



# 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 27, 2019

# Contents

# 1 Introduction

The purpose of this document is to describe the *Functional requirements*, the *composition* and *behaviour* of each component of the project in order to have a clear description of what the system shall do and which will be the tests to be executed in order to check the correct behaviour of the system respect to the described model.

# 2 Central Unit Requirements

## 2.1 Data Dictionary

### 2.1.1 Events

| Signal Name | Description | Direction | Trigger | Data Type | Min | Max | Unit |
|---|---|---|---|---|---|---|---|
| B_NEXT | Next page button | input | rising | Boolean | 0 | 1 | |
| B_PREVIOUS | Previous page button | input | rising | Boolean | 0 | 1 | |
| B_SETTINGS | Settings button | input | rising | Boolean | 0 | 1 | |
| B_PLUS | Plus button | input | rising | Boolean | 0 | 1 | |
| B_MINUS | Minus button | input | rising | Boolean | 0 | 1 | |
| PollingRoomId | Id of the room displayed | Output | | Real Positive | 1 | 8 | |
| DesiredTemperature | Desired Temperature set by the user | Output | | Real Positive | 15.00 | 30.00 | Celsius° |
| RoomId | Id in incoming Room Status message | Input | | Real Positive | 1 | 8 | |
| RoomTemperature | Temperature in incoming Room Status message | Input | | Real Positive | 15.00 | 30.00 | Celsius° |
| RoomHumidity | Humidity in incoming Room Status message | Input | | Real Positive | 0.00 | 100.00 | % |
| RoomValve | Valve position in incoming Room Status message | Input | | Natural | 0 | 100 | % |
| RoomEco | Eco status in incoming Room Status message | Input | | Boolean | 0 | 1 | |

### 2.1.2 Parameters

| Data | Description | Data Type | Min | Max | Unit | Default |
|---|---|---|---|---|---|---|
| POLLING_PERIOD | period for requesting room' status | Real Positive | 5 | 60 | Seconds | 10 |
| BuildingTemperature | Average Temperature of the building | Real Positive | 0 | 1 | Celsius° | 0 |
| BuildingHumidity | Average Humidity of the building | Real Positive | 0.00 | 100.00 | % | 0 |
| BuildingUsage | Average Usage of the building | Natural | 0 | 100 | % | 0 |
| BuildingEco | Eco status of the building | Boolean | 0 | 1 | | 0 |
| BuildingWarning | Warning status of the building | Boolean | 0 | 1 | | 0 |
| SelectedRoomId | Id of the displayed room | Natural | 1 | 8 | | 0 |

## 2.2 Functional Requirements

### 2.2.1. Graphical User Interface

#### 2.2.1.1. Main Page

2.2.1.1.1. If at least one room is set to **Energy Saving mode** then the module shall set to true the *BuildingEco* false otherwise

2.2.1.1.2. If at least one room is marked as **crashed** the module shall set to true the *BuildingWarning* false otherwise

2.2.1.1.3. If the *B_NEXT* event is set the module shall move in the *Room page* and set the *SelectedRoomId* to the lowest Id among the initialized rooms, if no rooms are available then shall move in the *Main page*

2.2.1.1.4. If the *B_PREVIOUS* event is set the module shall move in the *Room page* and set the *SelectedRoomId* to the greatest Id among the initialized rooms, if no rooms are available then shall move in the *Main page*

2.2.1.1.5. If the *B_SETTINGS* event is set the module shall move in the *Settings page*

2.2.1.1.6. The module shall represent the *BuildingTemperature* in Celsius°

2.2.1.1.7. The module shall represent the *BuildingHumidity* in %

2.2.1.1.8. The module shall represent the *BuildingUsage* in %

2.2.1.1.9. The module shall represent the *BuildingEco* and *BuildingWarning* as boolean

2.2.1.2. **Settings Page**

2.2.1.2.1. The module shall allow the user to increase or decrease the *DesiredTemperature*

2.2.1.2.2. If at least one room is set to **Energy Saving mode** then the module shall set to true the *BuildingEco* false otherwise

2.2.1.2.3. If at least one room is marked as **crashed** the module shall set to true the *BuildingWarning* false otherwise

2.2.1.2.4. If the *B_NEXT* event is set the module shall move in the *Room page* and set the *SelectedRoomId* to the lowest Id among the initialized rooms

