IOT PROJECT

Filippo Minutella

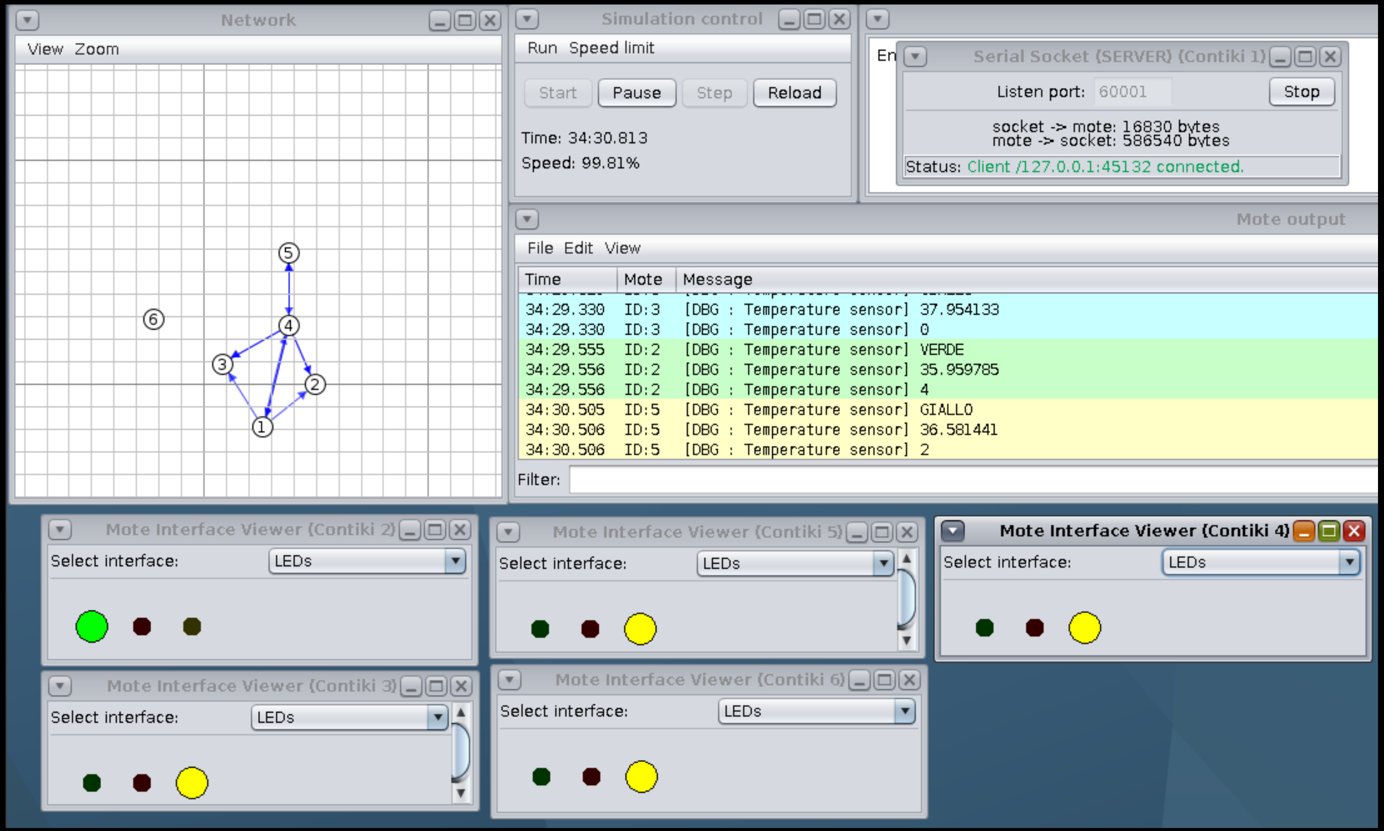
PURPOSE OF THE PROJECT

- The purpose is to monitor and control the temparute in buildings, this can be useful in many use cases, for instance:
  - Monitoring of data centers
  - Monitoring of cold rooms and so on

- Each node used to monitor the temperature has two resources:
  - A sensor used to observe the temperature
  - An actuator used to cool the air

CONTIKI SIMULATION

- Each node has three led:
  - GREEN: temperature below to given threshold
  - YELLOW: temperature above the given threshold but below than threshold * 1.01
  - RED: temperature is very high, above the 110% of the threshold


Network
View Zoom
1
2
3
4
5
6
Simulation control
Run Speed limit
Start
Pause
Step
Reload
En
Serial Socket (SERVER) (Contiki 1)
Stop
Mote output
File Edit View
Mote Interface Viewer (Contiki 2)
Mote Interface Viewer (Contiki 5)
Mote Interface Viewer (Contiki 4)
Mote Interface Viewer (Contiki 3)
Mote Interface Viewer (Contiki 6)

