Middleware for the IoT: TP3

Fast application prototyping for IoT

LIEVRE Agathe

NGUYEN Assia

5ISS

2021/2022

1. Objectives

The objective is to use what we did in the previous TPs to create a complete application that will interact with several real and virtual devices.

2. Deploy the architecture

We deploy what we did in the previous TPs.

3. High level application

We have two options to create a high level application : implement everything with our HTTP/MQTT client or use a tool like Node RED. We choose to use Node RED because it is faster and easier.

4. Configure and use Node RED

a. Installation

We installed the version LTS 8.x of Node.js. Then, we proceed with the installation of Node RED. We use the node package manager, npm, that comes with Node.js

npm install -g --unsafe-perm node-red

b. Integration of oneM2M nodes in Node RED

After downloading the zip file on Moodle,

cd /path/to/LOM2M/node-red/nodes

npm install request

cd ~/.node-red (the path where Node-RED was installed)

npm install /path/to/LOM2M/node-red/nodes

After the installation of Node-RED and the integration of LOM2M nodes, we can run Node RED using the command:

node-red

You can then access the Node-RED editor by accessing this webpage through your web browser: http://localhost:1880

We installed the following palettes : “node-red-contrib-ide-iot” and “node-red-contrib-lom2m”.

We did the tutorials :

https://nodered.org/docs/tutorials/first-flow

https://nodered.org/docs/tutorials/second-flow

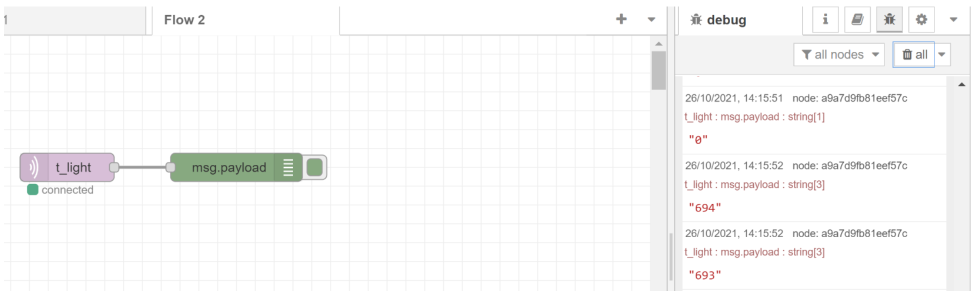
5. Applications

a. Simple test, get sensor values and display them

We used two nodes:

- MQTT in: add the name of the topic, server of the broker (172.20.10.3)
- Debug: to display the payload of the message in the debug console

We successfully retrieved and displayed the luminosity sensor data on Node-red.

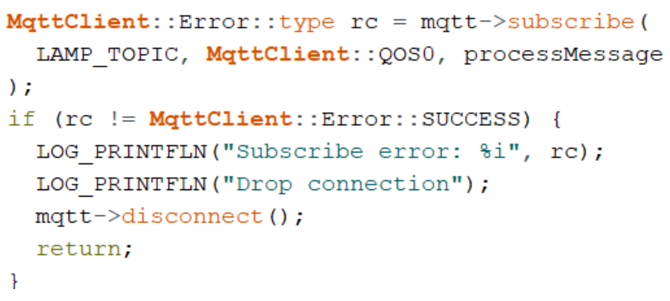


```
26/10/2021, 14:15:51   node: a9a7d9fb81eef57c
t_light : msg.payload : string[1]
"0"
26/10/2021, 14:15:52   node: a9a7d9fb81eef57c
t_light : msg.payload : string[3]
"694"
26/10/2021, 14:15:52   node: a9a7d9fb81eef57c
t_light : msg.payload : string[3]
"693"
```

Screenshot of the flow and the debug console

b. Sensors and actuators

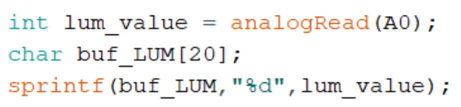
We first modified the Arduino code to subscribe the EPS8266 to the "LAMP_TOPIC" which controls the activation of the LED.



```
MqttClient::Error::type rc = mqtt->subscribe(
  LAMP_TOPIC, MqttClient::QOS0, processMessage
);
if (rc != MqttClient::Error::SUCCESS) {
  LOG_PRINTFLN("Subscribe error: %i", rc);
  LOG_PRINTFLN("Drop connection");
  mqtt->disconnect();
  return;
}
```

Screenshot of the Arduino subscription code

We stored the luminosity value in a string to later display it on the node-RED dashboard.

