

Curso Placas Robóticas 2020

CIFO La Violeta

LORA GPS TRACKING

MEMORIA

Autor	Fco. Javier Pérez Vicente
Versión	01.00
Fecha documento	27-11-2020

Tabla de contenido

1	Introducción	3
1.1	Objetivos	3
2	Arquitectura de la solución	3
3	Componentes necesarios.....	5
3.1	Hardware	5
3.2	Software	6
4	Comprobación cobertura LoRaWAN	6
4.1	Montaje MKR 1310 + RGB Led	7
5	Implementación del sistema de Tracking GPS LORA	8
5.1	Sistema base	8
5.2	Thethingsnetwork (TTN).....	8
5.2.1	<i>Registrar dispositivos MKR 1310 y LGT92 en TTN</i>	<i>8</i>
5.3	Node-red.....	10
5.3.1	<i>Instalación</i>	<i>10</i>
5.3.2	<i>Configuración</i>	<i>11</i>
5.4	InfluxDB.....	13
5.4.1	<i>Instalación InfluxDB</i>	<i>13</i>
5.4.2	<i>Configurar InfluxDB.....</i>	<i>14</i>
5.4.3	<i>Conectar Node-RED a InfluxDB</i>	<i>15</i>
5.5	Graficado	16
5.5.1	<i>Instalar y configurar Grafana</i>	<i>16</i>
5.5.2	<i>Conectar Grafana a InfluxDB</i>	<i>16</i>
5.5.3	<i>Instalar el plugin Trackmap de Grafana.....</i>	<i>17</i>
5.5.4	<i>Añadir Dashboard.....</i>	<i>17</i>
6	Resultados	19
7	Anexos.....	19
7.1	Sketch cobertura LoraWAN	19
7.2	Payload Decoder TTN	21
8	Bibliografía.....	23

1 Introducción

El objetivo del proyecto es disponer de un **rastreador GPS de bajo consumo, partiendo de un dispositivo LoRaWAN con módulo GPS y dibujar el recorrido en un mapa en tiempo "real"**.

ThethinsNetwork (TTN) es una red LoraWan *opensource* que proporciona una infraestructura descentralizada para el Internet de las Cosas (Iot).

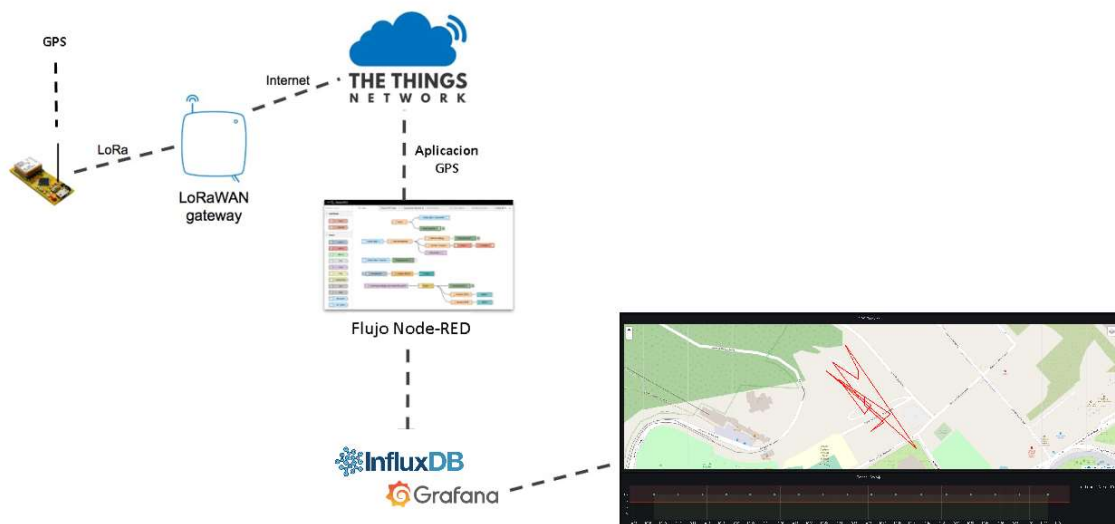
Proporciona una consola para generar la carga útil que se recibe de los dispositivos. Es funcional pero no es muy inteligible. Se busca una forma de "dibujar" la información útil (estado batería, recorrido) que vaya reportando dicho dispositivo en un mapa en tiempo "real".

1.1 Objetivos

Objetivo/s:

1. Describir en un mapa el recorrido del dispositivo.
2. Monitorizar el Estado de la batería
3. Quizás conectar algún sensor útil (objetivo secundario)

