



MÁSTER UNIVERSITARIO EN INGENIERÍA DE SISTEMAS Y DE CONTROL

Trabajo Final de Máster

SISTEMA DE MONITORIZACIÓN BASADO EN TECNOLOGÍAS IOT APLICADO A ENTORNOS AGRÍCOLAS

Estudiante: **ÁNGEL JESÚS MALDONADO CRIADO**

Director: **JOSÉ SÁNCHEZ MORENO**

Curso:2020/2021
Defensa: Febrero 2021





MÁSTER UNIVERSITARIO EN INGENIERÍA DE SISTEMAS Y DE CONTROL

Trabajo Final de Máster

SISTEMA DE MONITORIZACIÓN BASADO EN TECNOLOGÍAS IOT APLICADO A ENTORNOS AGRÍCOLAS

Proyecto específico propuesto por el alumno.

Estudiante: **ÁNGEL JESÚS MALDONADO CRIADO**

Director: **JOSÉ SÁNCHEZ MORENO**

Autorización

Autorizamos a la Universidad Complutense y a la UNED a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la memoria de este Trabajo Fin de Máster, como el código, la documentación y/o el prototipo desarrollado.



Firmado: **Ángel Jesús Maldonado Criado**.

Agradecimientos

A mis padres y a mi hermano por su apoyo incondicional. Agradecer a mi hermano que me propusiera la idea de la que ha surgido el presente trabajo, así como los medios para probar en un entorno real el proyecto desarrollado.

A Silvia por toda la paciencia que ha tenido y tiene conmigo en esta aventura llamada vida. Hubiera sido muy difícil realizar este Máster sin su ayuda y apoyo constante.

A mi director del Trabajo Fin de Máster, José Sánchez Moreno, por aceptar la dirección de este TFM y por sus consejos durante su realización.

'The hardest thing in life is to get knocked down and to get back up constantly, but we do it because we love it and we know that deep down, if it's the right idea, it could be bigger than all of us.'

Gordon. Halt and Catch Fire

Resumen

El paradigma del Internet de las Cosas (IoT) es un concepto que se basa en la inteconexión digital entre dispositivos, personas y la propia Internet permitiendo el intercambio de datos entre ellos. En los últimos años, con el crecimiento explosivo de las tecnologías del IoT, se puede encontrar un número cada vez mayor de aplicaciones en campos como la seguridad, la agricultura, el seguimiento de activos, las ciudades o los hogares inteligentes. Sin embargo, la baja disponibilidad de estos sistemas, complejidad, así como el coste elevado de algunas plataformas comerciales, crean un barrera de acceso a estas tecnologías.

El presente trabajo tiene como objetivo mostrar el desarrollo y despliegue real de un sistema IoT de bajo coste, aplicado a la monitorización de variables ambientales en entornos agrícolas. Tras la realización de una evaluación previa de las tecnologías actuales empleados en los sistemas IoT, se ha podido enfocar el desarrollo del sistema. En primer lugar se ha realizado el diseño electrónico y desarrollo del firmware del nodo sensor con capacidad para integrarse en una red de comunicaciones inalámbricas LoRaWAN. Además, el nodo permite la conexión cableada con diferentes sensores para medir la temperatura del suelo, temperatura y humedad del aire, humedad del suelo o geoposición.

También se ha detallado cómo realizar el despliegue de un gateway en una red LoRaWAN, así como su configuración e integración con el servidor de red The Things Network, que permite gestionar la conexión de los nodos sensores con la plataforma en red. La última parte del desarrollo ha consistido en el despliegue de una máquina virtual en un servidor remoto y el despliegue en esta de una base de datos y una plataforma de visualización para el acceso a todos los datos adquiridos por los nodos sensores.

Para garantizar el correcto funcionamiento del sistema se han definido diferentes pruebas efectuadas sobre una parte o un conjunto de elementos del mismo, por ejemplo el consumo energético de los nodos, sentido físico de los datos recibidos, recepción continuada de los datos, cobertura de la red. Tras está validación y mejoras, se ha realizado un despliegue completo del sistema en un entorno de producción real en el que se ha realizado en la última etapa de validación.

Palabras Clave

IoT, LoRa, redes de sensores, instrumentación electrónica, The Things Network, bajo coste, Arduino, nodeRED, Grafana, InfluxDB, sistemas empotrados.

Índice general

Índice general 1

Índice de figuras 4

Índice de tablas 7

Acrónimos 8

1. INTRODUCCIÓN 11

1.1. IoT en la agricultura . . . 12
1.2. Problemática detectada y propuesta inicial . . . 13
1.3. Estructura del trabajo . . . 14

2. ESTADO DEL ARTE 15

2.1. Sistema IoT . . . 16
2.2. Conectividad inalámbrica . . . 16
2.2.1. Narrow Band IoT (NB-IoT) . . . 18
2.2.2. LoRa/LoRaWAN . . . 18
2.2.3. Sigfox . . . 19
2.2.4. ¿Por qué LoRa/LoRaWAN ? . . . 20
2.3. Estructura de red LoRaWAN . . . 21
2.3.1. Nodos finales . . . 22
2.3.2. Gateways . . . 23
2.3.3. Servidores de red . . . 25
2.3.4. Servidores de aplicación . . . 26
2.4. Explotación de datos . . . 27
2.4.1. Base de datos . . . 27
2.4.2. Visualización de datos . . . 27
2.5. Control de versiones Git . . . 29

