IOT Exercise #2 report

Francesco Spangaro - Luca Tosetti

20 March 2025

Academic Year 2024 - 2025

Contents

1 Introduction

The IOT network we were assigned to is composed by:

- A battery-powered, Wi-Fi-enabled valve that receives a temperature measurement every 5 minutes. Every 30 minutes it computes the average of all the received temperatures and either opens or closes itself based on the result of the computation.
- A battery-powered, Wi-Fi-enabled sensor that measures and transmits temperature data every 5 minutes.
- A Raspberry pi, connected to the power grid, which supports only MQTT communication.

Both the sensor and the valve can communicate either via CoAP or MQTT, with a specified, pre-defined topic or resource. The topic/resource length is 10 bytes and the payload size is 8 bytes which are already included in the following packet payloads.
The payloads for each message are:

CoAP		MQTT	
GET Request	60 B	Subscribe	58 B
GET Response	55 B	Sub ACK	52 B
PUT Request	77 B	Publish	68 B
PUT Response	58 B	Pub ACK	51 B
Empty ACK	14 B	Connect	54 B
		Con ACK	47 B
		Ping req	52 B
		Ping resp	48 B

Assume that:

1. Transmit and receive cost per bit are: $E_{TX} = 50\frac{nJ}{bit}$, $E_{RX} = 58\frac{nJ}{bit}$;
2. The Wi-Fi network is ideal (no losses);
3. The processing cost on the valve to compute the average temperature every 30 minutes is $E_c = 2.4\ mJ$;
4. The sensor and valve start in power-off state.

2 EQ1

We were tasked with finding the total energy consumed by the two battery-powered devices over a period of 24 hours when using CoAP (EQ1a) and MQTT (EQ1b), each in their most efficient configuration energy-wise.

2.1 EQ1a

To solve the assignment using CoAP, we had to make some assumptions:

- Packet transmission is instantaneous (no delay time);
- The sensor is in charge of transmitting the measured temperature. Thus, we use a system configuration in which the sensor is the only one that sends CoAP messages (specifically PUT requests) acting as a Client. The valve receives the data sent by the sensor, acting as a Server;
- We use CoAP NON-confirmable messages since, from the exercise assumptions, we know that the WiFi network is ideal, meaning that we don't need acks to confirm whether the packets were correctly received or not;
- We use CoAP observation mode. It allows a CoAP server to periodically notify a Client subscribed to a specific topic of any value changes occurred in that topic.

We computed the results needed using the following formulas:

$$\begin{aligned} \#\ of\ measured\ temp. &= \frac{24h}{5min} = 288 \\ \#\ of\ avg\ computed &= \frac{24h}{0.5h} = 48 \end{aligned}$$

$$\begin{aligned} E_{rx,\ GET\ request} &= E_{rx} * 60B * 8 = 27.8\ \mu J \\ E_{tx,\ GET\ request} &= E_{tx} * 60B * 8 = 24\ \mu J \\ E_{rx,\ GET\ response} &= E_{rx} * 55B * 8 = 25.5\ \mu J \\ E_{tx,\ GET\ response} &= E_{tx} * 55B * 8 = 22\ \mu J \end{aligned}$$

$$\begin{aligned} E_{tot,\ valve} &= E_{tx,\ GET\ request} + E_{rx,\ GET\ response} * \#\ of\ measure\ temp. + E_c * \#\ of\ avg\ computed = 122.57\ mJ \\ E_{tot,\ sensor} &= E_{rx,\ GET\ request} + E_{tx,\ GET\ response} * \#\ of\ measured\ temp. = 6.36\ mJ \\ E_{tot,\ system} &= E_{tot,\ valve} + E_{tot,\ sensor} = 128.93\ mJ \end{aligned}$$