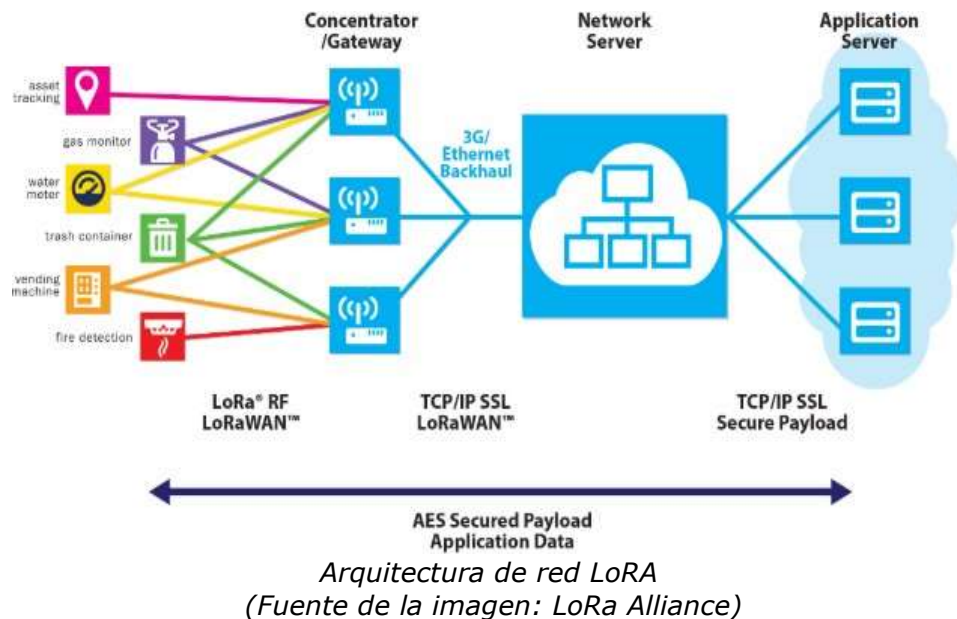
2 Arquitectura de la solución



Componentes:

- **GPS** (Global Positioning System) es un sistema que permite ubicar sobre la tierra cualquier objeto (una persona, un vehículo) con una precisión de hasta centímetros.

- **LoRa** (Long Range) es la capa física de la red conocida como LoRaWAN (Long Range Wide Area).



Conforma una red en forma de estrella. Son redes fáciles de implementar y gestionar ya que no necesitan elementos de enrutamiento.

Los **gateways** retransmiten la información que procede de los dispositivos finales a un servidor de red. De esta forma las estaciones base funcionan a modo de puentes lo que resulta en un diseño muy sencillo de red.

Es una red LPWAN (Low Power Wide Area Network), una red inalámbrica que permite transmitir pequeñas cantidades de datos (0.3 kbps a 50 kbps) a grandes distancias. Esto la invalida para transmisiones de datos constantes, para realizar llamadas y para enviar texto.

Ventajas

- **Larga duración de la batería:** la baja señalización permite una duración de las baterías de años.
 - **Bajo coste:** los protocolos simplificados y livianos de LPWAN reducen la complejidad en el diseño hardware reduciendo los costes.
 - **Cobertura amplia:** el alcance operativo de LPWAN varía desde pocos kilómetros en áreas urbanas hasta **más de 10 km en entornos rurales**. También permite una efectiva comunicación de datos en ubicaciones interiores y subterráneas.
 - **Baja potencia:** optimizados para el consumo de energía, los transceptores LPWAN pueden funcionar con baterías pequeñas y económicas hasta por 20 años.
- **LoraWAN** es el protocolo de red que usa la tecnología LoRa para comunicar y administrar dispositivos LoRa, es la capa de acceso al medio.
 - **TheThingsNetwork (TTN):** es una infraestructura de código abierto cuyo objetivo es proporcionar una cobertura de red LoRaWAN gratuita.

3 Componentes necesarios

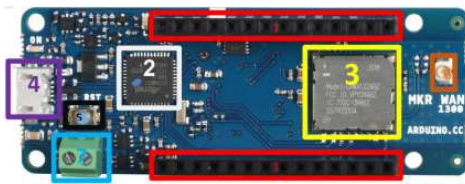
A continuación se muestran los componentes necesarios para el proyecto, tanto **hardware** (placa MKR 1310, led RGB y placa Dragino lgt92 con módulo GPS) como **software** (sistema operativo, node-red, Influxdb y grafana).

3.1 Hardware

Nota:

Para el presente proyecto **no fue posible disponer, como hubiera sido deseable, de un *shield* GPS para Arduino MKR 1310. En su lugar, para el mapeo de coordenadas, se empleó el rastreador GPS de código abierto Dragino LGT92.**

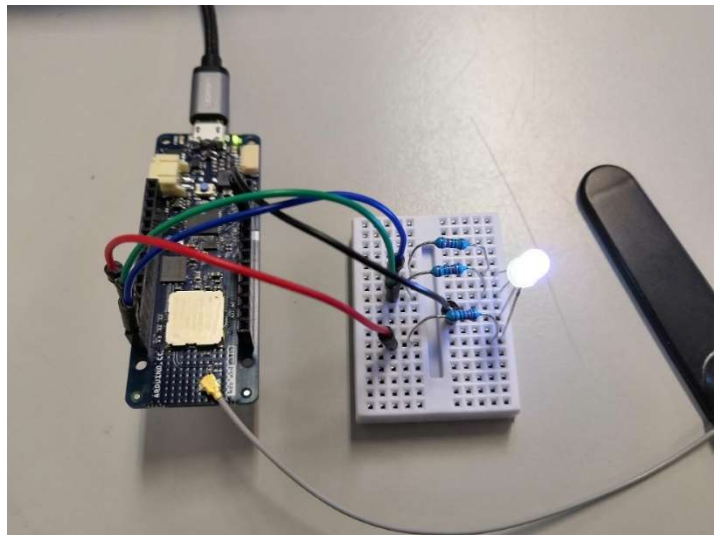
Arduino MKR 1310



RGB Led



Montaje MKR 1310 + RGB Led



Dragino LGT92

Es un rastreador GPS de código abierto con micro controlador de baja energía STM32L072, módulo LoRa SX1276/1278 y módulo GPS L76-L.