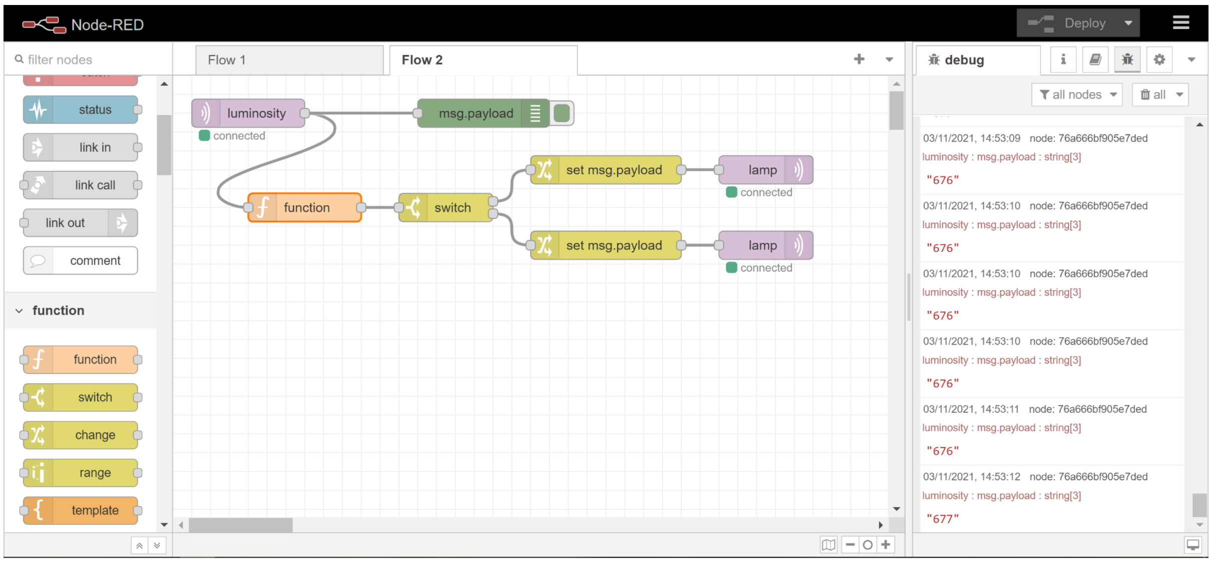


```
int lum_value = analogRead(A0);
char buf_LUM[20];
sprintf(buf_LUM,"%d",lum_value);
```

Screenshot of the Arduino conversion code

On Node-RED, we added more nodes:

- Function: this node converts the payload from string to integer.
- Switch: we used this node instead of the "Simple condition" node to condition the activation of the LED (if the value is inferior to 500, the LED is activated, if not, the LED is deactivated)
- Change: it is used to set the payload of the message ("ON" to turn on the LED, "NON" to turn off the LED)
- MQTT out: to publish the messages on the LAMP_TOPIC



```
03/11/2021, 14:53:09   node: 76a666bf905e7ded
luminosity : msg.payload : string[3]
"676"
03/11/2021, 14:53:10   node: 76a666bf905e7ded
luminosity : msg.payload : string[3]
"676"
03/11/2021, 14:53:10   node: 76a666bf905e7ded
luminosity : msg.payload : string[3]
"676"
03/11/2021, 14:53:10   node: 76a666bf905e7ded
luminosity : msg.payload : string[3]
"676"
03/11/2021, 14:53:11   node: 76a666bf905e7ded
luminosity : msg.payload : string[3]
"676"
03/11/2021, 14:53:12   node: 76a666bf905e7ded
luminosity : msg.payload : string[3]
"677"
```

