Karlsruhe Institute of Technology

Sensorgetriebene Information Appliances

Gruppe 1

SmartSecurity

Semester: Wintersemester 2016/2017

Teilnehmer (Matrikelnr.) : Christian Marten (1703791)
Daniel Dimitrov (1741166)
Rumen Donchev (1702732)
Joachim Göbgen (1708832)

Contents

1. Motivation

In the year 2014 the amount of burglaries reached the 150.000-stamp, which was the highest value for the last 16 years. In over 80 percent ground-floor windows or terrace doors were forced open or simply smashed. Luckily 40% of these burglaries were only attempts because the thieves were timely disturbed or just failed to complete their robbery for other reasons. Most of these robberies took place when none of the residents were at home – especially while they were at work.[1]

According to the German police, catching a thief in the very act of him breaking into the house is, in almost every case, the only way to get him.[2] Therefore a silent alarm is the best choice compared to the classic loud alarm which scares away the thief and leaves you with a broken window. A silent alarm just informs a person, security companies or the police about a possible burglary without letting the thief know he's already detected.

Classic alarm systems are unappealing with their high costs for the approach of a silent alarm. First of all the initial costs are very high (thousands of euros) if you try to secure all your ground-floor rooms. Secondly most of these systems were not able to alert the user directly but instead informs a security company which results not only in extra expenses but also in him not being able to react situation-dependent and time-critical.

This is where the 'Smart-Security'-approach kicks in: when an unusual behavior in the user's home is detected, informing him just in time is the first priority. Moreover is should be portable and worthwhile for every person. The already obtainable 'SmartSecurity'-approaches are based on a camera detecting motion (changing pixels) between frames. Our approach uses a temperature-change as detection that occurs when a burglar opens or smashes a window (mainly caused by the draught).

2. The Concept

The overall concept is to create home security as a modular system. The base version we narrowed our original concept down to provide a reliable break-in-detection by evaluating the temperature sensors data at the base station (old smartphone) and notifying the specified phone numbers for unusual changes via SMS. The user can then react to this anomaly by e.g. calling the neighbors.

[1] http://www.polizei-dein-partner.de/themen/einbruchschutz/einbruchschutz-intensiv/detailansicht-einbruchschutz-intensiv/artikel/einbruch-statistik.html on 26th of January 2016

[2] http://www.polizei-dein-partner.de/nc/themen/einbruchschutz/artikel-einbruchschutz/artikel/wie-einbrecher-vorgehen.html on 26th of January 2016

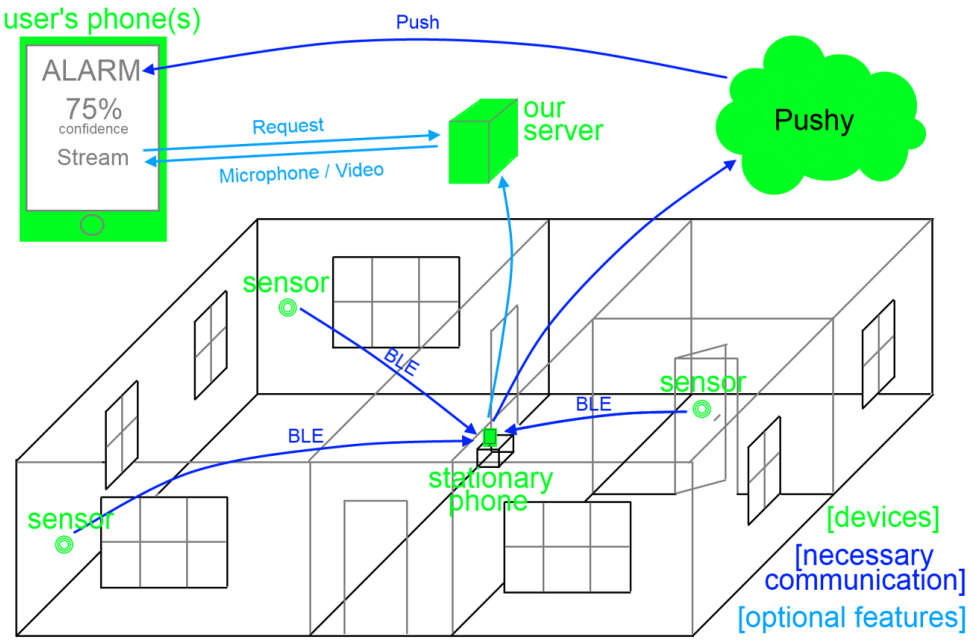

Fig. 1: Previous (and future) concept[3]

