Inga is using a "SearchHotel" (composite) service and will provide her location. She knows that her query for an available room will be sent to 23 hotels in his neighbourhood. If she chooses to disclose more personal data (i.e. less privacy) such as smoker/non-smoker and nationality, which are expected by some hotels to check for availability, her query will reach 47 hotels. She may get a question like: "With your default DH-preferences, 23 out of 78 services will be queried. A majority of users of this service have less restrictive policies matching 47 out of 78 services. Do you want to proceed or change your preferences?". The service could also use a more appealing scenario and offer a visual representation of the tradeoff between privacy and functionality. To summarize, the user can specify "optional" part in data handling policies. For instance, the user must provide her location and she can optionally provide other personal data to improve the quality of service.

2.8 UC12. Logging/Auditing Use Case

When personal data is processed the law requires many actions, such as modification - in some cases even access - to be logged for auditing purposes. A typical scenario that arises in this context is the following:

A governmental agency stores personal data for their purposes (could be tax, health, or social security information). It grants access to this information to its employees.

- During the process the data is first provided in writing by a citizen.
- Subsequently, the data is entered into the system by an employee.

- Later the citizen calls in and some information is changed by a different employee.
- Another employee who has had difficulties with the citizen privately, accesses the data and changes it incorrectly.
- Finally a decision on some application by the citizen is taken based on the incorrect data.
- The citizen now wants access to the logbook to prove that someone has changed the information without a valid reason.

2.9 UC13. Privacy of location-based information

The diffusion and reliability that mobile technologies have achieved provide the means to exploit location information to improve current location-based services and applications in a novel way. In this case, location is one of the user's attributes that might be passed to the service provider and can therefore represent a piece of PII. In this context, privacy concerns are increasing, calling for more sophisticated solutions to provide users with different and manageable levels of privacy.

Two privacy aspects that might need to be considered.

- Location information, just as any other PII might be subject to restrictions (cannot be known or further communicated). From a policy language point of view, a language for location-based applications can give the users the ability to specify protection degrees on her location information. For instance, the user - whose identity might be known - does not want her location to be known at a fine grained-level (e.g., area of not less that 20km). This might impact the policy language if we want the user to be able to express desiderata for protection degrees on her location information.
- Location information can be exploited for re-identifying the users in scenarios where the user identity is otherwise not known. This aspect might be more related to data protection (than to support required in the policy language). It resembles exposure risks due to user profiling (and user re-identification based on this), as it aims at avoiding re-identification of users joining the locations from where requests originate with information on the locations of each user (available in a database or through observations). However the peculiarity of location information requires considering specific attacks aimed at determining the identity of the users by exploiting information on their locations and developing effective solutions counteracting them.

Chapter 3

Other perspectives

The following set of use cases describes larger perspectives on the use of policy languages. They have been brought to our attention by members of the W3C Policy Language Interest Group.

3.1 UC8. Privacy Policy Management

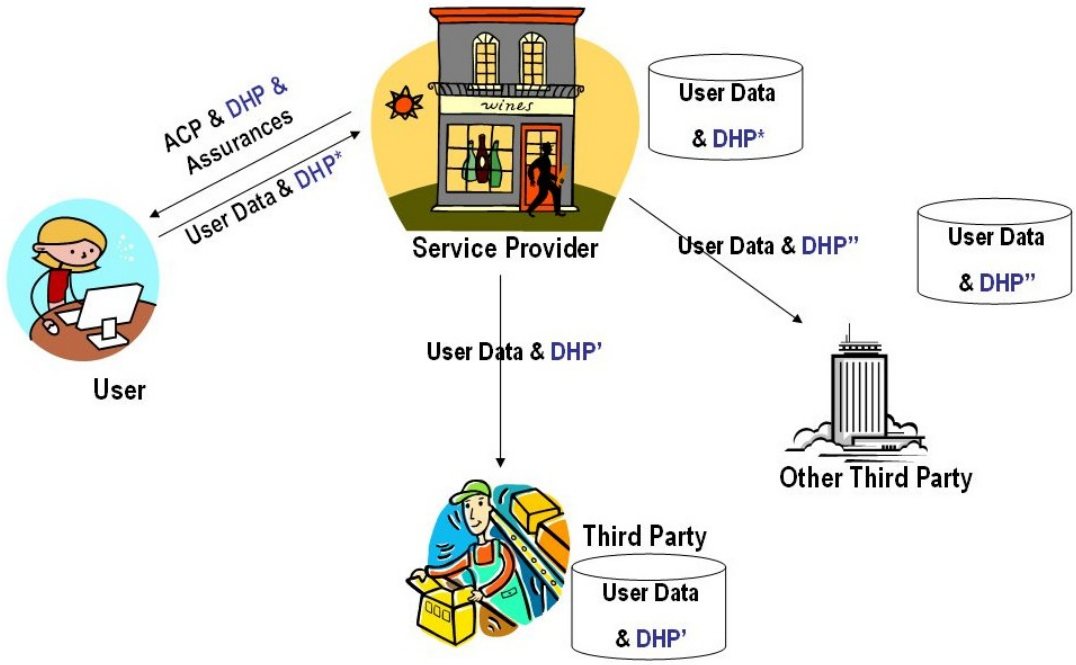


Illustration by D. Sommer

Original Author: Marco Casassa-Mont, HP Labs, UK (with kind permission from the author, Thank you, Marco)

from: http://www.w3.org/Policy/pling/wiki/UseCases [PLING-UC]

Description: A recurrent use case that I came across in various contexts (EU PRIME Project, interactions with customers, etc.) is how to use policies to deal with privacy enforcement and compliance checking in organisational contexts. This is pretty much consistent with some of the points already highlighted in Michael Wilson's use case.

Organisations and enterprises collect a lot of personal data and sensitive information in order to enable their businesses. In doing this, they need to comply with laws, legislation (HIPAA, COPPA, EU Data Protection, etc.), standards of business conduct, guidelines, etc.

Three key aspects are of interest: (a) policy representation (b) enforcement of (privacy) policies (c) policy compliance checking

Basic privacy constraints require handling users' consent, allowing access to the data only for agreed purposes and managing the lifecycle (e.g. data transformation, minimisation, deletion, etc.) of personal data driven by privacy principles. However, also other aspects such has security and business constraints need to be kept into account.

Personal data can be stored in a variety of data repositories (databases, LDAP directories, file systems, etc.) and can be accessed by people, applications, services. This data can be processed and disclosed to third parties.

A "blend" of personal preferences, business, security and privacy constraints need to be kept into account into "policies" dictating how to access, use, process, and disclose this information.

Some common requirements:

- need for more "integration" of business, security and privacy policies - need to leverage state-of-the-art Identity Management solutions (that might use, in some cases, proprietary/ad-hoc policy languages ...) - need to measure and demonstrate compliance to guidelines, laws and legislation

Policies (and policy management frameworks) can play a key role to deal with privacy enforcement and compliance checking tasks. However, one of the current limitations is that these aspects are currently addressed in a "compartmentalized" way, by using different policy languages and policy management systems (for security, privacy, etc.) that do not interoperate i.e. act in stand-alone ways. This creates issues in terms of alignment of policies, their consistency and overall impact.

How to make progress by recognizing that on one hand there are multiple policy languages and policy management systems and, on the other hand, more coordination and integration is required?

3.2 UC9. Federated Policy Management

Original Author: Marco Casassa-Mont, HP Labs, UK, (with kind permission from the author, Thank you, Marco)

from: http://www.w3.org/Policy/pling/wiki/UseCases [PLING-UC]

An enterprise/organisation uses a broad variety of IT tools and solutions. The enterprise IT infrastructure includes systems, tools and solutions deployed at different levels of abstractions: network, system, OS, information, application, service, business, etc.

Many of the involved systems, tools and solutions are configured with and driven by "local policies", defined by using specific (sometimes ad-hoc ...) languages. These "local policies" are often the effect of (human-based) "refinements" and "deployments" of high level business/ security/etc. policies. Different policies, policy decision points and policy enforcement points are used for security, business, privacy and other aspects.

- How to make sense of all of them? - How to ensure that their overall impact on the IT infrastructure is consistent with the high level policies and guidelines - How to understand what the impact of changes of "local policies" (let's say at network level or at the application level) is on the high level policies? - How to understand what the impact of changes of high-level policies is on some of these "local policies"?

This is what I call a "federated policy management" use case i.e. a use case where there is the need to understand and keep into account the overall set of policies deployed in an IT infrastructure and have an "integrated, coordinated and consistent" management of these policies.

In this use case many different policy languages are used, operating on different IT entities (at different levels of abstraction) and enforced by different policy enforcement points. It is unlikely that all these existing (local) policy languages are going to be replaced by a unique "comprehensive" language ...

Some requirements are about having a consistent "meta-representation/abstraction" of the core principles/aspects/constraints expressed by various "local policies" along with ways of defining dependencies and relationships between them.

This would help to better understand the overall heterogeneous set of operational policies, link them back to high-level policies and reason on top of it.

3.3 UC10. Authentication Strength Policies

Original Author: Giles Hogben, European Network and Information Security Agency

from: http://www.w3.org/Policy/pling/wiki/UseCases [PLING-UC]

Description: Governments and companies operating remote authentication systems often need to set machine-readable policies (or even human-readable policies used within organisations), specifying the minimum strength of authentication tokens required to access a service. Other features of the authentication context (as it is called in SAML) are also of interest such as the linkability features of the tokens used, and the registration and issuance procedures which surround how the token is bound to the person's identity.

Many proposals abound for sets of levels describing authentication strength, but there is no standardisation in this area as yet.

3.4 UC11. Corporate Security Policies

Original Author: Giles Hogben, European Network and Information Security Agency

from: http://www.w3.org/Policy/pling/wiki/UseCases [PLING-UC]

Description: We see a need for a common format for sharing controls used in corporate security policies. Obviously no company would want to disclose their internal security policy, but they would want to be able to view and modify standard expressions of security policies. This would facilitate certification bodies to apply standard evaluation procedures.

Chapter 4

Requirements

Note: the requirements have unique numbers for future references from other documents. They are not necessarily consecutive nor significant.

4.1 General principles

The following requirements are considered highly desirable for all the scenarios, therefore are described as general principles.

RG11 Measurability

This property of a policy language is fulfilled when the construction of the language allows to check that the policy has been followed correctly. A mechanism to prove that a rule has been applied is useful but not sufficient to demonstrate this property, since some policies can be applied a long time after the moment when it has been stated.

RG12 Unified model for access control and data handling

Data Handling policies being tightly related to Access Control policies, a unified model is a key success factor for a policy language. Even though in this document we often focus on access control and data handling policies separately, they are in fact closely related. A server's access control policy should not only specify what PII it wants from the user, but also how it is planning to treat the data. Here, the DHP is part of the ACP. On the other hand, a user's DHP may specify which third parties are allowed to see the PII, so that the ACP becomes part of the DHP.

RG13 Semantic compatibility with P3P

As much as possible, a policy language should be semantically compatible with the Platform for Privacy Preferences (P3P). Details of P3P semantics can be found in section 3 of The P3P 1.1 Specification [P3P11]).

RG14 Stickable policies

The 'stickability' is the property of the policy language that allows to attach a policy to data no matter how, where and when the data is sent.

RG15 Revocability

It must be possible to revoke a Data Handling policy the same way it is possible to revoke a credential.

RG16 Transparency

A policy language must be able to express that the data flow trace resulting of the transfer of the data between entities should be kept. There should be mechanisms in place to log the usage of personal data. Such log will span multiple trust domains in case of downstream usage and parts of it may travel with the data (sticky logs)

RG17 High-level Policies

The policies should be expressible not only on a low, i.e. more technical, level but also on a higher, i.e. more abstract, level. The benefits of this are for example that the policies can become shorter, easier to understand and also easier to formulate. Among other techniques, ontologies could be leveraged to bring the policies on a high(er) level. Instead of talking about credit-card-number, credit-card-name, etc., an ontology could describe such data under the class 'credit-card data', or even more general, 'payment data'. Then the policy can refer to concepts like credit-card or payment data if needed.