Screenshot of the flow

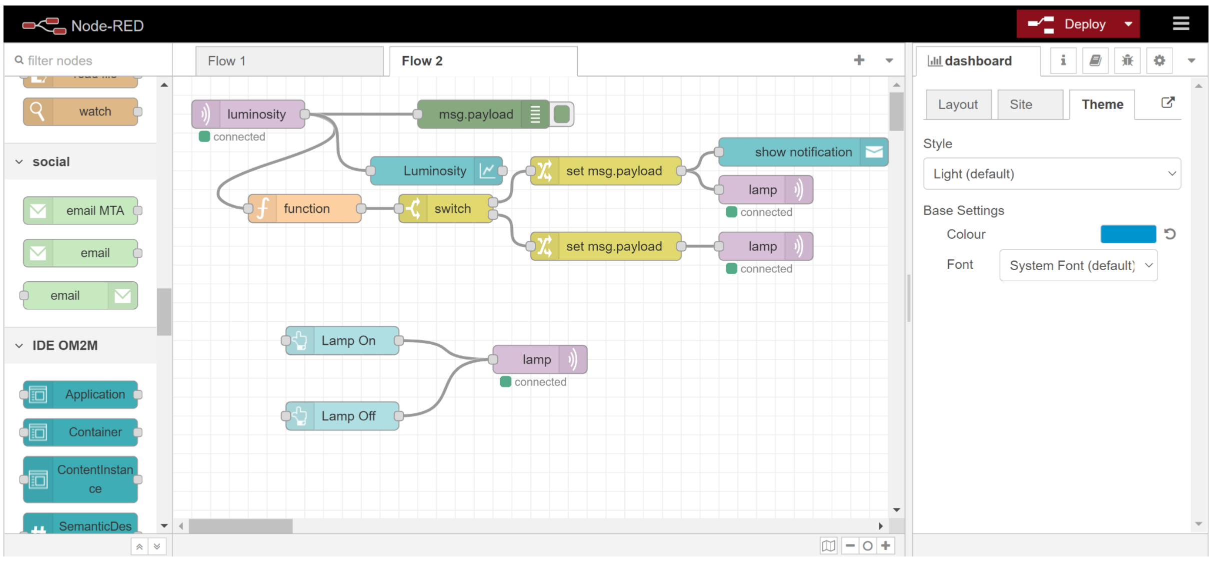
The node flow resulted in a correct output: the LED reacted as we wanted, by lighting up when the luminosity was under the defined threshold and turning off otherwise.

