CONSTRUYENDO UNA RED GLOBAL DE INTERNET DE LAS COSAS ENTRE TOD@S

Facendo IoT con LoRaWAN. Introdución

juanfelixmateos@gmail.com

THE THINGS NETWORK

UNA RED IOT ENTRE Y PARA TOD@S

THE THINGS NETWORK

The Things Network: Misión

Crear una red IoT descentralizada y tecnológicamente independiente, en la que los usuarios son a la vez los propietarios y los operadores del sistema

THE THINGS NETWORK

The Things Network: Principios

Tus datos son tus datos

Neutralidad de la red

Código abierto

THE THINGS NETWORK

Cómo empezó todo

Julio de 2015, Amsterdam

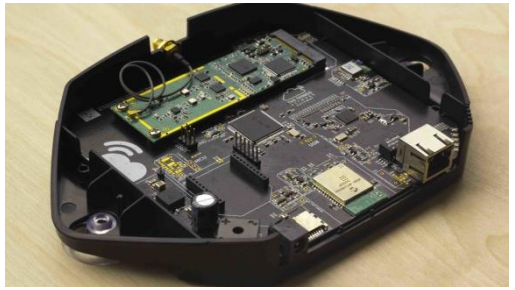
THE THINGS NETWORK

Juan Félix Mateos

Long Range Wide Area Network

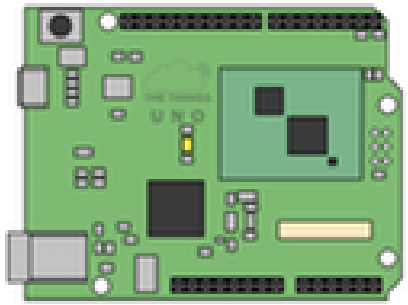
THE THINGS NETWORK

KICKSTARTER



TheThings
GATEWAY

TheThings UNO



Año 2015
KICK STARTER

295.331 €
de la meta de 150.000 €

934
patrocinadores

TheThings NODE

THE THINGS NETWORK

AELORA

www.aelora.nl

THE THINGS NETWORK

BORBORA – CHILDREN AIR QUALITY

www.waag.org

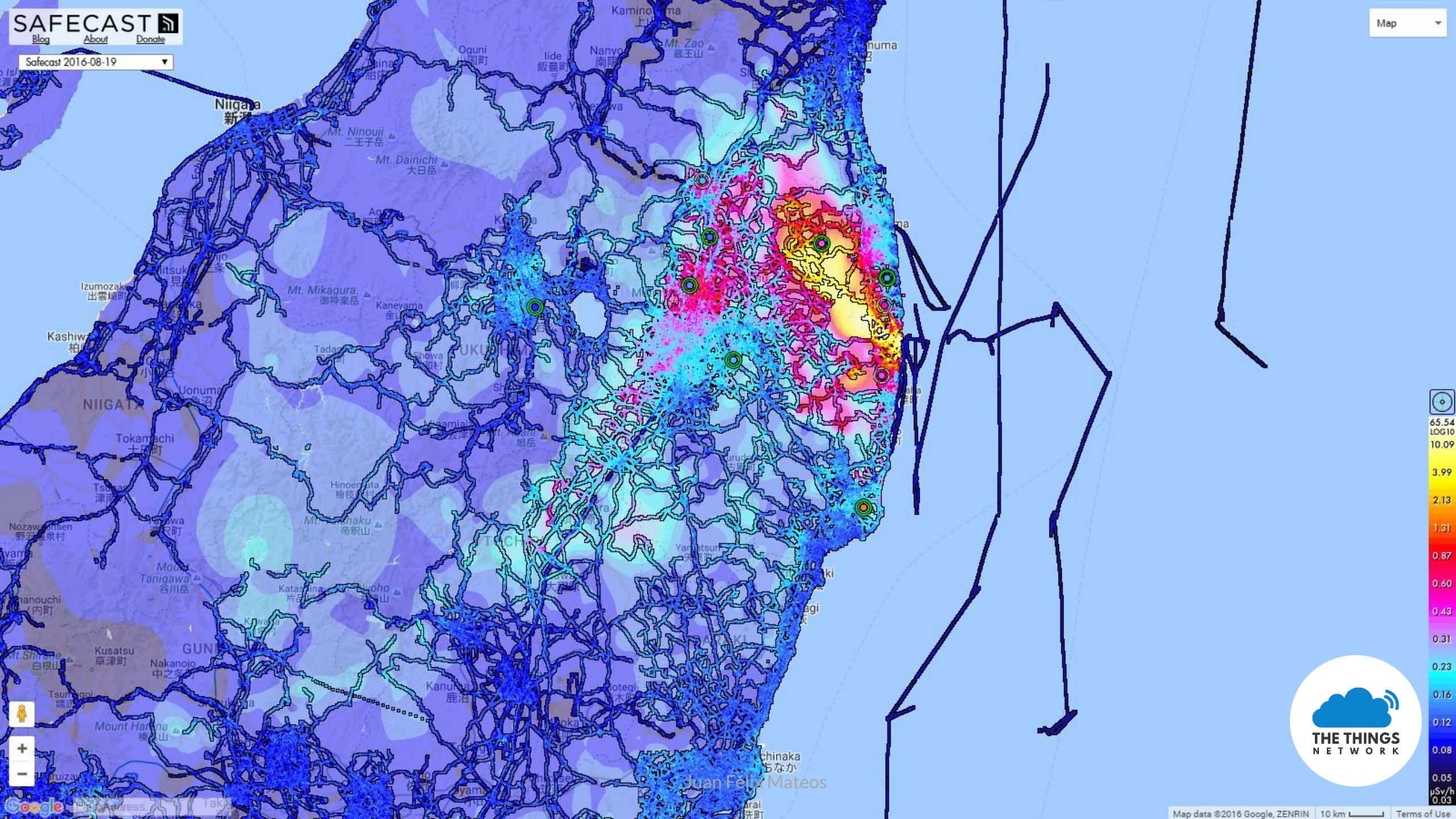
Juan Félix Mateos

SAFECAST

www.blog.safecast.org/

SAFECAST

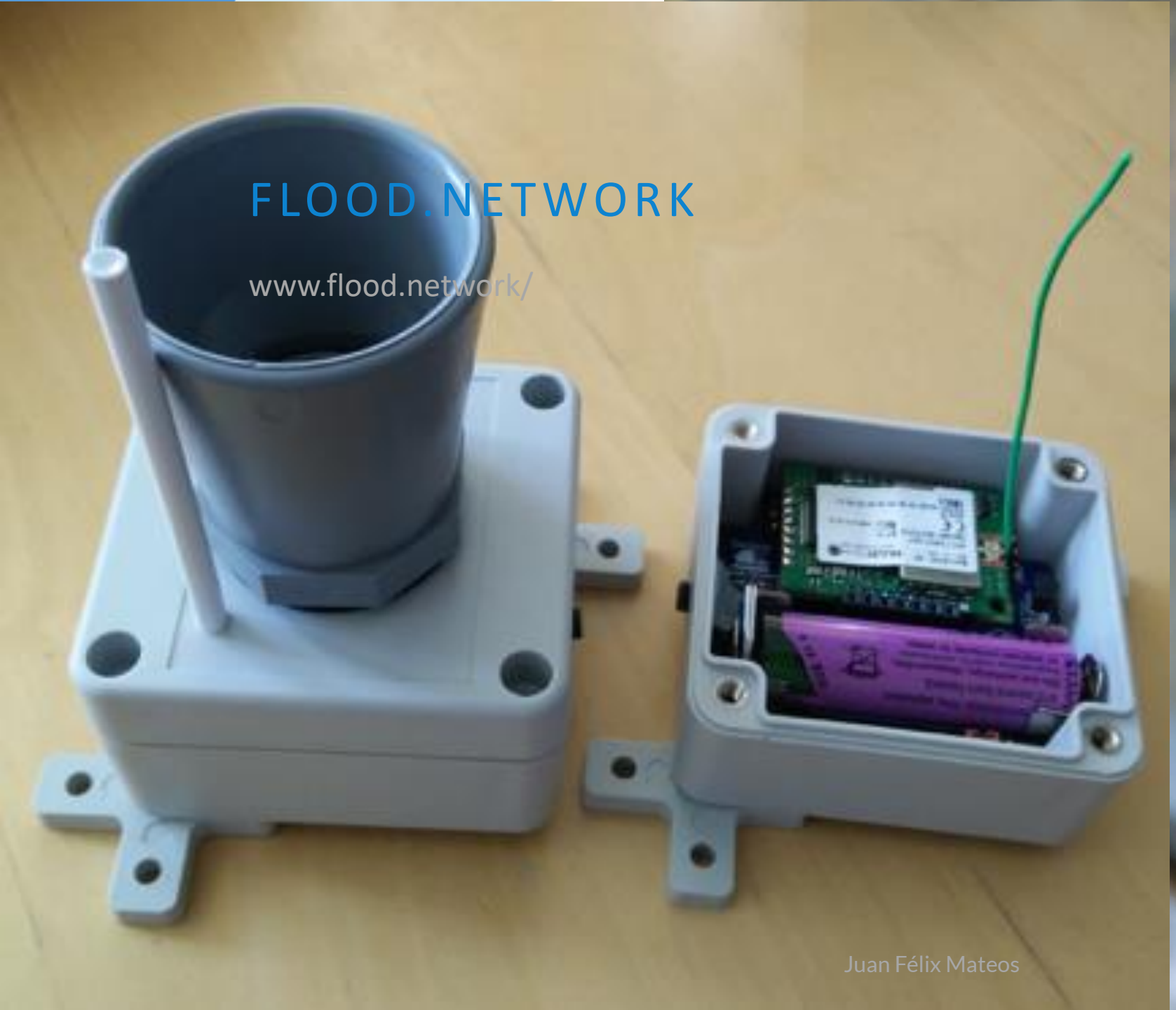
THE THINGS NETWORK


SAFECAST
Blog
About
Donate
Safecast 2016-08-19
Map
65.54
LOG10
10.09
3.99
2.13
1.31
0.87
0.60
0.43
0.31
0.23
0.16
0.12
0.08
0.05
μSv/h
0.03
THE THINGS
NETWORK
Juan Felix Mateos
Map data ©2016 Google, ZENRIN
10 km
Terms of Use


