Official Master Degree
INTERNET OF THINGS

Sensor networks

LoRaWAN connectivity implementation for the embedded plant monitoring IoT system platform using the B-L072Z-LRWAN1 ARM Mbed-based platform

Project 1

V1.2.4

Mariano Ruiz
Eduardo Barrera
Sergio Esquembri
Pedro J. Lobo

2024/25

Universidad Politécnica de Madrid

Table of contents

Table of figures

1 INTRODUCTION

1.1 Document Overview

This document describes the objectives and the specifications of Project 1 to be developed during the "Sensor Networks" course.

1.2 Project Objectives

The main goal of this project is to implement the LoRaWAN connectivity for an embedded Mbed based IoT system to monitor the environmental conditions and health of a plant throughout its life. This could allow vendors, for example, to adjust the final price of particular plants, like bonsais or very delicate plants, for which the environmental conditions they have suffered throughout their lives are crucial. The system could also be useful as a monitoring system in greenhouses by monitoring several plants randomly.

The system should continuously monitor basic environmental parameters, such as temperature, relative humidity, and ambient light, which directly affect the plants' health. Additionally, the soil moisture in which the plant is growing must also be recorded periodically. Even to offer more information, the system will also monitor the plant's colour to have information about its evolution over time. Other essential aspects of the plants' health are the possible issues they have to suffer during storage or transport, such as sudden movements, falls, hits, overturns, and base unevenness. All these parameters should also be monitored. Finally, the IoT system should provide continuous information about the plant's location using GPS.

Figure 1 shows a picture of a possible implementation of the whole system.

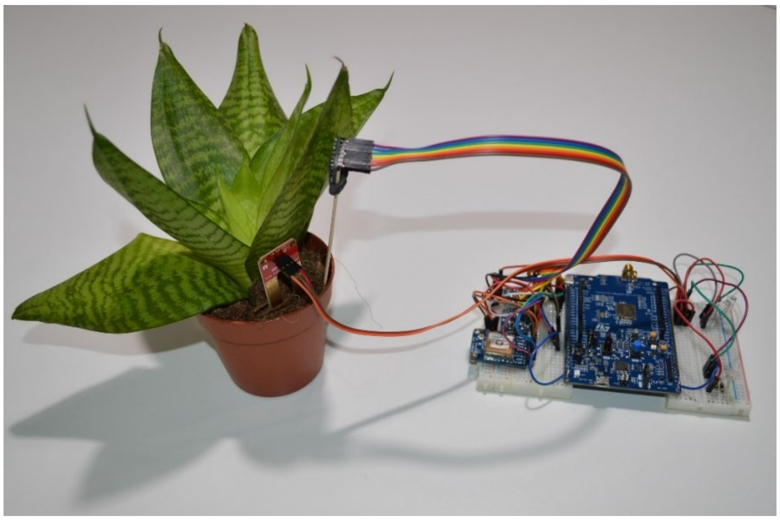


Figure 1. Final implementation of the IoT system

The hardware and software for the sensor management were developed in the course "Embedded platforms and communications for IoT" during the previous bimester. In this course, the LoRaWAN communication module will be added to the previous implementation to send the environmental and health parameters of the plant to the network server.

1.3 Acronyms

ADC	Analog to Digital Converter
COM	Communication port (serial)
EEPROM	Electrical Erasable Programmable Read Only Memory
FSK	Frequency Shift Keying
GPIO	General Purpose Input Output
GPS	Global Positioning System
I/O	Input / Output
I2C	Inter-Integrated Circuit
IDE	Integrated Development Environment
IoT	Internet of Things
LED	Light Emitting Diode
MDK	Microcontroller Development Kit
OOK	On-off keying
OS	Operating system
OTG	On-The-Go
PWM	Pulse-width modulation
RGB	Red-Green-Blue
SPI	Serial Peripheral Interface
UART	Universal Asynchronous Receiver Transmitter
USB	Universal Serial Bus

2 PROJECT SPECIFICATIONS

2.1 Introduction

The system's goal is to send the acquired data from a sensor to the LoRaWAN network server using LoRaWAN/LoRa technology. The basic structure of a LoRaWAN network is depicted in Figure 2. The main parts of the LoRaWAN network are: a) the Nodes, b) the Gateways, and c) the Network and Application Servers.

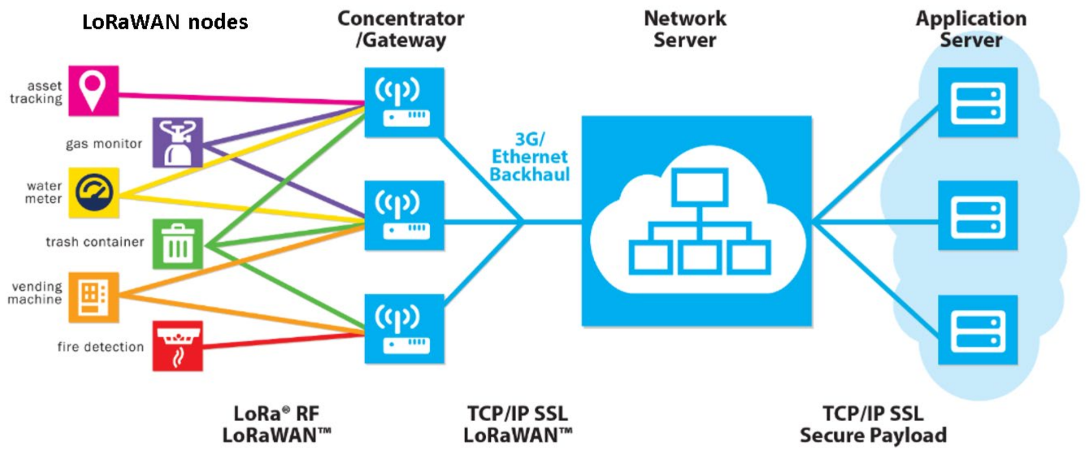


Figure 2. LoRaWAN network

In the case of this development, the **Nodes** will be the hardware based on the *B-L072Z-LRWAN1 LoRa®/Sigfox™ Discovery kit* and the different sensors for monitoring all the required parameters related to the plant's health, developed in the previous course *Embedded Platforms and Communications for IoT*.

There is a *Multitech Conduit* LoRaWAN **Gateway** at the university campus, located at the roof of Building 8, as shown in Figure 3.

