Technische Universität München
Department of Informatics

Bachelor's Thesis in Informatics

A system for giving practical advice for energy savings based on sensor information

Lawrence H. Krug


Technische Universität München

Department of Informatics

Bachelor's Thesis in Informatics

A system for giving practical advice for energy savings based on sensor information

Ein auf Sensordaten basierendes System zur Energiesparung

Author	Lawrence H. Krug
Supervisor	Prof. Dr.-Ing. Georg Carle
Advisor	Dr. Holger Kinkelin, Marcel von Maltitz, M. Sc.
Date	July 15, 2015

Informatik VIII
Chair for Network Architectures and Services

I confirm that this thesis is my own work and I have documented all sources and material used.

Garching b. München, July 15, 2015

Signature

Acknowledgements

First of all, I would like to thank Prof. Dr.-Ing. Georg Carle and all the members of the chair for the chance to to write my thesis there. Especially I would like to thank my advisors, Dr. Holger Kinkelin and Marcel von Maltitz, for their support, input and criticism.

Additionally, I would like to thank my father, for his support when writing this thesis.

Abstract

As smart buildings become increasingly popular, detailed energy monitoring is available on a broader scheme. An advice system, providing the inhabitants with feedback on their energy consumption, can save up to 15 percent in energy consumption. To provide users with reasonable feedback, different types of energy wastage have to be detected.

This thesis develops three algorithms detecting energy wastage. These algorithms are then implemented prototypically. Additionally as means of communication with the user, the advice system provides a web interface, and is integrated into a communication infrastructure for smart buildings.

Zusammenfassung

Durch die Einführung von intelligenten Gebäuden ist eine sehr genaue Erfassung der Stromverbrauchsdaten möglich geworden. Wenn die Bewohner eines Gebäudes Feedback zu ihrem Verbrauchsverhalten erhalten, sind Einsparungen von bis zu 15% möglichen. Damit die Bewohner qualitativ hochwertiges Feedback erhalten können, müssen Ineffizienzen in ihrem Verhalten entdeckt werden.

Diese Arbeit entwickelt drei Algorithmen, die Ineffizienzen entdecken. Diese Algorithmen werden dann prototypisch in einem Empfehlungssystem implementiert. Das Empfehlungssystem erhält eine Web-oberfläche und wird außerdem in eine Kommunikationsinfrastruktur für intelligente Gebäude eingebunden.

Contents

List of Figures

List of Tables

Code Listings

Chapter 1

Introduction

During the last decade “smartness” has been added to a broad variation of devices, the most popular example being smartphones. This development however, is not restricted to mobile devices, the term “smart” has been added to buildings as well. It means the capability exists to measure various variables, such as heating, ventilation, air conditioning, etc.

Most importantly, electrical energy consumption data can be collected and evaluated in a way that allows to direct users towards energy savings. This evolution is encouraged even further by smart meters advancing into people’s homes. These devices measure electricity consumption more precise and detailed than ever before.

1.1 Motivation

Given this whole new level of detailed information related to power consumption, it should be possible to realize additional energy saving. However simply presenting users with energy consumption data does not guide them towards appropriate actions to save energy.

A system, detecting inefficient behavior, would leverage the energy saving potential smart buildings provide. If, for instance each room were to be equipped with a brightness sensor, the user would be able to receive a push notification to a mobile device, letting him know it is bright enough to switch the lights off.

Apart from letting the user know, where energy is wasted, information on the amount of wasted energy should be given as well. (Motivation!) This however leads to another challenge: Electricity is measured in technical units which are not meaningful to users, as it is difficult for them to translate directly into monetary or CO_2 savings.

1.2 Research Questions

Adhering to the situation described above, one main research question comes to mind: How to leverage electrical energy consumption data in order to influence the users behavior? To answer this question it is necessary to split it into several sub-questions:

RQ1 How can the relevant information be extracted from the raw data? This question is focused on how to aggregate and interpret the data in a meaningful way.

RQ2 Is there additional information necessary to interpret the data? As a variety of factors might influence energy consumption it may be necessary to take them into account while analyzing the data.

RQ3 How is the extracted information best presented? As described above a problem with energy is, most people don't know how to interpret an energy unit. Hence, a way must be found to translate these units into tangible facts for the users frame of reference.

1.3 Goals

The aim of this thesis is develop a concept which allows the detection of energy wastage. This concept shall then be implemented prototypicaly. A web interface will act as means of communication, visualizing the evaluated data. As for time sensitive communication, the system should interact with an Android application, which is able to notify the user via push notifications, of possible improvements towards energy savings.

1.4 Outline

At first some background knowledge is provided to facilitate understanding the thesis (Chapter 3). Subsequently the use cases and any requirements of the system are analysed (Chapter 4). Leading to the design (Chapter 5) and implementation (Chapter 6). The thesis is then completed by the evaluation (Chapter 7) and conclusion (Chapter 8).

Chapter 2

Related Work

Several papers and studies address the possibility of saving energy by providing the inhabitants with feedback. In this chapter, four papers are presented, showing the potential of an advice system with regard to energy consumption.

Influencing electricity consumption via consumer feedback

This paper by C. Fischer [2], is aimed towards finding the best way to direct users towards energy savings. This means which kind of feedback has the most successful influence on the user. According to this paper, successful feedback exhibits the following properties:

- it is based on actual consumption
- it is given frequently
- it involves appliance-specific breakdown,
- it is given over a longer period of time
- it is presented in an understanding and appealing way

This paper thus implicitly answers the third research question, describing how the feedback is presented in the best possible manner.

The impact of informational feedback on energy consumption and **The effectiveness of feedback on energy consumption**

Both papers, by A. Faruqui et al. [3] and S. Darby [4], are written three years apart, in 2009 respectively 2006. Furthermore, both papers have similar findings. Both find, that in a residential household, the energy consumption can be reduced by up 15%, when

providing the inhabitants with adequate feedback. This illustrates, the potential of such an advice system, with regard to energy consumption.

Dormitory residents reduce electricity consumption when exposed to real-time visual feedback and incentives

This article, by J. Petersen et al. [5], in the International Journal of Sustainability in Higher Education, describes the effects of introducing a monitoring system for water and electricity, in a dormitory. The researches, introduced a real-time web-based feedback system on energy and water use in two dormitories. The introduction of the system resulted in a 32% reduction of energy use. The use of water however, only fell by 3 %.

Chapter 3

Background

Since some of the terms and knowledge used in the thesis is specific to the field of building automation, some of the core ideas will be introduced.

