MASTER

AAL security and privacy
Transferring XACML policies for end to end access and usage control

Vlamings, H.G.M.

Award date:
2010

Disclaimer
This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
AAL Security and Privacy

Transferring XACML policies for end to end access and usage control

H.G.M. Vlamings
Eindhoven, September 2010

Master Thesis
Eindhoven University of Technology
Department of Mathematics and Computer Science

Computer Science and Engineering program
Information Security Technology track

Committee members:
prof. dr. M. Petkovic (supervisor)
ir. R.P. Koster (tutor)
dr. D.S. Jarnikov PDEng

Carried out at:
Philips Research, Information & System Security group
Abstract

The growing share of elderly people within societies has a big impact on healthcare expenses. Furthermore the quality of live for these elderly people degrades when they can no longer live an independent life. Several research projects propose so called Ambient Assisted Living systems and services to provide a solution for that problem using information and communication technology.

Such systems will deal with a large amount of information that is privacy sensitive and can be of critical importance to the health and wellbeing of its users. Consequently AAL systems must be designed with strong attention to security in order to protect the confidentiality, integrity and availability of its resources.

Existing AAL frameworks show some serious shortcomings with respect to the security controls they offer. An important challenge is to enable users to stay in control of their data in a setting where that data is exchanged between several different service providers and where users want to share data with their friends and relatives.

This requires a policy language that is expressive enough to capture user consent and target specific data objects. It also requires protocols for transferring policies along with the data they govern and means to securely enforce these policies.

This report presents a high level security architecture that describes a set of security controls that must be offered by AAL frameworks and proposes the advanced authorization architecture for AAL. The advanced authorization architecture solves a number of shortcomings of the existing MPOWER framework. It incorporates the eXtensible Access Control Language and defines a protocol for transferring policies in order to support end to end access and usage control in AAL systems.
## Contents

1. Introduction .................................................................................................................. 3  
   1.1. Objective ................................................................................................................ 3  
   1.2. Problem description ............................................................................................ 3  
   1.3. Solution approach .............................................................................................. 3  
   1.4. Intended audience ............................................................................................... 4  

2. AAL Framework Overview .......................................................................................... 5  
   2.1. Amigo .................................................................................................................. 5  
   2.2. Continua .............................................................................................................. 7  
   2.3. GENESYS ........................................................................................................... 8  
   2.4. MPOWER .......................................................................................................... 9  
   2.5. OASIS ............................................................................................................... 12  
   2.6. PERSONA ......................................................................................................... 13  
   2.7. SOPRANO ......................................................................................................... 14  
   2.8. Discussion and Summary .................................................................................. 15  

3. Scenarios .................................................................................................................... 16  

4. Privacy and security requirements ............................................................................. 17  
   4.1. Stakeholders and assets ...................................................................................... 17  
   4.2. High level privacy and security requirements .................................................... 18  
   4.3. Technical privacy and security requirements .................................................... 19  

5. High level security architecture .................................................................................. 20  
   5.1. High level AAL architecture .............................................................................. 20  
   5.2. Security mechanisms ....................................................................................... 21  

6. Authorization limitations and improvements ............................................................ 23  
   6.1. Current limitations ............................................................................................. 23  
      6.1.1. Policy language .......................................................................................... 23  
      6.1.2. Policy management .................................................................................... 24  
      6.1.3. Policy enforcement .................................................................................... 24  
   6.2. Possible improvements ..................................................................................... 24  
      6.2.1. Policy language .......................................................................................... 25  
      6.2.2. Policy management .................................................................................... 25  
      6.2.3. Policy enforcement .................................................................................... 27  

7. Advanced authorization architecture ......................................................................... 29  
   7.1. Policy language ................................................................................................. 29  
      7.1.1. Language structure .................................................................................... 29  
      7.1.2. XACML in AAL .......................................................................................... 32  
      7.1.3. Components and data flow ....................................................................... 33
1. Introduction

Elderly people make up an increasing share of the societies in Europe and North America. These people often experience problems with living an independent life and may eventually have to move to an elderly care facility. This not only decreases their quality of life, but the intensive care they require also comes with great costs [1].

Several research projects propose information and communication technology (ICT) solutions to support elderly in living an independent life, improving their quality of life and reducing the need for expensive personal care. These solutions are often referred to as Ambient Assisted Living (AAL) systems, since they assist people to lead their life and are integrated with their everyday environment.

Such AAL systems may comprise mechanisms for remote health monitoring and advice, indoor and outdoor location tracking and navigation, emergency detection and response and various communication mechanisms to allow assisted persons to stay in touch with their friends and relatives. Consequently AAL systems will deal with vast amounts of privacy sensitive data gathered from sensors and user input.

In order to develop a successful AAL system, security and privacy issues and data protection legislation will have to be taken into account during its design.

1.1. Objective

This report presents the results of a project that aims to come up with a security architecture that can be integrated with an AAL system to provide the necessary mechanisms and components to securely protect the sensitive assets of the system from inappropriate use. The project focuses on access and usage control mechanisms that can cope with the highly decentralized setting of AAL systems and enable users to stay in control of their data.

1.2. Problem description

The main problem addressed by the project is the question what a security architecture for AAL should look like. Which security controls are needed and what technologies can be used to implement them?

A specific aspect of this problem that is addressed in much more detail is the question of how to support end to end access and usage control of personal data in a decentralized setting. How can policies that govern some data be enforced if the data is transferred to another system? How can users stay in control of where their data goes and who can access it? Are there any existing standards and protocols that can be applied, or is it necessary to design new ones?

1.3. Solution approach

The project consists of a number of steps towards a solution to these problems. These steps include an analysis of the documentation of some AAL frameworks, the analysis of the open source code of the MPOWER framework, a literature study on security requirements for such systems, the design of a high level security architecture, the design of an advanced authorization architecture and a security analysis of this authorization architecture. This report presents the results of the project according to the following structure.

Chapter 2 describes a number of AAL frameworks to get an overview of the state of the art and to show which security controls these frameworks offer. Chapter 3 describes a selection of scenarios that capture some important aspects of AAL systems and form the basis for the stakeholder analysis and security and privacy requirements in chapter 4. The report describes a high level security architecture that covers these requirements in chapter 5, before looking at authorization in more detail.
To illustrate the challenges of authorization in AAL systems, chapter 6 discusses the limitations of and possible improvements to the authorization approach of MPOWER. The “advanced authorization architecture for AAL” that chapter 7 proposes incorporates these possible improvements to support end to end access and usage control of personal data in a decentralized setting. The security analysis in chapter 8 shows the improvements achieved by the proposed architecture and points out some limitations that need to be solved in future work.

Chapter 9 concludes the report with a summary of the proposed solutions and discussion of some remaining challenges with respect to AAL security.

1.4. Intended audience

This report presents the results of the author’s graduation project for the Information Security Technology master program at the Eindhoven University of Technology and as such forms the master thesis of the author. The work is carried out at Philips Research in the light of two European AAL research projects, UniversAAL and CoLiving, in which the Information and System Security group of Philips Research is involved. The report is targeted at security researchers and AAL framework developers. Some understanding of general computer science and information security principles is assumed. References to detailed background information are provided in footnotes and references where necessary.
2. AAL Framework Overview

There have been and still are quite a number of projects that work on developing an AAL framework. This chapter gives an overview of seven of them, focusing on the privacy and security mechanisms they offer. The information presented here is mostly gathered from project websites, deliverables and publications and serves as a means to understand the current application and technical approaches in the AAL.

The seven projects that have been studied are Amigo, Continua, GENESYS, MPOWER, OASIS, PERSONA and SOPRANO. The following sections contain short descriptions of the goals, target domains and progress as well as a more detailed description of the framework architecture and the proposed privacy and security features for each of these projects.

A summary of the security and privacy features that were encountered and a short discussion thereof in section 2.8 concludes the overview.

2.1. Amigo

The Amigo project¹ is a European project under the Sixth Framework Programme² (FP6). It ran from 2004 till 2008 and its goal was to improve interoperability in home networking, not only for healthcare and safety systems, but also for entertainment systems. It is targeted at a broad range of users, not just elderly people or people with disabilities.

---

² [http://ec.europa.eu/research/fp6/](http://ec.europa.eu/research/fp6/)
**Amigo Architecture**

Since the Amigo project targets highly heterogeneous and dynamic environments, Amigo follows a Service Oriented approach. Instead of incorporating a specific service oriented platform, it supports multiple platforms, such as UPnP\(^3\) and WS\(^4\), each with their own communication and discovery mechanisms. Service discovery and interaction mechanisms from the Amigo Base Middleware provide the necessary glue to enable interoperability between applications developed for these different platforms.

The Base Middleware also provides authentication and authorization mechanisms which applications can access through the so called Security & Privacy Service, which is part of the Intelligent User Services that are built on top of the Base Middleware (Figure 1)[2][3].

**Amigo Privacy & Security**

The security architecture of Amigo specifically targets the networked home environment [2]. The Security & Privacy Service provides an interface to mechanisms for authentication and authorization. These mechanisms are based on the Kerberos\(^5\) mechanism and are provided by the authentication service and the authorization service that are part of the Base Middleware.

Devices and users can authenticate themselves to the authentication service, which will then provide them with a security token, which has a similar function as the ticket-granting-ticket from Kerberos.

This token can from then on be used throughout a session to request access to all kinds of services at the authorization service. With this mechanism, Amigo provides a Single-Sign-On (SSO) mechanism for users and devices.

The authorization service makes an access control decision and either grants or denies the user or device access to a service. Access is granted by providing a service-specific token, also known as an authorization token. The authorization services implements Role Based Access Control (RBAC). Users and devices have roles, which in turn have privileges to access certain resources.

These roles and permissions are chosen in such a way that they can be understood by Amigo users, so that no expert knowledge is required to maintain the RBAC policies.

To further enhance this RBAC mechanism, the Amigo project has developed a framework for making the access control policies context aware [4]. Their CAPriCE (Context Aware Privacy preferences Collection and Enforcement) Framework is basically an extension of the XACML\(^6\) standard. The context awareness that they added consists of two aspects.

First, the context of the *situation* in which an access request occurs is taken into account while making the access control decision. This way it is possible to model the situation where a medical professional is only allowed access to personal information if a sensor at the client’s house has detected a medical emergency. As long as there is no emergency, the personal data is not accessible.

Second, the CAPriCE framework allows policies to capture the *precision* of the context to which a user or device is granted access. This allows privacy sensitive information like someone’s current location to be presented in a detailed way when necessary, but in a restricted, privacy preserving, way otherwise. For instance, when no emergency is detected, a medical professional can see

---

\(^3\) Universal Plug & Play, [http://www.upnp.org/](http://www.upnp.org/)

\(^4\) Web Services Architecture, [http://www.w3.org/TR/ws-arch/](http://www.w3.org/TR/ws-arch/)

\(^5\) [http://www.ietf.org/rfc/rfc1510.txt](http://www.ietf.org/rfc/rfc1510.txt)

whether his client is at home or not, but cannot see in which room the client is. When there is an emergency, the location info is provided with more detail, such that the medical professional can see in which room the client is.

Apart from access control through authentication and authorization, Amigo identifies the need for mechanisms for providing confidentiality and integrity of data that is being transferred. The need for such mechanisms is decided by communication endpoints, and therefore the Amigo framework does not provide these mechanisms itself. The reference architecture suggests some candidates: SSL/TLS\textsuperscript{7} and WS-Security\textsuperscript{8}. Key distribution for such solutions can be supported by the authentication service and the security tokens.

2.2. Continua

The Continua Alliance\textsuperscript{9} differs from the other projects in the sense that it is not a research project developing an AAL framework, but an alliance of healthcare and technology companies that selects connectivity standards and defines guidelines for developing personal healthcare applications. This way Continua attempts to increase interoperability between devices and services in that field. The project targets three main application areas: supporting elderly people to live independently, supporting people with a chronic illness to manage their situation and supporting people to get the most out of their fitness programs.

The first set of guidelines was published in February 2009 and currently a new version is under construction.

Continua Architecture

Although Continua is not really developing an architecture, they do present a reference topology of the systems they are targeting (Figure 2). From this topology it becomes clear that there are four communication interfaces that they attempt to cover in their guidelines.

The Personal Area Network and Local Area Network that connect devices like for instance medical sensors to a central Application Hosting Device (AHD), the Wide Area Network that connects the AHD to for instance a Telehealth Service Center (TSC), and finally the Health Record Network, which is an interface over which for instance the TSC communicates with a central electronic or personal health record service.

Continua Privacy & Security

The Continua V1.5 guidelines recommend mechanisms that could or should be used on each of those interfaces, as well as mechanisms that provide end-to-end security [5].

For some of the interfaces, the selected communication

\textsuperscript{7} Secure Sockets Layer / Transport Layer Security, \url{http://www.ietf.org/dyn/wg/charter/tls-charter.html}

\textsuperscript{8} Web Service Security, \url{http://www.oasis-open.org/committees/wss/}

\textsuperscript{9} \url{http://www.continuaalliance.org/}
protocol provides some security mechanism. For instance on the PAN interface, the Bluetooth Health Device Profile\textsuperscript{10} is used, which provides mechanisms for secure pairing of devices (authentication) and encrypted communication (confidentiality protection). The Zigbee protocol which is proposed for the LAN interface offers mechanisms for confidentiality and integrity protection.

Mechanisms like TLS and S/MIME\textsuperscript{11} are recommended for WAN and HRN interfaces and provide confidentiality and integrity. Furthermore, on these interfaces the guidelines also recommend compatibility with the IHE ATNA\textsuperscript{12} audit mechanism. The use of SAML v2.0\textsuperscript{13} assertions through WS-Security headers is recommended on the WAN interface to provide authentication.

### 2.3. GENESYS

The GENESYS project\textsuperscript{14} is a European research project funded by the Seventh Framework Programme\textsuperscript{15} (FP7). Its goal was to develop a cross-domain reference architecture for embedded systems. Automotive, avionic and tele-health applications are among the target domains. As such the GENESYS project was not targeted specifically at AAL.

The project ran from January 2008 until July 2009 and resulted in a reference architecture, a number of prototypes and a development methodology.

![GENESYS Architecture Model](Figure 3 GENESYS Architecture Model, from the GENESYS book [6])

**GENESYS Architecture**

The GENESYS project has taken a component oriented approach targeted at improving reusability and interoperability of embedded systems. As is depicted in Figure 3 the components


\textsuperscript{11}Secure/Multipurpose Internet Mail Extensions, http://www.ietf.org/dyn/wg/charter/smime-charter.html

\textsuperscript{12}http://wiki.ihe.net/index.php?title=Audit\_Trail\_and\_Node\_Authentication

\textsuperscript{13}Security Assertion Markup Language http://www.oasis-open.org/committees/security/

\textsuperscript{14}http://www.genesys-platform.eu/

\textsuperscript{15}http://cordis.europa.eu/fp7/
are connected through a platform containing a set of so called Core Platform Services. These services provide mechanisms for the components to interact with each other. Apart from these Core Services, the GENESYS reference architecture also describes a number of Optional Services and Domain Specific Services. These can be implemented inside components or as separate components and are included in a system if necessary [6].

