Final Project

Sistemas Embebidos em Rede

Contiki based Wireless Sensor Network for server room temperature control

Group 4:

81866	Bruno Figueiredo
81007	Francisco Natário
68557	Pedro Costa

Conteúdo

1 | Introduction

In Networked Embedded Systems, a wireless sensor network is an integration of low cost-devices responsible for sensing, communicating and processing information allowing the creation of a plethora of applications. In this wireless sensor network project, we will be focusing in a environment sensing and control application for a data center. The main focus of our project is the physical environment inside the data center, specifically the temperature and our main objective will be controlling the temperature surrounding the racks of servers. For this reason a wireless embedded system will give us the opportunity to interact with the environment by sensing the the temperature close to the racks, gathering and storing this information by sending it to a control hub that after processing the information will then communicate the instruction needed to maintain the right temperature in the data center to an actuator. The actuator will then apply the right temperature to the room.

The choice of a wireless sensor networking for this kind of project was taken into account. A wireless sensor network allow the sensors to be close as possible of the racks in the server, which means closer to the servers that need to be cool down. The temperature close to the racks is the most important to keep track of. Research shows that ”more than 23% of data center outages are caused by servers’ self-protective shutdowns because of overheating”. Self-protective shutdowns and high temperatures are also related to most hardware malfunction and deterioration. For this reason temperature must be monitored and controlled in order for a data center to operate effectively, [6] [7].

Due to high temperature created by the functional servers, the cooling systems consume excessive power. In order to avoid consuming excessive power and maintain proper server room conditions, it is necessary to verify the airflow and temperature circulating through the data and telecommunication racks, plus the airflow in the room, cold air intake and hot air return. This is why it is necessary to monitor and actuate in many places throughout the room. Plus, Room temperature monitoring is recommended in addition to rack monitoring because it will allows to measure the incoming air temperature/humidity to compare to the rack mounted sensors in order to make adjustments.

This many sensors and actuators in many places and/or closer to the racks that have to be observed, bring an obstacle to wired communication. Furthermore, wired communications are expensive. Thus, wireless communication is the most reliable and the cheapest form of communication to ensure the environment control and success of this project.

There are different defined temperatures for the server operating ambient within the community. We made a choice to defined the temperature to be within the small range $18°$ to -$27°$C. and the room should be act upon the $15°$ to $27°$C mark, [4].

There are plenty of research over temperature control in data centers due to the complexity of the temporal evolution of temperature distribution, for instances: temperature differences according to server workload, the complex thermal and air dynamics and the heat dissipation that is highly dependent on the racks and other physical structures (figure 1.1 [4]) inside the data center, [7]. Thus, we will not endeavor into further complexity then its required for this project.

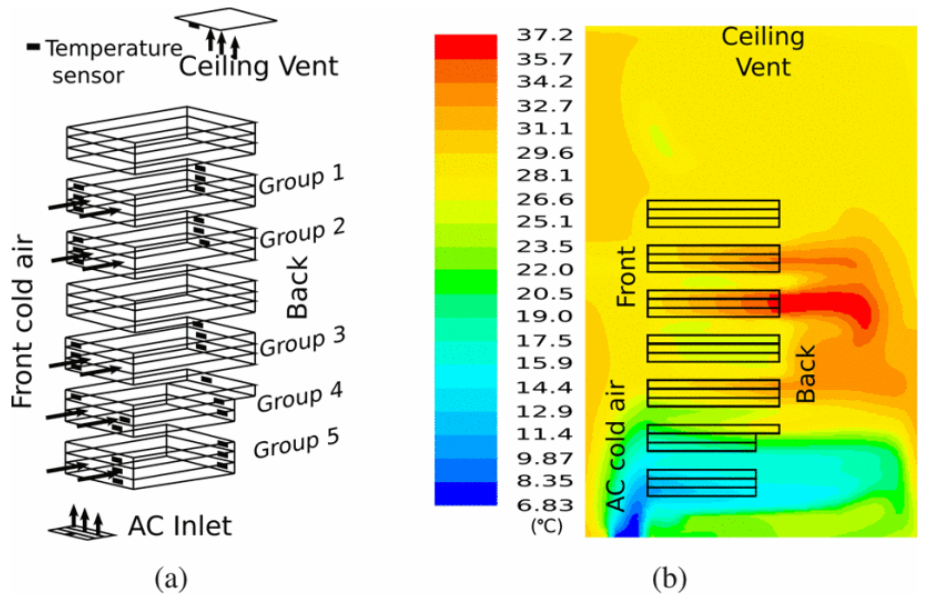


Figura 1.1: Data center:An example of the complexity of thermal and air dynamics in Rack ventilation

2 | Node and Network Architecture

Before choosing our network architecture we should know what equipment we need for our application. A simple scheme of a sensor node architecture is represented in the Figure 2.1.