3.2 Software ---

Las aplicaciones necesarias para el proyecto se ejecutarán sobre una máquina virtual (MV) **Ubuntu 18.x LTS**.

- **Node-Red** es una herramienta de programación visual para conectar dispositivos de hardware, API y servicios en línea.
- **InfluxDB** para almacenar los datos que vaya informando el dispositivo en TTN.
- **Grafana** para graficar y mostrar en un mapa en tiempo real el recorrido que ha realizado el dispositivo.

4 Comprobación cobertura LoRaWAN

Para comprobar la cobertura LoRaWAN, se armó un sistema que mostrase visualmente cuándo se envían o reciben mensajes.

Se hicieron cambios en el sketch original LoRasendandreceive (ver anexo) para simplificarlo:

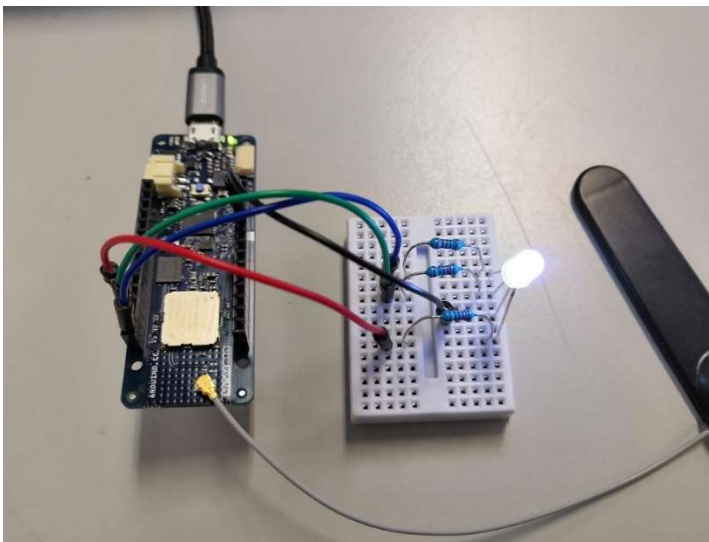
- Se cambió el factor de propagación (Spreading Factor (SF)) para obtener mayor frecuencia de comunicación.
 - «spreading factor» (SF): define el número de bits usados para codificar un símbolo. A mayor SF, menor velocidad de transferencia tendremos pero mayor inmunidad a interferencias.

- Se seleccionó SF7 para la frecuencia de comunicación, esto permite una frecuencia de comunicación cada 6 segundos respetando el ciclo de trabajo. Básicamente podemos actualizar nuestra posición cada 10 segundos.
- si la comunicación falla en SF7, el firmware envía un segundo frame y un tercer frame en diferentes SF hasta 12.

Disponemos de un *sketch* que gestiona la confirmación, reintento y desconexión de la transmisión. Normalmente enviaba una trama cada 14-15 segundos.

4.1 Montaje MKR 1310 + RGB Led

Se agregó un led RGB para mostrar el resultado.



Se añaden 3 resistencias de 1kohm en cada uno de los cátodos para limitar la corriente.

El ánodo está conectado a VCC

Los cátodos están conectados a GPIO 3/4/5.

Sketch empleado (ver anexos al final del documento)

El código se modifica para tener:

- **luz verde** cuando se confirma la comunicación.
- **Luz roja** cuando la comunicación ha fallado.
- **Azul** cuando el dispositivo está desconectado.

5 Implementación del sistema de Tracking GPS LORA

5.1 Sistema operativo

1. Instalar y configurar máquina virtual Ubuntu server LTS 18.XX
2. Poner el sistema en hora:

```
timedatectl list-timezones | grep Europe
```

```
Europe/Amsterdam
```

```
...
```

```
Europe/Luxembourg
```

```
Europe/Madrid
```

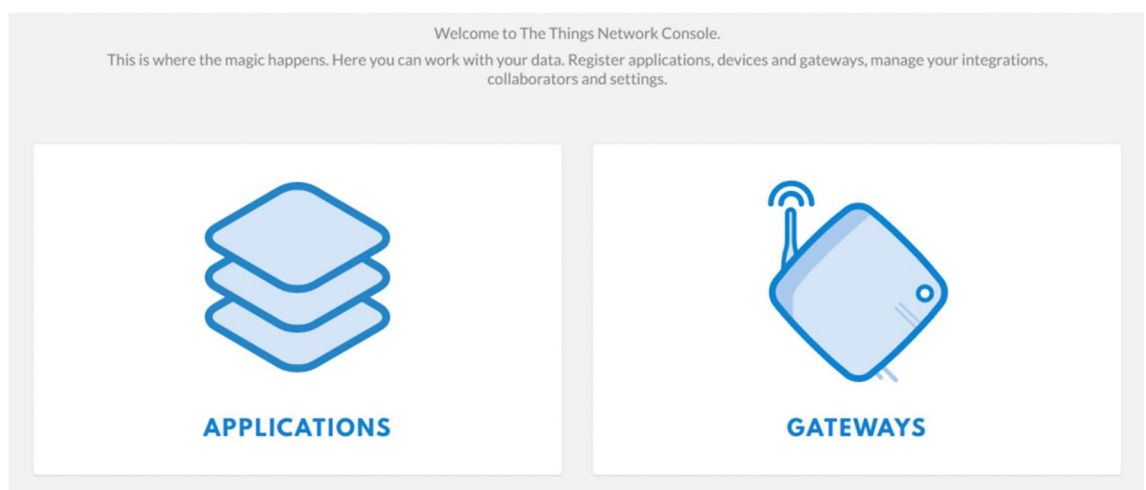
```
sudo timedatectl set-timezone Europe/Madrid
```

5.2 Thethingsnetwork (TTN)

5.2.1 Registrar dispositivos MKR 1310 y LGT92 en TTN

Es necesario tener una cuenta en TTN para Añadir MKR 1310 y LGT92 a la red TTN.