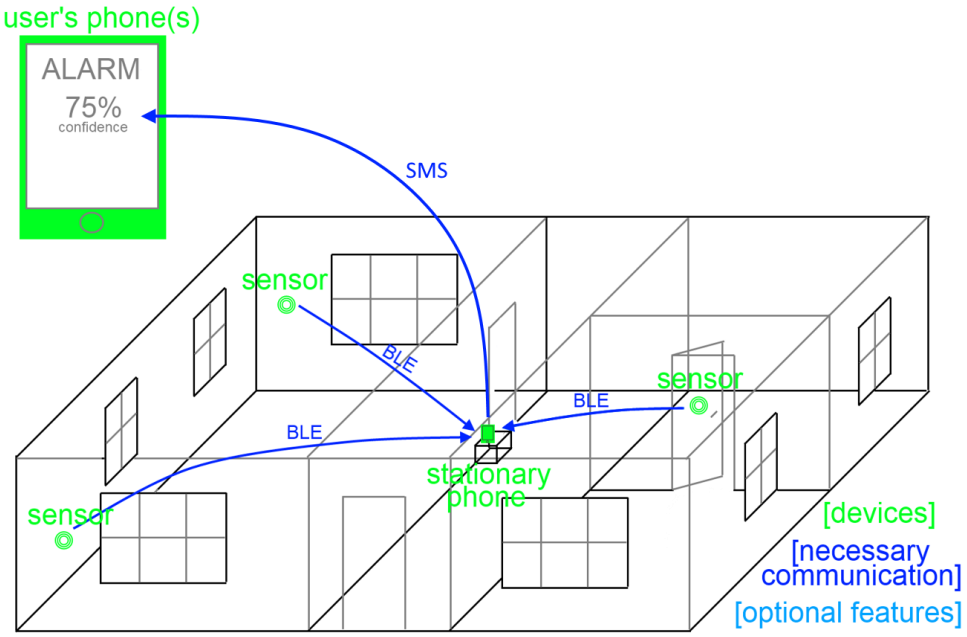

Fig. 2: Current (base) concept[4]

Our main concern is to quickly expand our system by services that give more options to the user, like for example a push-service as redundant notifier, a client app for more remote-options and a microphone or video stream. With a powerful ‘allrounder’ as base station (smartphone) we’re able to achieve this aim pretty quickly. For further details, see “7. Prosepects”.

[3] Own illustration
[4] Own illustration

3. Technical Details

3.1 Bluetooth

We built a Bluetooth connection according to the Bluetooth Low Energy standard. Each second the base station receives a data object with the current temperature. They are asynchronously packed into a sensor data packet as soon as they are submitted and passed to the prediction logic as soon as all packets are received (but by no later than one second).

3.2 Service

We needed a way to keep the app going even after it gets paused, say for example by pressing the power button (Standby) or by clicking the home button. We wanted our app to be unstoppable once turned on, so even after the UI part of the app gets killed the service restarts in the background and continues in the state it was on before that event. We run the service in a separate process, so we needed IPC (Inter Process Communication) to connect the UI and the Service. For that purpose we used AIDL (Android Interface Definition Language). The Bluetooth connections, the whole logic and SMS-sender are in this Service side of the app. The GUI is responsible for giving information to the service and control to the user.

3.3 Graphical User Interface

With the design of the user interface we tried to keep things simple. By using a taskbar that is consistent in every activity, the user's memory load is reduced. The buttons are simple and easily recognizable, often used in applications.
There are three main activities: Overview of the sensors, Settings, and Turning the alarm on and off.
The “Contacts” and the “Sensors” fields in the Settings activity can be modified. Objects in both fields can be added, deleted and edited, which automatically saves them in a database. Getting the users’ input happens in the same activity, through the usage of dialogs. The only exception is when adding sensors. Because it is scanning for sensors and might output a long list the available devices we decided that another window would be more pleasing than a small dialog box. By choosing the sensor and giving it a location or a name, the user is automatically returned to the “Settings activity”.