3.1 Smart Building

Literature does not provide a clear definition of the term smart building. As the focus of this thesis is on smart meters and outlet monitoring, a general definition of a smart building will suffice. The following general definition has been introduced by the universities of Toledo and Piraeus: “Generally speaking, smart buildings are expected to address both intelligence and sustainability issues by utilizing computer and intelligent technologies to achieve the optimal combinations of overall comfort level and energy consumption” [6].

Smart Meter

The term smart meter is commonly used, but not precisely defined. In connection to this thesis the following definition of smart meters proposed by the research department of Toshiba will be applied: “A smart meter is an advanced meter (usually an electrical meter, but could also integrate or work together with gas, water and heat meters) that measures energy consumption in much more detail than a conventional meter. [...] Smart meters are expected to provide accurate readings automatically at requested time intervals to the utility company, electricity distribution network or to the smart [building]” [7].

Outlet Monitoring

Smart meters monitor electricity consumption very unspecific (e.g. a floor, an apartment, etc.), meaning it is often not possible to attribute energy consumption to a specific user. As this may not be detailed enough, outlet monitoring has been introduced. Outlet monitoring is the process of monitoring how much power is consumed through each individual plug. This offers very detailed data on energy consumption.

3.2 Energy Monitoring Solutions

As most buildings are not pre-equipped with an energy monitoring solution, two retrofittable systems that are currently used at the chair are presented.

deZem

deZem is a system which is attached to the fuse box, where it monitors energy consumption. Being located at the fuse box, the *deZem* system is not capable of outlet monitoring. As it only measures the consumption of an electrical circuit, it has to be seen as a smart meter instead of an outlet monitoring device. However, it is able to collect data on a broader scheme, such as each floor or department. Once the data is collected through the *deZem* system it is logged centrally into a database. A key feature of the *deZem* system is, it does not interfere with privacy concerns, since it is not possible to match energy consumption to specific users [8].

HomeMatic

The *HomeMatic* system takes a more specific approach to energy monitoring, since it is a retrofittable outlet monitoring solution. The *HomeMatic* system consists of at least two parts: (1) an adapter to put between the outlet and the device connected and (2) a base station the adapters communicate with. The *HomeMatic* system can easily be extended by adding additional adapters and other sensors e.g. some that are able to detect whether a window is open or closed. The main reason to use the *HomeMatic* system is to supply additional detail where the *deZem* system is to inaccurate [9].

Chapter 4

Analysis

Energy wastage can be classified into two basic categories: (1) the devices are wasteful by themselves e.g. a filament lamp transforms most of the consumed energy into heat instead of light. This problem is solved by installing more efficient devices such as LED, which transform most of the consumed energy into light. (2) people are not using devices in the most energy saving ways. E.g. a printer is often left in stand-by over several days without it being used. The advice system developed in this thesis will only focus on habit driven energy loss.

4.1 Use Cases

Each of the following use cases will be set in the same scenario, describing a different overall setting where energy is wasted.

4.1.1 Setting

We assume a location in an office building where all outlets, lights and blinds are connected to a monitoring service. Furthermore it is expected, all employees have a certain amount of energy wastage due to habits e.g. leaving lights on, etc.

4.1.2 Dependency

An increasing amount of devices require running mode of other devices to be fully functional. This is commonly forgotten by the users since the idling device will go into stand-by fairly quick. Thus whenever a device is turned off it should be checked whether there are any other device which could be turned off as well e.g. whenever a PC is shut down if should be coupled with turning off the corresponding screen.

Use Case 1	**Dependent Devices**
Scope:	Local
Primary Actor:	Employee
Stakeholders and Interests:	• Employee: no additional hassle. • Facility Management: saving energy costs.
Preconditions:	The user leaves the workplace with one of the two dependent devices turned on.
Postconditions:	The remaining device has been turned off.
Main Success Scenario: 1. An employee shuts down the PC. 2. The system informs the employee that he didn't turn off the monitor. 3. The employee turns the monitor off.	
Frequency of Occurrence:	high
Open Issues:	• The PC might be rebooting. • A laptop running on battery might be connected to the screen.

4.1.3 Conflicts

Another common source of energy waste is when a user performs two actions that void each other. E.g. opening windows while the heating is turned on, or switching on the lights while the blinds are down.

Use Case 2	**Devices in Conflict**
Scope:	Local
Primary Actor:	Employee
Stakeholders and Interests:	• Employee: no additional hassle. • Facility Management: saving energy costs.

Preconditions:	The employee has lowered the blinds and the lights are turned on.
Postconditions:	• The blinds have been lifted or • the lights have been turned off.

Main Success Scenario:

1. An employee lowers the blinds.
2. After a certain amount of time the lights are turned on.
3. The employee is informed to either turn the lights off or lift the blinds.
4. The employee lifts the blinds or turns the lights off.

Frequency of Occurrence:	medium
Open Issues:	The sun might be glaring through the windows.

4.1.4 Time-Outs

In contrary to the first two cases the last case does not rely on two devices or actions. It simply specifies the temporal aspect of energy consumption, i.e. a device left in stand-by over an extended period of time instead of turning it off.

Use Case 3	**Time-Out**
Scope:	Local
Primary Actor:	Employee
Stakeholders and Interests:	• Employee: no additional hassle. • Facility Management: saving energy costs.
Preconditions:	The employee has printed a document.
Postconditions:	The printer has been turned off.

Main Success Scenario:

1. An employee uses the printer.
2. The employee returns to the workplace.
3. The employee is informed to turn off the printer.
4. The printer is turned off.

Frequency of Occurrence:	high
Open Issues:	Another employee might be about to use the printer.

4.2 Requirements

The scenarios described in the use-cases above yield some functional and non-functional requirements.

4.2.1 Functional Requirements

- **Detailed Energy Monitoring:** All use cases require the system to be aware of which device uses how much energy. This can only be achieved by very specific energy monitoring i.e. by monitoring the power consumption of each device.
- **Energy Wastage Detection:** The system must be capable to detect suspected energy losses. Any decisions made in this context should be based on a predefined set of rules.
- **Push-Notification Service:** A deficiency in energy consumption is a time sensitive matter. The employee might have left the building already, thus he is not in a position to resolve the detected inefficiency or without taking major inconvenience into account. Consequently this requires prompt means of communication.
- **Web Interface:** Any messages delivered via a push-notification are bound to be limited in length. For any further and more detailed information, as well as any management options, it thus is essential to provide a web interface.

4.2.2 Non-Functional Requirements

- **Real-Time Analysis:** For a user to be able to resolve a detected inefficiency in energy consumption the system must be able to detect the inefficiencies in real-time.

