D4.3 Context Aware Privacy Policies

Project	SWELL
Project leader	Wessel Kraaij (TNO)
Work package	WP4
Deliverable number	D4.3
Authors	Bob Hulsebosch (Novay), Martijn Oostdijk (Novay), Bas van Schoonhoven (TNO)
Reviewers	Saskia van Dantzig (Philips Research), Sanne Huveneers (TNO)
Date	Due date: December 11, 2012
Version	1.0
Access Rights	Internal (workpackage only)
Status	FINAL

SWELL Partners:

Ericsson, NCSI, Noldus, Novay, Philips, TNO, Radboud Universiteit Nijmegen, Roessingh Research and Development, & Universiteit Twente.

Summary

In SWELL, a range of well-being and well-working applications are envisioned that involve sharing of personal information in different contexts, including medical, working and home contexts, and involving different social contexts. In order to preserve the privacy of the individuals using the SWELL applications, the policies that govern what information may be shared must be maintained in a manner sensitive to the context in which the application is used.

The XACML policy expression language, along with the complementary OAuth standard and UMA protocol may prove to be useful in implementing context-sensitive privacy management. What context information is, and what its characteristics are, is often unclear. This document describes both aspects and introduces a context model that is used as a reference model for the case-study.

The policies that the user expects to be maintained vary in different contexts. The initial case study of a typical SWELL application shows that there is some promise in automating part of the context-sentitive privacy policy management.

Some points of attention with regard to the context-sentitive policy management are:

- Usability: are the policies easy to configure?
- Policy accuracy: does the formulated policy result in behavior that the user expects and wants?
- Contextual parameter accuracy: some parameters may not be trustworthy or accurate enough to ensure reliable knowledge about what context a user is in and consequently what policy should be enforced.

Contents

1 Introduction

The privacy debate often boils down to calling personal information “public” or “private”: once information is “public” it is no longer “private” and hence privacy no longer applies. This view is simplistic and does not take into account the reality of how human beings share information about themselves. Privacy researcher Helen Nisenbaum notes this also, and states: “Most people have a robust sense of the information about them that is relevant, appropriate, or proper to particular circumstances, situations, or relationships. When information is judged appropriate for a particular situation it usually is readily shared […]”[1]

Instead of the binary model of “public” versus “private”, privacy with regards to personal information should be understood in relation to the *context* in which the information is shared. While a person may not have objections to sharing certain personal information in a medical context, for example during a visit to a doctor, he may strongly object to sharing the same information in a different context, for example a job interview. As Nisenbaum describes it, there are *norms* governing what information may be shared that are a context: “For the myriad transactions, situations and relationships in which people engage, there are norms—explicit and implicit—governing how much information and what type of information is fitting for them.“[1] She introduces the term “contextual integrity” to identify situations in which these norms are maintained.

In SWELL, a range of well-being and well-working applications are envisioned that involve sharing of personal information in different contexts, including medical, working and home contexts, and involving different social contexts. In order to preserve the privacy of the individuals using the SWELL applications, the norms that govern what information may be shared must be maintained in a manner sensitive to the context in which the application is used.

Ultimately, it is not laws or social conventions that determine the norms governing personal information sharing, but the preferences of the individual to whom the information concerns. A person should be able not only to determine whether information about him or herself is private or public, but where, when and with whom it is shared. This power makes people able to define the nature and degree of intimacy of various relationships, as James Rachels argues.[2] In providing the user with such control over what information is shared in what context, it is important to understand that “usability is just as important as engineering principles and practices”, as Ira Rubinstein puts it.[3] Ultimately, the challenge is to provide the user with meaningful control over the sharing of his or her information, while avoiding common pitfalls when designing applications from a user perspective, such as obscuring potential information flow, obscuring actual information flow, emphasizing configuration over action, lacking coarse-grained control, and inhibiting existing practice, as described by Scott Lederer.[4]

To wrap up this discussion:

1. Norms governing collection and sharing of personal information are context-specific.
2. A person must control what norms with regards to the sharing of his or her personal information are maintained in a given context.

3. Usability of solutions that provide the user with such control is crucial.

1.1 Context aware privacy policies

How does the idea of norms governing the flows of personal information translate to a technological context? A computer-readable variety of such norms can be found in the form of *access policies*. Access policies describe what resources (such as data or applications) are accessible to whom (or what application), under what conditions. For example, an access policy could describe that only a person who has logged in as a doctor may get access to medical records.

The situation becomes more complicated, however, if we take into account that the norms governing the sharing of personal data are context specific. The context aware privacy policy paradigm allows for adaptation of access to sensitive information depending on a set of relevant information collected from the dynamic environment and the preferences and capabilities of the interacting entities, i.e. the context. As the environment evolves, the context changes and so should privacy control in order to dynamically cope with new requirements. We argue that privacy control can be made less intrusive, more intelligent, and able to adapt to the rapidly changing contexts of the environment.

The well-being and well-working applications envisioned in SWELL involve sharing of personal information between users and services. For instance activity monitoring data of a user recovering from medical treatment may need to be shared with a health professional, or location information needs to be shared between an employee and his employer during working hours. Such sharing of personal information on the Web has several shortcomings:

- Users often don’t have any control over what information to share with whom.
- Access control to personal information lacks sophistication since it is a side issue for most applications.
- Users need to use many diverse and bespoke policy management tools with diversified user experience.
- Policies expressed in diverse and possibly incompatible policy languages cannot be reused for distributed resources.
- Poor support for policies does not allow users to express their sharing settings in a flexible way.
- Lack of central management of access relationships between services hosting and accessing personal data.

Driven by a rapid invasion of a variety of connected context sources and an explosion of services allowing individuals to exchange personal information and content that they have created, users increasingly have difficulty managing their privacy online. The goal of a flexible, user-centric privacy control infrastructure must be to allow the user to intuitively determine what information, given a certain context, will be revealed to which parties and for what purposes, how these parties will

handle the information, and to provide insight into what the consequences of sharing this information will be.

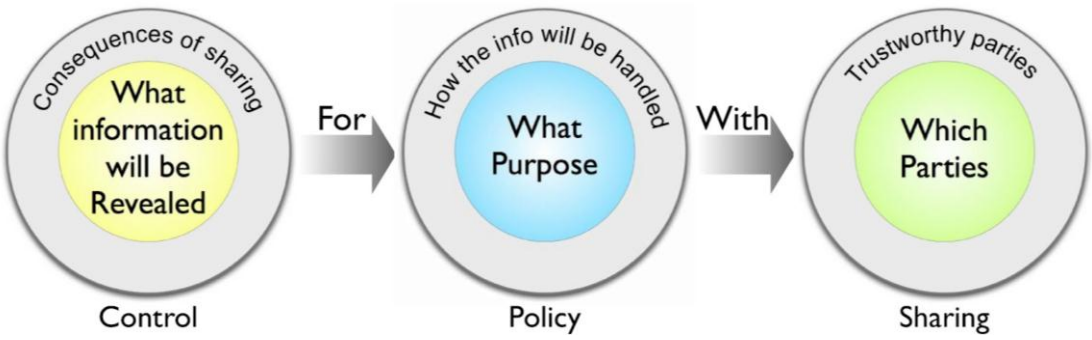

Figure 1: Insight in the consequences of sharing personal information.