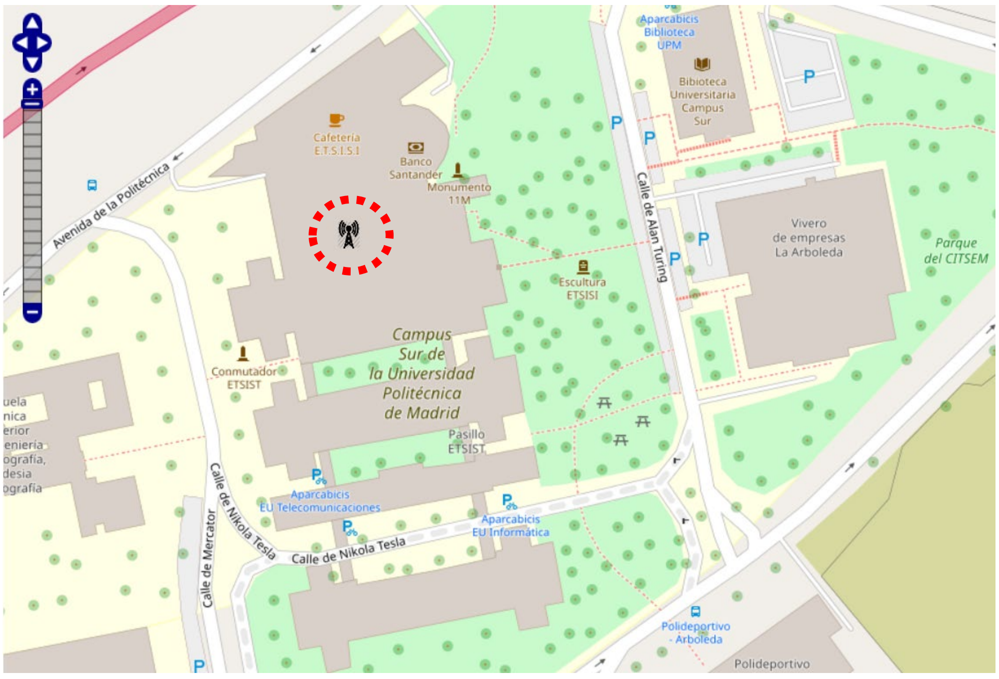

Avenida de la Politécnica
Cafetería E.T.S.I.S.I
Banco Santander
Monumento 11M
Aparcabicis Biblioteca UPM
Biblioteca Universitaria Campus Sur
Calle de Alan Turing
Vivero de empresas La Arboleda
Parque del CITSEM
Escultura ETSISI
Campus Sur de la Universidad Politécnica de Madrid
Conmutador ETSIST
Pasillo ETSIST
Aparcabicis EU Telecomunicaciones
Aparcabicis EU Informática
Calle de Nikola Tesla
Calle de Mercator
Polideportivo Arboleda
Polideportivo

Figure 3. LoRaWAN Gateway location at the University Campus (Building 8, roof)

The **Network Server** used for the development of this project will be the *ResIOT.io™ - LoRaWAN Network Server and IoT Platform* (https://www.resiot.io/en/), which allows monitoring the system evolution (Figure 4).

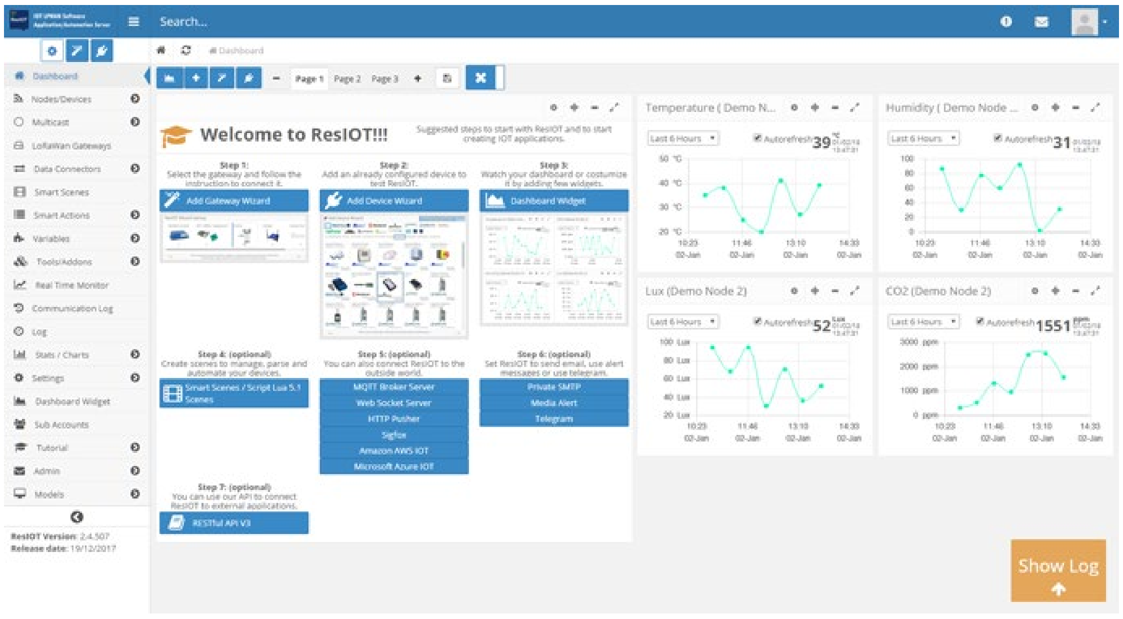


Figure 4. ResIOT *LoRaWAN Network Server and IoT Platform* dashboard

2.2 First phase: analysis of mbed-os-example-LoRaWAN

The first phase of the project will consist of analyzing and assessing the example provided by the instructors for LoRaWAN based Mbed systems.

The required steps are:

1. Download the Mbed example from the course Moodle site link. This is an example provided by the instructors with customized mbed-os from the original mbed-os-example-LoRaWAN (https://github.com/ARMmbed/mbed-os-example-lorawan). Please, USE ONLY THE EXAMPLE PROVIDED BY THE INSTRUCTORS.
2. Unzip the downloaded file, open the project in Mbed Studio, and examine the project structure. You should see something similar to Figure 5.