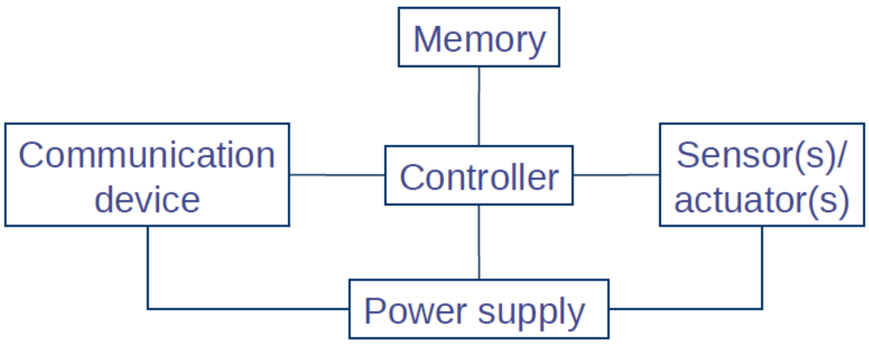


Figura 2.1: Sensor node architecture

In our case, if we weren't working in cooja, our mote would be using a 2.4 GHz IEEE 802.15.4 radio as a communication device and a temperature sensor.

Choosing the right approach in a network architecture is important to optimize the aspect that is more important in each application: quality of service, energy efficiency or scalability.

Our Wireless Sensor Network should be composed of multiple source nodes and a single sink node connected to the ac controller (Figure 2.2).

We chose a multi-hop network to not only facilitate scalability but to make it more robust in case of a node failure. In a server room the racks are relatively close so a node should be able to connect to multiple nodes.

Since our network is fixed we can also take advantage of the servers USB ports (5V) to supply power to our node. This reduces maintenance costs and possible power related node failures.

Each source node will be installed next to a server rack and do periodic measurements of the air temperature. When there is a change in temperature it sends its data to a nearby source or sink node.

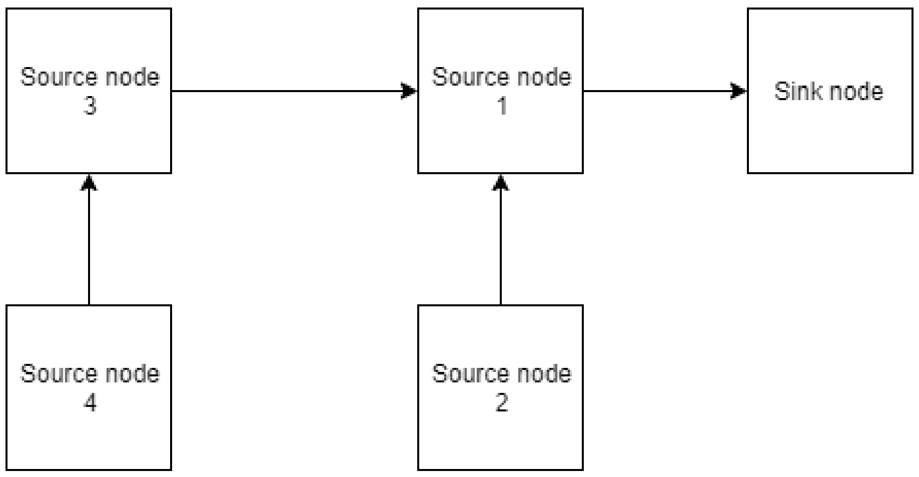


Figura 2.2: Network architecture.

The sink node then receives the data, processes it and sends commands to the ac controller if necessary.

This will be a node-centric network, with each node associated with a different zone to be refrigerated.

3 | System Support and Special Requirements

Since our project will run in the Cooja simulator, our nodes will be using the Contiki OS.

Concurrency is needed because the sensor node's CPU has to service the radio modem, the sensor, perform computaion for the application, etc.

In an event-based programming model, the CPU reacts to events when they happen immediately, like an interrupt handler. For it to work well we should not remain in the interrupt handler for too long.

With multi-threading we can have longer running computations, but there are larger memory requirements.

Contiki combines both ways, with an event-based kernel and multi-threading implemented as a library, but since our application shouldn't require long running computations it will be mostly event-based.

We should develop a console to monitor and alter temperatures for testing purposes. For that, we will use the serial port monitor of cooja, showing some sort of menu that allow us to check current temperatures and ac status, as well as changing temperatures of specific nodes.

We also think it is important that we implement a failsafe, detecting when there is a node failure and not triggering an alert flag.

4 | Sensors and Actuators

4.1 Selection Criteria

The project requires measurements of the air temperature inside the server's room and switch ON/OFF the cooling equipment, such as an AC system or cooling fans. There are a variety of options for temperature measurement, but it is important to choose the correct sensor that is best suited to this form of use. In this scenario, we'll need the following components for this project:

- Temperature sensors - Air temperature measurements, $0 - 70°C$ range, fast response to temperature variations, working voltages $3.3 - 5V$, need to be simple and cheap.
- Relays - modules for switch power on/off the cooling equipment. Need to be rated for 230V AC and, at least, 10A [5]. It is preferable to have a coil voltage lower than 5V (typical Voltage supply for an embedded system).

