Comfort Home

Project Description

T-CB2 – Group 7

Group Members

Omar Shekho - 3588734

Danail Georgiev – 3781496

Ilian Stoev - 361859

Viktor Rumenov - 3608530

2019/2020

Comfort Home

- Introduction

- Background

- Project description:

- Use cases.
- Functional requirements.
- Non-functional requirements.

- Project organization:

- Roles & responsibilities.

- Project way of working

- Stand up meeting setups.
- sprint demos.
- communication plan.
- testing strategies.

- Deliverables

- Configuration management

- Risks analysis

Introduction

Houses are getting more and more insulated. As a result, the need of making a healthier inner living environment was arisen.

A sufficient ventilation system would be a perfect solution for this situation. Such a system will be able to remove condensation, reduce asthma, reduce temperatures, decrease VOC concentrations, and lower concentrations of Radon Gas.

Background

Airios, a market leader in control systems for the residential OEM industry, would like to have a system that can ventilate a room based on measurements done by various wireless sensors.

These sensors must communicate wirelessly with a ventilation box that can decide, based on the collected measurements, if and how much it should ventilate the room to improve the indoor climate (Air Quality).

Project description

- Use cases:

- **Use case: Switch ON or OFF**

Actor: Customer.

Main Success Scenario:

1- Customer enters the room, which the ventilation system is in.

2- Switches ON or OFF the system via remote control or phone app.

Extensions: (System has no connection with the app or remote control)

The system indicates malfunction in communication.

- **Use case: Ventilation cools off the room**

Actor: The system

Main success scenario:

1- The temperature sensor detects high temperature

2- The system gets the reading from the sensor

3- The ventilation is started to cool the room

- **Use case: Ventilation reduces humidity in the room**

Actor: The system

Main success scenario:

1- The humidity sensor detects the moisture in the atmosphere and sends it to the system

2- The system gets the reading from the sensor and checks if it is above the threshold

3- The ventilation is started to get fresh air in the room

- **Use case: Ventilation decreases VOC concentrations**

Actor: The system

Main success scenario:

1- The VOC sensor measures concentrations associated with bad air quality.

2- The system gets the reading from the sensor

3- The ventilation is started to get rid of the contamination

- **Use case: Ventilation reduces CO2 levels in the room**

Actor: The system

Main success scenario:

1- The CO2 sensor detects the levels of CO2 in the atmosphere and sends the reading to the system

2- The system gets the reading from the sensor and checks if it is above the threshold

3- The ventilation is started to get fresh air in the room

- **Use case: Sensor malfunction**

Actor: The system

Main success scenario:

1- The system indicates that a sensor is out of boundary or malfunctioning

2- The system indicates visually which sensor is out of boundary to the user.

- **Use case: Ventilation malfunction**

Actor: The system

Main success scenario:

1- The system receives information about malfunctioning ventilation

2- The system indicates visually malfunction to the user

- **Use case: Fix technical problem**

Actor: Technical support.

Main success scenario:

1- Actor receives notification about malfunction

2- Actor analyses log data

3- Fixes the problem

- Functional Requirements:

1- The system must trigger the ventilation when a sensor reaches a certain threshold.

2- The system will dynamically use the sensors in order to have continuous measure samples.

3- There must be communication between the sensors and the system.

4- The system must show visual cues with LEDs.

5- The system must display with LEDs which sensor is triggering the ventilation.

6- The visual cues from the LEDs need to indicate loss of traffic through the ZigBee.

7- The systems must be able to turn on the ventilation based on an algorithm, using the readings from the sensors.

- Non-Functional Requirements:

1- The system must be developed in a way to allow scalability and upgrades in the future.

2- The modules must be built independently so that the whole system does not crash if one of them does, which ensures the continuous runtime of the system.

3- The hardware shall be constructed as compact and simple as possible to allow for easy installation.

4- The embedded systems themselves need to be completely secure so that they don’t malfunction or represent any sort of hazard.

5- Measurements from the sensors must be sent frequently to the computer, at least once every 15 minutes.

6- Every part of the systems must need no adjustments from the user. after the installation, it should be configured to work continuously and independently.

7- The system must use a predetermined, configurable algorithm to control the ventilation.

8- The communication between the sensors and the system must be bi-directional.

Project organization

- Roles & responsibilities:

- Team leader - Danail Georgiev:

A team leader is someone who provides direction, instructions and guidance to the group, creating an inspiring team environment with an open communication culture. He or she should provide clear team goals and listen to team members' feedback and resolve any issues or conflicts.