c. Dashboard

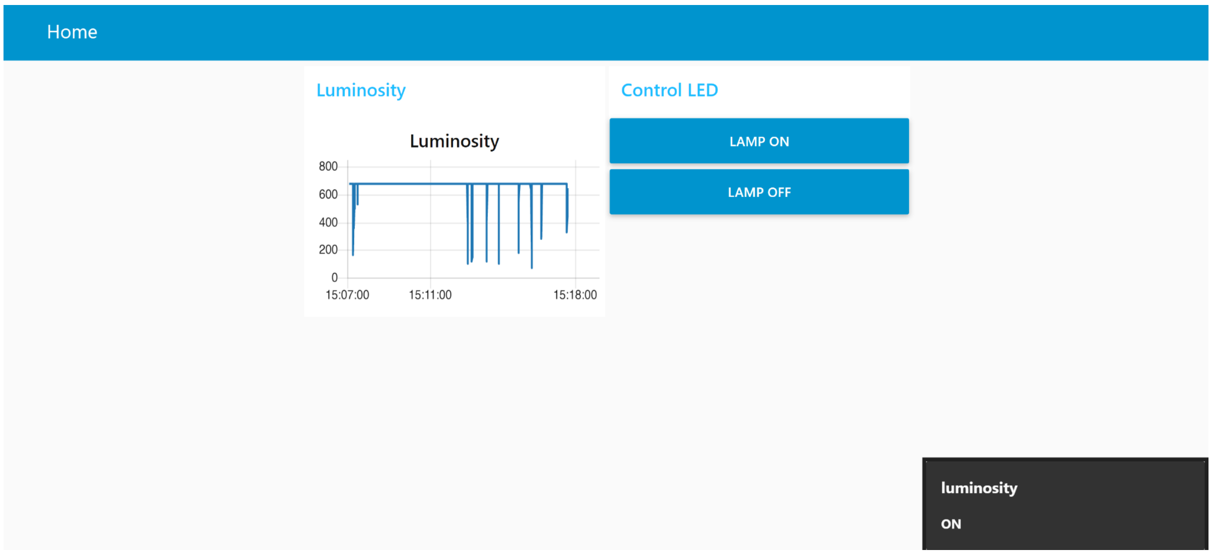
To create a dashboard, we download a new palette of nodes named "node-red-dashboard" and we used:

- Button: we created two buttons to turn on or off the LED via the dashboard.
- Chart: to create a graphic of the luminosity values.
- Notification: to alert the user when the LED is on or off.

We also created two categories “Luminosity” and “Control LED” to group the different elements of the dashboard.



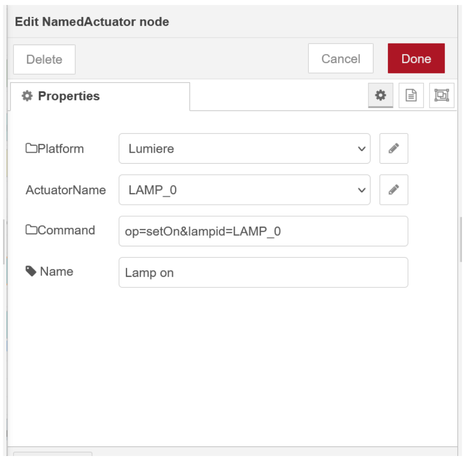
Screenshot of the flow



Screenshot of the dashboard

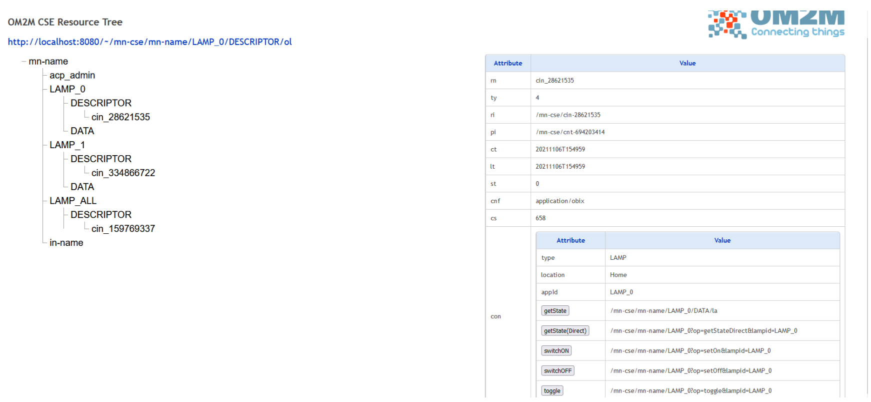
d. Imagine

For the last part, we tried to control a simulated lamp via the oneM2M platform. This lamp is accessible by opening the Sample Simulated IPE in the mn-cse. In order to do that, we added one more type of node: Named Actuator. This node is connected to the actuator LAMP_0 on the following URL: http://127.0.0.1:8181/~/mn-cse/mn-name. We used two nodes of this type to turn on or off the LAMP_0. To turn on the lamp, we used the command “op=setOn&lampid=LAMP_0” and to turn it off, “op=setOff&lampid=LAMP_0”.