HOOSJE BOOTJE
Juan Félix Mateos
THE THINGS
NETWORK

FLOOD.NETWORK

www.flood.network/



Juan Félix Mateos

FLOOD.NETWORK

Juan Félix Mateos

SMART MOUSE TRAP

www.xignal.com

THE THINGS NETWORK

THE THINGS NETWORK

CONCEPTOS TÉCNICOS

THE THINGS NETWORK

LoRa™

Conexiones punto a punto

Largo alcance

Bajo consumo de energía

Bajo ancho de banda

THE THINGS NETWORK

LoRaWAN™

Nodo ↔ **Mis** gateways ↔ Internet

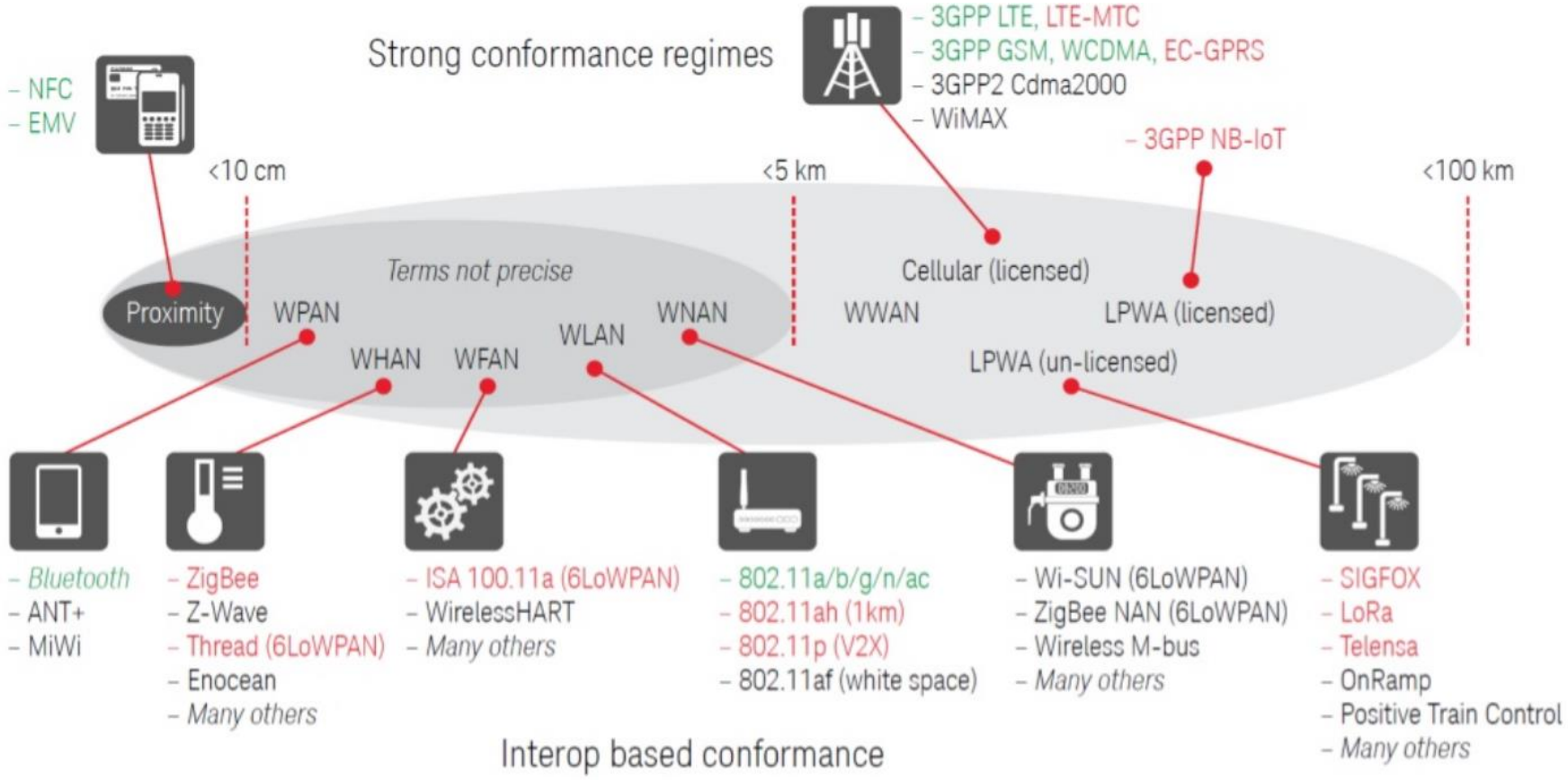
Solución local

THE THINGS NETWORK

Nodo ↔ **Cualquier** Gateway TTN ↔ The Things Network

Solución global

TECNOLOGÍAS DE CONECTIVIDAD INALÁMBRICA



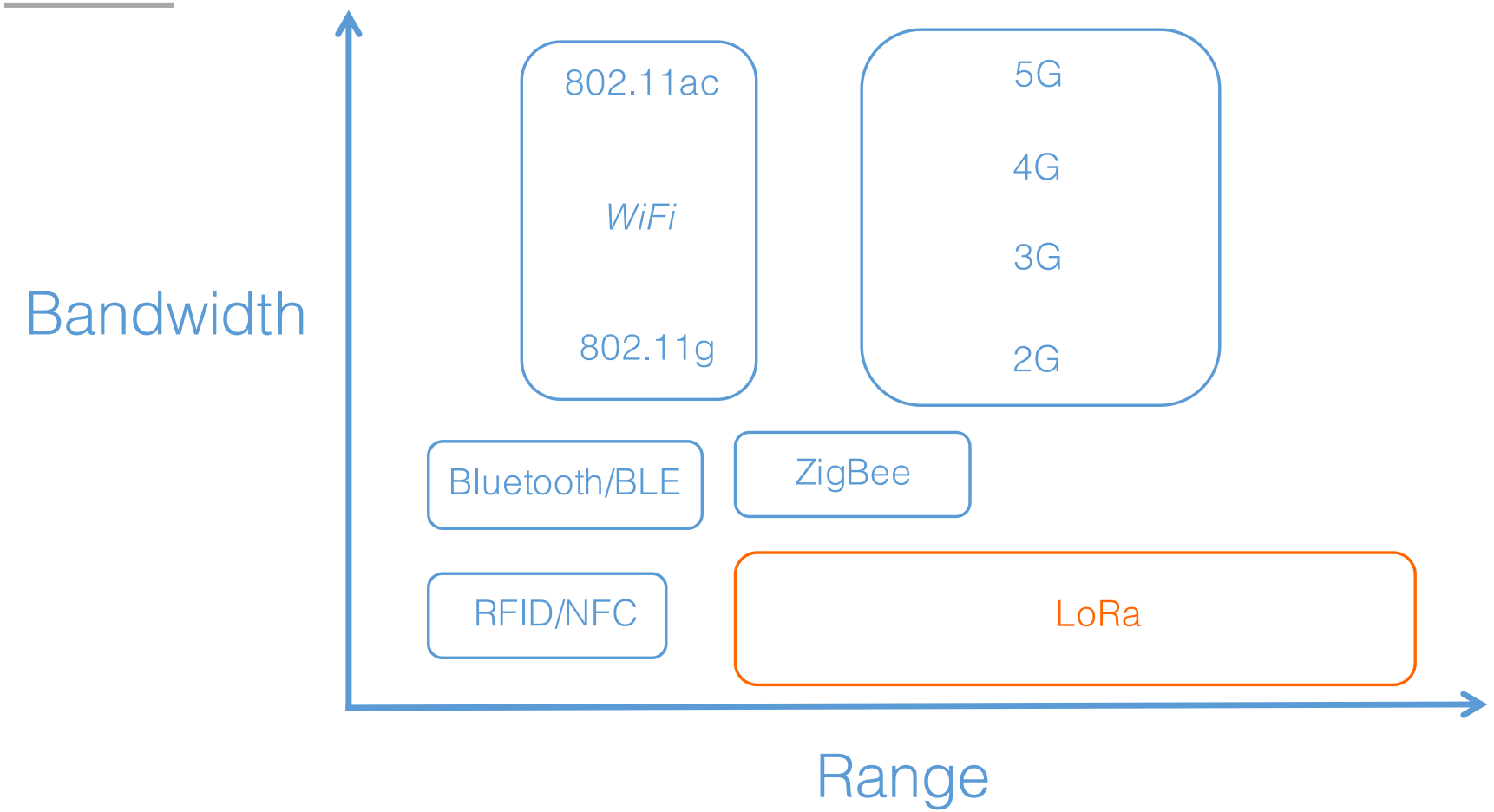
■ : > Billion units/year now
■ : Emerging

WPAN: Wireless Personal Area Network
WHAN: Wireless Home Area Network
WFAN: Wireless Field (or Factory) Area Network
WLAN: Wireless Local Area Network

WNAN: Wireless Neighborhood Area Network
WWAN: Wireless Wide Area Network
LPWA: Low Power Wide Area

THE THINGS NETWORK

LORA



THE THINGS NETWORK

LoRa

COMPARISON – main LPWAN technologies

Vertical M2M

Feature	LORAWAN	SIGFOX	LTE Cat 1	LTE M	NB - LTE
Modulation	SS chip	UNB / GFSK / BPSK	OFDMA	OFDMA	OFDMA
Rx Bandwith	500 – 125 KHz	100 Hz	20 MHz	20 – 1.4 MHz	200 KHz
Data Rate	290bps – 50Kbps	100 bit / sec 12 / 8 bytes Max	10 Mbit /sec	200 kbps – 1 Mbps	Average 20K bit / sec
Max. # Msgs/day	Unlimited	UL: 140 msgs / day	Unlimited	Unlimited	Unlimited
Max Outpu Power	20 dBm	20 dBm	23 – 46 dBm	23/30 dBm	20 dBm
Link Budget	154 dB	151 dB	130 dB+	146 dB	150 dB
Battery lifetime – 2000 mAh	105 months	90 months		18 months	
Power Efficiency	Very High	Very High	Low	Medium	Med high
Interference immunity	Very High	Low	Medium	Medium	Low
Coexistence	Yes	No	Yes	Yes	No
Security	Yes	No	Yes Oui	Yes	Yes
Mobility / localization	Yes	Limited mobility, No localization	Mobility	Mobility	Limited mobility, No localization

Source: LoRAWAN Alliance, 2015

www.vertical-m2m.com

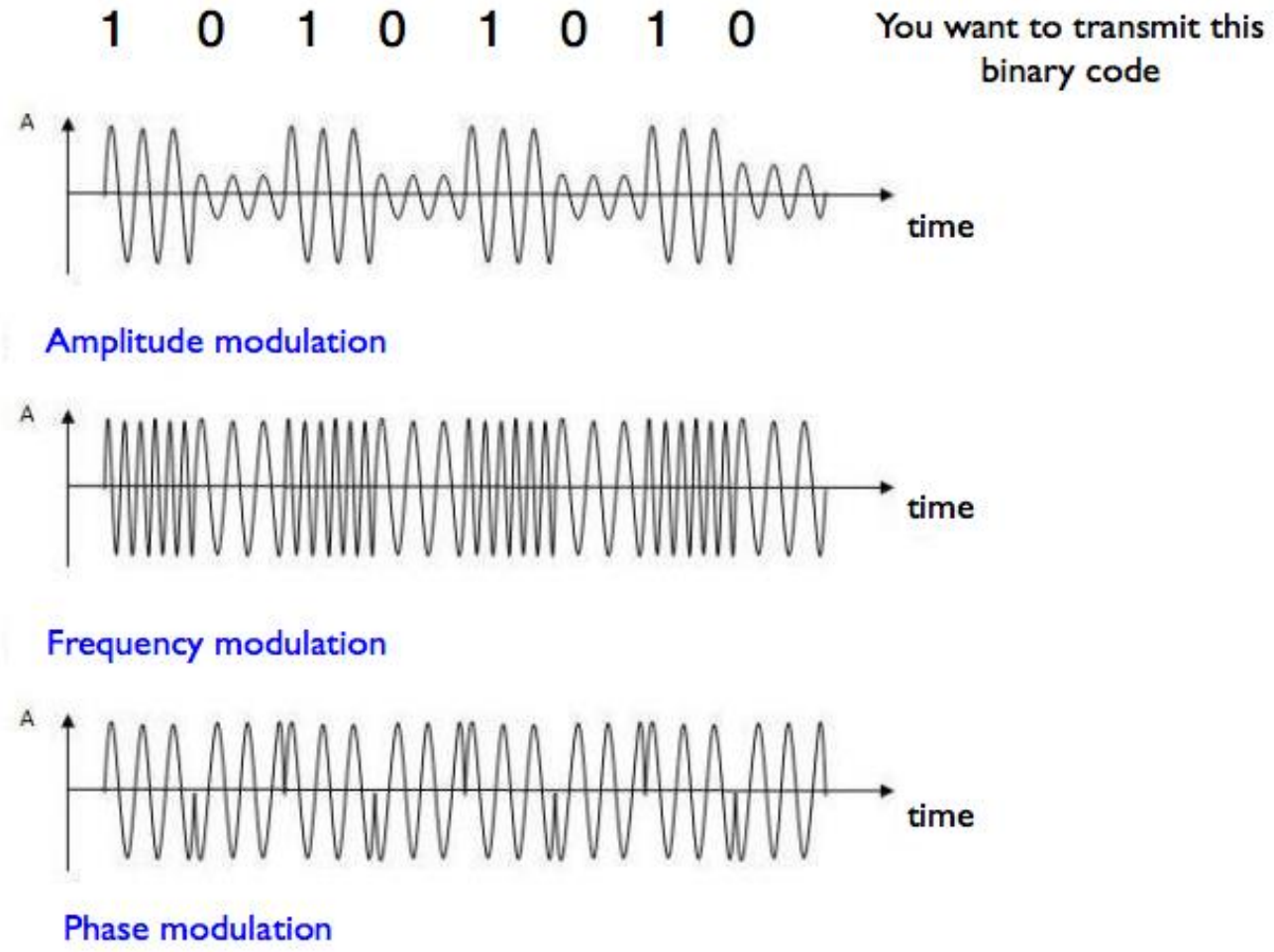
THE THINGS NETWORK

LORA

ALCANCE	Yes / No	~1Km (urbano) / ~10Km (rural)
BAJO CONSUMO	Yes / No	La baterías duran años
MILES DE DISPOSITIVOS	Yes / No	Conectados a un mismo gateway
BANDA SIN LICENCIAS	Yes / No	868MHz (EU) 915MHz (US)
ANCHO DE BANDA	Yes / No	Max 50kb/s

THE THINGS NETWORK

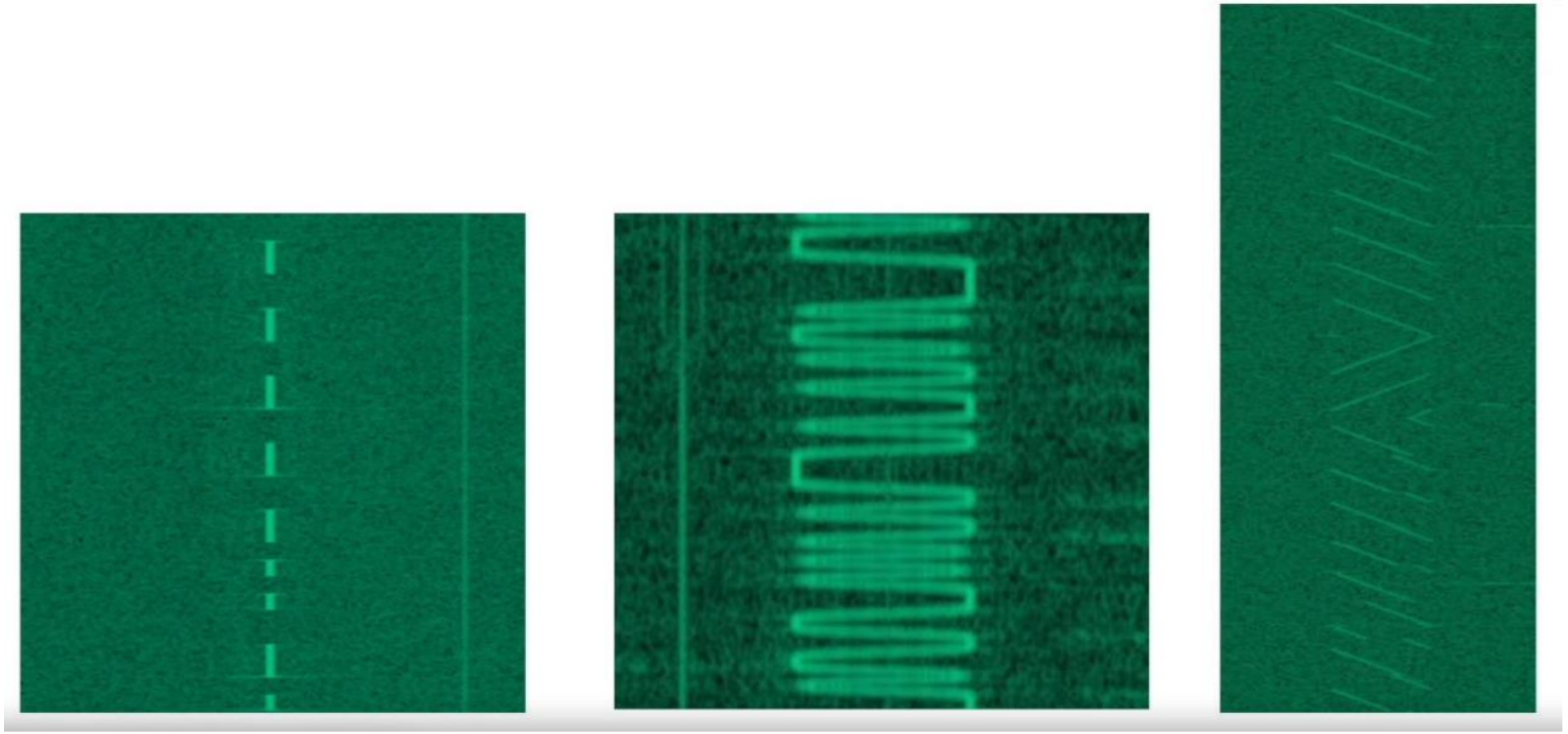
Modulación de radio



https://spotlight.unavco.org/how-gps-works/gps-basics/gps-and-geosciences.html

Modulación:

On-Off keying VS Frecuency-shift keying VS LoRa



http://www.sghoslya.com/p/lora-is-chirp-spread-spectrum.html

Modulación LoRa

Chirp Spread Spectrum

Tecnología de los 40

Utilizada en ámbitos militares → Rádar

- Largo alcance y penetración
- Difícil de detectar
- Alta resistencia a interferencias

LoRa: Bit, Symbol, Chip, Chirp

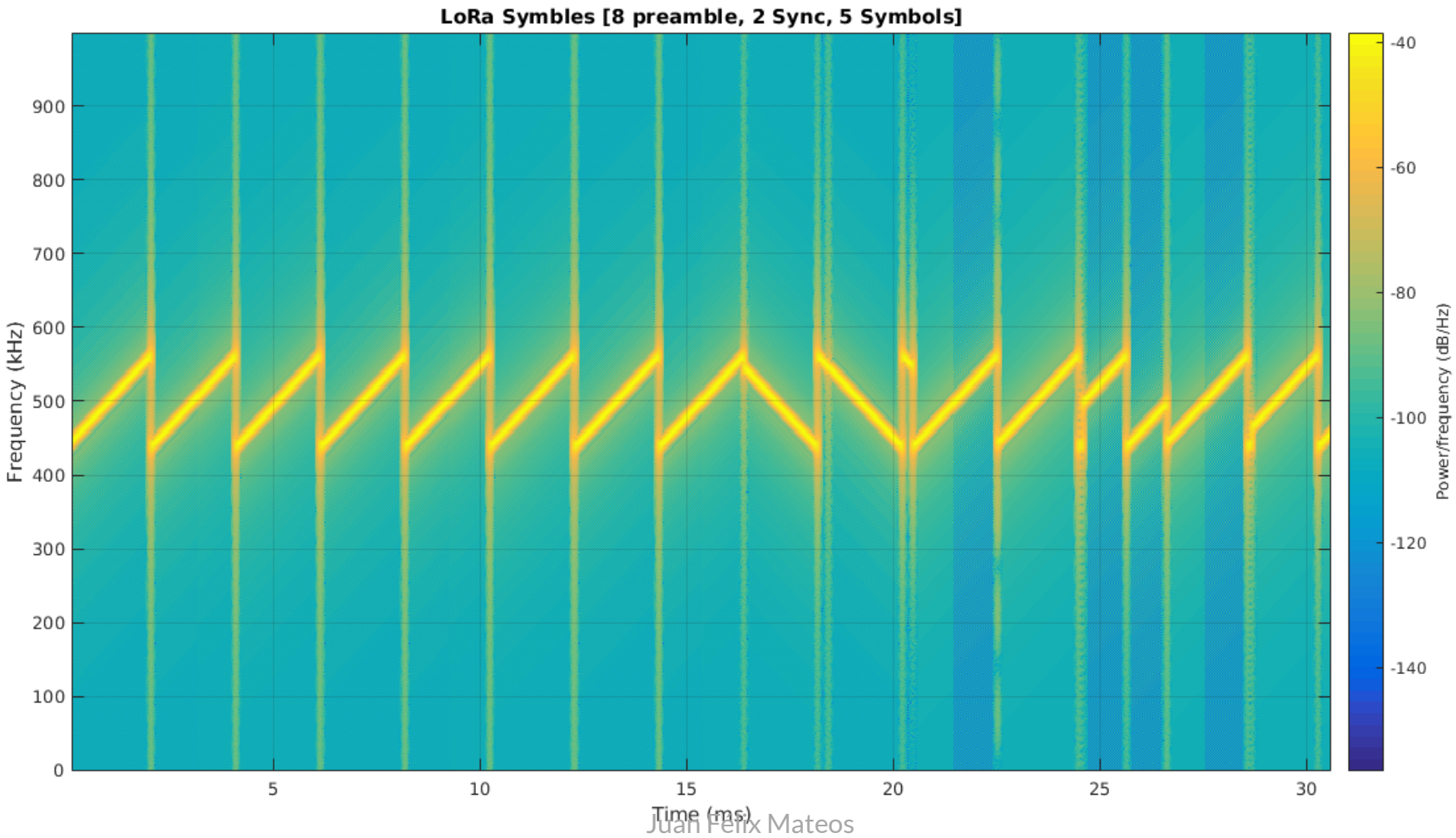
Bit: Unidad mínima de información a transmitir

Symbol: Codificación de la información para transmitirla a través de un medio determinado. Un symbol puede codificar un bit o más, y añadir información adicional como corrección de errores.

