



Technische Universität München

Department of Informatics

Chair for Network Architectures and Services



User controlled privacy settings for Smart Environments

Bachelor's Thesis in Informatics

accomplished at

Chair for Network Architectures and Services

Department of Informatics

Technische Universität München

by

Besjan Saidi

July 2015



Technische Universität München

Department of Informatics

Chair for Network Architectures and Services



User controlled privacy settings for Smart Environments

—

User einstellbare Privacy settings für Smart Environments

Bachelors's Thesis

accomplished at
Chair for Network Architectures and Services
Department of Informatics
Technische Universität München

by

Besjan Saidi

Supervisor: Prof. Dr.-Ing. Georg Carle

Advisors: Dr. Holger Kinkel

Marcel von Maltitz, M.Sc.

Submission date: 06/15/2015

I confirm that this Bachelor's Thesis is my own work and I have documented all sources and material used.

Abstract:

Smart Environments play an increasingly important role in our modern global society. The services provided by these environments offer the opportunity to considerably improve the quality of life of the specific users. At the same time, during the last years there has been a significant increase of the awareness of the importance of privacy issues among users. Especially in a work environment the handling of privacy-related information collected and further processed in a Smart Environment can prove to be extremely precarious.

The aim of the present work is the implementation of a tool named PrivacyControl, that will offer users the possibility to carry out trade-offs between the services provided to them and their privacy in a Smart Environment. Furthermore some essential scientific questions about privacy in a Smart Environment are presented and answered in this work.

Furthermore it is defined, which of the data collected in a Smart Environment constitutes a violation of the users privacy, it is analyzed which possibility there is to offer a service in a Smart Environment requiring privacy-related data without restricting the privacy of the users. It also aims to determine what options are available to measure privacy in a Smart Environment. All these findings subsequently have an effect on the process of implementing PrivacyControl.

Kurzfassung:

Smart Environments nehmen in unserer heutigen Gesellschaft einen zunehmend hohen Stellenwert ein. Die durch diese Environments angebotenen Dienste bieten die Möglichkeit, die Lebensqualität der User beträchtlich zu verbessern. Gleichzeitig steigt unter den Usern auch das Bewusstsein für die Bedeutung der Privacy. Besonders in einem Arbeitsumfeld kann sich der Umgang mit privacy-relevanten Informationen, die in einem Smart Environment erhoben und weiterverarbeitet werden, als äußerst prekär erweisen.

Gegenstand der vorliegenden Arbeit ist die Implementierung eines Tools, PrivacyControl genannt, das dem User die Möglichkeit bieten soll, Trade-Offs zwischen den ihm angebotenen Diensten und seiner Privacy in einem Smart Environment durchzuführen. Des Weiteren werden in dieser Arbeit einige grundlegende wissenschaftliche Fragen zur Privacy in einem Smart Environment dargelegt und beantwortet. Dabei wird definiert, welche erhobenen Daten in einem Smart Environment die Privacy der Nutzer verletzen, es wird analysiert, wie Services in einem Smart Environment angeboten werden können, die privacy-relevante Daten benötigen, ohne dadurch die Privacy der User einzuschränken. Außerdem soll ermittelt werden, welche Möglichkeiten es gibt, um Privacy in einem Smart Environment zu messen. Alle diese Erkenntnisse kommen anschließend bei der Implementierung von PrivacyControl zum Tragen.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Environment	1
1.2.1	Implementation environment	2
1.2.2	Privacy-Preserving Post-Processing and Storage	2
1.3	Scientific questions	2
1.4	Outline	3
2	Background	5
2.1	Smart Environment	5
2.2	Privacy	5
2.2.1	Privacy in the smart environment?	6
2.2.2	Privacy in software design	7
2.3	Privacy-Preserving Post-Processing and Storage	8
3	Related Work	9
4	Analysis	11
4.1	Defining privacy goals in a smart environment	11
4.1.1	INFORM	11
4.1.2	CONTROL	11
4.1.3	Data collections violating user privacy	11
4.2	Protecting privacy goals without interfering with smart services	12
4.2.1	Services and Data consumers	13
4.2.2	Mandatory services	13
4.2.3	Optional services	14

4.2.4	Aggregation	14
4.2.5	Conflicting privacy goals	14
4.3	Requirement analysis	15
4.3.1	Functional requirements	15
4.3.2	Non functional requirements	16
5	Design	17
5.1	User preference collection	19
5.1.1	Front-end	19
5.1.2	Back end	20
5.2	Data consumer requirement collection	20
5.3	Conflict resolver	20
5.3.1	XML reader	20
5.3.2	Resolver	20
5.4	Graphbuilder	21
6	Implementation	23
6.1	Web fronted	23
6.1.1	Flow	23
6.2	Conflict resolver	24
6.2.1	XML reader	25
6.2.2	Resolver	25
6.3	Graphbuilder	27
7	Evaluation	29
7.1	Code validation	29
7.2	Privacy friendliness	29
7.2.1	CONTROL	29
7.2.2	INFORM	30
7.2.3	Uncertainty of Information	30
8	Interpretation	33
8.1	Privacy	33
8.1.1	CONTROL	33
8.1.2	INFORM	33
8.2	Data precision	34

9 Conclusion	35
9.1 Future Work	35
A Building and deployment	37
A.1 Building Privacy control	37
A.2 Deploy	37
A.2.1 Web front end	37
A.2.2 ConflictResolver	37
A.2.3 Graphbuilder	37
A.3 Running	37
A.3.1 Web frontend	37
A.3.2 Server (<i>graphbuilder</i>)	37
B Configuration	39
B.1 Graphbuilder configuration file	39
B.2 user preference xml file	39
B.3 data consumer xml file	39
B.4 mapping config file	40
List of Figures	43
List of Tables	45
Literature	47

1. Introduction

1.1 Motivation

Smart Environments are becoming an increasingly important topic in our global society. We have gained awareness of the need to change our consumer behaviour and economize on our resources, if we want to maintain the balance of our planet. Smart Environments and Energy Monitoring Systems on the one hand help us realizing such goals, but on the other hand they are often in conflict with the privacy goals of the modern user. This paper aims to empower the users of smart environments to make trade- offs between the services of such an environment and their privacy goals.

1.2 Environment

The environment this paper presupposes is a smart environment. According to Mark Weiser, (see [WeGB99])the definition of a smart environment is:

a physical world that is richly and invisibly interwoven with sensors, actuators, displays, and computational elements, embedded seamlessly in the everyday objects of our lives, and connected through a continuous network.

In such a smart environment multiple services can be implemented. These services use all the data provided to offer the users a set of tools, which, by presenting them a solution allowing them to monitor their consumption data in quite an easy way, helps to improve the lives of the consumers. But some of the data collected, especially when interpreted, can violate the privacy of individuals, in particular, if such a system is implemented in a work environment.

1.2.1 Implementation environment

Like stated above, the environment where a smart environment is implemented has a major influence on the privacy goals of the user. Depending on this implementational environment the privacy goals can be completely different. In a home environment the user has control over the system. This means that if a privacy violation occurs, it is in the user's responsibility to change the system configuration in order to match his privacy goals. On the other hand, if such a system is implemented in a work environment not only the privacy goals are much more strict than in an home environment, but, moreover, the user (mainly the employee) has no influence on a changing of the system configuration, and even if he could this, would affect the system's ability to provide its crucial services, representing the major benefits of a smart environment. Therefore this paper assumes that the smart environment is implemented in a work environment where no direct access is possible to the privacy settings of the smart environment. The scientific question and the implemented tool are solved in this regard.

1.2.2 Privacy-Preserving Post-Processing and Storage

The Privacy-Preserving Post-Processing and Storage (P4S) is a Server and Client based tool developed by Benedikt Peter, that can process storage and provide the data for the services and users in a smart environment in a secure and privacy friendly way. For more information about this system see the Background chapter 2.3, as well as [Pete15]. The implementation of the PrivacyControl is primarily based on needs of this system, but during the design phase the applicability on other systems as well, was a quite decisive requirement.