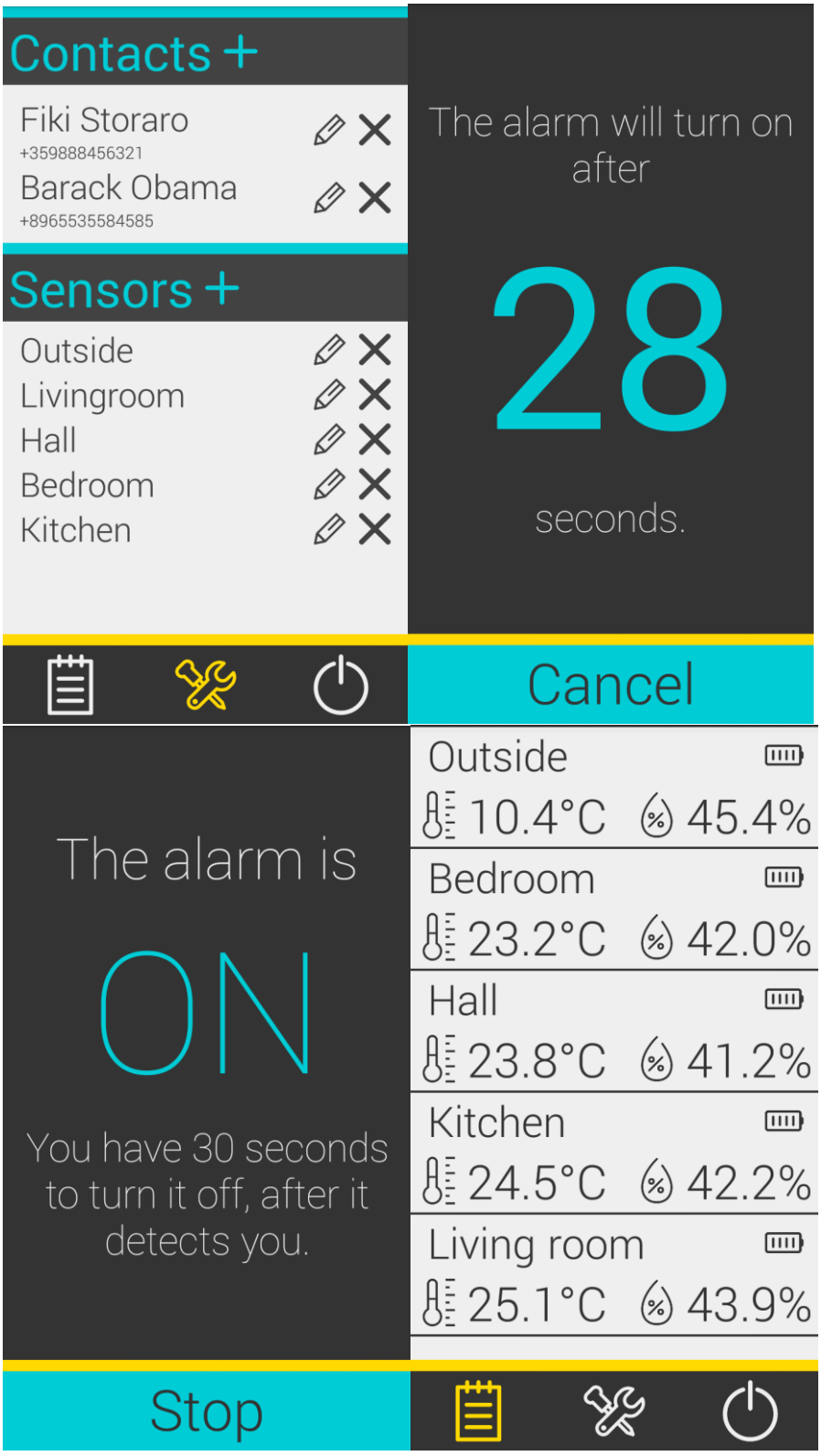

Fig. 3: Screenshots of the GUI[5]

[5] Source: Android app "SmartSecurity"

3.4 Prediction-logic

We tried out many more and less complex algorithms to detect a possible burglary from temperature change. Currently we decided for the following two methods, whose effectiveness in different situations has to be compared in even more tests.