Chip: Unidad mínima de codificación en Spread Spectrum: Se necesitan varios chips para codificar un symbol.

Chirp: Es un Symbol en la terminología de Spread Spectrum.

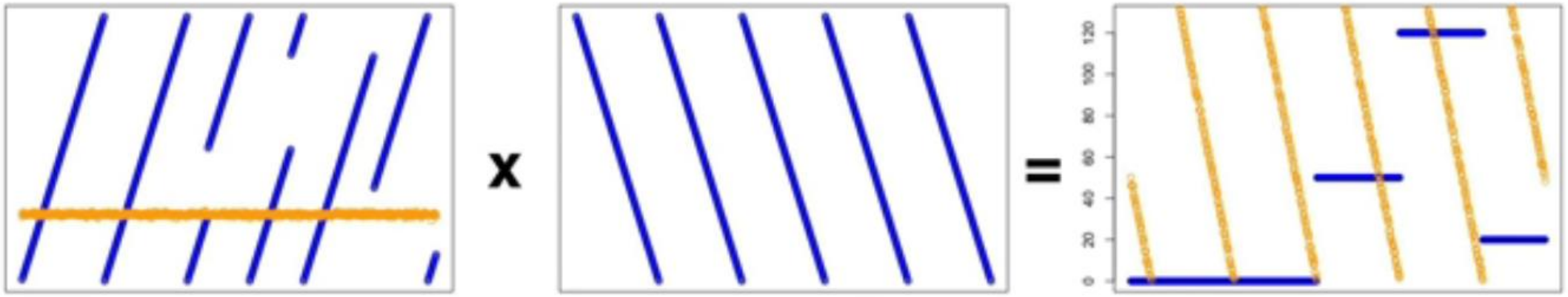
LoRa Frame



http://www.sghoslya.com/p/lora-is-chirp-spread-spectrum.html

THE THINGS NETWORK

LoRa Demodulación



http://www.sghoslya.com/p/lora-is-chirp-spread-spectrum.html

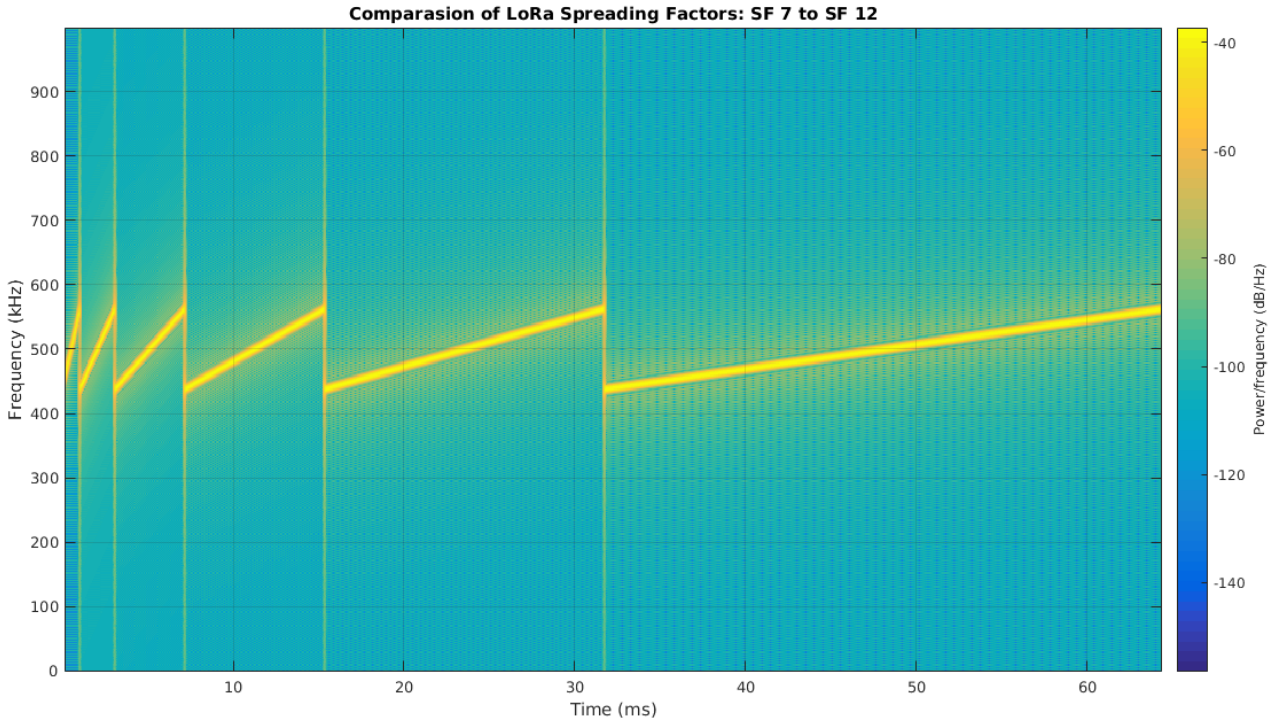
THE THINGS NETWORK

LoRa: Spread Factor

Velocidad vs Alcance vs Consumo

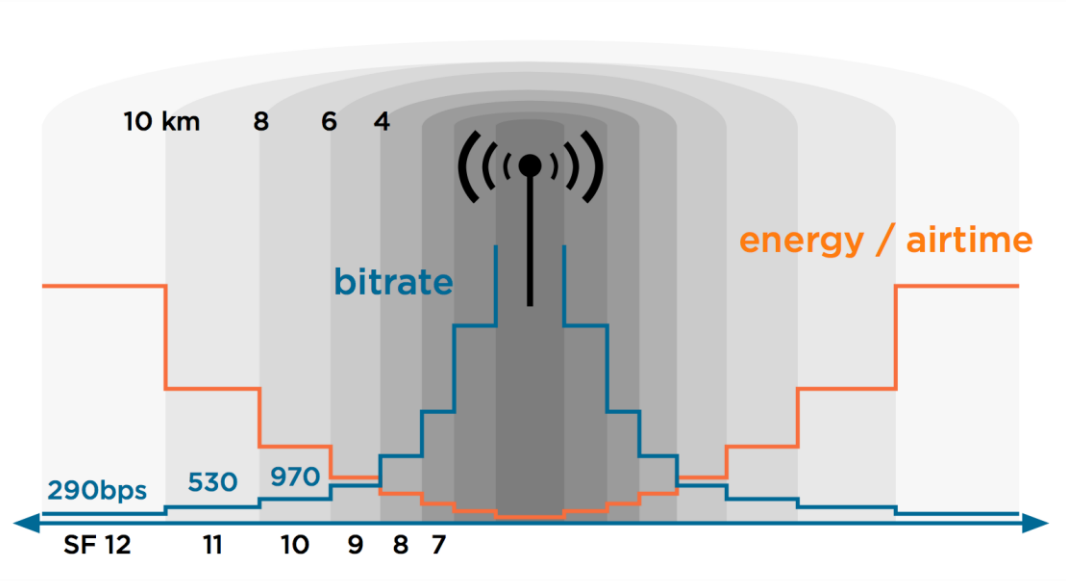
A mayor velocidad, menor alcance

A mayor velocidad, menor consumo



THE THINGS NETWORK

LoRa: Spread Factor



https://blog.surf.nl/en/lora-the-internet-of-things/

LoRa Spreading Factors (125kHz bw)

Spreading Factor	Chips/symbol	SNR limit	Time-on-air (10 byte packet)	Bitrate
7	128	-7.5	56 ms	5469 bps
8	256	-10	103 ms	3125 bps
9	512	-12.5	205 ms	1758 bps
10	1024	-15	371 ms	977 bps
11	2048	-17.5	741 ms	537 bps
12	4096	-20	1483 ms	293 bps

http://www.techplayon.com/lora-link-budget-sensitivity-calculations-example-explained/

THE THINGS NETWORK

LoRa: Duty Cycle – 36 segundos por hora

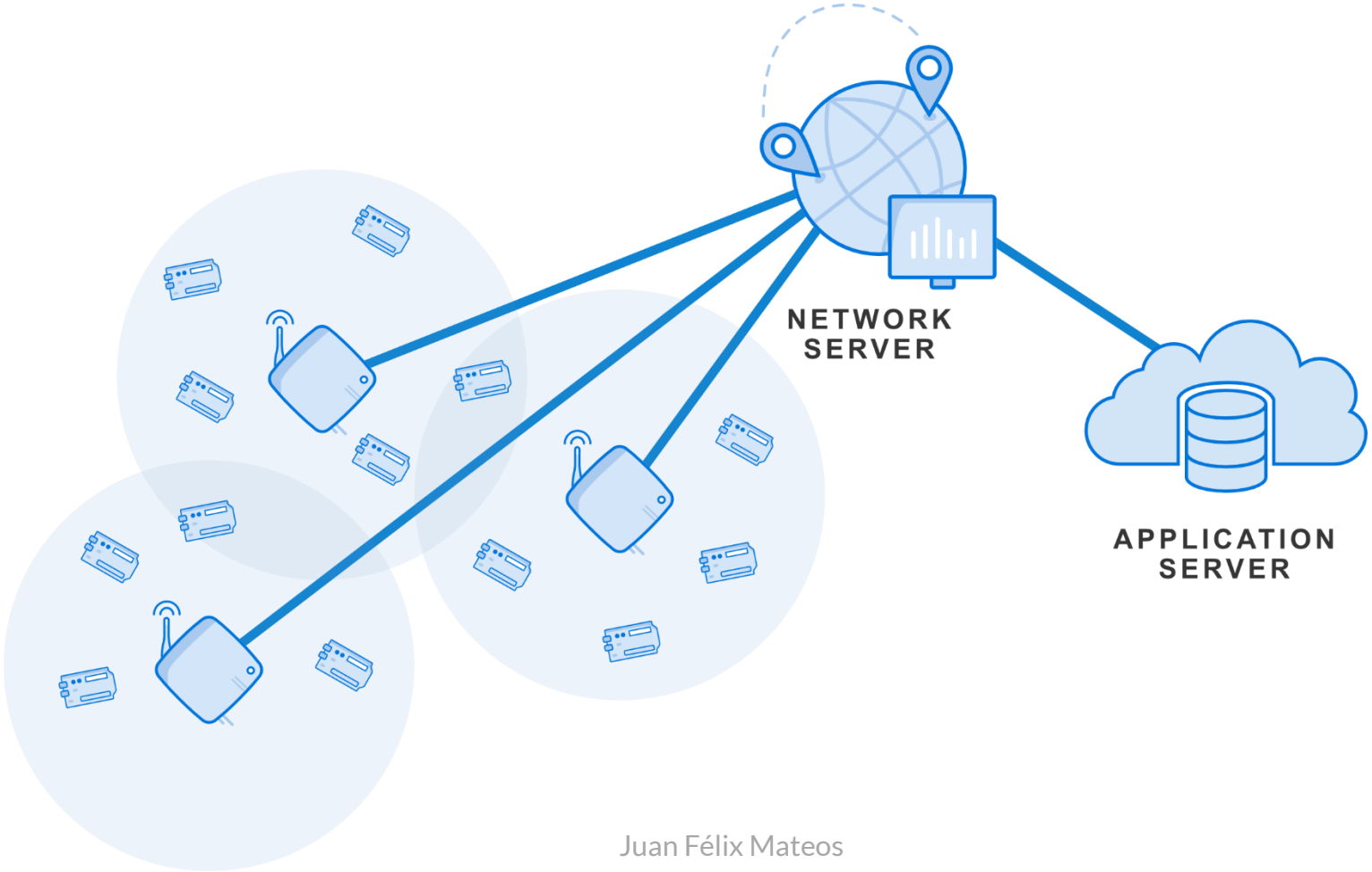
Sección 7.2.3 de la norma ETSI EN300.220

- g (863.0 – 868.0 MHz): 1%
- g1 (868.0 – 868.6 MHz): 1%
- g2 (868.7 – 869.2 MHz): 0.1%
- g3 (869.4 – 869.65 MHz): 10%
- g4 (869.7 – 870.0 MHz): 1%

The Things Network: política de acceso justo

- La red es un procomún
    - Todo@s nos beneficiamos de un uso razonable
- **Uplink**: 30 segundos por día y nodo (aprox. 550 mensajes en SF7)
- **Downlink**: 10 mensajes por día y nodo

LoRaWAN



THE THINGS NETWORK

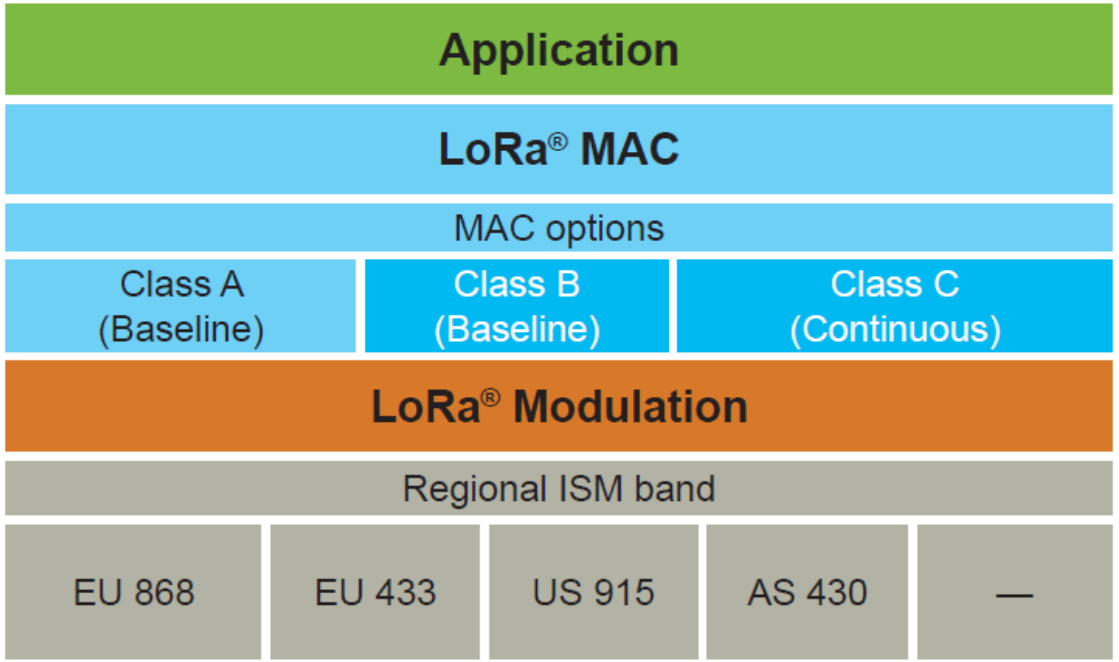
¿Cómo funiona?

