



# Design of Embedded Systems

## ESSTA, Energy Saving Smart-home distributed Temperature control Application

## Requirements

*Falzone Giovanni*

*jointly M.Sc Embedded Computing Systems*

*Sant'Anna School of Advanced Studies*

*University of Pisa*

June 10, 2019

# Contents

# 1 Introduction

The purpose of this document is to describe the *User requirements* and the *Functional specification* of the project in order to have a clear description of what the system shall do and which will be the tests to check the behaviour of the system.

# 2 User Requirements

## 2.1 Central Unit

### 2.1.1. Graphical User Interface on Central Unit module

#### 2.1.1.1. Main Page

2.1.1.1.1. Whenever the *Main page* is selected the module shall represent the average values among all the rooms for *Temperature*, *Humidity* and *Usage*.

2.1.1.1.2. Whenever the *Main page* is selected the module shall represent the *Energy Saving* if at least one room is set to **Energy Saving mode**.

2.1.1.1.3. Whenever the *Main page* is selected the module shall represent the *Warning* if at least one room is set to **crashed**.

2.1.1.1.4. Whenever the *Main page* is selected the module shall allow the user to move in the *Settings page*, next and previous *Room page*.

#### 2.1.1.2. Settings Page

2.1.1.2.1. Whenever the *Settings page* is selected the module shall represent the *Desired Temperature* and shall allow the user to increase or deacrease it by a factor of 0.5 C°in the range of 15 C°and 30 C°.

2.1.1.2.2. Whenever the *Settings page* is selected the module shall represent the average values among all the rooms for *Humidity* and *Usage*.

2.1.1.2.3. Whenever the *Settings page* is selected the module shall represent the *Energy Saving* if at least one room is set to **Energy Saving mode**.

2.1.1.2.4. Whenever the *Settings page* is selected the module shall represent the *Warning* if at least one room is set to **crashed**.

2.1.1.2.5. Whenever the *Settings page* is selected the module shall allow the user to move in the *Main page*.

#### 2.1.1.3. Room Page

2.1.1.3.1. Whenever the *Room page* is selected the module shall represent the average values among all the rooms for *Temperature*, *Humidity* and *Usage*.

2.1.1.3.2. Whenever the *Room page* is selected the module shall represent the *Energy Saving* if at least one room is set to **Energy Saving mode**.

2.1.1.3.3. Whenever the *Room page* is selected the module shall represent the *Warning* if at least one room is set to **crashed**.

2.1.1.3.4. Whenever the *Room page* is selected the module shall allow the user to move in the *Main page*, *Settings page*, next and previous *Room page*.

## 2.2 Room

### 2.2.1. Energy Saving mode

2.2.1.1. Whenever a motion is detected in the last 30s the module shall move in **Enery Saving** mode and show it through the ENERGY_SAVING_LED

2.2.1.2. Whenever a motion is detected shall show it through the ENERGY_SAVING_LED

# 3 Functional requirements

## 3.1 Central Unit

### 3.1.1. Communication

3.1.1.1. The *Central Unit* shall send a *Room Request message* polling among all the rooms

3.1.1.2. The *Room Request message* shall include the Id of the room and the *Desired Temperature* in Celsius°

3.1.1.3. The *Room Status message* shall include the Id of the room, the *Energy Saving mode* one if active zero otherwise, the *Temperature* in Celsius°, the *Humidity* in % and the *Valve position* in %

3.1.1.4. Whenever a *Room Status message* is corrupted or doesn't arrive within 5s from the sent of the *Room Request message*, the same *Room Request message* shall be resent until 3 times before marking the room as **crashed**

## 3.2 Room

### 3.2.1. Initialization

3.2.1.1. The module shall check the position of the valve moving in CLOSED, LOW, MIDDLE, HIGH and OPEN

3.2.1.2. The module shall check the correctness of the sensors

3.2.1.3. The module shall turn on all the LEDS and the turn off

### 3.2.2. Running

#### 3.2.2.1. Energy Saving mode

3.2.2.1.1. Whenever the module is in energy saving mode the module shall turn on the ENERGY_SAVING_LED

3.2.2.1.2. Whenever the module is in energy saving mode the *Goal Temperature* shall be set to the *Desired Temperature* minus the *Energy Saving Temperature Offset*