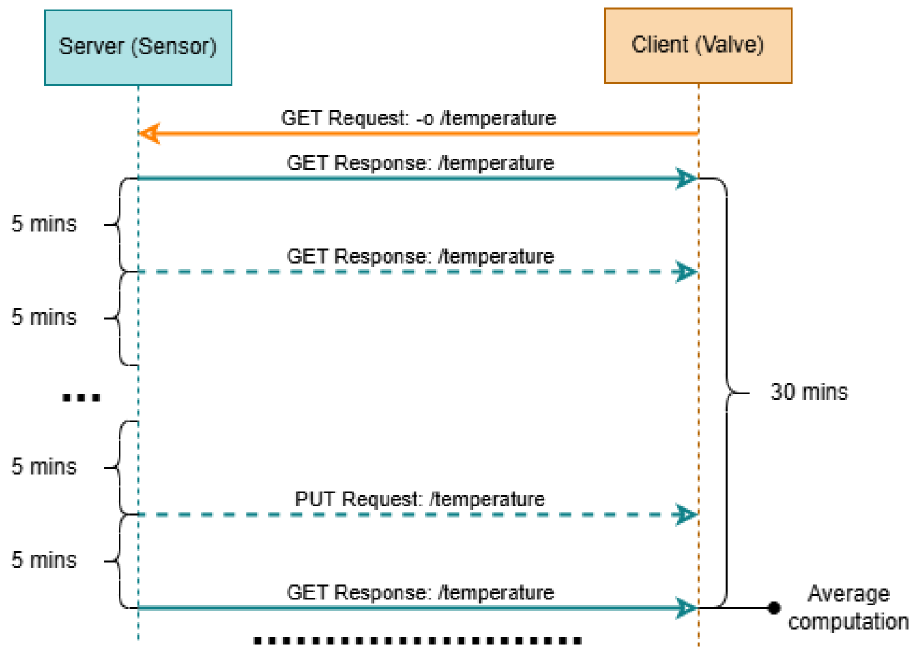

Figure 1: CoAP communication design

2.2 EQ1b

To solve the assignment using MQTT, we had to make some assumptions:

- Packet transmission is instantaneous (no delay time);
- We use the Raspberry Pi as an MQTT broker;
- We use a QoS = 0 (at most once), since from the exercise description we know that the WiFi network is ideal;
- We assume that the sensor Client specifies, at the moment of the connection with the broker, a value for the KeepAlive field that is greater than the periodic time with which it sends the temperature values to the broker (for instance, we could use a value of 6 minutes). For the valve device, instead, we directly set the KeepAlive's value to 0, to disable it and therefore avoid periodically sending PING packets to keep the session alive.

We computed the results needed using the following formulas:

$$\# \ of \ measured \ temp. = \frac{24h}{5min} = 288$$
$$\# \ of \ avg \ computed = \frac{24h}{0.5h} = 48$$
$$P_{connect} = 54B * 8 = 432 \ bits$$
$$P_{connack} = 47B * 8 = 376 \ bits$$
$$P_{subscribe} = 58B * 8 = 464 \ bits$$
$$P_{suback} = 51B * 8 = 408 \ bits$$
$$P_{publish} = 68B * 8 = 544 \ bits$$
$$E_{conn} = E_{tx} * P_{connect} + E_{rx} * P_{connack} = 43.4 \ \mu J$$
$$E_{sub} = E_{tx} * P_{subscribe} + E_{rx} * P_{suback} = 46.86 \ \mu J$$

$$E_{tot, \ valve} = E_{conn} + E_{sub} + E_{rx} * P_{publish} * \# \ of \ measured \ temp. + E_c * \# \ of \ avg \ computed = 124.37 \ mJ$$
$$E_{tot, \ sensor} = E_{conn} + E_{tx} * P_{publish} * \# \ of \ measured \ temp. = 7.87 \ mJ$$
$$E_{tot, \ system} = E_{tot, \ valve} + E_{tot, \ sensor} = 132.24 \ mJ$$

EXTRA: If we want to check the energy consumed by the broker:

$$E_{tot, \ broker} = (E_{rx} * P_{connect} + E_{tx} * P_{connack}) * 2 + E_{rx} * P_{subscribe} + E_{tx} * P_{suback} + \\ +(E_{rx} + E_{tx}) * P_{publish} * \# \ of \ measured \ temp. = 17,05 \ mJ$$