Method I)
The logic behind the alarm decision consists of the following steps for each sensor:

- Linear regression
- Extrapolating newest values
- Compare extrapolated with measured values
- Send SMS if the differences' sum reaches a threshold

This procedure can be controlled by four parameters:

- Amount of values considered by the regression
- Amount of values to extrapolate and compare
- Emphasis of the newer values
- Threshold for positive detection

The steps in detail:

- The last 27 of the last 30 measured temperature values are weighted steadily between 0 and 3 (newer values weighted more).
- Make a linear Regression by the weighted least squares (WLS) method
  (in the short interval of 30s the values' trend can be assumed as linear, see „5. Testing" for further details)
- Predict the newest 6 values
  (all 6 values are assumed to be outlier and therefore not taken into the regression for not falsifying it)
- Calculate the difference between the predicted and the measured temperature value for each value, if the measured value is lower than the predicted one.
- Square the individual differences to punish a sudden deviation more than many slight deviations.
- Sum up those squared deviations and send an SMS if it's over a specified threshold.

Method II)
This is a simpler algorithm but the tests show that it's the better one in the standard cases. Instead of a comparison with extrapolated values, the measured values are compared with its predecessor.

This procedure can be controlled by four parameters:

- The exponent to punish a sudden deviation more than many slight deviations.
- The amount of values to be compared with its predecessor and added up simultaneously.

The steps in detail:

- Calculate the difference between the measured value and its predecessor for the last 6 values.
- Calculate the 1.33^{th} power of each difference to not ignore slight deviations too much but still value higher temperature jumps more.
- Sum up those deviations and send an SMS if it's over a specified threshold.

4. Economic Aspects

4.1 Market

The persona that is going to buy our product if John. John is 28 and has just got his master's degree. He has a job in Karlsruhe and is living with his girlfriend in a small flat. He has interest in new technologies, but is not known for spending ridiculous amounts of money for the newest gadgets. Although he thinks of himself as tech-savvy, he wants the product to just work. Both of them are working and often go out for the night or the weekend. In their adventures their most trusted companion is their smart phone.
John heard that there were a lot of burglaries in the neighborhood recently, but finds the thought of investing in an expensive security system appalling. Such young people do not put down roots fast and often change cities depending on the job prospects.

4.2 Business Model

There are two basic ways to sell our product: We could either sell the application and sensors or only sell the sensors.
For the moment the second way does not look that profitable, because people could buy the basic and non-modified sensors on their own and download our application for free. But on the other side paying for application may discourage some of our potential buyers. A possible way to solve the problem of people buying their own sensors would be creating our own.
If we choose to sell application and sensors, an unlock-code could be given with the purchase of the sensors so that the customer can download our application for free.

Regardless of the way we chose to sell our product the main focus of our efforts would be in providing additional services (see "4.4 Pricing strategy" for further information).

4.3 Competitors

There are quite a few competitors, who produce smart security systems. Most of them are located in the US and are not that widespread on the European market. The price of their devices is also quite high. Examples are "Canary", "Cocoon" and "Netatmo Welcome".

Another competitor is Samsung. They have developed a smart-hub named "SmartThings" (router-like device), that supports various wireless protocols and a series of sensors around the house. The cost of this device is 99$, but buying the peripherals would add another 200$.

But all of them have a different approach than we do: Their burglary-detection bases on motion detection through a camera instead of temperature change. This makes our product direction-independent and pet-friendly. Moreover their products consist of only one device that carries out all the tasks. Therefore you can only monitor one room at once which makes our system better concerning temporal aspects if the thief breaks into another room first. Furthermore our system is not affected by a power blackout because the base station has a long lasting battery, as well as sending an SMS works over the mobile network and does not depend on an internet connection, which can also be affected by the blackout or cut off by the burglars.