1.3 Scientific questions

While this paper primarily aims at the implementation of a tool that helps to empower the user to make trade-offs between his privacy and the services provided, some scientific questions have to be answered as well.

What are the privacy goals of the users in a smart environment?

The privacy goals in a smart environment can be manifold depending on the user scenario. This paper aims to give a precise answer to this question.

How can the privacy goals of the users be protected without interfering with the services of a smart environment?

When users get the possibility to restrict data for the services in a smart environment, they can interfere with the ability of the services to work. How to avoid this problem is a crucial part of this paper.

How can be privacy measured?

In the evaluation process of this work a special question arose: how can privacy be measured? In the evaluation chapter 7 this question will be answered.

1.4 Outline

Chapter 1: Introduction

Chapter 1 introduces the environment, the motivation. Further it enumerates the scientific questions that are addressed in this paper.

Chapter 2: Background

Chapter 2 gives some Background information to make the explanation clearer for the viewer.

Chapter 3: Related Work

Chapter 3 Gives a overview about other similar work.

Chapter 4: Analysis

Chapter 4 answers the scientific question introduced in Chapter 1 and analyses the requirements for the implementation of PrivacyControl

Chapter 5: Design

Chapter 5 explains how the system was designed and why certain design decisions were made.

Chapter 6: Implementation

The chapter 6 illustrates how the system was implemented in practice, which library was chosen and why.

Chapter 7: Evaluation

Chapter 7 evaluates the system implementation and whether the planned goals are met.

Chapter 8: Interpretation

In chapter 8 the results from chapter 7 are discussed and interpreted.

Chapter 9: Conclusion

Chapter 9 summarizes the findings in this paper and states possible future work.

2. Background

Some Background information is required to allow the exposition and explanation of the scientific questions, the motivation to build PrivacyControl and the decisions taken during the design and implementation phase of this paper.

2.1 Smart Environment

Definiton was smart environment usw. [GudS08]

A smart environment is a context sensitive system based on ubiquitous computing, in which the environment interacts with its inhabitants through embedded dedicated devices. The design and construction of a smart environment requires the collaboration among several areas, such as intelligent man-machine interfaces, pervasive communications, ambient intelligence, scalable systems and mobile computing systems and mobile computing.

2.2 Privacy

In general, privacy goals can be defined in many ways. In their article "Privacy Protection Goals in privacy and data protection evaluations" [Thom12] for the "Unabhängiges Landeszentrum für Datenschutz", Thomas Probst and Marit Hansen define privacy goals as unlinkability, transparency and intervenability.

"Unlinkability ensures that privacy-relevant data cannot be linked across privacy domains or used for a different purpose than originally intended."

"Transparency ensures that all privacy-relevant data processing including the legal, technical and organizational setting can be understood and reconstructed."

”Intervenability ensures that data subjects, operators and supervisory authorities can intervene in all privacy-relevant data processing.”

In turn Alan F. Westin in his book [West70] defines privacy as:

”privacy is the claim of individuals, groups, or institutions to determine for themselves when, how and to what extent information is communicated to others”

2.2.1 Privacy in the smart environment?

A more tangible definition in the domain of smart environments may be again a definition of Alan F. Westin, summarized in the table below:

Openness and transparency	No secret record keeping. minimal amount possible.
Individual participation	The subject should be able to see the records.
Collection limits	Record collection should be appropriate for the application
Data quality	Record collection should be accurate and relevant to the application.
Use Limits	Records should only be used for specified purposes and by only by authorised people.
Appropriate security	Reasonable efforts should be made to secure the records.
Accountability	Record keepers must be accountable.

Table 2.1: This table lists the principles of fair information practices stated by Westin [NWET04].

All these principles are in comparison with the first definition applicable to a Smart Environment. They are very similar to the privacy design strategies stated by Hoepman, listed in the paragraph below.

2.2.2 Privacy in software design

Another important part of this paper is the implementation of the PrivacyControl tool. For this purpose we have to look at some design criteria for privacy conform software.

MINIMISE	“The amount of personal data that is processed should be restricted to the minimal amount possible.”
HIDE	“Any personal data, and their interrelationships, should be hidden from plain view.”
SEPARATE	“Personal data should be processed in a distributed fashion, in separate compartments whenever possible.”
AGGREGATE	“Personal data should be processed at the highest level of aggregation and with the least possible detail in which it is (still) useful.”
INFORM	“Data subjects should be adequately informed whenever personal data is processed.”
CONTROL	“Data subjects should be provided agency over the processing of their personal data.”
ENFORCE	“A privacy policy compatible with legal requirements should be in place and should be enforced.”
DEMONSTRATE	“[...] A data controller [should] be able to demonstrate compliance with the privacy policy and any applicable legal requirements.”

Table 2.2: This table lists the eight strategies for designing privacy-friendly applications stated by Hoepman in [Hoep12].

The strategies stated by Hoepman are very useful in the designing and implementation part of PrivacyControl. Especially **INFORM** and **CONTROL** is a big part of this paper.

2.3 Privacy-Preserving Post-Processing and Storage

The Privacy-Preserving Post-Processing and Storage tool (short P4S), (see [Pete15]) helps smart environments to enhance, process, store, and provide data for the services in a privacy-friendly manner. It implements the Hoepman strategies MINIMIZE, HIDE, SEPARATE and AGGREGATE.

This server client system has been done like this work as a part of the IDEM project([idem]). Therefore it focuses more on energy consumption rather than general smart environment.

PrivacyControl is a configuration tool that helps P4S to implement INFORM, CONTROL.

3. Related Work

The purpose of this chapter is to give a brief overview about other works that discuss closely related concepts or problems. The works are briefly introduced and then a comparison between the subjects and the this paper is made.

In the article *"Privacy By Design und die Neuen Schutzziele"* (see [RoBo11]) the German authors Martin Rost and Kirsten Bock describe 6 new privacy goals that are in accordance with the precautionary measures for privacy of the German Bundesamt für Sicherheit in der Informationstechnik. These 6 new privacy goals are:

Transparenz	Transparency. No secret keeping
Intervenierbarkeit	Intervening. The subject should be able to to intervene and control the records.
Nichtverkettbarkeit Zweckbindung	Unlinkability. Record collection should be earmarked for the application.
Verbindlichkeit	Liability. Record keeper must be liable.
Anonymität	Anonymity. The users of the system should be concealed.
Zurechenbarkeit	Accountability. Record keepers must be accountable.

Table 3.1: This table list the new Privacy goals defined by Martin Rost, Kirsten Bock. [RoBo11].

As you can see these are very similar goals that are defined by Hoepman [Hoep12] and by Westin [West70]. That means that they are already incorporated in the design of PrivacyControl. The difference in those privacy goals consists in them being a bit more focused on data security, as you can see by the privacy goal Verdecktheit.

In the article *Privacy-Friendly Smart Environments* (see [APPR09]) the authors Ibrahim Armac, Andriy Panchenko, Marcel Pettau and Daniel Retkowitz describe methods on how to make smart environments more privacy-friendly. In contrast to this work their paper goes a more cryptographical approach on the matter. Thus the findings from this work are more suited for the privacy-friendly monitoring system.

Sebastian Vogl in his work "Mitigation of privacy issues in ontology-based knowledge bases" /citeSebastianVogl faces a very similar problem. The conflict solving between the data consumers requirements and the user preferences is very similar in his work.

4. Analysis

Since the purpose of this thesis is the implementation of PrivacyControl, a tool that empowers users to make trade-offs between their privacy and the provided services of a smart environment and to answer the scientific question enumerated in chapter 1. This chapter is divided in two parts.

The first part answers the scientific question and the second part analyses how to implement a tool to realize those findings.

4.1 Defining privacy goals in a smart environment

In general, privacy goals have a very broad definition. This chapter provides a description of the relevant goals for our system.

The primary privacy strategies relevant for this work and necessary to be implemented are the two privacy design strategies stated by[Hoe12] **INFORM** and **CONTROL**.