- **Ease of Use:** The system must not cause the user any additional hassle. Since the user won't be willing to use the system otherwise.
- **Efficiency:** The system should not consume more energy than it saves when it is deployed.
- **Fault Tolerance:** The system should not run into a fatal error when one of the energy monitoring components encounters a failure.
- **Extensibility:** It should be possible to extend the scope of the energy monitoring system, its functionality as well as integrating it into a larger building management system.

4.3 System-Context

As most buildings are not yet equipped with an energy monitoring solution, it is essential for the advice system to work with retrofittable components.

4.3.1 Information Gathering

The detection of inefficiencies relies on very detailed energy monitoring,. Smart meters are therefore not the most sensible choice to collect the needed data. The focus to gather energy consumption data will be on outlet monitoring solutions, predominantly the *HomeMatic* system is used.

A challenging aspect of the energy monitoring process is the fact, that some devices vary heavily in power consumption over a very short period of time. This means energy usage should be checked at least once per second. Figure 4.1 shows the energy consumption of an *HP Officejet 4630* inkjet printer. As can be seen clearly, the consumed energy spikes for about one second, while the printer is printing, otherwise it remains stable at about 4 Watts. If the power consumption were to be measured every minute it is very likely to miss the actual utilization of the printer.

4.4 Decision Making

The collected energy consumption data has to be evaluated to detect possible wastages. As described in the use cases at the beginning of the chapter, there are three types of inefficiencies to be considered: conflicts, dependencies, and time-outs. Any decisions

adhering to the first two categories follow the same structure. For simplification the process will only be elaborated for conflicts.

1. Any device types which are in conflict have to be defined.
2. For any given location it has to be checked, whether two conflicting device types are present.
3. Whenever two conflicting device types are present it has to be checked if they powered on at the same time.

Assuming that the last query returned "true", an inefficiency has been detected. However examining the case a wastage is produced by a time-out instead of a conflict the course of actions is slightly different.

1. For each device type a time-out time has to be defined.
2. All locations must be inspected whether a device which has time-out is present.
3. The power consumption history of any detected devices must be pulled and checked if the time span during which the device has not been used (i.e. sitting in standby) is greater than the time-out time.

Again, in case the last query returned "true" an inefficiency has been detected. The exact specification of the algorithms can be found in the Design chapter.

Both these processes assume, it is possible to detect the state of a device via the energy consumption. To verify this assumption the power consumption of some exemplary devices is inspected. As can be seen in figure 4.1 the amount of power consumed by a printer varies measurably between "Off", "Standby", and performing work such as printing or copying. A similar observation can be made for an IPS screen. Again the power consumption varies measurably depending on the sate of the device, as can be seen in figure 4.2. This measurable difference in power consumption allows to making all the needed decisions.

4.5 User Interaction

Whenever some sort of not optimal behavior is detected the need arises to interact with the user i.e. usually the employee. Due to the different types of information that need to be conveyed, the system should support two types of interaction: immediate and detailed communication.

Immediate Communication

Since any detected wastage is time sensitive, it is necessary to inform the user as fast as possible. The obvious choice for this kind of communication is sending push-notifications to the users smartphones. As any push notification is limited in length the message will only contain the most relevant aspects of any detection i.e. the location and the involved devices.

Detailed Communication

Any more detailed information that can not be included with the push-notifications should be sourced out onto a website. This page will provide management options such as adding new devices and offering more elaborate statistics on conflicts and power consumption.

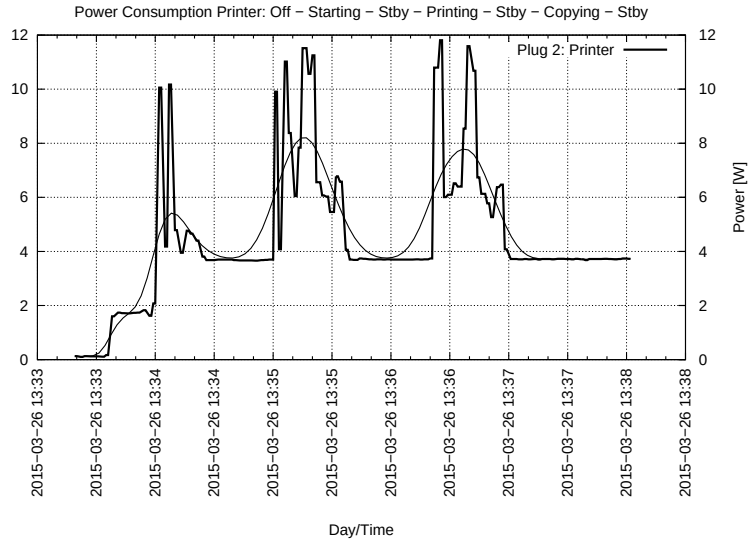

Figure 4.1: **Power Consumption Printer.** The power consumption of an *HP Officejet 4630* inkjet printer varies clearly over time depending on which actions are performed.

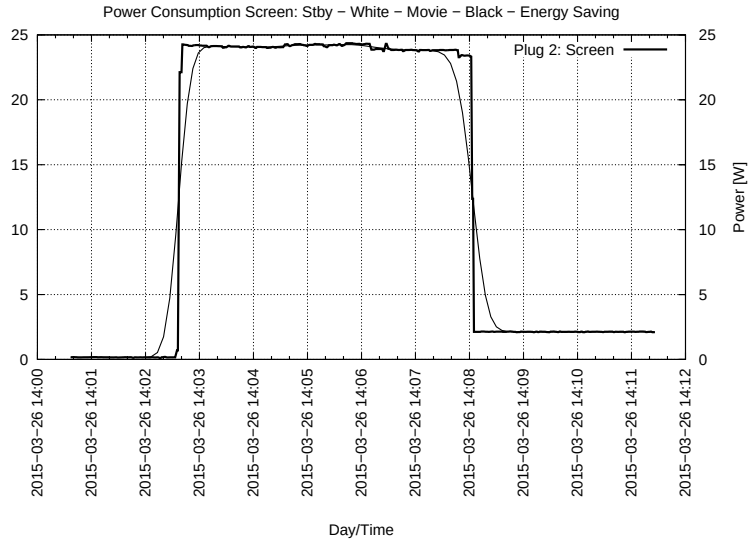

Figure 4.2: **Power Consumption Screen.** The power consumption of an Eizo IPS screen varies clearly over time depending on the state the device is in i.e. “Off”,“On”, and “Standby”.

Chapter 5

Design

The proposed advice system assumes a certain communication infrastructur within a smart building. Therefore a short overview about these assumptions is given. The detailed specification can be found in [1]. Any smart building is equipped with building server. This is a central gathering point where all the services the building provides are collected. The messages the building server receives will distributed to the users smartphone via push-notification. These independent services are deployed on separate servers. An overview of this design is given in figure 5.1.