**GENESYS Privacy & Security**

Security mechanisms in the GENESYS reference architecture template are spread over several levels [6]. By design, components can only communicate through the message passing mechanisms provided by the core services. This provides a clear separation of components. Tamper resistant hardware identification is part of the Core Services. Tamper resistance in this case is defined as resistant against changing a chip's hardware identification without destroying the chip itself.

An extensive list of security services is found on the level of Optional Services, these services are provided by a security component, which could implement them in software, but if a higher level of security is required, a tamper resistant hardware solution could be used as well. The services such a component provides consist of:

- Secure Key Management, which provides encrypted storage and transfer of key material. Root key that is used for encrypting other keys is stored in a secure, tamper resistant place that only the Key Management service can access.
- Encryption and Decryption, which provides mechanisms for encrypting and decrypting messages passed from one component to another.
- Random Number Generation, which provides the random numbers required for instance for key generation.
- Service Authentication, which governs the registration of new services.
- Secure Boot Service, which performs authentication and integrity checks on downloaded code before it is executed on a component.
- Service Access Control, which involves granting and controlling access rights to use a particular service.

Furthermore, on the level of Domain Specific Services, the following privacy related services are mentioned:

- Identity Management, which manages the user accounts of a system (including aliases and groups) and takes care of verification (authentication) of physical users and linking them to a user account.
- Trust Management, which links both services and users to a trust level by providing and verifying trust tokens.
- Privacy Protection, which provides anonymity by allowing services to check the trust level of a user while hiding privacy sensitive information about the identity of a user. Furthermore it provides a mechanism to only grant access to personal information with consent of the owner.

**2.4. MPOWER**

Like Amigo, MPOWER is a European research project funded by FP6. The 30 month project started in October 2006 and resulted in a reference architecture, a set of middleware services, a development methodology and two pilot applications. Its goal was “... to create a middleware
platform supporting rapid development and deployment of services for cognitive disabled and elderly”

**MPOWER Architecture**

The reference architecture proposed in the MPOWER project is a Service Oriented Architecture based on IBM’s SOA reference architecture [7]. Figure 4 depicts a typical target system for MPOWER, consisting of a smart home that contains all kinds of sensors and other devices, represented by services, which is connected to a care center that has its own collection of services running. All these services can communicate directly or through a set of common services provided by a separate node. MPOWER uses the HL7v3 messaging standard for this communication.

Apart from this reference architecture the MPOWER project also produced a set of what they call MPOWER Middleware Services. These services implement some basic functionality, specific to the smart home and care center setup, and can be reused and distributed across the network of an MPOWER system. A set of security services is part of these MPOWER Middleware Services.

---

16 http://www.sintef.no/Projectweb/MPOWER/
18 Health Level Seven version 3, http://www.hl7.org/
MPOWER Privacy & Security

The most important functionality that the security services in the set of MPOWER Middleware Services provide is access control. The access control system consists of mechanisms for managing users, roles and access policies as well as authentication mechanisms. These mechanisms are spread out over several components. Security tokens are used to grant access to a service. One of the components of the security services takes care of managing these tokens.

Two authentication methods are implemented in the set of services delivered by the project: password based and digital signature based login. The latter uses the Public Key Infrastructure (PKI) component from the set of security services. This PKI component is also used by mechanisms for secure communication and secure storage, which use encryption techniques provided by an encryption service.

The security services provide an audit component as well. Services can contact it to store and retrieve audit logs. Figure 5 presents an overview of the services that have been developed by the MPOWER project.

![Figure 5 MPOWER Security Services, from MPOWER Overall Architecture [7]]
2.5. OASIS

The OASIS (Open architecture for Accessible Services Integration and Standardisation) project – not to be mistaken for the open standards body that goes by the same name\(^\text{19}\) - is a European research project funded by FP7. As it is a four year project which started in 2008, it is still running.

The project targets elderly people (55+) and aims to “... develop an open and innovative reference architecture, based upon ontologies and semantic services, that will allow plug and play and cost-effective interconnection of existing and new services in all domains required for the independent and autonomous living of the elderly ...”\(^\text{20}\).

OASIS Architecture

Like most of the frameworks studied here, OASIS also follows a service oriented approach. Central to the approach of OASIS is the use of ontologies for providing interoperability [8]. Several components have been developed to create a framework around this approach (see Figure 6). This framework consists of several modules for managing and applying ontologies, but also a module that stores user profiles, a module that supports creation and adaptation of user interfaces and the so called Trust & Security Framework that will be discussed in more detail below.

Figure 6 OASIS Architecture, from OASIS Architecture and Component Specifications [8]

\(^{19}\) [http://www.oasis-open.org/](http://www.oasis-open.org/)

\(^{20}\) [http://www.oasis-project.eu/](http://www.oasis-project.eu/)
OASIS Privacy & Security
The Trust & Security Framework (TSF) module that is a part of the OASIS platform mainly performs user authentication. According to the component specification it also takes care of authorization. The document does however not describe any authorization mechanism.

The authentication mechanism provides a session ID to a user after verifying the user’s credentials (username and password). The session ID uniquely identifies the user and the device which requested authentication. To support this, the TSF module also takes care of the management of all user accounts, by providing mechanisms for registering new user accounts and for editing or removing the profile of existing users [8].

2.6. PERSONA
The PERSONA project[21] is another FP6 European research project. It started in January 2007 and is planned to run until June 2010. It has already resulted in a reference architecture and some pilot sites have been set up in 2009. The goal of the project is to develop a technological platform on which all kinds of services can be built that are targeted at assisting elderly people to live independently.

PERSONA Architecture
The core of the PERSONA architecture is a piece of middleware that is included in every node and makes it “PERSONA-aware” [9]. It抽象s from underlying network technologies and enables communication between nodes by providing a set of 4 logical communication busses. This set of four busses consists of an input bus, an output bus, a context bus and a service bus (see Figure 7).

User input is passed to the relevant nodes over the input bus and System output travels over the output bus. As such these buses connect all kinds of user interfaces to the system. The context and service busses enable inter-component communication in two styles. The context bus provides mechanisms for event based (publish/subscribe) communication, whereas the service bus provides mechanisms for communication in a service oriented fashion.

PERSONA Privacy & Security
In early phases of the project the need for a “Privacy-Aware Identity and Security Manager” was identified [10], but at the moment

only a very limited set of security mechanisms is provided. These mechanisms deal only with user identification, component-level authentication and access control and secure communication.

User identification is achieved by including the user-ID in messages and calls, based on this ID, the component that receives a message or service request itself needs to make access control decisions. The forwarding of a user-ID from component to component is supposedly done in a reliable way, however it is unknown which techniques are used to achieve that reliability.

System administrators manually install shared keys inside components that need to communicate in a secure way. In this way, components that have the shared key are authorized to communicate with each other. Also by being able to decrypt communication a component can show its authenticity.

Administrators can also define security policies on the busses, defining which components can take on which roles on which busses and also defining the allowed or disallowed communication partners of each component.

2.7. SOPRANO

The SOPRANO project\(^\text{22}\) is another project that is still running. This European research project, also funded by FP6, started in January 2007 and is planned to run for 40 months. It is targeted at elderly people taking into account that many of these people suffer from the effects that aging has on their cognitive abilities. The goal of SOPRANO is to develop an AAL system integrating smart-home applications and remote healthcare services with a strong user-centric focus.

Concerning the architecture of the platform the (public) results so far consist of a proposal to use OSGi\(^\text{23}\) as framework upon which the SOPRANO architecture can be built [11].

SOPRANO Architecture

SOPRANO proposes a Service Oriented architecture based on the OSGi framework. On top of this framework sits the SOPRANO middleware which provides several functions (Figure 8) [11]:

- service registry that performs matching using ontologies
- access to and management of context data
- a workflow management system (trigger actions based on events)
- context-aware translation of abstract actions to concrete actions

SOPRANO Privacy & Security

No explicit information on security & privacy features available, however OSGi does provide an optional security layer, which is based on the Java 2 Security Architecture. In short, this provides mechanisms for secure service deployment and code based access control. The first of which prevents unwanted services to be connected to the system and code based access

\(^{22}\) http://www.soprano-ip.org/
\(^{23}\) http://www.osgi.org/

Figure 8 SOPRANO Middleware Framework [11]
control can be used to restrict communication between services.

2.8. Discussion and Summary

The seven frameworks studied in the previous sections all take their own approach at the security and privacy features they provide or propose in their AAL frameworks. Most frameworks provide some kind of access control mechanism, either coarse grained like PERSONA and SOPRANO (component level only), or highly detailed like the context aware approach of Amigo.

Mechanisms for secure communication are also fairly common and this is also one of the areas where the use of standards is mentioned. Continua proposes the most extensive mechanisms with respect to communication security, which is not a surprise given the fact that they are focused on selecting communication standards.

Unfortunately, for many of the features that are mentioned in the project documents, the details of how they are to be achieved are unknown. Maybe even more alarming is that true privacy enhancing techniques like mechanisms that provide anonymity or pseudonymity are hardly seen at all. The GENESYS book mentions something about this, but they seem to only recognize the possible need for such mechanisms and do not really provide them in their framework. Of course the Amigo project approach to access control is interesting with respect to privacy awareness, but access control alone may not be enough to protect privacy sensitive data.

Table 1 presents a brief overview of the security and privacy features that are provided by the studied frameworks. The rightmost column lists the standards that are used to implement these features.

<table>
<thead>
<tr>
<th>Framework</th>
<th>Features</th>
<th>Standards</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amigo</td>
<td>Secure data transfer, RBAC with context aware policies and Single-Sign-On</td>
<td>SSL, WS-Security, XACML</td>
</tr>
<tr>
<td>Continua</td>
<td>Some optional and some required mechanisms mostly for secure transfer of data</td>
<td>TLS, WS-Security, SAML, IHE ATNA</td>
</tr>
<tr>
<td>GENESYS</td>
<td>Extensive set of security/privacy services, details of mechanisms unknown</td>
<td>?</td>
</tr>
<tr>
<td>MPOWER</td>
<td>RBAC, Encrypted storage/transfer, Auditing, details of mechanisms unknown</td>
<td>?</td>
</tr>
<tr>
<td>OASIS</td>
<td>User Management and simple Authentication/Authorization</td>
<td>?</td>
</tr>
<tr>
<td>PERSONA</td>
<td>Encryption, Authentication, Access control on busses, user identification</td>
<td>?</td>
</tr>
<tr>
<td>SOPRANO</td>
<td>Possibly uses the optional security layer of OSGi</td>
<td>Java 2 Security Architecture</td>
</tr>
</tbody>
</table>

Table 1 Summary of privacy and security features
3. Scenarios

This chapter describes a few scenarios that are relevant to ambient assisted living systems. The scenarios are based on scenarios that were identified by the projects that were discussed in the previous section. Appendix A.1 contains the original scenario descriptions. Again the focus is on privacy and security aspects. Therefore this section presents some scenarios where these aspects play a particularly important role.

Scenario 1

User is subscribed to several services. Compatible services can be combined for added value. For instance a collaboration of health monitoring, nutritional advice and activity coaching services for supporting elderly people to stay fit. The services collaborate by exchanging personal data, which is originally gathered from user input and sensor devices (e.g. sensors monitoring health status and activity, possibly including position).

Scenario 2

User carries GPS device when leaving home. Caregiver can monitor position of user in order to be able to send help to the right place in case of an emergency. An emergency can be detected in several ways, e.g. using an alarm button that the user presses, active monitoring by a caregiver or by some automated system.

Extension: user can query for other users in his or her vicinity to continue their outdoor activity together.

Scenario 3

Users, living together in one building, login to an intranet application which allows them to communicate (video chat) with each other. Users can also connect to the intranet from outside (i.e. when they have to stay in a hospital for a while) to stay in touch with friends or relatives.

The intranet system also provides mechanisms for making announcements for activities and for ordering groceries at a supermarket.

Extension: moving towards a social network approach, with users from not just one building but on a much larger scale. This would provide users with the opportunity to stay in touch with all of their friends and relatives once they have to move to special elderly housing. This not only includes mechanisms for exchanging messages, but users can also share sensor data (e.g. to inform others of their health status or position).
4. Privacy and security requirements

Before designing a security architecture that is appropriate for the scenarios presented in chapter 3, a set of relevant privacy and security requirements must be specified. Section 4.1 first describes the stakeholders and assets that are involved. Section 4.2 then derives a set of high level requirements from the viewpoint of these stakeholders and assets. Section 4.3 concludes the chapter with a set of more technical requirements as applicable to a service oriented AAL framework. Security aspects that where identified during this part of the project have also been used as input for the UniversAAL project, as described in appendix A.2.

Some extensive law and regulations exist nowadays that govern data protection. The MPOWER project investigated these, as described in a paper by Jensen et. al. [12] following the security requirements engineering recommendations from Firesmith [13]. Both these papers have provided valuable input for this chapter.

4.1. Stakeholders and assets

This section presents brief descriptions of the stakeholders and assets of AAL systems that are relevant for investigating the privacy and security aspects of such systems. The descriptions for the stakeholders explain their role within the AAL system and their interests on collection and processing of personal data. The paragraph on assets describes the assets and describes why they are sensitive with respect to privacy and security.

**Stakeholders**

- **AAL platform provider**
  
The organization that provides the central components of the service oriented platform. This could for instance include components like a central service repository and PKI facilities. Collection of personal data could be in the form of a central user database (e.g. for PKI or other identity management facilities). Such a central provider might be able to detect with which services a user is registered, which could reveal sensitive information about a person.

- **Service providers**
  
  Service providers use services and components offered by the AAL platform provider to offer services to end-users. Different kinds of organizations could become AAL service providers. This not only includes public health care organizations, but also commercial companies that provide services to support elderly people in an active lifestyle. Consequently some of these providers may have direct contact with the elderly people, while some provide just remote online services. Depending on the kind of service they offer, they will receive, store and process possibly very sensitive information. This information is also forwarded to other service providers in order to collaborate.

- **End-users**
  
The people that use the services provided by the service providers can also be divided in three groups. The most important group is the group of assisted persons, the elderly people that use the AAL system to support an active and independent life. An important part of their usage of such systems consists of communicating with friends and relatives. Consequently these friends and relatives will also make up part of the end-users. Employees of the service provider are the third group of end-users. All end-users, but especially the assisted persons, will submit a lot of personal data to service providers in order to use services. They want the benefits of these

24 http://www.universaal.org/
services, but no negative effects caused by their sensitive personal information falling into the wrong hands.

**Sensitive assets**

- **User data**

  End-users submit all kinds of personal information while using services. This data is stored and processed by the service providers and also exchanged between service providers. The information can range from simple user profiles containing for example a username and an e-mail address to very sensitive data about the health of the assisted person.

- **Sensor data**

  A significant part of the information that is submitted to service providers is gathered through all kinds of sensors (e.g. health, position). This data is not sent to the service provider directly by the sensor, but is usually collected and possibly even processed at some kind of gateway first. The raw sensor data that is transferred from the sensor to such a gateway is a sensitive asset itself, before it becomes “user data”, because of the kind of information that is transmitted.