#### 3.2.2.2. Normal mode

3.2.2.2.1. Whenever the module is in normal mode the module shall turn off the ENERGY_SAVING_LED

3.2.2.2.2. Whenever the module is in normal mode the *Goal Temperature* shall be set to the *Desired Temperature*

#### 3.2.2.3. Error mode

3.2.2.3.1. If the **communication error** or **valve error** or **sensor error** is set the module shall turn on the ERROR_LED

### 3.2.3. Sensors Control

3.2.3.1. The module shall check the motion every MOTION_PERIOD seconds

3.2.3.2. Whenever a motion is detected in the last MOTION_TIMESLOT the module shall move in **Enery Saving** mode otherwise in **Normal mode**

3.2.3.3. Whenever a motion is detected the module shall turn on the ENERGY_SAVING_LED otherwise shall turn it off

3.2.3.4. The module shall read the Temperature every TEMPERATURE_PERIOD seconds and check if it is between MIN_TEMPERATURE and MAX_TEMPERATURE Celsius °

3.2.3.5. The module shall read the Humidity every HUMIDITY_PERIOD seconds and check if it is between MIN_HUMIDITY and MAX_HUMIDITY in percentage

3.2.3.6. The module shall set the **sensor error** and move in **Error mode** whenever at least one read data is not consistent with the allowed range of values

3.2.4. **Valve Control**

3.2.4.1. The module shall check and move the position of the valve every VALVE_PERIOD seconds

3.2.4.2. The valve shall be in OPEN_POSITION whenever the difference between the actual temperature and the *Desired Temperature* is below -HIGH_THRESHOLD C°

3.2.4.3. The valve shall be in HIGH_POSITION whenever the difference between the actual temperature and the *Desired Temperature* is greater or equal then -HIGH_THRESHOLD C°and below -APPROACHING_THRESHOLD C°

3.2.4.4. The valve shall be in MIDDLE_POSITION whenever the difference between the actual temperature and the *Desired Temperature* is greater or equal then -APPROACHING_THRESHOLD C°and below or equal then APPROACHING_THRESHOLD C°

3.2.4.5. The valve shall be in LOW_POSITION whenever the difference between the actual temperature and the *Desired Temperature* is greater then APPROACHING_THRESHOLD C°and below or equal then HIGH_THRESHOLD C°

3.2.4.6. The valve shall be in CLOSED_POSITION whenever the difference between the actual temperature and the *Desired Temperature* is greater then HIGH_THRESHOLD C°

3.2.4.7. Whenever the valve is in OPEN_POSITION or in CLOSED_POSITION the module shall check the consistency of the status using the OPEN_SWITCH and CLOSED_SWITCH

3.2.4.8. If the status of the valve is not consistent the module shall set the **valve error** and move in **Error mode**

3.2.5. **Communication**

3.2.5.1. Whenever the module does not receive the *Room Request message* within COMMUNICATION_DEADLINE seconds from the last request message, the module shall send a *Room Status message*, set the **communication error** and move in **Error mode**

3.2.5.2. The module shall check the correctness and consistency of the *Room Request message* and discard the corrupted messages

3.2.5.3. The module shall send the *Room Status message* within COMMUNICATION_PERIOD seconds from the last *Room Request message* reception

## 3.3 parameters

in the following table are reported the constant values required and used in the previous requirements.

| **Parameter** | **description** |
|---|---|
| MOTION_PERIOD | 2 seconds |
| TEMPERATURE_PERIOD | 2 seconds |
| HUMIDITY_PERIOD | 2 seconds |
| VALVE_PERIOD Saving | 4 seconds |
| COMMUNICATION_PERIOD | 1 seconds |
| COMMUNICATION_DEADLINE | 30 seconds |
| OPEN_POSITION | valve position to have the 100% of the maximum flow |
| HIGH_POSITION | valve position to have the 75% of the maximum flow |
| MIDDLE_POSITION | valve position to have the 50% of the maximum flow |
| LOW_POSITION | valve position to have the 25% of the maximum flow |
| CLOSED_POSITION | valve position to have the 0% of the maximum flow |
| HIGH_THRESHOLD | 2 Celsius ° |
| APPROACHING_THRESHOLD | 1 Celsius ° |
| MIN_TEMPERATURE | 15 Celsius ° |
| MAX_TEMPERATURE | 30 Celsius ° |
| MIN_HUMIDITY | 0 Celsius ° |
| MAX_HUMIDITY | 100 Celsius ° |