4.1.1 INFORM

The **INFORM** strategy is a process strategy implying that whenever a user avails himself of a system, he should be informed of which information about himself is being collected. Moreover, users need to be informed of the purpose of the collection and of how the data is collected. Another aspect of Inform is openness about the security in the system and about third parties the information is shared with.

4.1.2 CONTROL

The counterpart to the **INFORM** strategy is the **CONTROL** design strategy for privacy conform environments. Informing users of what data is collected without giving the possibility to control the collecting of the data is useless. Control is about giving the user the possibility to exert data protection rights.

4.1.3 Data collections violating user privacy

Since **CONTROL** operates on the collected data, it is important to state which physical values are collected in a smart environment.

4.1.3.1 Location

The location of a user in a smart environment may be the location where the energy has been consumed. For the user of a Smart Environment this may be one of the most important privacy goals. The consuming of energy in a specific location in most of the cases implicates that the user was physically present in that location. Together with other data this fact is even more concerning for the user.

4.1.3.2 Time

The time when a specific event occurred. This value can have a granularity range from accurate time designations in seconds to inaccurate information like weeks or months. Depending on the granularity this value can be more or less concerning for the user.

4.1.3.3 Identity

This value is the most privacy violating factor in a smart environment. It includes information about a person such as name, gender, place of residence, position, work group, department. Even without context this physical value can violate the privacy of a lot of users. It is without question that with enough context the identity of the user can be dangerous for the privacy goals of nearly every user.

4.1.3.4 Energy consumption

The energy consumption of a user in some smart environments is logged for the purpose of energy saving. However, this information can be a privacy concern for the users. Energy consumption information can expose much more than just energy consumption itself. It shows that the user was present in a specific location where energy was consumed. Further we can guess from the amount of consumed energy and the room where the consumption took place, what kind of devices the user was using.

4.1.3.5 History

This privacy goal defines the perish ability of the data. The system should be able to forget old data. In order to ensure this goal, it is necessary to guarantee that an easy export of the data is not possible, since, if the data can be exported, the history can be easily restored. In particular we have to ensure that there is no communication to a non-certified client.

4.2 Protecting privacy goals without interfering with smart services

The data gathered in smart environments is analysed by person and software. In this chapter we will define the services and the minimal information they need for the fulfilment of their tasks. A service or data consumer is:

4.2.1 Services and Data consumers

The data gathered in smart environments is analysed by person and software. In this chapter we will define the services and the minimal information they need to fulfil their tasks. A service or data consumer is any person or software that analyses the user data to provide some sort of benefit for the smart environment. Examples for such service are:

- **Free Room Finder** The free room finder is a service that can find free rooms to work in. This service is optional and needs location information about the user. It needs room-precise data to work optimal, but could work with floor-precise data, too.
- **Person Finder** The person finder is an optional service that allows the users to search other persons in a smart environment. This service needs location-based data. Room-precise data would be optimal, but building-precise data is enough to rudimentarily provide this service.
- **Energy Manager** The energy manager has the task to analyse the energy consumption of the smart environments and possibly find ways to economize on consumption. For his task fulfilment he needs to have all the data from every office in full granularity.
- **User** The user itself has access to all his information. This service is not privacy concerning. On the contrary, it helps to implement the INFORM design strategy.
- **Accountant** The accountant is responsible for the total amount of energy consumed. To fulfil his task he needs access to the overall consumption of a smart environment aggregated over a month.
- **Different Software** Depending on the software different physical values are required to fulfil this task. One example would be a public display, which could display the total amount of energy consumed in an office building. The public display would need to know the total amount of energy consumed in the building it is displaying. Another example would be a gamification service that allows the users of a smart environment to take place in a competition, where the main target is trying to be the most effective energy saver. Most of those services are optional.

4.2.2 Mandatory services

Mandatory services in smart environments have important tasks to fulfil. Without them the benefit of a smart environment would vanish. Examples are the Energy Manager that helps a company to detect where too much energy is wasted and where there is potential for savings. To ensure that those mandatory services can run properly, they need data with at least a certain amount of precision. If the user of those services sets his privacy settings below the minimal precision requirements, the user preferences have to be ignored.

4.2.3 Optional services

Optional services can be services that are convenient to have, but do not fulfil important tasks in a smart environment. An example for such a system would be the person-finder service. Optional services, like mandatory services, require a certain level of information precision. The difference between mandatory and optional services is that the user's priority has more importance than the service itself. If the user preference precision setting is below the minimum threshold of the service requirement the service for that user does not gather any information.

4.2.4 Aggregation

As we can see in Chapter 4.2.1, services can run with different degrees of precision. We can use this to our advantage and use aggregation to help the user control the precision of data provided to the service. Aggregation is the key mechanism in this paper and helps providing privacy for the user without interfering too much with services of a smart environment. Aggregating numerical values like energy consumption and time values is an easy-to-grasp concept, but non-numerical values collected about users can be aggregated as well, if enough information is available for the system to aggregate.

- Numerical aggregation: The numerical aggregation is a tool, which can be used on numerical values to fulfil the privacy goals of the user without relinquishing to data. It can be used with numerical values such as energy consumption and simply consists of the aggregation of information over a period of time.
- Structural aggregation: This is another tool at our disposal to fulfil the tasks of a modern smart environment without violating the privacy goals of the user. The structural aggregation, unlike the numerical aggregation, does not summarize values over a time period, but it aggregates over structures of the physical value. An example for a physical value, where structural aggregation can be used, is location. To do this aggregation the structure of the location is required. For example, to do an aggregation from the room to the floor, there has to be defined in which floor the room is located, to do a structural aggregation from the floor to the building the information where the floor is located needs to be defined. The advantage of this aggregation method is that it can be used for all the other values where numerical aggregation cannot be used.

4.2.5 Conflicting privacy goals

Even with the use of aggregation techniques there can be conflicts between the privacy preferences of the user and the service.

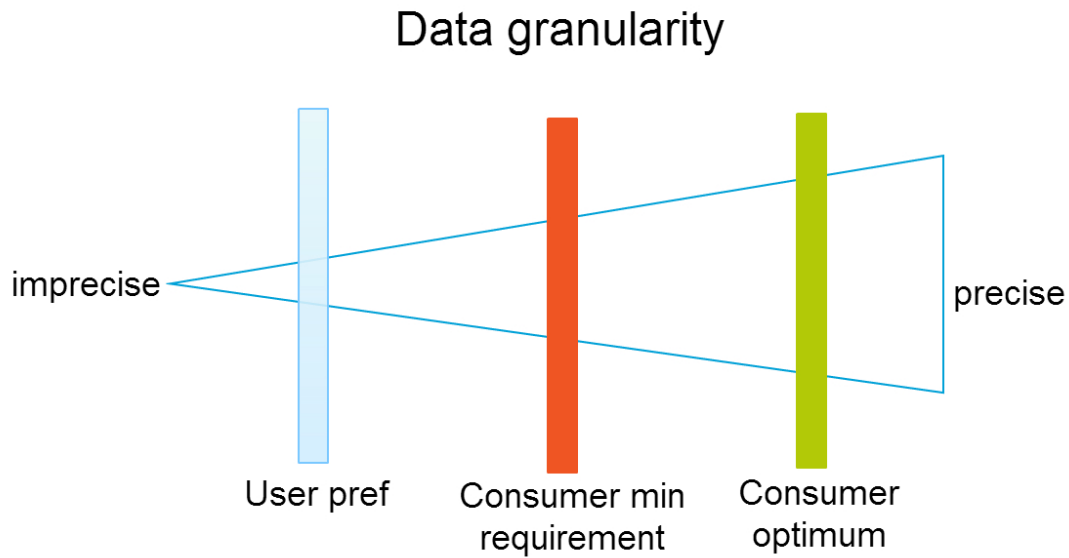


Figure 4.1: An example: a conflict between a user privacy preference and a service requirement. The user demands data that is less precise than the minimal precision for the service to work

If the user would have complete control over the smart environment, such a conflict between the requirements and the preferences of a user can have a big impact on the functionality of the services. Some mandatory services could not work properly. To avoid this, mandatory services in the smart environment have to be defined. Further the precision of data for those services has to be always above the minimum requirement, even when the user preferences are set differently.