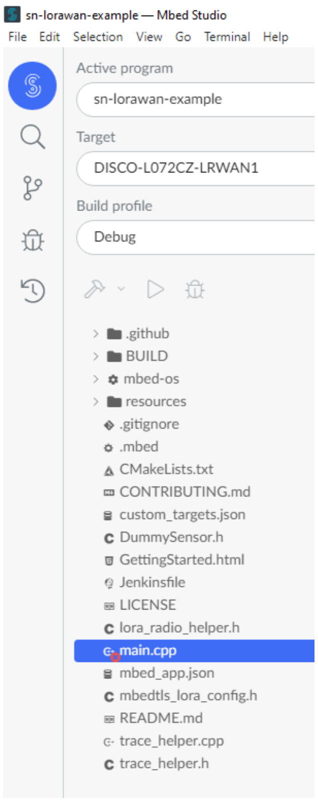
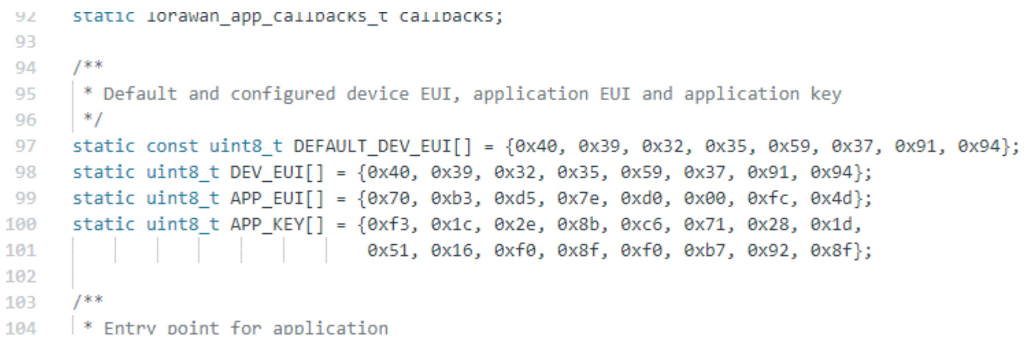


Figure 5. LoRaWAN mbed-os project structure

3. Open the `main.cpp` file and examine it.

   The default device EUI is defined in file `mbed_config.h`, in the macro `MBED_CONF_LORA_DEVICE_EUI`, which in turn is generated from the contents of mbed_app.json; but you can see how its value is overridden later in the code (Figure 6). As you know, the device EUI must be different for every device in the LoRaWAN network. Therefore, you must configure the device identification of your node, updating the value of `DEV_EUI` according to the information provided in the file *ResIOT student accounts* located on the course website.



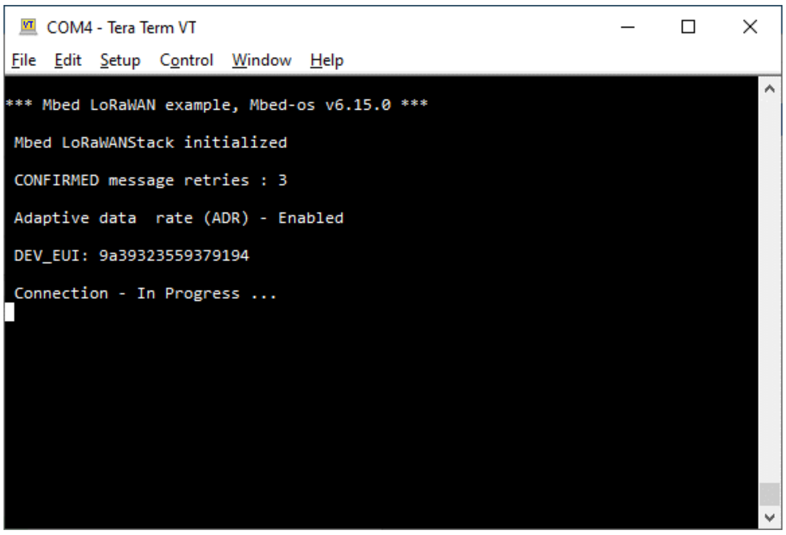
```
static lorawan_app_callbacks_t callbacks;

/**
 * Default and configured device EUI, application EUI and application key
 */
static const uint8_t DEFAULT_DEV_EUI[] = {0x40, 0x39, 0x32, 0x35, 0x59, 0x37, 0x91, 0x94};
static uint8_t DEV_EUI[] = {0x40, 0x39, 0x32, 0x35, 0x59, 0x37, 0x91, 0x94};
static uint8_t APP_EUI[] = {0x70, 0xb3, 0xd5, 0x7e, 0xd0, 0x00, 0xfc, 0x4d};
static uint8_t APP_KEY[] = {0xf3, 0x1c, 0x2e, 0x8b, 0xc6, 0x71, 0x28, 0x1d,
                            0x51, 0x16, 0xf0, 0x8f, 0xf0, 0xb7, 0x92, 0x8f};

/**
 * Entry point for application
```

Figure 6. main.cpp file detail

4. Compile the project and check for possible errors.
5. Connect the hardware to the computer and download the project.
6. Open and configure TeraTerm or equivalent terminal application, connect it to the Mbed COM and run the application. You should see the messages displayed in Figure 7.

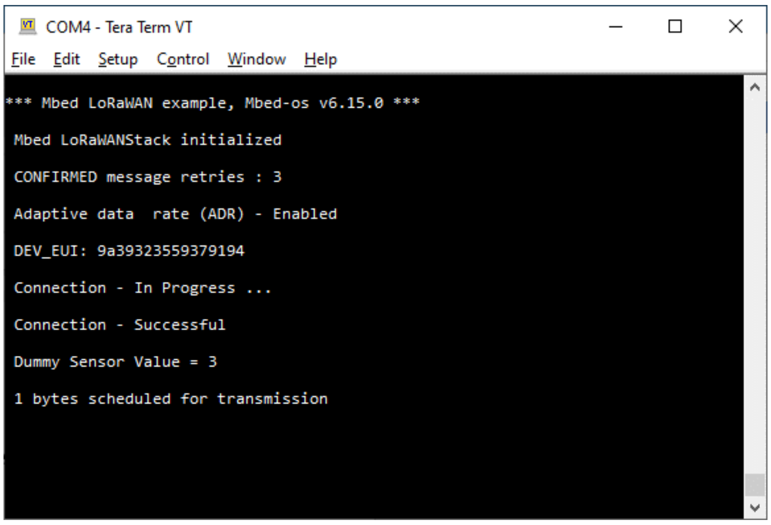


```
*** Mbed LoRaWAN example, Mbed-os v6.15.0 ***
 Mbed LoRaWANStack initialized
 CONFIRMED message retries : 3
 Adaptive data  rate (ADR) - Enabled
 DEV_EUI: 9a39323559379194
 Connection - In Progress ...
```

Figure 7. Messages before connection

7. After some time (up to a few minutes if many devices are trying to join the network simultaneously), the connection should be established, and the messages should start being sent. You should see the messages displayed in Figure 8.