RG21 Data minimization

The policy language should support -- and encourage -- minimization of the amount of personal information that is revealed in order to gain access to a resource. The architecture should definitely not assume that all information about the subject is readily available when the access decision is made. Rather, the list of attributes that need to be revealed, or the predicate that needs to be proved, should be explicitly specified by the server, or perhaps even be the result of a negotiation between the client and the server. The client should then have the option to reveal only those attributes that are strictly necessary. Whether this is possible of course not only depends on the policy language, but also on the underlying authentication technology. In case for example X.509 certificates are used, the user has to reveal all its attributes so that the digital signature can be checked. The data handling policy should be determined in a similar "minimal" way: the general principle should be that the DHP only allows those actions that are needed to perform the service, rather than forcing the user to agree to a default, more lenient DHP. It should therefore be possible to associate different DHPs to different resources, and these should be tightly coupled to the relevant ACPs.

RG22 Anonymous and/or pseudonymous access control

A user shall have the possibility to access a resource in an anonymous or pseudonymous way. For an anonymous access, the server makes sure that the user fulfills the necessary requirements, e.g, age>18, while the required attributes allow the user to stay anonymous. This is of course only possible if (1) the required attributes (like age>18) are applicable to a big number of people and the user can therefore not be identified, and (2) the underlying technology supports proving of the attributes in an anonymous way (for example using the technology of 'anonymous credentials'). A pseudonymous access is similar to the anonymous one, with the difference that for every access a user does, he provides some kind of identifier - a pseudonym - which the server uses to recognize that the same (pseudonymous) user requests access. However, the server only knows the pseudonym and not the real identity of the user. This is important if a user wants to keep some profile on the server side, while the user still wants to be anonymous to the server. From a legal point of view, services must be offered pseudonymously whenever that can be considered reasonable for the service.

4.2 Requirements for language model and expressivity

R410 Meta-policies and policy generation

In some cases it is necessary to constrain the possible policies that can be attached to data, by some rules or guidelines. Those guidelines are locally enforced when defining preferences and policies. For instance, a data subject may define a rule (i.e. a policy) that forbids the creation of any preference allowing the use of medical data for advertisement. Such a policy could also be provided by a trusted third party.

This can be achieved by defining meta-policies that specify how policies can be customised (see R115 for a definition of constrained customisation of policies). The same mechanism can be used to specify constraints on policies that are generated, e.g. by a service in response to a user request. Those constraints can be derived from trust or access control assertions. This mechanism can rely on a way to express policy generation rules in the policy language itself.

4.2.1 Requirements for Data Handling policies

R111 Business logic to describe data usage

The business logic of an enterprise determines what actually happens with the data after it is received. If this business logic is described in a standardized way, for example using WS-BPEL (Business Process Execution Language), then it should be possible to automatically derive the DHP from it, or perhaps the BPEL description itself could even be part of the DHP.

R112 New usage should trigger consent

The policy language must support a mechanism to acquire new consent from the data subject if the data controller wants to change the policy. However, the data subject can indicate in her preferences that she will never agree to any changes to the policy, and that she therefore does not want to be bothered with requests for policy changes. Alternatively, one could have an opt-in mechanism, where the data subject has to explicitly state in her preferences that she would consider changes to the policy.

R113 Legal policies need differentiated layering (not much semantics for enforcement)

A policy language must include the possibility to express and address at least three layers of human-readable text to describe a policy to the user. This is recommended by the Op. 100 of the Article 29 group: Opinion on More Harmonised Information Provisions [Art29Op100] and Annex [Art29Op100A]:

- a *short version of the privacy policy,* with an addressable substructure to be defined
- a *condensed version of the privacy policy*, with an addressable substructure to be defined
- a *full (lawyers readable) version of the policy*, with an addressable substructure to be defined
- a fourth *layer to express the policy with iconography* should be available, with a set of icons to be defined.

R114 Technical representation of legal policies

The policy language should be able to express legal policy concepts (e.g. liability, data controller, data processor, etc. See also R329) and conditions relevant for machine-based

decision making, in a form supporting their digital storage, transmission, and processing. The semantics of the representation should be carefully considered and be compatible with the capabilities of an efficient processing engine.

R115 Constrained customisation of privacy policies

In specific cases, it is necessary to let the data subject create a sticky policy that is slightly different from the privacy policy of the data controller. We assume that 1) the data controller constrains which modifications are legitimate and 2) the data controller checks whether the provided sticky policy is indeed compliant with the initial policy.

Constrained customisation may be used to handle optional parts of a policy (e.g. data is collected for "current" purpose but, if user consents, may be shared with Z for statistics purpose). In this case, customization may be implemented as a picking an appropriate policy in a set of static policies. Customisation may also require instantiating variables (e.g. "commit to delete at date <now> + 6 Months" may become "commit to delete at date 01/01/2010" and "notify <data-subject>" may become "notify Bob").

R118 Support nested policies

The policy language must support nested policies. Thus, a policy could include a number of specific policies for further processing of the data.

R119 Express user preferences

A policy language must allow to express users preferences about handling of their data. In particular, it must be possible to express preferences for use of given credentials for a given purpose. The user should also be able to express general trust relationships independently of a given scenario or purpose. The language should be extensible enough to express new user-defined preferences.

R120 Describe server policies

There are two possible cases in this context,

- The server has no Data Handling policy. For instance, a basic data repository where the user stores his data and associated policy in the repository (whose duty is enforcing user policies).
- The server has its own Data Handling policy (one or multiple). The user's Data Handling policy should match one of the available server policies.

For simple user experience, complex Data Handling policies and preferences should be visualised in a simple way.

R121 Originator's policy

A policy language must include a mechanism to identify what the data subject allows, no matter who transmits the data.

R122 Logging/Monitoring/Auditing Policies

- It must be possible to inform the user about the data collected during the usage of the service (date, location, actions, credentials used, etc.)
- It must be possible to express the scope of the retention (page, session, duration) and usage (extend user experience, debugging, legal requirements) of the collected data.
- It must be possible to express how the data is collected: how, when, where it is stored.

R131 Data model primitives

The policy language must make consistent use of (at least) the following concepts:

- date and time
- location

Those concepts are essential for the expression of data usage constraints, e.g. some data can be displayed in some particular locations or can be valid for a limited period of time.

R135 Security levels

The level of trust does not only depend on the claims made by the subject, but also on the underlying technology that is used to prove the validity of these claims. Probably the policy designer should not be bothered with technical details such as cryptographic algorithms and keylenghts, but given that the language should be useful in both low- and high-security environments, some notion of "security levels" seems appropriate. What these security levels imply on the underlying technology and infrastructure could then be specified in a separate ontology.

R136 DHP ontology

The language should provide in a standardized list (i.e., an ontology) of data purposes and types of data. This list should be extendable, as we can impossibly foresee all items that should appear in this list. A requirement that can be derived from scenario UC2 (blogging use case) is that we also need a classification of "conversation topics", so that the policy one can relate the topic of an article to the qualifications of an author. Obviously this list should be extensible as well.

R139 Enforcing DHPs

Technological means to enforce data handling policies are limited. A trusted software infrastructure can assist in automatically adhering to a DHP (e.g. deleting data on time) and in logging access for audit purposes, but eventually these systems can always be circumvented by a malicious user (e.g. by forwarding a picture of the screen displaying the sensitive information). In the end one will have to either trust the receiver to adhere to the DHP that was agreed upon, or to trust an external auditing agency to correctly certify such receivers. It should be possible to express this type of trust in the policy.

R140 Auditability -- obligations toward logging access

- A policy language must support mechanisms to express what types of processing (Access, Modification, Transferal, Deletion) need to be logged, and what specifically needs to be logged (change management, role, name or pseudonym of the person in charge of the processing)
- A language defining the logging also must be defined

R141 Breaking the glass

A special case of a policy: the law prescribes certain areas, where access and processing is compliant, although clearly not within the prior consent. In these cases, it might be reasonable to invoke special mechanisms for transparency (i.e. the glass is broken = the prior consent has been exceeded). Such a policy could state that certain entities are entitled to access the data, but then certain obligation regarding information of other parties, esp. the data subject or data protection supervisors might be triggered.

R142 Capture user intent

(As opposed to the service purpose of data handling. See also R143) This property is needed to differentiate the user intent when using a service (e.g. "buy a book about philosophy") from the purpose of the service itself (e.g. "sell products: book or CD or DVD or video game"). The user intent does not necessarily match the service purpose. The mechanism for capturing the user intent may be simple, however, processing it requires use of semantics and logic.

R143 Purpose of data processing

The purposes of data processing or data handling by a service usually stay the same across the usage of the service for all transactions. However, it is not always clear for the user whether a given piece of data is going to be reused for a purpose other than the most obvious one. For example, an address can be used for shipping, but also for later marketing actions. The policy language must allow to express all the different purposes of data handling by the service, so that the user can be informed about the less obvious purpose of data processing.

R144 Express obligations

A policy language needs to express all the obligations of the processing party. Such obligations can be derived from the law, but also from consent. Full coverage by a machine readable policy will never be achievable, as the law often requires human interpretation (that is why we have judges). Obligations should be ready to cover the scope of purpose limitations, which is difficult to translate, since it is hard to describe, whether an access, storage, or transfer was made for one or another purpose.

R146 Notification/feedback channels

It should be possible for the Data Subject to require notification messages be sent back to him to inform him about the obligation enforcement conditions of his exported data. This is an important feature since actually few security systems are able to provide a report to the Data Subject about the usage of his private information. In case of misbehavior, these notification messages can be used as a proof for accountability. It should be possible to link different notification messages for the same piece of exported data together, so that they form a trace of how the data usage and possibly a proof of privacy policy violation.

4.2.2 Requirements for Access Control policies

R211 Declarative language to represent access control policies

The policy model should be accompanied by a language enabling the specification of access control policies. The language should be declarative and accompanied by a clear and unambiguous semantics for the policy specifications.

R212 References to policies

The policy model should provide support for reusing a policy. Referencing can be done either directly or by inheritance:

- Direct reference to a policy (inside another policy), e.g. using reusable identifiers for policies.
- Inheritance of a policy applied to another object (a service, a piece of data...) by explicitly referring to that object.

R213 Role models - family, friends, wider access control

The access control model should support multiple access control paradigms, including role-based access control and attribute-based conditions. Roles could also be incorporated in attribute-based conditions by the consideration of proper attributes.

R214 Information from third party sources

The policy model/language should be able to leverage information certified by a given third party (e.g., government).

R215 Technology-independent credentials

The policy model should support expressions on attributes contained in digital credentials (see also R212). Different types of credentials may be integrated in the policy model/language, such as, for instance, anonymous credentials, X.509 credentials, pseudonym/password, Kerberos tickets, etc.

R216 Attribute based access control to data on fora

The policy model/language should support policies making explicit references to attributes of involved parties (e.g., requester of access, data on which access is requested, respondent/owner of data). Attribute values can be provided by means of credentials or can be metadata associated with objects.

R217 Expiration date: there should be an option for access control policies to expire after an amount of time

Access control policies should support conditions and reasoning about time. Time can impact the validity of certain conditions in the policies or be used to support policies that might be valid only up to some time or after some time (e.g., embargo on data, data that become public after a given time, data that should be deleted after a given time).

R218 Time or event of begin of validity

Access may be granted or denied for user or groups of users after an amount of time or after an event occured (for instance to support history scientists etc.). The policy model may support event-based conditions other than those expressed as a time (see R217). Event-based conditions make policy restrictions/permissions become valid at the occurrence of certain events.