4.3 Requirement analysis

Besides the requirements that follow from the first part of the analysis, there are some functional and non-functional requirements which need to be defined for the implementation and the design of the PrivacyControl.

4.3.1 Functional requirements

4.3.1.1 Access

The user should be able to access PrivacyControl from any device by entering a password and user name. No installation should be required. The user preference settings should be clear and understandable to anyone.

4.3.1.2 Conflict resolving

The conflict resolving should enforce user preference as much as possible without affecting the ability of mandatory services to do their job. The output format of the PrivacyControl should be a GraphML file that is compliant with the P4S specification for the Flowgraph. Further the user preferences and data consumer requirements should be met as best as possible.

4.3.2 Non functional requirements

PrivacyControl should be able to generate a GraphML file for the P4S system and be deployable for other Smart environments. Therefore Flexibility is a priority.

5. Design

In the last chapter 4.3 functional and non-functional requirements were defined. This chapter discusses how those requirements can be designed into one concept.

Further it defines the modules that are needed to implement PrivacyControl and how those modules work together.

Since the target of this implementation is to build a software that enables the user to have influence on the collected data for different smart environments that can be seen in figure 5.1, as well as to create a compliant GraphML file, as specified in [Pete15], that fulfils the user preferences and the data consumer requirements. For this purpose it was decided to build three modules.

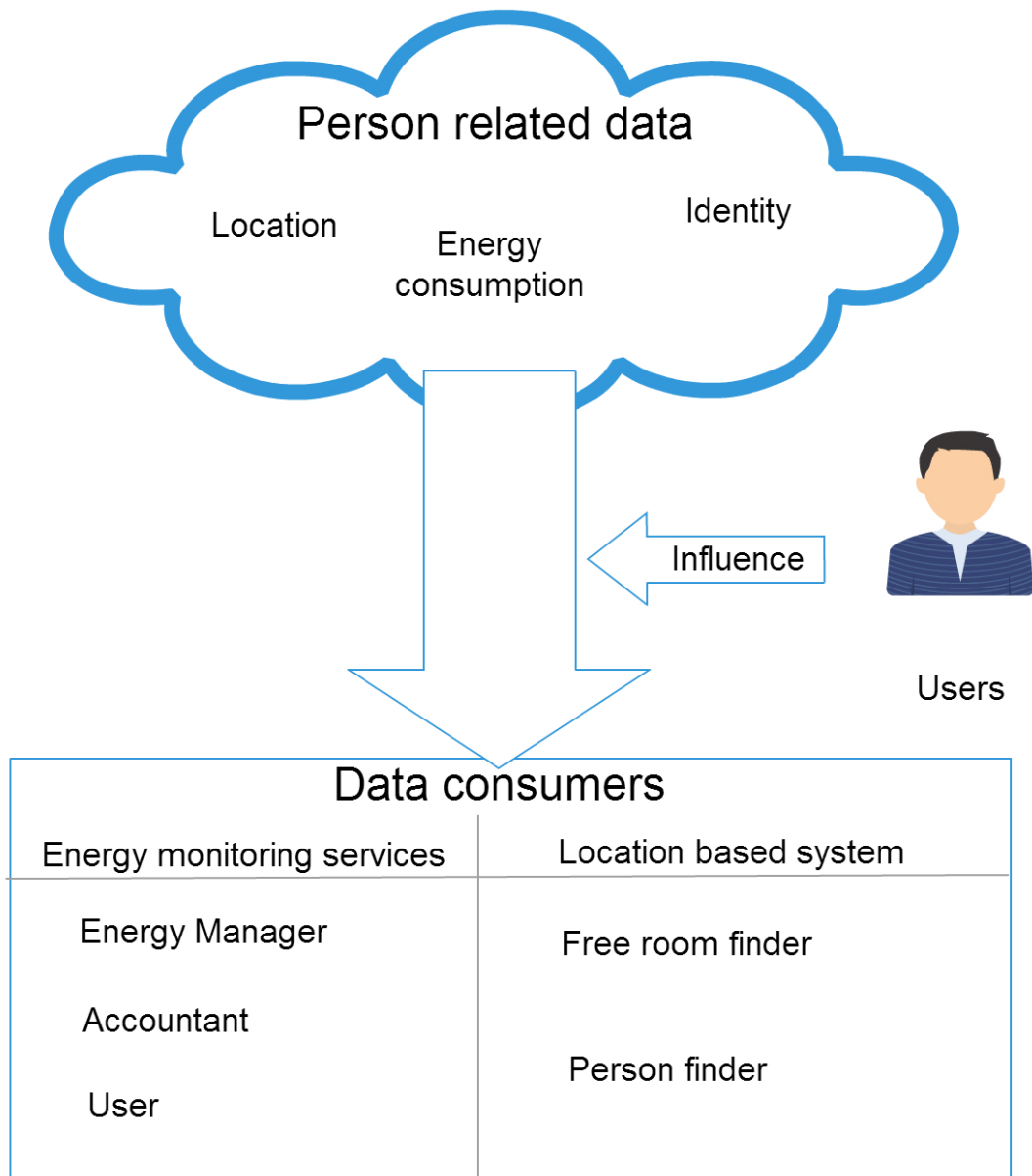


Figure 5.1: A smart environment with the user influence on data

The first module is the preference collection module. This module has the purpose to collect the user preference regarding which data with how much granularity has to be collected and to save them to the disk for further processing.

The second module is the conflict resolver. The conflict resolver module is the core module of PrivacyControl, it processes the user preference and the data consumer requirements and solves the conflicts between those data points.

The third module is the Graphbuilder module. This module is specially built for the P4S system. It implements a filter from the data collected and processed by the preference collection module and the conflict resolver module. To get a visual reference see figure 5.2