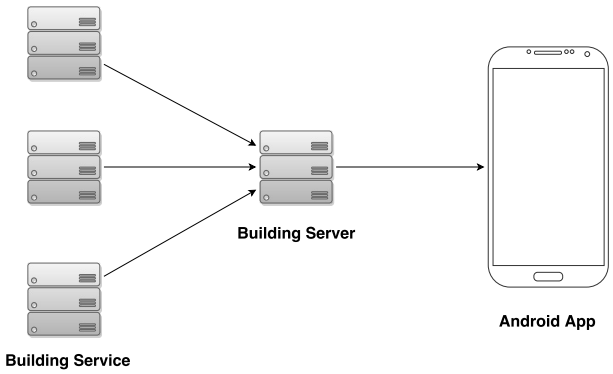

Figure 5.1: **Communication Design in a Smart Building.** The services a building provides are run on separate servers which send their communication requests to building server. The building server then relays the communication onto the users smartphone.

5.1 Components

Due to the inherent properties of the advice system, is split into two subsystems: (1) An interaction component and (2) an evaluation component. A basic overview is given in

figure 5.2.

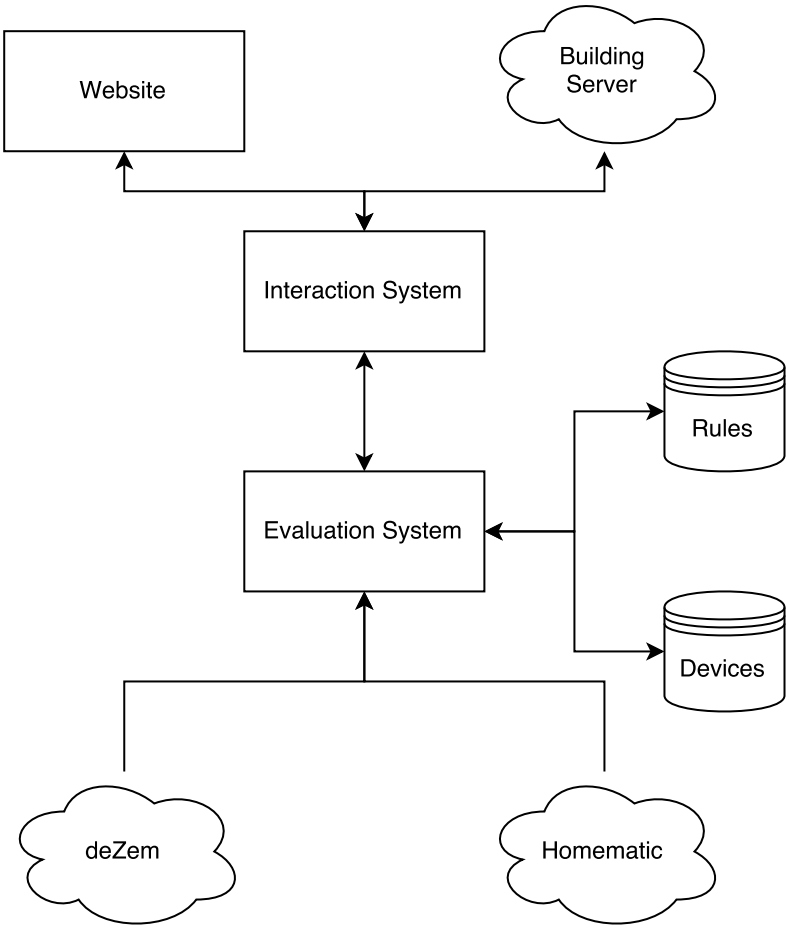

Figure 5.2: **System Design.** The entire advice system is split into two subsystems: the evaluation system at the bottom of the figure. And the interaction system at the top part of the figure.

5.1.1 Evaluation Component

The evaluation component is the core of entire advice system. This is the part where the logic is applied and inefficiencies are found. The evaluation system has access to three databases:

1. **Rules** This database stores potential inefficiencies. Since in general devices that are in conflict in one location are in conflict at another location as well, this database only supports device types instead of specific devices. The database is capable of storing 3 types of rules: (1) device types which depend one another, (2)

device types that are in conflict and (3) time-outs for device types. Some sample entries into the rules database can be seen in table 1. The first row shows a rule which describes that it running a screen relies on running a PC as well. Similar, the second row shows a rule describing the conflict between having the blinds down and the lights turned on. In both cases however the last column is not populated since it does not provide any meaningful information. The third row on the other hand shows a rule stating, a printer may be turned off after being unused for 30 minutes.

Table 5.1: Sample entries in the rules database.

Rules			
Device Type	**Device Type 2**	**Rule Type**	**Time-Out**
PC	Screen	1	-
Blind	Light	2	-
Printer	-	3	30
...	...	...	...

2. **Devices** The second database the advice system has access to, stores all relevant information on the present devices. This database is not implemented with the advice system, since it has been assigned sperately to a research assistant at the chair. Thus, the actual realization of the database is unknown. On a conceptual layer however the information stored is fixed. The database stores the following information on a device: (1) the location, (2) the device type, (3) the state the device is in during the day i.e. on, off, standby, (4) the state the device is in during the night, (5) the state the device is in during the weekend, (5) the power rating of the device, thus the amount of power it should consume at most, (6) the amount of power consumed during standby, and (7) the ID of the sensor it is connected to. For values containing information on power consumption the unit is Watt. The sensor ID is the ID of the Homematic plug the device is connected to.

3. **Power Consumption** This database collects all information associated with the energy consumption, thus the values provided the *HomeMatic* and *deZem* system. Implemtening this database has been assigned to the research assistant at the chair as well. Any information contained the database is accessible via a REST-interface.

As mentioned above, the task of the evaluation system is to detect any energy wastage, according to the rules. The algorithm to do detect conflicts assumes it receives two lists, which contain all devices matching the device types specified in the rule, it is supposed to check. E.g. to detect conflicts between lights and blinds, the algorithm requires a list with all the lights and a second list containing all the lights connected to the system. The algorithm then iterates through both lists trying to find any two devices at the same location. In case two devices are at the same location it is checked whether both are turned on. If this is the case the user is notified via the interaction component. A

representation in pseudocode notation is shown below.

Algorithm 1 check for conflicts

```
1: for each device d_1 in Device Type 1 do
2:     for each device d_2 in Device Type 2 do
3:         if d_1.location = d_2.location then
4:             if d_1.state = on ∧ d_2.state = on then
5:                 notify user
6:             fi
7:         fi
8:     od
9: od
```

Algorithm 2 check for dependencies

```
1: for each device d_1 in Device Type 1 do
2:     for each device d_2 in Device Type 2 do
3:         if d_1.location = d_2.location then
4:             if d_1.state = on ∧ (d_2.state = off ∨ d_2.state = standby) then
5:                 notify user
6:             fi
7:         fi
8:     od
9: od
```

In contrary to the first algorithm, the one detecting dependencies does not check if both devices are running. Instead it checks if one device is on and the other one is off or in standby.

