TP4 Machine to Machine : Node-Red

Introduction

Node-Red is a flow-based tool allowing us to connect hardware, API and services for Internet of Things through a visual programmation method based on Node.js. It gives several boxes representing our architectures in every layer and makes them work together. In this TP we will first master the basics of Node-Red and then connect our different works from the first TP with OneM2M and MQTT.

Installation :

The first step is to install node-red in our devices, with npm command and also downloading the oneM2M plugin coming with this TP. It provides oneM2M nodes like AE, cIN and CSE but also programming tools with fake sensors and so.

Node-Red online tutorial :

In order to understand the basics of Node-Red and how boxes interact between them we first follow the online Node-Red tutorial (https://nodered.org/docs/tutorials/first-flow).

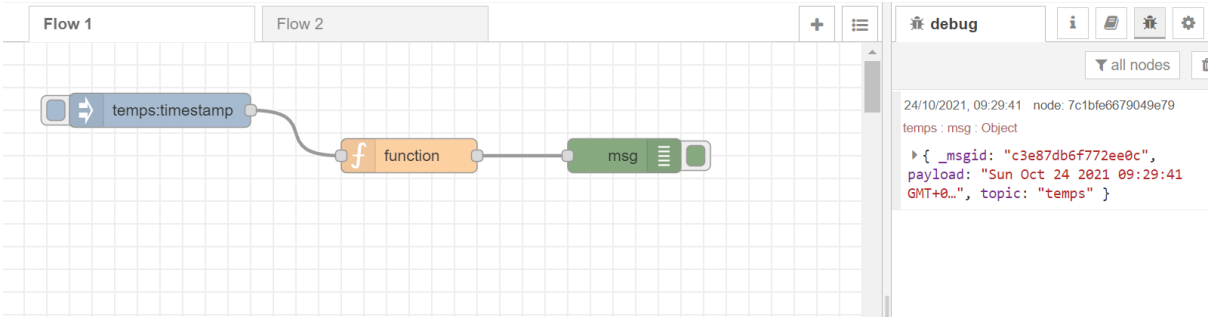

Figure 1 : First Flow on Node-Red

This first flow is composed of 3 Node-Red basics blocks : Injects, Function and Debug. A Node-Red block can be defined with his inputs, outputs, attributes/properties and behaviour. For example, here, the Inject block input is a button representing a boolean and his attributes define the type of message he is transmitting to his output (the payload is a timestamp and the topic is called temps). After that, the function is configured to receive a timestamp and traduct it as a date. The last green block is a debug block which displays the output of the previous block in debug configuration.
Now, we know how to link blocks with each other and have the basics tools to create the architectures of our oneM2M application by configuring well the blocks given with this practical.

LOM2M - Node-RED flow

In this part, we will focus on the LOM2M nodes provided with this practical. Indeed, thanks to the Named Sensor Data and Content Extractor blocks we will be able to get the value of our sensor present in the MN-cse of our local oneM2M application.

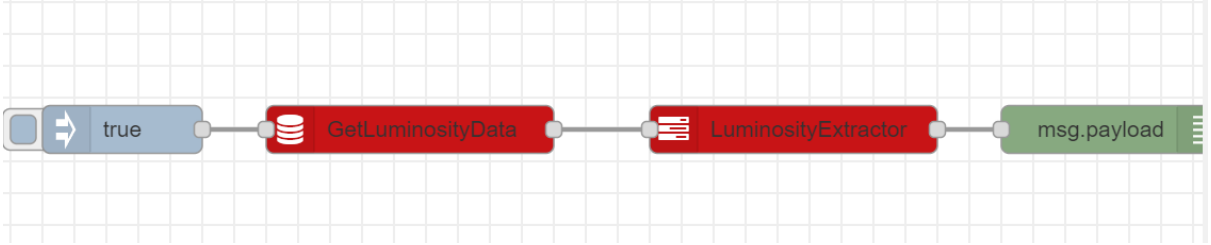

Figure 2 : LOM2M first Node-Red flow

With such a view of the architecture we can't understand how we are able to extract values from our local oneM2M application, indeed, everything is done inside the blocks through the attributes of each of them.

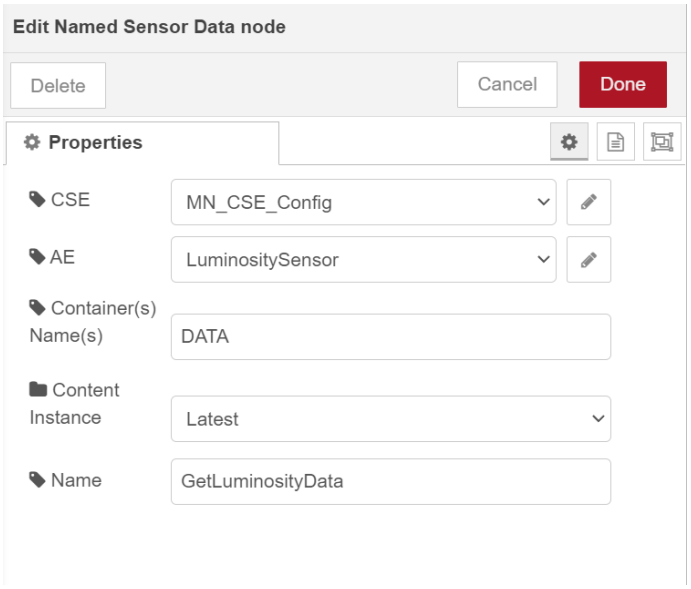

Figure 3 : Properties of GetLuminosityData

The figure 3 shows the different attributes needed in the Named Sensor Data block in order to get the value of a defined sensor represented as an AE in our oneM2M architecture. Some of those properties also need to be configured.