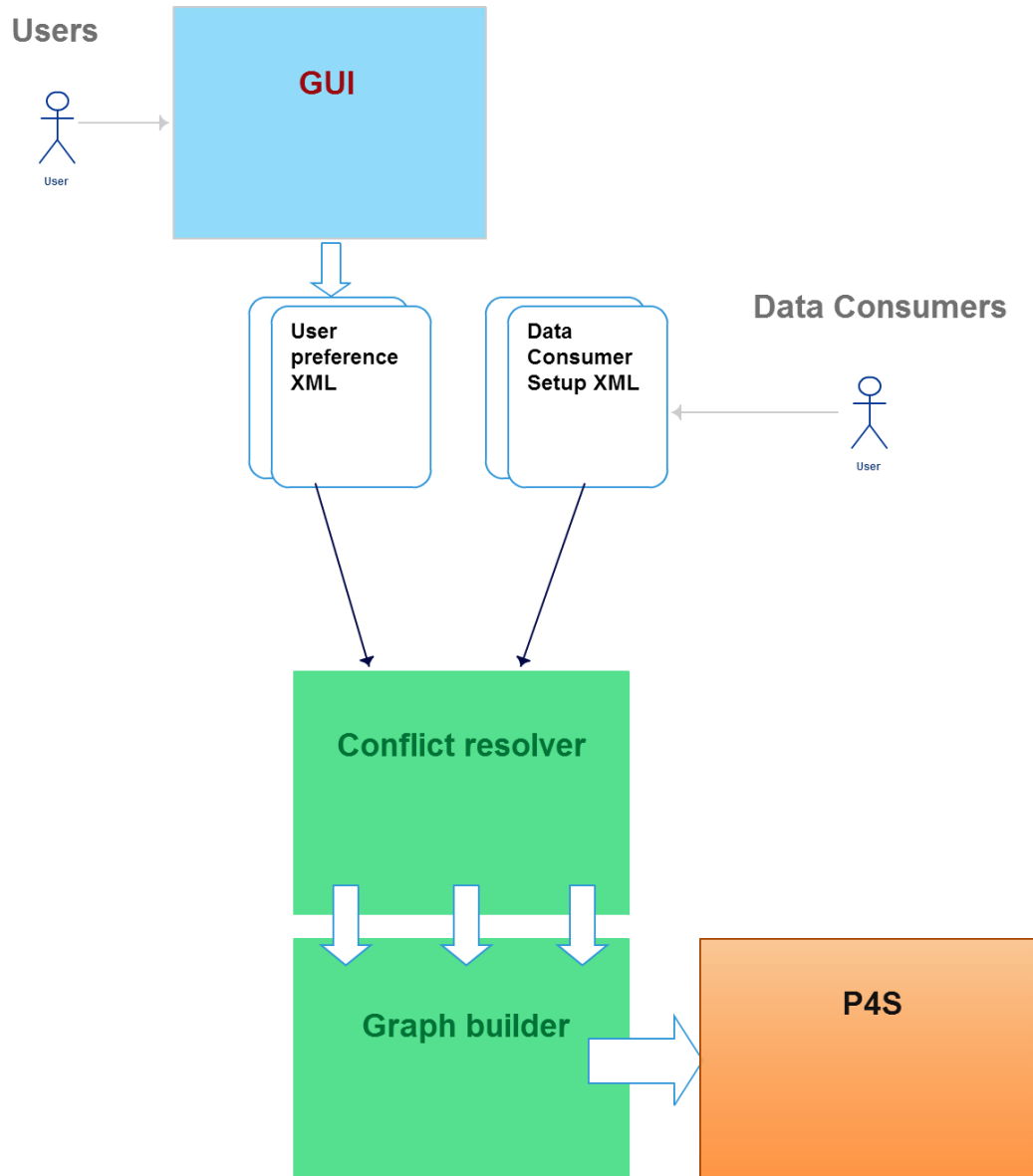


Figure 5.2: This is the system overview over all components in the system.

5.1 User preference collection

The user data collection is the only contact point for the user to control the smart environment. Therefore this module has to be accessible to different devices with different operating systems. Further no installation should be required. A website offers such criteria, thus presenting a viable user-friendly solution.

5.1.1 Front-end

The front-end that the user comes in contact with is implemented in HTML with the Bootstrap framework, JavaScript and Thymeleaf. The Bootstrap framework is

used to implement a dynamic website design depending on the screen size. This framework has the advantage that readability and design are retained, regardless of the screen sizes and resolutions the user's device uses. Thymeleaf is used to communicate with the website back-end. Thymeleaf has the advantage that it is easy to integrate in the Spring MVC which is used in the back-end and responsible for the processing of the data.

5.1.2 Back end

The back-end of the user preference collector is a Java Maven project with Spring MVC. MVC stands for Model View Controller. Model is a data container that contains data from the application. It helps transport the data from the view (the website) to the controller (the data processing unit and flow controller). View is the GUI, in our case the website. Controller is a Singleton that processes the data collected from the View and does the flow control.

5.2 Data consumer requirement collection

The data consumer requirement collection is done manually. This design choice was made for security reasons, since a web service creating such files can be a threat to the privacy of all users. By setting every service on mandatory and creating new services, an attacker could gather any dataset of the user he is interested in. Whereas if a person is responsible for creating those files manually there is no such threat. The system administrator has to create the requirement files for every data consumer. He has to specify if a service is mandatory and the optimal and minimal data precision requirements for those files. For the exact file configuration see chapter B.3.

5.3 Conflict resolver

The conflict resolver module is a Maven project. Maven was chosen for easy building and deployment. The module is divided into 2 sub modules: the conflict resolver itself and the XML reader. The conflict resolver can be used by different smart environment as an API in order to build control filters.

5.3.1 XML reader

The XML reader needs the user preference files and the data consumer requirement files as input. Starting from those files it generates a list with all the data consumers and their requirements and a list with all the users and their preferences.

5.3.2 Resolver

The conflict resolver gets the user preferences and the data consumer requirements from the XML reader. Then it processes the data and solves the conflicts, depending if a service is mandatory or optional in different ways.

- For Mandatory services if the user preference granularity is below the data consumer minimum requirement the user preference is ignored. For every other case it is implemented.
- For Optional services if the user preference is below the data consumer minimum requirement the data consumer preference is considered and the data consumer gets no data.

5.4 Graphbuilder

The Graph builder has the task to build a flow graph from the data generated by the conflict resolver. This flow graph is saved on to the disk in GraphML. GraphML is a file format for graphs. GraphML is based on XML. It was chosen because the P4S system can use such a flow graph, let its data "flow" through this graph and filter the data from the source to the actual database exit. GraphML has multiple advantages. It can represent directed and undirected graphs. It can further have application-specific attributes, can reference to external data and can be parsed easily.

6. Implementation

In contrast to the previous chapter, the main focus of this chapter is not on the design of the software but on the actual implementation of the system. This system can be divided in 3 parts. The first part is the web front-end which lets the user log in and enter his privacy preferences for the different privacy settings of the system. The second part is the conflict resolver. This part is the most important part of this work. In this chapter the algorithm for solving the privacy conflicts is explained in detail. The third part of the implementation is the Graphbuilder which builds a graph in Graph ML from the output of the conflict resolver. This graph is specifically destined for the P4S system. [Pete15]

6.1 Web fronted

The web front-end has the task to let the user log in and enter his privacy preferences. Like mentioned in the previous chapter, a web front-end is chosen to make this part of the project accessible to numerous devices. With this kind of architecture the user can access his privacy preferences from any device equipped with a browser and Internet connection. The website is implemented using Bootstrap as the front-end framework. For the back-end of the website, Spring is used to evaluate the preferences and save them so that the conflict resolver can process them.

6.1.1 Flow

The flow of the website is controlled by the controllers. This Singleton Object has the purpose to process the Get and Post request and to manage the flow of data.

6.1.1.1 Homepage

On the homepage of the user preference collector the user should have the possibility to log in with a user name and a password.

6.1.1.2 Preference settings

After the user is logged in he gets forwarded to the preference setting website. Here the user can set up his preferences for every person-related data category. He can toggle on and off every person-related data category and choose the granularity.

6.1.1.3 Results

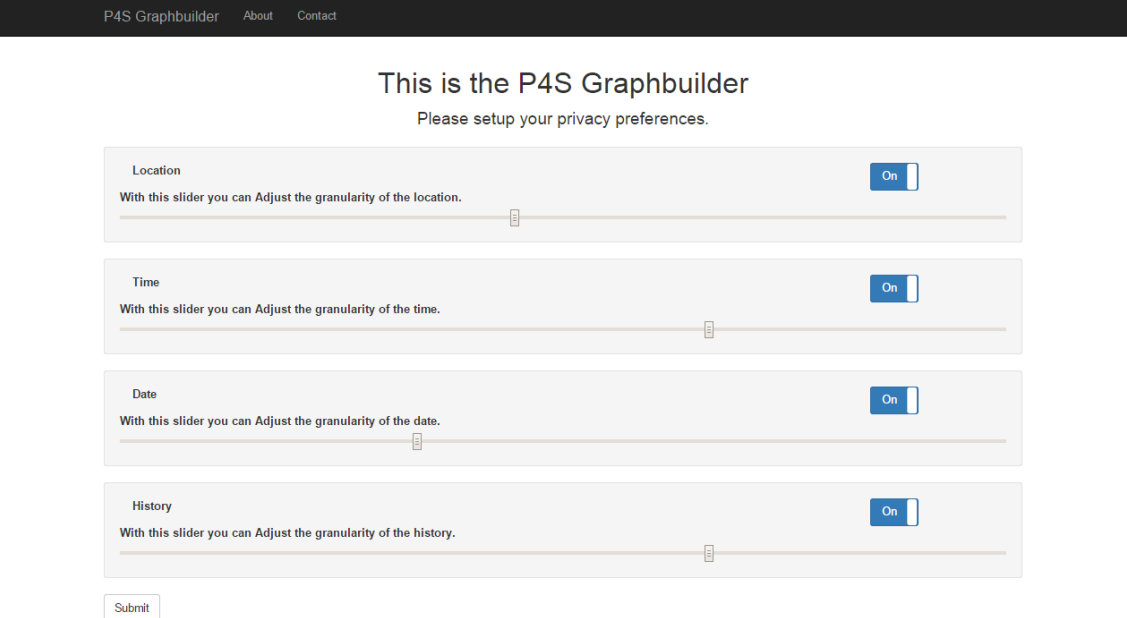
After the user submits his settings from the preference settings website he gets forwarded to the results website. Here he can see and control all his settings, go back to the preference controller or submit the data. When the user submits his data the responsible controller method writes the preferences into a XML file on the disk.