4.4 Pricing strategy and marketing

Our main aim is not to maximize the profit of the sensors' initial selling but maximize the range we can achieve with it.
Due to the fact that we have competitors in the 150$ to 250$ range, we can't make enough profit by the initial selling while being competitive enough in terms of functionality yet. That's why we try to pursue a strategy the other way round: Make the product cheap and benefit from a bigger group of buyers that we can later on convince with the smarter features which they pay for monthly. This is how we can bring long term revenue to the company.

To achieve this aim we have to announce these features already in the first version to display our value added and not make the people think it's the final product or overrun our buyers later on. Due to a very limited budget we can only draw on online marketing, e.g. through Google or Facebook ads. Another option to finance the project is going on Kickstarter. The additional benefit here is that a project promoted on Kickstarter is like free advertisement if the project is interesting enough for many backers or, in the best case, interesting enough for the populace.

Because our system works best with a high temperature difference between indoor and outdoor, spreading over Scandiavian countries could also be a lucrative option.

5. Testing

5.1 Testing Object

We used the most challenging house we were able to find:

- An old building (made in the 16th century)
- The bad isolation results in a fast temperature drop after stopping to heat the room
- A high ceiling and big rooms result in fast temperature balance within the room and therefore a small temperature change at one spot
- We were able to create low temperature differences between indoor and outdoor

5.2 Setup

The sensor was placed in three different spots inside the room (150-200cm away, 50-100cm away and mounted directly at the wall), as well as different indoor and outdoor temperatures. The window was opened for only up to 15 seconds to simulate the little time needed to climb in.

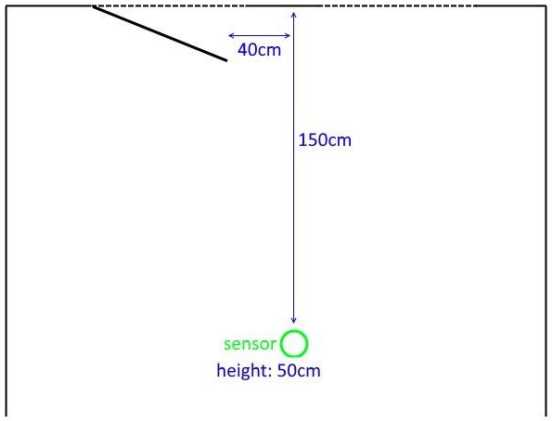

Fig. 4: Setup [A] [6]

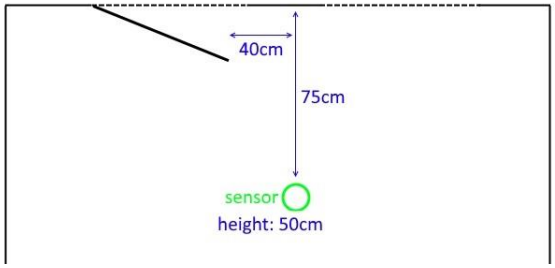

Fig. 5: Setup [B] [6]

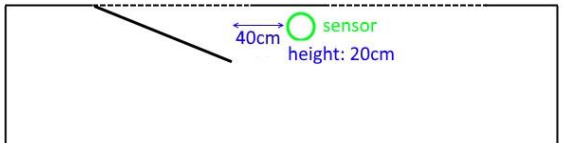

Fig. 6: Setup [C] [6]

5.3 Results

Explanation of the columns:

Temperature (in °C): The measured temperature of the sensor each second

Delta T (Method I): The difference between the current temperature and the initial one

Result (Method I): The value calculated by method I and to be compared with the threshold

Delta T (Method II): The difference between the current temperature and the previous one

Result (Method II): The value calculated by method II and to be compared with the threshold

The subtext shows the setup used for this measurement, as well as the difference between the indoor and outdoor temperature and the delay before the first change was detected.