2.2.1.2.5. If the *B_PREVIOUS* event is set the module shall move in the *Room page* and set the *SelectedRoomId* to the greatest Id among the initialized rooms

2.2.1.2.6. If the *B_SETTINGS* event is set the module shall move in the *Main page*

2.2.1.2.7. If the *B_PLUS* event is set the module shall increase the *DesiredTemperature* by a factor of 0.5 Celsius°if it is not exceeding the MAX_TEMPERATURE

2.2.1.2.8. If the *B_MINUS* event is set the module shall decrease the *DesiredTemperature* by a factor of 0.5 Celsius°if it is not less then MIN_TEMPERATURE

2.2.1.2.9. The module shall represent the *DesiredTemperature* in Celsius°

2.2.1.2.10. The module shall represent the *BuildingHumidity* in %

2.2.1.2.11. The module shall represent the *BuildingUsage* in %

2.2.1.2.12. The module shall represent the *BuildingEco* and *BuildingWarning* as boolean

2.2.1.3. **Room Page**

2.2.1.3.1. If the *B_NEXT* event is set the module shall move in the *Room page* and set the *SelectedRoomId* to the next greater Id among the initialized rooms, if no rooms are available then shall move

in the *Main page*

2.2.1.3.2. If the *B_PREVIOUS* event is set the module shall set the *SelectedRoomId* to the previous Id among the initialized rooms, if no rooms are available then shall move in the *Main page*

2.2.1.3.3. If the *B_SETTINGS* event is set the module shall move in the *Settings page*

2.2.1.3.4. The module shall represent the Temperature of the *SelectedRoomId* in Celsius°

2.2.1.3.5. The module shall represent the Humidity of the *SelectedRoomId* in %

2.2.1.3.6. The module shall represent the Usage of the *SelectedRoomId* in %

2.2.1.3.7. The module shall represent if the *SelectedRoomId* is in **Energy saving mode** or **Normal mode**

2.2.1.3.8. The module shall represent wheter the *SelectedRoomId* is considered **Crashed** or not

### 2.2.2. Communication managemnet

#### 2.2.2.1. Entry

2.2.2.1.1. The module shall send the *InitialMessage* in broadcast

2.2.2.1.2. The module shall set all the rooms as uninitialized

#### 2.2.2.2. During

2.2.2.2.1. The *Central Unit* shall send a *Room Request message* including the *PollingRoomId* and the *DesiredTemperature* in Celsius°every POLLING_PERIOD ±1 second cycling between all the initialized rooms

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

2.2.2.2.3. The module shall check the correctness of the *Room Status message* and the consistency of each parameter, if the *Room Status message* is not valid shall be ignored

2.2.2.2.4. Whenever a *Room Status message* doesn't arrive within *POLLING_PERIOD* ±1 seconds from the sent of the last *Room Request message*, the same *Room Request message* shall be resent at least 3 times before marking the room as **crashed**

2.2.2.2.5. Whenever a *Room Status message* arrives and it is valid the module shall update the *Room Status* and *Building Status*, if the *room* associated to the *RoomId* of the incoming *Room Status message* is uninitialized, the module shall initialize it

# 3 Room Requirements

## 3.1 Room Data Dictionary

### 3.1.1 Events

| Signal Name | Description | Direction | Trigger | Data Type | Min | Max | Unit |
|---|---|---|---|---|---|---|---|
| OPEN_SWITCH | 1 when the valve is open | Input | rising | Boolean | 0 | 1 | |
| CLOSED_SWITCH | 1 when the valve is closed | Input | rising | Boolean | 0 | 1 | |
| motion | 1 when a motion is detected | Input | rising | Boolean | 0 | 1 | |
| Temperature | Temperature from sensors | Input | | Real Positive | 0 | 1 | Celsius° |
| Humidity | Humidity from sensors | Input | | Real Positive | 0.00 | 100.00 | % |
| ValvePosition | position of the valve | Output | | Natural | 10 | 160 | ° |
| PollingRoomId | Id of the room displayed | Input | | Real Positive | 1 | 8 | |
| DesiredTemperature | Desired Temperature set by the user | Input | | Real Positive | 15.00 | 30.00 | Celsius° |
| RoomId | Id of the room | Output | | Real Positive | 1 | 8 | |
| RoomTemperature | Temperature of the room | Output | | Real Positive | 15.00 | 30.00 | Celsius° |
| RoomHumidity | Humidity of the room | Output | | Real Positive | 0.00 | 100.00 | % |
| RoomUsage | Usage of the heating in % | Output | | Natural | 0 | 100 | % |
| EcoMode | Eco status of the building | Output | | Boolean | 0 | 1 | |