The algorithm detecting time-outs on the other hand functions slightly different. Similar to the first two algorithms it expects a list containing all the devices of the device type specified in the rule. Then for each of the devices in the list, the previous power consumption is reviewed, determining whether the device has been used or not. If the device has not been used, the user is informed via the interaction component. These

Algorithm 3 check for time-out

```
1: for each device d in Device Type do
2:     powerUsed ← d.previousPowerConsumption(time-out duration)
3:     for each value p in powerUsed do
4:         if p > d.standbyPower then
5:             notify user
6:         fi
7:     od
8: od
```

algorithms conclude the description of the evaluation component.

5.1.2 Interaction Component

Whenever the evaluation component detects an energy wastage, the interaction components task is to interact with the user. As described in the Analysis chapter, user interaction splits into two parts, push and pull communication.

The push communication is handled through the building server [1]. This server expects a JSON message containing some meta-information and a text to display the user. Additionally there are two ways of limiting the users receiving the notification: (1) a location can be included, thus only users in a certain radius of the location will be notified, (2) a tag can be included, meaning only users which subscribed to a tag will be notified. The interaction system will wrap any findings the evaluation system made into a text, containing information on which which device(s) are concerned an what kind of inefficiency has been detected i.e. conflict, dependency or time-out.

The website on the other hand will display any content to large or detailed for a push notification, such as power consumption graphs. Exemplary graphs for aggregated power consumption and power consumption by device type, that could be placed on the website, are shown in figures 5.3 and 5.4.

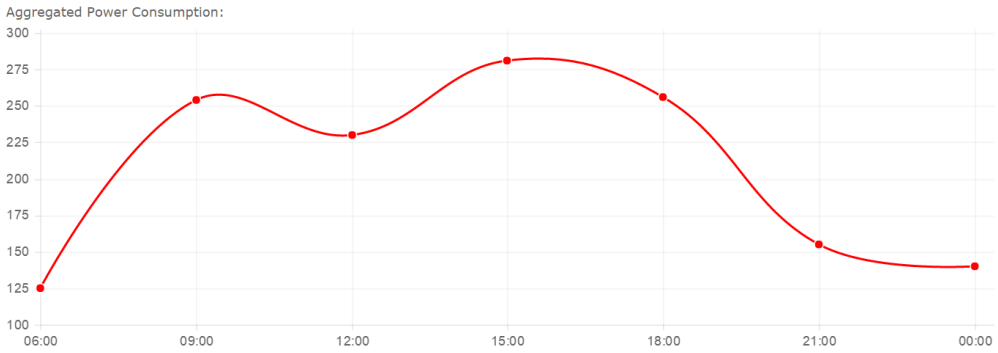

Figure 5.3: Example Graph - Aggregated Power Consumption

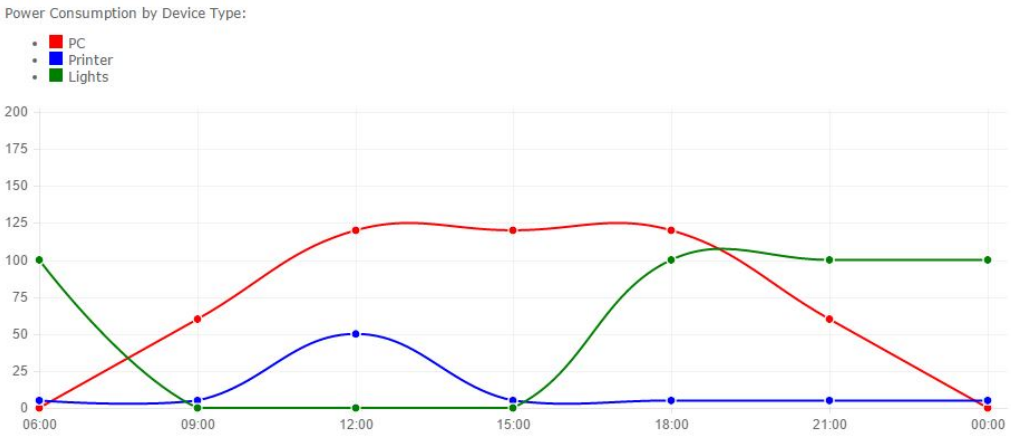

Figure 5.4: Example Graph - Power Consumption by Device Type

Chapter 6

Implementation

The core of the advice system is implemented in Java, since it is one of the most widely used programming languages. Furthermore as Java is run in the JVM it very portable and does not rely on a certain configuration or operating system to be fully functional.

6.1 Evaluation Component

The evaluation component consists of four classes: `Device, Rule, Message` and `Evaluation`. Additionally it is extended by the enums `DeviceType` and `RuleType` and a parser `ParseRules` allowing to input rules from a text file.

`DeviceType`

The enum `DeviceType` contains all the device categories know to the advice system. The implementation is shown in listing 6.1.

Listing 6.1: Enum `DeviceType`

```
1 public enum DeviceType{
2         PC, PRINTER, SCREEN, WINDOW, LIGHT, HEATING, BLIND;
3
4 //...
5 }
```

`RuleType`

The enum `RuleType`, similar to the `DeviceType` enum, contains the three rule types known to the system (Listing 6.2).

Listing 6.2: Enum `RuleType`

```
1 public enum RuleType{
2         CONFLICT, DEPENDENT, TIME;
3
4 //...
5 }
```

Device

Each device connected to the advice system is modeled by an instance of the class `Device`. To achieve this functionality the `Device` class comes with the following attributes:

Listing 6.3: Attributes of the class `Device`

```
1 public class Device{
2         private DeviceType type;
3         private String location;
4         private int powerRating;
5         private int standbyPowerRating;
6         private String sensorId;
7
8 //...
9 }
```

The attributes `powerRating` and `standbyPowerRating` contain data, on how much power the device uses when it is turned on, or on standby. Apart from the getter and setter methods, each device offers some additional functionality. The function `getState()` returns the state of the device, thus whether it is on, off or in standby. The function `getPreviousPowerConsumption(int min)` returns an array, containing power consumption of the previous `min` minutes.

Rule

The class `Rule` represents the entries into the rules database as a java object. The representation is shown in listing 6.4.