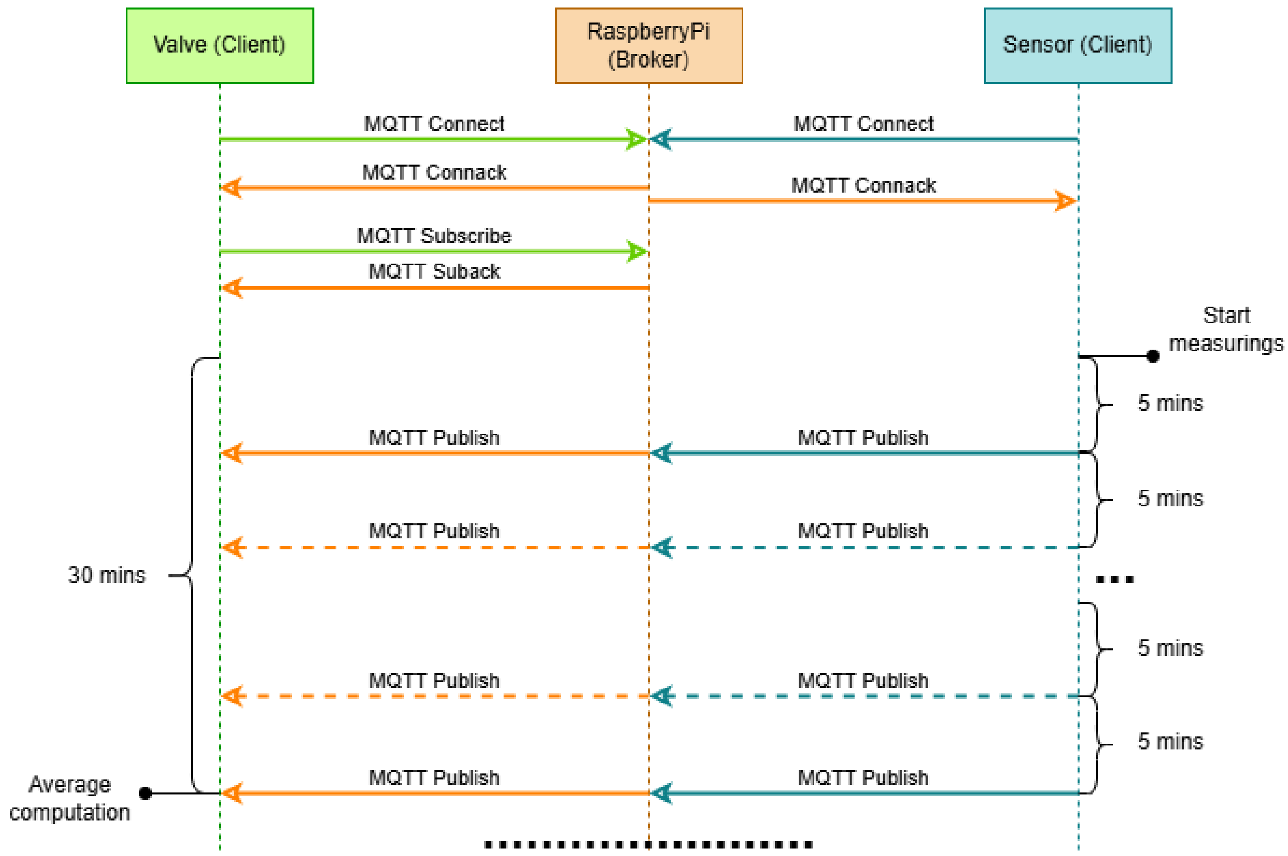

Figure 2: MQTT communication design

3 EQ2

To further improve the energy efficiency of the system while using MQTT and the Raspberry Pi as broker, we thought of two possible solutions.

