University of Plymouth

PEARL **https://pearl.plymouth.ac.uk**

Faculty of Science and Engineering School of Engineering, Computing and Mathematics

2019

Privacy Enhancing Data Access Control for Ambient Assisted Living

Kuijs, H

http://hdl.handle.net/10026.1/20461

10.5220/0007735804480455
Proceedings of the 9th International Conference on Cloud Computing and Services Science
SCITEPRESS - Science and Technology Publications

All content in PEARL is protected by copyright law. Author manuscripts are made available in accordance with publisher policies. Please cite only the published version using the details provided on the item record or document. In the absence of an open licence (e.g. Creative Commons), permissions for further reuse of content should be sought from the publisher or author.

Privacy Enhancing Data Access Control for Ambient Assisted Living

Hendrik Kuijs[1], Timo Bayer[1], Christoph Reich[1], Martin Knahl[1] and Nathan Clarke[2]

[1] *Institute for Cloud Computing and IT Security, Furtwangen University, Furtwangen, Germany*

[2] *Centre for Security, Communications and Network Research, Plymouth University, Plymouth, U.K.*

Keywords: Privacy, Data Flow, OSGi, Access Control, Context Aware, Ambient Assisted Living.

Abstract: As private data is key to applications in the field of Ambient Assisted Living, access control has to be in place to regulate data flows within the environment and to preserver the privacy of a user. We present a data access control system based on an easy to understand policy language with the ability to be extended by context information. Context information are enabling applications in the field of Ambient Assisted Living to adapt their behaviour to temporal, emergency or environmental conditions. The system is able to monitor and control data flows in OSGi environments by proxy services, without the need of modifying the core platform or the service logic of bundles. This is necessary to inform data subjects, to enable the data subject to control the environment and to enforce data access policies that are compatible with legal requirements.

1 INTRODUCTION

Ambient Assisted Living (AAL) emerged out of technologies which are known as Ambient Intelligence (AMI), the technological foundation for sensitive and adaptive environments that have the ability to respond to actions of persons or real world objects and cater for their needs. This development is fostered by two social and political phenomena: The increasing average life expectancy and a decrease of birth-rate leads to a continuously shrinking proportion of the young working population in developed countries worldwide (United Nations, 2001). Politics try to compete with this trend by introducing programs for new nursing or day-care facilities (Llena-Nozal et al., 2011). But without trained personnel and already existing facilities one key concept is to minimize the needed time-span for professional care facilities by introducing technology in the known living environment of the target group. Surveys show that this trend is supported by elderly people who want to stay at home as long as possible (Grauel and Spellerberg, 2007). With the introduction of supporting systems in the home of the elderly, new concerns arise about the protection of privacy. Focus groups and users are feeling observed and do not want an uncontrollable surveillance system which they don't trust incorporated in their living environment (Dario and Cavallo, 2014, pp. 89 ff.)(Rosencrantz et al., 2015).

In our work we show a system that can raise transparency and hand over the control on the private data collected and processed in an AAL system back to the user.

Section 2 presents related work in the field of privacy for AAL and data security in OSGi, the main middleware for platforms in AAL. In Section 3 a new solution for data access management is presented in three parts: A new module for monitoring and control in OSGi, the access control system and the basics of the new privacy policy. An evaluation of the main concepts of the policy language and access control system is shown in Section 4, followed by conclusions and remarks on future work in Section 5.

2 RELATED WORK

The AALIANCE (Broek et al., 2010) determined in their AAL Roadmap of 2010 that in existing AAL systems a huge amount of data is collected by sensors in the living environment to assist the user in his daily life. These data (vital data, location data) is then aggregated with other information (from communication systems, organizational tools, or household appliances) to draw conclusions about the current situation, the health state and the well-being of the user. In many cases this information is made available to relatives and informal caregivers (primary group of stakeholders in AAL (Dario and Cavallo, 2014)), or professional care providers, e.g. doctors or nurses (secondary group of stakeholders).

Kuijs, H., Bayer, T., Reich, C., Knahl, M. and Clarke, N.
Privacy Enhancing Data Access Control for Ambient Assisted Living.
DOI: 10.5220/0007735804480455
In *Proceedings of the 9th International Conference on Cloud Computing and Services Science (CLOSER 2019)*, pages 448-455
ISBN: 978-989-758-365-0

Copyright © 2019 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

2.1 Privacy in Ambient Assisted Living

According to (Memon et al., 2014) privacy preservation is not yet seen as a quality attribute of AAL systems. Although security (encryption and access control) is of importance for acceptability among the end-users, other major quality attributes are usability, reliability, maintainability, efficiency, safety, accuracy, and dependability.