6.1.1.4 Submit

After submitting his preferences the user can see what he submitted. He cannot go back and resubmit another preference when he sees this website.

6.1.1.5 Actual Configuration

After the user has set his privacy preferences and if the data consumer configuration is available, the user can see the actual implementation of his preferences. This website implements a part of the INFORM design strategy defined by Hoepman.



The screenshot shows a web interface for 'P4S Graphbuilder'. At the top, there is a navigation bar with links for 'P4S Graphbuilder', 'About', and 'Contact'. The main heading reads 'This is the P4S Graphbuilder' followed by the instruction 'Please setup your privacy preferences.' Below this, there are four distinct preference sections, each with a title, a descriptive sentence, a horizontal slider, and a toggle switch labeled 'On'. The sections are: 'Location' (Adjust the granularity of the location), 'Time' (Adjust the granularity of the time), 'Date' (Adjust the granularity of the date), and 'History' (Adjust the granularity of the history). At the bottom left of the form area is a 'Submit' button.

Figure 6.1: Here the user can set up his preferences for every person-related data category. He can toggle on and off every person-related data category and choose the granularity.

6.2 Conflict resolver

The conflict resolver is the most important part of this implementation. It collects the data generated by the web front-end and the data consumer requirements and solves the conflicts between them. This module can be subdivided into two other modules. The XML reader which reads the files and generates easy-to-interpret Java objects and the actual Conflict resolver.

6.2.1 XML reader

The XML reader needs the file path to the folders where the user preference setting file and the data consumer requirements are stored. Then the XML reader can load those files and parse them accordantly.

6.2.1.1 Usage:

Even when the actual resolver takes care of the instantiation of the XML reader and of the call of the other methods, it is important for the understanding of the module to state briefly the general usage. First the path to the data consumer folder and user preference has to be specified. The path to the data consumer requirements folder should not contain any blank space. Further the folder should only contain user preference files. The same applies to the user preference folder path. To set the data consumer requirements path this method should be called:

```
public void setPathDataConsumerXml(String pathDataConsumerXml)
```

For setting the user preference path this method is responsible

```
public void setPathUsersXml(String pathUsersXml)
```

After those paths are set the XML reader can load and parse the XML files and return the required objects. To get the user preference call:

```
public List<User> getUsersPref()
```

For the data consumer requirements call:

```
public List<DataConsumer> getDataConsumerReq()
```

6.2.2 Resolver

The conflict resolver instantiates the XML reader with the path given to him and solves the conflicts between the user preferences and the data consumer requirements. For every combination of user data consumer preference the algorithm you can see in figure 6.2 is performed. First it is checked, whether the physical value i.e. location, identity etc. is mandatory for the data consumer and, if that is the case, it is checked, whether it is the maximum value between the minimum data precision and the user preference. If the physical value is not required, it is checked whether the user enabled that value collection. If that is true, it is checked whether the user preference is bigger than the optimal precision needed for that service. If that is true, the optimal precision is used. If not, it is checked whether the user preference is bigger than the data consumer minimal requirement. If that is true, the user preference is used.

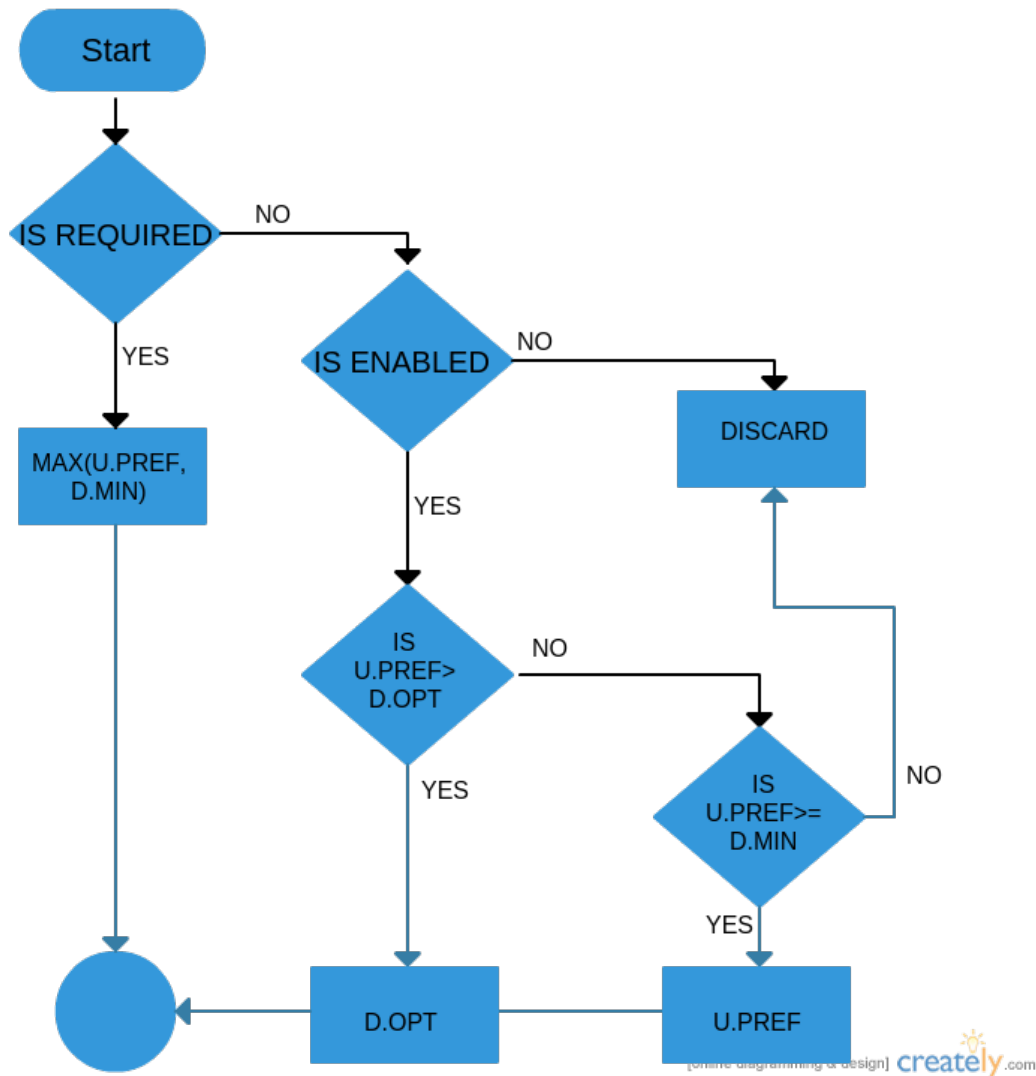


Figure 6.2: Here you can see the algorithm for solving the conflicts between the user and the services, so-called data consumers in the conflict resolver module

6.2.2.1 Usage

To use the conflict resolver a ConflictResolver object needs to be initiated. After that, the user preference path has to be set with:

```
public void setUserFilepath(String userFilepath)
```

Likewise the data consumer preference has to be set with:

```
public void setDataconsumerFilepath(String dataconsumerFilepath)
```

After this step the ConflictResolver is ready to be used and the conflict resolving described above can be started with:

```
public List<UserDataConsumerPref> resolve()
```

6.3 Graphbuilder

The Graphbuilder uses the data from the conflict resolver to build a flow graph, as can be seen in figure 6.3. This graph is built to the specification of the P4S system and is then saved on the disk.