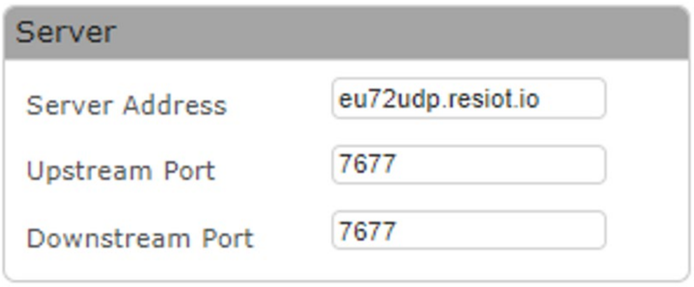
[VERY IMPORTANT]: The communication between the node and ResIOT requires an available connection with the gateway. This can stop or slow the development and testing if no gateway is available, there is a poor signal between node and gateway, or the gateway becomes saturated by device connections. Continue reading the manual for further instructions for development and testing using ResIOT tools.



```
*** Mbed LoRaWAN example, Mbed-os v6.15.0 ***
 Mbed LoRaWANStack initialized
 CONFIRMED message retries : 3
 Adaptive data  rate (ADR) - Enabled
 DEV_EUI: 9a39323559379194
 Connection - In Progress ...
 Connection - Successful
 Dummy Sensor Value = 3
 1 bytes scheduled for transmission
```

Figure 8. Messages after connection

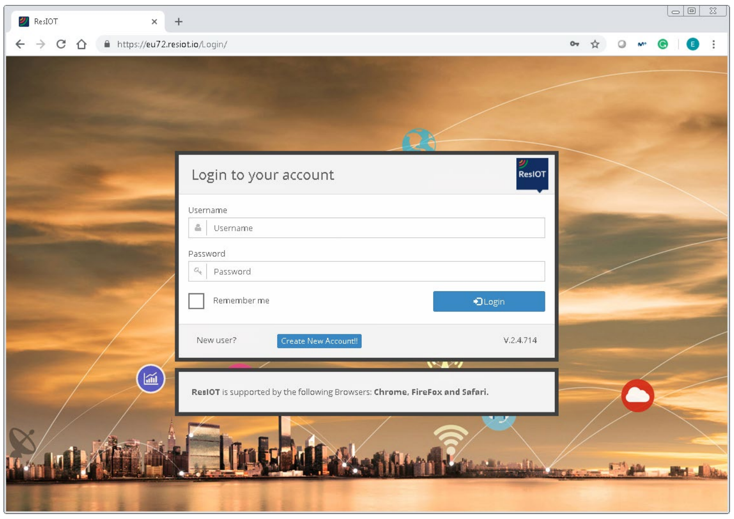
8. Connect to the LoRaWAN gateway and examine the configuration (ask your instructor how to do this). The gateway is configured in PACKET FORWARDER mode. The messages are forwarded to the specified server address (Figure 9).


Server

Server Address	eu72udp.resiot.io
Upstream Port	7677
Downstream Port	7677

Figure 9. Server configuration for packet forwarding

9. As explained above, the network server will be the *ResIOT.io™ - LoRaWAN Network Server and IoT Platform*. Each student has an account in the network server. Check the file *ResIOT student accounts* located on the course website for more information. Connect to https://eu72.resiot.io/ and log into the system.



Login to your account

Username: Username

Password: Password

Remember me

Login

New user? Create New Account!

V.2.4.714

ResIOT is supported by the following Browsers: Chrome, FireFox and Safari.

Figure 10. ResIOT login page

10. When you have logged in the application, you should see the ResIOT Dashboard, similar to the one displayed in Figure 11. You can see several Tabs on the dashboard. Test Node tab shows several widgets with the values of different parameters provided by the instructor's node. Examine the dashboard, but, please, do not modify anything.

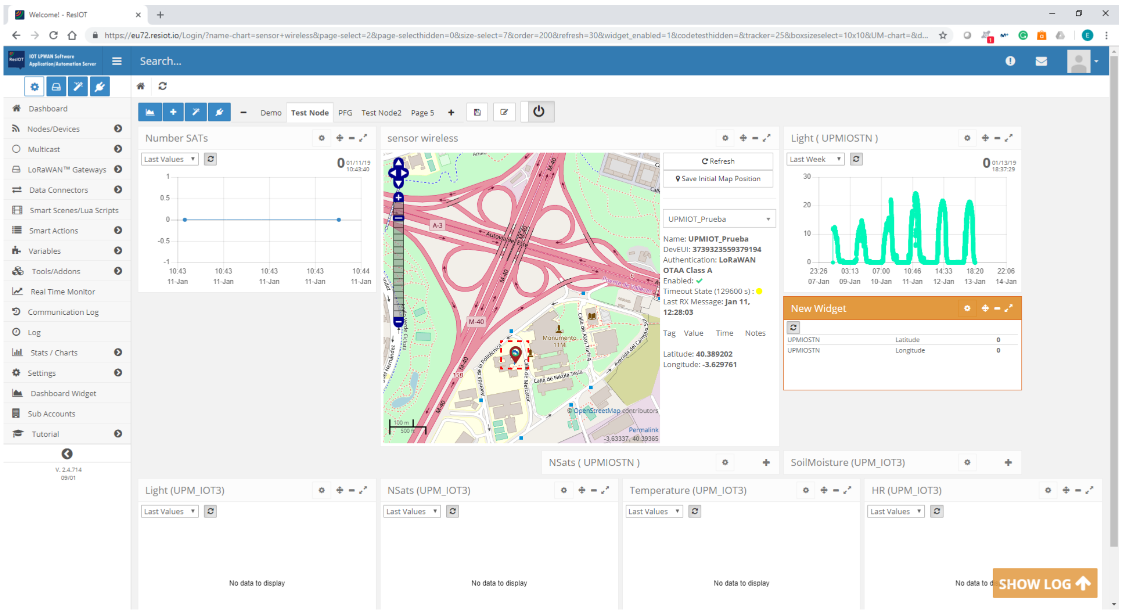
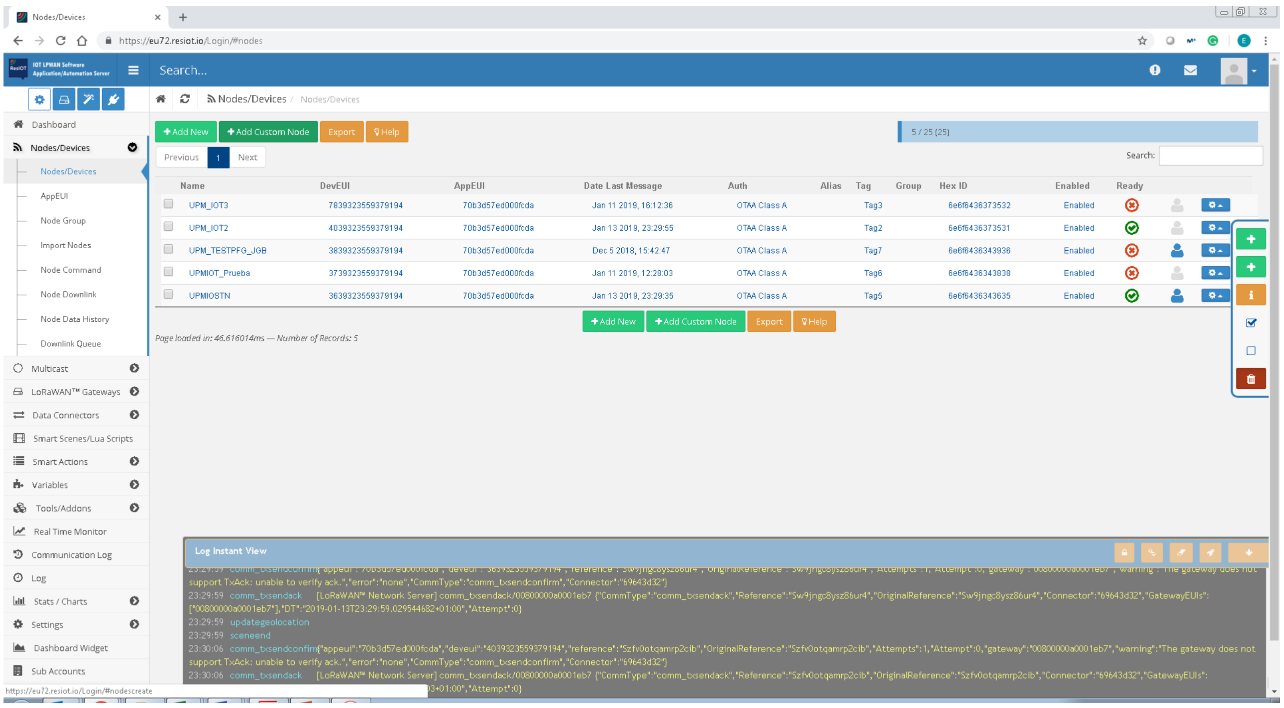