Braun et al. show in their study that the target audience emphasize that they have a sense of shame when confronted with a video system, but are at the same time willing to accept sharing of personal data with other entities in AAL environments (Braun et al., 2016). In their view this could be due to the abstract nature of the term data privacy, but the immediate feeling of shame. However a loss of data to criminal subjects is seen as a real threat by the target audience.

J. van Heek et. al directly focus on data security and privacy in their study (van Heek et al., 2017). Although the average needs for data security and privacy were on a moderate level with regard to all interviewed participants, they stated that they wanted to have control over the access of their data, see data security as an important attribute for AAL, and realize the possibility of abuse of leaked data. The own privacy of the participants is as important to them as the privacy of others and they want to control their own privacy.

Stutz et al. define measures that have to be implemented by system developers and providers to support data protection and information security in AAL systems (Stutz et al., 2016). Their work is based on the approach of Privacy by Design and by Default (Cavoukian, 2010), that is also included as a central concept in the EU General Data Protection Regulation (The European Parliament and the Council of the European Union, 2016). They propose a consent based system where a user has to be informed about the data usage and no data is allowed to be transferred, stored or processed without the consent of the user.

In this work a model and a first implementation for data protection and data flow control in AAL systems is presented based on OSGi.

2.2 Data security in OSGi

The AALIANCE consortium recommends using a modular and open approach for new services and software in the context of AAL (AALIANCE, 2010). For remote management for software upgrades and installing new software to an existing platform, OSGi is seen as the most adopted middleware in AAL environments The central feature of OSGi is the possibility to include, update, start and stop services as modular bundles during runtime without the need for a restart of the whole environment. Existing systems based on OSGi are among others universAAL (Sadat et al., 2013), openAAL (Wolf et al., 2010) or Soprano (Schmidt et al., 2009).

The downside to the flexible extensibility and adoptability of OSGi based systems is the vulnerability to malicious software extensions that can violate security concepts or disrupt the whole system. Huang et al. propose an Advanced OSGi Security Layer (AOSL) to audit bundles and detect anomalies during runtime (e.g. denial-of-service attacks by exceeding load or thread-duration) (Huang et al., 2007). A similar way of anomaly detection is presented by Wang et al. (Wang et al., 2012): By using a Service Monitor and JVM Monitor and service proxies they keep track of the execution time and memory consumption of services to provide assistance in finding anomalies. Although the presented work does not focus in anomalie detection, service proxies are used to intervene in the platform processes without the need of modifying the deployed bundles.

A solution of Security-by-Contract (SxC) in OSGi (Gadyatskaya et al., 2012) is presented by Gadyatskaya et al. The manifest-file of each bundle is extended by a contract that contains information about the functional requirements and the accessibility by other bundles of the OSGi environment and is evaluated during deployment by the Platform Security Policy. If the evaluation fails and contracts are incompatible with the current platform policy, the bundle will not be deployed. Putting initial requirements and further descriptive entries in the manifest-file is also of interest in the presented approach to further detail the purpose and the implications of a new installed service. Parrend et al. present a Privacy-Aware Service Integration (Parrend et al., 2007). They propose a framework for ensuring privacy policy compliant services and a metadata language for privacy checks in an OSGi environment. The service provider has to ensure that all offered services are secure and tested prior installation and that they provide the information needed by the platform during deployment and for privacy evaluations. Service restrictions are then realized by so called *RestrictedContext*: Bundles are only able to access other bundles they are allowed to discover by policy.

Due to the believe that it is not easy to control both, the service provider and the client, in a way to guarantee a sound, secure and privacy consistent platform, the presented solution has lesser necessary preparatory work and basic assumptions.

3 DATA ACCESS MANAGEMENT

The security and privacy enhanced cloud infrastructure for Ambient Assisted Living (SpeciAAL) is a Platform as a Service (PaaS) approach for an AAL environment that is based on OSGi. (Kuijs et al., 2015; Kuijs et al., 2016). As personal information about the user is the key data for AAL, security measures for data protection (e.g. data access, access roles and privacy policies) have to be implemented to secure and monitor data access and data transfer at all time.