3.1 First solution

The most energy-consuming task of the system is the computation of the average temperature value. To lower the system's energy consumption, we moved this computation to the Raspberry Pi, which is connected to the power grid. Being connected to the power grid, we do not have to limit its power consumption since we do not have the risk of its battery running out of energy. To properly evaluate the improvements this solution brings the system, we had to make some assumptions:

- Packet transmission is instantaneous (no delay time);
- We use a QoS = 0 (at most once), since from the exercise description we know that the WiFi network is ideal;
- We assume that the sensor Client specifies, at the moment of the connection with the broker, a value for the KeepAlive field that minimizes the energy consumed. This minimizes the number of pings that will be sent between the system components. The maximum value is approximately 18 hours, but for the sake of simplicity we used 12 hours.
- We assume that the valve Client specifies a KeepAlive value of 0 seconds. This way, the KeepAlive mechanism is disabled, and the valve does not send a PING request packet to the broker to keep the session alive. We have made this decision based on the assumption that the network is reliable and therefore it is not necessary to check the maintenance of the session.
- Since the valve does not need to receive the measurement every 5 minutes, it subscribes to a different topic. We assume that the new resource/topic and payload are respectively 10 and 8 bytes.

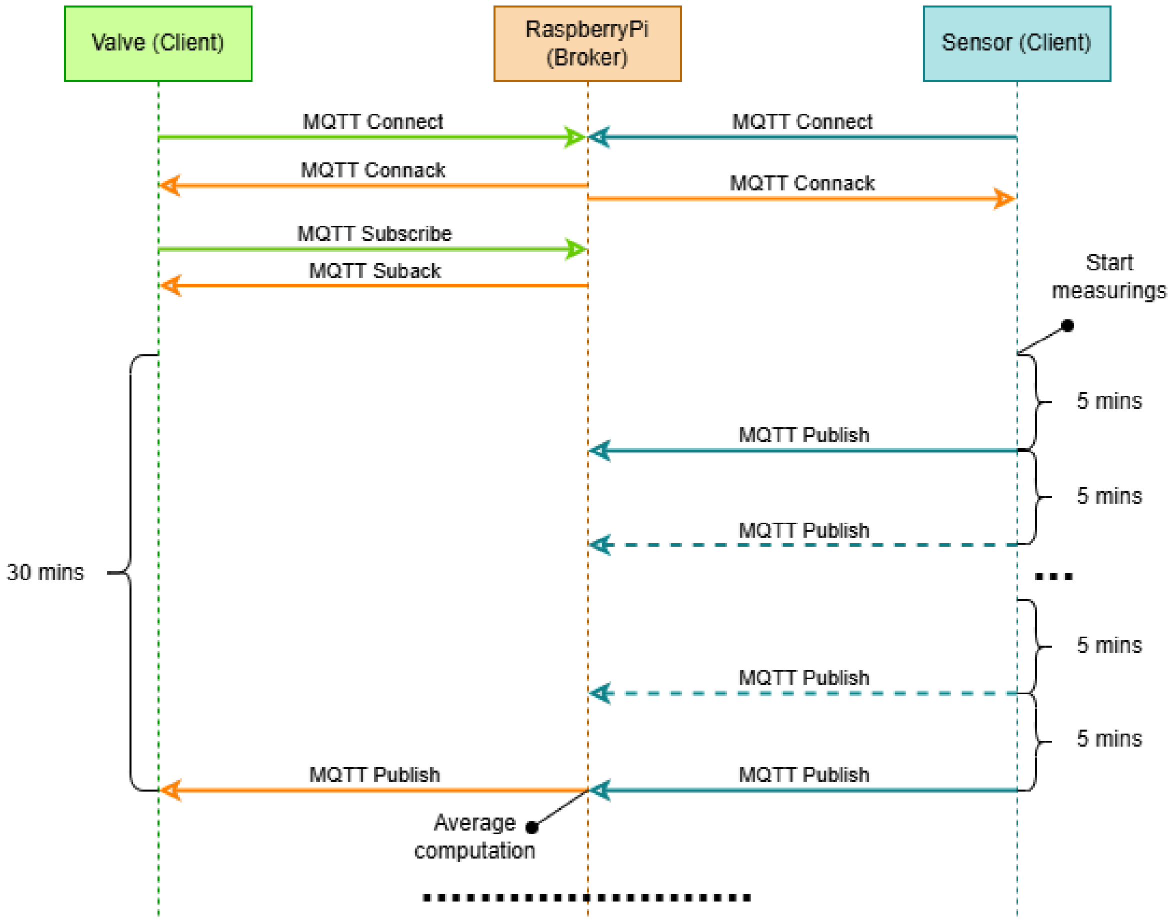

Figure 3: First solution communication design

We used these formulas to evaluate the improvement in energy consumption of the system:

$$
\begin{aligned}
\#\ of\ measured\ temp. &= \frac{24h}{5min} = 288\\
\#\ of\ avg\ computed &= \frac{24h}{0.5h} = 48\\
P_{connect} &= 54B * 8 = 432\ bits\\
P_{connack} &= 47B * 8 = 376\ bits\\
P_{subscribe} &= 58B * 8 = 464\ bits\\
P_{suback} &= 51B * 8 = 408\ bits\\
P_{publish} &= 68B * 8 = 544\ bits\\
P_{pingreq} &= 52B * 8 = 416\ bits\\
P_{pingresp} &= 48B * 8 = 384\ bits\\
E_{conn} &= E_{tx} * P_{connect} + E_{rx} * P_{connack} = 43.4\ \mu J\\
E_{sub} &= E_{tx} * P_{subscribe} + E_{rx} * P_{suback} = 46.86\ \mu J
\end{aligned}
$$

$$
\begin{aligned}
E_{tot,\ valve} &= E_{conn} + E_{sub} + E_{rx} * P_{publish} * \#\ of\ avg\ computed = 1.6\ mJ\\
E_{tot,\ sensor} &= E_{conn} + E_{tx} * P_{publish} * \#of\ measured\ temp. = 7.87\ mJ\\
E_{tot,\ system} &= E_{tot,\ valve} + E_{tot,\ sensor} = 9.47\ mJ
\end{aligned}
$$

The system in this case is: $\frac{E_{tot,\ system,\ base}}{E_{tot,\ system,improved}} = \frac{132.24\ mJ}{9.47\ mJ} \approx 13.96$ times more energy efficient in respect to the system configuration we started with in section EQ1b.

3.2 Second solution

The second most energy-expensive task in the system is the transmission of the sensed temperature to the Raspberry Pi from the sensor. We decided that it would be best to only send the new measurement if the new value is different from the last sent one. If it is the same, the Raspberry Pi increases a counter corresponding to the measurement, then, to evaluate the new average, it uses the formula:

$$avg = \frac{\sum_{n=1}^{k} T_n * c_n}{\sum_{n=1}^{k} c_n}$$

With T_n being the temperature and c_n the corresponding counter. The counter is reset to zero after the average is computed, and only the last received temperature reading is kept in memory. We applied this improvement to the system described in the first solution. To evaluate the improvement this new architecture would bring to the system, we evaluated three cases:

- A best case scenario, where the temperature remains constant for 24 hours.
- An average case scenario, where the temperature changes every hour.
- A worst case scenario, where the temperature changes every 5 minutes.

3.2.1 Best case scenario

In the best case scenario, the temperature never changes. The sensor has to send only the first publish message and a ping packet every 12 hours.

$$E_{tot,\ valve} = E_{conn} + E_{sub} + E_{rx} * P_{publish} * \#\ of\ avg\ computed = 1.6\ mJ$$
$$E_{tot,\ sensor} = E_{conn} + = E_{tx} * P_{publish} + (E_{tx} * P_{pingreq} + E_{rx} * P_{pingresp}) * 2 = 156.7\ \mu J$$
$$E_{tot,\ system} = E_{tot,\ valve} + E_{tot,\ sensor} = 1.75\ mJ$$

The system in this case is: $\frac{E_{tot,\ system,\ base}}{E_{tot,\ system,\ best}} = \frac{132.24\ mJ}{1.75\ mJ} \approx 75.56$ times more energy efficient in respect to the system configuration we started with at section EQ1b.