1.2 Reading guide

This deliverable focuses on one particular aspect: how to make privacy control context aware and particularly on how to include context parameters in access policies and how to select the right policy for a given situational context? The idea behind is that access control to privacy sensitive information should be user friendly and intuitive. The use of context information to control access can help achieve these requirements. One way to achieve this is to make privacy policies context aware. This deliverable studies the possibilities, advantages, and drawbacks of making privacy policies context aware.

In chapter 2, different possible contextual parameters are inventoried, that may be used to implement context-aware privacy policies. In chapter **Error! Reference source not found.** a number of prominent technologies that are used, or may be used, for implementing privacy policies in information systems are discussed. This is applied to the SWELL domain in chapter 4, by analyzing a scenario in which a SWELL application is used. Chapter 5 draws conclusions on what this means for the usefulness of context-aware privacy policy control for applications such as envisioned in SWELL.

2 Context parameters and conditions

Context awareness is a topic that has been receiving a lot of attention the last five years or so. Gartner names context-aware computing as one of four IT-enabled initiatives most worthy of client consideration in the near future [5]. The central idea behind context awareness is that aspects of the user's context such as time, place, method of access, recent behaviour, and relationships with others should be taken into account while the user is interacting with online services, in order to improve some aspect of those services. Gartner places context delivery architecture on their emerging technologies Hype Cycle [6]. Forrester identifies the single most important driver of context as the rise of the mobile device [7]. Indeed: *Mobile devices offer opportunities (new sensors) and limitations (due to their size) in terms of user interface. The former are enablers for context, the latter necessitate context.*

What context information is, and what its characteristics are, is often unclear. This section describes both aspects and introduces a context model that will be used as a reference model for our context-enhanced authorization exploitation.

2.1 Context model

Many definitions of context exist. Probably the most common and systematic definition of context comes from Dey and Abowd [8]. They define context as any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.

Context information can be classified at various levels of abstraction. In this white paper we distinguish context domains, context types and context sources. Context domains define the high-level physical, mental and virtual environment or situations the user could be in. Each context domain can be described or characterized by a number of specific context types. Several context domains and types can be identified and described as follows:

- Activity-based context: What is the user doing? Represents the relevant activities that are going on at the moment the service is initiated. Is the user working, travelling or in a meeting?
- Social context: With whom is the user? Represents the party of people who are surrounding the smart-phone user. Are they together with the user? Are they friends of the user? Does the user behave in a normal manner? How does the social network context looks like
- Location: Where is the user? What is the accuracy of the location? Is he in a country or city, in the office building or at home, is he nearby something or someone, or are his longitude/latitude coordinates known?
- Time: What time is it? Relevant context types are time of the day, office hours, lunch time, and between certain points in time (e.g. from 8:00 till 17:00 hours, from 21:00 till 23:00 hours).
- Physiological context: What is the user's physical condition? Relevant information can be obtained from skin temperature, heart rate or voice characteristics (e.g. trembling of voice).

- Environmental context: In what environment are the user and the application? What is happening with the environment around the user? Is it raining outside, what is the smog level of the air, are there a lot of pollen in the air? But the environmental state could also take into account recent events as generated by a Security Incident and Event Management system (SIEM) of the organization (are we under attack?).
- Mental context: How is the user feeling? What emotion does the context of the user bring about? For instance, is the user happy, scared, sad, or stressed?
- Network context: How is the user communicating? Is he using a VPN or working via an office LAN, what IP-address is he communicating from, is he using WiFi or 3G?
- Device and application context: What kind of device is the user using? Is it a mobile device or a desktop PC, is it a company controlled device or a private device (i.e. the bring your own device (BYOD) paradigm), what is that patch status of the device, i.e. does the device have the most recent virus scanners, what is the installed operating system on the device, and what mobile apps are running on the device? What is the security state of the application?

Each of the context domains and their context types are shown in Figure 2. This is an adaptation of the model defined by Van der Klein et al. in [9].

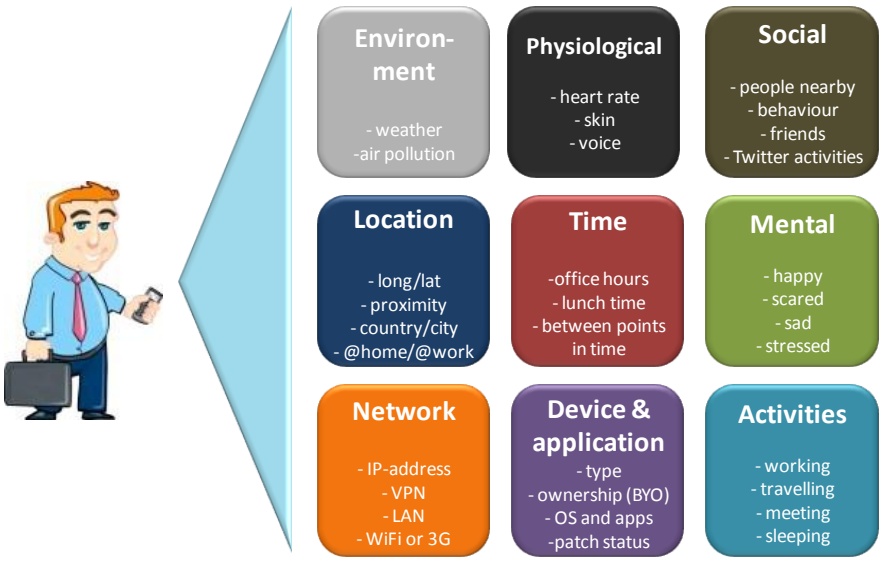

Figure 2: Context domains and types.

Note that for most of these context domains a time-dimension could be added. Tracking the user's context domain in time allows for detection of patterns and derivation of behaviour.

Context sources are the actual generators or owners of context information such as sensors and reasoning engines. For each of the context types several context sources can be identified. **Error! Reference source not found.** give examples of sources.

Context Domain	Context Type	Context Source
Environment	Weather	Meteorological service
	Air pollution	Weeronline.nl
	Security events & incidents	SIEM
Physiological	Heart rate	ECG sensor, skin color (Philips vital signs cam app)
	Blood pressure	Digital blood pressure meter (cuff)
	Blood alcohol concentration	Breathalyzer,
	Respiratory rate	Chest movement (Philips vital signs app)
	Skin	Skin temperature sensor
	Voice	Voice recording tool
	Physical activity level	Activity monitor/accelerometer
Social	People nearby	Bluetooth, Google Latitude, Outlook Calendar
	Behaviour	Key stroke logger, monitoring service, Outlook Calendar
	Friends / Colleagues	LinkedIn, Facebook, Twitter
	Social network activity	Twitter messages, LinkedIn user connections update
Location	Longitude/latitude	GPS (of mobile device), Google Latitude, GSM/UMTS cell ID
	Proximity	Bluetooth (pairing), GSM/UMTS cell ID, RRIF/NFC reader
	Country / City	IP-address, GPS, Outlook Calendar, reverse geo-coding
Time	Office hours	Server time
	Lunch time	Outlook Calendar
	Between points in time	Server time
Mental	Emotion (happy, sad)	Sound detector (laughter), face recognition camera
	Scared	A “galvanic-skin-responses” sensor, a skin temperature sensor.
	Stressed	Voice recorder, Outlook Calendar, heart sensor, camera
Network	IP-address	Network access gateway
	VPN	Network access gateway
	LAN	IP-address
	WiFi – 3G	Network access gateway
Device	Type	Device management system, java script
	Ownership (company or BYO)	Device management system
	OS	TCP/IP fingerprinting, java script
	Patch status	OS
Activity	Working	Outlook Calendar, GPS,
	Travelling	GPS, GPS derived velocity, Outlook Calendar, TripIt
	Meeting	Outlook Calendar, GPS, togetherness via proximity
	Sleeping	Heart sensor, brain activity sensor, sound detector, activity monitor

Table 1: Overview of context sources per context type per domain.