To protect privacy, design strategies can be used when designing a system from scratch. In general, a design strategy describes an abstract approach to achieving a particular design goal. A listing of eight different strategies is presented by the European Union Agency for Network and Information Security (ENISA) (Danezis et al., 2015). In addition, each strategy identifies design patterns for implementation. In our approach we focus on three defined strategies:

- Inform: Data subjects should always be informed about which information is processed, for what purpose, and by which means when using a system. This necessary information includes the ways of data protection, the overall security of a system, possible sharing of data with third parties, and personal data access rights. This strategy is supported by different policy languages, mechanisms, and frameworks.
- Control: To gain user consent for processing of personal data the user has to have agency to view, update or delete personal data at any given time. Above this a user can control and define the data that is processed and whether to use a certain system. Design Patterns for establishing consent are a user centric identity management, and end-to-end encryption support control. An example for intervenability is a sensor in a smart home environment that can be deactivated during a visit of friends for privacy reasons.
- Enforce: To ensure that a privacy policy is in place, there should be always be one enforced by default that is compatible with legal requirements. This strategy can be implemented by access control, or privacy rights management.

The described architecture is based on two main components. A OSGi component that is enabling the user to monitor and control the environment and the authorization system. On top of these we developed a policy language that is (as described earlier) able to be enriched with context conditions to support the special needs of users in AAL environments.

3.1 Monitoring

In order to protect the potentially personal data inside the platform, monitoring has to be initially implemented at all critical points. It must take place between installed bundles, between bundles and the database, but also at interfaces to other parts of the AAL system (e.g., web-interfaces or interfaces to external services). In the presented approach this is realized by a central protection bundle, that automatically creates a proxy service during the deployment of a bundle in OSGi that mirrors the original bundle with all its methods. By injecting logging facilities inside the proxied bundles it is possible to log access to the data source of the service but also indirect access between bundles and to external interfaces. The gathered information of the logging implementation can then be stored and presented to the user in a simplified and non-technical way. In a second step these proxies can be extended by decision logic to allow or deny communication with other services within the platform. This enables us to have detailed control over the data flows inside the environment without the need to have full control of all installed bundles before deployment or doing security checks and code inspections of all bundles to whitelist them for use in the AAL environment.

The OSGi-Core specification describes an Event Listener Hook, to respond to events in the Service Registry (The OSGi Alliance, 2014, Chapter 55.4). This hook is used to inject program code in the service registration process. The automatic generation of the proxy is triggered and by the use of Java Reflection (Glen McCluskey, 1998) a proxy class is generated that acts on behalf of the actual class and accepts their calls. It offers the same methods as the original class and can be extended by code functionality to log its activities and control its behaviour. To ensure that the proxy and not the original bundle is called by other bundles in the OSGi environment, the proxy is prioritized higher in the Service Registry of OSGi (The OSGi Alliance, 2014).

This is realized by implementing a *Component Protection Bundle* within the OSGi environment and allows us to apply the generated access rules to the system. Figure 1 shows the sequence diagram of the access between a caller bundle and the service bundle through the presented proxy mechanism. In this example the requested *name* is only accessible if the access is authorized inside the *Proxy Service*.

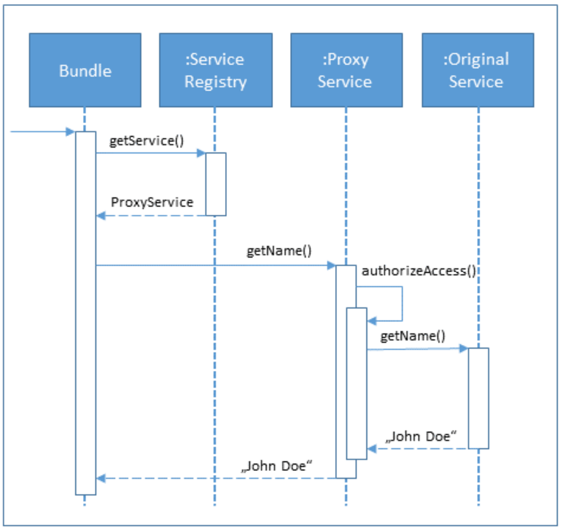


Figure 1: Sequence diagram for access control.