R219 Priority of policies or combination rules for policies

In case of contradicting policies we need a clear prioritization, that is a rule which determines which policy supersedes all others and how the others are combined with that policy and each other (you may think of a hierarchy of policies as well). The policy model should support a mechanism for combining policies according to different composition operators. The policy model/language should be accompanied by a clear definition of the possible composition operators and their semantics should be provided.

R220 Data Subject should be able to keep control over PII

A Data Subject should be able to control (modify, delete, etc.) his PII that has been collected and stored by a Data Controller. In this case, the Access Control of the Data Controller, which is under obligation of letting Data Subject control his PII, has to take into account obligations, i.e. let the Data Subject access his PII.

The control of the Data Subject on his PII may range from getting read access to stored PII, being allowed to update collected PII, accessing log regarding usage of collected PII, modifying sticky policy referring to collected PII, etc. For instance, a Data Subject provides his home address (PII) to a Data Collector. Subsequently, the Data Subject may use a dedicated endpoint (e.g. WS, Web page) offered by the data collector to access and check logs related to the usage of his PII in order to figure out whether his PII was used appropriately.

R221 Choose strength of protection

The policy model should provide different levels of protection and give users the ability to tune protection according to their need. For instance, requesting the application of specific cryptographic measure in communication or storage of private information.

R222 Change policy

In some scenarios, users can store/outsource their information to external servers. Access control policies can then be attached to the data to regulate their management. The users should be able to change/update those policies when needed.

R226 Support for complex claims

The rules expressed in policies can take into account complex claims, i.e. certified attributes. For instance, claims can be identity (e.g. name), role (e.g. researcher), attribute (e.g. age), context (e.g. current location), relationship (e.g. friend/employee of Z), authorization (e.g. can read file X), rule (e.g. can delegate rights), etc.

R227 Support for complex rules

Access rules should allow expressive enough operators to compare and combine attributes (e.g. age>18 AND nationality = {German OR Swiss}) as well as simple requests for information (e.g. any user requesting the service must provide his phone number)

R229 Information disclosure to third parties

The policy language should allow to express access rules that require the requestor to reveal certain information to a third-party recipient, rather than to the verifier itself. Important use cases are:

- minimal disclosure: The requestor gives his information only to those parties who need it. For example, he reveals his delivery address only to the shipping company, rather than to the webshop itself.
- anonymous attestation and identity escrow: The requestor reveals identifying information to a trusted third party so that the latter can de-anonymize the requestor in case some form of fraud is detected.

Technically this feature could be elegantly implemented using verifiable encryption, where the relevant information is encrypted under the third party's key. Optionally, a data handling policy or the release conditions could be specified in the decryption label. Alternatively, it could be implemented in a "lower-tech" fashion by letting the requestor hand over the information to the third party directly, and showing some kind of receipt to the verifier.

R230 Ontologies for credential types, delegation

Ontologies can be a powerful tool to adapt the language to the particular needs of a particular context while maintaining interoperability. For example, a common ontology on the structure of personal identity information can be used to guarantee compatibility of digital passports issued by different countries. An ontology on countries and their governments can be used to determine which instances are certified to issue passports for which countries.

4.2.3 Requirements for Trust policies

R311 Link to Data Handling policies

The policy language should provide the possibility to link a Data Handling policy with trust policies; this would permit the explicit representation of trust on the correct enforcement of the Data Handling policy. When the binding with the trust policy is not expressed, trust is implicitly assumed. The binding between Data Handling and Trust Policy could occur at different levels, expressing requirements on the data subject (e.g., an identity must be provided, verified by a trusted identity manager) or on the source of the credentials referred in the policy (e.g., the data controller has to exhibit a valid certificate issued by the Better Business Bureau).

R312 Trust establishment

Several factors should be taken into account in the decision to trust another entity or not. A first could be the exchange of credentials ("I'll show you mine if you show me yours"). Trust could also be based on statements made by others, for example reputations or privacy seals.

R316 Statement and Certification

Certification validates that a server is authentic and trustworthy, so that the user can feel confident that their interaction with the server has not been eavesdropped and that the server is who it claims to be. The certificate is provided by a third party that should be trusted by the user.

- A trust statement is the explicit expression of a perceived trust level. It is made by the truster and represents the subjective judgment of the trustee's trustworthiness, according to the truster's point of view.
- A certificate is a digital document that describes a written statement from the issuer (certification authority, often considered trustworthy) about the identity of another party (in the form of public key and the identity of the owner) or a permission to use a specific service. It can be considered as a trust statement issued by a reputed third party.

R317 Context-dependent Trust Mechanisms

Trust mechanisms should be chosen according to the application context. For example a trust policy should express a rule specifying that I provide my e-mail address to: - an online book store, if its reputation is greater than 8/10 - an online tax declaration website, if this one is certified by the government.

R318 Security breach

Security breaches come in different flavors, they include (but are not limited to):

- Loss of control on data (e.g., storage device stolen or lost)
- Unauthorized access control
- Social engineering
- Phishing
- Malicious proxy server

In some cases, legislation requires that when a privacy breach occurs, a notification has to be sent to the affected individual or organization. Technological means to enforce these rules have to be put in place, and a formal and quantitative definition of 'Privacy breach' assess to evaluate when the breach occurs and what is the level of risk associated.

R319 Link trust with Access Control

Conditions on policy may include trust evaluation, e.g. allow data writing if user complies with ACP and trust level greater than X or he has a certain certification.

R320 End user trust

Trust affects all levels of end-user interaction with the system, i.e. whenever a user wants to access a service on the web. Trust should be assessed for all the layers involved in the transaction: user application, network, service provider. Trust being by definition related to a personal perception, each user has to able to edit trust policies and trust preferences in an intuitive way.

R321 Building trust through a third party

Users may establish trust relationships using third party trust assessment. This may guarantee maximally a level of trust equal to the level of the certification authorities

(best case scenario). Trust mechanisms have to support certificates as produced by certificate authorities (e.g., CAcert, Thawte, etc.) and the corresponding hierarchical mechanisms (web of trust). Trust reasoning has to allow to combine this information with other trust metrics (e.g. reputation based).

R322 Trust reasoning

A trust policy evaluation component should be able to reason about trust, including composing various trust metrics (e.g. reputation system, PKI ...) and hierarchical structures. E.g. A can assess level of trust on B, combing information from a reputation system, certificates provided by B (or by a third party for B), measuring some indicators related to B (e.g., recent similar transactions). What kind of mechanisms could be used, and how to combine them (e.g., the relative weights in the combination) have to be context dependent, as well as modifiable by the user.

R324 Trust ontologies

Trust credentials and trust assessment mechanisms should be represented in an ontology. This ontology should categorize the different types and sub-types of credentials (for example *passport*, *ID* or *driving license* are subsets of the type *personal identifiers*). For each type of credential we can attach an information about the trust assessment mechanism supporting it.

R323 Transparency, reciprocity

Transparency should be considered as one component of trust assessment (typically, transparency increases trust perception). If a data holder is able to monitor at any time the usage of his data by a server, his trust feeling will increase. Due to the fact that transparency techniques may include: historical data, previous behaviors, access to log files, etc. This requirement is strongly related to *R122: Logging/Monitoring/Auditing Policies*. Reciprocity should be possibly taken into account as often characterised trust interaction, but this is not the general case, e.g. I trust a mail provider for storing my personal mails but it does not necessarily trust me for storing the same kind of information.

R329 Specification of liabilities

- Towards data subject: data protection obligations under the 95/46 Directive have to be fulfilled by data controllers. Data controllers are liable for data protection violations unless they can prove they are not responsible for the damage. It is necessary to differentiate between data controller and data processor. The role of data processor is reduced; he solely processes personal data as directed by the controller. The policy language should be able to express the role of each entity for each action to determine who is liable. Liability: compensation from the controller for the damage suffered. Remedy: a right to a judicial remedy for any breach of guaranteed rights. Sanctions: to be defined by member states in their national laws.
- Towards relaying parties: for ensuring data accuracy the following policies are important: validation of data at the moment of collection, procedures for reporting and dealing with suspected inaccuracies, regular updates, restriction of modification rights to authorized entities.

4.3 Requirements for Policy Composition

These are particular requirements when composing different policies. "Composition" can be understood within a larger scope than service composition, i.e. any kind of interaction between policies may imply requirements listed here.

4.3.1 Composition of Data Handling policies

RC11 Prior agreement and contract

Technical policies generally implement legal requirements (contract, directive, agreement, etc.). The technical policy should contain a pointer to or the full text of the source document (i.e. legal requirements). Regarding privacy policies this requirement is addressed in more detail in R113.

RC12 Support for composition of service policies and composition of user preferences

The language(s) used to define user's data handling preferences and service's data handling policies should support composition. The result of composing service policies is one policy that describes the behavior of a set of services. In other words, this composite policy will not be violated by enforcing individual policies. The result of composing user preferences is one preference that describes how a set of personal data can be handled. In other words, matching such composite preference implies matching all individual preferences.

When composition can be automated, mechanisms to specify a composite policy for a set of services exposing individual policies and mechanisms to define a composite preference for a set of personal data with associated preferences would be valuable.

RC13a Cascading policies

When rules are defined at different levels (e.g. corporate, service, and action), mechanisms to select and aggregate appropriated rules should be provided.

RC13b Prioritization of rules

Priorities are only necessary to resolve conflicts between rules. Depending on the expressiveness of the language, priorities may be required.

RC14 Generalization of policies

Composite services rely on external services. Policies of composite services depend on properties of external services, which may be dynamically selected. The policy language should support such dynamic scenarios and offer a way to express general statements. For instance, instead of saying that "personal data could be shared with bank X, which deletes data after 6 months", the policy could state that "personal data could be shared with any bank in Europe that commits to delete before one year". Similarly, generalization of user preferences should be envisioned.

RC15a Multi-rounds policy definition

When associating a policy with some personal data, purely service-driven (e.g. P3P) or purely user-driven (e.g. arbitrary policy pushed by the user to the service) approaches do not work. A tradeoff letting the user customize some optional parts of a template should be possible. Such a mechanism already exists in PRIME-DHP. Multi-rounds could be added to let the service call back the user when an optional part was deliberately let open.

RC15b Policy negotiation

Negotiation only makes sense when the user and/or the service have a trade-off to make. (Without a trade-off, the optimal choice for both parties is deterministic and can be automatically deduced.) For instance, quality of service (features, speed, etc.) or service fees could change depending on the privacy (quantity and/or quality of released personal information). The policy language should be able to express what the trade-offs are.

4.3.2 Composition of Access Control policies

RC21 Delegation of Rights

Service providers (e.g. data owner) need a way to grant access rights to other parties (e.g. A says B can read dataZ). Delegation of rights is the ability to let another party provide rights (e.g. A says B can say x can read dataZ, B says C can read dataZ). This is a key feature of SPKI and SecPAL that enables complex distributed access control settings. As a special case, the policy language should support delegation of credential issuing rights. For example, the OECD can delegate the right to issue digital passports to national ID agencies.

RC22 Revocation of Rights

A party providing rights must have a way to revoke them. Depending on the required time granularity, this can be achieved by periodically deciding to renew rights or by explicitly revoking (e.g. Certificate/Credential Revocation List).

RC23 Composition of Access Control Policies

The policy language should enable the creation of composite applications relying on different services. For instance, Single Sign On, federations, or delegation of rights should be in place to enable coherent enforcement of authentication and authorization policies associated with different services. See also requirement RC12 on composition of DHPs.

RC24 Prior agreement and contracts

Similar to RC11, applied to access control policies.

RC25 Privacy-aware audit mechanism

Access control policy language must offer support for logging mechanisms. The policy must offer mechanisms to specify privacy aspects of logs. For instance, it should be possible to specify that denied accesses are logged for statistics but that the identity of the requestor is not part of the log.

RC26 Support for data and PII

Legislations treat personal data (PII) differently from other types of data (e.g. internal notes, trade secret). The developed policy language must support specific requirements of PII (e.g. purpose). This may be solved by more generic mechanisms.