2.2 Context characteristics

When working with context information in general and particularly for security purposes certain characteristics of the information have to be taken into account in the context model. Notable characteristics of context information are [10], [11]:

- Heterogeneity and mobility: Context information models have to deal with a large variety of context information sources that differ in their update rate and their semantic level.
- Timeliness: Context-aware applications may need access to past states and future states (prognosis). Therefore, timeliness (context histories) is another feature of context information that needs to be captured by context models and managed by the context management system.
- Imperfectness: Context information may be incorrect if it fails to reflect the true state of the world it models, inconsistent if it contains contradictory information, or incomplete if some aspects of the context are not known. Buchholz et al. define Quality of Context (QoC) as "any information that describes the quality of information that is used as context information. Thus, QoC refers to information and not to the process nor the hardware component that possibly provide the information" [12].
- Representation variability: Much of the context information involved is derived from sensors. There is usually a significant gap between sensor output and the level of information that is useful to applications, and this gap must be bridged by various kinds of processing of context information. Therefore, a context model must support multiple representations of the same context in different forms and at different levels of abstraction. Moreover it must also be able to capture the relationships that exist between the alternative representations.
- Interrelatedness: Several relationships are evident between people, their devices and their communication channels (for example, ownership of devices and channels and proximity between users and their devices). Other less obvious types of relationships also exist amongst context information. Context information may be related by derivation rules which describe how information is obtained from one or more other pieces of information.
- Trustworthiness: How trustworthy is the provided context information? Does it come from a reliable source or not? The probability should be taken into account that a context source is spoofed or otherwise compromised. Confidentiality, integrity and availability are important factors to consider when working with context information.
- Privacy sensitiveness: Context information is often privacy sensitive information, e.g., location. The collecting, usage and storing of this information therefore has to be carefully analyzed with respect to the consequence for the privacy, and these privacy consequences should be minimized where possible through privacy-by-design. For example, collecting context at the lowest (and thereby least privacy sensitive) possible quality, storing it only when necessary and enforcing that the context is only used for its intended purpose[1] of context-enhanced privacy control. The privacy risks have to outweigh the benefits of context-enhanced privacy. There is no generic answer if the benefits outweigh the privacy risks, this basically depends on the used context (how privacy sensitive is it), and what the actual benefits are in a specific case.

[1] In Dutch: doelbinding

2.3 Context management and reasoning

Context management involves the aggregation and interpretation of distributed context information. A specific and important part in most of these infrastructures is the context reasoning component. Context reasoning can be defined as deriving higher-level context information from lower-level context information. Context information is qualified as 'higher-level' if it is more useful for a context consumer (ultimately a context-aware application) than the original context information, which is then qualified as 'lower-level'. Context information is more useful if it better describes a situation in which the context consumer is interested, or if the QoC associated with the context information is more appropriate for the needs of the context consumer. The 'lowest-level' context information is the 'raw' context information provided by context sensors or other original context sources, such as calendars and network databases.

During the reasoning process, collected data may be transformed into knowledge through interpretation. To achieve this transformation, an interpreter requires a model that describes how to map specific types of data it is provided with to useful information. For example, location coordinates may be interpreted as a logical location (e.g. ‘home’ or ‘hospital’) based on a model that describes the geographic boundaries of a set of logical locations that are frequently visited by a certain end-user. Additional knowledge can be derived through the process of inference. In general, inference can be described as deriving conclusions from what is already known. Similar to the interpretation approach, a model is used that describes how various types of knowledge can be combined to derive new knowledge. For example, an inferrer may derive whether an office worker is in a meeting based on his schedule, current location and number of people in his vicinity. Inference is also referred to as vertical reasoning. This term is used to distinguish from a process where the quality of knowledge is improved through the process of aggregation, i.e. combining context information on the same aspect of the state of the world, also referred to as horizontal reasoning. For example, the location information on a user may be more accurate if information from different location sources can be combined.

3 Privacy Policy Technologies

To make privacy management in an application context-aware, a number of key technologies and technological components are required. In this chapter we give a brief overview of those technologies and architectural components, along with a more extensive discussion of the key technologies for the reader with an interest in information technologies.

3.1 Overview

We regard privacy policies to be computer-readable norms with regards to the collecting and sharing of personal data in a specific context. In order for a computer to be able to understand and maintain these policies, they must be described in a computer-readable format, such as the XACML standard that is designed specifically for creating policies and automating their use to control access. We discuss the technical details of XACML later in this chapter.

An architecture for a SWELL application needs to contain several components to be able to manage and enforce privacy policies:

- A **norm setting** component – where the user can overview and adjust the privacy policies in a user-friendly manner.
- A d**ecision making (and context information gathering)** component – which is able to understand the privacy policies set by the user, gather information on the context the user is in and decide what policy applies in the given context, and whether acess to certain personal information is allowed.
- A **decision enforcement** component – to enforce the decisions made by the decision making component, and restricting access to personal information to unauthorized sources.

A global overview of these components is shown in Figure 3.

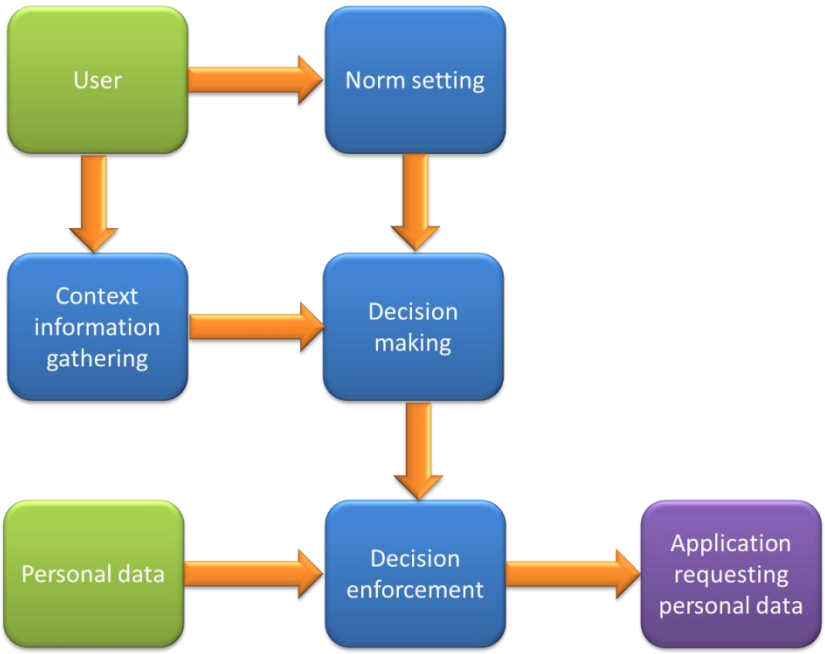

Figure 3: Global overview of components required for context aware privacy policies.