3.2 Access Control

To realize the design strategies "control" and "enforce" (Danezis et al., 2015) the presented approach is an access control system based on the established XACML reference architecture (Kafura, 2004). The structure of the developed system, as shown in Figure 2, is divided into two responsibilities, the creation and administration of authorization policies and the securing of the access patterns defined during a request for stored data. A particular focus of the developed platform was to ensure the highest possible interoperability for the integration of a large number of individual services. In order to keep it simple to operate for the target audience, the system also has a central user interface for administering the policies. The purpose of the user interface is to provide the user with the highest possible level of transparency and comprehensive control over the granted authorizations while abstracting the technical details. Thus, the user is always able to manipulate existing permissions, remove them when deactivating a service or grant new permissions.

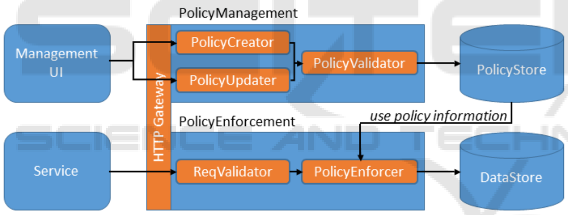


Figure 2: Architecture overview of the access control system.

The *PolicyManagement* module contains the subcomponents for creating and managing authorization policies. The creation of a policy can be done via user-friendly input forms within the user interface or by importing pre-defined XML files. In the case of form-based authoring, the information about the authorizations to be granted is provided via HTTP endpoints of the sub-component *PolicyCreator*. Using the subcomponent *PolicyValidator*, these are first checked for their semantic correctness and then a corresponding XML file is generated. However, a particular focus of the implementation of this component is the ability to integrate pre-defined authorization policies by allowing services to provide them directly to the user during service deployment. To preserve transparency and control for the user, the access rules defined within the XML files are initially interpreted by the platform and presented in a readable form within the user interface. The user thus has the option of easily disabling individual rules or adapting them to his wishes. One speciality of the user interface is the possbility of adding context to the access rules, that may permit the access of data or functionality only at a special time, a special location or in special occasions. If new policies are inserted in this way as an XML file, an additional review of syntactic correctness will take place.

The created policies are stored within the *PolicyStore*, in which the contents of the authorization policies are transferred to a relational data model for high-performance and targeted querying. Due to the defined interfaces, the platform supports both local and remote database systems, which can be used individually depending on the desired form of the platform. Since the allowed access patterns can also change over time, an additional option for manipulating existing policies has been integrated into the platform. This is done via the component *PolicyUpdater*. Manipulating existing policies is equivalent to creating them by updating existing information either through user interface forms or directly within the corresponding XML files. Such a case has relevance both in the delivery of a new version of the service concerned and in the need for manual adaptation of the rules by the user. The associated component also provides this functionality via HTTP endpoints, which, after successful completion, result in an update of *PolicyStore* data. Because of the resulting interfaces, it is also possible to integrate this functionality into higher-level applications, such as a central instance for managing all permissions within a given domain.

The other part of the system is responsible for the enforcement of the created authorization policies within the module *PolicyEnforcement*. An entity attempts to access the stored and protected data within the *DataStore* by specifying its own identity, the desired resource and the respective access pattern by calling the defined HTTP endpoints of the *Request-Validator*. The indication of the identity has particular relevance to counteract possible abuse. For this, a certificate-based matching of the identities within the *RequestValidator* takes place. If the given identity can be confirmed, the request is forwarded to the *PolicyEnforcer* component. Using the stored policies, this component checks whether the requested access corresponds to the defined authorizations and, if positive, responds with the requested data. In the negative case, access to the data is denied. The verification of the permitted access patterns is done in a similar way to a conventional firewall systems in decreasing granularity. As soon as a corresponding rule has been found, access is immediately permitted or denied. This allows for efficient processing of data access by considering only a subset of all rules. If the data access is permitted, the data can be queried by use of the connector component of the respective persistence media.

3.3 Privacy Policy

The privacy policies that are currently discussed, are the Platform for Privacy Preferences (P3P) (Cranor et al., 2006), S4P (Becker et al., 2010), and SIMple Privacy Language (SIMPL)(le Métayer, 2009). P3P describes privacy information of websites, such as editor information, collected data, dispute scenarios and retention time of data. S4P on the other hand is used for privacy descriptions of services and what personal information is used for what purpose. SIMPL is used to specify preferences and policies with a small subset of English. Only a small part is already specified for future use and it is not easy to comprehend for humans. To keep the policies in a natural language and easy to understand for the target group of AAL the approach is to give answers to the 5W1H questions (shown in Figure 3). On the left side the questions are listed (each followed by a corresponding example), divided in a service description and access description part. On the right side the technical implementations and corresponding configuration options are shown. *Title* and *Description* can be found in the manifest-file of a bundle, data defines the accessed data category of the information to be processed by the service, and the Access-Rights are based on the CRUD acronym. Special to our use case in AAL is the possibility to add context to each policy. In this early stage we are supporting environment based and temporal context, where environment considers the location of the user, external influences, or emergency situations. Temporal context-based access could be periodic, e.g. during working hours on week-days, to preserver the users privacy during night and on weekends.