DEVICES **GATEWAYS** **NETWORK SERVER** **APPLICATION SERVER**

THE THINGS NETWORK

LoRaWAN

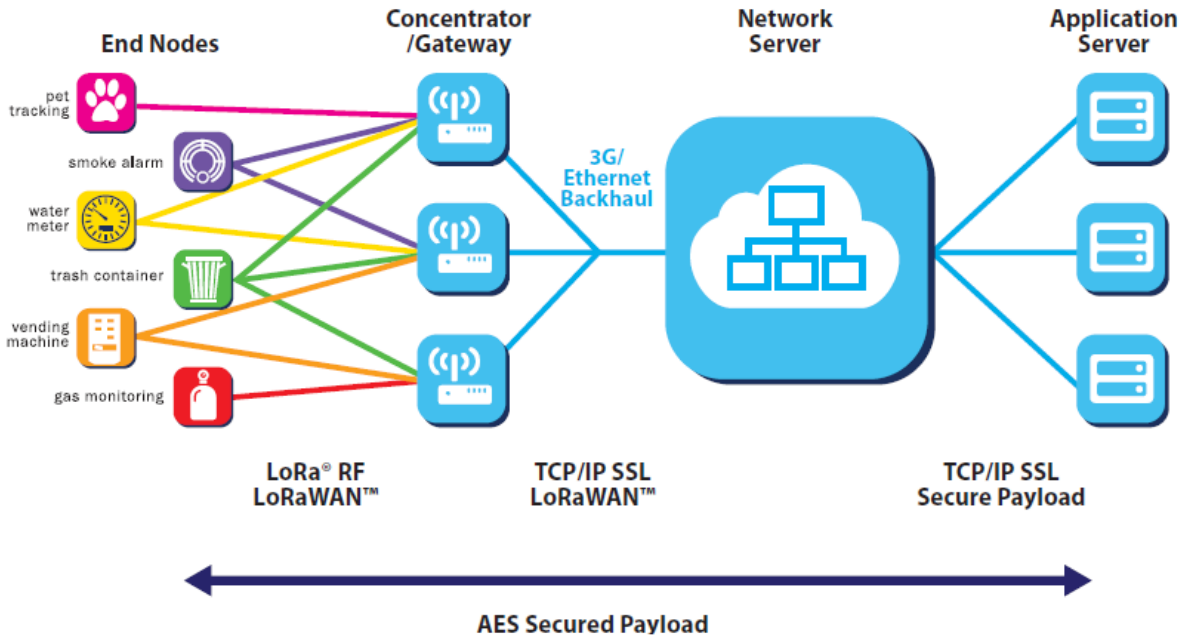
- LoRa → Capa física de largo alcance
- LoRaWAN → Arquitectura de protocolo y capa de red
- El protocolo y la arquitectura tienen una gran influencia sobre
  - Consumo de batería
  - Capacidad de la red
  - QoS
  - Aplicaciones finales sobre la estructura



THE THINGS NETWORK

LoRaWAN

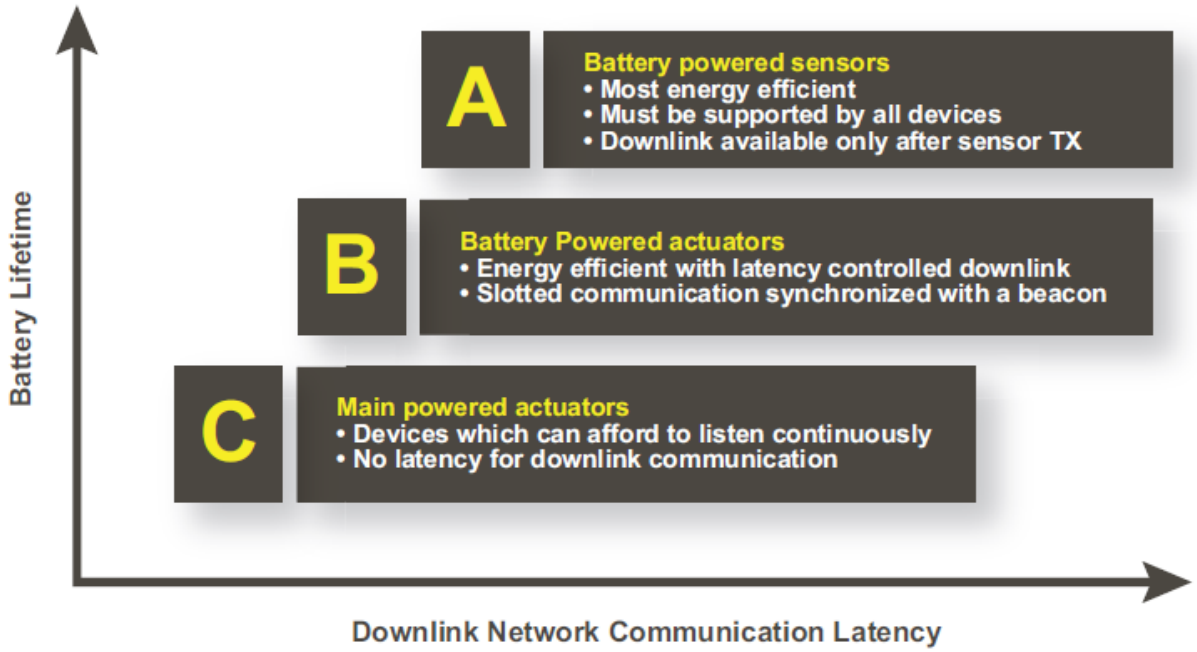
- En LoRaWAN no existe asociación entre nodo y gateway
- La inteligencia y la complejidad se desplazan al plano del servidor, que es el que lida con duplicidades, seguridad, planificación de notificaciones,
- No existe el concepto de handover



THE THINGS NETWORK

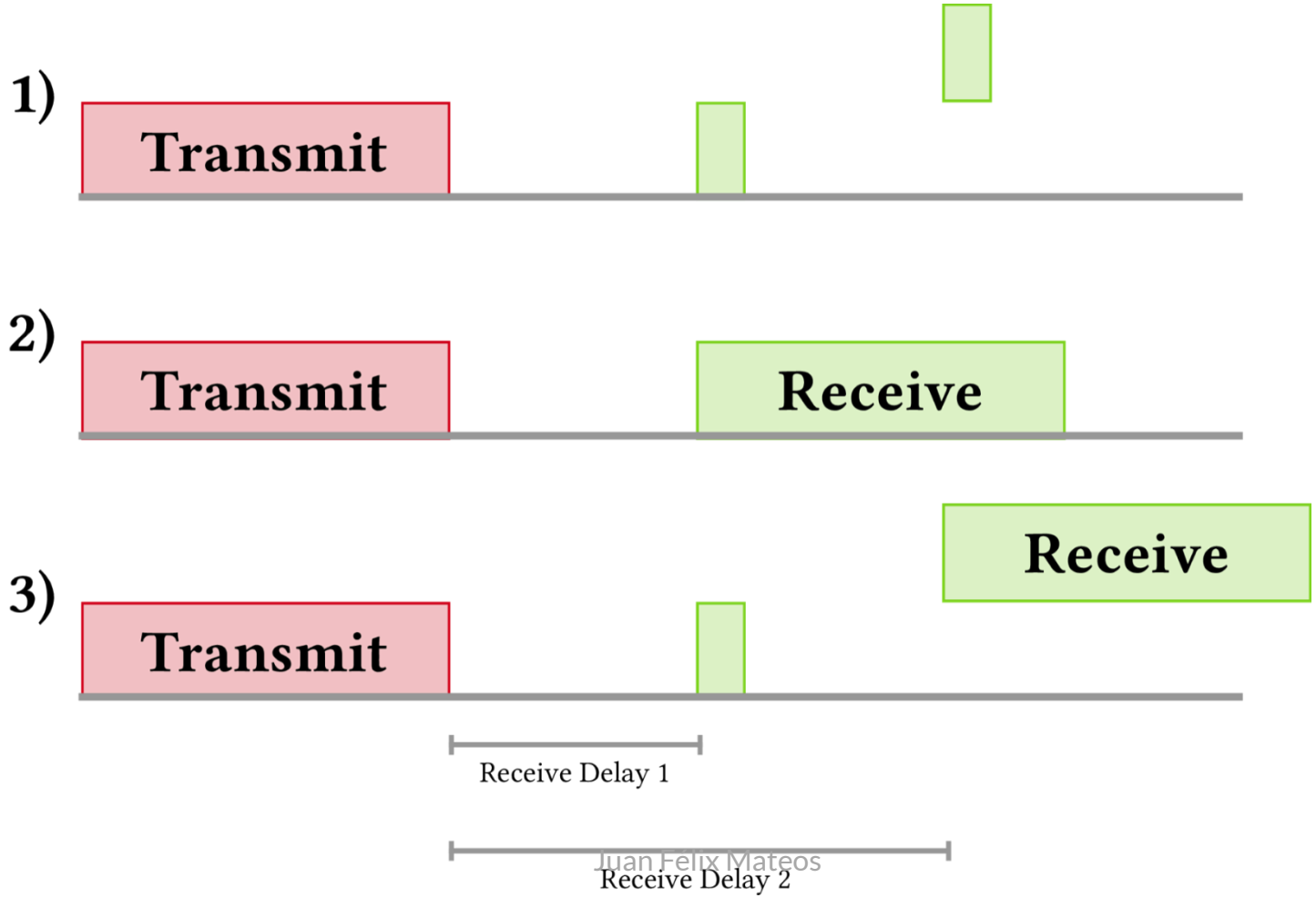
LoRaWAN

- Tres clases de dispositivos en función del uso del espectro:
  - Clase A: Sensores
  - Clase B: Actuadores
  - Clase C: Actuadores críticos



THE THINGS NETWORK

LoRaWAN: Uplink y Downlink en clase A



THE THINGS NETWORK

LoRaWAN: Identificación única de los componentes

GatewayEUI (64 bits, único).

DevEUI (64 bits, único)

AppEUI (64 bits, único)

LoRaWAN: Información requerida por el nodo

- **Device Address** (32 bits, puede no ser único)
  - 7 MSB: Network Identifier (0x26, 0x27 = The Things Network)
  - 25 LSB: Network Address
- **Network Session Key**
- **Application Session Key (ABP)**
- **Frame counter (uplink** y **downlink)**

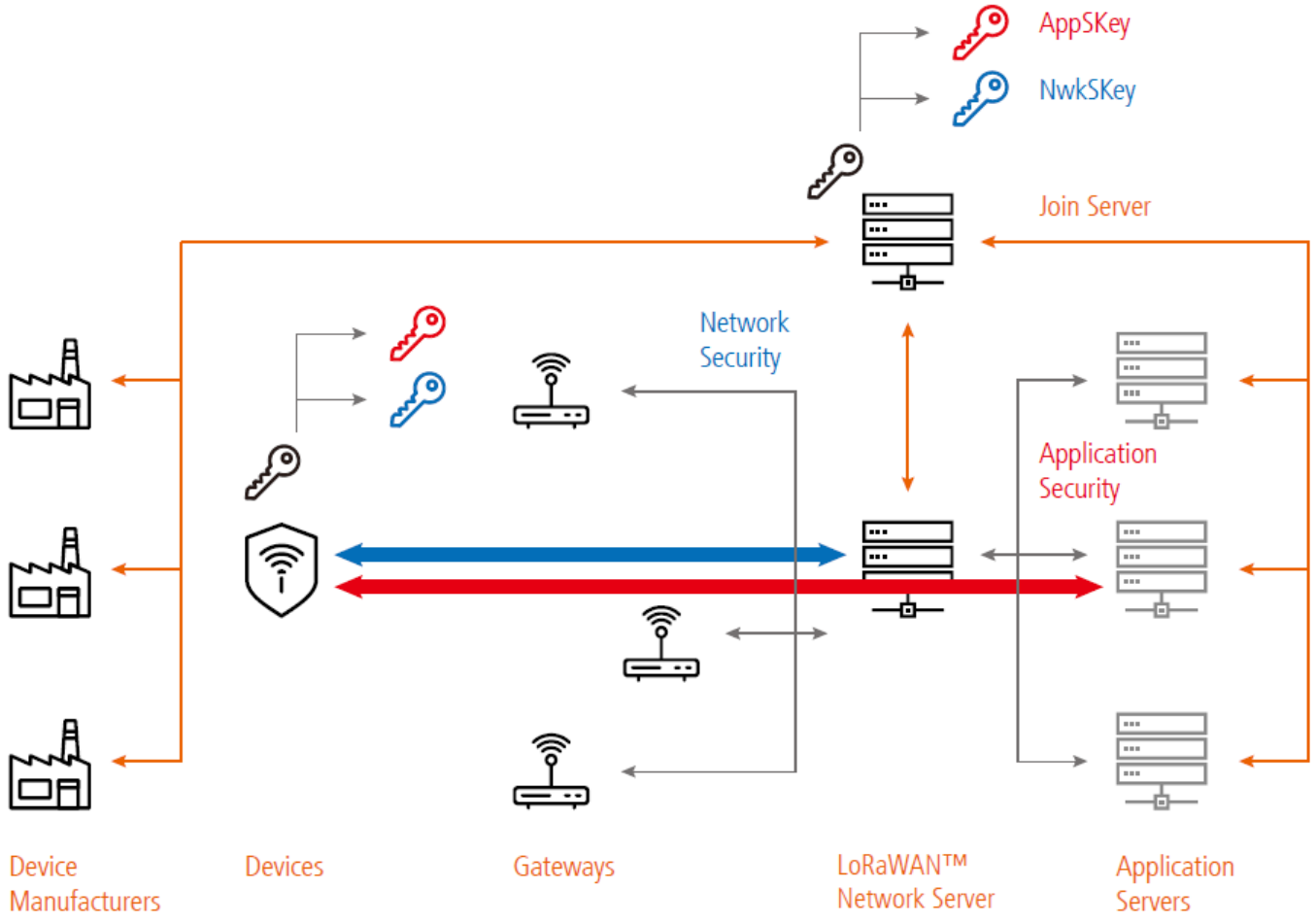
LoRaWAN: Seguridad

Autenticidad

Integridad

Token

Encriptación



THE THINGS NETWORK

LoRaWAN: Personalización y activación OTAA vs ABP

ABP: Activación por personalización

- Devide Address, Network Session Key y Application Session Key programados “a fuego” en el nodo

OTAA: Activación por el aire

- El nodo envía el DeviceEUI y el ApplicationEUI, y a partir de la respuesta recibida y haciendo uso de la ApplicationKey, deriva el Devide Address, Network Session Key y Application Session Key.

LoRaWAN: Formato de un mensaje

Radio PHY layer:

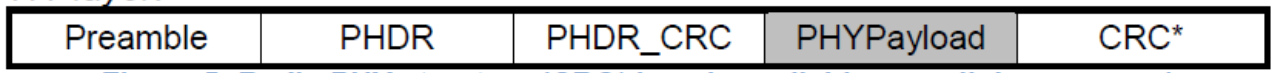


Figure 5: Radio PHY structure (CRC* is only available on uplink messages)

PHYPayload:

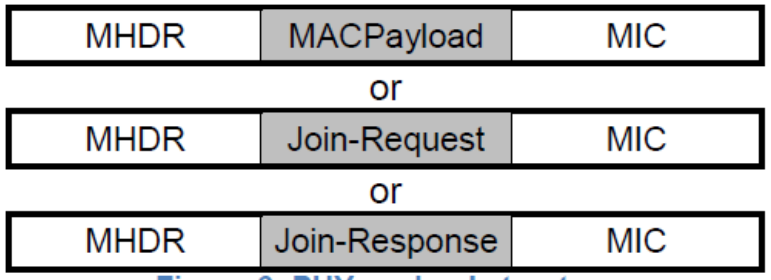


Figure 6: PHY payload structure

MACPayload:

Figure 7: MAC payload structure

FHDR:

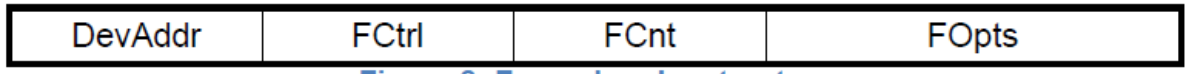


Figure 8: Frame header structure

THE THINGS NETWORK

THE THINGS NETWORK

APLICACIONES CIUDADANAS

THE THINGS NETWORK

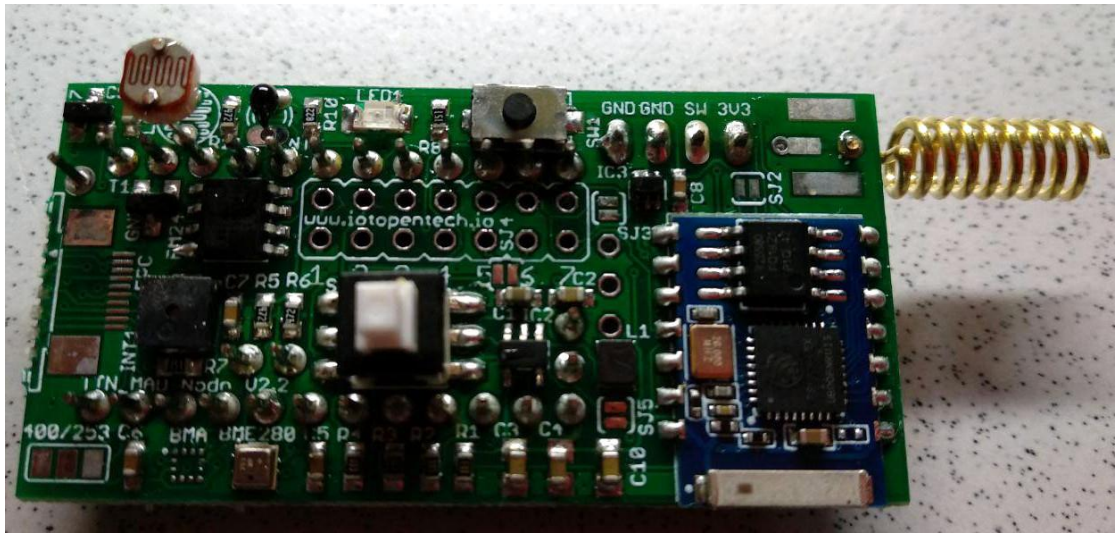
¿Qué es el nodo TTN MAD v2.2?