We started by selecting the type of sensors for this application: Thermocouples, Thermistors, RTD, and Infrared temperature sensors. The option of using Infrared sensors can be excluded because the air has a high transmittance, which means for an ambient room temperature needs to be measured indirectly, for example, an object has the same temperature as the involving air.

Since our main goal is having a good price-accuracy compromise and fast response times, the Thermistors and Thermocouples are the candidates. However, we still have other requirements such as operating voltage and high thermal sensitivity.

Inside the thermistors category, we have other sub-types: NTCs and PTCs. The difference is how the resistance varies with the temperature. The NTC's resistance will decrease with temperature and the resistance from PTCs will increase. Usually, the NTCs are used for temperature measurements and PTCs for circuit protection.

$$ln(\frac{R}{R_0}) = \beta(\frac{1}{T} - \frac{1}{T0}) \tag{4.1}$$

Sensor type	Advantages	Disadvantages
Thermocouples	Simple, Rugged Wide temperature ranges No external power	Small sensitivity, small voltage output Requires cold junction compesation Least stable
RTD	Most stable Good Linearity Accurate	Low sensitivity Costly Small output resistance Self-heating error
Thermistors	High sensitivity Can be cheap High output	Limited temperature range Nonlinear Self-heating error

Tabela 4.1: Comparision of advantages and disvantages from some types of temperature sensors.

The R_0 is the resistance at a reference temperature, T0 is the reference at temperature 25 degrees and β is a the material characteristic. The idea here is measuring the voltage drop at the Thermistor, knowing the voltage source value and the current that passes through it, to calculate the resistance value (and converting into temperature values). To minimize the load effect from the voltage source and calibrate the circuit output to a certain voltage reference, we can use a Wheatstone bridge 4.1.The balancing can be achieve using the equation (4.2).

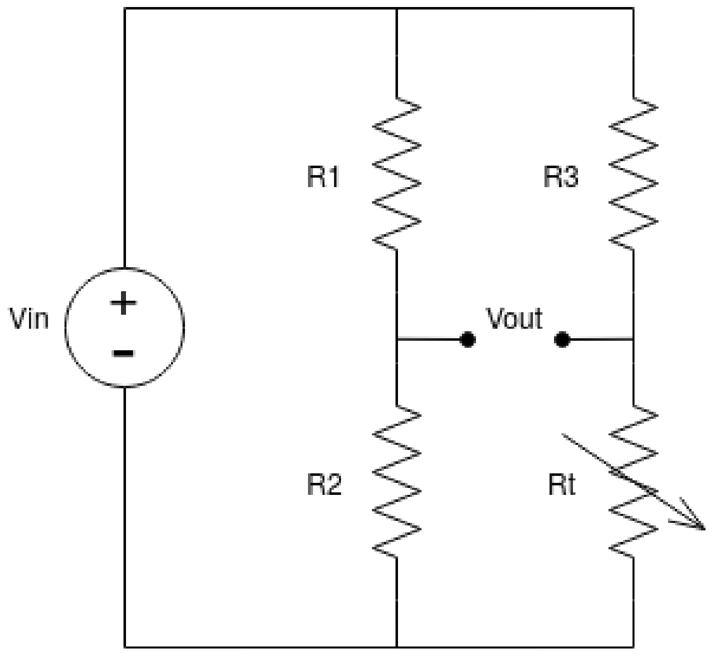


Figura 4.1: Wheatstone bridge circuit

$$\frac{R_1}{R_2} = \frac{R_3}{R_t} \tag{4.2}$$

Using a voltage divider with a thermistor the output voltage equation can be more simple but, for resistance measurements, the Wheatstone bridges are a better approach. The output voltage from the Wheatstone bridge using one NTC can be obtained by using equation (4.3):

$$Vout = Vin(\frac{R2}{R1 + R2} - \frac{R_t}{R3 + R_t}) \tag{4.3}$$

The resistors R1, R2, and R3 must have a low temperature coefficient and be exactly matched to guarantee accuracy. Also, will be required a resistance tolerance lower than 0.1%.

The particularity about this circuit is the fact of it can be designed with a cost of around 1 or 2 euros and, the current consumption can be less than 0.1mA. It's obvious, the overall circuit's accuracy will depend on the ADC resolution and sometimes is required the use of a buffer to give some gain and also increase the output impedance. An alternative to designing a circuit by ourselves is using a dedicated ICs to do the job. The **DS18B20** [1] IC is a good example of an alternative for this type of usage.

Component type	Reference	Price/Unit
NTC	T9GV1L14-5 [2]	0,09 euros
Relay	NCU15XH103F60R [3]	2,81 euros

Tabela 4.2: Main components - BOOM list.

5 | Energy Management

For wireless sensor nodes, storing energy and providing power in the required form is a crucial system component. The wireless devices should withstand various consuming patterns. There are three quite different levels of power consumption in every state. Some devices can consume quite some power over time and actually draw high current in certain operation states.