Figure 11. ResIOT Dashboard

11. Create a new tab with the name **SN_GROUP_n**, where *n* is the letter assigned to your group in the file *ResIOT student accounts* located on the course website. Be aware that all of you can modify the configuration of all tabs, so **PLEASE MODIFY ONLY YOUR TAB**.
12. Add a new Node in the network server with your device identification using the Nodes/Devices option (Figure 12). The name of the node must be **NODE_SN_GROUP_n**, where *n* is again the letter assigned to your group.



Name	DevEUI	AppEUI	Date Last Message	Auth	Alias	Tag	Group	Hex ID	Enabled	Ready
UPM_IOT3	7839323559379194	70b3d57ed000fcda	Jan 11 2019, 16:12:36	OTAA Class A		Tag3		6e6f6436373532	Enabled	✗
UPM_IOT2	4039323559379194	70b3d57ed000fcda	Jan 13 2019, 23:29:55	OTAA Class A		Tag2		6e6f6436373531	Enabled	✓
UPM_TESTPFG_JGB	3839323559379194	70b3d57ed000fcda	Dec 5 2018, 15:42:47	OTAA Class A		Tag7		6e6f6436343936	Enabled	✗
UPMIOT_Prueba	3739323559379194	70b3d57ed000fcda	Jan 11 2019, 12:28:03	OTAA Class A		Tag6		6e6f6436343838	Enabled	✗
UPMIOSTN	3639323559379194	70b3d57ed000fcda	Jan 13 2019, 23:29:35	OTAA Class A		Tag5		6e6f6436343635	Enabled	✓

Page loaded in: 46.616014ms — Number of Records: 5

```
support TxAck: unable to verify ack.","error":"none","CommType":"comm_txsendconfirm","Connector":"69643d32"}
23:29:59  comm_txsendack    [LoRaWAN™ Network Server] comm_txsendack/00800000a0001eb7 {"CommType":"comm_txsendack","Reference":"Sw9jngc8ysz86ur4","OriginalReference":"Sw9jngc8ysz86ur4","Connector":"69643d32","GatewayEUIs":
["00800000a0001eb7"],"DT":"2019-01-13T23:29:59.029544682+01:00","Attempt":0}
23:29:59  updategeolocation
23:29:59  sceneend
23:30:06  comm_txsendconfirm{"appeui":"70b3d57ed000fcda","deveui":"4039323559379194","reference":"Szfv0otqamrp2cib","OriginalReference":"Szfv0otqamrp2cib","Attempts":1,"Attempt":0,"gateway":"00800000a0001eb7","warning":"The gateway does not
support TxAck: unable to verify ack.","error":"none","CommType":"comm_txsendconfirm","Connector":"69643d32"}
23:30:06  comm_txsendack    [LoRaWAN™ Network Server] comm_txsendack/00800000a0001eb7 {"CommType":"comm_txsendack","Reference":"Szfv0otqamrp2cib","OriginalReference":"Szfv0otqamrp2cib","Connector":"69643d32","GatewayEUIs":
3+01:00","Attempt":0}
```