RC27 Link between AC and DH

Data Handling policy of the data owner impacts Access Control policy of the data consumer. In other words, let's assume that A sends some PII to B with attached data handling sticky policy. If B let's C access the PII provided by A, the access control of B regarding this PII must reflect A's sticky policy. This is already illustrated in PRIME-DHP.

4.3.3 Composition and Trust policies

RC31 Dynamic Trust

Trust is not static: mechanisms to bootstrap, modify, and revoke trust are necessary.

RC32 Scope

Trust is not unconditional. The scope of the trust relationship has to be defined. A user will trust a medical association in the scope of health but not in the scope of finance.

RC33 Proof of enforcement

Trust aims at deciding whether a party will behave as expected. In the specific case of privacy and data handling policies, trust is necessary to decide whether a party will handle PII as specified in the data handling policy. Mechanisms to prove that a trusted platform (e.g. TCG) and/or audit mechanisms are in place are required.

4.4 Requirements for use of anonymous credentials

RP01 Technology-independent certification of data by trusted third parties

The policy language should be able to express access, trust, and data handling requirements over information that was certified by an external, trusted third party. The language should depend as little as possible however on exactly which underlying technology is used to certify this information. That is, the information could be certified by a trusted LDAP server (username/password), Kerberos tickets (symmetric crypto), X.509 certificates (asymmetric crypto), anonymous credentials (= advanced asymmetric crypto), … The policy language should make abstraction of the underlying technology and speak in terms of general concepts that are common to all technologies. Yet still, the language should be expressive enough to fully leverage the different capabilities of each of the technologies.

RP02 Trust in certified data

The policy language needs some elements to state what the requirements on a piece of data are in order to trust the authenticity of this piece of data. This can be done by stating who is trusted to certify this kind of data, i.e. by stating who the allowed issuer(s) of a certificate for this data are. For example, if a policy states that one has to reveal her last name, then the policy also needs to specify who is trusted to certify this last name (e.g., this might be a government, a city authority, etc).

RP03 Predicates over attributes, extensible with ontologies

The ACP should allow to express complex predicates over attributes, including AND/OR combinations, equality, greater/less-than, simple arithmetic, … These predicates should be extensible with new ontologies too. For example, if frequent flyer statuses are such that platinum is higher than gold, then the policy should be able to express ‘status>gold’.

RP04 Embedded or referenced ontologies

It should be possible to have the ontologies that are used in a policy contained in the policy itself. An alternatively approach would be to reference to the ontologies, i.e., to point to some resource/location where the ontology can be obtained. However, the approach of referencing might lead to problems during an online policy evaluation, if the ontology for any reason cannot be obtained.

RP05 Expression of proved statement (by using same policy language)

The user should also be able to express the statement that he is proving, preferably using the same language. This statement may be different from the statement required by the server, but it should imply it. The server's policy engine needs to be intelligent enough to check whether the statement proved by the user actually implies the access conditions.

RP06 Derived PII

One issue needing more thought is how one defines the DHP of derived PII. With derived PII we mean sensitive information that is computed based on actual PII. For example, one could associate a DHP to one's exact date of birth, but what then is the DHP for just your year of birth, your zodiac sign, or the mere fact that you're over 21? Are these bound to the same DHP? It seems natural to loosen the DHP as the quality of the information degrades, but quantifying this is highly non-trivial. For example, one could be tempted to say that by proving you're over 21 you give 1 bit of information away, but it's actually more complicated than that: by proving you're over 0 you give no information away, by proving you're over 115 you give quite a lot away since the number of people over 115 is probably very low. And what is the DHP on PII that is derived from multiple pieces of PII? A conservative solution would be to let the DHP of the derived information be the "intersection" of the individual DHPs, meaning allowing only those actions that are allowed by all DHPs.

RP07 Revealing of data

The policy language should clearly distinguish between, on the one hand, the PII that the requestor has to *reveal* in order to gain access to a service, and, on the other hand, the *restrictions* that the PII have to satisfy to be granted access. This difference is often neglected in existing languages, probably because the only way to prove a statement about an attribute using "classical" technologies is by revealing its exact value. Anonymous credentials, however, do allow to keep these very different types of access requirements separate, with great privacy benefits for the requestor.

To illustrate this difference, think of a service where users have to be over 18 and submit their phone number. The service is not interested in the exact value of the user's birhdate; rather, it wants to ensure the mere fact that it is longer than 18 years ago. On the other hand, the server does want to know the full telephone number, but there are no conditions associated to the number that could influence the access decision (other than the number being valid, perhaps).

RP08 Alternative data recipient + associated access conditions

It should also be possible to express the recipient of the information, i.e. to whom this information should be sent. By default this is probably the server itself, but in certain applications the server may want to be assured that the client sent some specific piece of information (e.g., her address for shipping) to someone else. When the recipient is not the server itself but a third party, then it should also be specified under what conditions the third party should reveal the information. This can probably be implemented as an ACP on the transmitted information.

RP09 Notion of atomic credentials

The language should support the concept of an atomic credential (or information card, or whatever one wants to call it) as a collection of certified attributes. This is important to prevent "mixing-and-matching" of attributes from several credentials in cases one has multiple credentials of the same type and by the same issuer. For example, if I have two credit cards, then I shouldn't be able to complete the checkout transaction by revealing the number of one card and the expiration date of the other card.

RP10 Macros

The policy language should have some sort of macro language to increase readability. For example, in the buying use case, the macros ‘anygovt’ and ‘goodbusiness’ act as easy links from the access control part into the trust part of the policy. In general, use of macros can avoid repetition throughout the policy, and make the policy layout more modular. In particular, it provides an easy way to group credential issuers into classes.

RP11 Limited spending

The policy language should have provisions to express that one can only authenticate oneself with the same credential for a limited number of times. There are different ways to calculate the number of shows: overall, at this website, at this service, on this day, etc. However, it is not clear whether the number of times a credential has been shown is an attribute of a credential or not, since it might depend on some “context” information like e.g. the time of day, the weather, the price of a stock, etc.

RP12 Alternative DHP

It might be the case that a party does/can not accept the DHP of another party. For example, the party has not the technical means to fulfill the required obligations. The policy language should then be able to express an alternative DHP that it is willing/able to fulfill.

RP13 Choice based Access Control Policy

The policy language should allow for making access control requirements on a resource dependent on (multiple) choices the user can make. For example consider a credit card application, where the access control requirements depend on choosing the monthly limit the credit card shall have. When choosing a limit of 2000 Euro, different access control requirements apply than with a choice of 5000 Euro.

RP14 Signing statements

The policy language should be able to express that a particular statement needs to be signed before access to a resource is granted. It should also be possible to specify using which credential or combination of credentials this statement should be signed. "Signing" is not to be interpreted in the strict sense of public-key cryptography, it could e.g. also be implemented by the user clicking an OK button. For the specific case of signing using anonymous credentials, the signature reveals that someone with the specified list of attributes has signed, without disclosing who exactly signed however.

Chapter 5

State of the art

Technical improvements of Web technologies have fostered the development of online applications that use private information of users to offer enhanced services. As a consequence, the vast amount of personal information thus available on the Web has led to growing concerns about user privacy. A number of projects and research works on access control, privacy, and identity management have been presented in the last years, although not many of them have addressed the issue of protecting the user privacy.

5.1 Privacy-enhanced policy languages

In this context, the research community has devoted huge efforts in the study and specification of policy languages for regulating access in distributed scenarios. Some work has also focused on the definition of privacy-aware languages XACML [XACML20], WS Policy [WSPol], EPAL [EPAL11], and P3P [P3P11] that support preliminary solutions to the privacy protection issue, for instance by providing functionality for controlling secondary use of data (i.e., how personal information could be managed once collected).

Extensible Access Control Markup Language (XACML), which is the result of OASIS standardization effort, proposes an XML-based language to express and interchange access control policies. In addition to the language, XACML defines both an architecture for the evaluation of policies and a communication protocol for message exchange.

Platform for Privacy Preferences Project (P3P) is a World Wide Web Consortium (W3C) project that addresses the need of a user to assess whether the privacy practices adopted by a server provider comply with her privacy preferences. P3P provides an XML-based language and a mechanism ensuring that users can be informed about privacy policies of the server before the release of personal information. The corresponding language that would allow users to specify their preferences as a set of preference rules is called *A P3P Preference Exchange Language* APPEL [APPEL10]. APPEL can be used by users agents to make automated or semi-automated decisions regarding the acceptability of machine-readable privacy policies from P3P enabled Web sites. PReference Expression for Privacy (PREP) [AL 2005] is another XML-based privacy preference expression language used for representing

the user's privacy preferences within Liberty Alliance, a project that aims to produce standards, guidelines and best practices for identity management.

Privacy Preferences Expression Language PPEL [PPEL] provides a solution to manage privacy preferences within Liberty Alliance's ID-WSF Web Services Framework Liberty [Liberty], a basic framework that makes available a set of identity services, such as identity service discovery and invocation. PPEL gives a high-level example based on P3P of how privacy preferences can be managed in the communication between a Service Provider and Web Services Provider (acting on the Principal's behalf). In this context, where a multi-leveled policy approach is adopted, a limited, hierarchical set of privacy policies is used to describe the privacy practices of a Service Provider, and the privacy preferences of a Principal.

Enterprise Privacy Authorization Language EPAL [EPAL11] is an XML-based markup language and architecture for formalizing, defining, and enforcing enterprise-internal privacy policies. It addresses the problem on the server side and supports the need of a company to specify access control policies, with reference to attributes/properties of the requester, to protect private information of its users. EPAL is designed to enable organizations to translate their privacy policies into IT control statements and to enforce policies that may be declared and communicated in P3P. XACML, however, includes most (if not all) of the expressive power of EPAL. Additionally, EPAL does not allow protection of user data released outside enterprises boundaries.

Looking at standardisation efforts, Web Services specifications stand out. In particular, WS-* IBM/MS WS Roadmap [IBMMSWS] is focused on a strategy for addressing security in a Web Service environment and provides several specifications supporting security functionality at different levels of abstraction. It defines a comprehensive Web service security model that supports, integrates, and unifies different security models, mechanisms, and technologies to enables a variety of systems to securely interoperate. WS-* is a set of standards and specifications on Web Services. While security is a key aspect in WS-Security, WS-Trust, WS-Federation, WS-SecurityPolicy, and WS-SecureConversation, privacy remains under-represented. For instance, there is no support for anonymous credentials in WS-Security (and XML Digital Signature). Most aspects of the former WS-Privacy are now part of WS-Federation: pseudonym services are defined, it is recognized that attributes may be very sensitive and thus that user consent may be required (or that attributes may be encrypted). Moreover, the privacy statements of relying parties (i.e. data controllers) are mentioned. A placeholder for privacy statement is even proposed but their definition remains out of scope. Finally, the OASIS Identity Metasystem Interoperability committee defines privacy aspects mainly related to user control through delegation selectors.

SecPAL ([BFG 2006],[BFG 2007]) is a declarative security policy language developed to meet the access control requirements of large-scale Grid Computing Environments. It is a declarative, logic-based language that builds on a large body of work showing the value of such languages for flexibly expressing security policies. It is designed to be comprehensive and provides a uniform mechanism for expressing trust relationships, authorization policies, delegation policies, identity and attribute assertions, capability assertions, revocations, and audit requirements. This provides tangible benefits by making the system understandable and analyzable. It also improves security assurance by avoiding, or at least severely curtailing, the need for semantic translation and reconciliation between disparate security technologies.