Listing 6.4: Attributes of the class `Device`

```
1 public class Device{
2         private RuleType type;
3         private DeviceType dev1;
4         private DeviceType dev2;
5         private int time;
6
7 //...
8 }
```

For simplicity, the rules are saved in text file instead of a database. As they exhibit a simply structure and no complex queries to the database are necessary, no functionality is lost. To read the rules from the text file they are saved in, the advice system offers the class `ParseRules`.

Evaluation

The `Evaluation` class implements the algorithms for detecting inefficiencies as described in the Design chapter. The following attributes shown in listing 6.5 are provided.

Listing 6.5: Attributes of the evaluation class

```
1 ArrayList<Rules> rules;
2 ArrayList<Device> devices;
3 ArrayList<Message> messages;
```

The first list contains all the rules known to the system. Whenever a check for wastage is run, all of the rules in this list are checked. The second list contains all the devices connected to the advice system, while the third list consists of all the inefficiencies which currently occur.

The main functionality of the `Evaluation` class is in the `check` method. On a logical level the method can be split into two parts: (1) finding the relevant devices, this part is shown in listing 6.6. The loops iterate through all the connected devices, to find those which are specified in the rule, currently being checked. However if no relevant devices are connected, the `check` method will return. As rules describing time-outs, do not have a second device type specified, the `if`-statement in line 12 is essential. And (2) detecting the conflicts, dependencies and time-outs of the devices. The algorithms established in the Design chapter have been implemented in java to achieve this goal. Listings 6.7, 6.8, 6.9 show the realization of algorithms 1 , 2 and 3

Listing 6.6: Finding relevant devices in the `check` method

```
private void check(Rule r) {
        ArrayList<Device> dev1 = new ArrayList<>();
        ArrayList<Device> dev2 = new ArrayList<>();

        for(Device d1 : devices){
                if(d1.getType() == r.getDev1())
                        dev1.add(d1);
        }
        if(dev1.isEmpty())
                return;

        if(r.getType != TIME) {
                for(Device d2 : devices){
                        if(d2.getType() == r.getDev2())
                                dev2.add(d2);
                }
                if(dev2.isEmpty())
                        return;
        }
// ...
```

Listing 6.7: Implementation of algorithm 1, detecting conflicts.

```
private void check(Rule r) {
// ...
switch(r.getType()) {
case CONFLICT:
        for(Device d1: dev1){
                for(Device d2: dev2){
                        if( d1.getLocation().equals(d2.getLocation) &&
                            d1.getState() == "on" &&
                            d2.getState() == "on"){

                            messages.add(
                                new Message(d1, d2, d1.getLocation));
                        }
                }
        }
        break;
//...
```

Listing 6.8: Implementation of algorithm 2 detecting dependancies.

```
1  private void check(Rule r) {
2  // ...
3  case DEPENDENT:
4          for(Device d1: dev1){
5                  for(Device d2: dev2){
6                          if( d1.getLocation().equals(d2.getLocation) &&
7                              d1.getState() == "on" &&
8                                  (d2.getState() == "off" ||
9                                   d2.getState() == "standby")
10                             ){
11
12                             messages.add(
13                                 new Message(d1, d2, d1.getLocation));
14                         }
15                 }
16         }
17         break;
18 //...
19 }
```

Listing 6.9: Implementation of algorithm 3, detecting time-outs.

```
1  private void check(Rule r) {
2  // ...
3  case TIME:
4          for(Device d1: dev1){
5                  double[] powerUsed =
6                          d1.getPreviousPowerConsumption(r.getTime());
7                  boolean used = false;
8                  for( double d : powerUsed){
9                          used = (d1.getState(d) == "on") || used;
10                 }
11                 if(!used)
12                         messages.add(
13                            new Message(d1, r.getTime(), d1.getLocation));
14         }
15         break;
16 //...
17 }
```

6.2 Interaction Component

The interaction has been split into two components, the website and the push communication, as outlined in the design chapter.

Website

The website is realized using `javascript`, enabling advance user interaction as well as websocket based communication. Websockets allow session based communication, meaning new data can be loaded to the website, without reloading the entire page. With the before mentioned choices, three functionalities have been implemented:

1. **Device Manager:** The device manager, which is implemented as a dynamic list on the website, offers two actions: adding a device and removing one.

2. **Charts:** The charts visualizes the consumed energy. This tab has been implemented using the `chart.js` library. `chart.js` is a lightweight `javascript` library producing interactive charts. This means the data in the chart can by updated dynamically, whenever a new value has been detected by the advice system.

3. **Conflicts:** The conflicts tab, is a dynamic list, similar to the device manager. It shows all inefficiencies currently detected by the advice system.

Push-Communication

As described in the design chapter, the building server offers the functionality to notify users on their smartphones. To do so, the server requires a message containing all relevant information. Listing 6.10 shows the format of the expected JSON-message. Not all of the included attributes are mandatory, for example the location is not a required attribute, as it offers the possibility to notify users in a certain proximity of a location. The information to be displayed in push-notification is included in the message tag. The title and text attributes contain the text shown to the user, the url tag specifies a link where the user can find additional information [1].

Listing 6.10: Publisher Notification [1]

```
{
  "publisherMessageId": "0",
  "messageType": "PublisherNotification",
  "location": {
    "latitude": ...,
    "doc_type": "GeoLocation",
    "radius": ...,
    "longitude": ...
  },
  "message": {
    "title": "...",
    "url": "...",
    "text": "...",
    "viewId": ...,
  },
  "publisherId": "..."
}
```

Chapter 7

Evaluation

7.1 Latency

Latency can occur in two places in the context of the advice system. (1) When detecting wastages and (2) when informing the user.

(1) The latency the advice system produces, is highly dependent on the hardware it is run on. As a high end server will produce less latency due to its capability to perform more operations per given time. Furthermore the current implementation checks the rules and connected devices sequentially. This causes the system to experience some latency when a large number of rules and/or devices are connected. However, the implementation could be adjusted to leverage modern multicore architecture, resulting in lower latency from the advice system.

(2) Another source of latency may be the smart building infrastructure informing the user via push notification. According to [1] this infrastructure experiences latencies of up to ca. 6 minutes when delivering a push notification. 6 minutes is an unacceptable latency in the given context, since the user might have left the building, or the inefficiency might no longer exist. As this issue is beyond the scope of this thesis, no concrete solutions will be shown, however with a different implementation of the interaction infrastructure, better results could be achieved.

7.2 Extendibility

If the advice system should be extended this is possible in multiple ways. (1) Adding new device types to the system and (2) and new types of inefficiencies.

(1) When a new device type shall be added to the system, it suffices to do two changes. Adding the new device type in the `deviceType` enum, and extending the parser to be

able to parse the new device type whenever it is mentioned in a rule.