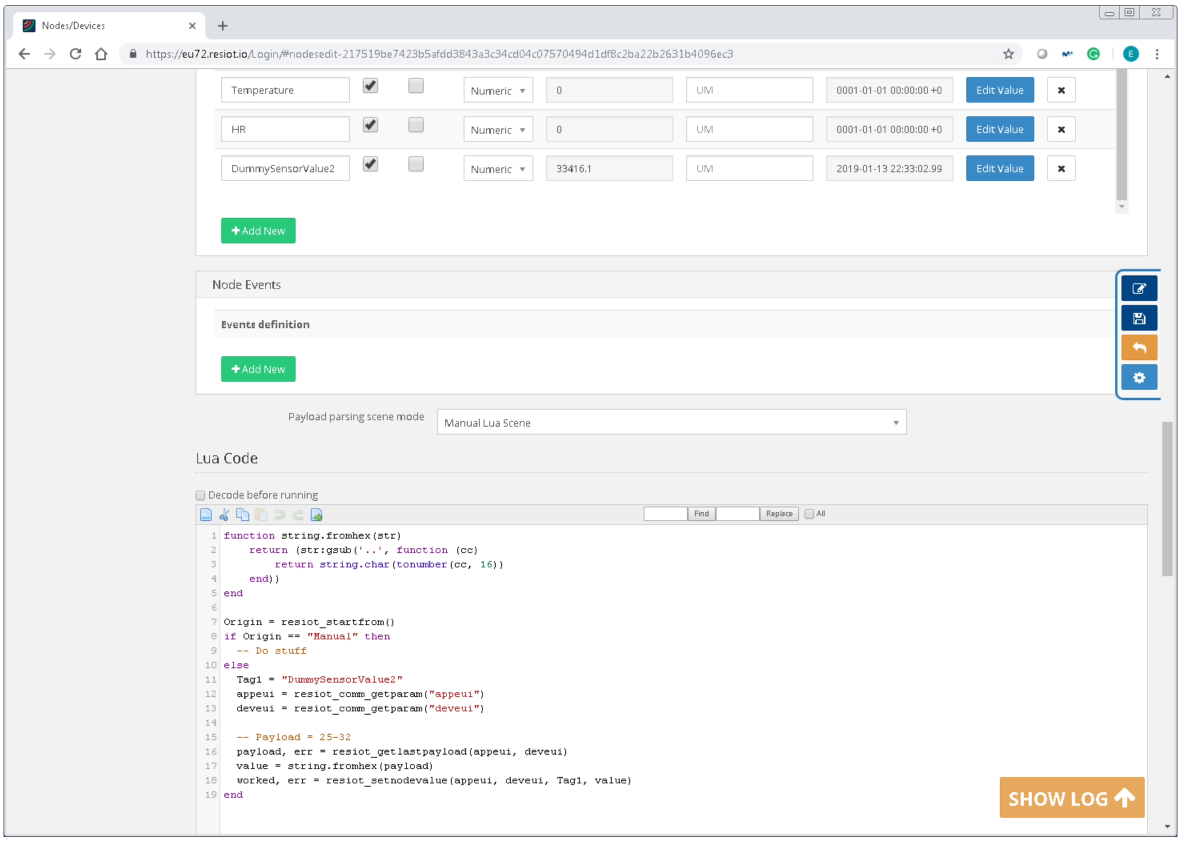
Figure 12. Nodes/Devices page in ResIOT

13. Add the next LUA code to retrieve the information sent by your Mbed node (Figure 13). Substitute “GG” in “SN_TEST_GG” with your group letter (e.g. “SN_TEST_A”) or you will not be able to tell apart your log messages from those of all your classmates:

```lua
--You can define your own functions as is done here for parsePayload
function parsePayload(appeui,deveui,payload)
      Tag1 = "dummy"
      strvalue = resiot_hexdecode_ascii(payload)
      --Call for LUA Script engine prints
      resiot_debug(string.format("SN_TEST_GG Tag: %s  Strvalue: %s \n",Tag1, strvalue))
      value = string.match(strvalue, 'Dummy Sensor Value is ([0-9]+)')
      resiot_debug(string.format("SN_TEST_GG Tag: %s  Value: %s \n",Tag1, value))
      worked, err = resiot_setnodevalue(appeui, deveui, Tag1, value)
      if (not worked) then
             resiot_debug(string.format("SN_TEST_GG Set Value Error %s \n",err))
      else
             resiot_debug("SN_TEST_GG Set Node value successful\n")
      end
end

Origin = resiot_startfrom() --Scene process starts here

if Origin == "Manual" then
      -- Manual script execution for testing
      -- Set your test payload here in hexadecimal
      payload = "44756d6d792053656e736f722056616c756520697320323437"
      -- Set your Application EUI here
      appeui = "70b3d57ed000fc4d"
      -- Set your own Device EUI here
      deveui = "4039323559379194"
else
      -- Normal execution, get payload received from device
      appeui = resiot_comm_getparam("appeui")
      deveui = resiot_comm_getparam("deveui")
      payload, err = resiot_getlastpayload(appeui, deveui)
      resiot_debug("SN_TEST_GG Test Auto Mode\n")
end
-- Do your stuff
parsePayload(appeui,deveui,payload)
```


Nodes/Devices
https://eu72.resiot.io/Login/#nodesedit-217519be7423b5afdd3843a3c34cd04c07570494d1df8c2ba22b2631b4096ec3
Node Events
Events definition
+ Add New
Find
Replace
All
SHOW LOG

Temperature	✓	☐	Numeric	0	UM	0001-01-01 00:00:00 +0	Edit Value
HR	✓	☐	Numeric	0	UM	0001-01-01 00:00:00 +0	Edit Value
DummySensorValue2	✓	☐	Numeric	33416.1	UM	2019-01-13 22:33:02.99	Edit Value

Payload parsing scene mode: Manual Lua Scene

Lua Code

☐ Decode before running

```
function string.fromhex(str)
    return (str:gsub('..', function (cc)
        return string.char(tonumber(cc, 16))
    end))
end

Origin = resiot_startfrom()
if Origin == "Manual" then
  -- Do stuff
else
  Tag1 = "DummySensorValue2"
  appeui = resiot_comm_getparam("appeui")
  deveui = resiot_comm_getparam("deveui")

  -- Payload = 25-32
  payload, err = resiot_getlastpayload(appeui, deveui)
  value = string.fromhex(payload)
  worked, err = resiot_setnodevalue(appeui, deveui, Tag1, value)
end
```

Figure 13. LUA code for the mbed-os-example-LoRaWAN

14. The provided code allows you to test the payload parsing script even without a device connected by setting your own payload value. As can be seen, the payload is given as a hexadecimal chain value. You can modify the payload to test with different values.
15. In order to test whether your application might work even without a connection between node and gateway, modify the node application to obtain the payload that would be sent to the gateway even if no connection with the gateway is detected.
16. Go to the dashboard, select your tab, and add a widget to display the value of the data sent by your node. You should obtain something like what is displayed in Figure 14.

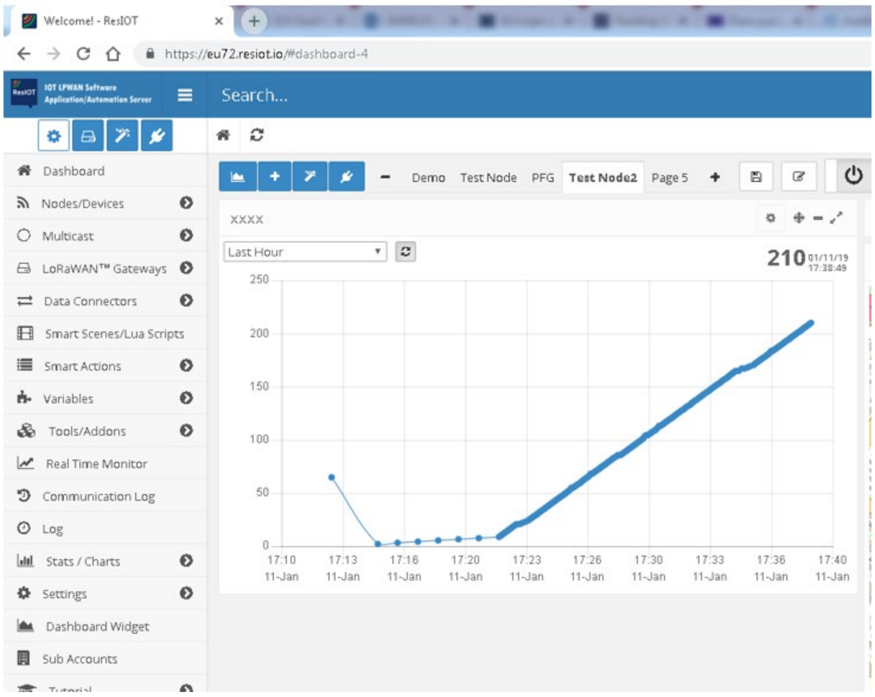


Figure 14. Data from mbed-os-example-LoRaWAN

17. ResIOT provides free mobile applications for Android and IOS. Install the application on your mobile phone.
18. Configure the LoRaWAN ResIOT server with your credentials (Figure 15).