1. Crear Aplicación



ADD APPLICATION

Application ID
The unique identifier of your application on the network

gpsapp

Description
A human readable description of your new app

Eg. My sensor network application

Application EUI
An application EUI will be issued for The Things Network block for convenience, you can add your own in the application settings page.

EUI issued by The Things Network

Handler registration
Select the handler you want to register this application to

ttn-handler-eu

2. Registrar 2º dispositivo

THE THINGS NETWORK
CONSOLE
COMMUNITY EDITION

Applications > gps-tracker-app-test

Overview | Devices | Payload Formats | Integrations | Data | Settings

APPLICATION OVERVIEW [documentation](#)

Application ID gps-tracker-app-test

Description testgps

Created last year

Handler ttn-handler-eu (current handler)

APPLICATION EUIS [manage euis](#)

<> A0 00 00 00 00 00 01 02

DEVICES [register device](#) [manage devices](#)


DEVICE OVERVIEW


Application ID `gps-tracker-app-test`


Device ID `gps-node1`


Description `gps-de-pruebas`


Activation Method `OTAA`


Device EUI `<> ↕ [REDACTED]` 

Application EUI `<> ↕ [REDACTED]` 

App Key `<> ↕ [REDACTED]` 

Device Address `<> ↕ [REDACTED]` 

Network Session Key `<> ↕ [REDACTED]` 

App Session Key `<> ↕ [REDACTED]` 

Status ● 44 minutos ago

Frames up 63 [reset frame counters](#)

Frames down 77

Una vez registrado los dispositivos obtenemos los datos necesarios para hacer la conexión con nuestros dispositivos a la red LoRaWAN.

5.3 Node-red

Node-RED es un motor de flujos con enfoque IoT, que permite definir gráficamente flujos de servicios, a través de protocolos estándares como REST, MQTT, Websocket, AMQP... además de ofrecer integración con "APIs" de terceros, tales como Twitter, Facebook, Yahoo!...

5.3.1 Instalación

Pre-requisitos (Instalar Node.js y npm)

```
sudo apt-get install nodejs
```

```
sudo apt install npm
```

Instalar node-red

```
sudo npm install -g --unsafe-perm node-red node-red-admin
```

Probar la instalación

```
node-red // inicia node-red
```

```
http://localhost:1880
```

Comprobar Versiones instaladas

```
sudo npm -v
```

3.5.2

```
sudo node -v
```

v8.10.0

Iniciar Node-Red como servicio

```
sudo npm install -g pm2
```

 \\ instala pm2 que es un gestor de procesos para Node.js. Permite ejecutar aplicaciones al arrancar

```
pm2 start /usr/bin/node-red -- -v // ejecuta node-red en el segundo plano
```

Iniciar el servicio

```
pm2 startup systemd
```

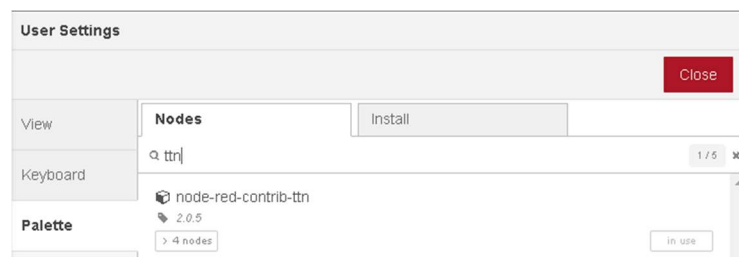
```
sudo env PATH=$PATH:/usr/bin /usr/local/lib/node_modules/pm2/bin/pm2  
startup systemd -u iot --hp /home/iot
```

Reiniciar el sistema y comprobar que todo funciona como se espera.