[6] Own illustration

Temperature ***In °C***	**Delta T** ***Method I)***	**Result** ***Method I)***	**Delta T** ***Method II)***	**Result** ***Method II)***
18,73				
18,04	-0,69	**0,4761**	-0,69	**0,61**
17,85	-0,88	**1,2505**	-0,19	**0,72**
17,66	-1,07	**2,3954**	-0,19	**0,83**
17,45	-1,28	**4,0338**	-0,21	**0,95**
17,31	-1,42	**6,0502**	-0,14	**1,03**
17,2	-1,53	**8,3911**	-0,11	**1,08**

Measured with setup [A] and 15.7°C difference after 12 seconds.

Temperature ***In °C***	**Delta T** ***Method I)***	**Result** ***Method I)***	**Delta T** ***Method II)***	**Result** ***Method II)***
14,9				
14,42	-0,48	**0,2304**	-0,48	**0,38**
14,24	-0,66	**0,666**	-0,18	**0,48**
13,98	-0,92	**1,5124**	-0,26	**0,64**
13,78	-1,12	**2,7668**	-0,2	**0,76**
13,62	-1,28	**4,4052**	-0,16	**0,85**
13,53	-1,37	**6,2821**	-0,09	**0,89**

Measured with setup [A] and 11.9°C difference after 10 seconds.

Temperature ***In °C***	**Delta T** ***Method I)***	**Result** ***Method I)***	**Delta T** ***Method II)***	**Result** ***Method II)***
14,7				
13,55	-1,15	**1,3225**	-1,15	**1,20**
13,45	-1,25	**2,885**	-0,1	**1,25**
13,39	-1,31	**4,6011**	-0,06	**1,27**
13,39	-1,31	**6,3172**	0	**1,27**
13,39	-1,31	**8,0333**	0	**1,27**
13,39	-1,31	**9,7494**	0	**1,27**

Measured with setup [A] and 11.7°C difference after 12 seconds.

Temperature ***In °C***	**Delta T** ***Method I)***	**Result** ***Method I)***	**Delta T** ***Method II)***	**Result** ***Method II)***
21,04	0			
20,87	-0,17	**0,029**	-0,17	**0,09**
20,76	-0,28	**0,107**	-0,11	**0,15**
20,66	-0,38	**0,252**	-0,1	**0,19**
20,62	-0,42	**0,428**	-0,04	**0,21**
20,58	-0,46	**0,64**	-0,04	**0,22**
20,52	-0,52	**0,91**	-0,06	**0,24**

Measured with setup [A] and 9.0°C difference after 12 seconds.

Temperature ***In °C***	**Delta T** ***Method I)***	**Result** ***Method I)***	**Delta T** ***Method II)***	**Result** ***Method II)***
19,61				
19,41	-0,2	**0,04**	-0,2	**0,12**
19,34	-0,27	**0,1129**	-0,07	**0,15**
19,37	-0,24	**0,1705**	0,03	**0,16**
19,33	-0,28	**0,2489**	-0,04	**0,17**
19,32	-0,29	**0,333**	-0,01	**0,17**
19,34	-0,27	**0,4059**	0,02	**0,18**

Measured with setup [A] and 7.6°C difference after 14 seconds.

Temperature ***In °C***	**Delta T** ***Method I)***	**Result** ***Method I)***	**Delta T** ***Method II)***	**Result** ***Method II)***
11,18				
10,59	-0,59	**0,3481**	-0,59	**0,49**
10,5	-0,68	**0,8105**	-0,09	**0,54**
10,46	-0,72	**1,3289**	-0,04	**0,55**
10,4	-0,78	**1,9373**	-0,06	**0,57**
10,4	-0,78	**2,5457**	0	**0,57**
10,4	-0,78	**3,1541**	0	**0,57**

Measured with setup [B] and 8.2°C difference after 9 seconds.

Temperature ***In °C***	**Delta T** ***Method I)***	**Result** ***Method I)***	**Delta T** ***Method II)***	**Result** ***Method II)***
10,88				
10,47	-0,71	**0,5041**	-0,41	**0,30**
10,37	-0,81	**1,1602**	-0,1	**0,35**
10,31	-0,87	**1,9171**	-0,06	**0,37**
10,25	-0,93	**2,782**	-0,06	**0,40**
10,22	-0,96	**3,7036**	-0,03	**0,41**
10,18	-1	**4,7036**	-0,04	**0,42**

Measured with setup [B] and 7.9°C difference after 5 seconds.