The works closest to the one in PrimeLife and in this deliverable are represented by the PRIME Languages [PRIME], XACML, P3P, and SecPAL. PRIME languages represent a good starting point and an inspiration for the work to be done in PrimeLife. XACML

represents the most accepted, complete, and flexible solution in terms of access control languages, which could be adapted and integrated with privacy preference-based solutions. P3P represents the most important work done in the context of privacy protection and secondary use management. SecPAL only focuses on authorization and does not support data handling. However, two key features make us think that SecPAL is a good candidate for data handling. First, SecPAL is used to specify claims as well as access control policies. Having one language for both aspects is an important step towards expressing privacy policies and user preferences in a single language and thus avoiding matching issues (e.g. P3P with APPEL). Second, the language is based on logic (DataLog with constraints) that enables safe authorization queries and abduction [BMD 2009]. Extending SecPAL to support data handling may be an interesting option.

Below, we describe these four proposals in more details.

5.1.1 PRIME languages

The PRIME project [PRIME] aims at developing an identity management system able to protect users' personal information. The goal of the PRIME project is to provide the users with solutions allowing them to keep their autonomy and to retain control over their personal information in interactions with other parties. In the PRIME vision, the user should have control of personal information and negotiate its disclosure for access to a service. The result of such a negotiation is an agreement between the user and the service provider, whereby the provider collects personal data for a stated, legitimate purpose. The PRIME reference scenario is a distributed infrastructure that includes: *i) users* (data subject), human entities that request online services; *ii) service provider* (data collector), the entity that provides online services to the users and collects personal information before granting access to its services; *iii) external parties* (downstream data collector), entities (e.g., business partners) with whom the service provider may want to share or trade personal information of users.

The service provider collects personal data that are necessary to provide access to services and stores them into *profiles* associated with each user. A profile can therefore be seen as a container of pairs of the form <*attribute_name*,*attribute_value*>, where *attribute_name* is the name of the attribute provided by the user and *attribute_value* its value. The set of message exchanges between a user and a service provider is called *negotiation* [BS 2002]. A negotiation always starts with a user request to a service provider and ends explicitly (done or stop) or implicitly (e.g. assumed after a certain timeout period). During a negotiation a user may, like a service provider, require the counterpart to fulfill some requirements (i.e., provide information) for it to proceed. In others words, a bidirectional negotiation is considered where both parties can require the other party to provide them with certain information or digital certificates necessary for service request or fulfillment. Each party has a *portfolio of credentials* (digital certificates) and *declarations* (unsigned data). Access to services and release of portfolio information are managed according to the rules specified by the parties. In particular, at service provider side a set of rules (i.e., access control policies) regulates service access, defining credentials and declarations that users must submit to gain access. At both the user and the service provider sides, rules (i.e., release policies) manage the release of personal information that may be requested from the counterpart. Such release policies may also allow release of information only if the counterpart presents some credentials and declarations. For instance, the service provider's access control policies may require users to submit a credit card number and contact information to complete a purchase. Since the user's release policies state that contact information can be released to other parties without any constraints and credit card information can be released only to parties that are members of the Better Business Bureau (BBB), the online transaction can be completed only if the service

provider is able to prove that it belongs to BBB. Anonymous communications are assumed to be in place. Therefore, at the beginning of a negotiation, the user is unknown to the service provider, meaning that no information about the user has been collected by the service provider. The above discussion is still valid when an external party wants to access personal information of the users stored at the service provider. The only difference is that, in this case, the service provider is responsible for protecting the privacy of user data.

In this context an important service for helping users to keep control over their personal information is represented by an access control solution enriched with the ability to support privacy requirements. More specifically, providing the users with the control of their personal data and permitting anonymous interactions are some of the main goals of this project. Another goal is to define privacy policies governing the system usage to provide a privacy-aware access control system. The policies should establish how to use the system, and should allow to define trust relationships, privacy preferences, and authorization rules. A privacy-aware access control system is thus a framework that, in addition to traditional access control functionality, guarantees on the one side the privacy of information exchanged between different parties and, on the other side, provides a solution to control the secondary use of information disclosed in the context of an access control process.

To fully address the requirements posed by a privacy-aware access control system, the following different types of privacy policies have been defined in the context of PRIME.

Access control and release policies

Access control policies define authorization rules concerning access to data/services [SD 2001]. Authorizations correspond to traditional (positive) rules usually enforced in access control systems. For instance, an authorization rule can require a user of age X and a credit card number (condition) to read (action) a specific set of data (object). When an access request is submitted to a service provider, it is evaluated against the authorization rules applicable to it. If the conditions for the required access are evaluated to true, access is permitted. If none of the specified conditions that might grant the requested access can be fulfilled, access is denied. Finally, if the current information is insufficient to determine whether the access request can be granted or denied, additional information is needed, and the requester receives a list of requests that she must fulfil to gain access.

Release policies instead define the party's preferences regarding the release of its PII by specifying to which party, for which purpose/action, and under which conditions a particular set of PII can be released [BS 2002]. For instance, a release policy can state that credit card information can be released only in the process of a *purchase* and to a party member of Better Business Bureau (BBB). The release of PII may only be performed if the release policies are satisfied.

The main functionality offered by PRIME access control model and languages are then summarized as follow.

- **Attribute-based restrictions** The language supports the definition of powerful and expressive policies based on properties (attributes) associated with subjects (e.g., name, address, occupation) and objects (e.g. owner, creation date). The language includes some operators for comparing attribute values and could be extended by adding nonstandard functions.
- **XML-based syntax** The language provides an XML-based syntax for the definition of powerful and interoperable access control and release policies.

- **Credential definition and integration** The language supports requests for certified data, issued and signed by authorities trusted for making the statement, and uncertified data, signed by the owner itself.
- **Anonymous credentials support** The language supports definition of conditions that can be satisfied by means of zero-knowledge proof [CL 2001].
- **Support for context-based conditions** The language allows the definition of conditions based on physical position of the users [ACDS 2006] and context information, and integration with metadata identifying and possibly describing entities of interest, such as subjects and objects, as well as any ambient parameter concerning the technological and cultural environment where a transaction takes place.
- **Ontology integration** Policy definition is fully integrated with subject and object ontologies in defining access control restrictions. Also, the language takes advantages from the integration with credentials ontology that represents relationships among attributes and credentials.

Access control and release policies share the same syntax and are composed by one or more rules, combined in OR logic between them, directly associated with an object component and the related set of actions. Syntactically, an access control policy (release policy, resp.) is an expression of the form *<actions> ON <object> with <object_expression> if <rules>*, where:

- *actions* is the set of actions to which the rules refer (e.g. read, write);
- *object* identifies the object to which the rules refer and corresponds to an object identifier or a named abstraction of values, if abstractions are defined on objects;
- *object_expression* is a boolean expression that allows the reference to a set of objects depending on whether they satisfy given conditions that can be evaluated on the object's profile;
- *rules* is a set of rules defined as follows.

An access control rule (release rule, resp.) represents the basic element used to regulate the access to the objects with which it is associated. Syntactically, an access control rule (release rule, resp.) is an expression of the form *<subject> with <subject_expression> can <actions> for <purposes> if <conditions>*, where:

- *subject* identifies the subject to which the rule refers and corresponds to a user identifier or a named abstraction of values, if abstractions are defined on subjects;
- *subject_expression* is a boolean expression that allows the reference to a set of subjects depending on whether they satisfy given conditions that can be evaluated on the user's profile;
- *actions* is the set of actions to which the rule refers (e.g., read, write, and so on). Note that, at rule level, the actions field can only be a specialization of the actions field defined at policy level;
- *purposes* is the purpose or a group thereof to which the rule refers and represents how the data is going to be used by the recipient;
- *conditions* is a boolean expression of conditions that an access request to which the rule applies has to satisfy.

The following example illustrates a PRIME Access Control Policy where each user can *read* his own credit card for purpose *access*. It is important to note that an element *evidence* is added to the policy. This element allows to define restrictions on the type of certification used in the release of the information.

```
<Q1:policy authors="UNIMI" id="MyId"
   type="http://www.prime-project.eu/policies/access"
   xmlns:Q1="http://www.prime-project.eu/policies/"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns="http://www.w3.org/2000/09/xmldsig"
   xmlns:cr="https://www.prime-project.eu/ont/XSD-ClaimRequest"
   xsi:schemaLocation="http://www.prime-project.eu/policies/policyschema.xsd">
 <Q1:object>#creditcard</Q1:object>
  <Q1:objectExpressions>
   <Q1:condition name="http://www.prime-project.eu/predicates/exist"
                 sanitization="false">
      <Q1:argument isLiteral="false">
                   http://www.prime-project.eu/credentials/name.last
      </Q1:argument>
   </Q1:condition>
  </Q1:objectExpressions>
  <Q1:actions>
    <Q1:action>http://www.prime-project.eu/policy/2006/07/14/read</Q1:action>
    <Q1:action>http://www.prime-project.eu/policy/2006/07/14/write</Q1:action>
  </Q1:actions>
  <Q1:rules>
    <Q1:rule id="r1">
     <Q1:subject>http://www.prime-project.eu/policy/2006/07/14/you</Q1:subject>
     <Q1:actions>
       <Q1:action>http://www.prime-project.eu/policy/2006/07/14/read</Q1:action>
     </Q1:actions>
     <Q1:purposes>
       <Q1:purpose>http://www.prime-project.eu/policy/2006/07/14/access</Q1:purpose>
     </Q1:purposes>
     <Q1:conditions>
      <Q1:subjectExprs>
        <Q1:group name="se1">
          <Q1:condition name="http://www.prime-project.eu/predicates/equal"
                        sanitization="false">
            <Q1:argument isLiteral="false">
                         http://www.prime-project.eu/credentials/name.last
            </Q1:argument>
            <Q1:argument isLiteral="false" context="object">
                         http://www.prime-project.eu/credentials/name.last
            </Q1:argument>
          </Q1:condition>
          <Q1:evidence/>
        </Q1:group>
      </Q1:subjectExprs>
     </Q1:conditions>
    </Q1:rule>
  </Q1:rules>
</Q1:policy>
```

Data handling policies

Data handling policies (DHPs, for short) provide the users with the possibility to define how their PII should be subsequently used by the receiving parties (e.g., information collected through an online service may be combined with information gathered by other services for commercial purposes). Users make use of these policies to define restrictions on secondary use of their personal information. In this way, users can manage the information also after its release.

In the data handling policy specification, two issues have been discussed: by whom and how a policy is defined. With respect to the first issue, when a user requires a service, a predefined policy template is provided by the service provider as a starting point for creating data handling policies. The template is customized to meet different privacy requirements. A user can directly customize the template or it can be supported by a customization process that automatically applies some user privacy preferences. If the customized data handling policy is accepted by the service provider, the personal information provided by the user is labeled

with it. Data handling policies are attached to the PII or data they protect, and transferred as *sticky policies* to the counterparts. This represents the most flexible and balanced strategy for the definition of a data handling policy. With respect to the second issue (i.e., how a DHP is defined), data handling policies are defined as independent rules. These rules also define privacy obligation policies that will be managed independently from access control policies.

As for access control, data handling language provides an XML-based syntax, attribute-based restrictions, and ontology integration. Syntactically, a data handling policy has the form <*PII*> *managedBy* <*DHP_rules*>, where: *PII* identifies a PII abstraction that represents the name of an attribute or the name of a data type, in case of a data handling policy template, a set of pairs of the form <*attribute_name*,*attribute_value*> (possibly represented by an abstraction) belonging to a privacy profile, in case of a customized data handling policy; and *DHP_rules* identifies one or more rules, composed in OR logic, governing the use of PII to which they refer. Syntactically, *DHP_rules* is an expression of the form <*recipients*> CAN <*actions*> FOR <*purposes*> [IF <*gen_conditions*>] [PROVIDED <*prov*>] [FOLLOW <*obl*>], where:

- *recipients* can be an identifier, a category, or a boolean expression;
- *actions* is the set of actions;
- *purposes* is the purpose or a group thereof;
- *provisions*, *obligations*, and *generic conditions* are optional boolean expressions of terms.