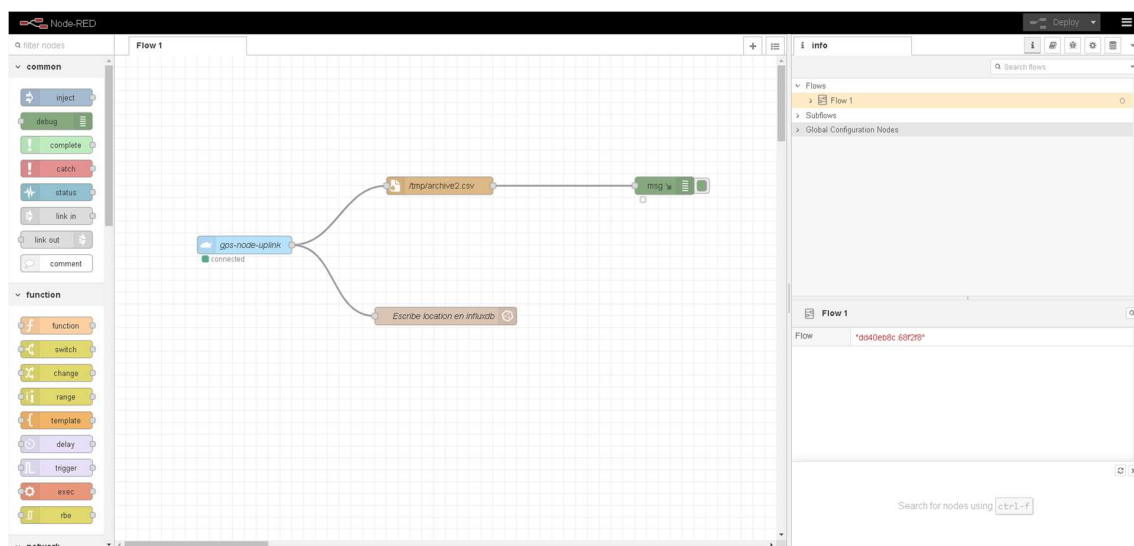
5.3.2 Configuración

Pre-requisito

Instalar paleta **node-red-contrib-ttn** vía icono "hamburguesa", esto nos proporcionará un entorno para conectar los dispositivos a TTN desde node-red.



Relación entre los nodos:



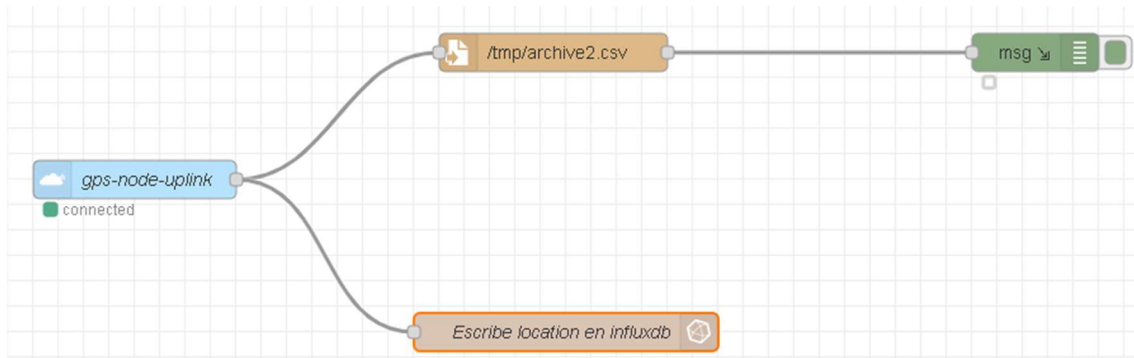
Para configurar node-red, se arrastrarán los nodos necesarios (**ttn-uplink**, **debug** e **InfluxDB out**) desde el panel lateral al panel central de "flujos".

1. Nodo **ttn-uplink**

Se cumplimentarán los campos requeridos. *Device ID* tal y como aparece en TTN = mkrwan1310/ gps-node1 (dispositivo Dragino LGT92)

2- **Nodo de debug** (verde) y conectaremos las cajas por los puntos grises. Editaremos el nodo para que proporcione *"complete msg object"*

3- Si todo es correcto, se pueden desplegar *"Deploy"* los nodos.



5.4 InfluxDB

Almacena datos como puntos en el tiempo y cada punto en el tiempo tiene varios atributos asociados.

5.4.1 Instalación InfluxDB

Añadir repo key

```
sudo curl -sL https://repos.influxdata.com/influxdb.key | sudo apt-key add -
```

Crear repo file

```
sudo echo "deb https://repos.influxdata.com/ubuntu bionic stable" | sudo tee /etc/apt/sources.list.d/influxdb.list
```

```
sudo apt update
```

Instalar InfluxDB

```
sudo apt install influxdb
```

Parar, iniciar y habilitar InfluxDB

```
sudo systemctl stop influxdb
```

```
sudo systemctl start influxdb
```

```
sudo systemctl enable --now influxdb
```

```
sudo systemctl is-enabled influxdb
```

```
sudo systemctl status influxdb
```

5.4.2 Configurar InfluxDB

```
sudo nano /etc/influxdb/influxdb.conf
```

```
[http]

# Determines whether HTTP endpoint is enabled.

enabled = true

# Determines whether the Flux query endpoint is enabled.

# flux-enabled = false
```

Reiniciar InfluxDB

Crear cuenta de administrador (por ejemplo iotdragino)

```
curl -XPOST "http://localhost:8086/query" --data-urlencode
"q=CREATE USER iotdragino WITH PASSWORD 'tu_password' WITH
ALL PRIVILEGES"
```

ó

```
create user iotdragino with password '66666666' with all privileges
```

Si se desea modificar la contraseña: SET PASSWORD FOR iot = 'la_contrasenya_deseada'

Login en influx:

```
influx -username 'iot' -password '66666666'
```

Consultar bases de datos en InfluxDB:

```
curl -G http://localhost:8086/query -u iot:66666666 --data-
urlencode "q=SHOW DATABASES"
```