(2) Adding new types of inefficiencies means extending the scope of the advice system, beyond the detection of dependencies, conflicts and time-outs. To achieve this goal, similar to adding a new device type, a new rule type has to be added to the corresponding enum and parser. Furthermore the `check` method has to be extended by an additional algorithm implementing the new type of inefficiency.

In conclusion, adding a new device type is very simple task when extending the advice system. Extending the scope, however, is a more complex task.

7.3 Efficiency

Generally, the larger the scale, the advice system is rolled out on, the more efficient it is. This is because as more devices are connected, the potential to detect inefficiencies increases. The following exemplary calculations will show the energy saving potential the advice system provides.

During the calculations it is assumed, that the advice system is run on an average x86 server. According to [10], such a server consumes on average 350W of power. This means the energy saving potential of the advice system has to greater than 350W, to run economically. To simplify the example, the only monitors and commercial grade printers will regarded. Furthermore, it is assumed, that the advice system is able to lower the time these devices are in standby. As for the setting, it is assumed, the advice system is running in an office environment, consisting of 30 offices. Each office is equipped, with a PC and two monitors. Furthermore, two commercial grade copy machines are present.

Ex. 1 - Monitors

According to the usage pattern, shown in table 7.1, the average power consumption of a screen evaluates as follows:

$$(32\% \cdot 36W + 36\% \cdot 10W + 32\% \cdot 1W) = 15,44W$$

Applying the assumptions from above, reducing the time spent in standby, the new average power consumption is:

$$(32\% \cdot 36W + 0\% \cdot 10W + 68\% \cdot 1W) = 12,2W$$

This equals a $3,24$ (20%) decrease in energy consumption, of each individual monitor. Generalizing these savings to the entire office environment, the total saving reach 194,4 Watts. This means, simply by reducing the time the monitors spend in standby, 65% of the power consumed by the advice system is economized. On an even larger scale, these savings would further increase.

Table 7.1: Average usage pattern of a monitor [11]

	Active	Standby	Off
Power Consumption	36 W	10 W	1 W
% of time in each state	32%	36%	32%

Ex 2 - Copy machine

Similar to the first example, the average power consumption, according to the usage pattern shown in table 7.2 evaluates to:

$$(5\% \cdot 3000W + 30\% \cdot 700W + 65\% \cdot 3W) = 362W$$

Again, applying the assumptions, the optimized average energy consumption is:

$$(5\% \cdot 3000W + 0\% \cdot 700W + 95\% \cdot 3W) = 153W$$

This is an energy saving of 57% (209 W). Applying these savings to both printers, the total savings would be 418 Watts. This means by simply optimizing the energy consumption of two printers, the power consumption of the advice system is easily economized.

Table 7.2: Average usage pattern of a commercial grade copy machine [11]

	Active	Standby	Off
Power Consumption	3000 W	700 W	3 W
% of time in each state	5%	30%	65%

Critical Remarks

Of course, these examples generalize in a way that disregards certain aspects. For example, reducing the amount of time a device spends in standby to 0, or disregarding that some devices, especially printers, consume an increased amount of power when booting. Nevertheless, the examples illustrate the enormous potential when saving energy in an office environment, especially on a large scale.

7.4 Other Requirements

Apart from the three requirements which have been evaluated extensively, further requirements were defined in the analysis:

- **Detailed Energy Monitoring** By using the *HomeMatic* components, the advice system is capable of monitoring the power throughput of each individual plug. This is the detailed level of energy monitoring required to detect wastage.
- **Energy Wastage Detection** Through the algorithms defined in the analysis chapter, the advice system is capable of detecting power wastage. This holds as long as the rules contain all relevant scenarios where energy could be wasted.
- **Push-Notification Service** The advice system is integrated into a larger communication infrastructure for smart buildings [1]. This infrastructure offers the possibility to notify users via push notifications.
- **Web Interface** As described in the implementation chapter, the advice system offers a website. This website provides access to a device management tool, as well as a detailed evaluation of the energy consumption data.

Chapter 8

Conclusion

8.1 Summary

In a large amount of buildings energy is not used in a most efficient manner. Studies show, energy savings of up to 15 percent can be achieved [4]. The measures to achieve this goal are split into two categories. (1) investive measures and (2) non-investive measures.

Investive measures to save energy, are measures where existing technology is replaced by a more energy saving model. For example replacing an old refrigerator with a new one, would qualify as an investive measure. Non-investive measures however, aim to change the habits of the users towards a more energy saving handling of technology. The developed advice system aims to be a non-investive measure. Further examining the energy wasting habits of inhabitants, three main sources of energy wastage arise. (1) Devices which depend on another device. An increasing amount of devices require another device to exhibit their full potential. A screen, for example, does not do anything, without a PC connected to it. (2) Devices/Actions which are in conflict. The actions provoking such conflicts, are most commonly subconscious actions. E.g. lowering the blinds, while the sun is glaring through the window. Then after some time, when the sun has moved on, the lights are turned on because it has become to dark, instead of lifting the blinds. (3) Time sensitive errors, these energy wastages occur when a device is left in standby or idling over an extended period of time, instead of turning it off. For all three types of wastage, algorithms have been developed, detecting them (RQ1). Furthermore, no additional information is necessary, to detect these types of inefficiencies, as all the necessary information can be found in the energy consumption data (RQ2). Moreover, a web interface allows detailed user interaction, and the integration into a communication infrastructure for smart buildings allows immediate communication via push notifications (RQ3).

The implementation proofs, it is possible to detect energy wastage in real time, and communicate with the user. However, the implementation has to be optimized to achieve the same functionality, when rolled out on a very large scale.

8.2 Outlook

Regarding the further development of the advice system, several possibilities exist.

1. As mentioned in the evaluation chapter, the system can be optimized to take advantage of modern multi-core architectures. This would allow real-time analysis on a large scale as well.
2. Presently, the advice system informs the user where energy is wasted. However, the potential savings are not translated into a non-technical unit, such as money or CO_2.
3. In the current state, the system simply informs users, where energy is wasted. In a next step, the system could become capable of resolving the inefficiencies by itself. This means turning of certain unused devices. Therefore the dependency on human actions would be eliminated.
4. The advice system could be integrated, even further, into the smart building infrastructure. For example, if a room booking system is present, the advice could be able to identify rooms which don't need to consume any energy at all.
5. As different users have different patters of energy use, thus different definitions of when energy is wasted, the advice system could implement multitenancy. This would allow each user to specify their own rules, instead of relying on global ones.
6. Currently the advice system relies on outlet monitoring to detect any inefficiencies. However, advanced data mining methods could be applied to aggregated power consumption data, to identify individual consumers. This would allow a similar advice, without equipping the entire facility with an outlet monitoring solution.