Es un nodo LoRaWAN diseñado con los siguientes requerimientos:

- Fácil de montar
- Económico (monetaria y energéticamente)
- Ampliable
- Muy flexible.

THE THINGS NETWORK

Nodo TTN MAD v2.2



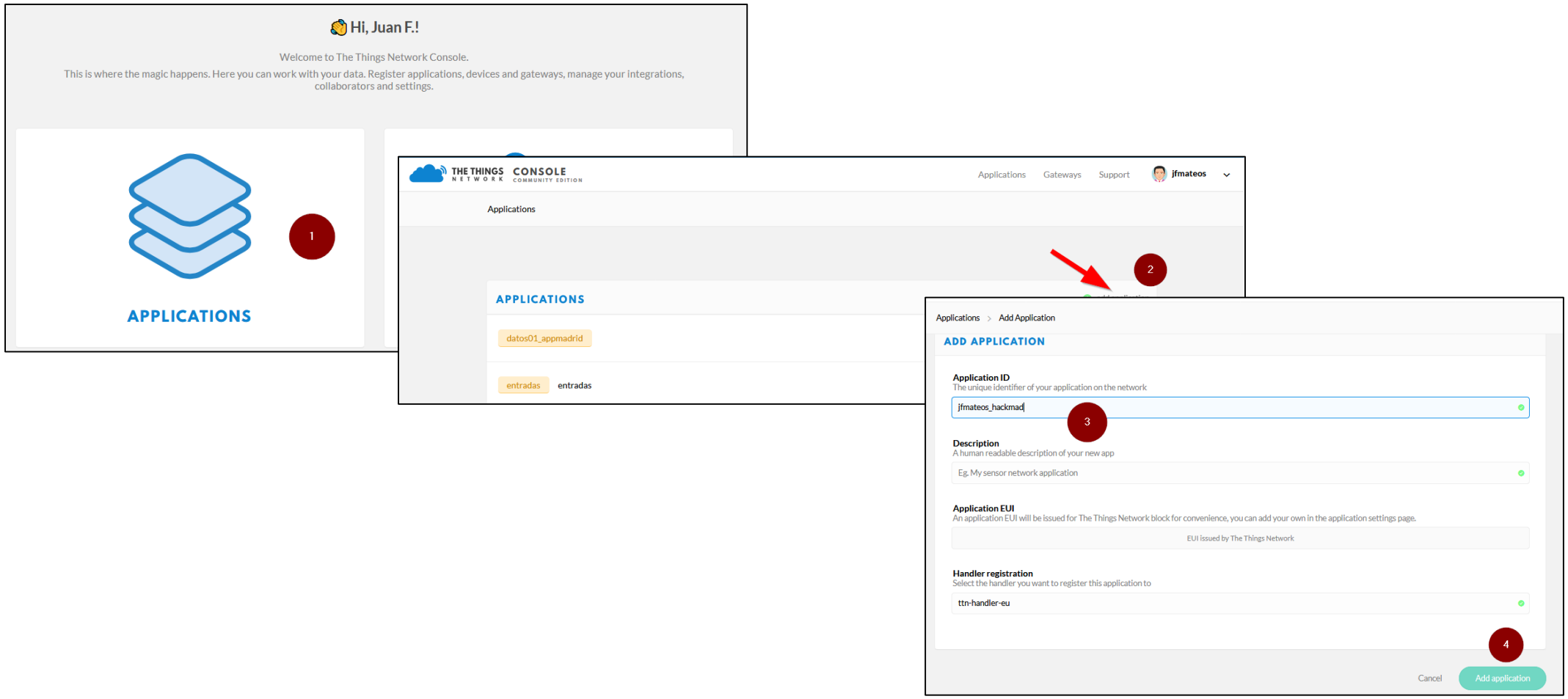
THE THINGS NETWORK

La consola de The Things Network

Vamos a crear una aplicación nueva y, dentro de ella, un dispositivo que se una a la red por ABP.

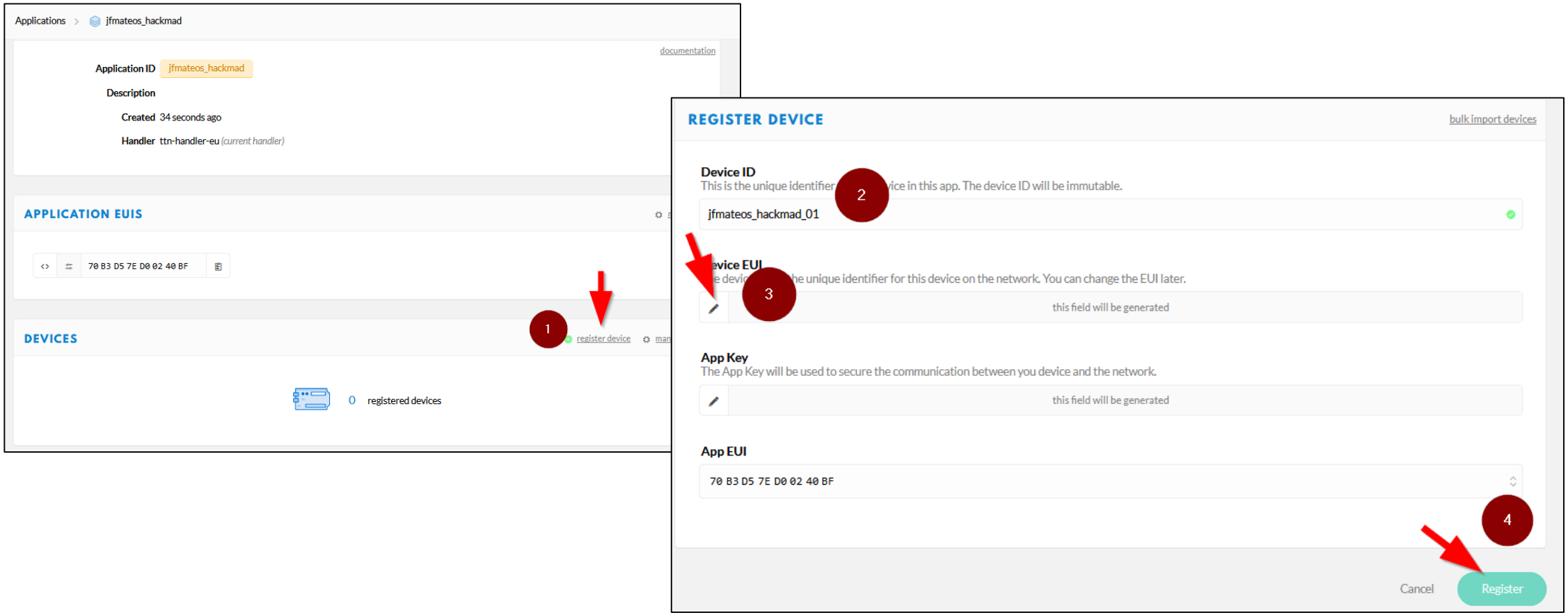
THE THINGS NETWORK

Crear una aplicación



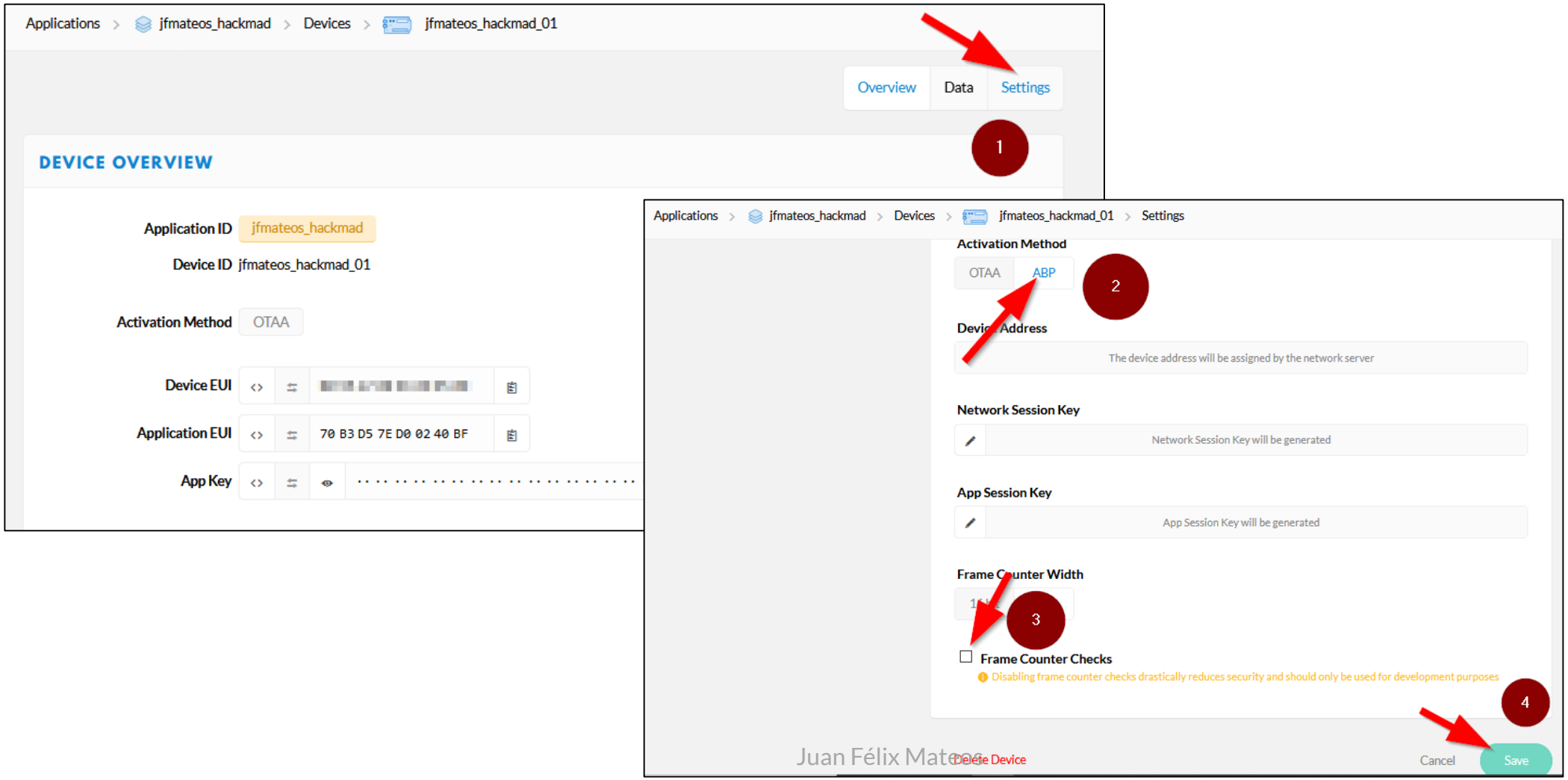
THE THINGS NETWORK

Añadir un dispositivo



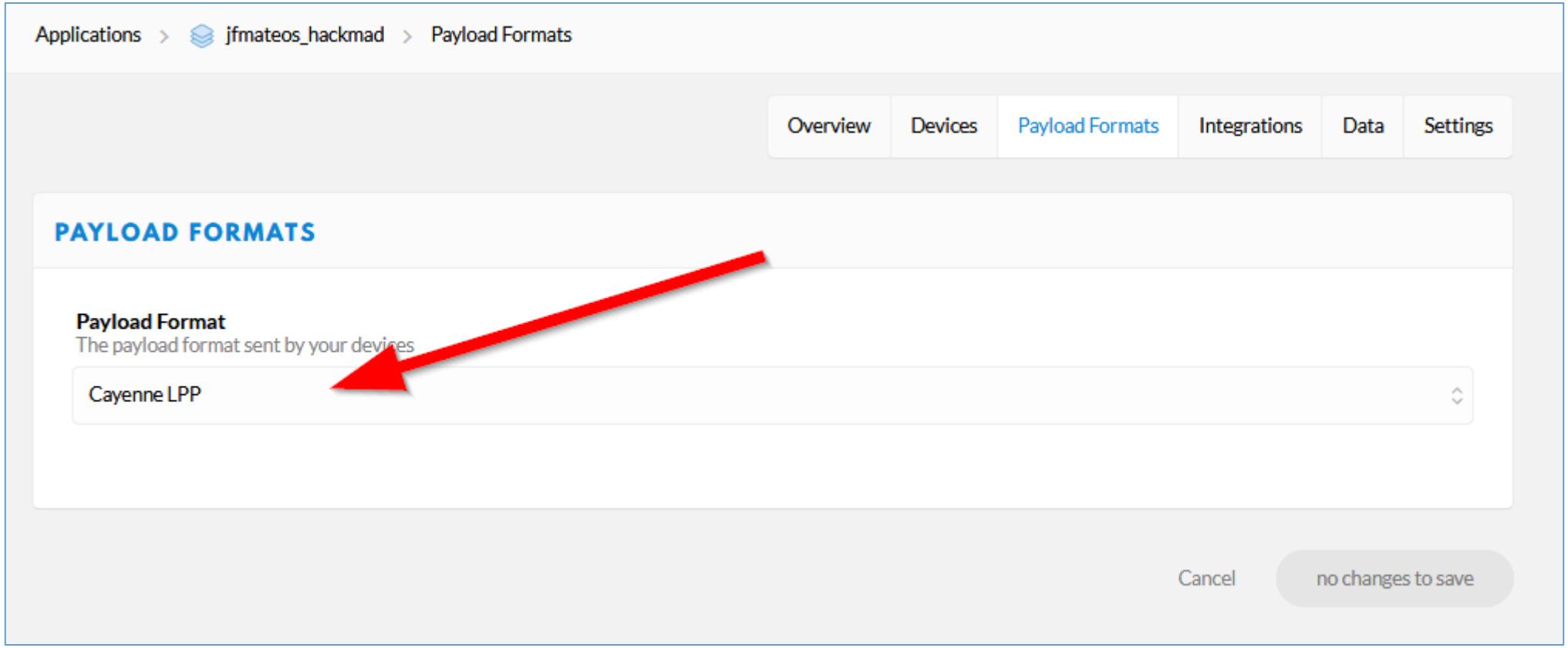
THE THINGS NETWORK

Configurar el dispositivo para activarse por ABP



THE THINGS NETWORK

Activar el formato de carga de pago Cayenne LPP



THE THINGS NETWORK

Programa en Arduino para el nodo TTN MAD v2.2

Vamos a crear un programa que envíe un frame con la tensión de la batería cada 30 segundos.

