WellBean

HSLU, Informatik Department, Switzerland

Group 2 (A. Althaus, P. Bucher, P. Kiser, B. Kuhn)

December 22, 2019

Group Members

Adrian Althaus
adrian.althaus@stud.hslu.ch
Patrick Bucher
patrick.bucher@stud.hslu.ch
Pascal Kiser
pascal.kiser@stud.hslu.ch
Benno Kuhn
benno.kuhn@stud.hslu.ch

Contents

1 Abstract 2

2 Introduction 2

3 Related Works 3

4 System Design and Implementation 3
- 4.1 System Overview . . . 3
- 4.2 System Architecture . . . 4
- 4.3 Software Architecture: Layers & Modules . . . 4
- 4.4 System Implementation, Functional Software Architecture . . . 5

5 Evaluation, Experiments, Results, Discussion 6

6 Application(s) 6

7 Conclusion 7

8 Contributions / Acknowledgments 7

9 Major Milestones & Deliverables 8
- 9.1 Team and Roles . . . 8
- 9.2 Project Planning, Timelines, Milestones & Deliverables . . . 8

References 9

1 Abstract

The *WellBean* project aims to monitor and assess objective and subjective parameters that have an influence on the quality of a workplace. Objective parameters, such as air quality (CO_2 concentration), temperature, and humidity are measured and monitored automatically. Subjective parameters, such as the perceived well-being, and interruptions from co-workers, can be tracked manually. Correlations and visualizations based on these parameters make it possible to gain insight on what measurements are considered to be beneficial to the workplace quality, and, thus, to overall workplace performance.

2 Introduction

Many factors have an influence on an employees well-being at the workplace. Some are subjective, others can be measured precisely with sensors.

The right temperature is important. Is it too cold, or too warm? Temperature is subjectivly felt more intensly as the humidity gets higher. 28 °C might still feel comfortable in dry conditions, but not so as the air's water saturation approaches 100%.[1] It is harder to focus on the work if the air quality is bad. High levels of CO_2 might cause drowsiness, and even cause headaches or nausea.[2] Interruptions can be harmful if one wants to focus on the work at hand. If too many of them occur in a given period, the work done soon approaches zero.

The idea of the IoT *WellBean* project is to track those parameters mentioned, as well as a subjective assessment of the well-being perceived by the employee, in real-time. The collected and visualized data allows to gain insight concerning the relationship between objective measurements (air quality, temperature) and the perceived well-being.

Such questions could be: *What temperatures and humidity levels do employees consider «good»? What is the effect on interruptions on the employee's well-being? Is it possible that interruptions cause bad air quality due to additional people in the room?*

[1] *«Cold air with high relative humidity ‹feels› colder than dry air of the same temperature because high humidity in cold weather increases the conduction of heat from the body. Conversely, hot air attended by high relative humidity ‹feels› warmer than it actually is because of an increased conduction of heat to the body combined with a lessening of the cooling effect afforded by evaporation.»* (Source: `https://www.infoplease.com/encyclopedia/earth/weather/concepts/humidity`)

[2] `https://www.kane.co.uk/knowledge-centre/what-are-safe-levels-of-co-and-co2-in-rooms`

3 Related Works

A general approach to the topic of measuring air quality with IoT devices is described in (Ibaseta, Molleda, Díez, & Granda, 2018). The article provides an overview of the most important gases determining air quality. Nicol and Humphreys investigate the notion of *Thermal Comfort*, which is a combination of objective and subjective factors in the perception of room climate (Nicol & Humphreys, 2002). Epstein and Daniel investigate on the relationship between thermal stress and the translation of the stress in terms of physiological and psychological strain (Epstein & Moran, 2006).

The effect of CO_2 concentration was investigated in (Satish et al., 2012). It can be shown that concentrations exceeding the level of 1000ppm increasingly start to have a negative impact on people's concentration. More information on CO_2 in general and its negative effect on well-being and efficiency can be found online as well[3].

4 System Design and Implementation

4.1 System Overview

The system consists of three major components. These are:

1. an Arduino with sensors,
2. a Raspberry Pi connected to the Arduino, and
3. a laptop computer serving as «the Cloud».