Time: 34:30.813
Speed: 99.81%

Listen port: 60001

socket -> mote: 16830 bytes
mote -> socket: 586540 bytes

Status: Client /127.0.0.1:45132 connected.

```
Time       Mote  Message
34:29.330  ID:3  [DBG : Temperature sensor] 37.954133
34:29.330  ID:3  [DBG : Temperature sensor] 0
34:29.555  ID:2  [DBG : Temperature sensor] VERDE
34:29.556  ID:2  [DBG : Temperature sensor] 35.959785
34:29.556  ID:2  [DBG : Temperature sensor] 4
34:30.505  ID:5  [DBG : Temperature sensor] GIALLO
34:30.506  ID:5  [DBG : Temperature sensor] 36.581441
34:30.506  ID:5  [DBG : Temperature sensor] 2
```

Filter:

Select interface: LEDs

Select interface: LEDs

Select interface: LEDs

Select interface: LEDs

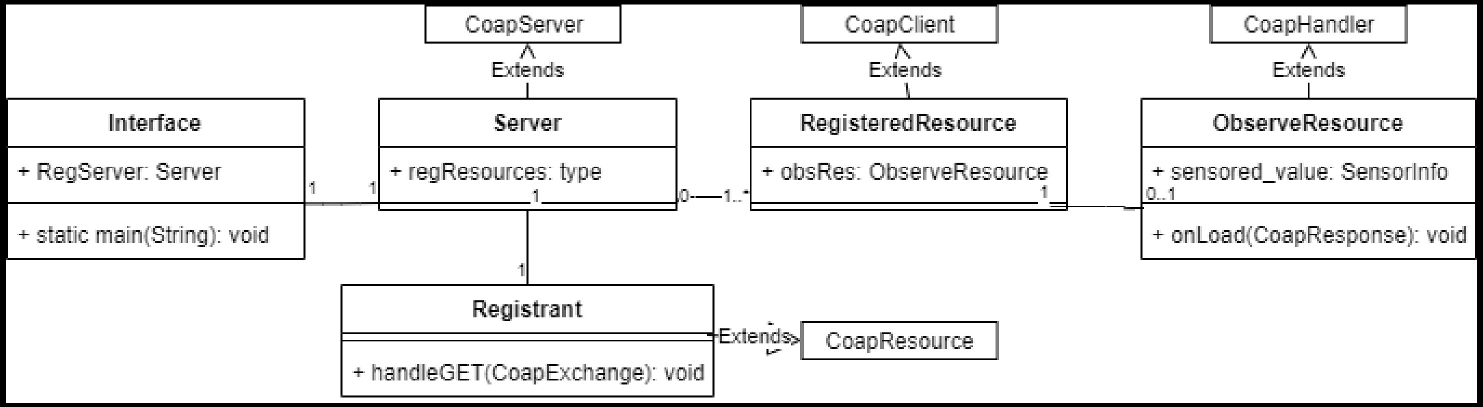
Select interface: LEDs

THE INTERFACE

- The actuator receives a given threshold in which the sensored temperature should be below, so automatically the actuator start cooling when the temperature is above the threshold.
- The operations permitted are:
  - Get the current temperature by one or all sensors
  - Get the current info by one or all actuators
  - Set the threshold to one or all actuators
  - See historic sensored info of one sensor
- There is also a botton on the node that tells the actuators to cooling while the button is pressed

CORE JAVA PROJECT: AN OVERVIEW TO THE PROJECT STRUCTURE

- Interface: The entry point of the program that display the interface and start the Server
- Server started in a thread handles the registering requests using the Registrant resource
- Registrant registers the CoapClient (sensors or actuators) as RegisteredResource and retrieve the information issuing a request to .well-known/core
- RegisteredResource is a CoapClient to the resource and instantiates the CoapHandler to the resource that have to be observed



CoapServer

Extends

CoapClient

Extends

CoapHandler

Extends

Interface

+ RegServer: Server

+ static main(String): void

1

1

Server

+ regResources: type

1

0

1..*

RegisteredResource

+ obsRes: ObserveResource

1

0..1

ObserveResource

+ sensored_value: SensorInfo

+ onLoad(CoapResponse): void

1

Registrant

+ handleGET(CoapExchange): void

Extends

CoapResource

THE NODE AND ITS TWO RESOURCES

- The core part of the node is composed by 3 files:
  - cool_node.c is the entry point, it starts the protothread, instatiates and activates the two resources, trigger the sensor and listens for the button
  - res_temperature.c is the sensor resource, it register the temperature and it is an observable resource
  - res_cooler.c is the actuator resource, it expose a post request to set the max temperature in which the node must starts to cool the air