5.1 Consumption management

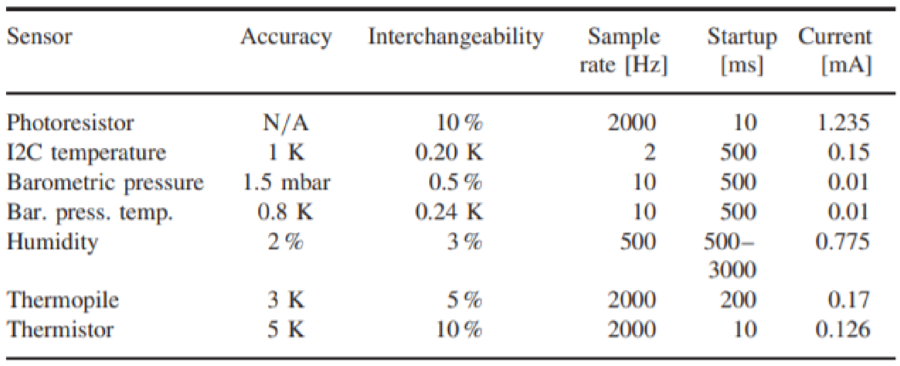
The main consumers of energy are the controller, the radio front ends, to some degree the memory, and, depending on the type, the sensors. In order for a sensor node to operate in such a way that reduces the amount of energy spent in running, the sensor can and must change his operational state according to the tasks at hand. The sensor will be in sleep mode when there is no instructions or events for him to run,thus saving energy. However, The transition between states takes both time and energy. The deeper the sleep state is, the greater time and energy is spent to awake the sensor up to full active state. In order to save energy while in sleep mode, the energy consumed by the sensor in sleep state plus the energy spent changing between active and sleep state must be less than the energy consumed by the sensor if he remained in the active state.

The main power consumption will be transmitting bits but it also depends in the number of times the sensor measures the environment temperature.

Looking at the energy consumption numbers and currents from different types of sensors 5.2 and 5.1 [6].

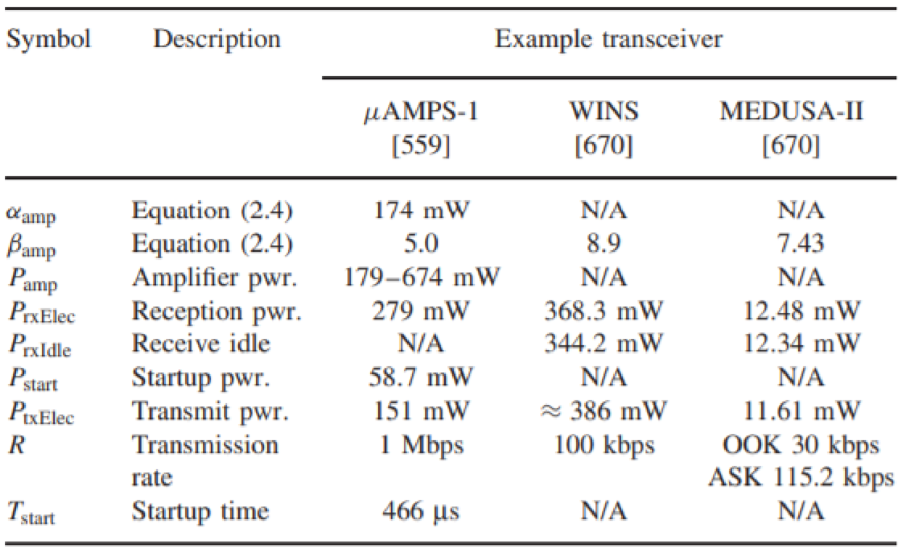
Disregarding the details, it is clear that communication is a considerably more expensive undertaking than computation. Still, energy required for computation cannot be simply ignored. Depending on the computational task, it is usually still smaller than the energy for communication, but still noticeable.

In addition, the sampling rate evidently is quite important. Not only does more frequent sampling require more energy for the sensors as such but also the data has to be processed and, possibly, communicated somewhere.



Sensor	Accuracy	Interchangeability	Sample rate [Hz]	Startup [ms]	Current [mA]
Photoresistor	N/A	10 %	2000	10	1.235
I2C temperature	1 K	0.20 K	2	500	0.15
Barometric pressure	1.5 mbar	0.5 %	10	500	0.01
Bar. press. temp.	0.8 K	0.24 K	10	500	0.01
Humidity	2 %	3 %	500	500–3000	0.775
Thermopile	3 K	5 %	2000	200	0.17
Thermistor	5 K	10 %	2000	10	0.126

Figura 5.1: Current for some type of sensors



Symbol	Description	Example transceiver		
		μAMPS-1 [559]	WINS [670]	MEDUSA-II [670]
α_{amp}	Equation (2.4)	174 mW	N/A	N/A
β_{amp}	Equation (2.4)	5.0	8.9	7.43
P_{amp}	Amplifier pwr.	179–674 mW	N/A	N/A
P_{rxElec}	Reception pwr.	279 mW	368.3 mW	12.48 mW
P_{rxIdle}	Receive idle	N/A	344.2 mW	12.34 mW
P_{start}	Startup pwr.	58.7 mW	N/A	N/A
P_{txElec}	Transmit pwr.	151 mW	$\approx$ 386 mW	11.61 mW
R	Transmission rate	1 Mbps	100 kbps	OOK 30 kbps ASK 115.2 kbps
T_{start}	Startup time	466 µs	N/A	N/A

Figura 5.2: Energy consumption for microcontrollers and radio transceivers

5.2 Energy Sources

Our sensor nodes will be kept working i.e. transmitting bits and sensing the environment 24 hours over 24 hours. Which means that the power consumption will be quite frequent. In order to power the sensors without the need for maintenance we choose powering the sensors by cable over batteries.