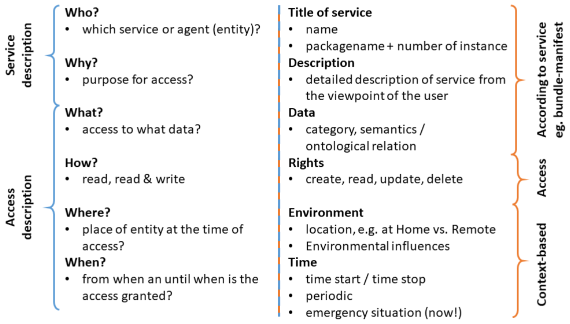

Service description
Access description
Who?
• which service or agent (entity)?
Why?
• purpose for access?
What?
• access to what data?
How?
• read, read & write
Where?
• place of entity at the time of access?
When?
• from when an until when is the access granted?
Title of service
• name
• packagename + number of instance
Description
• detailed description of service from the viewpoint of the user
Data
• category, semantics / ontological relation
Rights
• create, read, update, delete
Environment
• location, e.g. at Home vs. Remote
• Environmental influences
Time
• time start / time stop
• periodic
• emergency situation (now!)
According to service eg. bundle-manifest
Access
Context-based

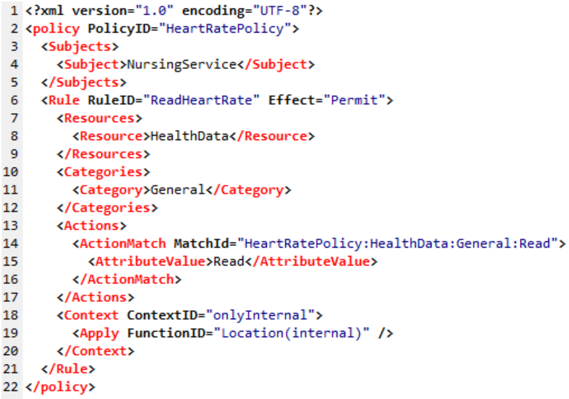
Figure 3: Proposed privacy policy based on 5W1H questions.

To put it in natural language:

An entity is given access to data for some defined task, if the particular context applies - otherwise access is forbidden.

This flexibility leads to very fine granular access control mechanisms that can be applied by applying our policy language.

The basis of our proposed access control system is the definition of the authorization policy. The chosen structure of the policy is based on the lightweight representation of the eXtensible Access Control Markup Language (XACML) (Kafura, 2004). XACML is a declarative, attribute-based markup language for the presentation of authorization policies and is characterized in particular by its high interoperability, extensibility and the resulting general validity. The main aspect of the definition of an authorization policy are the data to be protected, which vary depending on the specific application scenario, the sensitivity and purpose. To efficiently map this variability, the definition of an authorization policy is based on the requesting entity. This design choice results in increased flexibility in which new services can be added or their privileges can be revoked or manipulated without global impact.



```
<?xml version="1.0" encoding="UTF-8"?>
<policy PolicyID="HeartRatePolicy">
  <Subjects>
    <Subject>NursingService</Subject>
  </Subjects>
  <Rule RuleID="ReadHeartRate" Effect="Permit">
    <Resources>
      <Resource>HealthData</Resource>
    </Resources>
    <Categories>
      <Category>General</Category>
    </Categories>
    <Actions>
      <ActionMatch MatchId="HeartRatePolicy:HealthData:General:Read">
        <AttributeValue>Read</AttributeValue>
      </ActionMatch>
    </Actions>
    <Context ContextID="onlyInternal">
      <Apply FunctionID="Location(internal)" />
    </Context>
  </Rule>
</policy>
```

Figure 4: Monitoring data access in a distributed OSGi environment.