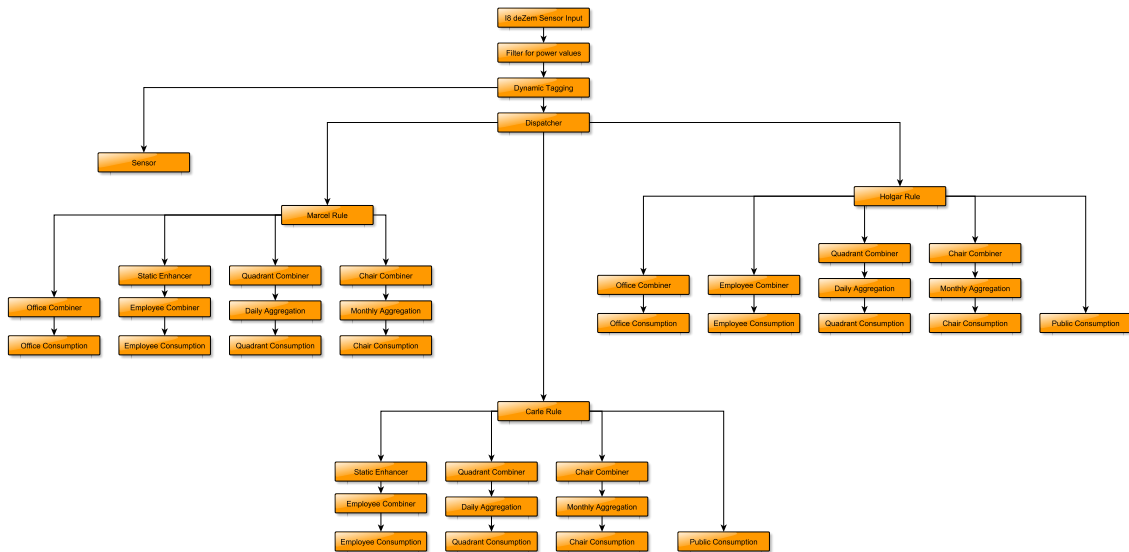


Figure 6.3: This is how a graph made for the P4S system looks like. Its main function is to filter the data for every user.

7. Evaluation

7.1 Code validation

The code validation in this work is done by Junit test. All normal and special cases are tested through those test. The Web front end is not tested with Junit tests since this is not an adequate testing method for this kind of architecture. The validation of the Web front-end is done by manual testing through the Websites.

7.2 Privacy friendliness

Since the purpose of PrivacyControl was to implement, **CONTROL** and **INFORM**, two of the 8 privacy design strategies defined in [Hoep12] by Hoepman we have to evaluated how they are implemented.

7.2.1 CONTROL

To evaluate how much control PrivacyControl has given the user we have to define a way to measure control. Depending on the service type (mandatory or optional) we have different results for measuring how much control PrivacyControl has given the user.

Before we define the metric how to measure how much more control the user has we have to make a assumption. The user preference is uniformly distributed over the granularity range. This assumption is pretty plausible since every user has different privacy perception and therefore different privacy goals.

7.2.1.1 Obligatory services

For obligatory services the control gain is depended on the service data requirement. The user can control the aggregation from no aggregation at all until minimal data consumer requirement. That means that the user has not total control over the system. To evaluate how much control PrivacyControl provides to the user for mandatory services basic this formula can be used:

- User control in percent: c .
- Minimal service requirement granularity(worst precision service can run): r .
- Maximal granularity(best precision overall): m
- Then the the formula for the user control is:

$$c = \left| \frac{r}{m} - 1 \right|$$

This formula assumes, like mentioned above that the user preference is uniformly distributed. It is a good indication for how much more control the all user together gained for this service with PrivacyControl. A one would mean the users have total control whereas a 0 indicates no control was gained.

7.2.1.2 Optional services

For optional services the user has total control over the service. It can set the precision on which data can be collected. But if the data precision is set below the minimal requirement of the service the service can not be provided.

7.2.2 INFORM

PrivacyControl helps the user view what services require his data in what aggregation. Further the user can see what his what services are required and which are not required. But there is a lot more to it to properly inform the user about the smart environment then just those information.

7.2.3 Uncertainty of Information

An other evaluation method for the user privacy gain is how much less information the data contains through aggregation. A metric for measuring how much information is in data is entropy. Entropy is the value of uncertainty or unpredictability information has. For more information on entropy see "A Mathematical Theory of Communication" by C. E. SHANNON [Shan01]. An example of such an entropy for our case would be the location of a certain user. If for example in a 3 floor building with 1500 rooms the location information of a user is given room precise the predictability of that information without any other information and assuming it is equiprobable that the user is located in any room, is:

$$p = \frac{1}{1500}$$

If we would know aggregate that information to floor precision the uncertainty would become:

$$p = \frac{1}{3}$$

Shannon's entropy formula can tell us how valuable the information is by giving us the uncertainty of that information in bits.

$$H(X) = - \sum_{x \in X} p(x) \log_2(p(x))$$

The room precise data has in our case has a entropy of

$$H(X) = -\log_2\left(\frac{1}{1500}\right) = 10.54bit \approx 11bit$$

bits, whereas the floor precise data has a entropy of:

$$H(X) = -\log_2\left(\frac{1}{3}\right) = 1.58bit \approx 2bit$$

bits.

Meaning that the room location data is approximately 512 times more precise than the floor related data. Entropy can be used on every value collected in the smart environment where probabilities can be associated to the value. Such an example would be the the energy consumption. Since a power supply line has a maximum current capability you could associate to every watt a probability and calculate the entropy of this data.

The entropy of multiple different values can be linked then together to evaluate how much information a service uses to provides his data.

8. Interpretation

The interpretation of the result can be divided into two parts: the interpretation of the privacy of the user and the interpretation of the data precision in the smart environment.

8.1 Privacy

To interpret the results gathered in the evaluation in chapter 7 of this paper we have to look at how the two design strategies for privacy defined by Hoepman, INFORM and CONTROL, are implemented.

8.1.1 CONTROL

How good is it possible to enable the user to control what data is collected, is mainly dependent on the data consumers requirement configuration. This configuration is done manually. Therefore the person doing this configuration should be independent. This person should neither be stakeholder of the user's interests, nor of the the smart environment's operators. Further the configurator should have good understanding of the services in the smart environment. Since the user can evaluate how good his control mechanism works through the web front-end, a bad data consumer requirement set-up is easily identified. Still, trust between the user and the operator of the smart environment should exist beforehand.

8.1.2 INFORM

The Inform design strategy is implemented in the web front-end, too. After the user entered his preferences and the data consumer requirements have been entered, the user can see how his preferences are implemented and what data is collected. Trust is a key issue here. This issue can be solved through the implementation of the DEMONSTRATE design strategy by Hoepman.

8.2 Data precision

To interpret how the data precision in a smart environment changes with the usage of privacy control we have to assume how the smart environment was before utilizing PrivacyControl.

A privacy-conservative smart environment is configured with only the basic mandatory services. Further those services have access to the data with the minimum precision they need to provide their services. Whereas a privacy-violating smart environment is configured with all the available services and those services get the data in optimal precision to provide the best service possible.

Since the PrivacyControl gives the users the possibility to control the data precision on the mandatory services from optimal precision to minimal required precision the data precision for a conservative privacy configuration, where every service beforehand runs with minimal requirements, can only increase. Further optional services can now be provided without violating the user's privacy, since the user has full control over those.

In a contrary configuration i.e. a privacy-violating smart environment as described above, the data precision will decline or no more data will be provided for certain services, since some services have to be made optional and other mandatory services may have to run on minimal requirements for certain users.