6 | Network Communications

6.1 Network type

WPANs (Wireless Personal Area Network) will be used as a network type in this project. The WPAN makes use of wireless network technology, such as Wi-Fi, not require that devices rely on physical cables to connect to the local area network.

6.2 Network topology

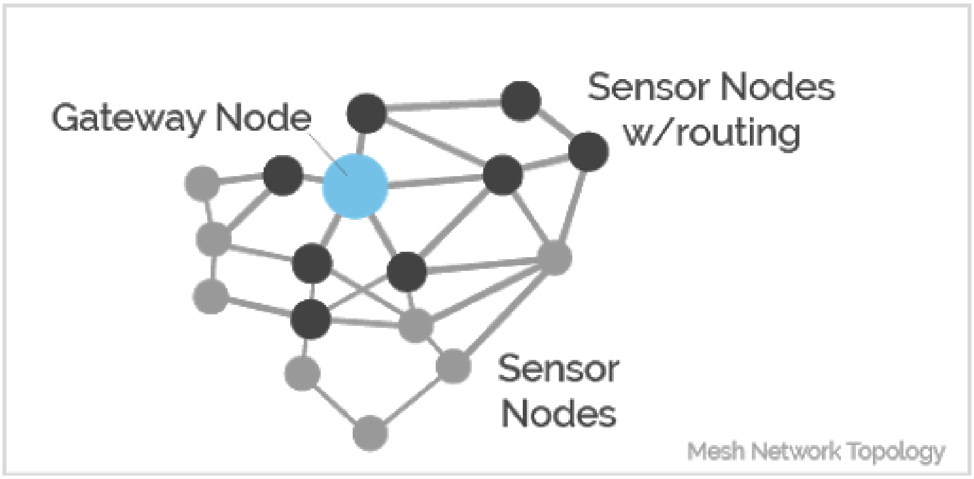


Figura 6.1: Mesh network topology illustration

Wireless mesh networks have been considered for IEEE 802.15.4 WPAN networks due to their flexibility and cost-efficiency. Typically, an extended service set of IEEE 802.15.4 consists of multiple basic service sets connected through a wired distribution system.

In IEEE standard 802.15.4 are considered two types of WiFi mesh network topologies: single-channel and multi-channel. Within a single-channel topology, all operating wireless frequencies are the same, but in a multi-channel topology, the mesh backhaul can work in different frequency channels by using multi-antenna devices. The use of a default channel binds one interface at each node to a single channel, thus limiting the flexibility offered by multiple channels, especially when each node has a limited number of radios.

6.3 Requirements and Guarantees

To project a successful network implementation, we need to take into account the traffic load that we expect to bring in. It is crucial to estimating the bandwidth required across every link of the network.

One way to estimate the bandwidth requirements is to identify how the users are currently performing their work: Frequency of use, Peak utilization traffic volumes, Average duration of each session, and each user groups' frequently accessed destinations.

In our case we want to have a maximum of 4 nodes connected to each receiver, to prevent an overload of traffic when broadcasts are made.

6.4 Medium access control (MAC)

The Medium Access Control Protocols are responsible for the medium access organization in wireless networks, they can be interpreted as rules that coordinate when a node transmits/receives packets. There are many protocols and they can be classified as contention-based or reservation-based protocols.But, only exists two types of MAC protocols in ContikiOS: *nullmac* and *csma* .

The *nullmac* protocol is a simple pass-through protocol that simply calls the appropriate RDC functions and the *csma* implements addressing, sequence number and re-transmissions. CSMA protocol keeps a list of packets to each of the neighbors and calculates statistics such as the number of re-transmissions, collisions, deferrals, and other things.

In our real world application we would choose *csma* as our prefered protocol.

7 | Simulation Results

In this chapter, we take a close look at the network architecture composed of control devices and temperature sensors, their interaction and consequent simulation results.

The control devices are nodes responsible for receiving the temperature from the temperature sensors, and for verifying if the temperature surpasses the safety temperature thresholds, which is the temperature that prevents the rack from overheating and thus damaging the servers, and then act accordingly. The temperature sensors are nodes that measure the temperature in the racks and send the information to the control devices.

In our simulation we ended up not using a multihop approach, so each temperature sensor has to be connected to a control device, as shown in figure 7.1. We also used sky motes in our simulation.