Table 1: Display Information

# 4 SySML Functional model

In the picture 1 is reported the functional Block Definition Diagram that describes the composition of the system, composed by one Central Unit and up to eight Rooms, the two modules are connected via two FlowPort as shown in 2. The Central Unit send a *RoomRequest* message composed as follows:

| **parameter** | **type** | **[Min,Max]** |
|---|---|---|
| Id | Natural | [1,8] |
| DesiredTemperature | Float | [15.00, 30.00] |

Table 2: Room Request variables

The Room module send a *RoomStatus* message composed as follow:

| **parameter** | **type** | **[Min,Max]** |
|---|---|---|
| Id | Integer | [1,8] |
| Eco | Boolean | [0, 1] |
| Temperature | Float | [15.00, 30.00] |
| Humidity | Float | [0.00, 100.00] |
| Valve | Integer | [0, 100] |

Table 3: Room Status variables

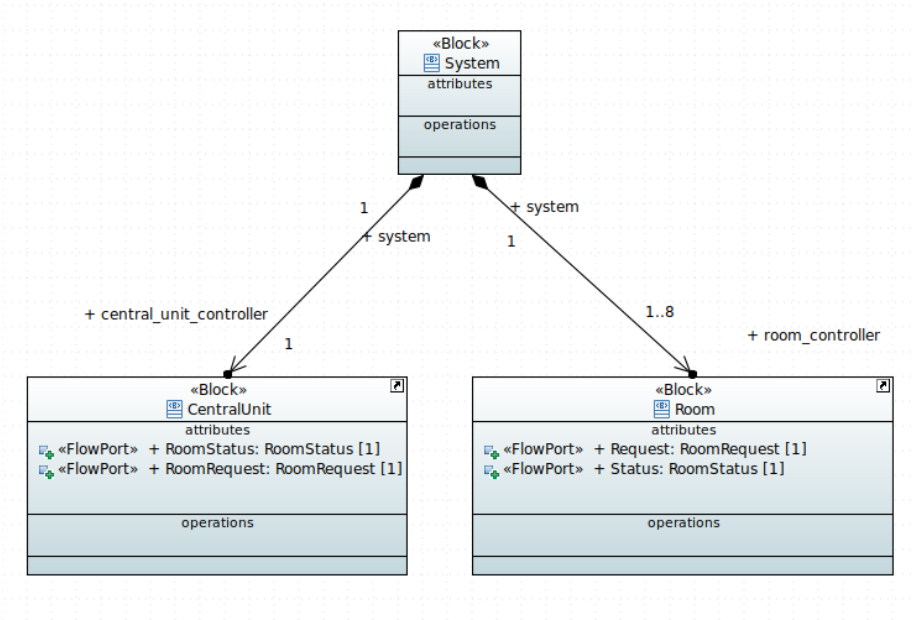


Figure 1: System Components

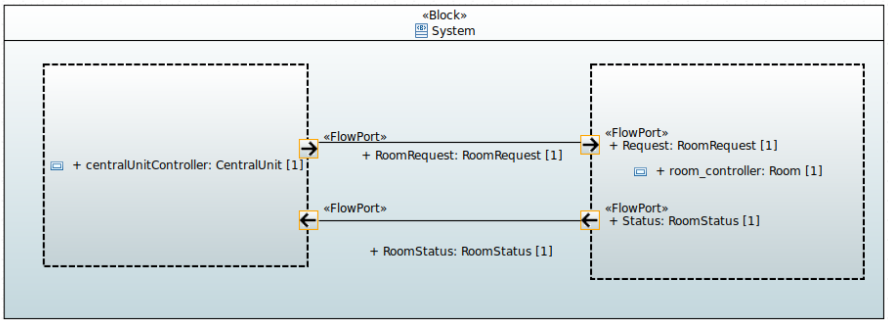


Figure 2: System Internals

## 4.1 Central Unit

The *Central Unit* is composed by two modules, the *RoomsManager* and the *UserInterfaceManager*. The *RoomsManager* implements the functionalities related to the status of each room. The *UserInterfaceManager* that implements the functionalities related to represent the status of the system. The two components exchange data as shown in 4.

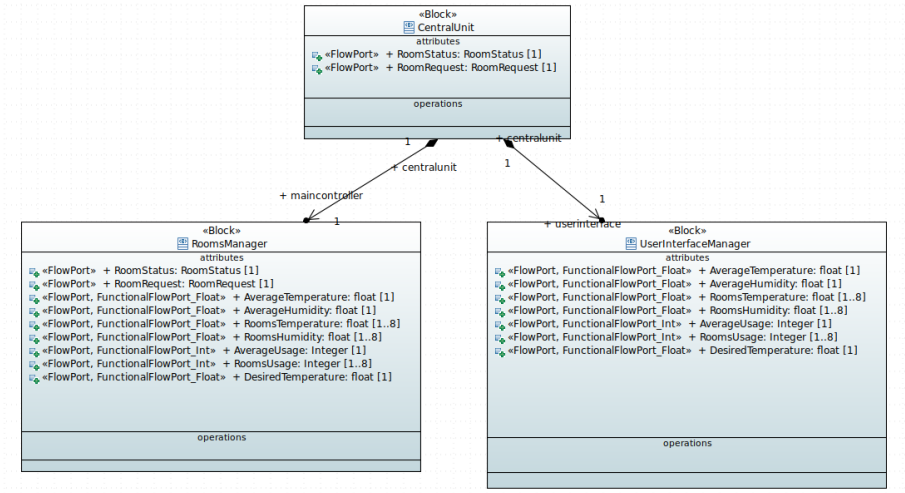


Figure 3: Central Unit components

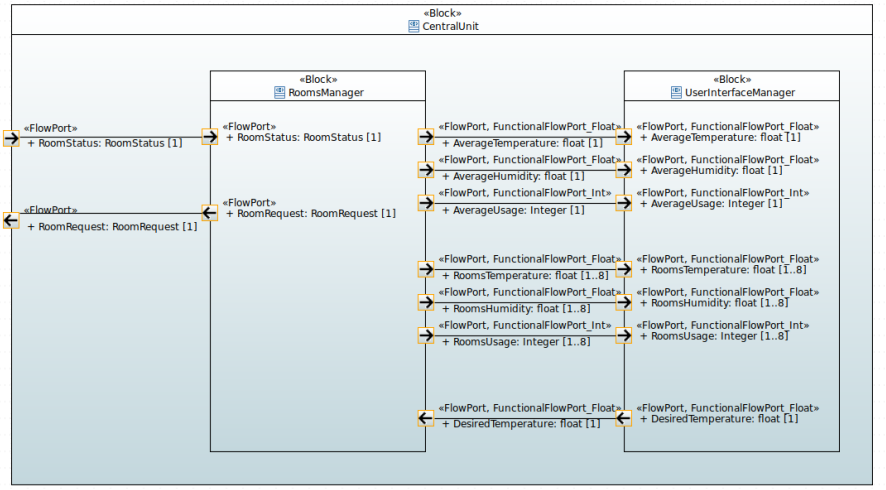


Figure 4: Central Unit internals

## 4.2 Room module

The main component of this module is the *MainController* composed by different functions as shown in 6.