A data handling rule specifies that *recipients* can execute *actions* on *PII* for *purposes* provided that *provisions* is satisfied, *generic conditions* is satisfied, and with *obligations*.

The following example illustrates a PRIME DHP policy where *John Doe* (recipient) can perform *any* action on the credit card data (PII) under purpose *access*. Disputes will be managed by *ACME director*.

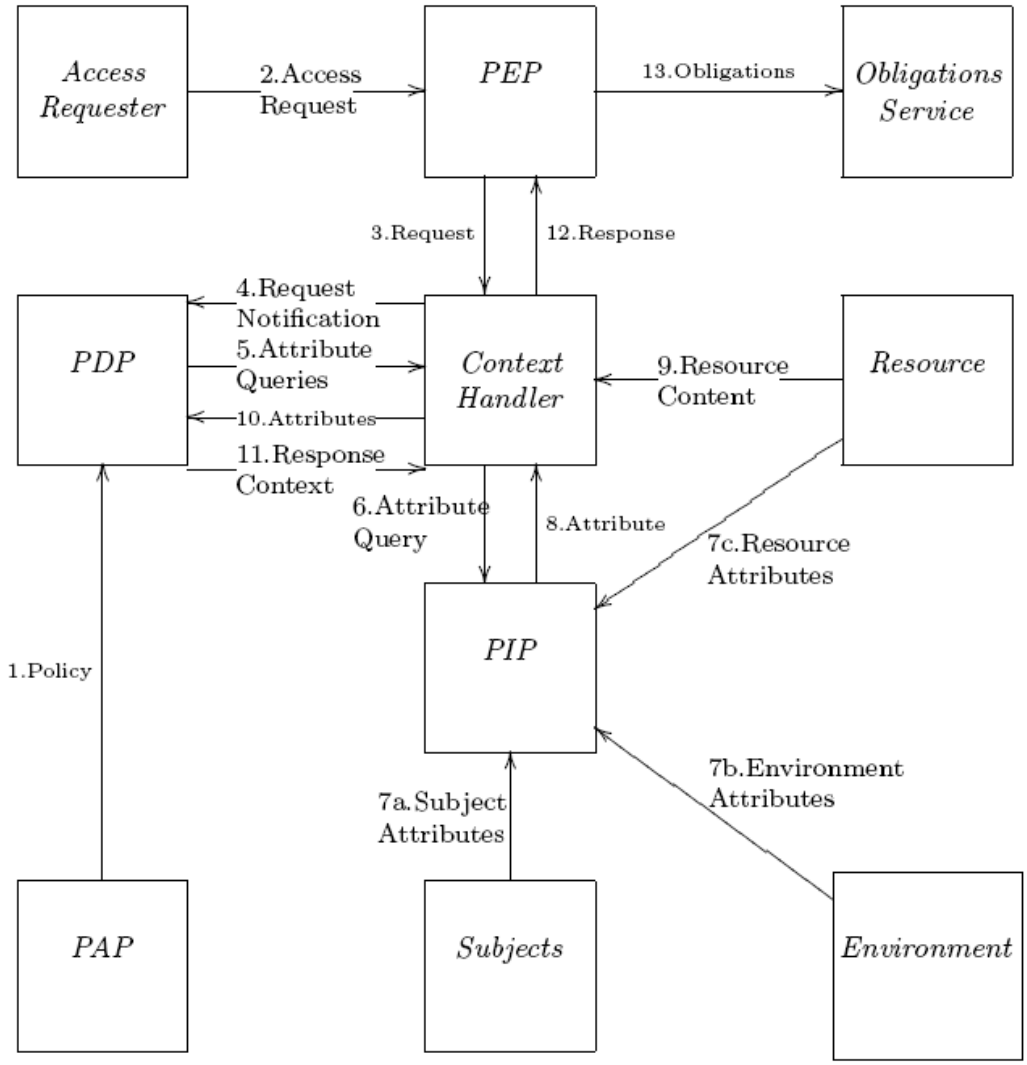
```
<DHP id="MyId" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xmlns="http://www.prime-project.eu/datahandling/">
    <PII>http://www.prime-project.eu/policy/2006/07/14/creditcard</PII>
    <actions>
      <action>any</action>
    </actions>
    <purposes>
      <purpose name="http://www.prime-project.eu/policy/2006/07/14/access"
               type="optional">
           Access
      </purpose>
    </purposes>
    <recipients>
      <recipient subject="http://www.prime-project.eu/policy/2006/07/14/you"
                 type="required">
        <condition name="http://www.prime-project.eu/predicates/equal">
          <argument isLiteral="false">
                  http://www.prime-project.eu/credentials/name.given</argument>
          <argument isLiteral="true">John</argument>
        </condition>
        <condition name="http://www.prime-project.eu/predicates/equal">
          <argument isLiteral="false">
                  http://www.prime-project.eu/credentials/name.last</argument>
          <argument isLiteral="true">Doe</argument>
        </condition>
      </recipient>
    </recipients>
    <obligations />
    <disputes>
      <dispute thirdParty="ACME">
        <remedies>Ask to the ACME director</remedies>
```

```
        <description />
      </dispute>
    </disputes>
</DHP>
```

5.1.2 Extensible Access Control Markup Language (XACML)

XACML [XACML20] is an XML-based language used to express and interchange access control policies. XACML is designed to express authorization policies in XML against objects that are themselves identified in XML. The language can represent the functionality of most policy representation mechanisms and has standard extension points for defining new functions, data types, policy combination logic, and so on. In addition to the language, XACML defines both an architecture for the evaluation of policies and a communication protocol for messages interchange. The major functionality offered by XACML can be summarized as follows:

- **Policy combination** XACML provides a method for combining policies specified independently. Different entities can thus define their policies on the same resource. When an access request on that resource is submitted, the system has to take into consideration all these policies.
- **Combining algorithms** Since XACML supports the definition of policies specified independently, there is the need for a method for reconciling such policies when their evaluation is contradictory. XACML supports different combining algorithms, each representing a way of combining multiple decisions into a single decision.
- **Attribute-based restrictions** XACML supports the definition of policies based on properties (attributes) associated with subjects and resources rather than their identities. This allows the definition of powerful policies based on generic properties associated with subjects (e.g., name, address, occupation) and resources. XACML includes some built-in operators for comparing attribute values and provides a method of adding non-standard functions.
- **Multiple subjects** XACML allows the definition of more than one subject relevant to a decision request.
- **Policy distribution** Policies can be defined by different parties and enforced at different enforcement points. Also, XACML allows one policy to contain or refer to another.
- **Implementation independence** XACML provides an abstraction layer that isolates the policy-writer from the implementation details. This means that different implementations should operate in a consistent way, regardless of the implementation itself.
- **Obligations** XACML provides a method for specifying some actions, called obligations, which must be fulfilled in conjunction with the policy enforcement.



As for PRIME, a typical scenario involving XACML is when someone wants to perform an action on a resource. For instance, suppose that a physician wants to access a patient's record for inquiry only. The physician would log on to the hospital information system, enter the patient identifier, and retrieve the corresponding record. Data flow through a XACML model can be summarized as follow (see the entities involved and the data flow in the figure above).

- The requester sends an access request to the *policy evaluation point* (PEP) module (Step 2), which has to enforce the access decision that will be taken by the policy decision point.
- The PEP module sends the access request to the *context handler* that translates the original request into a canonical format, called XACML request context (Step 3).
- The context handler sends the XACML request to the *policy decision point* (PDP) (Step 4).
- The PDP identifies the applicable policies among the ones stored at the *policy administration point* (PAP) module. It then retrieves the attributes required for the evaluation through the context handler. If some attributes are missing, the context handler queries the *policy information point* (PIP) module for collecting them (Steps 5-10). The PIP provides attribute values about the *subject*, *resource*, and *action*. To this purpose, PIP interacts with the *subjects*, *resource* and *environment* modules. The *environment* module provides a set of attributes that are relevant to take an

authorization decision and are independent of a particular *subject*, *resource*, and *action*.

- The PDP evaluates the policies against the retrieved attributes, and returns the XACML *response context* to the Context Handler (Step 11). The context handler translates the XACML response context to the native format of the PEP and returns it to the PEP together with an optional set of obligations (Step 12).
- The PEP fulfills the obligations (Step 13) and, if the access is permitted, it performs the access. Otherwise, the PEP denies access.

As described above, XACML defines a canonical form of the request/response managed by the PDP, allowing policy definition and analysis without taking into account application environment details. Any implementation has to translate the attribute representations in the application environment (e.g., SAML, .NET, Corba) into the XACML context. For instance, an application can provide a SAML [SAML11] message that includes a set of attributes characterizing the subject making the access request. This message has to be converted to the XACML canonical form and, analogously, the XACML decision has then to be converted to the SAML format.

Policy Set, Policy and Rule

XACML relies on a model that provides a **formal** representation of the access control policy and its working. This modeling phase is essential to ensure a clear and unambiguous language, which could otherwise be subject to different interpretations and uses. The main concepts of interest in XACML model are *rule*, *policy* and *policy set*.

A XACML policy has, as root element, either *Policy* or *PolicySet*. A *PolicySet* is a collection of *Policy* or *PolicySet* elements. A XACML policy consists of a *target*, a set of *rules*, an optional set of *obligations* and a *rule combining algorithm*. A *target* basically consists of a simplified set of conditions for the *subject*, *resource* and *action* that must be satisfied for a policy to be applicable to a given request. If all the conditions of a *target* are satisfied, then its associated *Policy* (or *PolicySet*) applies to the request. If a policy applies to all entities of a given type, that is, all subjects, actions, or resources, an empty element, named ***AnySubject***, ***AnyAction***, ***AnyResource***, respectively, is used. The components of a *Rule* are a *target*, an *effect*, and a *condition*. The target defines the set of resources, subjects, and actions to which the rule is intended to apply. The effect of the rule can be ***permit*** or ***deny***. The condition represents a boolean expression that may further refine the applicability of the rule. Note that the *target* element is an optional element: a rule with no target applies to all possible requests. An *Obligation* specifies an action that has to be performed in conjunction with the enforcement of an authorization decision. For instance, an obligation can state that all accesses to medical data have to be logged. Note that only policies that are evaluated to ***permit*** or ***deny*** can return obligations. Each *Policy* also defines a *rule combining algorithm* (i.e. ***RuleCombiningAlgId***) used for reconciling the decisions each rule makes. The final decision value, called *the authorization decision*, inserted in the XACML context by the PDP is the value of the policy as defined by the rule combining algorithm. XACML defines different combining algorithms, i.e., ***deny overrides***, ***permit overrides***, ***first applicable***, ***only-one-applicable***. According to the selected combining algorithm, the authorization decision returned to the PEP can be ***permit***, ***deny***, ***not applicable*** (when no applicable policies or rules can be found), or ***indeterminate*** (when some errors occurred during the access control process).