The sensors (SCD30, potentiometer, push button) take their input from the environment and the user, respectively. A small program on the Arduino collects the data every other second, and forwards it over a simple UART protocol via the USB port to the Raspberry Pi.

On the Raspberry Pi, there are running two processes: First, the *Data Collector* receives the measurements from the Arduino via the USB port, and converts them to a dictionary. Second, the *Data Forwarder* waits for incoming measurements in the form of a dictionary,

[3] https://www.businessinsider.com/office-air-co2-levels-making-workers-tired-2017-11?r=US&IR=T
https://ohsonline.com/articles/2016/04/01/carbon-dioxide-detection-and-indoor-air-quality-control.aspx
https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1600-0668.1999.t01-2-00008.x

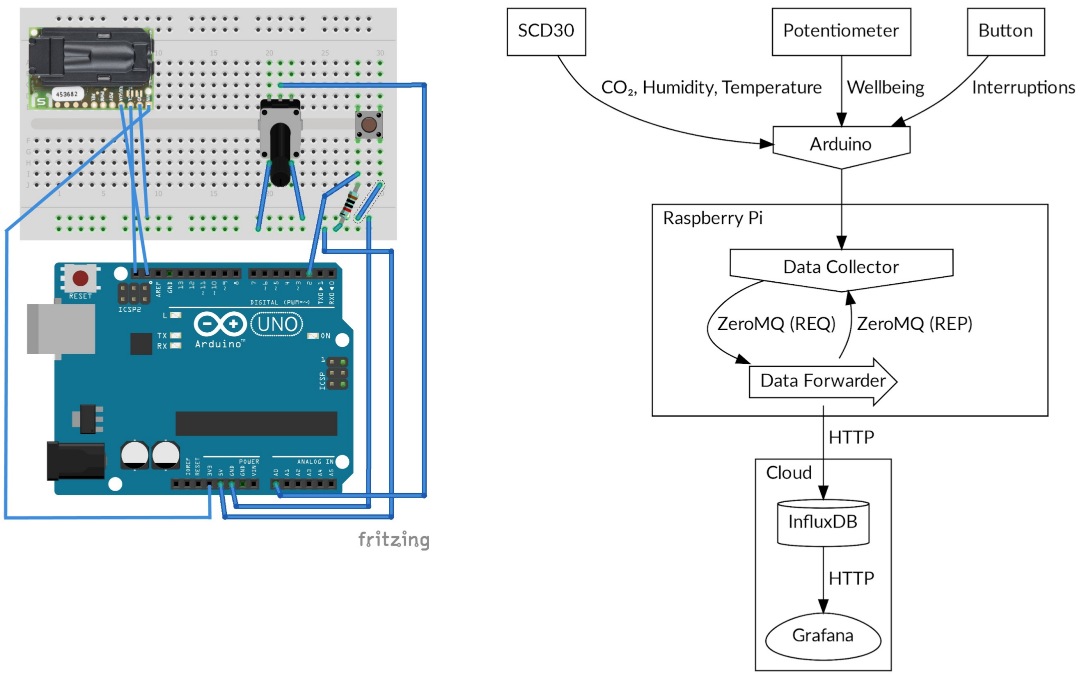

Figure 1: Model of Arduino circuit board and overall system architecture

converts it into a JSON structure, and forwards that to «the Cloud» using an InfluxDB client. On «the Cloud», the data is gathered in an InfluxDB and visualized in Grafana.

4.2 System Architecture

Figure 1, page 4 shows the overall system architecture.

4.3 Software Architecture: Layers & Modules

The Arduino and the Raspberry Pi communicate over a simple string-based protocol. The payloads consist of key-value pairs:

```
key=value,key=value,key=value
```

The following payload shows measurements for CO_2, temperature, humidity, interruption, and well-being.

```
co2=1024,temp=25.49,humid=47.39,interrupt=0,wellbeing=7
```

The SCD30 produces the three first measurements at least every two seconds. This time is spent actively waiting in the interruption detection loop for the button to be pressed. If the button was pressed during that period, 1 is reported, and 0 otherwise. This approach does not require any concurrency on the Arduino side, and only allows for an interruption being detected every other second, which is considered to be sufficient.