THE THINGS NETWORK

Librerías de Arduino

Instalar las siguientes librerías en el IDE de Arduino:

- MCCI LoRaWAN LMIC Library by IBM, Matthis Kooijman, Terry Moore, ChaeHee Won, Frank Rose. Versión 2.3.2.
- Low-Power by Rocket Scream Electronics. Versión 1.6.0.
- CayenneLPP by Electronics Cats. Versión 1.0.1

Configurar la librería MCCI LoRaWAN LMIC Library

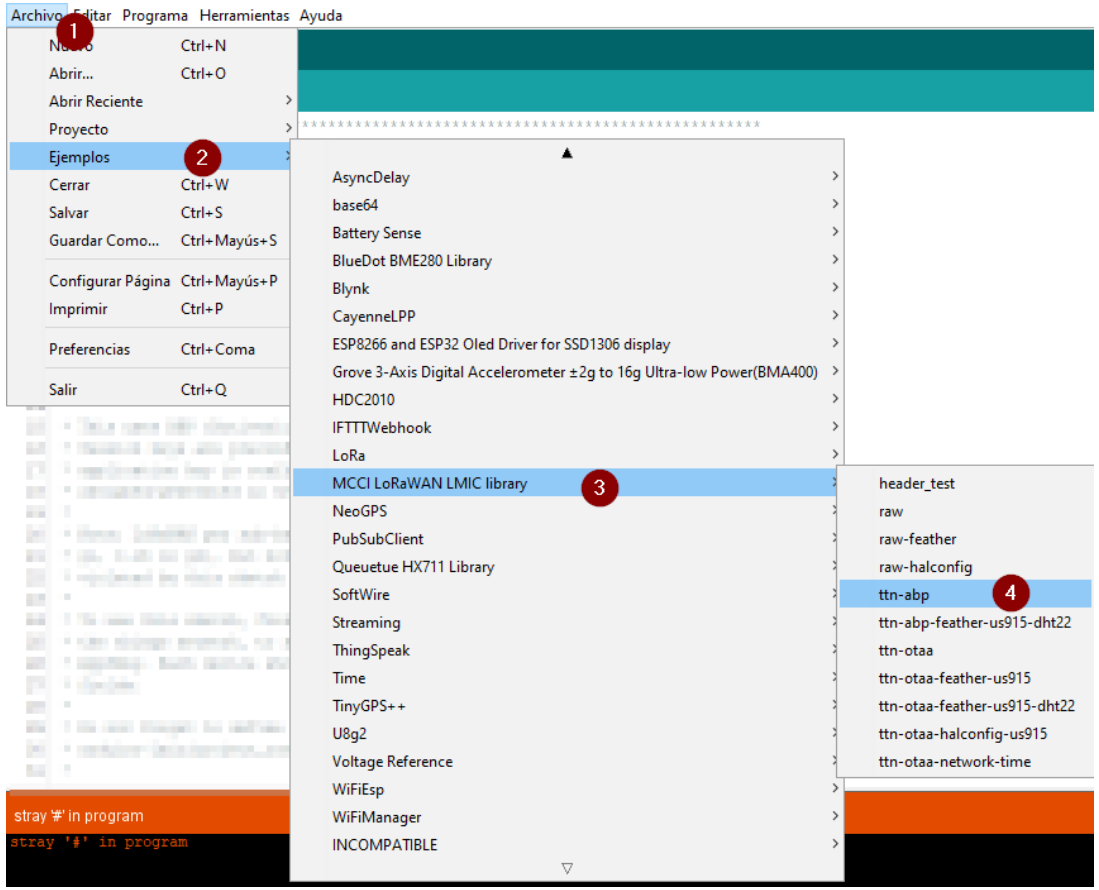
Para reducir al máximo los requisitos de memoria de la librería vamos a eliminar las funcionalidades que no necesitamos modificando el siguiente archivo:

MCCI_LoRaWAN_LMIC_library\project_config\lmic_project_config.h

```
// project-specific definitions

#define CFG_eu868 1

//#define CFG_us915 1

//#define CFG_au921 1//#define CFG_as923 1

// #define LMIC_COUNTRY_CODE
LMIC_COUNTRY_CODE_JP        /* for as923-JP */

//#define CFG_in866 1

#define CFG_sx1276_radio 1

//#define LMIC_USE_INTERRUPTS

#define DISABLE_PING

#define DISABLE_BEACONS
```

Punto de partida

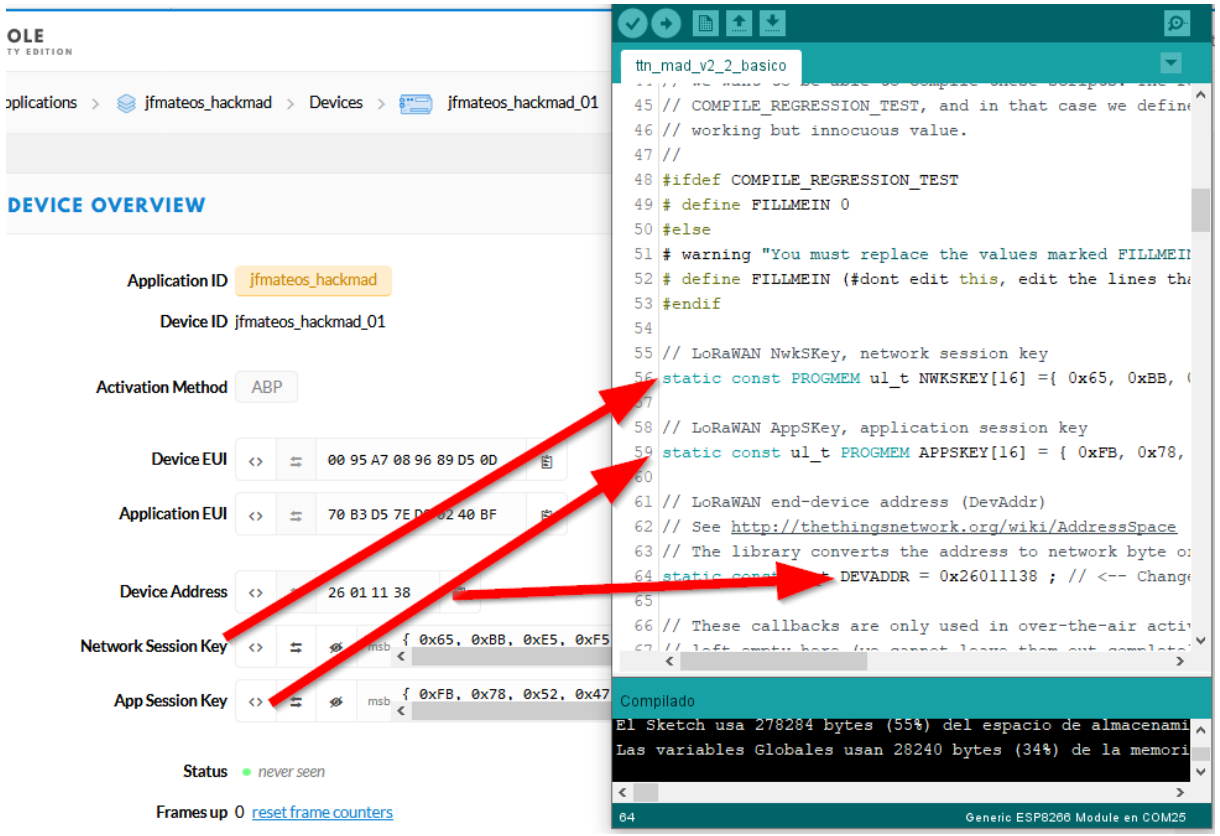
- Partimos del ejemplo ttn-abp de la librería MCCI LoRaWAN LMIC Library



THE THINGS NETWORK

Indicar las credenciales del nodo

- Copiamos el NwkSkey (msb), AppSkey (msb) y DevAddr (expresar en hexadecimal anteponiendo 0x) de la consola de TTN en los lugares correspondientes del código.



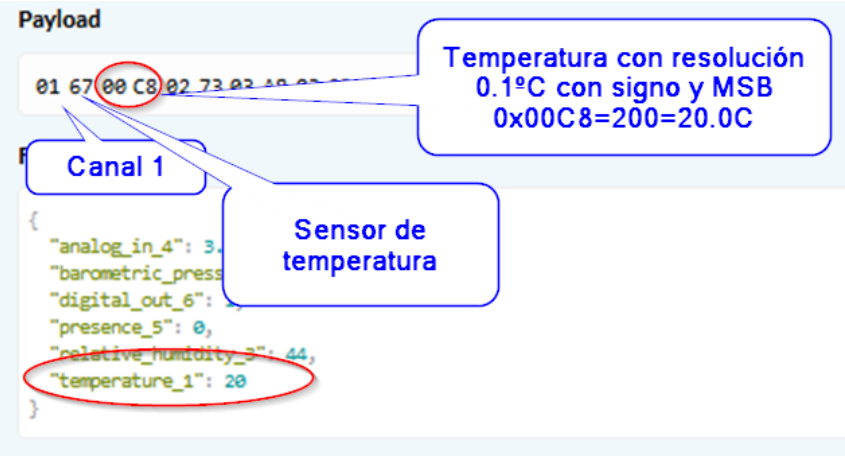
Modificar los pines que controlan el RFM95W

- Configurar la sección //Pin mapping como se indica

```
// Pin mapping
const lmic_pinmap lmic_pins = {
  .nss = 10,
  .rxtx = LMIC_UNUSED_PIN,
  .rst = 9,
  .dio = {2, 7, LMIC_UNUSED_PIN},
};
```

Cayenne LPP: Tipos de datos

- Cada sensor/actuador tiene asociado un tipo de dato



Type	IPSO	LPP	Hex	Data Size	Data Resolution per bit
Digital Input	3200	0	0	1	1
Digital Output	3201	1	1	1	1
Analog Input	3202	2	2	2	0.01 Signed
Analog Output	3203	3	3	2	0.01 Signed
Illuminance Sensor	3301	101	65	2	1 Lux Unsigned MSB
Presence Sensor	3302	102	66	1	1
Temperature Sensor	3303	103	67	2	0.1 °C Signed MSB
Humidity Sensor	3304	104	68	1	0.5 % Unsigned
Accelerometer	3313	113	71	6	0.001 G Signed MSB per axis
Barometer	3315	115	73	2	0.1 hPa Unsigned MSB
Gyrometer	3334	134	86	6	0.01 °/s Signed MSB per axis
GPS Location	3336	136	88	9	Latitude : 0.0001 ° Signed MSB Longitude : 0.0001 ° Signed MSB Altitude : 0.01 meter Signed MSB

THE THINGS NETWORK

Incluir las librerías lowpower y cayenne

```
37 #include <lmic.h>
38 #include <hal/hal.h>
39 #include <SPI.h>
40 #include "LowPower.h"
41 #include <CayenneLPP.h>
42 CayenneLPP lpp(4);
43 //
```

Insertar la función readVCC

https://gist.github.com/IoTopenTech

```
long readVcc() {

 ADMUX = _BV(REFS0) | _BV(MUX3) | _BV(MUX2) |
_BV(MUX1);

 delay(2);

 ADCSRA |= _BV(ADSC);

 while (bit_is_set(ADCSRA, ADSC));

 long result = ADCL;

 result |= ADCH << 8;

 result = 1126400L / result; // Back-calculate AVcc in mV

 return result;

}
```

```
86    .rxtx = LMIC_UNUSED_PIN,
87    .rst = 9,
88    .dio = {2, 7, LMIC_UNUSED_PIN},
89  };
90  long readVcc() {
91    ADMUX = _BV(REFS0) | _BV(MUX3) | _BV(MUX2) | _BV(MUX1);
92    delay(2);
93    ADCSRA |= _BV(ADSC);
94    while (bit_is_set(ADCSRA, ADSC));
95
96    long result = ADCL;
97    result |= ADCH << 8;
98    result = 1126400L / result; // Back-calculate AVcc in mV
99    return result;
100 }
101 void onEvent (ev_t ev) {
102     Serial.print(os_getTime());
103     Serial.print(": ");
```

Modificar el código de la función do_send

```
// Prepare upstream data transmission at the next possible time.
    lpp.reset();
    lpp.addAnalogInput(1, readVcc() / 1000.F);
    //LMIC_setTxData2(1, mydata, sizeof(mydata)-1, 0);
    LMIC_setTxData2(1, lpp.getBuffer(), lpp.getSize(), 0);
    Serial.println(F("Packet queued"));
```

```
181 }
182
183 void do_send(osjob_t* j){
184     // Check if there is not a current TX/RX job running
185     if (LMIC.opmode & OP_TXRXPEND) {
186         Serial.println(F("OP_TXRXPEND, not sending"));
187     } else {
188         // Prepare upstream data transmission at the next possible time.
189         lpp.reset();
190         lpp.addAnalogInput(1,readVcc() / 1000.F);
191         //LMIC_setTxData2(1, mydata, sizeof(mydata)-1, 0);
192         LMIC_setTxData2(1, lpp.getBuffer(), lpp.getSize(), 0);
193         Serial.println(F("Packet queued"));
194     }
195     // Next TX is scheduled after TX_COMPLETE event.
196 }
197
198 void setup() {
```


THE THINGS NETWORK

Modificar el código del estado EV_TXCOMPLETE

```
Serial.println("Me voy a echar una siestecita");
    delay(1000);// Para que tenrminen de imprimirse los mensajes en el terminal
    for (byte contador = 0; contador < 3; contador++) {

LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
    }
    os_setCallback (&sendjob, do_send);
```

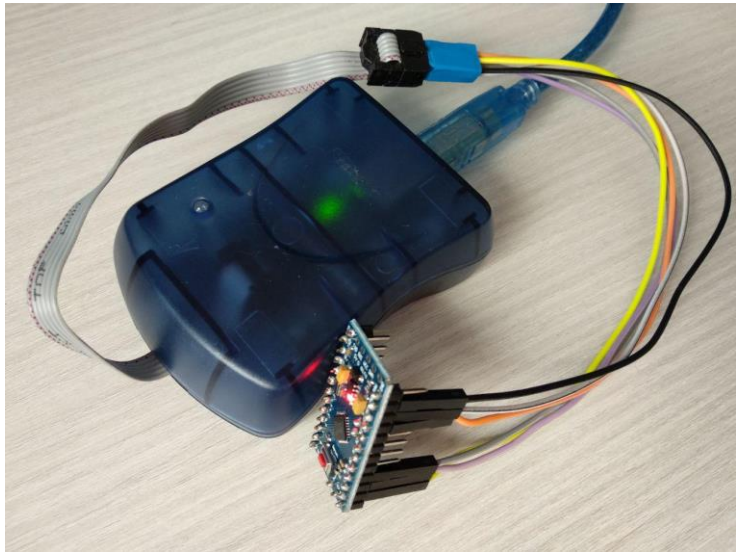
```
137   case EV_TXCOMPLETE:
138     Serial.println(F("EV_TXCOMPLETE (includes waiting for RX windows)"));
139     if (LMIC.txrxFlags & TXRX_ACK)
140       Serial.println(F("Received ack"));
141     if (LMIC.dataLen) {
142       Serial.println(F("Received "));
143       Serial.println(LMIC.dataLen);
144       Serial.println(F(" bytes of payload"));
145     }
146     // Schedule next transmission
147     //os_setTimedCallback(&sendjob, os_getTime()+sec2osticks(TX_INTERVAL), do_send);
148     Serial.println("Me voy a echar una siestecita");
149     delay(1000);// Para que tenrminen de imprimirse los mensajes en el terminal
150     for (byte contador = 0; contador < 3; contador++) {
151       LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
152     }
153     os_setCallback (&sendjob, do_send);
154     break;
155   case EV_LOST_TSYNC:
```

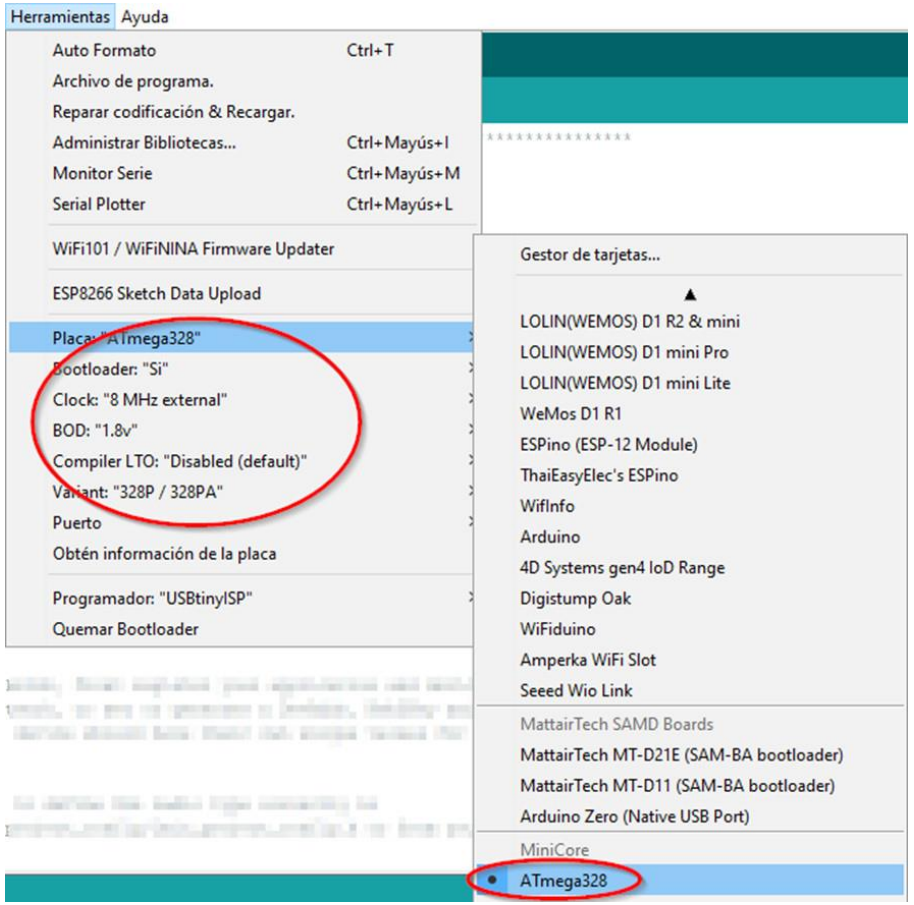

THE THINGS NETWORK

Sustituir el bootloader por el de MiniCore (opcional)