The remainder of this chapter contains a more technical discussion of the technologies and components that may be used to implement context-aware privacy policy management. In chapter 4, we apply the previous discussion to a use case involving a typical SWELL application.

3.2 Technical discussion

A privacy policy regulates access to resources such as medical dossiers, calendar information or sensor data. In order to properly enforce the policy, information needs to be collected about the subject (some of this information may have resulted from the authentication step, for example the user's role within an organization), and also information about the attempted action, about the resource, and possibly additional information (the 'environment'). The latter information can be considered as context attributes that are collected during the enforcement process.

The attribute based access control (ABAC) paradigm facilitates handling context attributes for access purposes. In an ABAC-style architecture attributes are used to convey authorization information from (central) authorities to relying parties. An important (perhaps, the de-facto) standard describing the architecture, syntax, and protocols for ABAC is XACML. In the remainder of this chapter the XACML policy expression language is introduced, along with the complementary OAuth standard and UMA protocol.

3.2.1 XACML policy expression language

The eXtensible Access Control Markup Language (XACML, [13]) is an XML-based language, or schema, designed specifically for creating policies and automating their use to control access to disparate devices and applications on a network. The power of XACML lies in the fact that access control is no longer something that resides inside the application, but is externally managed using a standardized policy language. Therefore, different applications could use the same policies, and these policies could be hierarchically defined at the federation level, the institution level, or the application level. This flexibility does have a downside: XACML suffers from complexity and

consequently has a steep learning curve for administrators to write policies, has poor performance characteristics, and poor interoperability between implementations. This has resulted in a slow and marginal uptake of XACML in commercial product suites. The XACML 3.0 specification which is in its final state for standardisation tries to tackle these issues. Amongst others it contains functionality for decentralised policy management via delegation and the definition of additional attribute categories.

The data flow model of XACML follows the pull model of the authorization decision query. Briefly, the access request is received by the policy enforcement point (PEP) which then communicates it to the decision point (PDP). The PDP evaluates the access request with regard to the applicable policy set, policy or rule and replies with an authorization decision. In order to make a decision, the PDP obtains attributes associated with the client issuing the access request, the resource that is being accessed and the environment in which the access request is taking place. Such attributes may very well be contextual attributes and are retrieved from the policy information points (PIPs). The pull model of XACML makes it less suitable for dealing with dynamically changing context parameters such as location during a user session (i.e., if the context changes, the PEP and PDP will only notice when the user tries to obtain access to a resource). Enforcing access on a transaction basis, however, can be pull-based and therefore be carried out via XACML. The XACML architecture is shown in Figure 4.

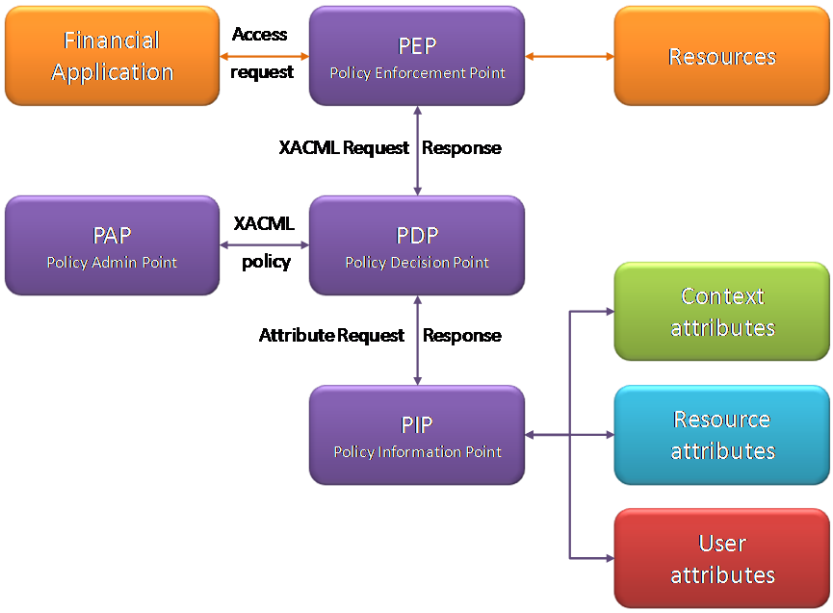

Figure 4: XACML architecture.

In XACML requests, responses, and policies may contain references to subject-, action-, resource-, and environment attributes that can be used to specify context. Built-in attributes are current date and time, but other attributes can be defined as well. Attributes originate from the client side (sent alongside the request from the PEP) or can be looked-up server side (by registering a PIP).

Most of the big identity and access management vendors have implemented XACML in their identity suites. Adoption in enterprise applications seems limited (hard numbers are not available, experts in

the field seem to indicate that simpler languages, such as SAML are being used to pass around authorization information[2]). The potential of using context information in an as part of authorization policies does not appear to be widely exploited.

XACML's architecture with a PIP attribute providers offers an important opportunity to start implementing context-enhanced privacy control. The resulting policies contain variables representing various types of context information. In order to not introduce too much complexity in the resulting policies, an appropriate level of abstraction needs to be chosen so that policies do not become overly complex. On the other hand, policies need to take into account that the quality of context information may not be perfect, so the level should not be too high. This results in a trade-offs.

When context parameters are added to access policy rules, these rules tend to become more complex. It may become harder to ensure that the produced policies are complete, safe, and conflict-free. It may, as a consequence, also get harder to determine which rules are no longer necessary.

How much the complexity of the context-aware policies increases depends on the abstraction level at which contextual aspects of these policies can be expressed. If this abstraction level is sufficiently high, the policy rules can remain simple. On the other hand, if the abstraction level is too high, it becomes harder to reason about the authenticity and quality of the context sources producing the context information. As an example of abstractions levels from high to low:

- "at home"
- "with 95% probability within 50 meter radius from home"
- "connected to VPN and
  cellphone was connected to celltower close to home in the last five minutes and
  coming from an IP address associated with home address".

XACML allows the policy administrator to specify policies independent of the hierarchy of the application that is to be protected. This makes it possible, for instance, to re-use policies in different parts of one application. Context information is available to the policy administrator through parameters which can be mapped to values delivered to the PDP via the PEP or values that need to be fetched from an attached PIP. So choosing the right level of abstraction is done by choosing appropriate context parameters and using them in appropriate policies.

Once defined, policies may be given a name and reused for different applications, in which case a XACML based centralized architecture makes sense. At this point it is unclear whether it scales to a realistic setting in terms of performance since context information is different from the usual static information on which today's access policies are based (the identity of the user, the role of the user within the organization): context information is only relevant when processed in (near) real-time, and there is much more context information (in terms of amounts of data). Whether identity and access management systems and policy evaluation engines can deal with the dynamicity and sheer

[2] See, e.g., https://identitysander.wordpress.com/2010/03/09/saml-vs-xacml-for-abac-authz/ and blogs referenced from that blog post.

size of context information obviously depends on how these systems are configured (features such as caching of environment information may need to be disabled). Policy evaluation can be a computationally intensive task when both the size of the sets of policies and the number of authorization requests per time unit grow.