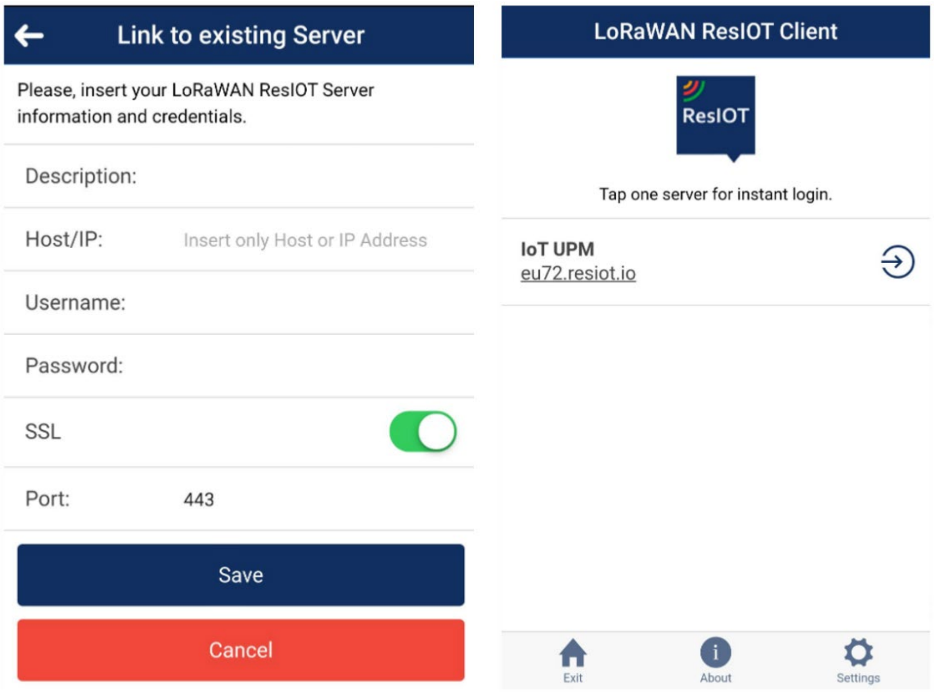

Link to existing Server
Please, insert your LoRaWAN ResIOT Server information and credentials.
Description:
Host/IP: Insert only Host or IP Address
Username:
Password:
SSL
Port: 443
Save
Cancel
LoRaWAN ResIOT Client
ResIOT
Tap one server for instant login.
IoT UPM
eu72.resiot.io
Exit
About
Settings

Figure 15. ResIOT mobile application configuration

19. Log into the system and go to your dashboard tab. You can see and manage your data using different widgets (Figure 16).

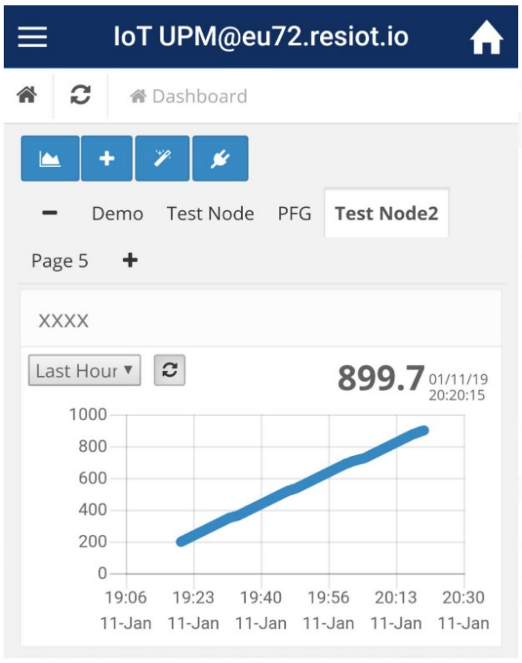
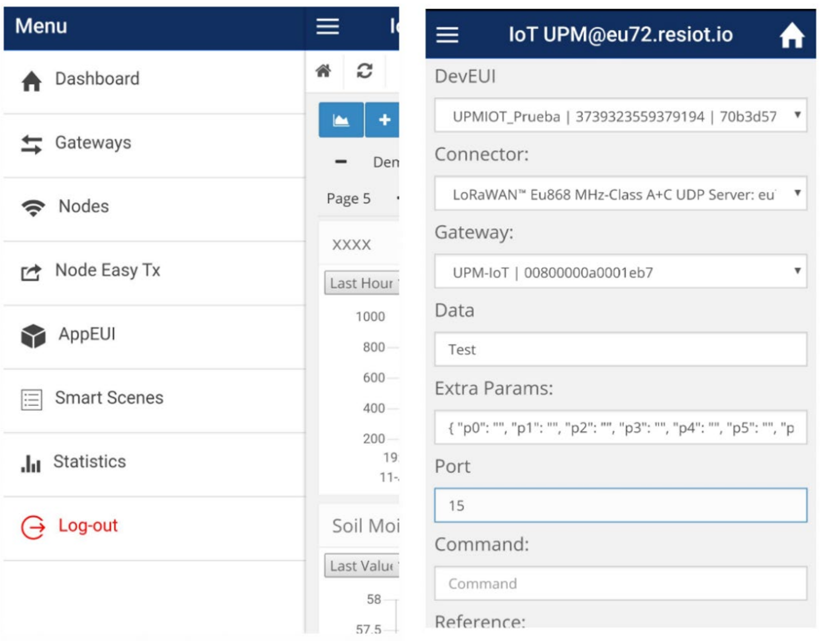