Sustituir el bootloader del Arduino Pro Mini por la versión de MiniCore (https://github.com/MCUdude/MiniCore) usando un programador (por ejemplo, el USBtinyISP). Para conectar el programador al Arduino se usarán los siguientes pines:

- VCC
- GND
- RST
- 11: MOSI
- 12: MISO
- 13: SCK



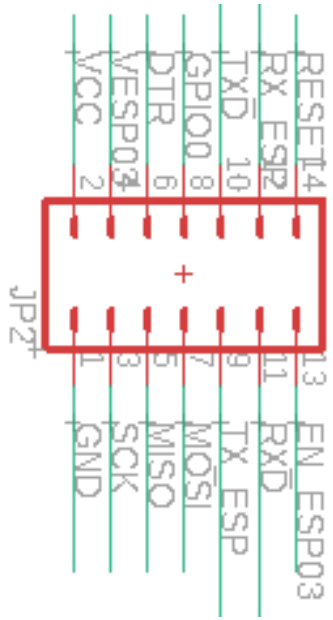


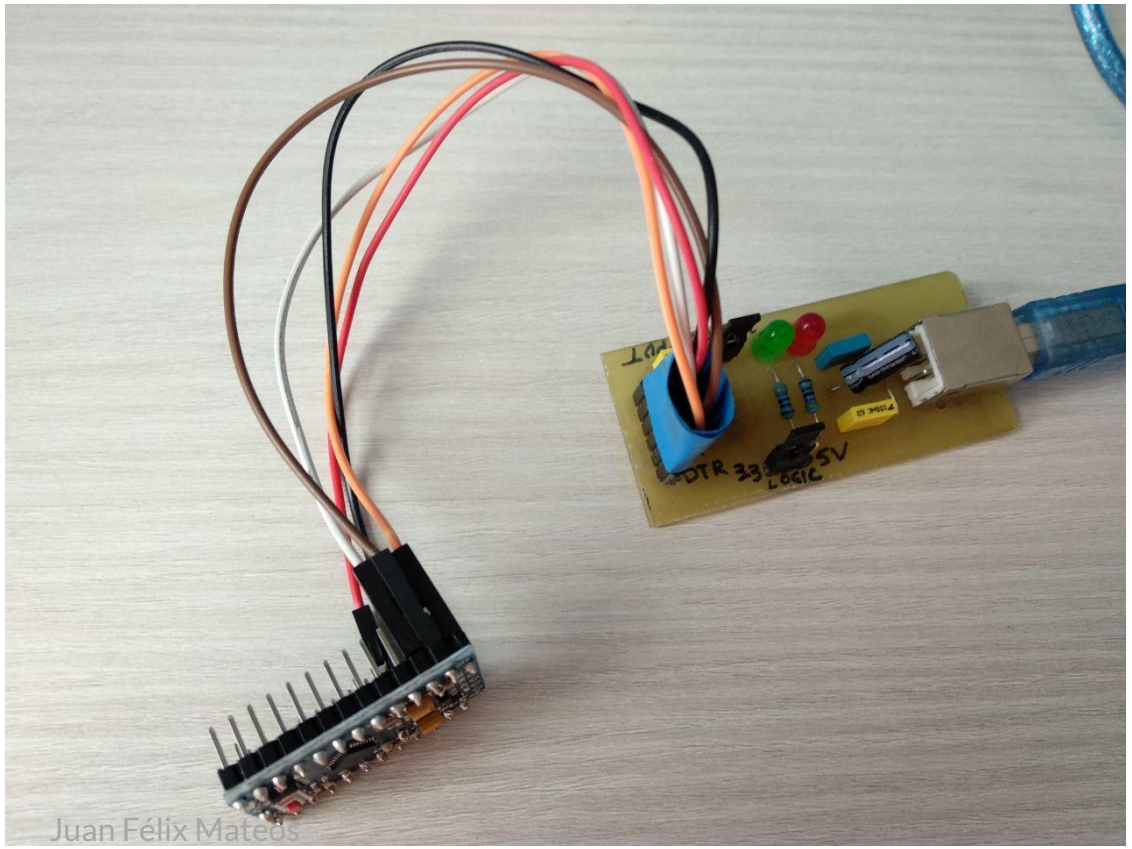
THE THINGS NETWORK

Programar el Arduino Pro Mini

Cargar el programa en el Arduino Pro Mini

usando un conversor USB → Serial





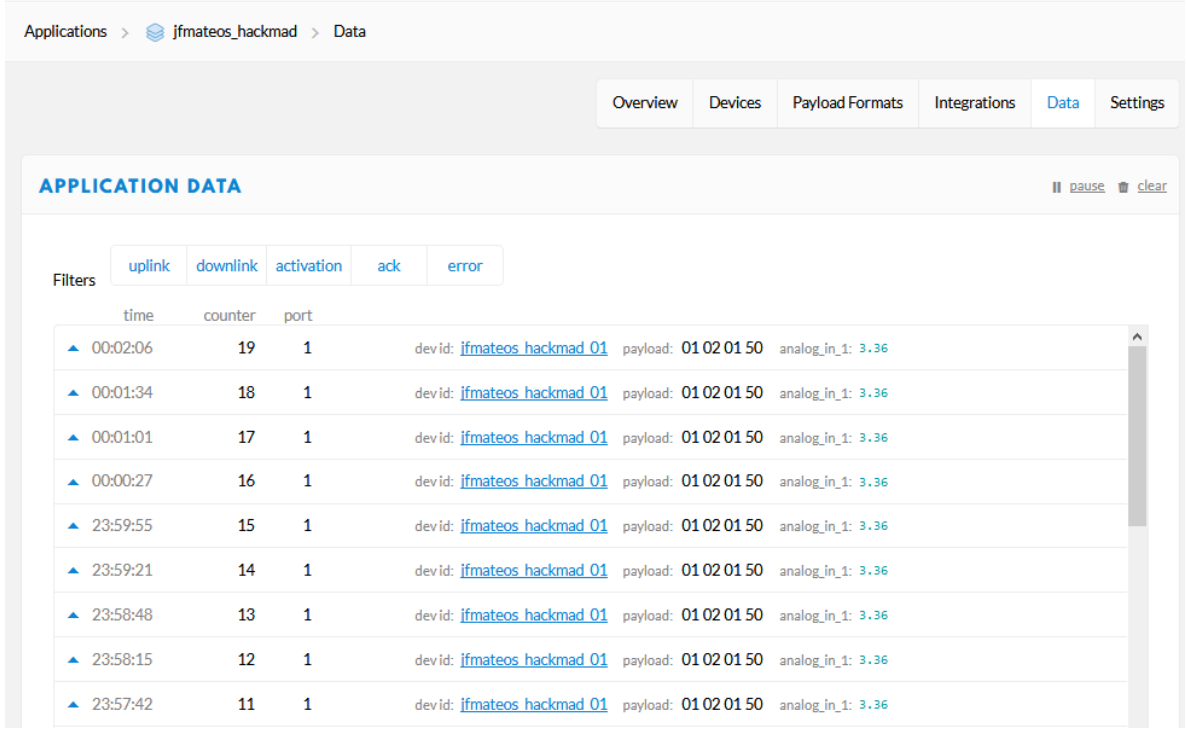
THE THINGS NETWORK

Puesta en funcionamiento

Insertar las baterías.

Pulsar el botón de encencido.

Los frames empezarán a aparecer en la consola de The Things Network.



THE THINGS NETWORK

BME280: Incluir las librerías y ampliar la carga de pago

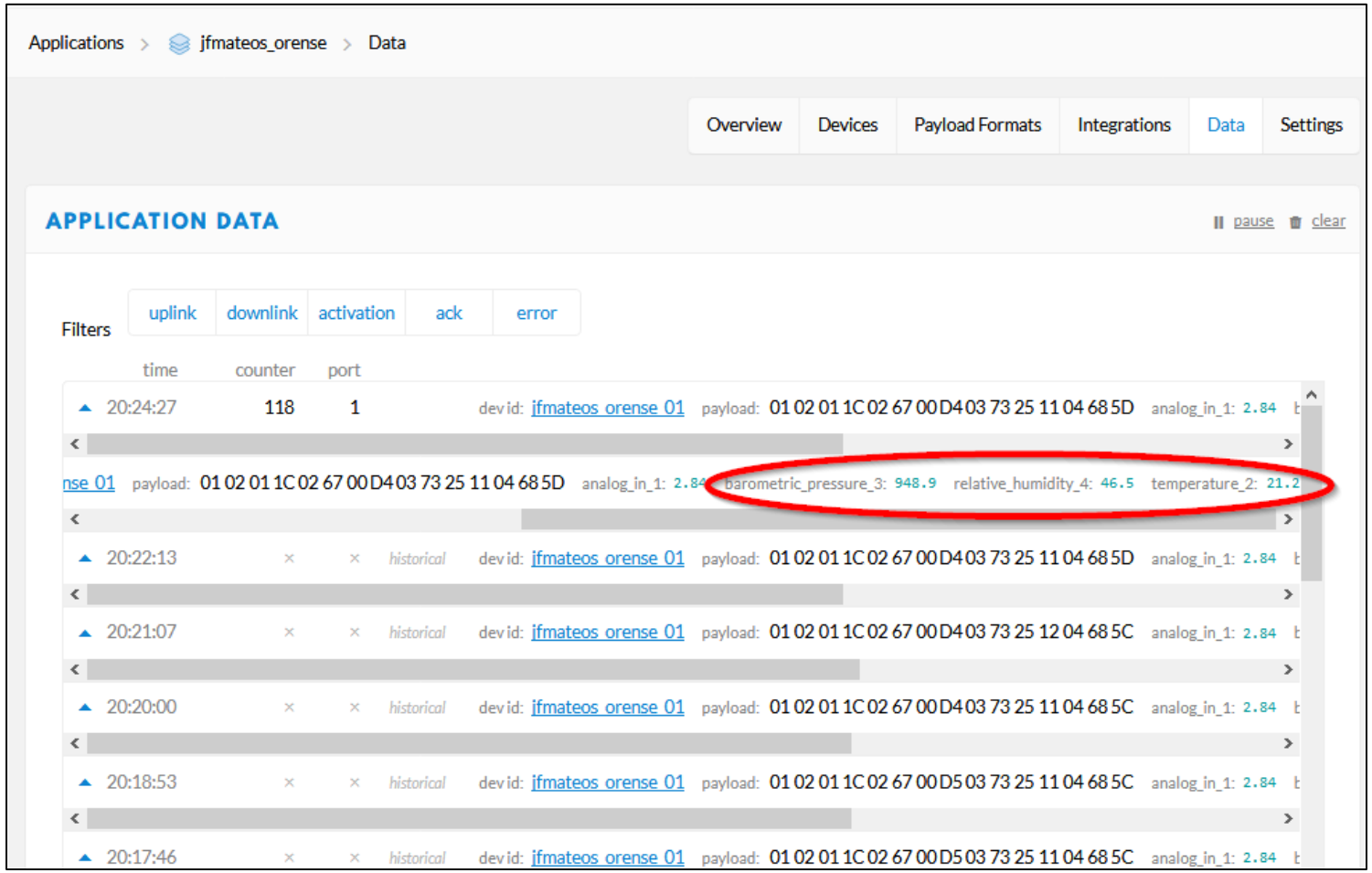
```
37 #include <lmic.h>
38 #include <hal/hal.h>
39 #include <Wire.h>
40 #include <SPI.h>
41 #include "LowPower.h"
42 #include <Adafruit_Sensor.h>
43 #include <Adafruit_BME280.h>
44 //En el archivo Adafruit_BME280.h hay que cambiar la siguiente línea
45 // #define BME280_ADDRESS                (0x76)
46 #define SEALEVELPRESSURE_HPA (1013.25)
47 Adafruit_BME280 bme; // I2C
48
49 #include <CayenneLPP.h>
50 CayenneLPP lpp(16);//Sólo voy a pasar la tensión y el BME280
51
```

BME280: Añadir la temperatura, presión y humedad

```
if (bme.begin()) {

bme.setSampling(Adafruit_BME280::MODE_FORCED,
                Adafruit_BME280::SAMPLING_X1, //
temperature
                Adafruit_BME280::SAMPLING_X1, //
pressure
                Adafruit_BME280::SAMPLING_X1, //
humidity
                Adafruit_BME280::FILTER_OFF   );
    lpp.addTemperature(2, bme.readTemperature());
    lpp.addBarometricPressure(3,
bme.readPressure() / 100.0F);
    lpp.addRelativeHumidity(4, bme.readHumidity());
  }
```

```
200   lpp.reset();
201   lpp.addAnalogInput(1, readVcc() / 1000.F);
202
203   //LMIC_setTxData2(1, mydata, sizeof(mydata)-1, 0);
204   if (bme.begin()) {
205     bme.setSampling(Adafruit_BME280::MODE_FORCED,
206                     Adafruit_BME280::SAMPLING_X1, // temperature
207                     Adafruit_BME280::SAMPLING_X1, // pressure
208                     Adafruit_BME280::SAMPLING_X1, // humidity
209                     Adafruit_BME280::FILTER_OFF   );
210     lpp.addTemperature(2, bme.readTemperature());
211     lpp.addBarometricPressure(3, bme.readPressure() / 100.0F);
212     lpp.addRelativeHumidity(4, bme.readHumidity());
213   }
214   LMIC_setTxData2(1, lpp.getBuffer(), lpp.getSize(), 0);
215   //LMIC_setTxData2(1, mydata, sizeof(mydata)-1, 0);
216   Serial.println(F("Packet queued"));
217 }
```

BME280: Comprobar en la consola



THE THINGS NETWORK

LED: Downlink

```
222         lpp.addBarometricPressure(4, bme.readPressure() / 100.0F);
223         lpp.addRelativeHumidity(5, bme.readHumidity());
224       }
225       lpp.addDigitalOutput(6, 1);
226       LMIC_setTxData2(1, lpp.getBuffer(), lpp.getSize(), 0);
227       Serial.println(F("Packet queued"));
228     }
229     // Next TX is scheduled after TX_COMPLETE event.
230 }
231
232 void setup() {
233     pinMode(4,OUTPUT);
234     digitalWrite(4,LOW);
235     //    pinMode(13, OUTPUT);
236     while (!Serial); // wait for Serial to be initialized
```

LED: Downlink

```
37 #include <hal/hal.h>
38 #include <Adafruit_Sensor.h>
39 #include <Adafruit_BME280.h>
40 #include <CayenneLPP.h>
41 CayenneLPP lpp(22);
42 //En el archivo Adafruit_BME280.h hay que cambiar la siguiente línea
```

LED: Downlink

```
152     case EV_TXCOMPLETE:
153       Serial.println(F("EV_TXCOMPLETE (includes waiting for RX windows)"));
154       if (LMIC.txrxFlags & TXRX_ACK)
155         Serial.println(F("Received ack"));
156       if (LMIC.dataLen) {
157         //Serial.println(F("Received "));
158         //Serial.println(LMIC.dataLen);
159         //Serial.println(F(" bytes of payload"));
160         //El downlink tendrá el formato canal (06), tipo de valor (00=digital), valor (0 o 100=0x64), FF (ACK)
161         if (LMIC.frame[LMIC.dataBeg + 2] == 100) {
162           digitalWrite(4, HIGH);
163         } else {
164           digitalWrite(4, LOW);
165         }
166       }
```