3.2.2 OAuth

OAuth is a standard for one service to provide access to resources such as contacts, photos, or other data to another service or application. Users can revoke access at any time and do not have to reveal their credentials, only a potentially anonymous token, to the requesting service or application. Access is granular — providing the requesting service access to your address book does not also give that same service the ability to post to your blog or view your advertising revenue.

What this means is that Web-based services and applications now have a potential way to solve the problem of the user exporting her personal information from one service to another in a way that is secure, revocable, limited in scope, more private, and encourages good user behavior. OAuth does not attempt to solve other problems that can arise such as privacy policy management or data duplication and skew.

OAuth does not describe the access policies themselves or how services and applications determine what user may access what resource. OAuth is about *authentication* (is the user really who he says he is?), while XACML is about *authorization* (is the – authenticated – user allowed to access this resource?). In this sense, OAuth is complementary to XACML as both could be used in the same solution. The OAuth process is illustrated in Figure 5.

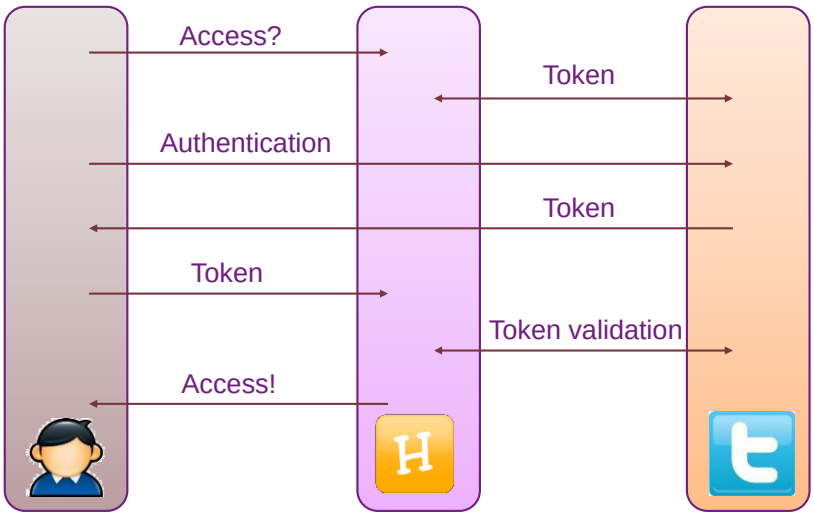

Figure 5: OAuth message flow.

How OAuth is reflected on today's Internet is Shown in below.

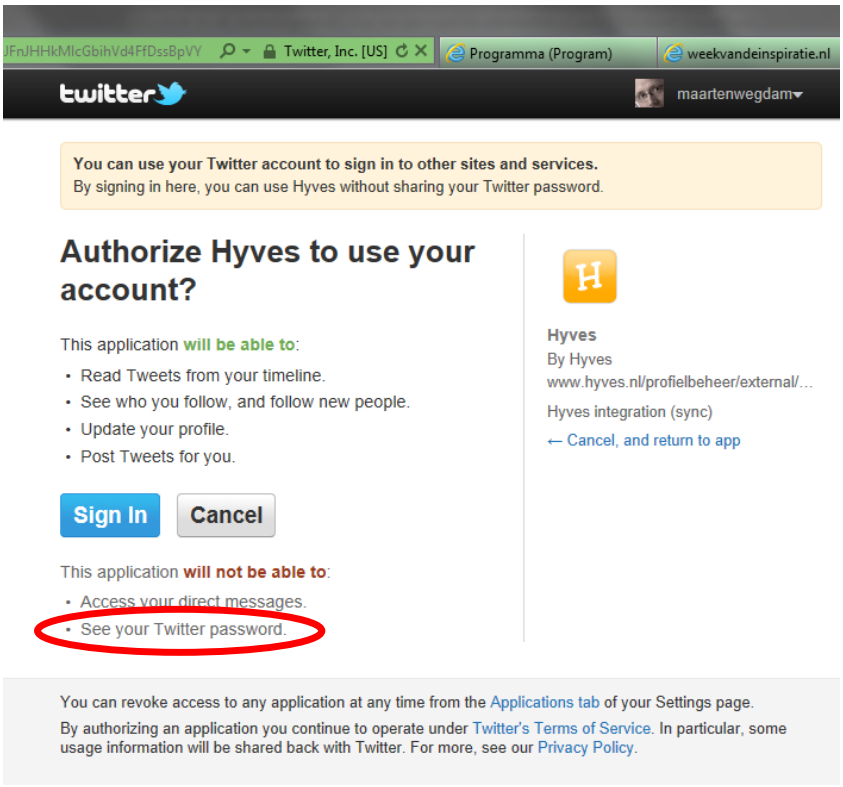

Figure 6: Delegated access via OAuth without sharing passwords.

3.2.3 UMA

Whereas XACML is expected to be used by resource managers and service providers, the access control of user-owned resources and services might be provided by UMA (User Managed Access). UMA is a user centric and OAuth-based web protocol that lets the user control authorization of data sharing and service access made between online services on his/her behalf. The UMA protocol is currently being developed in the Kantara UMA Work Group [14]. Version 1.0 of the UMA core protocol has been specified [15]. UMA provides a dedicated interface and service for:

- Authorizing data sharing and service access;
- Imposing sharing terms on any application wanting access;
- Monitoring, changing, and stopping access relationships;
- Letting services make requests of the user's authoritative sources directly.

UMA is a user centric and OAuth-based web protocol that lets the user control authorisation of data sharing and service access made between online services on his/her behalf. The UMA protocol is currently being developed in the Kantara UMA Work Group [16]. Version 1.0 of the UMA core protocol has been recently specified [17].

The idea behind UMA (User Managed Access) is that users should have a simple, standard way to control access to their online data or resources (files, photos, calendar, contacts, etc.) that doesn't depend on a single service provider (e.g., a large social network). Rather there should be a user-managed access protocol that works interoperably across different service providers, much the

same way Web servers or email servers work interoperably across a user's choice of different service providers

UMA builds on top of the IETF OAuth 2.0 standard, and adds the critical pieces needed for a user to set up and configure an online Authorization Manager (AM). An AM is a service that acts on behalf of a user to control access to the user's resources stored anywhere on the Web.