- **Actuators**

  Some services could be linked to actuators placed in the assisted person’s house or even on the body. The scenarios from OASIS mention actuators that operate lights and household equipment as examples (see appendix A.1 for details). Unauthorized access to such actuators could cause serious safety issues.

- **Private communications between end-users**

  End-users might not only communicate with service providers, but also use the AAL system to have communications directly with each other. Since these communications could be of private nature (i.e. discussing personal information) they are a sensitive asset.

- **Service membership**

  The fact that some person is a member of a certain service can already leak sensitive information about that person.

### 4.2. High level privacy and security requirements

This section presents a set of 4 high level privacy and security requirements, following the CIA-triad (Confidentiality, Integrity and Availability) extended with the principle of non-repudiation. It briefly describes each requirement and mentions the stakeholders and assets it applies to.

- **R1. Confidentiality:** End-users want to share personal information about themselves only with specific parties, unauthorized access must be prevented. End-users also want to retain a certain amount of control over the data they share with other parties and over what happens with this data. This applies to all assets that section 4.1 describes.

- **R2. Integrity:** All stakeholders want to protect the information that is stored and exchanged from unauthorized changes. This applies specifically to service user data, sensor data and commands sent to actuators, but could also apply to some extent to private communications between end-users.

- **R3. Availability:** End-users want to be able to use services whenever they need them. Especially for health related services the consequences of unavailability could be severe.

- **R4. Non-repudiation:** It must be possible to trace back every action on sensitive assets to the person or system component that performed it (e.g. to deal with misuse that could not be prevented with technical security mechanisms).
4.3. Technical privacy and security requirements

This section presents a set of more detailed and more technical requirements. These requirements will guide the design of the security architecture. Each requirement description is followed by a reference to one or more related requirements. These can be high level requirements that are (partly) covered by the technical requirement or dependencies between technical requirements.

TR1. System components must identify and authenticate human users and other system components that want to use them. [required by: TR2, TR3, TR5, TR6]

TR2. System components must authorize human users and other system components before allowing them access to resources to prevent unauthorized access to and disclosure or usage of sensitive assets. [depends on: TR1, covers: R1, R2]

TR3. System components must protect data in storage or communication to ensure confidentiality and integrity of this data. [depends on: TR1, covers: R1, R2]

TR4. All actions on sensitive assets, including failed access attempts, must be logged. [covers: R1, R2]

TR5. It must be possible to trace back actions on sensitive assets to the human or system component that was responsible for this action. [depends on: TR1, covers: R4]

TR6. Mechanisms for Single-Sign-On should be provided in order to increase usability. [extends: TR1]

TR7. Advanced authorization mechanisms should be provided in order to cope with the distributed setting of multiple third-party services exchanging personal data (e.g. policies attached to data instead of simple centralized access control). [extends: TR2]

TR8. Access control mechanisms should be context aware to for instance prevent leaking detailed information in situations where less detailed information would have sufficed. [extends: TR2]

TR9. A usable interface must be provided to capture the consent of the end-user. [extends: TR2]

TR10. Mechanisms to prevent profiling should be provided to prevent leaking of information about who is using which service (pseudonimity, decentralized subscription). [covers: R1]
5. High level security architecture

This chapter presents a high level security architecture for an AAL framework. The architecture is based on the scenarios from chapter 3 and requirements from chapter 4. Section 5.1 describes a high level architecture of an AAL system. Section 5.2 proposes a set of security mechanisms that are part of the security architecture.

5.1. High level AAL architecture

Before describing a security architecture for AAL frameworks it is necessary to describe the general architecture of such systems to be able to refer to certain components and the way they interact. Figure 9 describes the different components that generally make up an AAL system with a service oriented architecture.

![Figure 9 AAL Reference Architecture](image)

Lines between the components represent communication channels. The different “sites” (i.e. the rounded rectangles representing Clients and Service Providers) are connected through the internet. All sites can communicate with all other sites. Clients use (web) services provided by Service Providers, but can also setup direct communication with other Clients. Service Providers respond to service requests from Clients, but also communicate with other Service Providers – in a service oriented fashion - to exchange information as described in Scenario 1. The AAL platform provider is not mentioned explicitly here. If the platform provider does not only provide the components, but also some central services, it behaves similar to a Service Provider.

The services offered by the Service Provider run on the Application Server.

A Data Storage Server is used to store all the user data. Clients, Employees of the Service Provider and other Service Providers will access this data through the Application Server.

The Sensors and Actuators on the Client side are connected to the system through the Client Gateway. In the future sensors and actuators might receive capabilities (e.g. extended user...
interface and support for more appropriate communication protocols) that allow them to communicate to services directly, but for practical reasons some kind of gateway is used at the moment.

The Client Gateway does not only serve as an access point for sensors, but also provides a user interface. In this diagram the Client Gateway is depicted as one standard PC, in practice however, there might be multiple devices involved (e.g. a PC for connecting sensors, a networked TV that provides a user interface and a smart phone for connecting sensors and providing an interface in a mobile setting).

5.2. Security mechanisms

This section proposes a set of security mechanisms to cover the requirements from section 4.3. For each mechanism a reference to the associated technical requirement(s) is given in between square brackets.

Security mechanisms at the client

M1. Secure communication between the Sensors and Actuators and the Client Gateway to provide confidentiality and integrity of data collected from Sensors and commands sent to Actuators. This includes device authentication (secure pairing) and encrypted communication. Protocols such as Bluetooth (Health Device Profile) and Zigbee, as suggested by Continua, can provide such mechanisms. A dedicated wired connection, such as USB, would also provide sufficient protection of confidentiality and integrity. [TR1, TR3]

M2. Identification and authentication of the User at the Client Gateway. The architecture supports multiple modules for identification and authentication, because this can be achieved in several ways (e.g. username/password, biometrics or some form of electronic credentials like a smartcard). [TR1]

M3. The Client Gateway could contact a central identity management service to offer Single-Sign-On functionality. [TR6]

M4. The Client Gateway includes information on the identity of a user in service requests, to allow components at the Service Provider to authorize the request. [TR1, TR2]

M5. The Client Gateway verifies the authenticity of commands sent to Actuators and authorizes them before forwarding the commands to the Actuators. [TR1, TR2]

M6. The Client Gateway maintains a local log of all actions on personal data it is involved in. [TR4]

Security mechanisms at the service provider

M7. Communication channels on the intranet of the Service Provider are secured to provide confidentiality and integrity. This includes authentication of communicating systems through for instance the use of certificates as well as encryption of traffic (e.g. using TLS). [TR1, TR3]

M8. A network firewall restricts external access to the well defined service interfaces of the Application Server, denying direct access to the Data Storage Server. [TR2]

M9. Permission to use the services on the Application Server is based on the identity of the requesting component or person. [TR1, TR2]

M10. An Employee is identified and authenticated by the Workstation. In parallel to the User authentication on the Client side, the Workstation also supports multiple authentication modules. [TR1]

M11. The Application Server provides services and applications for configuring and selecting access and usage control policies that govern the data stored on the Data Storage Server. [TR9, TR2, TR7, TR8]
M12. Requests for access to the Data Storage Server are checked against the policies that are configured to govern the data that is stored on the Data Storage Server. [TR2, TR7, TR8]

M13. Data on the Data Storage Server is secured (e.g. by encrypting it) such that it can not be read other than through a well defined interface that includes M12. [TR2]

M14. Every access request to the Data Storage Server is logged (i.e. the details of the request and information about the decision that was made). [TR4]

**End to end security mechanisms**

M15. The Client Gateways and Application Servers provide mechanisms for setting up secured channels for communication with other sites over the internet. This includes authentication of communicating systems through for instance the use of certificates as well as encryption of traffic (e.g. using TLS). [TR1, TR3]

M16. Access and usage control policies are attached to the data they govern when data is transferred from one provider to another. [TR2, TR7, TR8]
6. Authorization limitations and improvements

Controlling access to data and enforcing obligations that come with access to certain data are the most important tasks that security mechanisms must fulfil in order to protect private data. This chapter describes the limitations of current AAL frameworks with respect to access and usage control and discusses a number of conceptual improvements. Chapter 7 proposes a concrete authorization architecture.

6.1. Current limitations

The overview of AAL frameworks in chapter 2 already mentions that although many frameworks include some authorization functionality, this is sometimes rather limited. To illustrate the limitations of authorization mechanisms in current AAL frameworks this section describes the mechanisms provided by the MPOWER framework and its limitations given the scenarios and requirements from chapters 3 and 4.

Authorization functionality in MPOWER is provided by the security middleware services (Figure 5). When an MPOWER service receives a request message it sends an authorization request to the Authorization Service to verify whether the user that initiated the request is allowed to perform that request. The request to the authorization service contains a security token provided by the user, an identifier of the service and the name of the service method the user requested.

The security token was handed to the user by the Authentication Service after verifying the identity of the user. The Authorization and Authentication Service rely on some of the other security middleware services to manage information on users, their roles and permissions and security tokens.

The limitations of the authorization mechanisms in the MPOWER framework can be divided into three areas: the policy language, policy management and policy enforcement. The next three sections each discuss one of these areas, describing the relevant aspects of the MPOWER approach and the limitations of it.

6.1.1. Policy language

The policy language of MPOWER is based on the Role Based Access Control (RBAC) principle [14]. In short, this means that there is a set of users, a set of roles and a set of permissions. Users are assigned to roles and roles are assigned to permissions. A permission defines an action that is permitted and always applies to one or more resources. This results in the abstract structure shown in Figure 10. The notion of policy is not part of the RBAC principle, but here it is used as a term for a data structure that describes a set of permissions, the roles they are assigned to and the resource they apply to.

![Figure 10 Abstract RBAC model](image_url)

Any system that implements RBAC defines its own interpretation of these entities and therefore its own policy language. In case of MPOWER the resources are restricted to services and the permitted actions are restricted to the methods offered by these services.
An obvious limitation of this approach is that permission assignment is rather coarse grained. A certain role can only be granted access to a certain service method, not to specific data. Consequently a user that is granted access to a method for retrieving medical data can retrieve all medical data that can be accessed through that method. This clearly conflicts with the confidentiality and integrity requirements R1 and R2.

Another limitation of MPOWER’s policy language is that it does not provide mechanisms for data owners to define obligations that have to be met when the data is used. This could for instance include the obligation to remove the data after using it or the obligation to write an entry in the audit log every time the data is used.

The obligation to remove data after use is especially useful in the case where multiple service providers collaborate and exchange sensitive data (as described in Scenario 1). The use of audit logs is useful in all situations where sensitive data is used and when this data concerns someone’s health it’s even a legal obligation [12]. Data owners have much more control over their sensitive data when they can define such obligations in a policy instead of relying on services being implemented in such a way that they write audit logs at appropriate moments.

### 6.1.2. Policy management

Policy management might also become a challenge in the MPOWER set up. When every service provider runs an authentication and authorization service and locally stores users, roles and permissions, these will have to be synchronized somehow to support the collaboration of service providers as described in Scenario 1.

The authorization service at one service provider must be able to decide whether an access request originating from another service provider is to be granted or not. And once obligations are introduced, a service provider that sends some data to another service provider must be able to communicate possible obligations that are involved. The MPOWER framework does provide mechanisms for managing a local policy repository – which only need to be enhanced to support a better policy language – that can be used to set up policies at each service provider. MPOWER does not provide mechanisms to communicate policies between services at different service providers.

### 6.1.3. Policy enforcement

In MPOWER the services must be trusted to verify a user’s permissions by contacting the Authorization Service. This is related to the very restricted “policy language” of MPOWER that limits permissions to service methods and in consequence does not support more fine grained access control. With a more expressive policy language, including obligations, more fine grained access control can be supported. This may allow other components than just the services to perform authorization checks, possibly adding multiple layers of checks. All this may increase the level of trust a data owner or data subject can have in a system.

There could however still be situations where a system cannot be trusted. Especially in situations where data is exchanged between different service providers. A user may trust a health care organization to have a secure system to store blood pressure data, but may not trust the system of a fitness program provider that wants to use that health data to provide a better fitness program. MPOWER does not provide mechanisms to protect data that is sent to a not trusted party from being used in ways that violate the policy.

### 6.2. Possible improvements

The limitations described in section 6.1 can be summarized as a lack of expressiveness in the policy language (i.e. only coarse grained permissions and no support for obligations) and challenges at the level of managing policies and securely enforcing them. This section discusses some conceptual solutions for each of these three areas.
6.2.1. Policy language

RBAC can be a suitable principle for designing a policy language for an AAL framework. As discussed in section 6.1.1 however, the way MPOWER has implemented RBAC is too restrictive. For data owners to stay in control of their data, it must be possible to specify the resource some permission applies to much more specifically.

A policy language for an AAL framework must support references to specific data objects, classes of objects (objects with specific properties) and other resources. This may very well include services as resources, to enable the kind of access control that MPOWER offers, but should not be restricted to services only.

Apart from supporting more fine grained resource references, it must also be possible to constrain permissions by defining a set of obligations that must be met upon granting or denying access. This can be achieved by extending the policy language with an obligation entity. Just like the resource entity, the obligation entity is related to the permission assignment. This results in the abstract model in Figure 11.

![Figure 11 Abstract RBAC model with obligations](image)

The policy language prescribed by the framework should provide a standardized way of referring to resources in policies and access requests, such that an Authorization Service can match these and take a decision. The same holds for obligations. Apart from adding an obligation entity to the policy language, all the parties must agree on a standardized set of obligations and support for performing the obligated actions must be included in all relevant components.

6.2.2. Policy management

To support the collaboration of services from different service providers as described in Scenario 1 two types of permissions must be configured. First of all a service at one service provider must be able to verify whether a service or user from another service provider can be granted access to certain data (access control). Secondly it must be possible to restrict how the receiving party can use the data (usage control).

Service collaboration is always configured on request of the user (assisted person). This configuration will from now on be referred to as “linking services”. Some application can be set up to provide an interface to the user to link services.

An important step in the linking process will be to establish access and usage control policies. This can be done through the existing Access Management services that allow for policy creation, revocation and policy updates at the involved service providers. With the introduction of a more expressive policy language, the policies will become more complex. This will require the development of new user interfaces for the Access Management service in order to provide user friendly tools for policy configuration.

Several approaches can be taken with respect to the location of the policies that govern data that may be exchanged between service providers. One approach would be to set up an access control policy at the provider that is the source of some data and a usage control policy at the provider where the data is consumed. A service at the source can then enforce the local access
control policy before exchanging data with a service at the consumer. The service at the consumer can then enforce the local usage control policy. The disadvantage of this approach is that a policy that governs data from a source at one party is stored at one or more other parties. That can make it more difficult for the owner of that data to make sure that all policies that govern that data are consistent and up to date. The more parties that are linked, the more challenging the maintenance of policies becomes.

Another approach would be to prepare both access control and usage control policies at the data source and then transfer the policies or references to policies along with the sensitive data when it is exchanged. The receiving party can then enforce them and possibly store them locally for later use. Sending the policy along with the data it governs, prevents the consuming party from enforcing a possibly outdated policy. Especially if not the entire policy is sent along, but only a reference to the policy that is stored at the data source. An update to the policy can then be performed without the need to transfer this updated policy to all relevant parties. This requires a lot less effort to keep policies consistent and up to date.