An important feature of XACML is that a rule is based on the definition of attributes corresponding to specific characteristics of a subject, resource, action, or environment. For

instance, a physician at a hospital may have the attribute of being a researcher, a specialist in some field, or many other job roles. According to these attributes, the physician can be able to perform different functions within the hospital. Attributes are identified by the ***SubjectAttributeDesignator***, ***ResourceAttributeDesignator***, ***ActionAttributeDesignator***, and ***EnvironmentAttributeDesignator*** elements. These elements use the ***AttributeValue*** element to define the requested value of a particular attribute. Alternatively, the ***AttributeSelector*** element can be used to specify where to retrieve a particular attribute. Note that both the attribute designator and ***AttributeSelector*** elements can return multiple values. For this reason, XACML provides an attribute type called *bag*, an unordered collection that can contain duplicates values for a particular attribute. In addition, XACML defines other standard value types such as string, boolean, integer, time, and so on. Together with these attribute types, XACML also defines operations to be performed on the different types such as equality operation, comparison operation, string manipulation, and the like.

```
<Policy PolicyId="Pol1"
        RuleCombiningAlgId=
        "urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:permit-overrides">
  <Target>
    <Subjects> <AnySubject/> </Subjects>
    <Resources>
      <Resource>
        <ResourceMatch
             MatchId="urn:oasis:names:tc:xacml:1.0:function:stringmatch">
          <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
             http://www.example.com/hospital/patient.xsd
          </AttributeValue>
          <ResourceAttributeDesignator
            DataType="http://www.w3.org/2001/XMLSchema#string"
            AttributeId="urn:oasis:names:tc:xacml:1.0:resource:target-namespace"/>
        </ResourceMatch>
      </Resource>
    </Resources>
    <Actions> <AnyAction/> </Actions>
  </Target>
  <Rule RuleId="ReadRule" Effect="Permit">
    <Target>
      <Subjects>
        <Subject>
           <SubjectMatch
                  MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
              head physician
            </AttributeValue>
            <SubjectAttributeDesignator
              AttributeId= "urn:oasis:names:tc:xacml:2.0:example:attribute:role"
              DataType="http://www.w3.org/2001/XMLSchema#string"/>
           </SubjectMatch>
        </Subject>
      </Subjects>
      <Resources> <AnyResource/> </Resources>
      <Actions>
        <Action>
          <ActionMatch
                  MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
              read
            </AttributeValue>
            <ActionAttributeDesignator
               DataType="http://www.w3.org/2001/XMLSchema#string"
               AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"/>
          </ActionMatch>
        </Action>
      </Actions>
    </Target>
    <Condition FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
      <SubjectAttributeDesignator
```

```
        DataType="http://www.w3.org/2001/XMLSchema#string"
        AttributeId="urn:oasis:names:tc:xacml:1.0:subject:head-physicianID"/>
      <AttributeSelector RequestContextPath=
              "/ctx:Request/ctx:Resource/ctx:ResourceContent/hospital:record\
/hospital:patient/hospital:patient-head-physicianID"
        DataType="http://www.w3.org/2001/XMLSchema#string"/>
    </Condition>
  </Rule>
</Policy>
```

As an example of XACML policy, suppose that a hospital defines a high-level policy stating that any user with role ***head physician*** can read the ***patient record*** for which she is designated as head physician. The example above illustrates the XACML policy corresponding to this high-level policy. The policy applies to requests on the ***http://www.example.com/hospital/patient.xsd*** resource. The policy has one rule with a target that requires a ***read*** action, a subject with role ***head physician*** and a condition that applies only if the subject is the head physician of the requested patient.

XACML 3.0: Privacy profile

The XACML 3.0 [XACML30] Privacy Policy Profile Version 1.0 is a standard issued by the OASIS group describing "a profile of XACML for expressing privacy policies". This profile defines two attributes:

- urn:oasis:names:tc:xacml:2.0:resource:purpose, which indicates the purpose for which a data resource was collected, and
- urn:oasis:names:tc:xacml:2.0:action:purpose, which corresponds to the purpose for which access to a data resource was requested.

A standards rule is defined, according to which access to the requested resource is to be denied unless the two above-mentioned purposes match by regular-expression match, as shown below:

```
<Rule xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-06"
      RuleId="urn:oasis:names:tc:xacml:2.0:matching-purpose"
      Effect="Permit">
  <Condition
      FunctionId="urn:oasis:names:tc:xacml:2.0:function:string-regexp-match">
    <AttributeDesignator
        category="urn:oasis:names:tc:xacml:3.0:attribute-category:resource"
        AttributeId="urn:oasis:names:tc:xacml:2.0:resource:purpose"
        DataType="http://www.w3.org/2001/XMLSchema#string" />
    <AttributeDesignator
        category="urn:oasis:names:tc:xacml:3.0:attribute-category:action"
        AttributeId="urn:oasis:names:tc:xacml:2.0:action:purpose"
        DataType="http://www.w3.org/2001/XMLSchema#string" />
  </Condition>
</Rule>
```

Such rule must be used in the scope of rule-combining algorithm urn:oasis:names:tc:xacml:2.0:rule-combining-algorithm:deny-overrides. To conform to such specification, any implementation should include, as an XACML request producer, make use of the attributes above, and, as an XACML policy processor, enforce the proposed rule, respectively. The profile deals with an important aspect that can be found also in the research context of the PrimeLife project, namely, purposes of data processing. Nevertheless, purposes are a very specific facet of Data Handling, and we consider the scope of this standard proposal too narrow if compared to the much more general concepts its name, "Privacy Policy Profile", refers to.

5.1.3 Platform for Privacy Preferences Project (P3P)

The P3P 1.0 Specification [P3P10] allows Web sites to declare their privacy practices in a standard and machine-readable XML format known as P3P policy. Designed to address the need of the users to assess whether the privacy practices adopted by a server provider comply with their privacy requirements, P3P has been developed by the World Wide Web Consortium (W3C). Supporting data handling policies in Web-based transactions allows users to automatically understand and match server practices against their privacy preferences. Thus, users need not read the privacy policies at every site they interact with, but they are always aware of the server practices in data handling. In summary, the goal of P3P is twofold. On one side, it allows Web sites to state their data collection practices in a standardized, machine-readable way. On the other side, it provides to users a solution to understand what data will be collected, how those data will be used, and what data/uses they may "opt-in" to or "opt-out".

Policy, Disputes, Statement

P3P provides a formal model for representing privacy practices of service providers. The main concepts of interest in the P3P policy language are *policies*, *disputes* and *statement*.

A P3P policy has, as root element, ***Policy***. A ***Policy*** element consists of an ***Entity***, an ***Access***, a ***Dispute-Group***, and one or more ***Statement*** elements. More in details, the ***Entity*** element gives some information about the legal entity responsible for the representation of the privacy practices. The ***Access*** element indicates whether the users are allowed to access their information stored at the Web sites, and it can assume different values as for instance: *i) nonident:* no identified data are collected, *ii) all:* access is provided to all data, *iii) ident-contact:* access is given to contact data, *iv) none:* no access to identified data is given. The ***Dispute-Group*** element consists of one or more optional ***Dispute*** elements, which describe the procedures that may be followed for disputes about an entity's privacy practices. Finally, the ***Statement*** element describes the real privacy practices used in the management of the data referenced by the ***Data-Group*** element. All data are handled according to the ***Purpose***, ***Recipient***, ***Retention***, and optionally ***Consequence*** elements, which are discussed below.

- ***Data-Group*** contains one or more ***Data*** elements describing the type of data to which the policy applies. Each ***Data*** element can provide hints about the intended uses of the data by specifying a ***Categories*** element. Categories make P3P user agents easier to implement and use.
- ***Purpose*** contains one or more purposes for which data are collected or used. Sites must classify their data practices into one or more of the predefined purpose values, as for instance: *i) current,* data are used to complete the activity for which have been released; *ii) individual-decision,* data are used to calculate statistics and determine interests and characteristics of the users; *iii) contact,* data are used to communicate with the users. ***Purpose*** elements provide to the users means to consent or not to some purposes, by defining an attribute *required* with three possible values: *i) always (default),* purpose is required; *ii) opt-in,* data can be used for the stated purpose only when the user explicitly requests this use; *iii) opt-out,* data can be used for the stated purpose unless the user requests otherwise. Also, an element named *primary purpose* is defined. It indicates the primary reason for which the data recipient is collecting data.

- ***Recipient*** defines one or more entities that can potentially collect user data. These recipients must respect the constraints defined in the policy. Some predefined recipient values are defined as for instance: *i) ours*, data can be collected by the collector itself or by entities related to it; *ii) same*, data can be collected by legal entities following collector practices. Same opt-in/opt-out functionality as for ***Purpose*** element is provided for the ***Recipient*** element.
- ***Retention*** indicates the kind of retention policy that applies to the data referenced in the statement. It provides a set of predefined values that restrict data collection to a certain period of time as for instance: *i) stated-purpose*, data are deleted as soon as possible; *ii) indefinitely*, data are retained without temporal restrictions.
- ***Consequence*** represents a human-readable description that provides an explanation of the site's privacy practices.

Users specify their privacy preferences through a policy language, called *A P3P Preference Exchange Language* APPEL [APPEL10], and enforce privacy protection by means of an agent. A user agent compares a user's privacy policy with a service provider's P3P policy and verifies whether the P3P policy conforms to user privacy preferences. Agrawal et al. [AKSX 2003] propose another language, called XPref, for user preferences definition.

Example

As an example of P3P policy, suppose that a company ACME provides a *book_a_flight* service. To complete the service release, the company requires to the requester attributes name, address, credit card number. Also, with explicit permission only, ACME uses e-mail addresses of the users to provide them with special offers.

```
<POLICY>
  <ENTITY>
    <DATA-GROUP>
      <DATA ref="#business.name">ACME</DATA>
      <DATA ref="#business.contact-info.postal.street">9, West 46Th Street</DATA>
      <DATA ref="#business.contact-info.postal.city">New York</DATA>
      <DATA ref="#business.contact-info.postal.stateprov">NY</DATA>
      <DATA ref="#business.contact-info.postal.postalcode">10036</DATA>
      <DATA ref="#business.contact-info.postal.country">USA</DATA>
      <DATA ref="#business.contact-info.online.email">acme@example.com</DATA>
      <DATA ref="#business.contact-info.telecom.telephone.intcode">1</DATA>
      <DATA ref="#business.contact-info.telecom.telephone.loccode">303</DATA>
      <DATA ref="#business.contact-info.telecom.telephone.number">7463914</DATA>
    </DATA-GROUP>
  </ENTITY>
  <ACCESS>
    <all/>
  </ACCESS>
  <DISPUTES-GROUP>
    <DISPUTES resolution-type="independent" service="http://www.example.org">
      <REMEDIES>
        <correct/>
      </REMEDIES>
    </DISPUTES>
  </DISPUTES-GROUP>
  <STATEMENT>
    <CONSEQUENCE>
      We use this information to provide the service.
    </CONSEQUENCE>
    <PURPOSE>
      <current/>
    </PURPOSE>
    <RECIPIENT>
      <ours/>
      <same/>
    </RECIPIENT>
```

```
    <RETENTION>
      <stated-purpose/>
    </RETENTION>
    <DATA-GROUP>
      <DATA ref="#user.name"/>
      <DATA ref="#user.home-info.postal"/>
      <DATA ref="#dynamic.miscdata">
         <CATEGORIES><purchase/></CATEGORIES>
      </DATA>
    </DATA-GROUP>
  </STATEMENT>
  <STATEMENT>
    <CONSEQUENCE>We use this information to provide special offers.</CONSEQUENCE>
    <PURPOSE>
       <individual-decision required="opt-in"/>
       <contact required="opt-in"/>
    </PURPOSE>
    <RECIPIENT>
       <ours/>
    </RECIPIENT>
    <RETENTION>
       <business-practices/>
    </RETENTION>
    <DATA-GROUP>
      <DATA ref="#user.home-info.online.email"/>
      <DATA ref="#dynamic.miscdata">
        <CATEGORIES><purchase/></CATEGORIES>
      </DATA>
    </DATA-GROUP>
  </STATEMENT>
</POLICY>
```

P3P 1.1

The P3P Note introduces new features, such as datatype extensions. The ***Statement*** construct is enriched with a grouping and naming mechanism. A new ***p3pattr*** attribute has been defined, to be used within other XML-based languages to refer to P3P policies. An optional ***Jurisdiction*** element has been added for declaring the jurisdiction of data recipients. An optional ***Our-Host*** element has been added for declaring domain relationships, allowing user agents to recognize when hosts in different domains are owned by the same entity or entities acting as agents for one another. P3P 1.1 also added user agent guidelines and improved the protocol allowing sites to identify related sites. The guidelines include a set of plain language translations of P3P vocabulary elements, to be used user-agent implementers. The P3P 1.1 specification has been published as a W3C Working Group Note in 2006.