LED: Downlink

Probar a encender y apagar el LED simulando los siguientes downlinks:

- 06 01 64 → Encender
- 06 01 00 → Apagar

THE THINGS NETWORK

OTAA + HALL + Interrupciones + Heartbeat

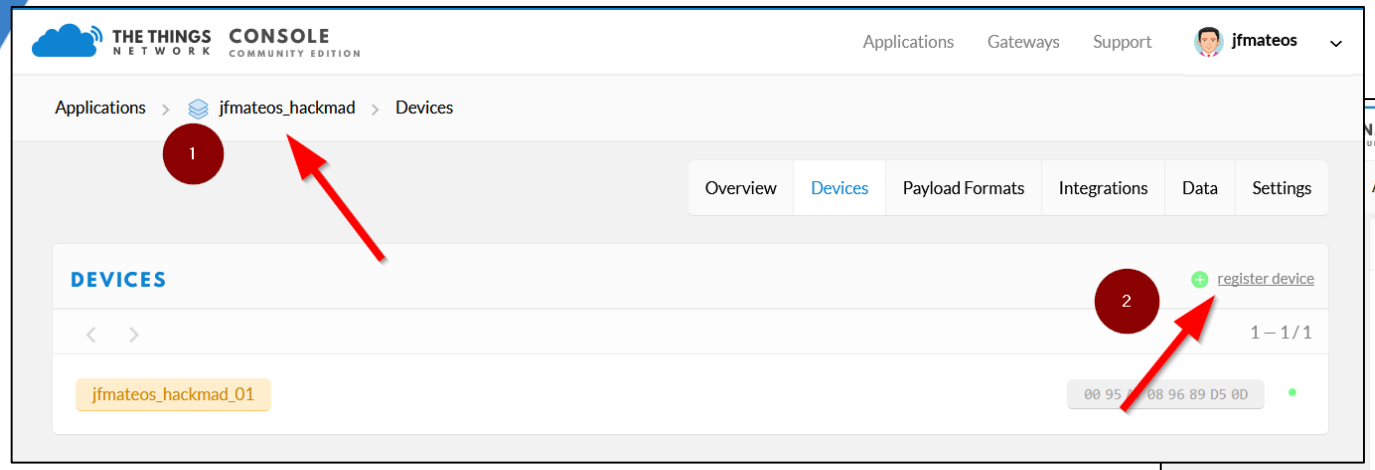
Vamos a programar el nodo para que, cada vez que se detecte un cambio en el estado del sensor HALL, envíe un mensaje y, adicionalmente, también envíe un heartbeat cada minuto.

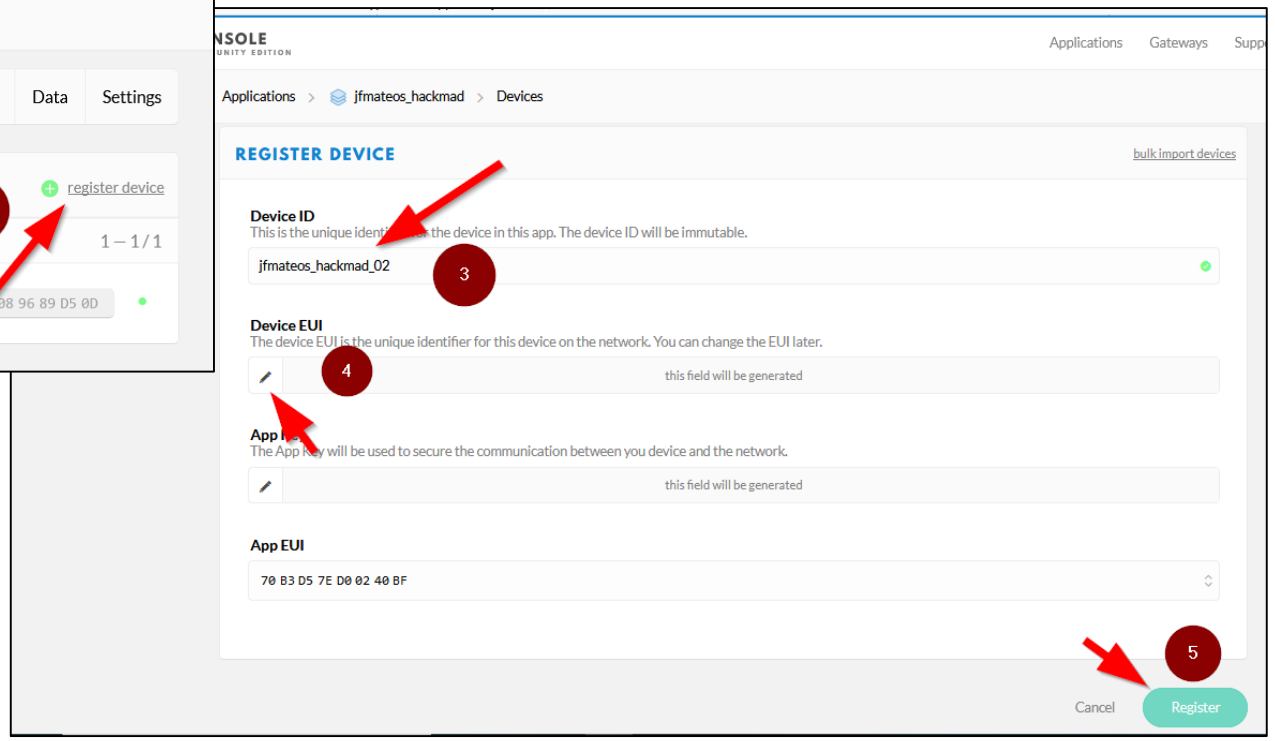
Partimos del ejemplo MCCI LoRaWAN Library LMIC Library:

- ttn-otaa.ino

OTAA + HALL + Interrupciones + Heartbeat

Dar de alta un dispositivo OTAA



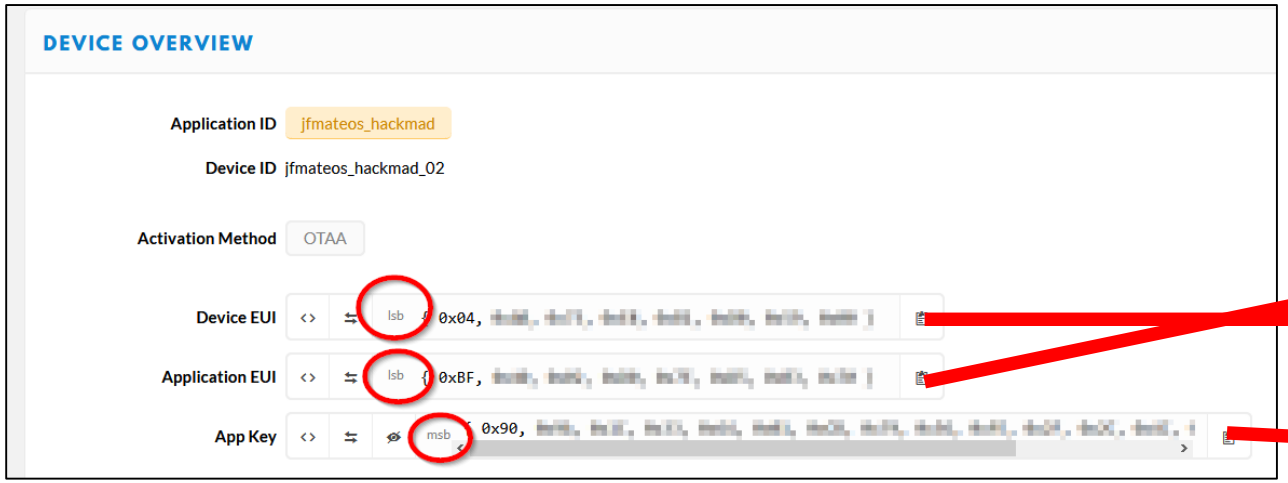


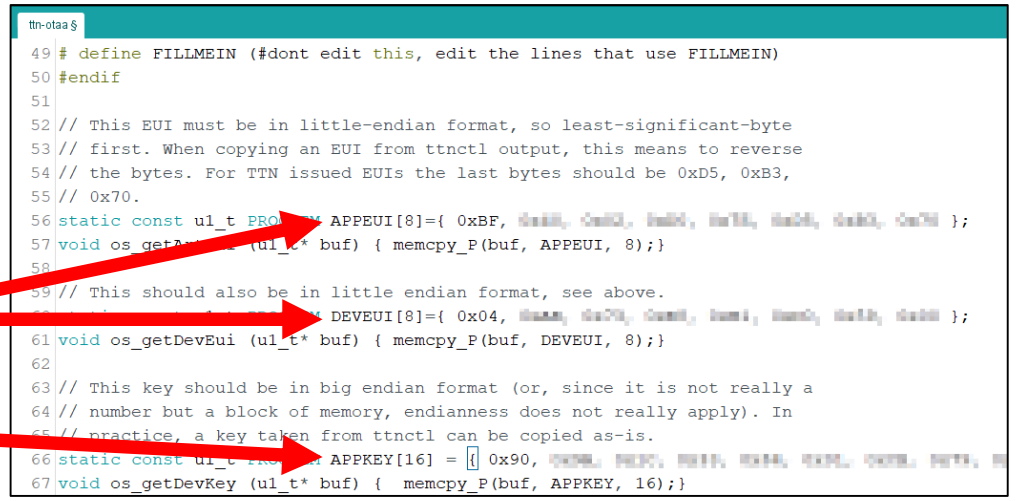
THE THINGS NETWORK

OTAA + HALL + Interrupciones + Heartbeat

Configurar el nodo en Arduino

¡PRECAUCIÓN! El AppEUI y DeviceEUI deben ir en **little-endian format**, pero el AppKey debe ir en **big-endian format**





THE THINGS NETWORK

Modificar los pines que controlan el RFM95W

- Configurar la sección //Pin mapping como se indica

```
// Pin mapping
const lmic_pinmap lmic_pins = {
  .nss = 10,
  .rxtx = LMIC_UNUSED_PIN,
  .rst = 9,
  .dio = {2, 7, LMIC_UNUSED_PIN},
};
```

Incluir las librerías lowpower y cayenne

```
34 #include <lmic.h>
35 #include <hal/hal.h>
36 #include <SPI.h>
37 #include "LowPower.h"
38 #include <CayenneLPP.h>
39 CayenneLPP lpp(7);
```

Insertar la función readVCC

https://gist.github.com/IoTopenTech

```
long readVcc() {

 ADMUX = _BV(REFS0) | _BV(MUX3) | _BV(MUX2) |
_BV(MUX1);

 delay(2);

 ADCSRA |= _BV(ADSC);

 while (bit_is_set(ADCSRA, ADSC));

 long result = ADCL;

 result |= ADCH << 8;

 result = 1126400L / result; // Back-calculate AVcc in mV

 return result;

}
```

```
86    .rxtx = LMIC_UNUSED_PIN,
87    .rst = 9,
88    .dio = {2, 7, LMIC_UNUSED_PIN},
89  };
90  long readVcc() {
91    ADMUX = _BV(REFS0) | _BV(MUX3) | _BV(MUX2) | _BV(MUX1);
92    delay(2);
93    ADCSRA |= _BV(ADSC);
94    while (bit_is_set(ADCSRA, ADSC));
95
96    long result = ADCL;
97    result |= ADCH << 8;
98    result = 1126400L / result; // Back-calculate AVcc in mV
99    return result;
100 }
101 void onEvent (ev_t ev) {
102     Serial.print(os_getTime());
103     Serial.print(": ");
```


THE THINGS NETWORK

Crear las variables necesaria para la lógica del programa

```
37 #include "LowPower.h"
38 #include <CayenneLPP.h>
39 CayenneLPP lpp(7);
40
41 volatile boolean envioEnCurso = false;
42 boolean puertaAbierta;
43 volatile boolean interrumpido = false;
44 byte minutosHearbeat = 1;
45
```

Modificar la función setup()

```
238   LMIC_reset();
239
240   // Start job (sending automatically starts OTAA too)
241   LMIC_setClockError(MAX_CLOCK_ERROR * 1 / 100); //Para mejorar la recepción de los downlinks
242   //Conecto en el pin 3 un interruptor a GND normalmente abierto
243   pinMode(3, INPUT);
244   delay(100);
245   // Start job
246   puertaAbierta = digitalRead(3);
247   enviarMensaje();
248   //do_send(&sendjob);
249 }
```

La ISR (Interrupt Service Routine) enviarMensaje()

```
104   result = 1126400L / result; // Back-
105   return result;
106 }
107
108 void enviarMensaje() {
109   envioEnCurso = true;
110   interrumpido = true;
111   os_setCallback (&sendjob, do_send);
112 }
113
114 void onEvent (ev_t ev) {
115   Serial.print(os_getTime());
116   Serial.print(": ");
```

Modificar el estado EV_TXCOMPLETE

```
166     case EV_TXCOMPLETE:
167       Serial.println(F("EV_TXCOMPLETE (includes waiting for RX windows)"));
168       if (LMIC.txrxFlags & TXRX_ACK)
169         Serial.println(F("Received ack"));
170       if (LMIC.dataLen) {
171         Serial.print(F("Received "));
172         Serial.print(LMIC.dataLen);
173         Serial.println(F(" bytes of payload"));
174       }
175       // Schedule next transmission
176       //os_setTimedCallback(&sendjob, os_getTime() + sec2osticks(TX_INTERVAL), do_send);
177       envioEnCurso = false;
178       break;
179     case EV_LOST_TSYNC:
```

Incluir la carga de pago en el método do_send

```
213 void do_send(osjob_t* j) {
214   // Check if there is not a current TX/RX job running
215   if (LMIC.opmode & OP_TXRXPEND) {
216     Serial.println(F("OP_TXRXPEND, not sending"));
217   } else {
218     // Prepare upstream data transmission at the next possible time.
219     //LMIC_setTxData2(1, mydata, sizeof(mydata) - 1, 0);
220     lpp.reset();
221     lpp.addAnalogInput(1, readVcc() / 1000.F);
222     lpp.addDigitalInput(2, puertaAbierta);
223
224     LMIC_setTxData2(1, lpp.getBuffer(), lpp.getSize(), 0);
225     Serial.println(F("Packet queued"));
226   }
227   // Next TX is scheduled after TX_COMPLETE event.
228 }
```

El método loop()

```
void loop() {
 //Esperamos a que concluya el envío
 while (envioEnCurso) {
  os_runloop_once();
 }
 delay(100);
 interrumpido = false;
```

```
if (puertaAbierta == digitalRead(3)) {
  //La puerta no ha cambiado de estado
  attachInterrupt(digitalPinToInterrupt(3), enviarMensaje, (puertaAbierta == 1 ? FALLING : RISING));
  for (byte minutos = 0; minutos < minutosHearbeat; minutos++) {
   if (interrumpido == true) {
    break;
   }
   for (byte contador = 0; contador < 7; contador++) {
    if (interrumpido == true) {
     break;
    }
    LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
   }
  }
```

El método loop()

```
if (!interrumpido) {
    enviarMensaje();
  } else {
    puertaAbierta = !puertaAbierta;
  }
 } else {
  //La puerta ha cambiado de estado durante el
envío del mensaje de estado anterior
  puertaAbierta = !puertaAbierta;
  enviarMensaje();
 }
}
```

- Si se está enviando un mensaje → Atender la máquina de estados
- Si no, comprobar si ha cambiado el estado de la puerta.
- Si sí ha cambiado, enviar mensaje.
- Si no ha cambiado, dormir en intervalos de 8 segundos mientras no se detecte un cambio de estado de la puerta o no se alcance el tiempo máximo de heartbear.
- Si se alcanza el tiempo de heartbet, enviarlo
- Si se detecta un cambio de estado de la puerta, salir del sueño y notificarlo.

YOU ARE THE NETWORK

LET'S BUILD THIS THING TOGETHER!

Juan Félix Mateos

juanfelixmateos@gmail.com

www.thethingsnetwork.org