The second approach especially makes sense for usage control policies. For access control policies it makes more sense to decide upon them at the source, before sending the data. Hence, a hybrid solution could be chosen. Store and enforce access control policies at the data source and attach a reference to a usage control policy to data when it is exchanged which is enforced at the data consumer.

Configuring and storing policies at the data source does not require any specific new mechanisms, since mechanisms to manage policies are already required for local authorization decisions. Transferring and handling policy references does require some new mechanisms. Figure 12 and Figure 13 describe two ways to implement the transfer of policy references along with data. Figure 14 describes how these policy references are handled by policy repositories.

One possible way to implement the transfer of references to policies along with the data is by extending the response messages sent by services with an attribute that contains the policy reference (Figure 12). The DataSource receives a request and responds – after enforcing access control rules and processing the request – with the requested data and a reference to the relevant usage policy. The DataConsumer stores this reference at the local PolicyStore for later use in authorization decisions.

A variant to that implementation would be to introduce some kind of proxy component that intercepts the response message from the DataSource, adds the relevant policy information and sends it on to the other service provider (Figure 13). There a second proxy intercepts the message, strips of the policy reference, stores that at the local policy store and sends the original response message to the DataConsumer.
The second approach is a bit more complex, but requires fewer changes to the existing services and their interfaces. Another advantage is that the proxies form a dedicated component for retrieving, attaching and storing the policy references, instead of every service taking care of this itself. It is however questionable whether proxies have enough application specific knowledge to retrieve the relevant policies. For either solution, the messages that are transferred between service providers stay more or less the same.

Once a reference to the usage control policy is stored at the local policy repository it can be used for authorization decisions. The DataConsumer can check whether a certain use of the data is permitted. Before an authorization component can make a decision it retrieves the relevant policies from the repository and at that point the repository will retrieve the usage control policy for which it stores a reference (Figure 14).

### 6.2.3. Policy enforcement

Once policies are in place, either set up in advance or communicated along with the data they govern, they have to be enforced. The current approach in MPOWER is to let the various services check the permissions of their users. An advantage of such an approach is that the application specific knowledge about the kind of data and actions involved is readily available. A
disadvantage of such an approach is that services must be trusted to verify the permissions of their users and obey the policies.

A more expressive policy language automatically supports permission checks by other components of an AAL system – assuming that the necessary policies will be configured. This can be taken advantage of when designing framework components in a secure way. It would for instance be a good idea to include permission checks in any framework component that functions as a natural gate through which many operations on sensitive data are passed. A framework mechanism that provides persistent storage might be a good example of such a component.

This allows for multiple levels of security. First the general permission to call a certain service method is checked by the service. Then the service wants to retrieve or write some data as part of processing the request. The data storage mechanism that actually performs this operation can then check whether the user has the permission to perform that specific operation on some specific data. In this way the security of the whole system does not rely solely on permission checks in the services built on top of the framework, but may be also enforced by components that are part of the framework. The framework is designed once, with great attention to security, and can then be used by all kinds of parties to build their services on top of it.

Not all operations on data will pass through these securely designed parts of the framework however, and often services and other components built on top of the framework must be trusted to enforce certain policies. A service provider may trust its own system, but maybe not the system of a provider it wants to collaborate with and an assisted person may trust the system of a health service provider more than the system of a fitness program service provider. In such cases the service provider or user that is the source or owner of the data may require the service provider that consumes the data to use a special trusted component.

This is the main idea behind Digital Rights Management (DRM) techniques. A data source provides encrypted data and a policy (often called license) to a data consumer that is not trusted. Decryption can only be done using a special component (hardware and/or software) that will securely enforce the policy. An example of such a “DRM agent” would be a piece of software that allows a user to view some data, but not store, copy or redistribute it.

DRM is often used in entertainment systems to prevent unauthorized access to and redistribution of copyright protected material such as music and movies. It could also be used in an AAL setting to for instance enable a fitness coach to view someone’s health data (e.g. heart rate and blood pressure statistics) in order to fine tune a fitness program. By using DRM the subject of the health data does not have to trust the entire system of the fitness service provider to preserve the confidentiality of the health data, but only the small DRM agent.
7. Advanced authorization architecture

This chapter presents the advanced authorization architecture for AAL that is aimed at solving the limitations described in section 6.1 by implementing the conceptual solutions discussed in section 6.2. Section 7.1 describes the policy language and the components used for managing and evaluating policies. Section 7.2 describes how usage control policies are transferred between service providers. Section 7.3 shows how DRM techniques can be used to strengthen policy enforcement.

7.1. Policy language

This section describes how the eXtensible Access Control Mark-up Language (XACML) [15] can be used as the policy language in an AAL framework. It first describes the structure of the XACML policy language in section 7.1.1. Secondly section 7.1.2 explains how this can be applied in the AAL context. Finally section 7.1.3 describes a set of framework components that together provide the authorization functionality in order to incorporate the XACML policy language.

7.1.1. Language structure

Section 6.2.1 describes an abstract structure for a policy language that follows the RBAC principle and supports obligations. Figure 11 shows the entities user, role, permission, resource and obligation and the relationships user assignment and permission assignment. The user assignment relation is out of the scope of the policy language, as well as the user and role entities. References to roles (and possibly to specific users) will appear in policies, but users, roles and their relationship are not defined in policies. These can be stored and managed separately by services like the MPOWER Role Management Service and User Management Service. Policies describe the permissions assigned to a role and the resources and obligations related to that assignment.

XACML provides all the basic constructs to support fine grained policies. The following overview presents the constructs and describes how they can be used to implement the entities and relations from the abstract model. The XACML standard document contains more detailed descriptions of these constructs [15].

**PolicySet**

A PolicySet is the top level component in XACML. It has a PolicySetID and a Target and contains a number of Policies or PolicySets which are to be combined according to the specified Policy Combining Algorithm. Policies and PolicySets can either be included completely, or by reference using the PolicyID or PolicySetID. A PolicySet may also contain a number of Obligations.

**Policy**

A Policy contains a number of Rules which are combined according to a specified Rule Combining Algorithm by the component that evaluates the Policy. A Policy has a PolicyID, a Target and possibly a number of Obligations.

**Rule**

A Rule has a RuleID, a Target, a Condition and an Effect.

**Target**

A Target specifies the Subjects, Resources, Actions and Environments a PolicySet, Policy or Rule applies to. The Target of a PolicySet or Policy may be computed from the Target element of the PolicySet, Policy or Rule elements it contains. The Target of a Rule can be omitted if it is the same as the Target of the Policy that contains the Rule, in which case the Rule inherits the Target of the Policy.
Condition
A Condition can contain Boolean expressions that further refine the applicability of the Rule. This can for instance be used to compare an attribute of a subject and an attribute of a resource.

Effect
The value of the Effect element can be either Permit or Deny and defines whether an authorization request that matches the Target and Condition of the Rule will be granted or denied.

Subjects
The Subjects element of a Target contains a number of Subject elements that specify the “targeted” subjects. Upon evaluation of a Policy or PolicySet the Subject elements in the Target are matched against the Subject elements in the Request.

Actions
The Actions element of a Target contains a number of Action elements that specify the “targeted” actions. This corresponds to the permissions in Figure 11. Upon evaluation of a Policy or PolicySet the Action elements in the Target are matched against the Action element in the Request.

Resources
The Resources element of a Target contains a number of Resource elements that specify the “targeted” resources. Upon evaluation of a Policy or PolicySet the Resource elements in the Target are matched against the Resource element in the Request.

Environments
The Environments element of a Target contains a number of Environment elements that specify some additional attributes about the “targeted” environments. Upon evaluation of a Policy or PolicySet the Environment elements in the Target are matched against the Environment element in the Request.

Obligations
The Obligations element of a Policy or PolicySet specifies a number of obligations. Some of these will be forwarded to the component that sent the authorization request after evaluating the Policy or PolicySet. The Obligations element specifies whether an obligation should be fulfilled upon a Permit or upon a Deny Effect of the evaluation.

Combining Algorithm
PolicySets have a Policy-Combining Algorithm that specifies how the results of evaluating the Policies in the PolicySet should be combined. Policies have a Rule-Combining Algorithm that specifies how the results of evaluating the Rules in the Policy should be combined. Examples of such algorithms are “Deny-overrides” and “Permit-Overrides”. The first means that if one Policy or Rule evaluates to “Deny”, then the entire evaluation results in a “Deny” response, the latter means that if one Policy or Rule evaluates to “Permit”, then the entire evaluation results in a “Permit” response.

Request
A Request contains one or more Subject elements, a Resource element, an Action element and an Environment element.

The Subject elements contain attributes about the subjects for which the authorization request is performed. Multiple Subject elements are supported to specify for instance both the human user and the application component from which request originates.
The Resource element contains a number of attributes that define the resource that is involved in the request. It may also contain (part of) the content of the resource, in a ResourceContent element.

The Action element contains a number of attributes that define for which action authorization is requested.

The Environment element contains a number of attributes that describe the environment that is associated with the request.

Response
A Response contains a Decision element an optional Status element and an optional Obligations element.

The Decision element contains the outcome of the evaluation of the Request and can take on one of the following values: “Permit”, “Deny”, “Indeterminate” (e.g. when evaluation failed due to some error or missing information) or “NotApplicable” (when no policy is found that applies to the Request).

The Status element is used for error reporting.

The Obligations element contains a number of obligations that must be fulfilled by the component that receives the response. If that component cannot fulfil them, it must act as if the result of the authorization request was “Deny”.

The XACML RBAC Profile describes a method for structuring policies in a specific way, in order to implement RBAC including role hierarchies [16]. It defines the following four types of policies.

Role PolicySet
This type of policy set only contains a Subject element in its Target that specifies the role to which it applies and a reference to a Permission PolicySet.

Permission PolicySet
For each role, one Permission PolicySet must exist that contains the actual permissions of that role in a number of Policy and Rule elements. A Permission PolicySet may contain references to Permission PolicySets that belong to another role in order to model role hierarchy (e.g. administrator role has a number of special permissions and also all permissions of the employee role). A Permission PolicySet and its contained elements may not restrict the Subject, since that could break a hierarchy construction.

Role Assignment Policy or PolicySet
Optionally, policies can be constructed that govern the assignment and enabling of roles to subjects. This relates to the notion of constraints as defined in RBAC2 [14]. The subject of such policies can be any subject, the resources are roles and the actions are for instance “assign” or “enable”.

HasPrivilegesOfRole Policy
Optionally each Permission PolicySet can be extended with a HasPrivilegesOfRole Policy. Such a policy specifies whether the subject has the privileges of a certain role. This can be used to check whether a subject with role X also has permissions of role Y (e.g. through inheritance based on the role hierarchy).

Since a Permission PolicySet does not restrict its subject, it should never be used in an evaluation, unless it is retrieved through a reference in a Role PolicySet that matches the Subject in the Request.

The reason to keep the Permission PolicySet (without a subject) separate from the Role PolicySet (with only a subject) is that this construction allows for the support of role hierarchies. If a policy with subject X would reference a policy with a different subject Y to achieve the inheritance of the permissions in the second policy, this would not work, because a Request would match only one of the policies, based on the subject.
7.1.2. XACML in AAL

To apply XACML in an AAL framework, the framework must contain components for managing, evaluating and enforcing policies. Section 7.1.3 describes these components. This section describes how the concepts from the XACML policy language relate to concepts from the AAL domain. Appendix A.3 contains some examples that further illustrate the relation between XACML and AAL concepts.

XACML policies – represented by PolicySets consisting of Policies that in turn contain Rules - can be used as a tool for service providers and individual users to describe access control rules that govern their sensitive assets. Policies can be used as a mechanism to capture the user’s consent with certain operations on the assets of the user. The following paragraphs describe how the XACML directives that make up the core of the policies relate to AAL concepts. Some of these directives are also used in XACML Requests, where they relate to AAL concepts in a similar way.

Policies use the Subject directive to refer to AAL system end-users and possibly also to system components by specifying attributes of the targeted “subject”. A policy can target one or more specific users by specifying their ID, but may also target a group of users through a role, or organization identifier. In case of applying the XACML RBAC profile, only the role attribute will be used.

The Action directive is used to refer to the operations that the policy allows or permits. In an AAL setting these operations may be service methods, to achieve similar access control rules as MPOWER, but other operations may be specified as well. Typical examples of operations that could occur in policies and requests are read and write, or create, retrieve, update and destroy.

The Resource directive is used to specify the asset governed by a policy. Again, to match the MPOWER authorization architecture, this could include services as assets. Other examples of possible assets are data objects, sensors and actuators. Assets are not only referred to by their identifiers, but may also be targeted by specifying certain other attributes of the asset and can even take the content of a resource into account. This content can be provided in the Request.

The Environment directive can be used to make policies context aware and to provide context information in a Request. This can for instance be used to define a policy that governs the detailed location data of an assisted person and only allows access to this data in case of emergency. Whether an emergency situation is present must then be specified in the Request.

The Obligation directive is used to require certain actions to be performed upon enforcing a policy. Triggering a service to write to an audit log or send a notification to some stakeholder are possible applications of this directive.

The Condition directive is used to further restrict the applicability of a Rule. This can be used to model some important concepts in AAL such as granting the owner of some data certain permissions (i.e. owner ID in the content of a resource matches the user ID in the Subject part of the Request). A second example is granting a permission to a user that has a certain relationship to the owner or subject of some data (e.g. assisted person – caregiver or assisted person – relative). This relationship must be documented somehow. This can be documented inside the content, like with the content owner example, but can also be specified separately in the Request, as an attribute of the Subject, Resource or Environment.

In theory a policy can match any subject, action, resource and environment; as long as it can be identified by some well defined properties. Some of these properties, or attributes, could be defined by an AAL framework, but some of them will be application specific. It is important that components that are used to configure policies and components that handle the resources and request authorization checks refer to attributes in a compatible way.

If the strict RBAC approach of the XACML RBAC profile would be followed, the notion of subject would be restricted to roles. This has the advantage that policies follow a nice structure and can support role hierarchies. The disadvantage is that such a restriction decreases the expressiveness of the policy language. When a policy can only target a subject by its role, it cannot target specific users, or target users that have some relationship to the target resource.
Both approaches can be combined in a hybrid solution. A collection of strict RBAC policies is then used for high level access control to service methods, while a collection of more expressive policies is set up for controlling access to specific data objects.

### 7.1.3. Components and data flow

When moving from a simple RBAC implementation like that of MPOWER to an implementation using XACML, the main structure of the components and their function can stay the way it is. Following the MPOWER setting, this means that there is a User Management Service, a Role Management Service, an Access Management Service and an Authorization Service. These services rely on an Authentication Service to verify the identity of users and provide them with security tokens.

The User Management Service handles the user information, including authentication credentials and assigned roles. The Role Management Service handles the role information. Both services provide some basic create, retrieve, update and delete functionality. Since user and role information is not affected by introducing XACML, these services do not need to be adapted.