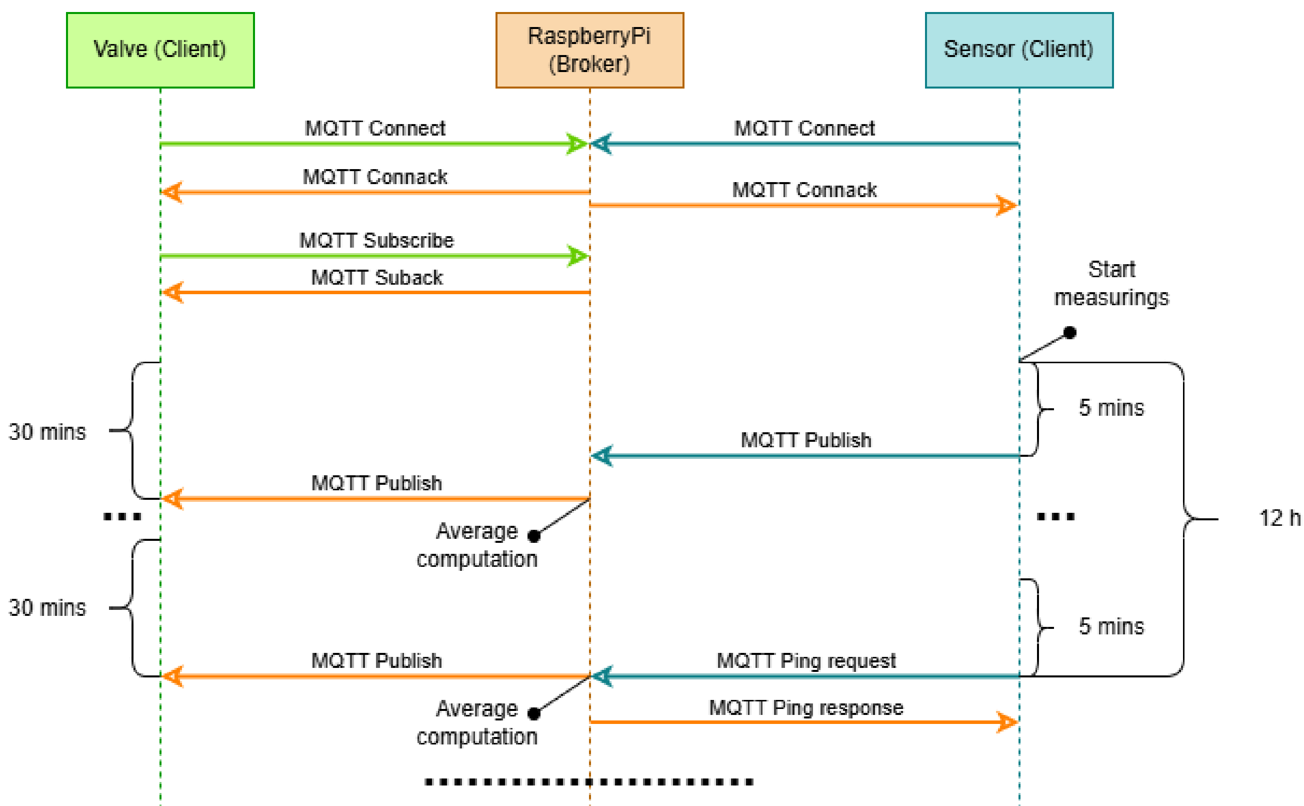

Figure 4: Best case scenario for the second solution

3.2.2 Average case scenario

In the average case scenario, the temperature changes once every hour. The sensor has to send a publish packet every hour in order to update the Raspberry Pi, but because the KeepAlive time is set to 12 hours, it will never send a ping packet.

$$\#\ of\ sent\ temp. = \frac{24h}{1h} = 24$$

$$E_{tot,\ valve} = E_{conn} + E_{sub} + E_{rx} * P_{publish} * \#\ of\ avg\ computed = 1.6\ mJ$$
$$E_{tot,\ sensor} = E_{conn} + E_{tx} * P_{publish} * \#of\ sent\ temp. = 696.4\ \mu J$$
$$E_{tot,\ system} = E_{tot,\ valve} + E_{tot,\ sensor} = 2.3\ mJ$$

The system in this case is: $\frac{E_{tot,\ system,\ base}}{E_{tot,\ system,\ average}} = \frac{132.24\ mJ}{2.3\ mJ} \approx 57.5$ times more energy efficient in respect to the system configuration we started with at section EQ1b.