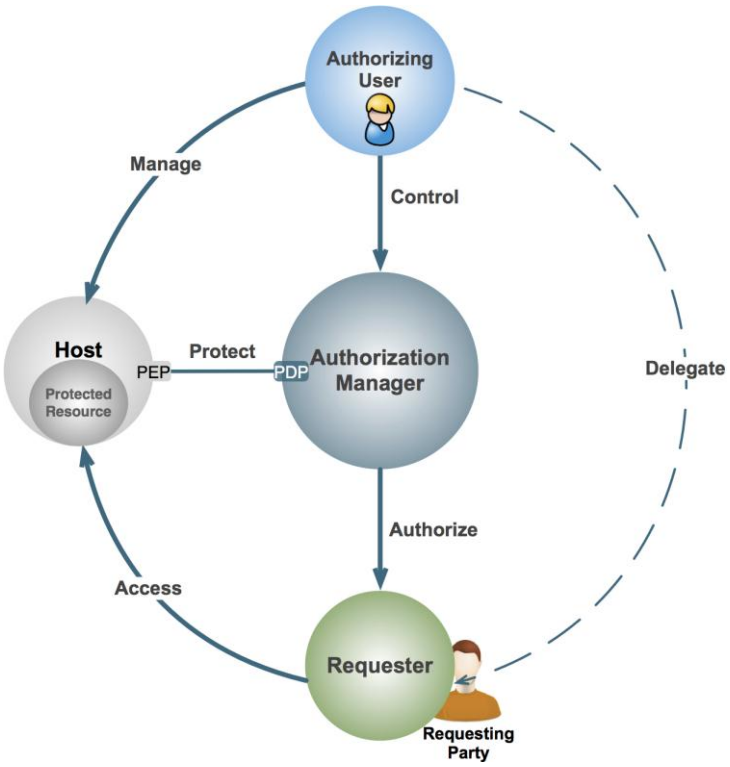

Figure 7: UMA architecture.

An authorizing user arranges to externalize resource protection from their chosen set of resource hosts (policy enforcement points) to an authorization manager or AM (policy decision and administration point), configuring the latter with policies that control how it makes decisions about delegation of access authorization when a requester attempts to access a protected resource. The requesting party "behind" the service running the requester endpoint may be the same person as the authorizing user, a different person, or a legal entity (such as a corporation).

For SWELL user-controlled access to any personal data or resource that has value, including a user's context, is required. UMA is a tool for providing that control.

For example, web user Alice (authorizing user) might authorize an online service (requester) to gain one-time or ongoing access to a set of personal data including her DirectLife Activity Monitor output data stored at a personal data service (host), having already instructed the host to check with her authorization decision-making service (AM) whenever requesters come calling. The requesting party might be a health service provider that advices Alice in recuperating from her cancer treatment.

Human understanding of the policies that are set is supported in UMA through the use of resource *scopes*. A scope is a bounded extent of access that is possible to perform on a resource set, which consist of a human-readable name, and optionally a graphical icon representing the resource set.

The conditions under which an Authorization Manager grants a requester access to a resource referenced by such a scope depends on the policies that the user has set.

The nature of these policies, however, is out of scope for UMA. An Authorization Manager must base its access grants (by giving out a requester permission token - RPT) on user policies. E.g. a user policy could for example be to give “view” access to a certain resource only to requesters that can prove that they have an agreement with the owner of the data. [15] These user policies can, for example, be defined in XACML.

4 Case study

The combination of context and authorization is rated as promising by many analysts. Gartner, for instance, in its Hype Cycle for Context Aware Computing [18], places *Context-Aware Security* close to the so-called peak of inflated expectations. Gartner rates the maturity of Context-Aware Security as "adolescent" and the benefit as "transformational". This chapter motivates and illustrates the power of the combination of context and authorization by a use case.

In SWELL Deliverable D1.3a ("User needs study") a scenario is defined based on a study of health behaviour models, available applications and interventions, techniques for measuring physical activity, and user needs. Here we apply the previous discussion of context-aware privacy policies to this scenario, to see whether or not using context-sensitive privacy management is a feasible option for such an application. Since the applications envisioned in the SWELL project are still in an early design stage, this scenario description is currently the best option for an analysis of a fit for context-aware privacy management for these applications.

In the scenario a man named Peter is introduced, aged 40. He was recently diagnosed with Chronic Obstructive Pulmonary Disease (COPD). The reduced lung capacity that he has because of this results in a vicious cycle: because he is tired sooner he avoids exercise, and because he avoids exercise his physical condition deteriorates further. In the scenario, Peter visits his doctor and agrees to using an activity coach that may help him in altering his habits and breaking through this vicious cycle.[3]

The general idea behind the activity coach is that it will monitor the behaviour and physical condition of Peter, and try to persuade him to improve his health habits. The activity coach actually consists of a number of different components:

- Web-based portal (with logins for Peter, his physiotherapist, and possibly peers). The portal allows Peter to view the recorded level of physical activity over time, fill in a set of short questionnaires to attain a baseline measure of his stage of change and susceptibility to the various persuasion principles, and communicate with his therapist or peers.
- Smart phone equipped with a triaxial accelerometer, GPS
- Heart rate monitor

These components are visualized in Figure 8.

[3] See for a more extensive description of the scenario the final chapter of SWELL Deliverable D1.3a ("User needs study")

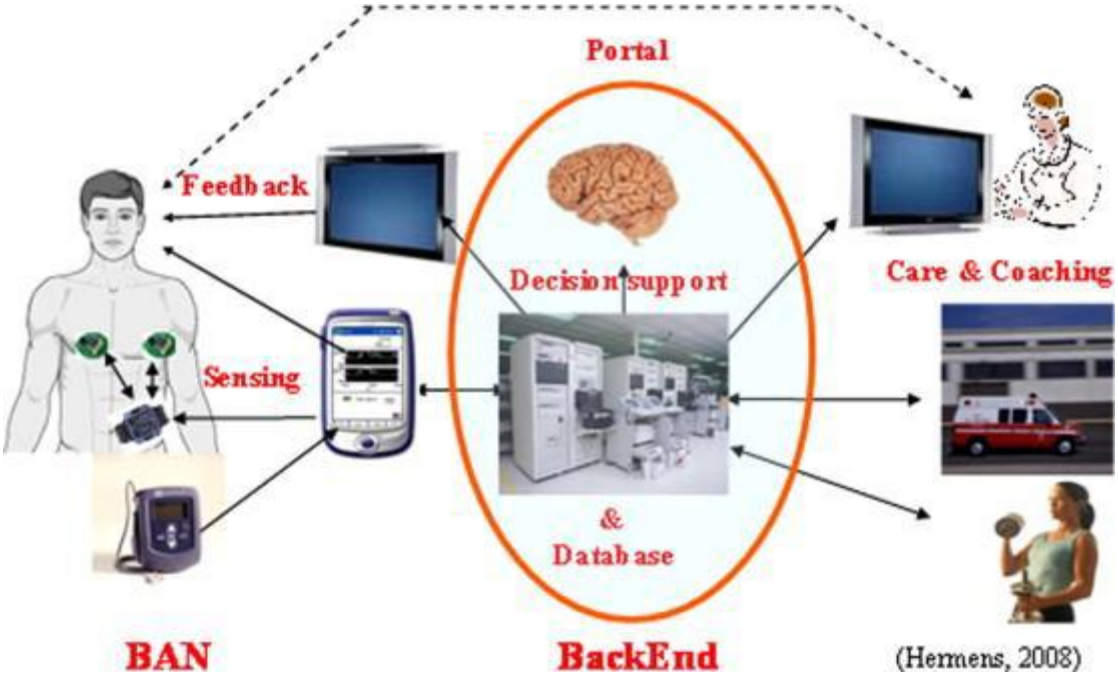

Figure 8: Components in the scenario SWELL application.

4.1 Personal information usage and sharing

It is obvious from the scenario description that the application that is envisioned in the scenario involves the handling of large amounts of data about Peter's behaviour, health habits, physical state, his susceptibility to various persuasion techniques, personal communications, and more. This information is collected within a specific context: a therapy aimed at improving Peter's health habits. If legal requirements are met, Peter will have actively and explicitly consented to being monitored and to the using of his personal data for the therapy before he engages with the health coach. He will likely assume that the data is treated confidentially and not shared with third parties without his consent.