The Access Management Service of MPOWER manages the assignment of permissions to roles. It provided methods to add a permission to a role, to retrieve the permissions of a certain role or user and to revoke a permission. Recall that permissions consist of service methods that are allowed to be called. In our advanced authorization architecture it takes on the role of the Policy Administration Point (PAP) from the XACML reference data-flow model, managing the XACML authorization policies [15]. It provides mechanisms to add, update, retrieve and revoke policies. Therefore its high level functionality is comparable to that of the MPOWER Access Management Service. The proposal replaces permissions by policies, which of course requires changes to the inner workings of the service and the data types used in its interface. Since policies are more complex than the MPOWER permissions, an update method is added as well. Figure 15 and the method descriptions below describe the new interface of the Access Management Service.

![Figure 15 Access Management Service interface](image)

- `getPolicySetByTarget` can be used to retrieve a policy set for a specific target (i.e. specified by subject, action, resource and environment attributes). In a strict RBAC setting the target parameter would only specify the role attribute of the subject.
- `getPolicySetByID` can be used to retrieve a specific policy, given its identifier. More about this in section 7.2.
- `addPolicySet` can be used to add a new policy. In a strict RBAC setting this would only be done upon setting up a new role in the system.
- `updatePolicySet` can be used to update an existing policy. For instance to add or remove a certain rule or change its target. The Access Management Service finds the policy set with the same identifier as the set passed as a parameter and replaces it with the policy set from the input.
- `revokePolicySet` can be used to remove a certain policy from the repository.

The Authorization Service of MPOWER handled authorization requests. It provided a method to perform an access control check for a specified service method and user and a method which returned all services a specified user was allowed to access. In our advanced authorization architecture it takes on the role of the Policy Decision Point (PDP) from the XACML reference
data-flow model [15]. It still provides a method to perform an authorization check, but instead of a service method and a security token it takes an XACML Request and a security token as input. Figure 16 and the method description below it describe the new interface of the Authorization Service.

```
interface AuthorizationService

isAuthorized(request : XACMLRequest, token : SecurityToken) : XACMLResponse
```

*isAuthorized* can be used to perform an authorization check. Given the information in the request parameter it retrieves all relevant policies from the Access Management Service, combines these and takes a decision based upon the rules in the policies. It returns a XACML Response. The security token can be used to check whether the request originates from an authenticated user.

Figure 17 describes the basic message flow for authorizing a service request. It assumes some policies have been set up in advance (e.g. by the service provider or by the user upon registration at a service or service provider) and the user is logged in and has received a SecurityToken from the Authentication Service. The steps in the process are described below. Terms in italic refer to the language constructs described in section 7.1.1.

- **Service1** receives a request from a user through an application. The service request consists of a request body and a SecurityToken that contains information about the requesting user including a userID and the active roles of the user (see Figure 18 for its structure).
- **Service1** then formulates an XACML Request. Subject information such as the userID and the roles of the user can be derived from the SecurityToken. The Request is sent to the Authorization Service, along with the SecurityToken.
- The Authorization Service sends a request to the Access Management Service to retrieve the policies that are relevant to the roles of the subject. The Access Management
Service can easily provide these because of the policy structure defined by the XACML RBAC Profile [16]. After constructing a XACML PolicySet, this is returned to the Authorization Service.

- The Authorization Service then evaluates the Request against the PolicySet and returns a XACML Response to Service1.
- Service1 needs to obey the Decision contained in the Response and fulfil any Obligations defined in the Response. It will either continue handling the request, or respond that authorization for the request has been denied. If it cannot fulfil the specified Obligations, it must also halt and respond that the request has not been authorized.

At the start of a service method, it might not be clear which actions will be performed on which resources, except for the service method as the action and the service as the resource. A service will always check first whether the user has the permission to call the service method. If specific data is accessed as part of processing the request, then an authorization check for that specific access attempt will be performed.

### 7.2. Policy transfer

When a user wants two services from different service providers to cooperate the user must reconfigure the access control policies at these service providers to allow access to data at one service provider by a user of a service at the other service provider. This does not require any changes to the setup presented in section 7.1.3, except for the requirement that services (including the Authorization Service) must be able to handle subjects (and their security tokens) of another service provider. This can be achieved by adding an identifier of the provider to the security tokens as they are used in MPOWER (Figure 18), which allows the user’s roles to be distinguished from local roles. Another approach could be to use globally unique roles.

Users and service providers may want to restrict what a user or service from a remote provider is allowed to do with their data after it has been transferred. To do this, a PolicySet can be set up that contains a usage control policy. When the data is transferred to the other provider, a reference to this PolicySet can be attached. An obligation to forward a reference to this PolicySet can be included in the original access control policy.

The reference is forwarded by including a XACML PolicySet with the service response message containing a Target and a PolicySetIDReference directive in the form of a URL. The Target could match the requesting user or his role, but could also match the data that was transferred. The receiving service stores this PolicySet at the local Access Management Service for later use. The local Access Management Service can use the PolicySetIDReference to retrieve the usage policy at the remote Access Management Service when it receives a request for policies matching the specified Target.
To adapt existing MPOWER services to include a PolicySet in the response message of certain service methods, the interface of that service must be changed. In MPOWER the service interface is defined in a WSDL file. Consequently the WSDL file of every MPOWER service that has to incorporate the policy transfer mechanism must be changed. This could be relatively easy if all response message types would inherit the properties of some generic response message type. In that case the generic type could have been expanded with a policy element.

Unfortunately not all MPOWER service interfaces have such a structure. For the MPOWER services that use the HL7 namespace the type hierarchy of HL7 might provide opportunities to insert a policy element in some generic type. For MPOWER services that do not use HL7 there is no hierarchy in response message types. Fortunately many response messages do include a status element, which is used for error reporting. This status element could be expanded with a policy element to update the interface of many service methods, without having to change each and every one of them separately.

Figure 19 describes the basic message flow for a request from a service at one service provider to a service at another service provider. The original user request that preceded this interaction and the initial authorization of that request are not shown here.

- The service request from Service 1 (S1) to Service 2 (S2) is done as usual and contains a body and the SecurityToken that was originally presented to S1.
- S2 formulates an XACML Request specifying the requested resource and relevant subject information and forwards it to the Authorization Service at Service Provider 2 (SP2) together with the SecurityToken. The subject information is derived from the SecurityToken and includes the identifier of Service Provider 1.

Figure 19 Policy transfer for inter-provider requests

---

25 Web Services Description Language: http://www.w3.org/TR/wsdl
• The Authorization Service at SP2 (AS2) extracts the relevant target information from the XACML Request and requests the relevant policies at the Access Management Service of SP2 (AMS2). This sequence diagram does not include SecurityToken verification. S2 could verify the validity of the SecurityToken, by contacting the Token Management Service of Service Provider 1.

• AMS2 constructs the PolicySet relevant to the specified Target. Since the request comes from another service provider, the PolicySet includes an obligation for Service2 to forward a reference to a PolicySet with usage control rules. AMS2 returns the PolicySet to the AS2.

• AS2 then evaluates the Request and returns a Response to S2, including the obligation to forward a reference to the PolicySet containing the usage control policy.

• S2 fulfills any local obligations and handles the request. As defined in one of the obligations the response message does not only include the requested data, but also a PolicySet which in turn includes a reference to a PolicySet stored at AMS2 that defines the usage permissions for the data that is exchanged.

• S1 receives the data and the PolicySet and stores the PolicySet at the local Access Management Service (AMS1).

• S1 continues with processing the data and may at some point need to authorize a certain action it wants to perform. At that point it contacts the local Authorization Service (AS1) as usual.

• AS1 sends a request for policies to AMS1 as usual.

• AMS1 constructs the set of relevant policies. This includes retrieving the PolicySet stored at AMS2, through the reference that was transferred and stored earlier. After getting the PolicySet from AMS2 a PolicySet is returned to AS1 as usual.

• AS1 takes a decision and returns a Response to S1.

Since XACML only standardizes the main structure of policy elements, but not the actual attributes used inside the various elements, policies from one service provider may contain attributes that are not supported by another service provider. Therefore the approach of transferring policies introduces a compatibility challenge. Upon retrieving a “remote policy” the local Access Management Service will have to make sure that the policy can be evaluated by the local Authorization Service. A number of possible solutions can be identified.

An AAL framework could include a specification of policy attributes that may be used and must be supported by all parties implementing the framework. Since an AAL framework is often very generic in nature, to support all kinds of AAL service providers, this may however not be achievable.

Service providers that want to support the linking of some of their services could agree upon a standardized set of attributes to be used in transferrable policies. This has the advantage that it does not require a global standardization of attributes.

A translation mechanism could be developed to enable an Access Management Service to translate a “remote policy” to its local policy dialect. Semantic techniques like ontologies, that are already investigated to provide compatibility between web services, might be of use here.

### 7.3. Policy enforcement

Sections 7.1 and 7.2 describe a setup where the various components that perform operations on data are responsible for enforcing access and usage control policies. Services and other components of the system must contact a policy decision point to verify whether the component and its user have permission to perform a certain operation. The owner or subject of the affected data must trust that these components are implemented in such a way they only perform operations that comply with the policy.
Section 6.2.3 mentions that DRM could be used in cases where a data owner or data subject wants to grant some party access to the data without the need to trust the system of the data consumer. The following sections discuss how an existing DRM standard - the DRM specification of the Open Mobile Alliance (OMA) - can be incorporated into an AAL framework. Section 7.3.1 briefly describes the main concepts of the OMA DRM standard and section 7.3.2 discusses the implications of applying it to an AAL context.

7.3.1. OMA DRM specification

The OMA DRM specification consists of the following main components.

**DRM Content Format**

The DRM Content Format (DCF) describes the file format that is used to transfer some data in a protected way [17]. It follows the ISO base media file format standard [18]. A DCF file mainly consists of some headers, a Content Object and optionally a Rights Object (RO). The headers provide information about the encryption method that is used, information about the length of some other sections of the file, a ContentID and a reference to a Rights Issuer. The Content Object contains the encrypted data. A Rights Object defines a set of permissions applicable to the content and holds protected key material to decrypt the content. It is usually retrieved from a Rights Issuer, but optionally it can be included in the DCF file.

**DRM Specification**

The DRM Specification mainly describes a number of protocols that can be used to exchange DCF Files and RO’s and the cryptographic mechanisms used to protect content and key material [19].

**DRM Rights Expression Language**

The rights expression language defines the structure of the rights in the RO [20]. Therefore it plays a role similar to that of an access control policy language as described in section 7.1.1.

![Figure 20 OMA DRM DCF and RO exchange](http://www.openmobilealliance.org)
Figure 20 illustrates the basic message flow for exchanging some DRM protected data and retrieving the RO that holds the permissions and decryption key.

- A user requests some data from a service, for instance through a browser at the client side and a web application at the service provider.
- The service will perform some access control checks (not shown here). If the user is allowed access to the requested data, the service contacts the Rights Issuer (RI) to set up a Rights Object specific for the requested data and the receiving client. The RO setup ends with the RI presenting a Rights Object Acquisition Protocol (ROAP) Trigger. This trigger can be used by the client to retrieve the RO.
- The service then sends a DCF file, which contains the encrypted requested data, and the ROAP Trigger to the client.
- At the client the DRM Agent will take over from there. Using information in the DCF file and the ROAP Trigger it retrieves the RO from the RI. The RO contains the key to decrypt the content in the DCF file. This key itself is also encrypted and can only be decrypted by this specific DRM Agent. After decrypting the content, it is made available to the user, but only to the extent that is allowed by the rights specified in the RO.

### 7.3.2. OMA DRM in AAL

OMA DRM was not originally designed for a setting like AAL, but especially for protecting non-free content sent to mobile devices (such as ringtones). Consequently it does not fit perfectly to the world of AAL services and applications exchanging all kinds of messages. The DCF file format is a good example of this. Such a binary file format is not ideal for incorporating into XML based SOAP messages. The OMA DRM Rights Expression Language (REL), which is based on ODRL\(^27\), is also not directly compatible with the XACML policy language that section 7.1 proposes for AAL.

This does however not mean that one has to reinvent the wheel. Important parts of the OMA DRM specification, such as the protocols and cryptographic mechanisms, can be incorporated into an AAL framework. Other parts, such as the DCF format and the REL, could be replaced or adapted to better fit the AAL setting.

The DCF format can be replaced by an XML message format. The WS-Security\(^28\) extension to the SOAP standard provides a good example of how to include encrypted data and some necessary meta-data in an XML message.

The REL could be completely replaced by XACML, or a mechanism could be developed to derive RO’s written in the REL from XACML policies. Since the REL is rather limited and strongly focussed on the domain for which OMA is designed it may not be capable of expressing the permissions that are relevant in the AAL domain. Therefore an AAL framework should not use such a restrictive language and replacing it with the XACML language described in 7.1 would be a better approach than implementing a translation mechanism.

By replacing or adapting the data structures but keeping the protocols for exchanging content and retrieving rights (or policies) as sketched in Figure 20 and the cryptographic methods for protecting the content and the content decryption keys, the security properties of the OMA DRM specification can be maintained.

---


8. Analysis

This section presents a security analysis of the advanced authorization architecture for AAL that is proposed in chapter 7. The goal of the analysis is to identify how the architecture improves over the authorization architecture of MPOWER and whether it has limitations that need to be addressed in future work.

The analysis is split in two parts. Section 8.1 describes a set of risks for e-health applications identified in an ENISA project [21] and analyses which controls each architecture offers to mitigate these risks. Section 8.2 describes a number of authorization mechanisms that must be present in e-health systems according to Accenture’s Institute for Health and Public Service Value [22] and analyses to what extend these mechanisms are present in the two architectures. Section 8.3 summarizes the findings of the two analyses.

Both “input documents” originate from the e-health domain. This domain is slightly different from AAL, but since health information is one of the most sensitive assets in AAL systems the risks and component requirements for e-health systems are also relevant to AAL systems.

The analysis focuses on risks and controls related to authorization, since the advanced authorization architecture for AAL that is proposed in chapter 7 only includes concrete improvements with respect to authorization mechanisms.

8.1. ENISA EFR Pilot risk analysis

ENISA identifies a large number of risks related to e-health systems. The following paragraphs briefly describe each risk and discuss how each architecture mitigates it. Table 2 provides a summary of these findings. The analysed risks are only the ones that are relevant to authorization, the numbers refer to the risks as they are presented in Annex II of the ENISA document [23]. That document contains more elaborate descriptions of the risks.

R02 The risk of non-compliance with informed consent legislation
Access to sensitive data without prior consent from the subject of the data may be prohibited by legal regulations. A system that is not protected against such illegal data access could cause privacy breaches that could have legal consequences. An AAL framework could provide mechanisms to capture and enforce the consent of its users to mitigate this risk.

The advanced authorization architecture for AAL supports the capturing and enforcement of consent by providing an access control policy language that is expressive enough to write policies that describe in detail to which operations on what data a user has consented. By providing policy transfer mechanisms, the consent can also be managed in a highly distributed setting. Two limitations are that, although it defines the policy decision and policy administration components, it does not provide a user friendly interface to configure policies and it may depend on components built on top of the framework to enforce the policies. The analysis of R04 discusses this second limitation in more detail.