This is depicted by the attribute *Subjects* shown in Figure 4. In addition to defining the services involved, a policy includes a variable set of rules that are used to determine the desired access permissions. A rule always refers to a concrete resource that can be further restricted by specifying a *Category* for fine granular authorization assignment. In a first attempt to categorize the data, the main concepts of the implemented ontology are used (Fredrich et al., 2014): Personal information, contacts, capabilities, preferences, interests, activities, health information, education and environment. The heart of such a defined rule is the attribute *Actions*. Within this the permitted access patterns can be defined (according to the CRUD acronym).

To ensure context-sensitive access authorizations, the defined rules can be supplemented with additional conditions by means of the *Context* attribute. The available conditions are defined by the Privacy Policy and include, for example, mapping time- or location-dependent access patterns. In the presented authorization policy includes a location restriction of the permitted accesses.

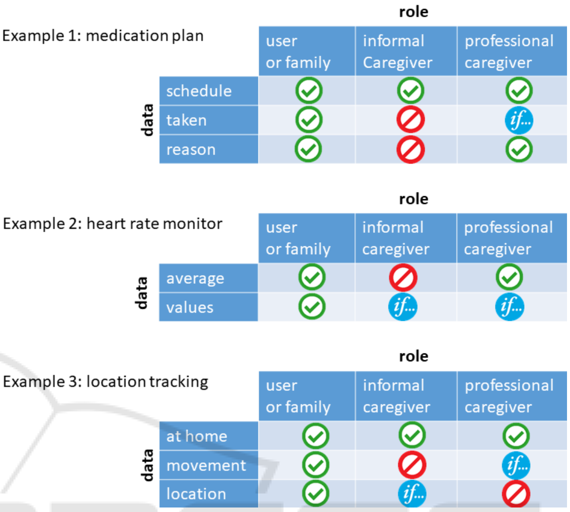
The default behavior of the developed solution prevents any data access without an appropriate policy, which is why all access patterns of a service must be defined within a corresponding rule. The individual rules are interpreted sequentially and therefore have to be defined in descending access granularity. The described structure of an authorization policy can be arbitrarily extended to represent more complex access authorizations and has the possibility to be converted into a format conforming to the XACML specification that provides complete interoperability with deviating implementations.

4 EVALUATION

This section is divided in two parts. First, the application of the policy language is clarified with three examples. After this it is examined if the overall system meets the functional and non functional requirements for an access control system in AAL.

4.1 Application of Policy

The first example is an online medication plan, that is connected with an drug dispenser. The three information that are accessible for this service are: the *schedule* for the drug ingestion, the information whether or not the drugs have been *taken*, and the *reason* for the medication. The user always has the full access on his own data. Informal caregivers (e.g. a neighbor) may have access to the schedule (name and time to take the drug), professional caregivers (e.g. nurses or doctors) also know the reason for the medication and are allowed to control the taking of the drugs, but only (context dependency) if they are visiting the patient. Figure 5 shows the example and the granularity of access rights, where the "if..." icon is indicating the described context based attribute to the access policy.



Example 1: medication plan

data	role: user or family	role: informal Caregiver	role: professional caregiver
schedule	✓	✓	✓
taken	✓	⊘	if...
reason	✓	⊘	✓

Example 2: heart rate monitor

data	role: user or family	role: informal caregiver	role: professional caregiver
average	✓	⊘	✓
values	✓	if...	if...

Example 3: location tracking

data	role: user or family	role: informal caregiver	role: professional caregiver
at home	✓	✓	✓
movement	✓	⊘	if...
location	✓	if...	⊘

Figure 5: Examples for policies based on three groups of stakeholders.

In our second example a nursing service has the right to access all *values* of a heart rate sensor only when accessing the system from within the users home (e.g. at the users own terminal during a daily visit). Otherwise the access rights only permit the access of *average* values. The informal caregiver has no rights to access the data, but this can be overridden by critical values. In this case the contextual portion of the access right could be set to *in emergency situations* (see Figure 5).

Our third example is based on a location tracking service, that is able to indicate, if the user is *at home*, *moving* inside his home, and can show his *location*. in this example all stakeholders have the right to see if the user is at home, professional caregiver are allowed to know if the user is moving during weekdays, and informal caregivers are allowed to access the location of the user in case of an emergency.

These brief examples show the flexibility of our developed data access policy language.

4.2 Compliance with Requirements

The Access Control component is able to meet different functional (F1 - F5) and non-functional requirements (NF1- NF3) that are briefly described below:

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

F1 **Introduction of a Central Control Authority for the Regulation of Access:** The main focus of the development of the access control system is the possibility to control all access to the stored data of the user. To make this possible, the platform is extended by a unit to control all data access. Since the accesses are made exclusively via the defined interfaces of the system, the data stores and data sources can be completely decoupled from the serving services.