Crear la base de datos (por ejemplo iot) que albergará los datos serializados que proporcionará TTN.

```
>influx

>create database iot

>show databases

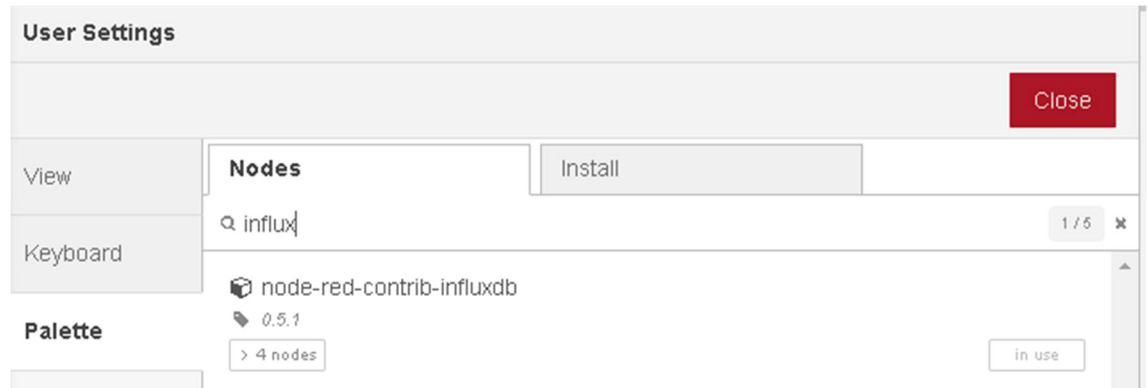
internal

iotdragino
```

5.4.3 Conectar Node-RED a InfluxDB

Pre-requisitos

Añadir paleta **node-red-contrib-influxdb**



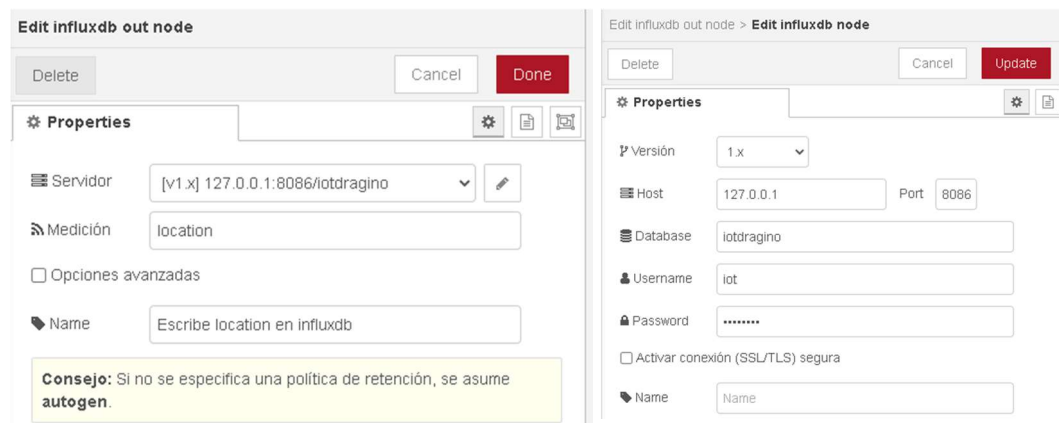
Arrastrar el nodo de salida **influxdb out** -> hacer doble clic en él y hacer clic en el lápiz cumplimentando los campos:

Server: 127.0.0.1:8086

Database: iot con su respectiva contraseña

Measurement: ubicación (o lo que vayamos a medir)

Done (aplica los cambios)



5.5 Graficado

Se representará la información utilizando **Grafana**. Es un software que permite:

- Visualización y el “formateo” de datos métricos.
- Crear cuadros de mandos y gráficas a partir de múltiples fuentes (InfluxDB, Graphite, etc.)

5.5.1 Instalar y configurar Grafana

Se descargará el e instalará el paquete grafana_5.X.X_amd64.deb

```
wget https://s3-us-west-2.amazonaws.com/grafana-releases/release/grafana\_5.2.2\_amd64.deb
```

```
dpkg -i grafana_5.2.2_amd64.deb
```

Se iniciará el servicio Grafana:

```
sudo systemctl daemon-reload
```

```
sudo systemctl enable grafana-server
```

```
sudo systemctl start grafana-server
```

```
sudo systemctl status grafana-server
```

Acceso a Grafana:

```
http://Direccion_IP:3000 admin/admin
```

5.5.2 Conectar Grafana a InfluxDB

The screenshot shows the Grafana Settings page for a new data source named 'Iotdragino'. The 'Query Language' is set to 'InfluxQL'. Under the 'HTTP' section, the 'URL' is 'http://localhost:8086', 'Access' is 'Server (default)', and 'Whitelisted Cookies' is empty. The 'Auth' section shows 'Basic auth' is enabled with 'With Credentials' checked. 'TLS Client Auth' is disabled, 'Skip TLS Verify' is disabled, and 'Forward OAuth Identity' is disabled. Under 'Custom HTTP Headers', there is an 'Add header' button. The 'InfluxDB Details' section shows 'Database' as 'iotdragino', 'User' as 'iot', 'Password' as 'configured', and 'HTTP Method' as 'GET'. A 'Reset' button is next to the password field.

Añadir *data source*:

Name: Iotdragino

Type: InfluxDB

HTTP URL: http://localhost:8086

Auth: With Credentials

Access: Server

InfluxDB Details: Database: iot User:iot
Password:66666666

5.5.3 Instalar el plugin Trackmap de Grafana

```
sudo grafana-cli plugins install pr0ps-trackmap-panel
```

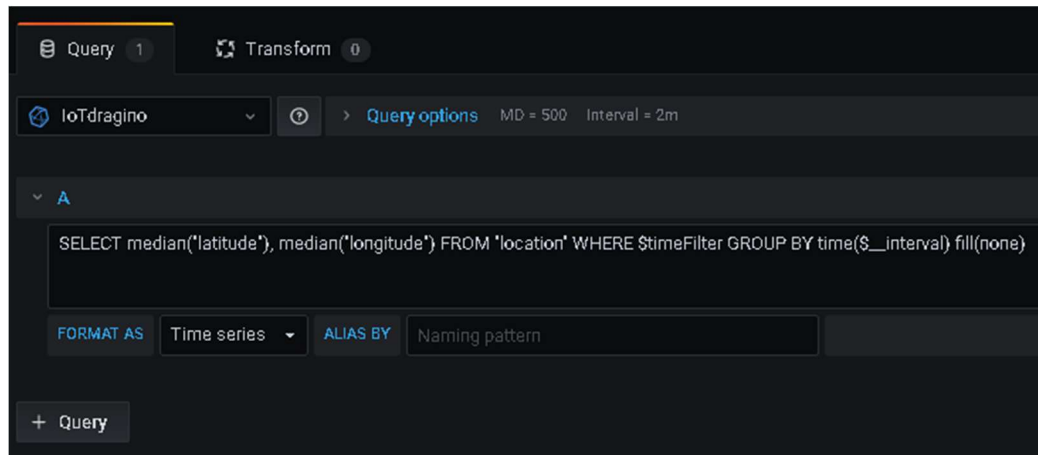
```
sudo service grafana-server restart
```

5.5.4 Añadir Dashboard

Elegir panel "TrackMap", añadir y editar Dashboard y elegir como Data source: Iotdragino

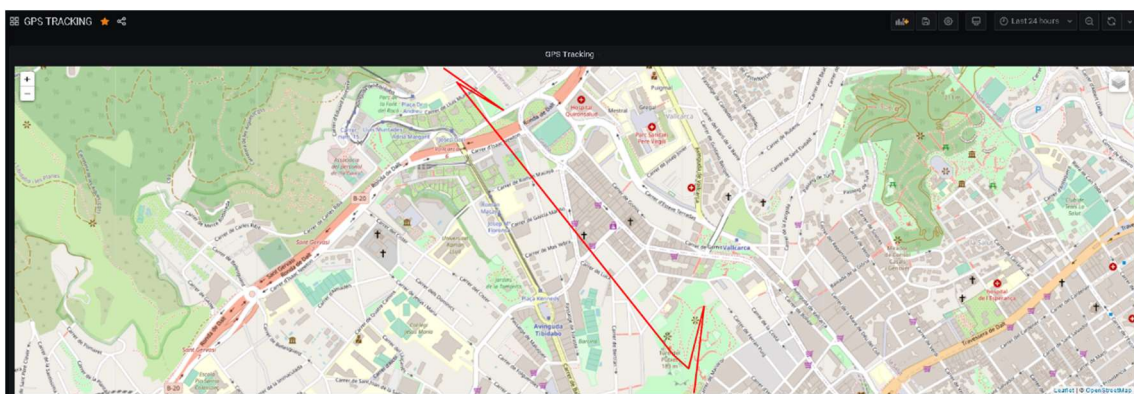
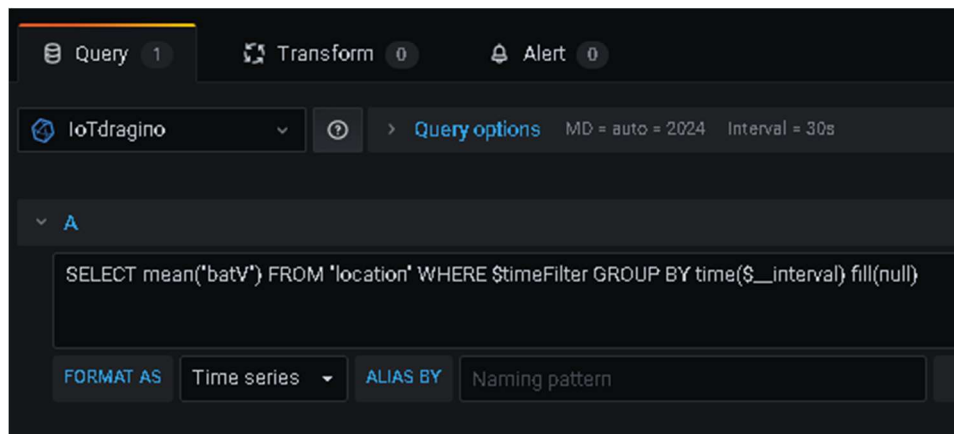
Añadir consultas:**Grafica de Tracking**

```
SELECT median("latitude"), median("longitude") FROM "location" WHERE $timeFilter GROUP BY time($__interval) fill(none)
```



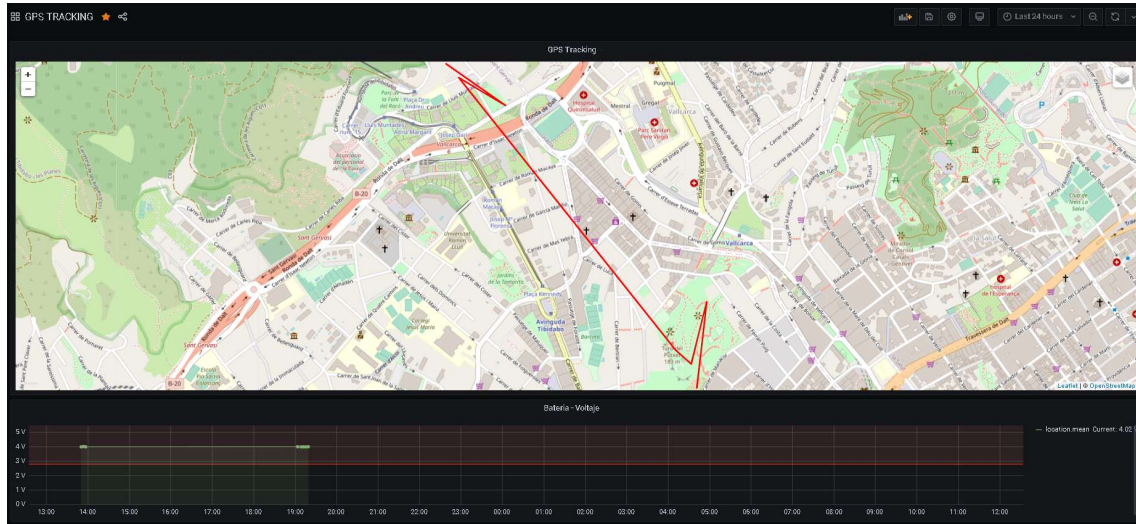
Grafica Consumo batería

```
SELECT mean("value") FROM "measurement" WHERE $timeFilter GROUP BY time($__interval) fill(null)
```