MPOWER lacks such a policy language that is expressive enough to capture consent and does not provide any mechanisms to explicitly capture and enforce consent.

R04 The risk of a confidentiality or integrity breach of processed data
Systems that do not provide adequate protection of data in storage or communication risk that the confidentiality or integrity of this data is breached. Access control is one of the important security mechanisms to mitigate this risk.

The advanced authorization architecture for AAL supports fine grained access control on multiple levels throughout the system. Not only does it support controlled access to service interfaces, but also to specific assets such as stored data, sensors and actuators. A limitation of the approach of the advanced authorization architecture is that it does not force the authorization of attempts to access and use assets. It provides the mechanisms to store policies and evaluate them, but relies on the components that process data and provide
access to assets to request authorization checks and enforce the outcome. Consequently
the security of the AAL system depends upon framework components and services and
applications built on top of the framework.

MPOWER only supports coarse grained access control. It provides a mechanism to restrict
access to service methods, but no means to restrict access to specific assets. Just like the
advanced authorization architecture, it relies on services to make use of the service method
access control mechanism.

R13 The risk of unauthorized access to data
The risk that unauthorized individuals gain access to sensitive assets and are able to modify
or delete them is related to risk R04 in the sense that such access would be a confidentiality
or integrity breach. Therefore the analysis of how the architectures mitigate risk R04 also
applies to risk R13.

R15 The risk of data surveillance and profiling
With large amounts of sensitive data being collected and processed, stakeholders may be
tempted to combine all kinds of information about an individual in order to gain extra
knowledge. This could impose a threat to the privacy of the individual. This is an example of
using data outside of the purposes for which it was originally intended (risk R16) and as
such it is mitigated in similar ways as risk R16.

R16 The risk of data being used outside the purpose originally intended
This risk is closely related to risk R02, in the sense that good consent management and
enforcement prevents data from being used outside the original purpose defined in consent
statements. This risk is especially relevant to AAL because of the heterogeneous landscape
of service providers that exchange all kinds of sensitive data.

The advanced authorization architecture for AAL mitigates this risk not only by supporting
fine grained access control policies that could be used to capture and enforce the intended
purpose and consent, but also by providing mechanisms to transfer such policies along with
the data they govern. It also shows how DRM techniques can be applied to strengthen the
security of policy transfer and enforcement.

Since data is supposed to be accessed to some level, it cannot be completely guaranteed
that someone who has access to an asset will not use the information inside it
inappropriately. Audit logs could serve as an extra safeguard against inappropriate use, by
recording who has accessed a certain asset. The advanced authorization architecture for
AAL supports the definition of obligations inside policies. These obligations can be used to
define when audit logs should be written.

MPOWER does not support policies that can capture intended purpose. MPOWER does
provide an audit log service, but no mechanisms to trigger its use.

R20 The risk of non-compliance with data protection legislation
If a system does not comply with data protection legislation then security breaches may not
only impact confidentiality and integrity of data and privacy of individuals, but may also have
legal consequences.

Both MPOWER and the advanced authorization architecture for AAL have taken data
protection legislation into account while investigating the security requirements [12].
### Table 2 Summary of risk analysis

<table>
<thead>
<tr>
<th>Risk</th>
<th>MPOWER</th>
<th>Advanced authorization for AAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>R02</td>
<td>No mechanism to capture consent</td>
<td>Consent can be captured in XACML policies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>User friendly interface to be developed</td>
</tr>
<tr>
<td>R04/R13</td>
<td>Only access control to service methods</td>
<td>Support for fine grained, multi-level, access control</td>
</tr>
<tr>
<td></td>
<td>Relies on services to request authorization checks</td>
<td>Relies on other components to request authorization checks</td>
</tr>
<tr>
<td>R15/R16</td>
<td>Only access control to service methods</td>
<td>Support for fine grained, multi-level, access control</td>
</tr>
<tr>
<td></td>
<td>No policy transfer or DRM mechanisms</td>
<td>Policy transfer mechanism and DRM support</td>
</tr>
<tr>
<td></td>
<td>Audit mechanism but no trigger mechanism</td>
<td>Logging/notification triggered by XACML obligations</td>
</tr>
<tr>
<td>R20</td>
<td>Legislation taken into account for requirements</td>
<td>Legislation taken into account for requirements</td>
</tr>
</tbody>
</table>

#### 8.2. Accenture information governance foundation for e-health

Accenture describes a number of components that must be included in e-health system architectures in order to ensure data privacy, data confidentiality, data security, data quality and data integrity [22]. The following paragraphs briefly describe each component and discuss whether it is included in MPOWER and in the advanced authorization architecture for AAL. Table 3 provides a summary of these findings. The analysed components are the ones that are relevant to authorization. More detailed descriptions can be found in the appendix of Accenture’s document. The numbering of components is not taken from Accenture’s document; it’s only for local reference.

**C1 Patient consent models and mechanisms**

According to Accenture “patient consent should be the prime access control in e-health systems”. This topic is already covered by the analysis risk R02 in section 8.1, which showed that MPOWER does not provide any consent mechanisms, whereas the advanced authorization architecture for AAL provides a policy language that can be used to capture consent.

**C2 Patient-provider relationship based access controls**

In healthcare a patient usually has one or more healthcare professionals responsible for his care. The responsible professionals should have access to the relevant data, while their colleagues that are not involved in the care for that patient should not. This principle also applies to the AAL domain, because e-health may be part of the offered AAL services and also because relationships between assisted persons and their relatives and friends could serve as a basis for access rules.

The advanced authorization architecture for AAL provides a policy language that is capable of capturing such relationships in access control rules (except when the XACML RBAC profile is used, as discussed in section 7.1.2).

MPOWER does not provide a mechanism to incorporate relationships between users into access control rules.

**C3 Patient access controls**

Accenture states that e-health systems should provide patients with access to data about them. Appropriate authentication and authorization mechanisms should be incorporated to
support such an interface. It will be rather important that patients can only view data about themselves, not about other patients.

MPOWER does not provide access control to specific data, only to service methods. Consequently it cannot restrict patients to only have access to their own data.

The advanced authorization architecture for AAL does support policies that govern specific data objects. It can also support policies that capture the general notion that a patient only has access to his own data, without needing a policy for every patient or every data object.

C4 Role-based access control models

Accenture suggests role-based access control models as a suitable mechanism to represent job-functions and roles and grant permissions to such groups of users. They stress that very specific permissions, capable of defining rules on access to data, usage of data and access to functionality, must be supported.

The advanced authorization architecture for AAL supports assigning permissions to roles by specifying the role in the subject element of a policy target. The policy language supports highly detailed permissions and provides mechanisms to transfer usage rules along with data that is exchanged. It even supports strict hierarchical RBAC through the XACML RBAC profile, but this reduces the expressiveness of the policy language and may conflict with the level of detail needed in policies.

MPOWER does provide a role-based access control mechanism, but its range of supported permissions is far too restrictive. It only supports controlling access to functionality on the level of service methods. No permissions on specific data objects are supported. It also does not support role hierarchies.

C5 Patient and provider record sealing

Accenture states that patients, health care professionals and administrators should be able to seal certain parts of a health record, thereby restricting access to these parts beyond the restrictions that are effective on the entire record. In contrast to e-health systems AAL systems are not built around a notion of a health record as some sort of structured document. The idea of restricting access to parts of data objects may however still be applicable. Especially since e-health systems with their health records could be integrated with AAL systems.

The advanced authorization architecture for AAL supports policies that define permissions on specific parts of a data object. It does not provide straightforward mechanisms to hide sealed parts of data. This could however be implemented in a data storage component or in the services that provide users with an interface to access the data.

MPOWER does not support permissions on specific data objects, let alone permissions on parts of these objects.

C6 Event audit and alerting

Logging, monitoring and reporting all security relevant events is vital to an e-health system, according to Accenture. This is a requirement that is also found in data protection legislation [12]. It is not a direct requirement for authorization mechanisms, but is strongly related to it. Audit logs and even notification add an extra layer of security. Organizations and users can keep track of what happens to their data and who has accessed it and possibly act upon this information in case something inappropriate took place.

The advanced authorization architecture for AAL does not define audit and notification mechanisms. These must be defined separately. It does define however the notion of obligations as part of policies. These obligations can be used to trigger logging and alerting. Since users can configure the policies, they can have some control over which events must be logged and for which events they want to receive notifications.

MPOWER does offer an audit service, but no means to trigger its use through policies.
AAL Security and Privacy  H.G.M. Vlamings

8.3. Analysis summary

The analysis shows that the advanced authorization architecture for AAL as it is presented in chapter 7 provides some substantial improvements over the authorization mechanism provided by MPOWER. The much more expressive policy language can capture important concepts such as informed consent and inter-user relationships and can target specific data objects. Usage control policies can be sent along with data and can optionally be securely enforced using DRM. It also added the concept of obligations. A concept that is not explicitly mentioned in ENISA’s risk analysis, or Accenture’s analysis of required components, but can be used to gain more control over functionality such as logging and notification that is deemed important by these two analyses.

This reduces risks of inappropriate and unauthorized access to and use of sensitive assets and also provides better compliance with data protection legislation.

The advanced authorization architecture for AAL does however not completely remove these risks. It still relies on other framework components and services and applications built on top of the framework to perform the necessary authorization checks. And some user friendly interfaces will have to be developed for the very expressive policy language to be really effective.
9. Conclusions

Ambient Assisted Living systems deal with great amounts of sensitive data gathered from sensors and user input. Strong security controls must be put in place to protect the privacy of the users and to prevent the loss of data integrity or availability problems that could impact the user's health. The highly distributed setting, with multiple service providers exchanging data and users sharing information with their friends and relatives introduces some extra challenges for securing an AAL system.

An important requirement that must be satisfied is that users must stay in control of their data, even if it is transferred from one system to another. Access should not be granted unless the user has been informed and has provided consent and users must be able to keep track of access attempts and be notified of relevant security events.

Consequently a security architecture for an AAL framework that covers these requirements must not just provide identity management, basic access control to data sources and system interfaces and mechanisms for securing communication channels. It must also provide a more advanced authorization mechanism that supports access and usage control policies for specific data objects and mechanisms to transfer these policies along with the data they govern. If policies are transferred with the data they govern, a user does not have to configure these policies in every system that processes that data. This makes it easier for users to stay in control of their data. A user friendly interface to configure such policies must be in place to capture user consent and notification and login mechanisms are needed to allow users to keep track of their data.

Existing AAL frameworks often do not cover all of these requirements. An analysis of the authorization mechanisms in MPOWER shows some serious shortcomings. Not only does it lack a policy language expressive enough to support fine grained access and usage control on specific data objects, capture user consent and define obligations, the MPOWER framework also lacks explicit support for transferring policies between service providers and securely enforcing them.

These shortcomings can be solved by defining a more expressive policy language that does not restrict the notion of resources and permissions to the methods of services and that does provide a means to define obligations. Policies can be transferred between service providers by extending the message types of existing protocols, either by adapting service interfaces or by introducing proxy components that transparently add policy information on the sending side and strip it on the receiving side. Digital Rights Management (DRM) techniques could be used to securely enforce policies that are attached to data that is sent to parties that are not trusted to properly enforce the policies.

This report presents the advanced authorization architecture for AAL that implements these conceptual solutions. It incorporates existing standards like XACML and OMA DRM and extends and redefines some of the security services and protocols from MPOWER in order to support end to end access and usage control in an AAL context.

It defines how the eXtensible Access Control Language (XACML) can be used as a policy language in an AAL system by mapping XACML concepts to AAL concepts and redefining MPOWER's Access Management service and Authorization service to support this language. The policy language is expressive enough to target specific data objects and capture user consent and supports the definition of obligations.

The advanced authorization architecture for AAL also defines an extension of the MPOWER protocol to support the transfer of policies along with the data they govern. Mechanisms of the Open Mobile Alliance DRM standard can be incorporated to securely enforce transferred policies, replacing its content file format and rights expression language by SOAP messages and XACML while keeping its protocols and cryptographic mechanisms for securely transferring sensitive data (content) and usage policies (rights).

An analysis based upon risks and required components for e-Health systems identified in two other projects confirms that the advanced authorization architecture for AAL solves many of the
shortcomings of MPOWER. The analysis however also reveals some remaining or new limitations. The almost unlimited expressiveness of the policy language could problematically increase the complexity of policies and therefore make the processing, configuration and maintenance of policies a difficult task. This could result in policies and authorization decisions that are not consistent with the intentions of the person or organization that configured them, policies that are not understood by the systems they are transferred to or policies that conflict with each other. This increases the risk of inappropriate access to and usage of assets and the risk of unavailability of assets.

Future work will therefore include the development of user friendly policy configuration interfaces and possibly also a specific XACML profile for AAL. Such a profile could restrict the expressiveness enough to keep complexity under control and ensure compatibility between various systems built on top of a framework, while still retaining a language that is strong enough to express the necessary rules.

Another limitation that applies both to MPOWER and the advanced authorization architecture is that policy enforcement is left mostly to the components that process the data and provide access to it. The authorization mechanisms support policy management and decision making, but applications and services will have to request authorization checks. Consequently these applications and services must be designed such that they make use of the mechanisms offered by the authorization architecture and obey policy decisions in order to arrive at a secure system. The advanced authorization architecture improves this aspect by supporting authorization checks in central components such as a data abstraction layer, instead of just access control to service methods. It also shows how DRM can be applied to securely enforce policies if the receiving system is not trusted. This reduces the number of components that must be trusted to enforce policy, but policy enforcement will still rely on components other than those offered by the authorization architecture.

Apart from further improvements to the authorization mechanisms, future work will also have to fill in the details for the other mechanisms described in the high level security architecture, like for instance an identity management solution that complements the authorization architecture.
References

16. OASIS. *Core and hierarchical role based access control (RBAC) profile of XACML v2.0*. 2005.
Appendices

A.1 Original scenarios from MPOWER and OASIS

This appendix contains some original scenario descriptions from the MPOWER and OASIS projects that formed the main inspiration for the scenarios that chapter 3 presents. These are taken straight from project documents and presented here for quick reference and to provide some extra detail.

MPOWER Activity scenario 11: Hannah and Jim in the city

Scenario: a
Hannah and Jim are still together but the moving broke a lot of social, informal contacts they had before. For them as elderly it is quite difficult to built a new network. The building they live in is especially for elderly. Most of them just moved in since the building is brand new. For sure these elderly face the same problem.

In this building an intranet facility is installed. By easy login the residents can open the line and communicate with a view. They can make announcements what is being organized for them in the lounge of just let other people know they are sick. Help can be offered. For residents who cannot leave the home, they could use the intranet to have a chat with webcam and view on 4 others (conference) If there is an emergency situation they could contact each other to warn them.

In the building is a small supermarket. Residents can log in and fill out the shopping list. The shopkeeper is willing to bring the groceries to the home.

The intranet can simply be integrated within the intranet the internet and make not only Jim and Hannah but also other residents in the building more acquainted with the internet.

E-banking and E-shopping will be more common for them then also.

Scenario b: The shopkeeper in the building retires

The shop as a very important facility closes its doors. It is a pity for the residents because it had also a social meaning for them.