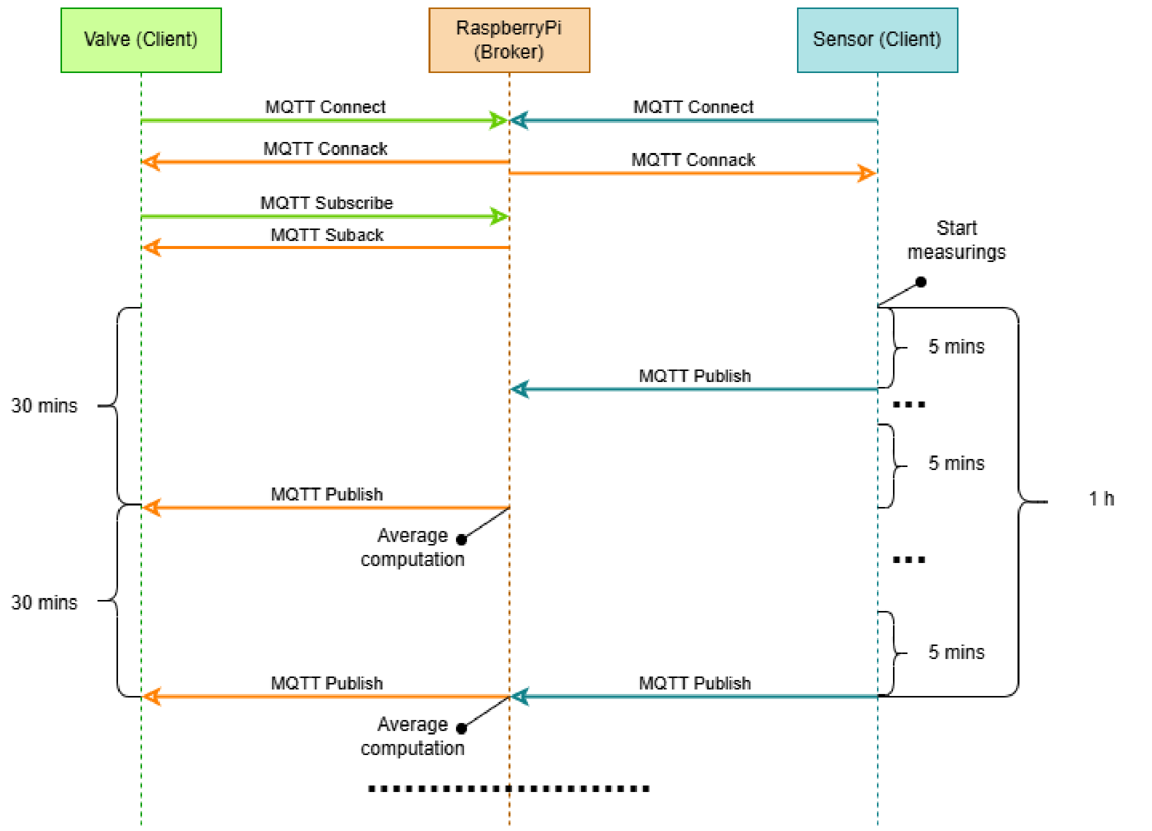

Figure 5: Average case scenario for the second solution

3.2.3 Worst case scenario

Since in the worst case scenario the sensor sends a publish message every 5 minutes, the improvement is the same as the one found when moving the average computation from the valve to the Raspberry Pi, it being almost 14 times more efficient than the configuration we found at section EQ1b.