### 3.1.2 Parameters

| Data | Description | Data Type | Min | Max | Unit | Default |
|---|---|---|---|---|---|---|
| MOTION_TIMESLOT | Period of time to consider the last motion for energy saving calculations | Real Positive | 1 | 60 | Seconds | 30 |
| TEMPERATURE_PERIOD | Period of time to read the temperature | Real Positive | 2 | 60 | Seconds | 2 |
| HUMIDITY_PERIOD | Period of time to read the humidity | Real Positive | 2 | 60 | Seconds | 2 |
| VALVE_PERIOD | Period of time to set the valve | Real Positive | 2 | 120 | Seconds | 4 |
| COMMUNICATION_DEADLINE | Relative time from last received request to send againg the status | Real Positive | 30 | 3600 | Seconds | 60 |
| OPEN_POSITION | preconfigured position of the valve | Real Positive | 0 | 180 | ° | 170 |
| HIGH_POSITION | preconfigured position of the valve | Real Positive | 0 | 180 | ° | 135 |
| MIDDLE_POSITION | preconfigured position of the valve | Real Positive | 0 | 180 | ° | 90 |
| LOW_POSITION | preconfigured position of the valve | Real Positive | 0 | 180 | ° | 45 |
| CLOSED_POSITION | preconfigured position of the valve | Real Positive | 0 | 180 | ° | 10 |
| HIGH_THRESHOLD | relative temperature offset to compute valve position | Real Positive | 0 | 10 | C° | 2 |
| APPROACHING_THRESHOLD | relative temperature offset to compute valve position | Real Positive | 0 | 5 | C° | 1 |
| GoalTemperature | goal temperature to control the valve | Real Positive | 15.00 | 30.00 | C° | 24.00 |
| TemperatureEcoOffset | offset used to reduce the desired temperature | Real Positive | 0.00 | 5.00 | C° | 2.00 |

## 3.2 Functional Requirements

3.2.1. **Energy Saving management**

3.2.1.1. **Entry**

3.2.1.1.1. The module shall start working in **Normal Mode**

3.2.1.2. **During each mode**

3.2.1.2.1. The module shall read and update the temperature every TEMPERATURE_PERIOD ±1 second

3.2.1.2.2. The module shall read and update the temperature every HUMIDITY_PERIOD ±1 second

3.2.1.2.3. If the read *Temperature* or the *Humidity* is not consistent then the module shall turn on the ERROR_LED

3.2.1.2.4. Whenever a motion is detected the module shall turn on the MOTION_LED and off when it is not

3.2.1.3. **Normal Mode**

3.2.1.3.1. **Entry**

3.2.1.3.1.1. The module shall set the *GoalTemperature* to the *DesiredTemperature*

3.2.1.3.2. **During**

3.2.1.3.2.1. If in the last MOTION_TIMESLOT seconds ±1 seconds no motion has been detected the module shall move in **Normal Mode**

3.2.1.4. **Energy Saving Mode**

3.2.1.4.1. **Entry**

3.2.1.4.1.1. The module shall set the *GoalTemperature* to the *DesiredTemperature* minus the *TemperatureEcoOffset*

3.2.1.4.1.2. The module shall turn on the ENERGY_SAVING_LED

3.2.1.4.2. **During**

3.2.1.4.2.1. If in the last MOTION_TIMESLOT seconds ±1 seconds at least one motion has been detected then the module shall move in **Normal Mode**

3.2.1.4.3. **Exit**

3.2.1.4.3.1. The module shall turn off the ENERGY_SAVING_LED

3.2.2. **Communication management**

3.2.2.1. **Entry**

3.2.2.1.1. The module shall send the *Room Status message* and start working in **Normal Mode**

3.2.2.2. **During each mode**

3.2.2.2.1. The incoming *Room Request message* must include the *RoomId* and the *DesiredTemperature*

3.2.2.2.2. Whenever a *Room Request message* arrives and it is not corrupted the module shall update the *DesiredTemperature* with the desired temperature in the message and shall send the *Room Status message*