The management of the building made arrangements with some delivery service in the neighbourhood. The residents can now as they did before get their deliveries to be brought to the home.

The intranet can simply be integrated within the intranet the internet and make not only Jim and Hannah but also other residents in the building more acquainted with the internet.

E-banking and E-shopping will be more common for them then also.

Scenario c: Jim has to undergo surgery to have a new knee and go after that to the rehabilitation centre. He will be away for several weeks.

Hannah is very depending on Jim. They start to know some people now in the building but she cannot ask them yet for advice. Jim had his operation but have a new knee don’t mean he is sick. He misses his wife very much and she cannot visit him too often.

Jim has the possibility to log in on the intranet form the hospital so that he can contact his wife daily. This helps him very much to get over the temporarily separation and helps him to recover more easily.

Hannah can ask him for advice if needed.

MPOWER Activity scenario 15: Jadwiga lives in the residential house

Scenario a: Fall in course of a walk
During one of her solitary walks, Jadwiga slipped down and collapsed. She cannot stand up without assistance. Jadwiga has got GPS transmitter (she always takes it with her when going for a walk without company). A nurse who is on duty in the residential house may trace on the monitor Jadwiga location. If her location does not change for predefined period of time (for example 5 minutes), the nurse initiates the search for Jadwiga. Jadwiga is found and accompanied to the residential house by a person from the staff.

Scenario b: Fall in course of a walk
During a solitary walk in the forest area around the residential house, Jadwiga slipped down and collapsed. She cannot stand up without assistance. Jadwiga has got GPS transmitter (she always takes it with her when going for a walk without company). A nurse, who is on duty, keeps track of the route taken by Jadwiga...
on the monitor. Additionally, Jadwiga triggers the button on GPS transmitter, which sends the alert about accident. A nurse receives the alert and she initiates search for Jadwiga. Jadwiga is found and led to the residential house under the care of the staff.

**OASIS SCENARIO 1: Independent Living**

David is a widower. He is 60 years old, overweight and has high blood pressure and cholesterol levels.

Following his doctor’s instructions, he is considering to lose weight. The doctor has recommended a hypo caloric diet with low levels of salt. The doctor strongly advises him to take on healthy habits and practice some aerobic exercise.

When he arrives at home he connects to the OASIS collaborative group experiences using an Internet enabled TV, a tool which allows him to join different groups of people with similar interests and virtual communities. He obtains information about a service called “MyDiet”. MyDiet is an OASIS compliant service, that means that it works following an open architecture that allows the interconnectivity of services. He decides to try this service and after inserting his profile (age, gender, weight, height,...) he is subscribed. He will receive every day a personalized menu and basic instructions on how to cook his food; they also provide the list of ingredients that he will need for every meal along with a list of nearby shops he can purchase them from. As an additional service he can order the meal to the catering service of MyDiet. In this case he will receive his pre-defined menu or a similar one.

While connected to the Internet, David’s son Julian, while on a business trip, decides to ask his father about his visit to the doctor. David receives a notification on the TV for the incoming videocall. He explains the doctor’s recommendations and Julian comments that he is using a service named “virtual activity coach”. This service offers individual training plans for everybody, including elderly persons, and he suggests his father to subscribe.

In the process of subscribing the OASIS-compliant service, he is requested to allow search for compatible services. David answers “yes” and the “virtual coach service” discovers that David is subscribed to service MyDiet. The system asks David if he wants to combine both services. By combining services, they can share information to provide added value, for example, results from MyDiet questionnaires about food intake or compliance will be used by the Virtual Activity Coach to generate new exercise sessions and results from exercise sessions (activity level, energy expenditure, adherence to training…) will be used by MyDiet to adjust the menu accordingly. David accepts.

It’s Monday morning, the comfort unit switches off the lights of the garden and opens the blinds. David, while having breakfast, receives the day’s menu. After having a quick look, he stores the menu and the related information in his 4G mobile device. He connects the mobile with the fridge to check if he has the ingredients and is notified that he needs 200g of chicken. As it is a small purchase, he stores the note in his reminder tool, in order to buy the chicken during his daily walk. While he is thinking on these things, the kitchen switches off boiling the water for coffee. David lately tends to forget more doing some trivial things, but the OASIS system helps him prevent unpleasant situations.

While in the bathroom, the intelligent carpet transmits his weight to the nutritional advisor, the tool that is part of the Mydiet service that runs on his mobile phone. This tool provides automatic feedback to the service. In the same way, when he monitors blood pressure, results are stored and sent to the doctor’s patient follow-up system, using the health monitoring system.

During his walk, the activity coach service tracks the distance he has covered, giving motivation messages. He is also reminded to buy the chicken. Also, while walking, he is alerted that the temperature of the house is too high, especially when nobody is at home, he remembers that he forgot to switch off the heating so he acknowledges to the Comfort Control Unit to decrease the house temperature.

Back at home, he thinks about the things he tends to forget. “Perhaps it’s time to exercise my memory” - David thinks. Again, using the collaborative user experience tool, he searches
information about memory problems in senior citizens. He is informed about a memory reinforcement program, the brain trainer. This program describes how he can exercise his brain using daily activities. He decides to use this service and of course he links it to myDiet and the activity coach. From this time onwards, the memory reinforce program will use data from the other services as training material for the memory exercises, for example, prompting him how to remember the shopping list.

Devices / technologies / infrastructure addressed in scenario

- Information capture elements: Mobile network including PDAs, phones, Webpads,…, Security sensors (gas, water, fire), Vital signs monitoring, Activity monitoring: distance, movement tracking.
- Output elements: internet enabled TV, Mobile network including PDAs, phones, Webpads,…, Actuators networks (blind, lights, curtains, small household appliances, air renovation, …).
- Devices and products: Intelligent oven, Intelligent Fridge, Remote controlled air conditioning and heating.

OASIS SCENARIO 2: Autonomous Mobility

Since their children got married, John and Claudia have lived alone. John is now 75 and Claudia took early retirement at 63 to look after him, but works part-time from home as a graphic designer. While Claudia still drives, John stopped 5 years ago. He is in good physical health, but his eyesight is not good and he tends to forget things and get easily confused. He is supposed to attend routine check-ups at the hospital every two weeks. Claudia is fairly healthy, but has experienced some hearing problems in the last few years.

Tele-working and car travel support for the ‘young’ elderly

Claudia gets up. Her program for the day is to go shopping, then work a bit and cook lunch. She plans to start early, around 8.30 am drive to the local shopping mall, 5 kms away, then bring the things home. She gives this information to her OASIS system by voice, using her Tablet PC. The system connects to the Traffic Management Centre, makes a quick check on the current and forecast traffic levels.

She feels a vibration in her specially equipped belt and sees that there is a message for her from the TMC. It tells her that there is a lot of congestion at the moment and suggests she leaves at around 9.15 am when the traffic has calmed down. She confirms that she will indeed go later. The system automatically connects to the shopping mall car park, and informs her that a space has been reserved in a special area for 9.30.

On her way to the shopping mall, her in-car navigation system (which already knows her destination through connection with the OASIS pre-trip planning service in use of COF, CCM and agents) informs her that an accident has just happened on her route and suggests a deviation. This allows her to avoid the traffic jam that is building up. She reaches the shopping mall in time and is guided to her reserved parking spot.

Once back from shopping, she gets lunch and then settles down to work. She opens her mailbox on her specially configured Tablet PC which has become her ‘faithful companion’. An email message tells her that her business colleagues have put on the corporate portal the new designs for a project she’s working on. However, she finds that she is unable to access the portal. By giving the password and some keywords to the system, it is able to retrieve the document and download it on her desktop. It recognizes her as an authorized user thanks to the biometric authentication programmed into her PC.

She starts working on the new designs, but is phoned by her neighbour who asks if she can look after her grandson for an hour, as she has to go out urgently.

Claudia doesn’t really want to stop in the middle of a creative task, but leaves her Tablet PC and goes over to the neighbour’s home. The boy is watching the TV, so she has time for herself and switches on the PDA (always in her bag) and continues the work session from where she left it. She manages to complete her work successfully by the time his grandmother returns and goes back home to prepare supper for John and herself.
Travel support for the non-motorised elderly

While Claudia was out shopping in the morning, John received a reminder on his mobile phone of his hospital appointment at 3.00pm. He uses his PDA to find the bus times. It suggests suitable buses, showing the times, numbers and stops in large characters. He prints it out to keep with him on the trip. 15 minutes before it is due an alert is given (a series of beeps), based on the dynamically estimated arrival time. He goes to the bus stop, where a high definition panel indicates the bus numbers in large letters. Next to the number of his service there is a red button which he presses to obtain updates on his bus. When it arrives, it announces, “Bus number 10 is now at the stop”, so he is sure to get on the right one.

On board, a vocal message system announces the stops, so he knows where to get off. The stop where he has to change has a panel identical to the previous one. He presses the number and listens to the message so he can identify his second bus. Five minutes before he reaches his final destination, he uses a voice activated system on his mobile to call the hospital. This alerts a special support service to send someone to the stop to collect him and take him to his appointment.

He informs the same person that he wants to make a booking for the return trip with the DRT (demand-responsive service) which operates from the hospital. This is a minibus which plans its route dynamically depending on the destinations of the booked passengers.

It takes a less direct route, but it can deliver him to the door of his house. When on the bus he uses a programmed number on his mobile to phone Claudia and reassure her he is on the way back. Since this system has been in place, he has not missed any appointments as he is no longer afraid to use buses alone and Claudia too is less worried about him.

Devices/technologies/infrastructure addressed in scenario

Information capture elements: Mobile network, including interface to TMIC, DRT service and business portal, Biometric authentication, Trip planning and route monitoring, both for car drivers and public transport passengers.

Devices and products: Tablet PC, PDA, Mobile phone, Bus stop information panel, Vibrating belt.

OASIS SCENARIO 3: Combined Independent Living and Mobility with and without OASIS

Stella is 72, but still in very good shape, with only some slight memory and hearing deterioration problems. She has been living alone since her husband, Giannis, died. She always adored her husband and his loss 3 years ago really devastated her, but luckily, she managed to overcome this difficulty and learned to live alone in an active and meaningful way. Besides, she has always been a very active person with lots of friends; when she was younger, she would never stay at home for more than 2 weekends in a row... Especially after her daughter, Nikolina, got married and left the house to live in Thessaloniki, Stella would join the weekend-trips of the trekking club that she and her husband belonged to at least once a month. She is a nature-lover, so she has to leave Athens every now and then, to feel closer to nature.

Of course, ageing does not come without consequences... She cannot travel as often as back then, nor can she climb mountains any more. At least she goes on calm weekend trips to hotels close to nature and takes long walks at a fast pace, which is almost like the jogging she used to do when she was younger, choosing the most picturesque, green and calm routes possible. Of course, she has to be careful about the pace of her walking, and about what she eats. But she refuses to give up all of these nice habits completely: “as long as I am physically capable, I will live as I like”, she notes stubbornly every time Nikolina insists that Stella limits her travelling or long walks in the forest. Nikolina knows that her mother would never be happy if she were not able to lead an active life and she fully understands, so she got her this athletic garment that monitors physiological signals, such as ECG, respiration and activity index, including a shock/fall detector. The garment is light and nice looking, so Stella accepted Nikolina’s present with joy. All of the sensor-based measurements of Stella’s physiological data are processed through the PDA that Stella is carrying with her, by OASIS, the services-based assistant that helps Stella lead an
active life within her home and out of it. So this reassures Nikolina that, in case of an emergency, OASIS will immediately notify her, as well as Stella’s physician.

This weekend Stella has arranged through the OASIS tourism support services a trip to Edipsos, a very popular and picturesque village in Evia, right next to a forest and very close to a lake. OASIS suggested this not only because the hotel has specific accommodation for people with hearing impairments, but also because during this weekend the hotel will host the re-union of Veteran Greek mountain climbers and of course, Stella loves to meet other mountain-climbers, hang around with them and exchange funny stories about this or that time on that mountain... Sometimes, she even arranges walks with them. This is what she did last time in Pilio, so OASIS now knows her preference.

As soon as Stella arrives at the hotel on Saturday morning, she can’t wait to arrange her afternoon walk. She has taken of course with her the PDA that allows her to use the OASIS elderly-friendly route guidance and asks it to arrange a 2-hour walk. It is so easy to use this device, you simply speak to it, using simple voice commands, and you are presented with all the information, summarized in a short text, accompanied by an easy to navigate map, with all POIs marked with large letters. OASIS suggests to Stella a nice walk in the forest ending up at the lake. While planning her walk, she can see on her PDA all the refreshment points that exist between the start and end point of her long walk, as well as the closest medical centre. Moreover, she feels even more re-assured to know that, if she gets lost in the multiple paths of the forest, OASIS will guide her turn by turn back to the hotel, or to the ending point of the walk, depending on her mood. Stella had used this OASIS localization and pedestrian route guidance service in the past, when she got lost, and knows that is very reliable and precise.

Luckily, this time she did not need to use it. The path suggested by OASIS was wide and very well waymarked. While walking, Stella received a notification on her PDA from the OASIS home control service that her favourite series was on and unless it is recorded, she will miss it. Of course! She had forgotten to program the video recorder! Using her PDA she connects to the home network and arranges for the video recorder to start recording. Sometimes, she does not recognize herself. She used to hate technology, and find it so complex... But the OASIS interfaces are so simple and intuitive it has never been easier to use small devices, such as PDAs and smart phones. In fact, the size of the letters in her screen adjust automatically to her age-related sight deterioration.

She really loved the walk in the forest and then along the lake. So, she bookmarks this route, to re-visit it soon with her close friend, Anny, who is also a widow. She uses the mobile part of the OASIS social communities service to open a discussion session with Anny about her walking experience, share with her photos that she took along the way and show her on the OASIS detailed map the exact path.

After a really enjoying, but quite tiring weekend, Stella returns home. These trips are so much fun, but they usually exhaust her... Indeed she has lost 1 kilo. The OASIS health monitoring assistant detects the change in her weight, as well as the excessive activity pattern, and instructs the nutritional advisor to adjust Stella’s diet accordingly, i.e. provide a calorie-rich but light nutritional plan for the coming weeks.

Devices/technologies/infrastructure addressed in scenario

The key device for the delivery of mobile services is the PDA. Its effective use will require the creation of a telematics platform which provides wireless cover for users as well as location-based services. Other devices / technologies used include: Light wearable monitoring devices, Smart PDA or Smartphone, Sensor Body Area Network, Always on connectivity; seamless connectivity, Remote Home control.
### A.2 uAAL security aspects

For the UniversAAL (uAAL) project a number of relevant security aspects have been identified. Table 4 presents the overview of these aspects as it was committed to the uAAL project. The aspects are written down like use cases since this work was done as part of the use case and scenario collection phase in uAAL. The rightmost column mentions the project from which the item was collected; PHILIPS means that the item was not collected from an existing project but newly introduced.