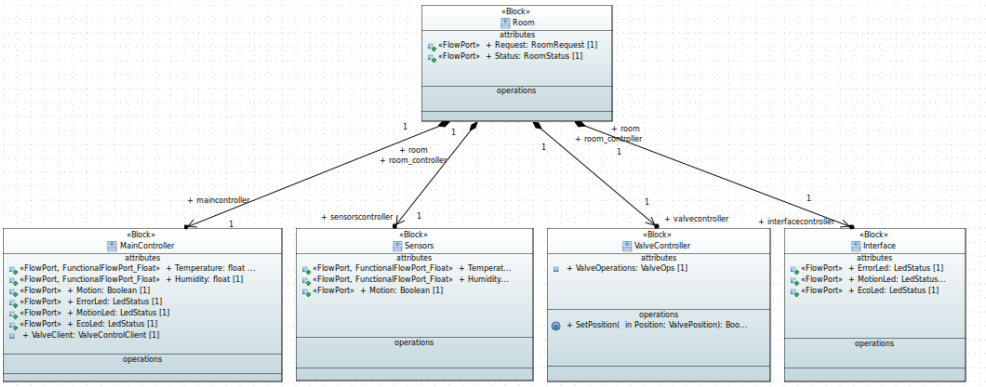


Figure 5: Room Components

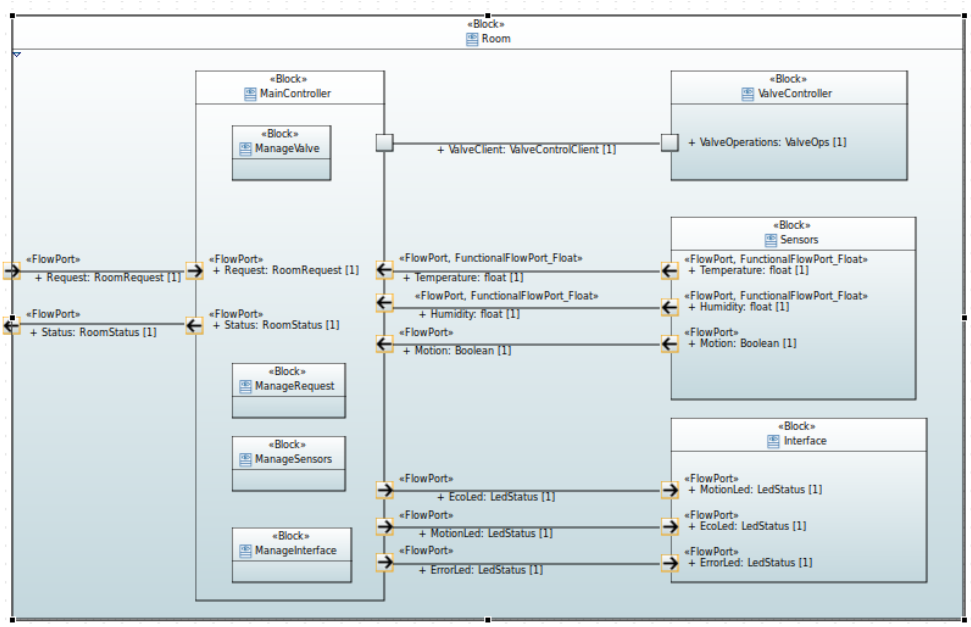


Figure 6: Room Internals

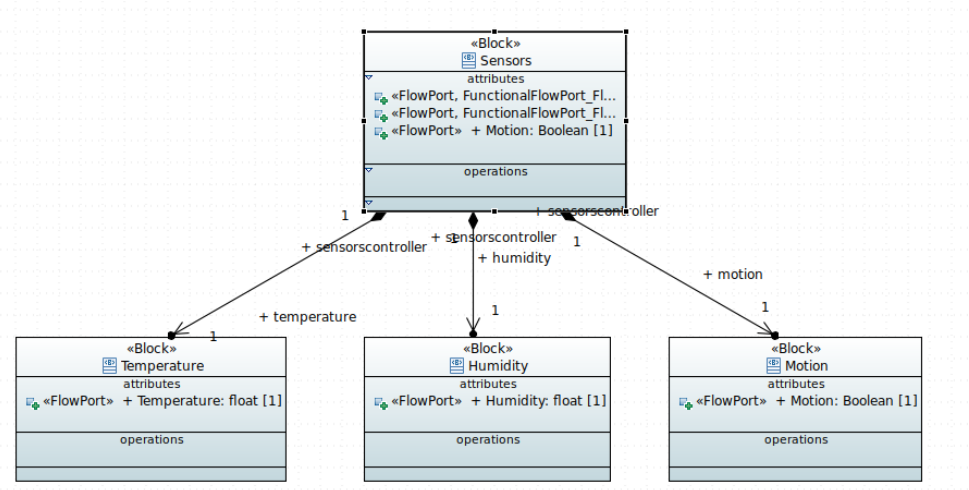


Figure 7: Room sensors components