3.2.2.3. **Normal Mode**

3.2.2.3.1. **During**

3.2.2.3.1.1. If the *Room Request message* does not arrive within the last COMMUNICATION_DEADLINE seconds ±2 seconds from the last *Room Request message* then the module shall move in **Error Mode**

3.2.2.4. **Error Mode**

3.2.2.4.1. **Entry**

3.2.2.4.1.1. The module shall send the *Room Status message*

3.2.2.4.1.2. The module shall turn on the ERROR_LED

3.2.2.4.2. **During**

3.2.2.4.2.1. Whenever a *Room Request message* arrives and it is not corrupted the module shall update the *DesiredTemperature* with the desired temperature in the message and move in **Normal Mode**

3.2.2.4.2.2. If the *Room Request message* does not arrive within the last COMMUNICATION_DEADLINE seconds ±2 seconds from the last *Room Request message* then the module shall send the *Room Status message*

3.2.2.4.3. **Exit**

3.2.2.4.3.1. The module shall turn off the ERROR_LED

3.2.3. **Control Valve management**

3.2.3.1. **Entry**

3.2.3.1.1. The module shall check the OPEN_POSITION and CLOSED_POSITION and then compute the HIGH_POSITION, MIDDLE_POSITION and LOW_POSITION

3.2.3.1.2. The module start working in **Normal Mode**

3.2.3.2. **Normal Mode**

3.2.3.2.1. **During**

3.2.3.2.1.1. The module shall check and move the position of the valve every VALVE_PERIOD seconds ±1 second

3.2.3.2.1.2. The valve shall be in OPEN_POSITION whenever the difference between the *Temperature* and the *GoalTemperature* is below -HIGH_THRESHOLD C°

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

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

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

3.2.3.2.1.6. The valve shall be in CLOSED_POSITION whenever the difference between the *Temperature* and the *GoalTemperature* is greater then HIGH_THRESHOLD C°

3.2.3.2.1.7. 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 and shall move in **Error Mode** if it is not consistent

3.2.3.3. **Error Mode**

3.2.3.3.1. **Entry**

3.2.3.3.1.1. The module shall turn on the ERROR_LED

3.2.3.3.2. **During**

3.2.3.3.2.1. The module shall move the valve in the opening direction until the OPEN_SWITCH is set and update the OPEN_POSITION with the new position then shall move the valve in the closing direction until the CLOSED_SWITCH is set and update the CLOSED_POSITION with the new position then shall compute and update the MIDDLE_POSITION, LOW_POSITION and HIGH_POSITION and shall move in **Normal Mode**

3.2.3.3.3. **Exit**

3.2.3.3.3.1. The module shall turn off the ERROR_LED

# 4 SysML Functional model

In the figure 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 1: 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 2: Room Status variables