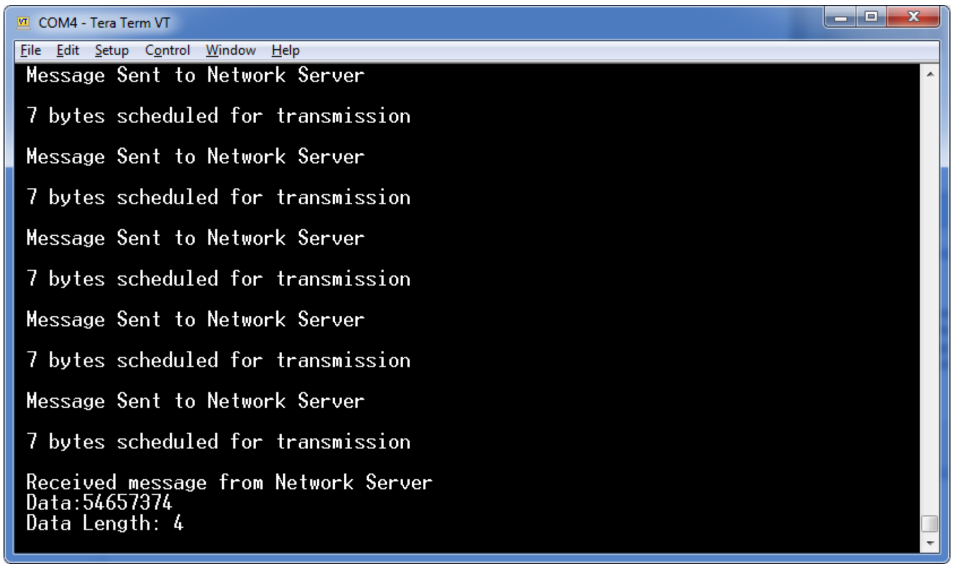
Figure 16. Data in ResIOT mobile application

20. It is also possible to send commands and data from the ResIOT server to the nodes. Go to the *Node Easy Tx* option of the menu, configure your node/gateway parameters, and send the data "Test" (Figure 17). You can use the TeraTerm window to check if the data has arrived at your node. In Figure 18 you can see the received data 54657374, which corresponds with the hexadecimal ASCII code of the 4 characters of the sent data "Test" ("T"-0x54, "e"-0x65, "s"-0x73, "t"-0x74). This option is also available on the ResIOT Webpage under *Nodes/Devices - Node Downlink*.



DevEUI
UPMIOT_Prueba | 3739323559379194 | 70b3d57
Connector:
LoRaWAN™ Eu868 MHz-Class A+C UDP Server: eu
Gateway:
UPM-IoT | 00800000a0001eb7
Data
Test
Extra Params:
{ "p0": "", "p1": "", "p2": "", "p3": "", "p4": "", "p5": "", "p
Port
15
Command:
Command
Reference:

Figure 17. ResIOT data transmission to nodes



```
Message Sent to Network Server
7 bytes scheduled for transmission
Message Sent to Network Server
7 bytes scheduled for transmission
Message Sent to Network Server
7 bytes scheduled for transmission
Message Sent to Network Server
7 bytes scheduled for transmission
Message Sent to Network Server
7 bytes scheduled for transmission
Received message from Network Server
Data:54657374
Data Length: 4
```

Figure 18. Data received at node

2.3 Second phase: adding the plant's health monitoring application

The second phase of the project will be focused on integrating the plant's health monitoring application developed by the students in the course “Embedded platforms and communications for IoT”. In this case, the target will be to send the location and at least 3 environmental parameters (temperature, humidity, soil moisture, light, etc.) to the ResIOT network server. Note that the maximum message length is 30 characters (bytes), so you be careful about the data encoding. Additionally, the node should be able to receive commands to change the colour of the RGB LED.

The required steps are:

1. Add to your mbed-os-example-lorawan project the code (from your previous plant's health monitoring project) to get the GPS location from the GPS module and to send it to the network server.

> **[VERY IMPORTANT]:** If the GPS signal is not available, you may send predefined coordinates value while the GPS signal is not fixed, but don’t use the coordinates of your actual location or you won’t be able to tell whether they are fixed or come really from the GPS receiver.

2. Go to your node configuration in ResIOT and modify the LUA code to retrieve the values of the latitude and longitude coordinates.

> **[VERY IMPORTANT]:** Remember to print the payload using the SerialPC interface of the node in order to test the script even without a connection between node and gateway.

3. Go to your dashboard tab in the ResIOT server and add a new widget to see the location of your node in a map.
4. Add to your mbed-os-example-lorawan project the code to get at least 3 environmental parameters from the sensors and send them to the network server. You can choose which parameters to use: temperature, light, soil moisture, etc.
5. Go to your node configuration in ResIOT and modify the LUA code to retrieve the parameter values.
6. Go to your dashboard tab in the ResIOT server and add new widgets to show the parameter values.
7. Test the system.

8. Modify the Mbed code to allow changing the colour of the RGB LED according to the command received from the ResIOT network server. The available commands must be: “OFF”, “Green”, and “Red”.
9. Using the ResIOT mobile application or the Webpage, send any of the defined commands and test the system.

2.4 Third phase: improving the plant's health monitoring application (OPTIONAL)

The third phase of the project will be focused on the improvement of the application to be able to send as many parameters as possible. Remember that the maximum message length is 30 characters (bytes).
The required steps are:

1. Add to the current project all the code from your previous project developed in the course “Embedded platforms and communications for IoT” to get all the parameters from all sensors.
2. Convert the data format of the system parameters from string to the most appropriate type to reduce their length. For instance, you can convert the latitude and longitude from string to float to use only 4 bytes per parameter. Configure the data to be able to send the maximum number of parameters.
3. Go to your node configuration in ResIOT and modify the LUA code to retrieve the values of the parameters defined in the previous point.
4. Go to your dashboard tab in the ResIOT server and add new widgets to see the new parameter values.
5. Test the system.

You can also add any enhancements beyond this specification that you might think of.

2.5 Evaluation criteria

The evaluation of Project 1 is based on three elements: a (short) functional assessment, the source code and a technical report. You are required to upload the source code of your project to the course web site before the start of the functional assessment session, and the technical report a few days later. You can see the exact dates in the next section, “Project schedule”.

The source code will be evaluated according to its functionality (Are the parameters well delivered? Is the ResIOT interface friendly and meaningful?), stability (Is the system stable? How resilient is it to service disruptions and errors?), modularity (Is the code well structured? Is it maintainable? Can it be easily modified or extended?) and efficiency (How large is the program? Are there unnecessary data manipulations? How efficient is the data encoding?).

2.6 Project schedule

- Project 1 will be developed between December 5th, 2024 and December 19th, 2024.
- The functional assessment of the project will be done during the last session of Project 1 on Thursday, December 19th, 2024.
- The **final code** of the project must be uploaded to the course website before the assessment session begins on **December 19th, 2024, at 18:00**.
- The **technical documentation** of the project development must be uploaded to the course website by Monday, **January 13th, 2025, at 8:30 AM**.