F2 **Unified Integration of Variable Data Sources:** The variety of possible data sources, such as health gauges or recording the user's temporary environmental conditions, requires the definition of common interfaces for the persistence of the information. Thus, the data sources may use the functionality of the platform to store their recorded data, regardless of the persistence technology used. In addition to increased interoperability, this enables a flexible exchange of technologies

F3 **Possibility of Finely Granular Definition of the Permitted Access Patterns:** Since the access management on the basis of the data categories often has no satisfactory granularity, it is necessary to subdivide these further. Thus, the defined policy language always offers the possibility of assigning individual subcategories.

F4 **Definition of Context based Allowed Accesses:** The allowed data accesses can vary greatly depending on the service in their requirements. The high flexibility of the authorization policies used to regulate the access allows an individual definition of specific access rules, which go far beyond the usual CRUD operations in this environment.

F5 **User-friendly Definition and Manipulation of Authorization Policies:** According to the target group of the developed system, the simple usage is another requirement. Since the used structure of the authorization policies is primarily optimized for efficient machine readability, the system is extended by a natural language definition of the permitted access patterns.

NF1 **Lightweight:** When operating on resource-poor hardware, it must not be restricted in its functionality. This requirement is fulfilled by the streamlined structure of the authorization policies as well as a lightweight and individual implementation for machine processing.

NF2 **Interoperability:** The Access Control component enables the integration of versatile data sources and intelligent systems, as well as the consistent use of the platform by the requesting services. The services can access the platform via defined HTTP endpoints, creating a variety of integration options, such as the use of mobile device applications or web applications.

NF3 **Extensibility:** Due to the generic structure of the platform, a variety of application areas are possible. In order to meet the specific requirements of the respective domains, the implementation is based on a modular structure. This provides clear interfaces between the modules, which can thus be extended by adaptation or additional modules.

5 CONCLUSION AND FUTURE WORK

The presented work is an access control component in a platform for AAL that consists out of three main parts: An OSGi mechanism to intervene in the communication process of bundles and have the ability to monitor and control data flows on service level, an access control system to manage access rights based on rules, and a policy language to define these rules, that is extended by context to better suit the requirements of AAL use cases.

Although first tests of the presented implementation seem promising, a larger field test with focus groups and thorough performance tests are still to be carried out. One still missing part is the preparation of user friendly logging information to further enhance transparency of data flows.

REFERENCES

AALIANCE (2010). Ambient Assisted Living Roadmap - AALIANCE Project - Deliverable 2.7. chapter Enabling T, pages 95–96.

Becker, M., Malkis, A., and Bussard, L. (2010). S4P: A generic language for specifying privacy preferences and policies. *Microsoft Research*, pages 0–35.

Braun, A., Kirchbuchner, F., and Wichert, R. (2016). Ambient Assisted Living. In *eHealth in Deutschland*, pages 203–222. Springer Berlin Heidelberg, Berlin, Heidelberg.

Broek, G. v. d., Cavallo, F., Odetti, L., and Wehrmann, C. (2010). AALIANCE Ambient Assisted Living Roadmap. In *Ambient Intelligence and Smart Environments*, volume 6, page 110.

Cavoukian, A. (2010). The 7 Foundational Principles - Implementation and Mapping of Fair Information Practices.

Cranor, L., Dobbs, B., Egelman, S., Hogben, G., Humphrey, J., Langheinrich, M., Marchiori, M., Presler-Marshall, M., Reagle, J., Schunter, M., Stampley, D. A., and

Wenning, R. (2006). The Platform for Privacy Preferences 1.1 (P3P1.1) Specification.

Danezis, G., Domingo-Ferrer, J., Hansen, M., Hoepman, J.-H., Le Métayer, D., Tirtea, R., and Schiffner, S. (2015). Privacy and Data Protection by Design - from policy to engineering. Technical report, European Union Agency for Network and Information Security (ENISA), Heraklion, Greece.

Dario, P. and Cavallo, F. (2014). Ambient Assisted Living Roadmap - September 2014. Technical report, AALIANCE2.

Fredrich, C., Kuijs, H., and Reich, C. (2014). An Ontology for User Profile Modeling in the Field of Ambient Assisted Living. In Koschel, A. and Zimmermann, A., editors, *SERVICE COMPUTATION 2014, The Sixth International Conferences on Advanced Service Computing*, volume 5, pages 24–31. IARIA.