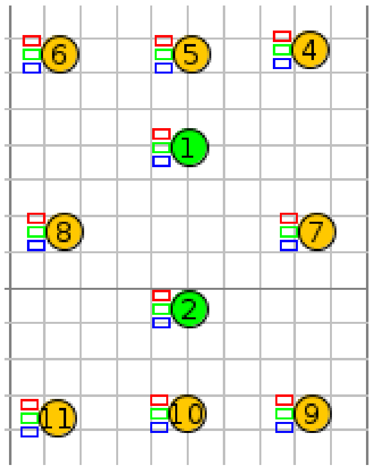


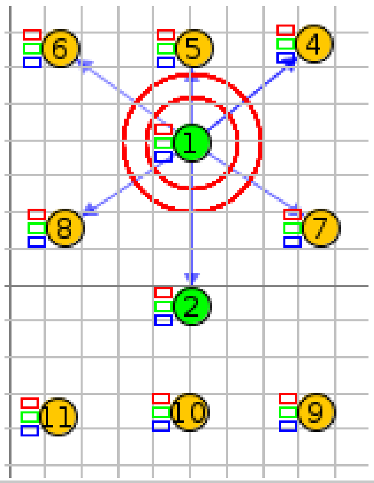
Figura 7.1: Network grid: Control devices (green) and Temperature sensors (orange)

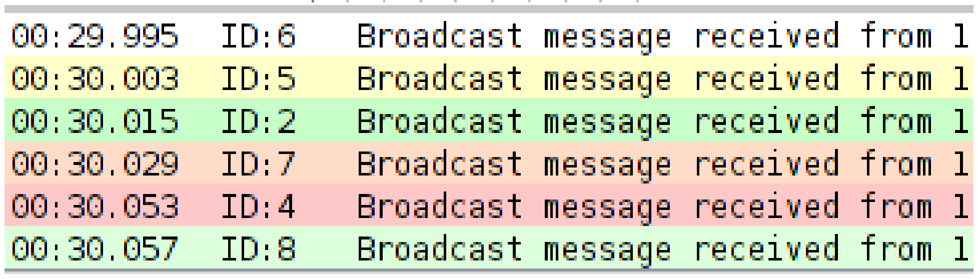
7.1 Broadcast vs unicast scheme

In order for the temperature and control nodes to communicate, two process threads were developed. The broadcast process thread and the unicast process thread.

- Broadcast process thread: Every node sends a broadcast message to all its neighbours. The control nodes stores all the neighbours in a list, which is used for the unicast messages, including both control and temperature sensors. The temperature nodes store only the address of the control nodes in a list. The broadcast thread runs in a infinite loop, which means that every node sporadically broadcast messages keeping their neighbour list updated.

In figure 7.2, we can see node 1 sending a broadcast and the mote output, with nodes 6, 5, 2, 7, 4 and 8 receiving it.



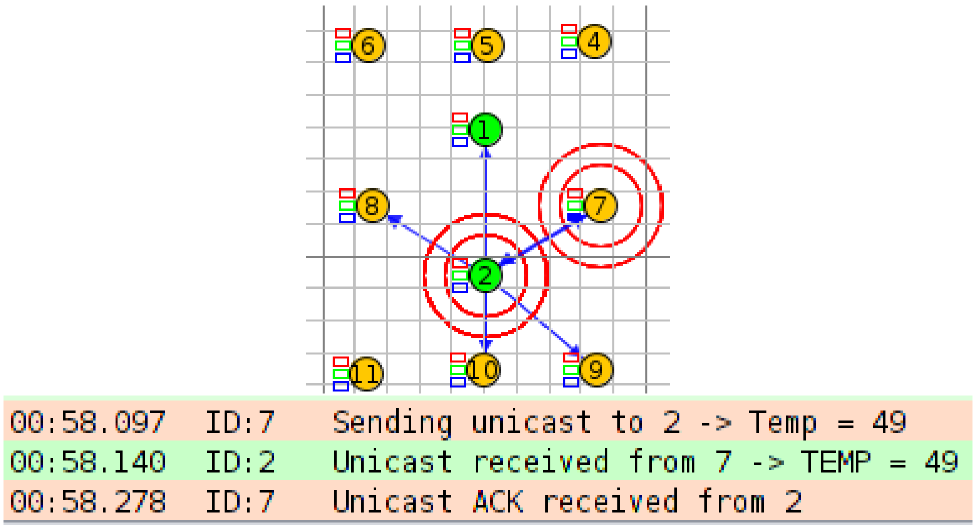


```
00:29.995   ID:6    Broadcast message received from 1
00:30.003   ID:5    Broadcast message received from 1
00:30.015   ID:2    Broadcast message received from 1
00:30.029   ID:7    Broadcast message received from 1
00:30.053   ID:4    Broadcast message received from 1
00:30.057   ID:8    Broadcast message received from 1
```

Figura 7.2: Broadcast example

- Unicast process thread: The temperature nodes use the unicast process thread to send the temperature information to the control nodes and to receive their acknowledge. The control nodes use the unicast process thread to respond to the temperature sensors.