9. Conclusion

As a conclusion we can say that smart environments can be made much more privacy-friendly by using PrivacyControl and implementing a filter from the data derived from the conflict resolver, like the P4S system. The user can gain a lot of control over the smart environment and gets an insight into what data is collected and why. However, even though the Hoepman strategies INFORM and CONTROL were implemented, PrivacyControl could not provide full control over the users' privacy. Apart from the fact that the mandatory services cannot always be fully controlled, the main reason why this cannot be guaranteed is the manual setting of the data consumer requirements. An automatic data consumer requirement setting by software would eliminate this problem by configuring an privacy optimal setting for the data consumers.

The INFORM part of the strategy helps to inform the user quite efficiently on what data is being collected and why. However, trust is an important factor here, too. The user has to rely in the correctness of the shown information. However, the Hoepman design strategy DEMONSTRATE would eliminate that and could be part of further research.

9.1 Future Work

Future Work should try to eradicate the human component in the set-up process of a smart environment. To achieve this automatic configuration of service should be implemented with optimal settings for the user's privacy. Further the Hoepman design component DEMONSTRATE, together with the design strategy INFORM, could be implemented to eliminate the need for human trust in a smart environment.

A. Building and deployment

A.1 Building Privacy control

All components of PrivacyControl use Maven as a build management tool. So to Build the project just execute Maven.

A.2 Deploy

A.2.1 Web front end

To deploy the web front-end we just need to build the JAR with Maven. The JAR contains the web server and everything else that is needed to run. Launching the JAR starts the web server and the website is ready to run.

A.2.2 ConflictResolver

Since the conflict resolver is meant to be used as a API there is no reason to deploy this module on its own.

A.2.3 Graphbuilder

To deploy the Graphbuilder just execute maven.

A.3 Running

A.3.1 Web frontend

To execute the web server deploy the application and execute the jar file.

A.3.2 Server (*graphbuilder*)

In order to run the Graphbuilder application, a valid set of configuration files has to be provided using the command-line option.

```
usage: graphbuilder path_user_preference_folder path_data_consumer_requirement_folder  
It is important that the path to the folders and the file contains no blanks.
```


B. Configuration

B.1 Graphbuilder configuration file

To map the values the conflict resolver generated for the granularity of the different metrics to actual real-life values for example location "3-floor" we need a configuration file. The format of this configuration file is defined here.

B.2 user preference xml file

The user preference file that defines the preferences of one user is a XML file. The conflict resolver needs it as input and it has to have this format. The root tag is <pref>. Within <pref> there has to be an <ident> tag with the identity of the user and there can be any amount bigger than 1 of <preference> tags. The <preference> tag has two attributes: type and enabled. The <type> defines the preference type. This can be set like the physical value is named. The enable attribute determines if the physical value collection should be allowed or not. This can be either true or false. Inside of the preference tag the granularity is defined. It is important that the file has exactly one ident tag. The amount of preferences is not important, but at least one preference should exist. Here you can see an example user preference file:

```
<?xml version="1.0" encoding="UTF-8"?>
<pref>
  <ident>identity</ident>
  <preference type="location" enabled="true">4</preference>
  <preference type="time" enabled="false">3</preference>
  <preference type="date" enabled="true">3</preference>
  <preference type="history" enabled="true">3</preference>
</pref>
```

B.3 data consumer xml file

The data consumer requirement file is a XML file with the following characteristics. The root tag of the file is the <consumer> tag. Inside of the <consumer> tag

there has to be a <ident> tag with the identity of the data consumer inside the tag. Further there can be one or more <demand> tags. The demand tag has two attributes: the type attribute which describes the type of the physical value and the required attribute that describes whether the physical value is mandatory or not. This can be set to true or false. Between the <demand> tag there has to be a <min> tag and a <opt> tag. The <min> describes the minimal data precision required and the <opt> tag the optimal precision. These files have to be set for every data consumer from the administrator of the smart environment. Here you can see an example file:

```
<?xml version="1.0" encoding="UTF-8"?>
<consumer>
  <ident>energy_manager</ident>
  <demand type="location" required='true'>
    <min>3</min>
    <opt>5</opt>
  </demand>
  <demand type="time" required='false'>
    <min>2</min>
    <opt>8</opt>
  </demand>
  <demand type="date" required='true'>
    <min>3</min>
    <opt>5</opt>
  </demand>
  <demand type="history" required='true'>
    <min>1</min>
    <opt>9</opt>
  </demand>
</consumer>
```

B.4 mapping config file

The mapping file defines the mapping between the granularity values in the user preference and the data consumer requirement files to physical values. The root tag is the <mapping> tag. Inside the root tag there needs to be a <map> tag with three attributes. The type attribute defines the type of the physical value like type in user preference file and data consumer requirement file. The min max attributes define the min and max values for the aggregation. They can be set from 0 to Integer.MAX VALUE. Min should be smaller than max. The <level> tag which is also inside the <mapping> tag has two attributes: the name attribute which is the physical value of aggregation and the number attribute which is the according number. Below you can see an example.

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping>
  <map type="location" min="0" max="10">
    <level name=floor number=0/>
```

```
        <level name=room number=10/>
</map>
<map type="energy" min="0" max="10">
    <level name=seconds number=10/>
    <level name=daily number=7/>
    <level name=monthly number=0/>
</map>
<map type="identity" min="0" max="10">
    <level name=name number=10/>
    <level name=gender number=7/>
    <level name=position number=0/>
</map>
<map type="history" min="0" max="10">
    <level name=name number=10/>
    <level name=year number=7/>
    <level name=month number=0/>
</map>
</mapping>
```


List of Figures

4.1	Conflict user preference and service requirement	15
5.1	Design Smart Environment	18
5.2	Overview of the system	19
6.1	Web front end	24
6.2	Conflict resolving algorithm	26
6.3	Flowgraph	27

List of Tables

2.1	Principles of fair information defined by Westin	6
2.2	Privacy design strategies defined by Hoepman	7
3.1	Die neuen Schutzziele definieren by Martin Rost and Kirsten Bock . .	9

Literature

- [APPR09] Ibrahim Armac, Andriy Panchenko, Marcel Pettau and Daniel Retkowitz. Privacy-Friendly Smart Environments. In Khalid Al-Begain (Hrsg.), *Third International Conference and Exhibition on Next Generation Mobile Applications, Services and Technologies (NGMAST 2009)*. IEEE Computer Society, 2009, S. 425–431.
- [GudS08] Crhistian Alberto Noriega Guerra and Flavio Soares Correa da Silva. A middleware for smart environments. In *AISB 2008 Convention Communication, Interaction and Social Intelligence*, Band 1. Citeseer, 2008, S. 22.
- [Hoep12] Jaap-Henk Hoepman. Privacy Design Strategies. *CoRR*, 2012.
- [idem] The IDEM project. <http://idem-project.de/>.
- [NWET04] P A Nixon, W. Wagealla, C. English and S. Terzis. Security, Privacy and Trust Issues in Smart Environments, 2004.
- [Pete15] Benedikt Peter. Privacy-Friendly Energy Monitoring, Technische Universität München, 2015.
- [RoBo11] Martin Rost and Kirsten Bock. Privacy By Design und die Neuen Schutzziele - Grundsätze, Ziele und Anforderungen. *Datenschutz und Datensicherheit* 35(1), 2011, S. 30–35.
- [Shan01] C. E. Shannon. A Mathematical Theory of Communication. *SIGMOBILE Mob. Comput. Commun. Rev.* 5(1), Januar 2001, S. 3–55.
- [Thom12] Marit Hansen Thomas Probst. Privacy Protection Goals in privacy and data protection evaluations, 2012.
- [WeGB99] M. Weiser, R. Gold and J. S. Brown. The Origins of Ubiquitous Computing Research at PARC in the Late 1980s. *IBM Syst. J.* 38(4), Dezember 1999, S. 693–696.
- [West70] A.F. Westin. *Privacy and Freedom*. Bodley Head. 1970.