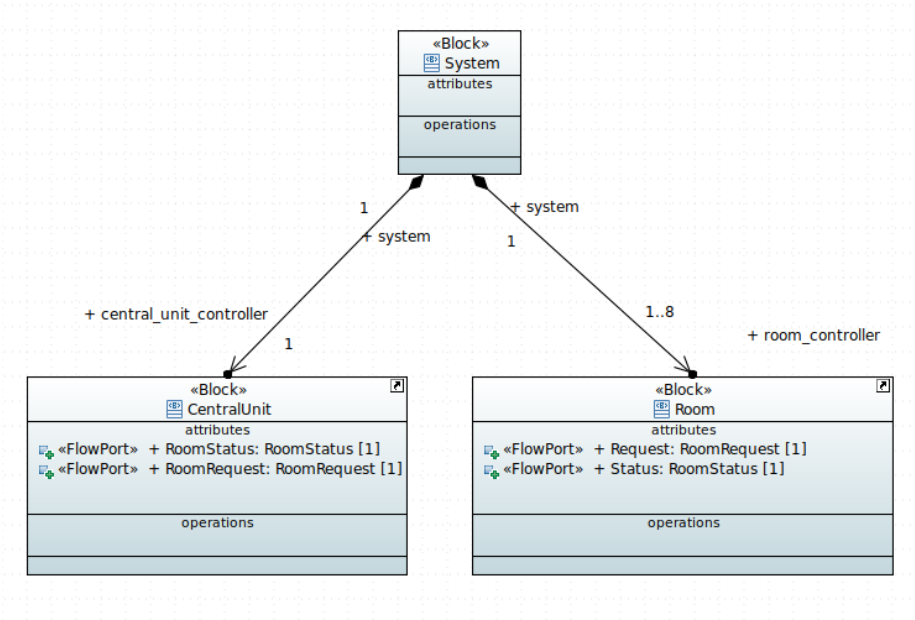


Figure 1: System Components

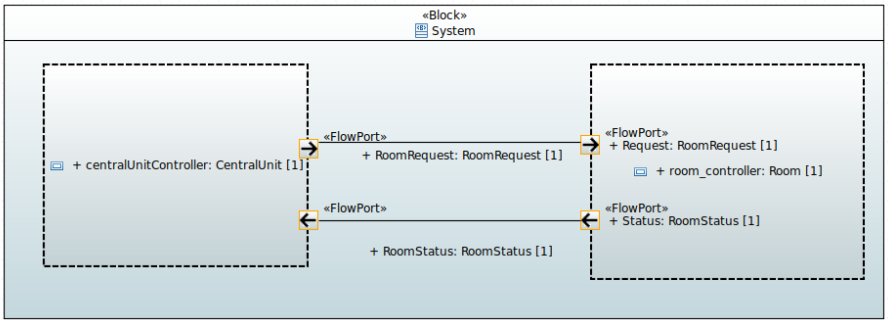


Figure 2: System Internals

## 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.

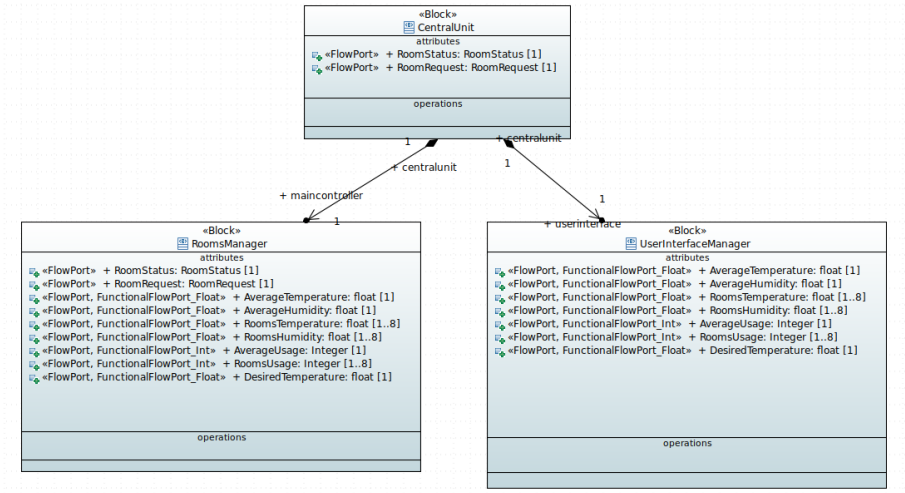


Figure 3: Central Unit components

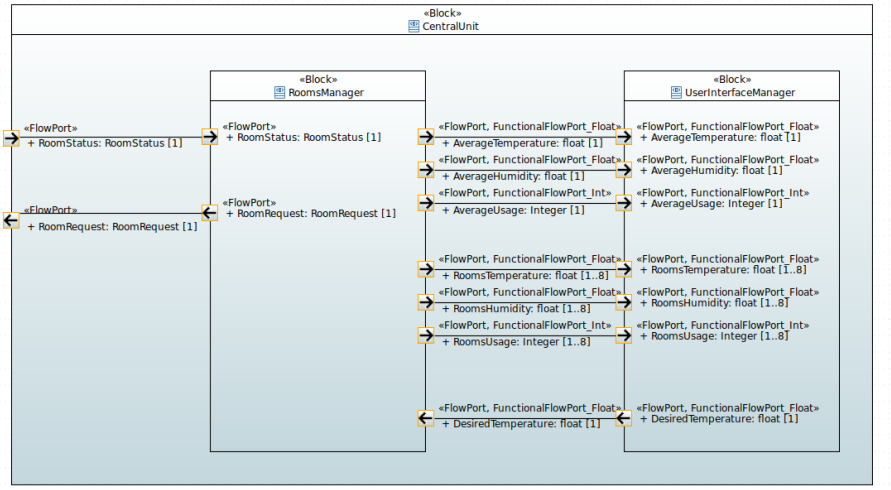


Figure 4: Central Unit internals