Going back to the discussion in chapter 1, Peter may or may not object if the personal information that is collected would be used in entirely different contexts. For example:

- Health insurance companies may be interested in knowing more about Peter's medical conditions, and possibly use behavioural information to judge whether to pay out certain insurance claims (e.g. based on the argumentation that Peter did not exercise not enough to avoid health problems).
- Advertising agencies may like to know more about Peter's medical conditions, behaviour, interests, activities, etcetera, to better target advertisements to his personal situation.
- His employer may want to know about his medical conditions, behaviour, etcetera, to determine whether he has "good enough" reason to be on sick leave, to know whether he is suitable for promotion or to keep him eployed, etcetera.
- His wife or a friend may want to support him in changing his behaviour and may like to know more about his behaviour or level of physical activity.

Using the detailed personal data collected in the context of the therapy in one of these other contexts without explicit permission from Peter constitutes a privacy violation, which may have very real negative consequences for Peter. Success for a SWELL application such as this is only possible when the privacy of the person undergoing therapy is protected, and contextual integrity[4] is maintained. However, Peter may also choose to share the personal information in specific contexts, for example the therapeutic context, sharing the information with his wife, or to receive targeted advertisements if he likes.

In order to make an informed choice about what to share in what context, Peter needs to know three things: what information will be revealed, for what purpose, and with what parties (see Figure 1). The tables below shows a short inventory of these three things in three of the contexts that were described (this is just a sample, in a practical implementation this will be a much longer list):

Therapy context		
Information	**Purpose**	**Parties**
Level of physical activity	Automated feedback for improving health habits	Digital health coach
Responsiveness to persuasion methods	Optimizing therapeutic approach	Digital health coach, therapist
Stage of change (of health habits)	Optimizing therapeutic approach	Digital health coach, therapist
Level of physical activity	Personal feedback for improving health habits	Therapist
Heart monitor	Signaling health emergencies for fast response	Emergency services

Family or friend support context		
Information	**Purpose**	**Parties**
Level of physical activity	Personal feedback for emotional support	Friends or family
Stage of change (of health habits)	Personal feedback for emotional support	Friends or family

Targeted advertising context		
Information	**Purpose**	**Parties**
Responsiveness to persuasion methods	Choosing optimal marketing strategy	Marketing company or department
Medical condition	Advertising products relevant for medical condition	Marketing company or department

[4] As defined by Helen Nisenbaum (see chapter 1).

4.2 Policies for different contexts

To protect Peter's privacy, contextual integrity must be maintained. This means that for the different contexts in which his personal information may be used, he has control over what is shared with whom, and for what purpose (i.e. Peter defines the norms that apply to a context with regards to personal information sharing). To make this possible, the health coach application (consisting of a web server application and a smart phone application) must be able to maintain privacy policies set by Peter for the different contexts. This poses us with a number of questions:

1. how to determine what context the user is in
2. how to differentiate this context from other contexts;
3. when this differentiation must be made, and by who or what; and
4. how Peter may put the choices on sharing of his personal information into a privacy policy in a user-friendly manner.

In part, maintaining contextual integrity is not a matter of privacy policies. For example, protecting personal data from theft by cybercriminals requires adequate information security in all contexts, and is not necessarily a matter of choosing privacy policies. In some cases, however, privacy policies may be used to define what personal information may be shared in what context, for example:

- When Peter is at work, he may not want to receive feedback from the health coach.
- If the web portal has different users (e.g. Peter's peers also taking part in the therapy), he may want to limit certain data to a specific social context (i.e. specific users or groups of users).
- Peter may not want to be monitored at some times during the week, e.g. when during some evenings he wants to relax and not have the feeling that he is being watched continuously.

Here a number of context parameters become visible: location (work, home), social context (e.g. certain forum peers), and day of week & time of day. These context parameters may be used to formulate privacy policies for specific contexts. In normal language such policies may be:

> *"Deactivate the mental coach feedback when I am near the location where I work, but continue the monitoring."*
>
> *"Only allow John and Mary to access my physical activity data on the web-based portal."*
>
> *"Turn off all monitoring during this evening."*

In principle, the contextual parameters required to enforce these policies automatically are available. However, there are several accuracy issues concerning the acquired parameter data, especially when it comes to location information. Location data is often not very accurate, typically having inaccuracies of meters (e.g. with GPS) to tens or hundreds of meters (e.g. with location based on nearby wifi or 3G networks). Nevertheless, the available sources of location information (e.g. nearby work-related wifi network plus proximity to certain GPS coordinates) may be sufficient for the purposes of context-sentitive privacy policy management this scenario: to distinguish between "work" and other contexts.

In chapter **Error! Reference source not found.**, the policy language XACML was discussed. If this policy language were to be used for context-sensitive privacy policies this SWELL application, a policy

that would represent some of Peter's choices with regards to his privacy could be formulated as follows (this is a simplified formulation for demonstration purposes):

```
[…]
 <Rule RuleId="temporaryFullPrivacy" Effect="Deny">
    <Description>Peter can disable all monitoring for a time period.</Description>
    <Target>
      <Actions>
        <Action>
          <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
              healthcoach.sensors.heartrate
              healthcoach.sensors.gps
              healthcoach.sensors.accellerometer
            </AttributeValue>
            <ActionAttributeDesignator>
              view
            </ActionAttributeDesignator>
          </ActionMatch>
        </Action>
      </Actions>
    </Target>
    <Condition>
      <Apply FunctionId=" urn:oasis:names:tc:xacml:1.0:function:dateTime-greater-than">
         <EnvironmentAttributeDesignator
          AttributeId="healthcoach.sensors.clock.currentTime"
          DataType="http://www.w3.org/2001/XMLSchema#dateTime">
           2012-12-5T18:00:00
         </ EnvironmentAttributeDesignator >
      </Apply>
    </Condition>
  </Rule>

[…]
```

While it appears very well possible to define context-specific privacy policies for the scenario SWELL application in this manner, the XACML policy is very verbose, technical, and may be difficult to comprehend for the average user of a SWELL application. Hence, this policy would not be formulated directly by the user (in this case, Peter), but instead be generated based on a more intuitive interface. A mock-up of such an interface is shown in Figure 9.

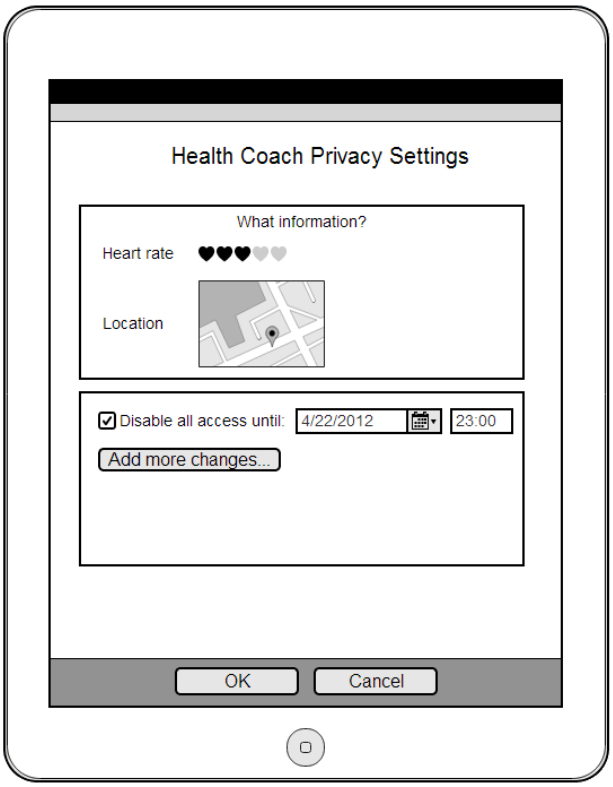

Figure 9: Example of privacy policy settings.