<table>
<thead>
<tr>
<th>Subcategory</th>
<th>Title</th>
<th>Brief description</th>
<th>Project</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEC: identification</td>
<td>username/password</td>
<td>User proves identity using something he knows (e.g. password)</td>
<td>OASIS</td>
</tr>
<tr>
<td></td>
<td>token/certificates</td>
<td>Users/devices/services prove their identity using something they have (e.g. hardware RFID token or digital certificate)</td>
<td>SOPRANO</td>
</tr>
<tr>
<td></td>
<td>biometrics</td>
<td>User proves identity using something he is (biometrics)</td>
<td>PERSONA, SOPRANO</td>
</tr>
<tr>
<td></td>
<td>Single Sign On (SSO)</td>
<td>User proves identity to various services once per session using federated identity management.</td>
<td>PHILIPS</td>
</tr>
<tr>
<td>SEC: authorization</td>
<td>Access Control Lists (ACL)</td>
<td>Authorization to access a resource is granted/denied using an ACL or access to a resource is granted/denied based on permissions linked to the role of the subject using RBAC</td>
<td>PERSONA</td>
</tr>
<tr>
<td></td>
<td>usage control</td>
<td>Authorization to access, disclose, and use a resource is granted/denied based on a policy that is included with the resource (sticky policies), possibly enforced by encryption</td>
<td>PHILIPS</td>
</tr>
<tr>
<td></td>
<td>delegation</td>
<td>User/service is authorized to act on behalf of another user</td>
<td>REMOTE</td>
</tr>
<tr>
<td></td>
<td>context aware policies</td>
<td>The context of the access request is taken into account upon making an authorization decision</td>
<td>PHILIPS</td>
</tr>
<tr>
<td>SEC: privacy</td>
<td>control data</td>
<td>A user has an interface to edit and delete personal data that is stored in the system</td>
<td>PHILIPS</td>
</tr>
<tr>
<td></td>
<td>secure transfer</td>
<td>Sensitive data that is transmitted over an untrusted network is protected to preserve confidentiality and integrity</td>
<td>PHILIPS</td>
</tr>
<tr>
<td></td>
<td>secure storage</td>
<td>Sensitive data that is stored in the system is protected to preserve confidentiality and integrity</td>
<td>PHILIPS</td>
</tr>
<tr>
<td>SEC: configuration</td>
<td>An expert (administrator) configures default security and privacy settings and policies of the system. Also performs key management of the various parties in the system.</td>
<td>OASIS</td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>skilled peer</td>
<td>A skilled peer helps a user fine tune security and privacy settings and policies of the system to personal preference</td>
<td>PHILIPS</td>
<td></td>
</tr>
<tr>
<td>user and consent control</td>
<td>The user applies predefined policies/settings to new data/situations. The user can provide, withdraw and change consent for authorization (like delegation) and his data usage control. A user-friendly interface is available to perform this.</td>
<td>OASIS</td>
<td></td>
</tr>
<tr>
<td>policy negotiation</td>
<td>The user-service or service-service can negotiate the policies that need to be applicable in an interaction.</td>
<td>REMOTE</td>
<td></td>
</tr>
<tr>
<td>SEC: tracking store logs and event reporting</td>
<td>The system logs all access (requests) to sensitive data and other security/privacy related actions such as authentication attempts. Events that are conflicting or unusual are reported for audit.</td>
<td>PHILIPS</td>
<td></td>
</tr>
<tr>
<td>Audit</td>
<td>The administrator or external party can analyze access logs to detect cases of misuse (routinely, or in case of suspected misuse from event reporting)</td>
<td>PHILIPS</td>
<td></td>
</tr>
<tr>
<td>Observability</td>
<td>A user can keep track of what data about him/her is stored in the system, where it is stored and who has accessed it</td>
<td>PHILIPS</td>
<td></td>
</tr>
<tr>
<td>non-repudiation</td>
<td>Data can be traced back to its origin in a reliable way for example using digital signatures.</td>
<td>PHILIPS</td>
<td></td>
</tr>
</tbody>
</table>

*Table 4 uAAL Security Aspects*
A.3 XACML examples

This appendix presents some example XACML fragments to illustrate how certain AAL concepts can be captured in XACML policies. First some fragments about role based access control to service methods and then some examples of more expressive policies targeting specific resources and subjects. The rbac examples are based on the examples in the XACML RBAC profile [16] and the other examples are based on the examples in the XACML standard [15]. These documents provide more detailed explanations on the syntax used in the XACML fragments.

Role based access control to service methods

The following four fragments show a policy structure that follows the XACML RBAC profile [16]. These policies describe a few permissions for a service called `FitnessProgramService` with methods `createFitnessProgram` and `getFitnessProgram`. The policies model the following textual access rules (note the hierarchy).

Someone with the `assisted-person` role can call the `getFitnessProgram` method of the `FitnessProgramService` service. Someone with the `fitness-coach` role can call the `createFitnessProgram` method of the `FitnessProgramService` service and also has all the permissions associated with the `assisted-person` role.

The `Role-PolicySet` for the `fitness-coach` role links subjects with that role to the associated `Permission-PolicySet`.

```xml
<PolicySet xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
PolicySetId="RPS:fitness-coach:role"
  <Target>
    <Subjects>
      <Subject>
        <SubjectMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:anyURI-equal">
          <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#anyURI">
            urn:oasis:names:tc:xacml:2.0:subject:role:fitness-coach
          </AttributeValue>
          <SubjectAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:2.0:subject:role"
            DataType="http://www.w3.org/2001/XMLSchema#anyURI" />
        </SubjectMatch>
      </Subject>
    </Subjects>
    <!-- Use permissions associated with the fitness-coach role -->
    <PolicySetIdReference>PPS:fitness-coach:role</PolicySetIdReference>
  </Target>
</PolicySet>
```

The `Role-PolicySet` for the `assisted-person` role links subjects with that role to the associated `Permission-PolicySet`.

```xml
<PolicySet xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
PolicySetId="RPS:assisted-person:role"
  <Target>
    <Subjects>
      <Subject>
        <SubjectMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:anyURI-equal">
          <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#anyURI">
            urn:oasis:names:tc:xacml:2.0:subject:role:assisted-person
          </AttributeValue>
          <SubjectAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:2.0:subject:role"
            AttributeId="urn:oasis:names:tc:xacml:2.0:subject:role:assisted-person"
            DataType="http://www.w3.org/2001/XMLSchema#anyURI" />
        </SubjectMatch>
      </Subject>
    </Subjects>
    <!-- Use permissions associated with the assisted-person role -->
    <PolicySetIdReference>PPS:assisted-person:role</PolicySetIdReference>
  </Target>
</PolicySet>
```
The **Permission-PolicySet** for the *fitness-coach* role permits calling the `createFitnessProgram` method of the *FitnessProgramService* service and includes the permissions of the *assisted-person* role by referencing the *Permission-PolicySet* for the *assisted-person* role.

```
<PolicySet xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
PolicySetId="PPS:fitness-coach:role"
PolicyCombiningAlgId="urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:permit-overrides">
  <!-- Permissions specifically for the fitness-coach role -->
  <Policy PolicyId="Permissions:specifically:for:the:fitness-coach:role"
    RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:permit-overrides">
    <!-- Permission to call the createFitnessProgram method of the FitnessProgramService service -->
    <Rule RuleId="AAL:example:rule:1"
      Effect="Permit">
      <Target>
        <Resources>
          <Resource>
            <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
              <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">FitnessProgramService</AttributeValue>
              <ResourceAttributeDesignator Attribute-Id="urn:oasis:names:tc:xacml:2.0:resource:serviceID" DataType="http://www.w3.org/2001/XMLSchema#string"/>
            </ResourceMatch>
          </Resource>
        </Resources>
        <Actions>
          <Action>
            <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
              <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">createFitnessProgram</AttributeValue>
              <ActionAttributeDesignator Attribute-Id="urn:oasis:names:tc:xacml:1.0:action:methodID" DataType="http://www.w3.org/2001/XMLSchema#string"/>
            </ActionMatch>
          </Action>
        </Actions>
      </Target>
    </Rule>
  </Policy>
  <!-- Include permissions associated with assisted-person role -->
  <PolicySetIdReference>PPS:assisted-person:role</PolicySetIdReference>
</PolicySet>
```
The Permission-PolicySet for the assisted-person role permits calling the getFitnessProgram method of the FitnessProgramService service.

```xml
<PolicySet xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
PolicySetId="PPS:assisted-person:role"
PolicyCombiningAlgId="urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:permit-overrides">
    <!-- Permissions specifically for the assisted-person role -->
    <Policy PolicyId="Permissions:specifically:for:the:assisted-person:role"
RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:permit-overrides">
        <!-- Permission to call the getFitnessProgram method of the FitnessProgramService service -->
        <Rule RuleId="AAL:example:rule:2"
Effect="Permit">
            <Target>
                <Resources>
                    <Resource>
                        <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
                                FitnessProgramService
                            </AttributeValue>
                            <ResourceAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:2.0:resource:serviceID" DataType="http://www.w3.org/2001/XMLSchema#string"/>
                        </ResourceMatch>
                    </Resource>
                </Resources>
                <Actions>
                    <Action>
                        <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
                                getFitnessProgram
                            </AttributeValue>
                            <ActionAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:action:methodID" DataType="http://www.w3.org/2001/XMLSchema#string"/>
                        </ActionMatch>
                    </Action>
                </Actions>
            </Target>
        </Rule>
    </Policy>
</PolicySet>
```
Granting access to the owner of a resource

The following policy and request fragments show how XACML can be used to grant a user read access to a fitness program of which the user is the owner. In this example the owner is specified explicitly as an attribute in the request. The examples in the XACML standard show how such information can also be retrieved from the resource content [15].

The **Policy** targets resources of type *FitnessProgram* and permits the *read* action if a certain condition is met. The condition matches the *user-id* attribute of the subject specified in the request to the *owner-id* attribute of the resource specified in the request.

```xml
  <Target />
  <Rule RuleId="AAL:example:rule:3" Effect="Permit">
    <Target>
      <Resource>
        <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
          <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
            FitnessProgram
          </AttributeValue>
          <ResourceAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:2.0:resource:type" DataType="http://www.w3.org/2001/XMLSchema#string" />
        </ResourceMatch>
      </Resource>
      <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 AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id" DataType="http://www.w3.org/2001/XMLSchema#string" />
          </ActionMatch>
        </Action>
      </Actions>
    </Target>
    <Condition>
      <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:and">
        <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
          <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
            read
          </AttributeValue>
          <AttributeId "urn:oasis:names:tc:xacml:2.0:example:attribute:owner-id" />
        </Apply>
        <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-one-and-only">
          <AttributeId "urn:oasis:names:tc:xacml:2.0:example:attribute:user-id" />
          <ResourceAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:owner-id" DataType="http://www.w3.org/2001/XMLSchema#string" />
        </Apply>
        <VariableReference VariableId="17590035" />
      </Apply>
    </Condition>
  </Rule>
</Policy>
```
The Request specifies the user-id of the subject that requests access, it specifies the type and owner-id of the resource to which access is requested and it specifies the action-id of the requested action.

```
<Request xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os"
  <Subject>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject-category">
      <AttributeValue>urn:oasis:names:tc:xacml:1.0:subject-category:access-subject</AttributeValue>
    </Attribute>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:user-id">
      <AttributeValue>AP123456</AttributeValue>
    </Attribute>
  </Subject>
  <Resource>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:owner-id">
      <AttributeValue>AP123456</AttributeValue>
    </Attribute>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:type">
      <AttributeValue>FitnessProgram</AttributeValue>
    </Attribute>
  </Resource>
  <Action>
    <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id">
      <AttributeValue>read</AttributeValue>
    </Attribute>
  </Action>
  <Environment />  
</Request>
```

Granting access to location information in case of emergency

The following policy and request fragments show how XACML can be used to grant a caregiver read access to the detailed location of an assisted person in case of an emergency with the obligation to write a log entry.

The Policy targets resources of type DetailedLocation and permits the read action if the state attribute of the environment matches emergency and if a certain obligation is fulfilled. The obligation consists of writing a log entry according to a certain format: “The detailed location of: owner-id has been accessed by: user-id in an emergency environment”.

```
<Policy xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
  PolicyId="AAL:example:policyid:2"
  RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:deny-overrides">
  <Target />
  <Rule RuleId="AAL:example:rule:3"
    Effect="Permit">
    <Target>
      <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
        <AttributeValue>DataType="http://www.w3.org/2001/XMLSchema#string">
          DetailedLocation</AttributeValue>
        <ResourceAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:resource:type" 
          DataType="http://www.w3.org/2001/XMLSchema#string" />
      </ResourceMatch>
    </Target>
  </Rule>
</Policy>
```
<Actions>
  <Action>
    <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal"
      xmlns="http://www.w3.org/2001/XMLSchema#string">
      <AttributeValue>
        read
      </AttributeValue>
    </ActionMatch>
  </Action>
</Actions>

<Environments>
  <Environment>
    <EnvironmentMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal"
      xmlns="http://www.w3.org/2001/XMLSchema#string">
      <AttributeValue>
        emergency
      </AttributeValue>
    </EnvironmentMatch>
  </Environment>
</Environments>

<Target>
  <Rule>
    <Obligations>
      <Obligation ObligationId="urn:oasis:names:tc:xacml:example:obligation:write-log"
        xmlns="urn:oasis:names:tc:xacml:example:obligation:write-log">
        <AttributeAssignment AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:text"
          xmlns="http://www.w3.org/2001/XMLSchema#string">
          The detailed location of:
        </AttributeAssignment>
        <AttributeAssignment AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:text"
          xmlns="http://www.w3.org/2001/XMLSchema#string">
          has been accessed by:
        </AttributeAssignment>
        <AttributeAssignment AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:text"
          xmlns="http://www.w3.org/2001/XMLSchema#string">
          in an emergency environment
        </AttributeAssignment>
      </Obligation>
    </Obligations>
  </Rule>
</Target>
The Request specifies the user-id and role-id of the subject that requests access, it specifies the type and owner-id of the resource to which access is requested, it specifies the action-id of the requested action and it specifies the state of the environment.

```xml
<Request xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    <Subject>
        <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject-category" DataType="xs:anyURI">
            <AttributeValue>urn:oasis:names:tc:xacml:1.0:subject-category:access-subject</AttributeValue>
        </Attribute>
        <Attribute AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:role-id" DataType="xs:string">
            <AttributeValue>caregiver</AttributeValue>
        </Attribute>
        <Attribute AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:user-id" DataType="xs:string">
            <AttributeValue>CG98765</AttributeValue>
        </Attribute>
    </Subject>
    <Resource>
        <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:type" DataType="xs:string">
            <AttributeValue>DetailedLocation</AttributeValue>
        </Attribute>
        <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:owner-id" DataType="xs:string">
            <AttributeValue>AP123456</AttributeValue>
        </Attribute>
    </Resource>
    <Action>
        <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id" DataType="xs:string">
            <AttributeValue>read</AttributeValue>
        </Attribute>
    </Action>
    <Environment>
        <Attribute AttributeId="urn:oasis:names:tc:xacml:2.0:environment:state" DataType="xs:string">
            <AttributeValue>emergency</AttributeValue>
        </Attribute>
    </Environment>
</Request>
```