## 4.2 Room module

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

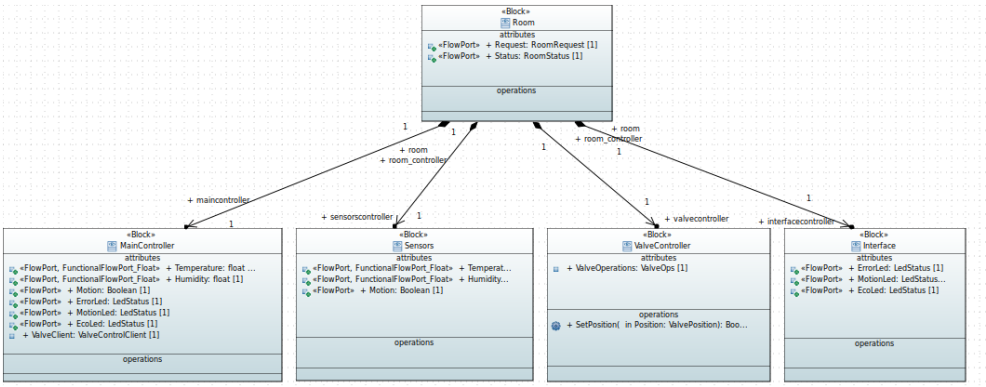


Figure 5: Room Components

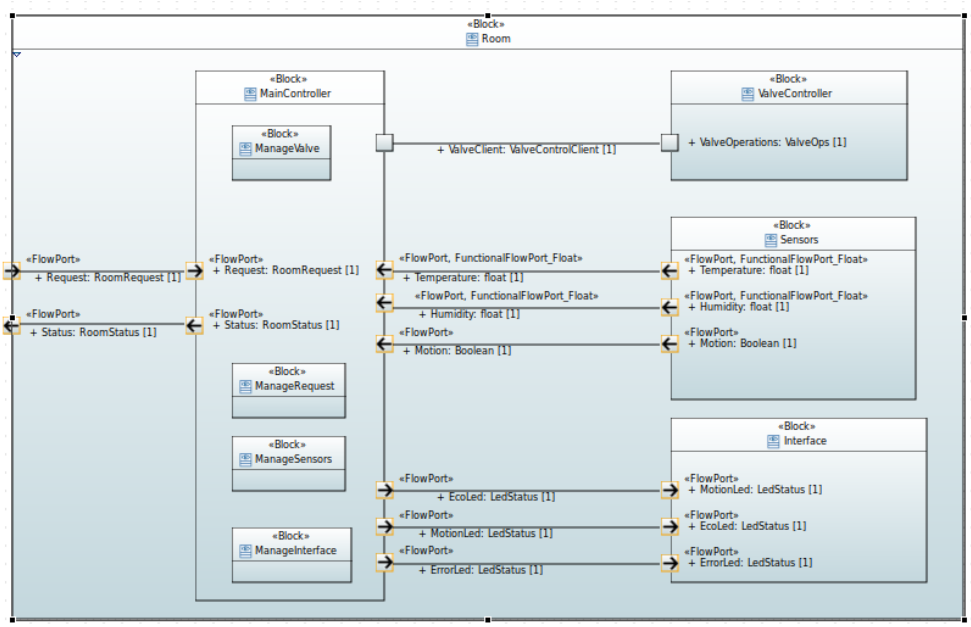


Figure 6: Room Internals

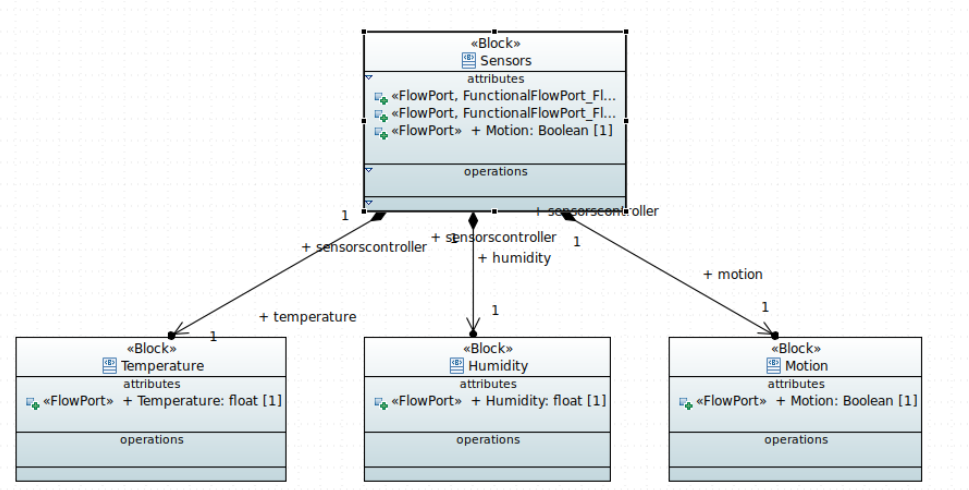


Figure 7: Room sensors components

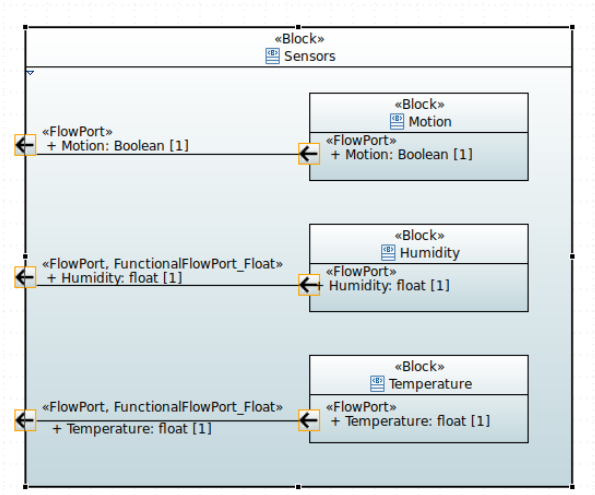