Temperature ***In °C***	**Delta T** ***Method I)***	**Result** ***Method I)***	**Delta T** ***Method II)***	**Result** ***Method II)***
21,14	0			
20,78	-0,36	**0,13**	-0,36	**0,26**
20,66	-0,48	**0,36**	-0,12	**0,32**
20,56	-0,58	**0,696**	-0,1	**0,36**
20,45	-0,69	**1,173**	-0,11	**0,41**
20,4	-0,74	**1,72**	-0,05	**0,43**
20,35	-0,79	**2,344**	-0,05	**0,45**

Measured with setup [B] and 9.1°C difference after 11 seconds.

Temperature *In °C*	**Delta T** *Method I)*	**Result** *Method I)*	**Delta T** *Method II)*	**Result** *Method II)*
19,43				
18,96	-0,47	**0,221**	-0,47	**0,37**
18,87	-0,56	**0,534**	-0,09	**0,41**
18,78	-0,65	**0,957**	-0,09	**0,45**
18,75	-0,68	**1,419**	-0,03	**0,46**
18,61	-0,82	**2,092**	-0,14	**0,53**
18,57	-0,86	**2,831**	-0,04	**0,54**

Measured with setup [C] and 6.6°C difference after 16 seconds.

Temperature *In °C*	**Delta T** *Method I)*	**Result** *Method I)*	**Delta T** *Method II)*	**Result** *Method II)*
18,83				
18,78	-0,05	**0,002**	-0,05	**0,02**
18,78	-0,05	**0,005**	0	**0,02**
18,78	-0,05	**0,007**	0	**0,02**
18,56	-0,27	**0,08**	-0,22	**0,15**
18,5	-0,33	**0,189**	-0,06	**0,17**
18,49	-0,34	**0,305**	-0,01	**0,18**

Measured with setup [C] and 4.3°C difference after 15 seconds.

5.4 Interpretation

Despite the theoretically diminishing temperature drop over the whole time frame after leaving a heated room, there is always a linear and mostly even constant temperature trend observable. Therefore even a slight temperature difference is a big indicator for a robbery.
Tests with the sensor in different spots show that the draught is the main reason for a fast temperature change. If the sensor is placed at the same distance to the window but on its non-opening side, it detects a temperature change only if the window stays open for a longer time (which should be given in the most scenarios where a thief does not close the window behind him).
Unfortunately there's also a delay between opening the window and recognizing the first temperature change. One reason for this delay is the sensor going into some kind of energy saving mode in which it doesn't measure the temperature every second. The other reason is, of course, the slow air spread to the different sensor spots. Placing the sensor between 50cm and 100cm far away from the window results in the lowest delay (5s to 11s), placing it one meter more far away results in a medium delay (10s to 14s) and placing it right at the wall results in a delay of 15s to 16s, but it still works.

6. Current limitations

6.1 Functionality limitations

- The functionality and parameters are highly dependent on the room's property and the position of the sensor in the room
  (more testing needs to be done to get clear specifications where to position the sensor)
- Positioning the base-station in a relevant spot (for future camera and microphone function) by not getting out of range to all BLE sensors.
- An extra SIM-card is needed for sending an SMS.
  (not anymore with push-service, see "7. Prospects" for further information)
- Yet it was not possible to test how to react on an outdoor temperature that is higher than or even to the indoor temperature.

6.2 Security issues

We can classify cyberattacks on our product in two categories:

1. Attacks on the underlying platform and user's usage of the device:

Examples of attacks on the underlying platform is the new found CVE-2016-0728 Kernel vulnerability. There's a bug in the Linux kernel (version 3.8 and higher) that lets an attacker get root access. Approximately tens of millions of Linux PCs and servers, and 66 percent of all Android devices are affected.

Another example for user's usage is the WebView addJavascriptInterface bug. This was a vulnerability in an android browser that users could download from the Playstore. When heading to a website that takes advantage of this bug the attacker could gain root access to the device.