Gadyatskaya, O., Massacci, F., and Philippov, A. (2012). Security-by-Contract for the OSGi Platform. pages 364–375. Springer, Berlin, Heidelberg.

Glen McCluskey (1998). Using Java Reflection.

Grauel, J. and Spellerberg, A. (2007). Akzeptanz neuer Wohntechniken für ein selbstständiges Leben im Alter. In *Zeitschrift für Sozialreform*, volume Heft 2 Jg., pages 191–215.

Huang, C.-C., Wang, P.-C., and Hou, T.-W. (2007). Advanced OSGi Security Layer. In *21st International Conference on Advanced Information Networking and Applications Workshops (AINAW'07)*, pages 518–523. IEEE.

Kafura, D. (2004). OASIS eXtensible Access Control Markup Language (XACML) TC.

Kuijs, H., Reich, C., Knahl, M., and Clarke, N. (2016). A scalable architecture for distributed OSGi in the cloud. In *CLOSER 2016 - Proceedings of the 6th International Conference on Cloud Computing and Services Science*, volume 2.

Kuijs, H., Rosencrantz, C., and Reich, C. (2015). A Context-aware, Intelligent and Flexible Ambient Assisted Living Platform Architecture. In *Cloud Computing 2015: The Sixth International Conference on Cloud Computing, GRIDs and Virtualization*. IARIA.

le Métayer, D. (2009). A Formal Privacy Management Framework. *Formal Aspects in Security and Trust*, 5491:162–176.

Llena-Nozal, A., Mercier, J., Tjadens, F., and Colombo, F. (2011). *Help Wanted?* OECD Health Policy Studies. OECD Publishing.

Memon, M., Wagner, S. R., Pedersen, C. F., Aysha Beevi, F. H., and Hansen, F. O. (2014). Ambient Assisted Living healthcare frameworks, platforms, standards, and quality attributes.

Parrend, P., Frenot, S., and Hohn, S. (2007). Privacy-Aware Service Integration. In *IEEE International Conference on Pervasive Services*, pages 397–402. IEEE.

Rosencrantz, C., Kuijs, H., Reich, C., Weber-Fiori, B., and Winter, M. H.-J. (2015). Entwicklung einer Informations- und Kommunikationsplattform für ältere Menschen. *ENI 2015, IT im Gesundheits-, Pflege- und Sozialbereich: Qualität und Effizienz durch IT*.

Sadat, R., Koster, P., Mosmondor, M., Salvi, D., Girolami, M., Arnaudov, V., and Sala, P. (2013). Part III: The universAAL Reference Architecture for AAL. In Sadat, R., editor, *Universal Open Architecture and Platform for Ambient Assisted Living*. SINTEF.

Schmidt, A., Schmidt, A., Wolf, P., Wolf, P., Klein, M., Klein, M., Balfanz, D., and Balfanz, D. (2009). SOPRANO Ambient Middleware: Eine offene, flexible und markt-orientierte semantische Diensteplattform für Ambient Assisted Living. *2. Deutscher Kongress Ambient Assisted Living, Berlin, Januar 2009*.

Stutz, O., Todt, S., Venzke-Caprarese, S., Boll, S., Heuten, W., and Wallbaum, T. (2016). Implementing Data Protection and Information Security in AAL. pages 59–68. Springer, Cham.

The European Parliament and the Council of the European Union (2016). REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC.

The OSGi Alliance (2014). The OSGi Alliance OSGi Core, release 6. *OSGi Secification*, (June):161.

United Nations (2001). World Population Ageing: 1950-2050. Technical report, UN: Department of Economics and Social Affairs - Population Division.

van Heek, J., Himmel, S., and Ziefle, M. (2017). Privacy, data security, and the acceptance of AAL-systems – A user-specific perspective. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 10297 LNCS, pages 38–56. Springer, Cham.

Wang, T., Wei, J., Zhang, W., and Zhong, H. (2012). A framework for detecting anomalous services in OSGi-based applications. In *Proceedings - 2012 IEEE 9th International Conference on Services Computing, SCC 2012*, pages 250–257. IEEE.

Wolf, P., Schmidt, A., Otte, J. P., Klein, M., Rollwage, S., König-Ries, B., Dettborn, T., and Gabdulkhakova, A. (2010). openAAL - The Open Source Middleware for Ambient Assisted Living (AAL). *AALIANCE conference*, (March):1–5.