Figure 8: Room sensors internals

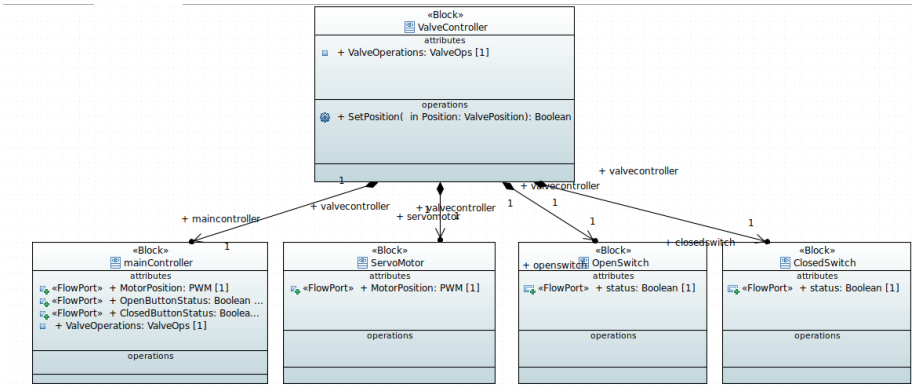


Figure 9: Valve Controller components

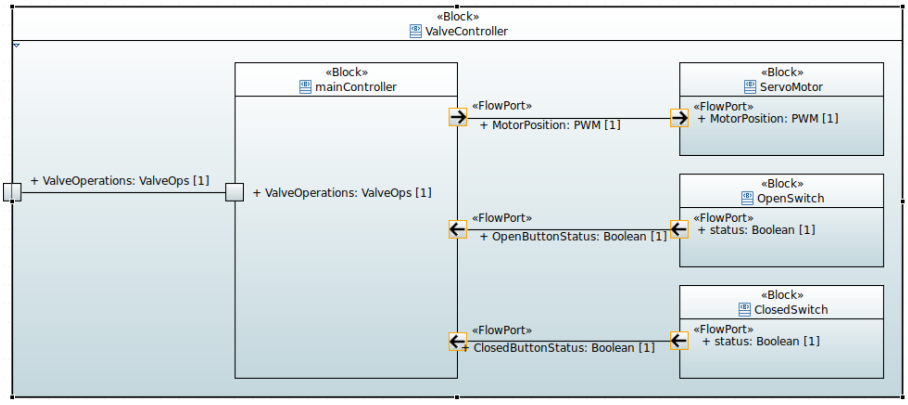


Figure 10: Valve Controller internals

## 4.3 Communication State Machines

In the following pictures are illustrated the behaviour of the communication between the *CentralUnit* and the *Room*.