In these cases we have no influence on the exploits.

2. Attacks on our Implementation:

Due to the limited time of development we could not concentrate on testing the security of our application. For the moment we know of only one theoretical attack on our product. It is packet injection. The attacker can sniff Bluetooth packages, spoof the mac address of the device and retransmit the same package he captured. Then he can remove our sensor and the base station (phone) will not notice an abnormal activity and therefore not inform the client of the intruder.

One possible solution of the problem would be an encrypted connection. When the phone and the sensor connect for the first time there must be a handshake to have further communication encrypted.

In Prospects we already mentioned some ideas on expanding our service. Those ideas can also be threatened by cyberattacks. Since this is not the main focus for development in the moment, we will just mention a small list of things to have in mind when expanding our application:

"Build a service for sending audio / video from the base station over our server to the client's smartphone on burglary-detection"

- Man in the middle attacks
- DOS attacks on our server
- Privacy

„Automatically recognize when user leaves house (WiFi Sniffing)"

- Mac address spoofing

7. Prospects

Having a smartphone as base station is a huge benefit in terms of expandability. We can use the full potential of the strong processor, microphone, camera etc.

The first aim is to make the SmartSecurity system even smarter by adding different possibilities how to activate it. For now we have considered implementing timestamps for automatic (de-)activation as well as WiFi-sniffing to see if the users' smartphones (and therefore its owners) are home or not.

The next step is to build a client app so the users can make all configurations on his own smartphone and not necessarily on the base station to e.g. turn the alarm on.

As a third increment we plan to implement a push-service to our client- and server-app to provide a redundant (or alternative) possibility to notify the users. The client app could then have options for directly calling a specified number.

Finally we can implement our main functions to make money: Provide a microphone and/or video/picture stream to the users' smartphones over our own server, so he can verify a possible burglary.

Afterwards we can start improving our existing system parts, for example:

- Use a weather API to get the outdoor temperature and let it influence parameters like the threshold.
- Build our own sensors to make them cheaper. With the same technical specs, manufacturing our own sensor would cost us roughly 20$ instead of 30$.
    - 4,73$ microchip with BLE[7]
    - 3,42$ temperature sensor[8]
    - 3,50$ accelerometer[8]
    - 3$ case
    - 5$ assembling on a board

[7] https://store.ti.com/CC2541F128RHAT.aspx on 30th of January 2016

[8] http://www.mouser.de/ProductDetail/Sensirion/SHT21/?qs=sGAEpiMZZMvxTCYhU%252bW9mTLxwKc1ZK7f5n%252bRbASwrmY%3d on 30th of January 2016

8. Development Process

Before the development process began we had to define the problem and find a suitable solution. We went through many ideas. Some of the were: games, weather station, smart home remote control. Each of those ideas was scraped, because they were either too simple, had a suitable solution or the already existing framework that we wanted to develop for, was not mature enough for consistent user experience. In the end we defined our problem and solution as:

Problem: Fear from breaking in and no possibility of installing an expensive security system.

Solution: System of sensors and a mobile application.

Through the first iteration of the design process we came up with nine ways to detect a breaking:

1. Sensors on the doors and windows
2. Motion detection
3. Infrared beam
4. Camera
5. Temperature
6. Accelerometer
7. Microphone
8. Humidity sensor
9. GPS

After careful consideration Accelerometer, Humidity and Temperature, were chosen as sensors to detect home breaking. Camera and microphone could be used as an additional features, because they were part of our base station (smart phone).

By checking temperature and humidity our security system could identify a possible breaking if those values were not changing linearly. The accelerometer could detect motion if attached on the window or door. The camera can be used as a motion detector, when it looks for pixel changes and can also stream video. The microphone has the same capabilities, can detect sounds and alarm the user and can also stream sound to a website.

The software development process was divided into three parts: Logic, Bluetooth connection and GUI. Due to the late date of us getting the sensors development could first start in January. There were three scrum cycles:

In the first cycle an GUI was developed but changes to the focus of the application was made, so it was scraped. In the second cycle logic was implemented and another GUI was made. In the third cycle Bluetooth connection was added and there were some fine adjustments made.