- Test manager – Everyone (Lead Test manager - Omar Shekho)

Prepares appropriate tests for the finished parts of the project so that the work can be evaluated properly. Ensures that these tests run smoothly and then collects feedback and presents it to the team.

- Developer – Everyone (Lead Developer - Ilian Stoev)

A developer is responsible for designing and developing software systems using scientific analysis and mathematical models to predict and measure outcomes and design consequences. Which results in producing clean, efficient code. In addition, consulting with departments or customers on project status and proposals is one of his/her tasks.

- Quality assurance - Everyone

Prepares the team and the project for the sprint demos, the tests and most importantly, the release of the product. Make sure that everything works to the best of its abilities before that.

Project way of working

Stand up meeting setups:

The team meets once a week to evaluate the progress made in the previous week. Afterwards, further actions towards the final goal should be planned and the tasks for the next week should be defined. Any issues that may have appeared will also be discussed and solved. Meeting the monitor of the project will take place every week after finishing the main tasks to get valuable feedback.

Sprint demos:

Every Monday there will be a meeting scheduled with the teacher and possibly a representative of the company to take a look at what the team has as a current version of the project. The aim of that is to receive important and helpful feedback, which should be implemented during the rest of the week.

Communication plan:

The team will always have a platform to communicate - either WhatsApp or Discord. That way, we will have a way to contact each other in urgent situations. We have agreed to give feedback to each other about our work frequently, in order to make sure that everyone is satisfied with the project. We also have a meeting designated especially for that every week.

Testing strategies:

The tests will be planned carefully, in order to be as informative as possible. We will try out different testing techniques and decide which one is the most objective and provides us with the most valuable feedback. That way, further testing will be done with the appropriate techniques and we will be able to find all the issues that may appear in the sprint demos or the final release of the product.

Deliverables

Project Plan document:

This document answers the most vital questions before starting with the project.

For instance, what are the functional and non-functional requirements, how are we going to work as a group? What are our roles, responsibilities and strategies? What are we going to deliver to the customer at the end of the project? In addition, this plan predicts all the risks related to such a project and tries to find the appropriate solutions in advance.

Nucleo hardware code:

The Nucleo code is going to provide a clear view of the logical function of how the project actually works, with the help of comments inside the code for better explanation.

Project Report:

In the project report there are going to be a couple of sections, which will explain from start to end what happened during the past 18 weeks. The information will include the project description in detail, what roles the

team has, what is the purpose of them and most importantly what we accomplished as a team, how we reached our goal and made a good functional project.

The System Hardware sample:

We will build a demo of the hardware the product requires in order to demonstrate it. It will include all the sensors and the wiring needed to connect them, as well as a simple air conditioning fan to show how the system reacts to the sensors.

C# Application:

The C# application will be a simulation of the ventilation box and will log all the sensors input control and the air conditioning. It will also have the ability to have new sensors added to it and it will decide based on an algorithm how to control the fans based on the readings it receives.

Configuration management

How to tag:

Tagging in Git refers to creating specific points in history for your repository. This is usually done to mark release points. Here is how to create a tag in Git:

Step - 1: Check the branch you want to create tag

Command: git checkout <branch name>

Example: git checkout master

Step - 2: Create tag with a name

Command: git tag <tag name>

Example: git tag v1.0

Step - 3: Display or show tags

Command: git show <tag name>

Step 4 - Push tags

Command: git push origin <tag name>

Example: git push origin v1.0

Example: git show v1.0

Bonus step: Delete a tag(if necessary)

Command: git tag -d <tag name>

Example: git tag -d v1.0

How to branch:

By default, the way Git should be used by creating different branches based on the status on the project and then committing code in that specific branch in order to not create issues with the master branch, where clean and working code is stored. Here is how to create a branch:

Creating a branch:

Command: git branch <branch name>

Example: git branch alpha

Check how many branches there are:

Command: git branch –list

Switch to the branch we have created:

Command: git checkout <branch name>

Example: git checkout alpha

Risks analysis

Category	Source of risk	Probability	Risk management
Technical	Lack of equipment or some components such as sensors and actuators	High	Check their availability at the ISSD early and buy the remaining if necessary
Social	The sudden absence of one of the team members	Low	Making sure that we inform one another about any accidental conditions.
Time management	Lack of needed time before delivering a working demo	Medium	Schedule the work well and predict delays for catching up.
Social	Miscommunication with the customer about some parts of the project	Medium	Double check all the information given and update the customer frequently on progress made

Technical	Activities involving integrations of programmes or technologies	High	Working on the same principles within the same environments
Project management	The potential for a member to fail to meet their obligations to the project	Medium	Assigning a task to the right member for a reasonable time