In conclusion, the advice system in its current state, offers all the basic functionality it is supposed to. However, many possible ways to enhance the current system remain.

Appendix A

Website

Figure A.1: Welcome Page

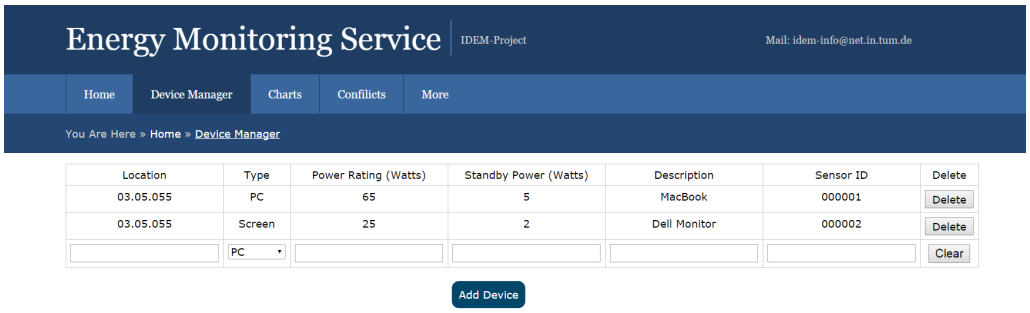

Figure A.2: Device Manager

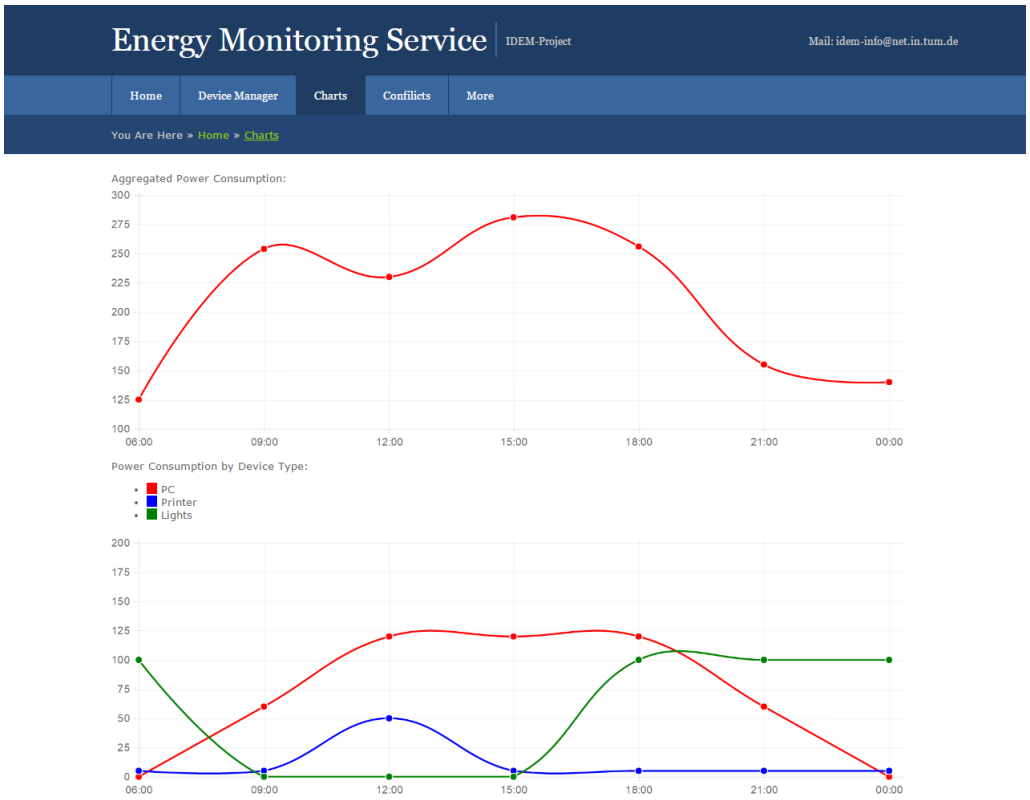

Figure A.3: Energy Consumption Charts

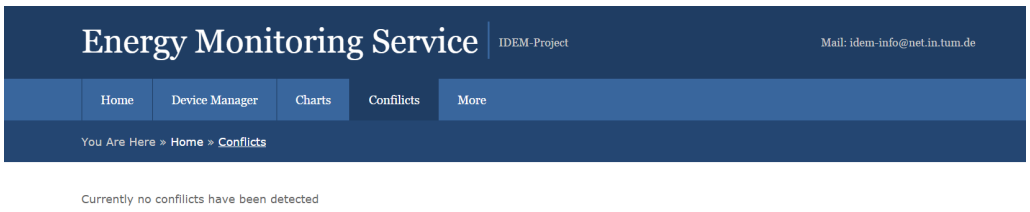

Figure A.4: Detected Conflicts

Bibliography

[1] A. Baus, "A feedback system for smart homes," Bachelor's Thesis, Technische Universität München, 2015.

[2] C. Fischer, "Influencing electricity consumption via consumer feedback: a review of experience," in *Saving Energy – Just do it!*, 2007.

[3] A. Faruqui, S. Sergici, and A. Sharif, "The impact of informational feedback on energy consumption," *Energy*, no. 35, pp. 1598–1608, 2009.

[4] S. Darby, "The effectiveness of feedback on energy consumption," Environmental Change Institute, University of Oxford, Tech. Rep., 2006.

[5] J. E. Petersen, V. Shunturov, K. Janda, G. Platt, and K. Weinberger, "Dormitory residents reduce electricity consumption when exposed to real-time visual feedback and incentives," *International Journal of Sustainability in Higher Education*, vol. 8, no. 1, pp. 16–33, 2007.

[6] Z. Wang, L. Wang, A. I. Dounis, and R. Yang, "Integration of plug-in hybrid electric vehicles into energy and comfort management for smart building," *Energy and Buildings*, vol. 47, pp. 260–266, 2012.

[7] C. Efthymiou and G. Kalogridis, "Smart grid privacy via anonymization of smart metering data," *Smart Grid Communications (SmartGridComm)*, 2010.

[8] "dezem energy controlling," 2015. [Online]. Available: https://www.dezem.de/

[9] "Homematic," 2015. [Online]. Available: http://www.homematic.com/

[10] S. Barielle, "Calculating tco for energy," *IBM Systems Magazine*, 2011. [Online]. Available: http://www.ibmsystemsmag.com/mainframe/Business-Strategy/ROI/energy_estimating/

[11] K. W. Roth, F. Goldstein, and J. Kleinamn, "Energy consumption by office and telecommunications equipment in commercial buildings," Arthur D. Little, Inc., Tech. Rep., 2002.