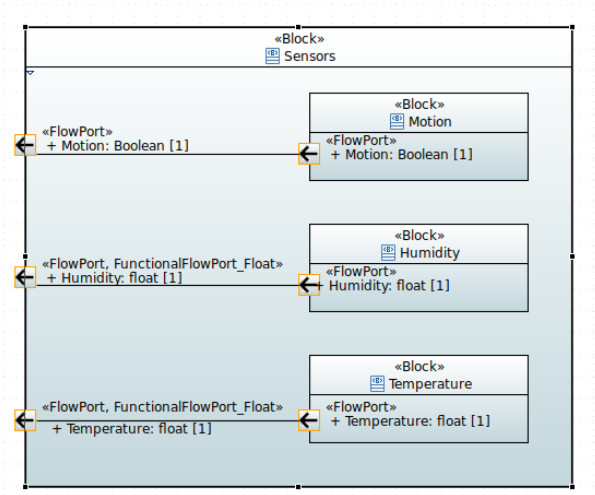


Figure 8: Room sensors internals

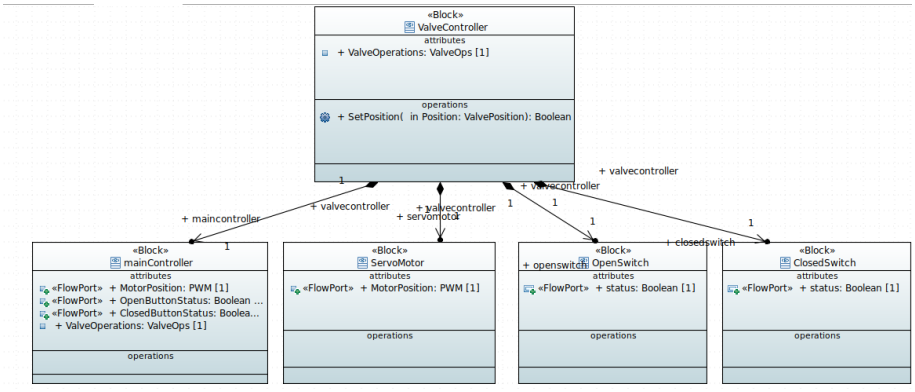


Figure 9: Valve Controller components

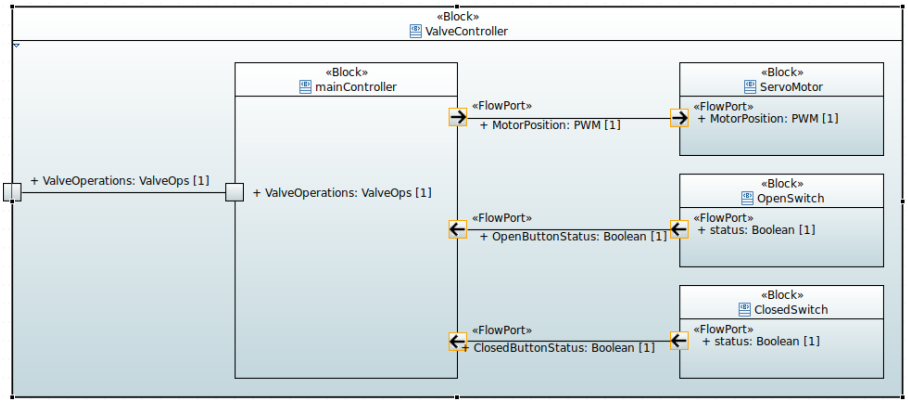


Figure 10: Valve Controller internals