5.1.4 Security Policy Assertion Language (SecPAL)

SecPAL [BFG2006, BFG 2007] is a security policy language developed to meet the access control requirements of large-scale Grid Computing Environments. SecPAL is declarative, logic-based, and builds on a large body of work showing the value of such languages for flexibly expressing security policies. It was designed to be comprehensive and provides a uniform mechanism for expressing trust relationships, authorization policies, delegation policies, identity and attribute assertions, capability assertions, revocations, and audit requirements. This provides tangible benefits by making the system understandable and analyzable. It also improves security assurance by avoiding, or at least severely curtailing, the need for semantic translation and reconciliation between disparate security technologies.

SecPAL is expressive enough to implement key requirements of authorization in distributed system. First, it allows different variants of delegation of rights, using the primitive "can say".

Next, it allows domain-specific constraints to express policies with parameterized roles, role hierarchies, separation of duties, threshold constraints, temporal constraints, periodicity constraints, policies on structured resources such as file systems, and revocation. Finally, negation is only supported in queries to avoid intractability and ambiguity due to negation while being able to express complex queries.

SecPAL has a clear and readable syntax close to natural language (e.g. Bob says Alice can read /project/data). Policy editors do not have to be familiar with underlying formal logic and policies are easier to read than XML-based languages. Naturally, SecPAL assertions can be serialized in XML when exchanged.

SecPAL relies on a succinct and unambiguous semantic based on only three deduction rules, which specify the meaning of SecPAL assertions. It is indeed important that syntax and semantic are simple and intuitive.

The evaluation of SecPAL authorization queries is decidable and efficient resolution is specified and implemented. SecPAL can be extended in a modular and orthogonal way. For example, the parameterized verbs, the environment functions, and the language of constraints can all be extended.

A very simple example could look as follows. Researcher Alice wants to access a file on a file server. The company's security token service (STS) issued a credential to Alice: "STS says Alice is a researcher". The assertions are encoded in XML and signed by the issuer of the assertion, in this case the STS. Let's assume that the file server has a security policy: a) "FileServer says STS can say x is a researcher" and b) "FileServer says y can read file://project if y is a researcher". Finally, Alice wants to read a file on the file server. She sends her read request together with her assertion to the file server. Assuming that the file server is in charge of access control decisions, the following SecPAL authorization query is evaluated: "FileServer says Alice can read file://project?". The SecPAL engine has Alice's assertion, the policy and the query and uses an inference mechanism to determine if the query can be deduced from the policy and the assertions. More complex example covering discretionary access control, mandatory access control, role-based access control, separation of duties, threshold trust, attribute-based delegation, constrained delegation, and bounded delegation are provided in SecPAL papers.

The assertions, the policy and the query are expressed in the same language. The verbs "can say" and "can act as" are special keywords in SecPAL that allow limited and unlimited delegation chains. In the previous example, we see this when the file server lets the STS define who is a researcher. SecPAL defines a set of verbs such as "read", "write", "execute" but is open for new verbs. However, the current implementation only allows a fixed set of verbs in the context of Grid Computing.

5.2 Drawbacks of existing languages

Although the important contributions made by PRIME, XACML, P3P, and SecPAL approaches, they have some drawbacks or lacking features that are discussed and evaluated for further work in the PrimeLife project activity 5 (WorkPackages 5.2 and 5.3).

5.2.1 PRIME

- *Credential definition and integration* Although PRIME supports requests for digital credentials in the policies, it is not fully integrated with anonymous credentials, requests for zero-knowledge proof and verifiable encryption.
- *Matching* PRIME data handling solution does not provide an automatic mechanism to match preferences of the user and privacy policies of the server.
- *Secondary use* PRIME provides a simple solution to secondary use, where data handling policies are attached to the data and follow them for the entire lifecycle. No further customizations based on the chain of releases are possible.

5.2.2 XACML

- *Support for privacy constraints* XACML does not explicitly support privacy features (until version 2.0).
- *Credential definition and integration* Although XACML supports digital credentials exchange, it does not provide requests for certified credentials. As a consequence, it is possible neither to define requests for anonymous credentials, nor to define conditions that must be satisfied by the same atomic credential.
- *Ontology integration* XACML does not provide extensive support for ontological reasoning.
- *Obligation support* XACML supports obligations as action to be done by the service provider after an access has been granted. No support for privacy obligation, that is, those actions imposed to the data collector in the further use of data (e.g., notification, deletion, obfuscation), is provided.
- *Definition of privacy-aware architecture* XACML assumes to have all the information about a requester available at the time of policy evaluation and access control decision. Also, support for anonymous interactions is not provided.

5.2.3 P3P

- *Web-oriented* P3P is mostly defined for Web scenarios.
- *Policy negotiation* P3P does not support privacy practices negotiation. The users in fact can only accept the server privacy practices or stop the transaction.
- *Attribute-based restrictions* P3P does not allow the definition of attribute-based restrictions, and is not integrated with obligations that define actions to be performed after an access has been granted (e.g., notification, deletion, obfuscation).
- *Privacy practices enforcement* P3P does not provide a solution to demonstrate that the Web site enforces the privacy practices presented to the users. Although our solution suffers of the same problem, its integration in the context of the PRIME infrastructure mitigates this problem, since PRIME can be considered as a trusted platform.
- *Sticky policies* P3P does not provide protection against chains of releases. Personal data are not tagged with data handling policies (i.e., sticky policies).
- *Matching* Interactions between P3P and APPEL result in serious problems when using APPEL: users can only directly specify what is unacceptable in a policy, simple preferences are difficult to specify, and writing APPEL preferences is error prone.

5.2.4 SecPAL

- *Access control and credentials* SecPAL does not support anonymous credentials. Extensions would be required to use SecPAL reasoning capabilities and delegation features on top of selective disclosure.
- *Data handling and obligations* SecPAL mainly lacks constructs to deal with obligations. Adding support for obligations without breaking the properties of the formal model is not an easy task. Moreover, dealing with "purpose" should not impact the formal description of the language but such extensions are not allowed by the current implementation.
- SecPAL is not a standard.

5.3 Conclusion and future work

The PrimeLife project Activity 5 is investigating in the field of next generation policy languages. This state of the art is an inventory of existing languages and their analysis with respect to the targeted requirements that we have presented earlier in this document. Further work is conducted in the same activity and will continue this analysis while designing next generation policy language(s). In particular, WorkPackage 5.3 may reuse and interoperate with the existing languages as far as possible.

References

[APPEL10]
A P3P Preference Exchange Language 1.0, W3C Working Draft, 15 April 2002
http://www.w3.org/TR/P3P-preferences/

[Art29Op100]
Article 29 Working Party, Opinion 100: Opinion on More Harmonised Information Provisions, accessed 10 December 2008.
http://ec.europa.eu/justice_home/fsj/privacy/docs/wpdocs/2004/wp100_en.pdf

[Art29Op100A]
Article 29 Working Party, Opinion 100: Annex, accessed 10 December 2008.
http://ec.europa.eu/justice_home/fsj/privacy/docs/wpdocs/2004/wp100a_en.pdf

[Art29Op136]
Article 29 Working Party, Opinion 136: Opinion on the concept of personal data, accessed 10 December 2008.
http://ec.europa.eu/justice_home/fsj/privacy/docs/wpdocs/2007/wp136_en.pdf

[D6.2.1]
PrimeLife Public Deliverable D6.2.1, Infrastructure for Trusted Content, 28 August 2008.
http://www.primelife.eu/images/stories/deliverables/
d6.2.1-infrastructure_for_trusted_content-public.pdf

[Dir2002/58/EC]
Directive 2002/58/EC on Privacy and Electronic Communications, accessed 10 December 2008.
http://eur-lex.europa.eu/pri/en/oj/dat/2002/l_201/l_20120020731en00370047.pdf

[Dir95/46/EC]
Directive 95/46/EC of the European Parliament and of the Council on the protection of individuals with regard to the processing of personal data and on the free movement of such data, accessed 10 December 2008.
http://ec.europa.eu/justice_home/fsj/privacy/docs/95-46-ce/dir1995-46_part1_en.pdf

[EPAL11]
Enterprise privacy authorization language (epal 1.1), 2003.
http://www.zurich.ibm.com/security/enterprise-privacy/epal

[IBMMSWS]
Security in a web services world: A proposed architecture and roadmap, 2002.
http://www.ibm.com/developerworks/webservices/library/ws-secmap/

[Liberty]
Liberty alliance project. Accessed 20 July 2009.
http://www.projectliberty.org/

[P3P10]
The Platform for Privacy Preferences 1.0 (P3P1.0) Specification, W3C Recommendation, 16 April 2002.
http://www.w3.org/TR/P3P

[P3P11]
The Platform for Privacy Preferences 1.1 Specification, W3C Working Group Note, 13 November 2006.
http://www.w3.org/TR/P3P11

[PLING-UC]
W3C Policy Language Interest Group, Use Cases, accessed 24 October 2008.
http://www.w3.org/Policy/pling/wiki/UseCases

[PPEL]
Liberty architecture framework for supporting privacy preference expression languages, 2003.
http://www.projectliberty.org/liberty/resource_center/papers/privacy_preference_expression_languages_whitepaper_pdf

[PRIME]
Privacy and Identity Management for Europe (PRIME). Accessed 20 July 2009.
http://www.prime-project.eu.org/

[SAML11]
OASIS Security Assertion Markup Language (SAML) V1.1, 2003.
http://www.oasis-open.org/committees/security/

[WSPol]
Web Services Policy 1.5 - Framework, W3C Recommendation, 04 September 2007.
http://www.w3.org/TR/2007/REC-ws-policy-20070904

[XACML20]
Extensible Access Control Markup Language Version 2.0, accessed 21 July 2009.
http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf

[XACML30]
OASIS XACML v3.0 Privacy Policy Profile Version 1.0, Committee draft 1, April 2009.
http://www.oasis-open.org/committees/document.php?document_id=32425

[ACDS 2006]
Ardagna et al. Supporting location-based conditions in access control policies. In Proc. of the ACM Symposium on Information, Computer and Communications Security (ASIACCS'06), Taipei, Taiwan, March 2006.

[AHKS 2002]
Ashley, Hada, Karjoth, and Schunter. E-P3P, privacy policies and privacy authorization. In Proc. of the ACM workshop on Privacy in the Electronic Society (WPES 2002), Washington, DC, USA, November 2002.

[AKSX 2003]
Agrawal, Kiernan, Srikant and Xu. An xpath based preference language for P3P. In Proc. of the 12th International World Wide Web Conference, Budapest, Hungary, May 2003.

[AL 2005]
Ahn and Lam. Managing privacy preferences in federated identity management. In Proc. of the ACM Workshop on Digital Identity Management, Fairfax, USA, November 2005.

[BFG 2006]
Becker, Fournet, and Gordon. SecPAL: Design and semantics of a decentralized authorization language. Technical Report MSR-TR-2006-120, Microsoft Research, September 2006.

[BFG 2007]
Becker, Fournet, and Gordon. Design and semantics of a decetralized authorization language. In Proc. of the 20th IEEE Computer Security Foundations Symposium (CSF 2007), Venice, Italy, July 2007.
[BMD 2009]
Becker, Mackay, and Dillaway. Abductive authorization credential gathering. In Proc. of the IEEE International Symposium on Policies for Distributed Systems and Networks (POLICY), London, U.K., July 2009.
[BS 2002]
Bonatti and Samarati. A unified framework for regulating access and information release on the web. Journal of Computer Security, 10(3):241-272, 2002.
[CL 2001]
Camenisch and Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Proc. of the Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 2001.
[SD 2001]
Samarati and De Capitani di Vimercati. Access control: Policies, models, and mechanisms. In Foundations of Security Analysis and Design, LNCS 2171. Springer-Verlag, 2001.