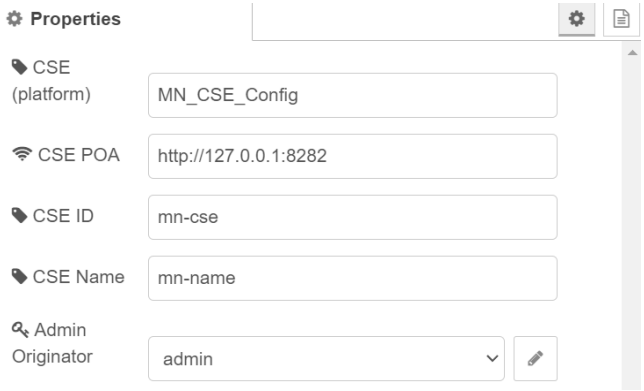

Figure 4 : Configuration of CSE property

It is at this layer that appears the POA of our MN-CSE and that Node-Red gets the missing information to get the CIN of our AE representing our sensor.
The output of GetLuminosityData is the entire HTML response; this is why we pass it through the content extractor to get a clean debug output.

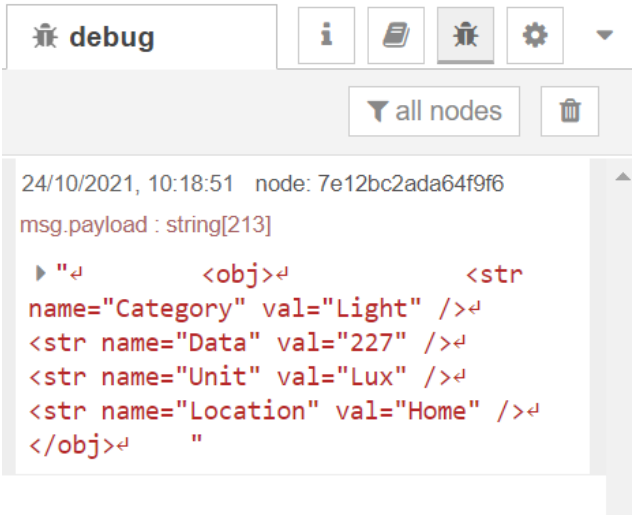

Figure 5 : Debug output of our flow

The output gives us the latest CIN in the LuminositySensor AE as we can see in the configuration (figure 3).

MQTT nodes :

Node-RED also allows us to use MQTT blocks that can publish and subscribe to our local broker (we are using mosquitto). You just need to setup the boxes like this :

Figure 6 : view of MQTT box properties

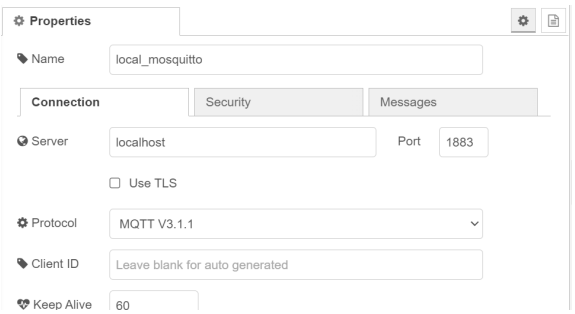

Figure 7 : view of Server Configuration(local_mosquitto)

In this example we have the view of a publisher, it will publish on test/TEST-ID/Sub topic in a broker running locally.
With those blocks we can now link our oneM2M architectures with the mosquitto broker and our ESP82, for example we can light up our LED in function of the last value entered in the luminosity sensor.

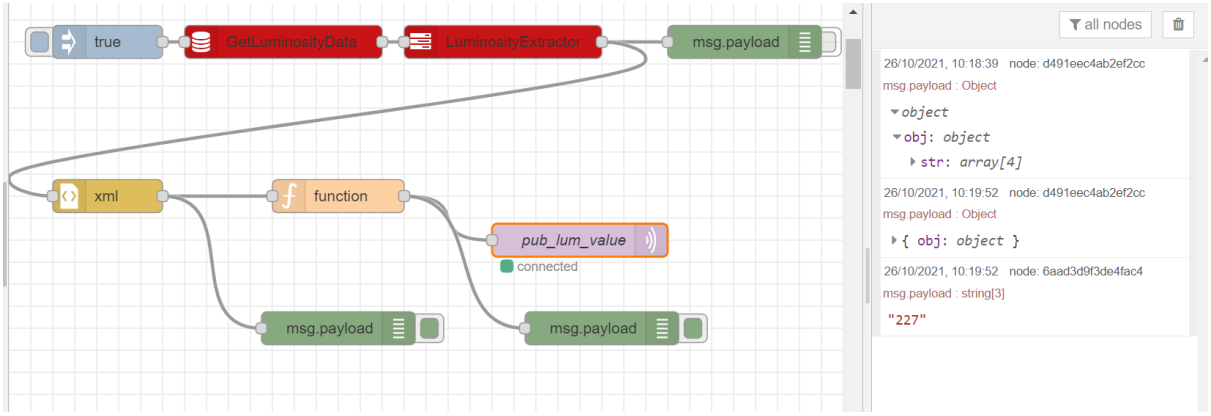

Figure 8 : Last Flow connecting our OneM2M to ESP through mosquitto broker

This last flow allows us with the xml parser and function block to enter only the sensor value in the MQTT publisher. The function block is detailed below :

Figure 9 : Function isolating the luminosity data and setting it in the message payload

Conclusion

This lab gave us the tools to understand if Node-RED is working and how it is efficient in order to fast prototyping IOT applications. Indeed we used OneM2M and MQTT blocks and then linked them between each other in order to create a complete application with heterogeneous devices. We sadly couldn't develop the high level/ front-end application with the dashboard. We hope we will be able to implement our integrator project with a real-time map in node-red.