  In figure 7.3, we have an example of node 7 sending a unicast message to node 2 with Temp = 49. It is received and mote 7 receives an ACK message, as shown in the mote output.



```
00:58.097  ID:7   Sending unicast to 2 -> Temp = 49
00:58.140  ID:2   Unicast received from 7 -> TEMP = 49
00:58.278  ID:7   Unicast ACK received from 2
```

Figura 7.3: Unicast Example

7.2 Fail safe

We developed a fail safe mechanism to protect both control devices and temperature sensors.

The fail safe mechanism lays its base in the neighbors list. The nodes send broadcast messages periodically to their neighbours in range. After receiving a broadcast message, the control node will store the structure of the sender (if there is no entry in the list) and will set a timer. This timer will reach a timeout if the node does not receive a broadcast message from the sender.

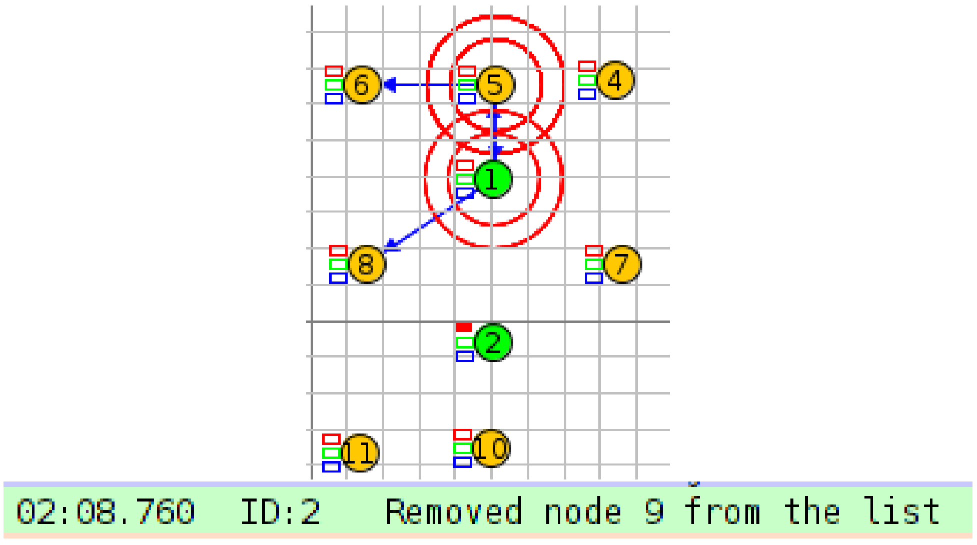
Following the timeout, the control node will remove the sender structure from the list. In case the control node does receive a broadcast, it will refresh the timer of the sender. This also works when the node that fails is a control node.

In figure 7.4, node 9 is separated from the other nodes, which in turn remove the leaving node from their lists. Then the red led is turned on.

7.3 Leds

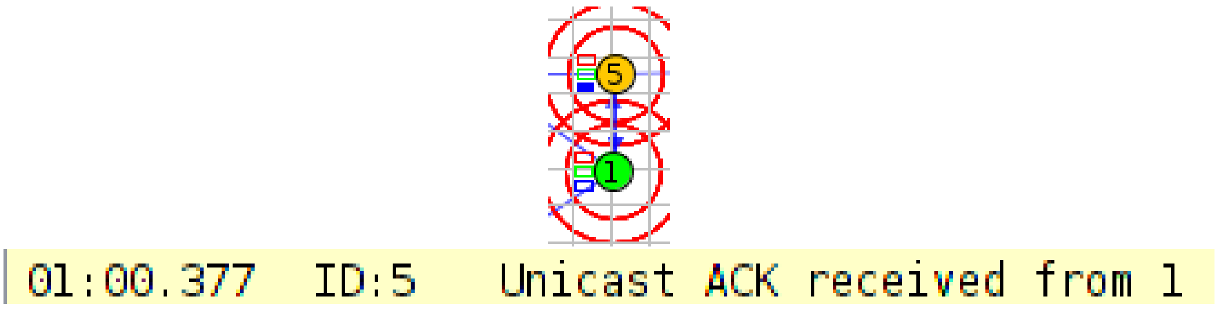
- Blue led: Toggles when the temperature sensor receives the acknowledge from the control node. In figure 7.5, the temperature node 5 receives the acknowledgment from the control node 1.

- Red led: The led turns on when a node fails to connect to a neighbour that it was



```
02:08.760   ID:2    Removed node 9 from the list
```

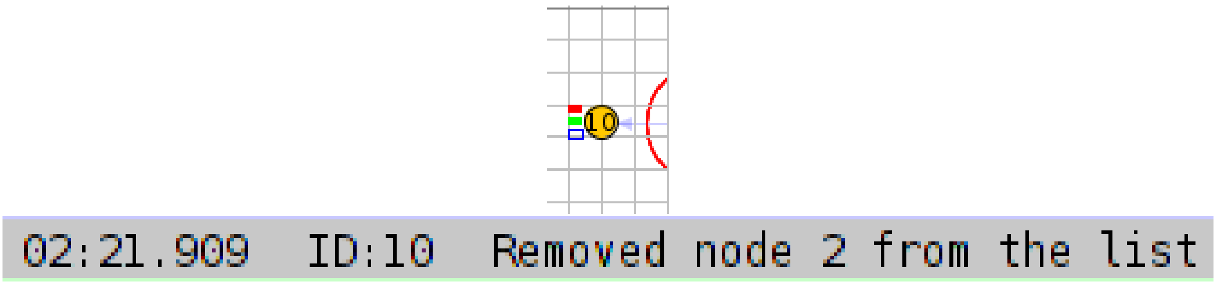
Figura 7.4: Fail safe mechanism



```
01:00.377   ID:5    Unicast ACK received from 1
```