The string payload is terminated by a newline character. When the reading Data Collector process on the Raspberry Pi starts, all the characters until the first newline character are discarded, since the process might be ready in the middle of a transmission. The key-value string is converted into a Python dictionary by applying two split operations on it, first using the comma, second using the equal sign. The created dictionary is «pickled» (Python jargon), i.e. serialized into a list of bytes, and sent over to the Data Forwarder.

The two processes – Data Collector and Data Forwarder – are connected through a ZeroMQ socket, using a simple request-response protocol. This protocol could easily be switched to a publish-subscribe pattern, which would allow for multiple Data Collectors to report to one or many Data Forwarders, which in turn would allow to track the inputs from multiple Arduino devices in a multitude of data base backends.

The Data Forwarder «unpickles» (Python jargon again) the received bytes back into a dictionary, which is then converted to a JSON structure. This JSON structure is sent to the InfluxDB on «the Cloud» using HTTP. Grafana is used to visualize the time series stored on Influx DB.

4.4 System Implementation, Functional Software Architecture

The architecture described allows for a lot of flexibility. The processes on the Raspberry Pi are unaware of any semantics, they only process data as key-value pairs in various formats. If a different Arduino with different sensors is plugged into the Raspberry Pi on one side, then the visualizations on Grafana just needs to be configured to deal with the new table. All the knowledge required on both sides is the name of the key that is transmitted from the Arduino to the Raspberry Pi, and that will serve as the table name in InfluxDB.

On the Arduino, the `Wire` library is used for serial data transmission, as well as the header `SparkFun_SCD30_Arduino_Library.h` used for the sensor SCD30. Other configurations of the Arduino use the `Adafruit_BME280`.

The libraries used on the Raspberry Pi are `zmq` (ZeroMQ), `InfluxDBClient` (InfluxDB), in addition to the Python standard library.

5 Evaluation, Experiments, Results, Discussion

As for the CO_2 concentration, the different kinds of sensors tested all seem to react very sensitively towards different CO_2 exposure. When leaving a building, the CO_2 drops immediately from 1100-1600 parts per million (ppm) to around 450 ppm. Dedicated CO_2 return a more stable CO_2 concentration in comparison to the ones that infer the CO_2 by measuring Volatile Organic Compounds (VOC). The correlation between subjectively set well-being and CO_2 concentration has to be investigated in a further project, as soon as enough data has been collected.

The SCD30 worked well on the Arduino, but not so on the Raspberry Pi. After multiple tests with different libraries and approaches, the SCD30 was left on the Arduino bread board.

The pins of the SCD30 have been soldered manually to the sensor at home. This worked fine at the beginning, however, after the transport to the university, the sensor didn't respond any longer. The contacts have been checked positively using a potentiometer, so the sensor was suspected to be dead.

While conducting further experiments with the SCD30 at home, one pin broke off. This pin was now suspected to have caused the problems. When measuring with the potentiometer, pressure is applied to the pin, thus creating contact. After the pin was soldered on back to the sensor, it worked again without any issues.

The SCD30 was run in the office of one team member for a day, with a LCD display attached to the Arduino. The CO_2 concentration used to be around 1000ppm during the day. Opening a single window for half an hour only lead to a drop of about 200ppm. Opening two windows on opposing sides of the office lead to a much sharper drop to around 500ppm within five minutes. It is far better to air an office (or apartment) quickly with multiple windows open than for a long time with only one window open.

6 Application(s)

The WellBean was designed to be used in an office space, where every employee has a desk with power plugs, sufficient space for an additional device with the size of an Arduino on the desk, and an internet connection. In a real-world setup, multiple end devices would be connected to a Raspberry Pi serving as the gateway to «the Cloud».

The data collected could be used then by a data science team to gain insight into optimal workplace conditions, both for the team as a whole, as well as for the individual members