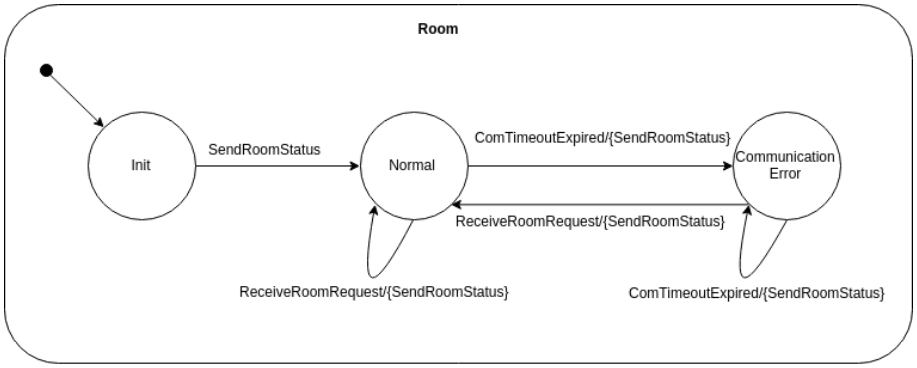


Figure 11: Room communication management

In the figure 12 is described the behaviour of the receiver part in the *CentralUnit*, for readibility is reported just the case of a paramentric room X.

In the figure 13 is described the behaviour of the sender part in the *CentralUnit*, for readibility is reported just the case of 2 rooms.

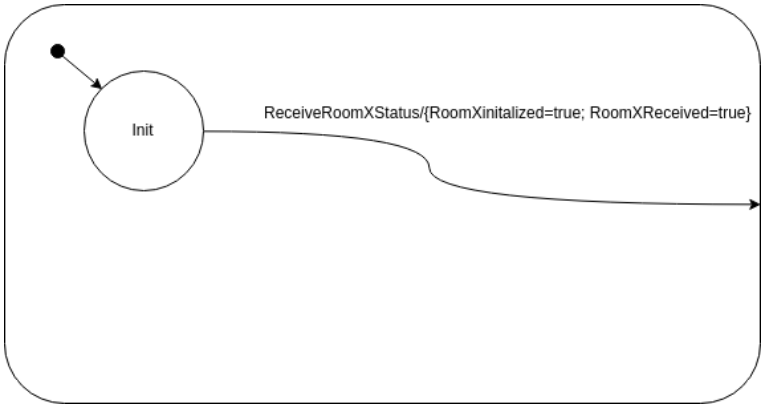


Figure 12: Central Unit communication receiver management

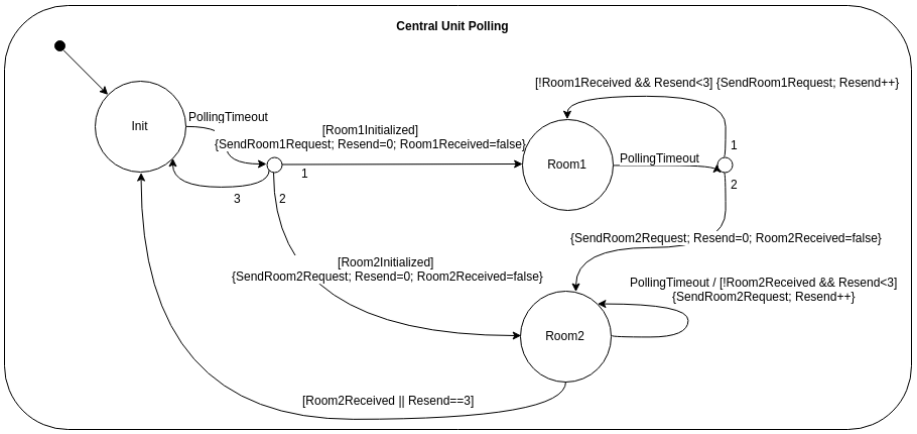


Figure 13: Central Unit communication sender management