Figura 7.5: Blue led

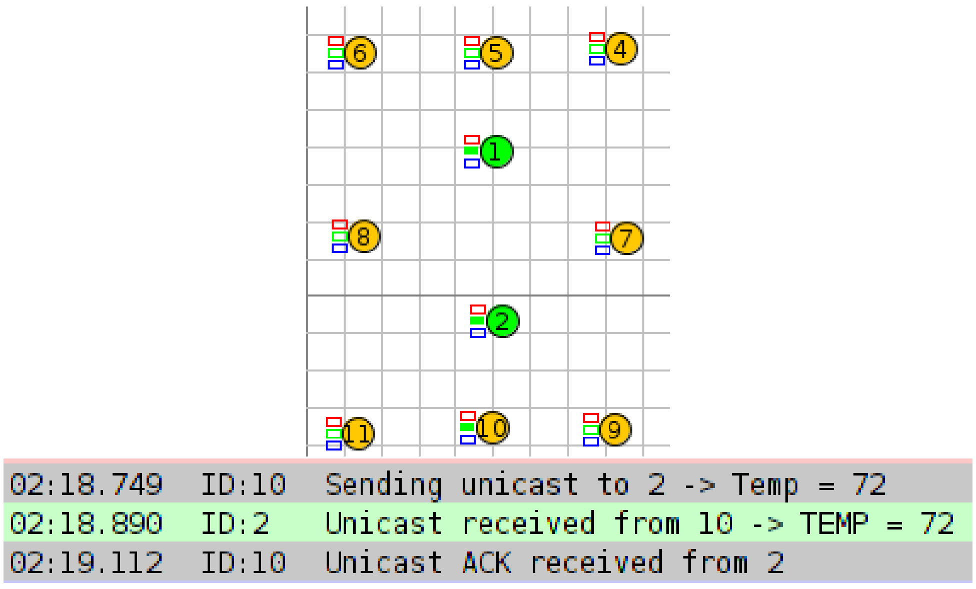
connected with. In figure 7.6, the node 10 removes the control node 2 and turns on the red led.



```
02:21.909   ID:10   Removed node 2 from the list
```

Figura 7.6: Red led

- Green led: The led turned on represents the AC turn on. In figure 7.7, the control node 2 receives the temperature 72, which is bigger then the threshold (70), leading to the green led turning on. The green led in node 1 also turns on because we implemented a function in which node sends a broadcast, that is interpreted by node 1, to also turn the AC on. After this node 1 will only turn off its AC when it receives another broadcast from node 2 with that command.



```
02:18.749  ID:10  Sending unicast to 2 -> Temp = 72
02:18.890  ID:2   Unicast received from 10 -> TEMP = 72
02:19.112  ID:10  Unicast ACK received from 2
```

Figura 7.7: Green led

8 | Bibliografia

[1] Datasheet ds18b20. `https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf`.

[2] Ntc t9gv1l14-5. `https://pt.mouser.com/ProductDetail/TE-Connectivity-PB/T9GV1L14-5?qs=sGAEpiMZZMtSzCF3XBhmW9nThoqLDM%252B2qP3bqmZSR4Y%3D`.

[3] Relay ncu15xh103f60rc. `https://pt.mouser.com/ProductDetail/Murata-Electronics/NCU15XH103F60RC?qs=sGAEpiMZZMtQRtO1VXT3j7hhkiBUBHi9QlmnOJxSfJ0%3D`.

[4] Jinzhu Chen ; Rui Tan ; Yu Wang ; Guoliang Xing ; Xiaorui Wang ; Xiaodong Wang ; Bill Punch ; Dirk Colbry. *A High-Fidelity Temperature Distribution Forecasting System for Data Centers*. 2012 IEEE 33rd Real-Time Systems Symposium, 2012.

[5] Adel A. Eidan, Kareem J. Alwan, Assaad Alsahlani, and Mohamed Alfahham. Enhancement of the Performance Characteristics for Air-Conditioning System by Using Direct Evaporative Cooling in Hot Climates. *Energy Procedia*, 142:3998–4003, 2017.

[6] Andreas Willig Holger Karl. *Protocols and Architectures for Wireless Sensor Networks*. John Willey & Sons, 2005.

[7] C.B. Bash ; C.D. Patel ; R.K. Sharma. *Dynamic thermal management of air cooled data centers*. Thermal and Thermomechanical Proceedings 10th Intersociety Conference on Phenomena in Electronics Systems, 2006. ITHERM 2006., 2006.