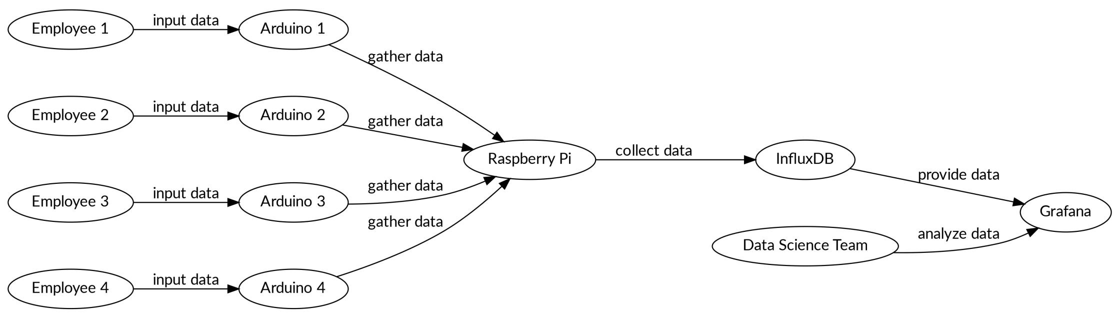

Figure 2: Possible Future Application for the WellBean (Codename: «BeanSalad»)

of the team, which might have different notions of good environmental conditions. Figure 2, page 7 shows such a possible setup.

7 Conclusion

The SCD30 turned out to be a very precise and useful sensor. ZeroMQ offers a lot of facilities that first need to be tried out and understood in order to implement M-to-N relationship between the sensor stations (Arduino or similar devices) and the backends (InfluxDB or others).

The system implemented provides a simple but solid framework to track various variables, be it in an office environment, or elsewhere. In order to be used productively for multiple workplaces, the data processing on the Raspberry Pi needs to be enhanced. Furthermore, an identity of the data source needs to be stored with the measurements on InfluxDB.

8 Contributions / Acknowledgments

Name	**Role**
Adrian Althaus	Planning, research using own devices, alternative CO_2 sensors, Testing CoaP
Patrick Bucher	Planning, research using own devices, InfluxDB, communication between devices
Pascal Kiser	Planning, research using own devices, Grafana
Benno Kuhn	Planning, research using own devices, Arduino input modules, modeling with Fritzing

9 Major Milestones & Deliverables

9.1 Team and Roles

Project Work Packages	Owner
Designing the Protocol (Arduino to Raspberry Pi)	Adrian Althaus
Testing 6V CO_2 and VOC sensor	Adrian Althaus
Testing various sensors	Benno Kuhn
Soldering SCD30	Patrick Bucher
Implementing Data Processing on the Raspberry Pi	Patrick Bucher
Configuring Grafana	Pascal Kiser
Designing the Logo	Pascal Kiser
Printing T-Shirts	Patrick Bucher

9.2 Project Planning, Timelines, Milestones & Deliverables

Our team did the planning and group organization during the first half of the project quite freely. Every group member possesses at least an Arduino or a Raspberry Pi, some of us also have sensors and actors for IoT devices. The first step consisted of gathering particular topics each member is interested in such as visualization, message queues or inter-device communication or electronics in general, which made it easy to assign the role and work packages to earch member. The group has done individual research and testing on their own devices by project in the first five weeks. Some team members bought additional sensors or displays, just for the sake of testing and playing around.

Afterwards, the idea of final architecture has become much clearer. Yet having two team members working independently on the goal using a different approach such as CoAP vs. HTTP communication has occurred througout the project. The decision whether or not a contribution will be part of the final system was also based on time resources. In terms of milestones, the team wanted to create a prototype consisting of a CO_2 sensor, Button, Potentiometer, RGB-LED, Arduino and a Raspberry Pi persisting all incoming sensor data in a local InfluxDB by November 29th.

The team used Gitlab[4] for all kinds of artifacts throughout the project.

[4] https://gitlab.enterpriselab.ch/IoT-I_BA_IOT/i_ba_iot_h19/group02

References

Epstein, Y., & Moran, D. S. (2006). Thermal comfort and the heat stress indices. *Industrial health*, *44*(3), 388–398.

Ibaseta, D., Molleda, J., Díez, F., & Granda, J. (2018, 11). Indoor Air Quality Monitoring Sensor for the Web of Things. *Proceedings*, *2*.

Nicol, J. F., & Humphreys, M. A. (2002). Adaptive thermal comfort and sustainable thermal standards for buildings. *Energy and buildings*, *34*(6), 563–572.

Satish, U., Mendell, M., Shekhar, K., Hotchi, T., Sullivan, D., Streufert, S., & Fisk, W. (2012, 12). Is CO_2 an Indoor Pollutant? Direct Effects of Low-to-Moderate CO_2 Concentrations on Human Decision-Making Performance. *Research*, *120*(12).