Edit NamedActuator node

Platform	Lumiere
ActuatorName	LAMP_0
Command	op=setOn&lampid=LAMP_0
Name	Lamp on

Screenshot of the properties of one of the NamedActuator node


OM2M CSE Resource Tree

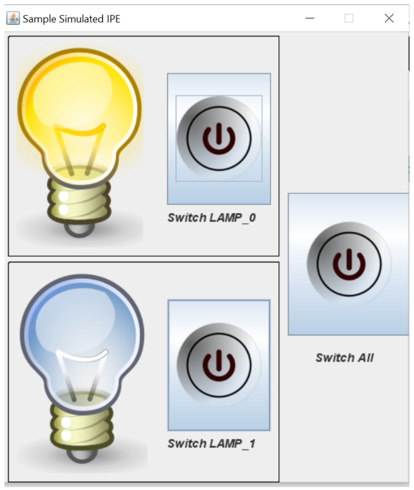
http://localhost:8080/~/mn-cse/mn-name/LAMP_0/DESCRIPTOR/ol

```
mn-name
  acp_admin
  LAMP_0
    DESCRIPTOR
      cin_28621535
    DATA
  LAMP_1
    DESCRIPTOR
      cin_334866722
    DATA
  LAMP_ALL
    DESCRIPTOR
      cin_159769337
  in-name
```

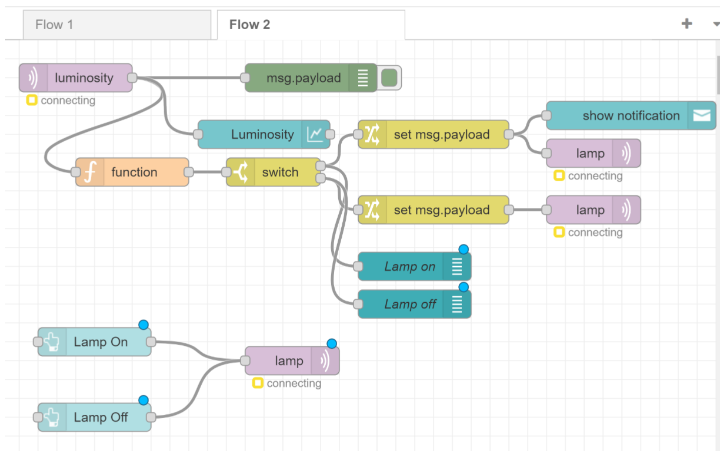
Attribute	Value
rn	cin_28621535
ty	4
ri	/mn-cse/cin-28621535
pi	/mn-cse/cnt-694203414
ct	20211106T154959
lt	20211106T154959
st	0
cnf	application/obix
cs	658
con	

Attribute	Value
type	LAMP
location	Home
appId	LAMP_0
getState	/mn-cse/mn-name/LAMP_0/DATA/la
getState(Direct)	/mn-cse/mn-name/LAMP_0?op=getStateDirect&lampid=LAMP_0
switchON	/mn-cse/mn-name/LAMP_0?op=setOn&lampid=LAMP_0
switchOFF	/mn-cse/mn-name/LAMP_0?op=setOff&lampid=LAMP_0
toggle	/mn-cse/mn-name/LAMP_0?op=toggle&lampid=LAMP_0

Screenshot of the AE LAMP_0 on OM2M



Screenshot of the simulated lamp



Screenshot of the final flow

6. Report

a. Node-RED flow

The file of flow is in the attached files of the email.

b. Benefits and drawbacks

The benefits of an application built with node-RED are that it becomes easy and fast to build an application and to create a dashboard because it has multiple dedicated libraries and embedded graphics such as ready-to-use charts. It is a visual tool that supports MQTT and OM2M. It is also an open source programming tool known for its simplicity and efficiency.

The drawbacks are that it may be limited to the existing node palettes. It lacks flexibility in the display. We are not experts but we did not find a lot of drawbacks to the use of Node-RED.