6 Resultados

Tenemos como resultado un montaje que nos permite rastrear la ubicación (y el consumo de batería) de cualquier dispositivo LoRaWAN sobre el mapa en tiempo "real".



7 Anexos

7.1 Sketch cobertura LoraWAN

```
#include
<MKRWAN.h>

LoRaModem modem;

#include "arduino_secrets.h"
// Please enter your sensitive data in the Secret tab or
arduino_secrets.h
String appEui = SECRET_APP_EUI;
String appKey = SECRET_APP_KEY;

#define LBLUE 5
#define LGREEN 4
#define LRED 3
#define BOARDLED 6

bool connected;
```

```
int err_count;

void setColor(int r,int g,int b) {
    digitalWrite(LBLUE,b);
    digitalWrite(LGREEN,g);
    digitalWrite(LRED,r);
}

void setup() {
    //Serial.begin(115200);
    modem.begin(EU868);
    delay(1000);      // apparently the murata dislike if this tempo is
removed...
    connected=false;
    err_count=0;
    pinMode(LBLUE,OUTPUT);
    pinMode(LGREEN,OUTPUT);
    pinMode(LRED,OUTPUT);
    pinMode(BOARDLED,OUTPUT);
    setColor(HIGH,HIGH,LOW);
}

void loop() {
    char msg[12] = {0,1,2,3,4,5,6,7,8,9,10,11};
    if ( !connected ) {
        int ret=modem.joinOTAA(appEui, appKey);
        if ( ret ) {
            connected=true;
            modem.minPollInterval(60);
            modem.dataRate(5);
            delay(100);
            err_count=0;
        } else {
            setColor(LOW,HIGH,HIGH);
        }
    }
    if ( connected ) {
        int err=0;
        modem.beginPacket();
        modem.write(msg,12);
        err = modem.endPacket(true);
        if ( err <= 0 ) {
            setColor(LOW,HIGH,HIGH);
            err_count++;
            if ( err_count > 50 ) {
```

```
        connected = false;
        setColor(HIGH,HIGH,LOW);
    }
    for ( int i = 0 ; i < 1200 ; i++ ) {
        setColor(HIGH,HIGH,HIGH);
        delay(50);
        setColor(LOW,HIGH,HIGH);
        delay(50);
    }
} else {
    setColor(HIGH,LOW,HIGH);
    err_count = 0;
    delay(10000);
}
}
```

7.2 Payload Decoder TTN

```
function Decoder(bytes, port) {
    // Decode an uplink message from a buffer
    // (array) of bytes to an object of fields.

    var alarm=(bytes[6] & 0x40)?true:false;//Alarm status
    value=((bytes[6] & 0x3f) <<8) | bytes[7];
    var batV=value/1000;//Battery,units:Volts
    value=bytes[8]<<8 | bytes[9];
    if(bytes[8] & 0x80)
    {
        value |=0xFFFF0000;
    }
    var roll=value/100;//roll,units: °
    value=bytes[10]<<8 | bytes[11];
    if(bytes[10] & 0x80)
    {
        value |=0xFFFF0000;
    }
}
```

```
var pitch=value/100; //pitch,units: °

var json={

roll:roll,

pitch:pitch,

batV:batV,

alarm:alarm

};

var value=bytes[0]<<16 | bytes[1]<<8 | bytes[2];

if(bytes[0] & 0x80)

{

value |=0xFFFFF000000;

}

var value2=bytes[3]<<16 | bytes[4]<<8 | bytes[5];

if(bytes[3] & 0x80)

{

value2 |=0xFFFFF000000;

}

if (value == 0x0FFFFF && value2 == 0x0FFFFF)

{

//gps disabled (low battery)

} else if (value === 0 && value2 === 0) {

//gps no position yet

} else {

json.latitude=value/10000;//gps latitude,units: °

json.longitude=value2/10000;//gps longitude,units: °

}

return json;

}
```

8 Bibliografía

[MKRWAN1300](#)

[Arduino MKR WAN 1300 LoRA: First Steps](#)

[Getting started with Arduino MKRWAN1300 & LoRaWan TTN](#)

[Plugin for automatically integrating TTN devices in Thinger.io platform](#)

[Arquitectura Lora](#)

[LoRa + TTN: comunicando cosas con Internet](#)

[Node-Red guía rápida](#)

[Visualizing Lora Node data with Node-red and Grafana](#)

[Instalar y configurar Grafana](#)

[TrackMap Panel](#)

***** FIN DEL DOCUMENTO *****