4.3 Context-sensitive privacy management architecture

As discussed in chapter 2, a privacy policy management system would consist of a number of key local components: a Policy Enforcement Point (PEP) that would enforce the resource access policies, a Policy Decision Point (PDP) that evaluate whether or not a policy rule applies, a Policy Admin Point (PAP) that would allow the user to set policies, and a Policy Information Point (PIP) that would provide the necessary (context) information.

In our simple scenario, there are a number of different ways in which these components may be implemented. There are two choices where a PAP can be placed:

1. On the mobile device: Peter can edit his privacy policy on the mobile device itself.
2. On the application server: Peter can edit his privacy policy while visiting the web portal.

These choices are not mutually exclusive: they may be combined, or each option may be used as a PAP for a subset of all policies. E.g. a mobile device PAP may be used to set policies regarding the sharing of sensor data with the outside world, while the web application PAP may be used to set policies on whom to share data stored on the web server with. Similarly, a PEP and PDP is required on both places: the mobile device and the web portal, assuming that personal information is stored or generated in both places. Most of the context information will be proviced by the mobile device, and this would be a logical place for a PIP.

Privacy policy management is further automated via OAuth. In the case of obtaining Peter's consent before the access token is issued to e.g. the mental coach, the OAuth authorisation server effectively plays the role of the XACML PAP at which the policy is defined and subsequently stored as an XACML

policy. In this case, the XACML policy might record the fact that Peter consented to the coach being able to obtain access to heart rate and location information only at certain times of the day (e.g. between 7:00 and 18:00 hours).

The UMA protocol provides Peter an overview of the consents given to third parties like the mental coach, John and Mary or his boss. Basically, UMA could provide Peter an policy self management interface on top of the XACML PAP.

5 Conclusion

A typical SWELL application involves collecting and handling large amounts of sensitive personal information. Maintaining the norms that the user sets with regards to sharing or using this information is an essential precondition for user trust, and for the application to become a success.

The norms that the user epects to be maintained vary in different contexts. The initial case study of a typical SWELL application shows that there is some promise in automating part of the context-sensitive privacy policy management.

Context information is usable for SWELL-specific of use cases and can technically be embedded in privacy control processes of applications via the XACML policy framework. The types of context information that can be readily applied today (such as *time-of-day*, *location*, *device-type*, *network*, *physical access*, and *proximity*) comprise only a small subset of all imaginable context information types. Still, the use case shows that these context sources can already be useful, for instance to implement more fine-grained access to privacy-sensitive systems.

The resulting XACML policies contain variables representing various types of context information. In order to not introduce too much complexity in the resulting policies, an appropriate level of abstraction needs to be chosen so that policies do not become overly complex. On the other hand, policies need to take into account that the quality of context information may not be perfect, so the level should not be too high. This results in a trade-offs in terms of complexity and flexibility. Once defined, policies may be given a name and reused for different applications, in which case a XACML based centralized architecture makes sense.

Emerging protocols like OAuth and UMA compose nicely with XACML and provide a more user centric and controlled approach to privacy policy management.

Some points of attention with regard to the context-sentitive policy management are:

- Usability: are the policies easy to configure?
- Policy accuracy: does the formulated policy result in behavior that the user expects and wants?
- Contextual parameter accuracy: some parameters may not be trustworthy or accurate enough to ensure reliable knowledge about what context a user is in.

These aspects remain to be validated in future user experiments with applications that take context aware privacy policies into account.

This publication was supported by the Dutch national program COMMIT (project P7 SWELL).

6 References

[1] Nissenbaum, H. (1998). Protecting Privacy in an Information Age: The Problem of Privacy in Public. Law and Philosophy, 17(5/6), 559. doi:10.2307/3505189

[2] James Rachels. (1975). Why Privacy is Important. Philosophy & Public Affairs, 4(4), 323–333.

[3] Rubinstein, I., & Good, N. (2012). Privacy by Design: A Counterfactual Analysis of Google and Facebook Privacy Incidents. New York. Retrieved from http://papers.ssrn.com/sol3/papers.cfm?abstract_id=2128146

[4] Lederer, S., Hong, I., Dey, K., & Landay, A. (2004). Personal privacy through understanding and action: five pitfalls for designers. Personal Ubiquitous Comput., 8(6), 440–454.

[5] The 2011 Gartner Scenario: Current States and Future Directions of the IT Industry, Gartner Research, ID Number: G00209949, January 2011.

[6] Hype Cycle for Emerging Technologies 2010, Gartner Research, ID Number: G00205757, 2010.

[7] Forrester Research, The Future Of Mobile Is User Context, July 2011.

[8] A. K. Dey, and G. D. Abowd, "Towards a better understanding of context and context-awareness," GVU technical report GIT-GVU-99-22, College Computing, GA Institute of Technology (1999).

[9] Raimo van der Klein, Claire Boonstra, Maarten Lens-FitzGerald, sprxmobile, http://www.slideshare.net/Thinkmobile/contextual-services-in-mobile-presentation (2011)

[10] K. Henricksen, J. Indulska, A. Rakotonirainy, Modeling context information in pervasive computing systems, in: 1st International Conference on Pervasive Computing (Pervasive), in: Lecture Notes in Computer Science, vol. 2414, Springer, 2002.

[11] Claudio Bettini, Oliver Brdiczka, Karen Henricksen, Jadwiga Indulska, Daniela Nicklas, Anand Ranganathan, Daniele Riboni, A survey of context modelling and reasoning techniques, Pervasive and Mobile Computing, Vol. 6, No. 2. (09 April 2010), pp. 161-180.

[12] T. Buchholz, A. Küpper, and M. Schiffers, "Quality of Context Information: What it is and why we need it," Proceedings of the 10th International Workshop of the HP OpenView University Association (HPOVUA'01), Vol. 2003, Geneva, Switzerland, July 2003.

[13] eXtensible Access Control Markup Language (XACML), see http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml.

[14] Kantara UMA Work Group charter, see http://kantarainitiative.org/confluence/display/uma/Charter.

[15] UMA Core Protocol Version 1.0, see http://kantarainitiative.org/confluence/display/uma/UMA+1.0+Core+Protocol.

[16] Kantara UMA Work Group charter, see http://kantarainitiative.org/confluence/display/uma/Charter.

[17] UMA Core Protocol Version 1.0, see http://kantarainitiative.org/confluence/display/uma/UMA+1.0+Core+Protocol.

[18] Hype Cycle for Context Aware Computing 2011, Gartner Research, ID Number: G00213793, 2011.