3. DISEÑO Y DESARROLLO **30**
3.1. Nodos - elementos sensores . . . 31
3.1.1. Unidad de control y procesamiento . . . 31
3.1.2. Transceptor de radio LoRa . . . 32
3.1.3. Convertidor DC-DC . . . 35
3.1.4. Interfaz de usuario . . . 36
3.1.5. Diseño de la PCB . . . 36
3.1.6. Sensores . . . 38
3.1.6.1. Humedad y temperatura ambiente - AM2302 . . . 38
3.1.6.2. Temperatura del suelo - DS18B20 . . . 40
3.1.6.3. Humedad del suelo . . . 41
3.1.7. Comunicaciones LoRaWAN . . . 43
3.1.8. Firmware . . . 49
3.1.8.1. Comunicación de radio . . . 50
3.1.8.2. Entorno en tiempo de ejecución . . . 51
3.1.8.3. Manejo del sensor . . . 51
3.1.8.4. Interfaz de línea de comandos . . . 52
3.1.8.5. Interfaz de dispositivo . . . 53
3.2. Despligue de red LoRaWAN - TheThingsNetwork . . . 53
3.2.1. Construcción y registro del Gateway. . . . 53
3.2.2. Registro de dispositivos en la red The Things Network (TTN) . . . 56
3.3. Plataforma . . . 59
3.3.1. Servidor - Google Cloud Platform . . . 59
3.3.2. Almacenamiento de datos - InfluxDB . . . 61
3.3.3. Conexión con TTN - Node-red . . . 62
3.3.4. Visualización de datos - Grafana . . . 64

4. VALIDACIÓN Y CARACTERIZACIÓN DEL SISTEMA **67**
4.1. Consumo energético . . . 68
4.2. Pruebas de nodos sensores . . . 72
4.3. Cobertura de la red . . . 78

5. DESPLIEGUE Y COSTE DEL SISTEMA **80**
5.1. Despliegue del sistema . . . 81
5.1.1. Mapa de cobertura . . . 82
5.1.2. Dashboard de visualización. . . . 83
5.2. Coste del desarrollo . . . 84
5.2.1. Coste material del sistema . . . 84
5.2.2. Coste del desarrollo . . . 86

6. CONCLUSIONES Y LÍNEAS FUTURAS **87**
6.1. Líneas futuras 89

Bibliografía **90**

A. Apéndice - Diseño Hardware **92**
A.1. Esquemático nodo LoRa 94
A.2. Gerbers 95
A.3. Escudo de radiación de temperatura 98

B. Apéndice - Software de plataforma **99**
B.1. Instalación y configuración del servicio NODE-RED 100
B.2. Instalación y configuración de InfluxDB 102
B.3. Instalación y configuración de Grafana 103

Índice de figuras

2.1. Velocidad de transmisión vs. alcance de diferentes tecnologías de radiocomunicación IoT. . . . 17
2.2. Comparativa de las tres soluciones LPWA: Sigfox, LoRaWAN y NB-IoT. . 20
2.3. Estructura de una red LoRaWAN. . . . 21
2.4. Shield Dragino LoRa 22
2.5. Arduino MKR WAN 1300. . . . 23
2.6. WaspMote Libellium. . . . 23
2.7. Gateway outdoor RAK7249. . . . 24
2.8. Gateway indoor LPS8. . . . 24
2.9. Raspberry PI - RAK831. . . . 24
2.10. Arquitectura de microservicios del stack LoRaWAN implementado en The Things Network. . . . 26
2.11. Ejemplo de paneles de visualización implementados en Grafana. . . . 28

3.1. Arduino Pro Mini. . . . 31
3.2. Módulo RFM95W. . . . 33
3.3. Diagrama de conexión del bus SPI. . . . 34
3.4. Diagrama conexión RFM95W con el microcontrolador. . . . 35
3.5. Convertidor DC/DC Buck. . . . 36
3.6. Render de la Printed Circuit Board (PCB) del nodo sensor generado con KiCAD. . . . 37
3.7. Imagen real de la PCB del nodo sensor sin componentes. . . . 37
3.8. Imagen real del nodo sensor ya montado con todos sus componentes. . . . 38
3.9. Sensor AM2302/DHT22 con conexionado. . . . 39
3.10. Sensor DS18B20 en sus dos formatos de encapsulado. . . . 40
3.11. Diagrama de conexión DS1B20 con el microcontrolador. . . . 41
3.12. Sensor capacitivo de humedad del suelo. . . . 41
3.13. Diagrama de conexión del bus I^2C. . . . 42
3.14. Relación entre la latencia y la duración de la batería para distintas clases de nodos. . . . 43

3.15. Protocolo de activación OTAA. 45
3.16. Chirps con distinto factor de dispersión. 46
3.17. Relación distancia vs bitrate vs airtime. 46
3.18. Tiempo de uso máximo del canal según el tamaño del mensaje y el SF. . . . 48
3.19. Capas de abstracción de la librería LMIC. 50
3.20. Concentrador LoRa iC880A-SPI. 53
3.21. Placa de adaptación ic880A para Raspberry Pi. 54
3.22. Vista superior del gateway. 55
3.23. Registro del gateway en la red TTN. 56
3.24. Registro de la aplicación en la red TTN. 57
3.25. Registro de dispositivos en la red TTN. 57
3.26. Información de activación del dispositivo. 58
3.27. Intercambio de mensajes entre el nodo y el servidor de red. 58
3.28. Formato del payload. 59
3.29. Tipos de Virtual Machine (VM) de Google Cloud Platform (GCP) para diferentes cargas de trabajo. 60
3.30. Configuración estática IP de la instancia en GCP. 60
3.31. Flujo implementado en Node-RED 62
3.32. Panel de configuración bases de datos en Grafana. 65
3.33. Creación de una query desde el panel de configuración. 66

4.1. Consumo del dispositivo sin recurrir a modos de bajo potencia. 68
4.2. Modos bajo consumo disponibles en el microcontrolador atmega328P. . . . 69
4.3. Consumo base del dispositivo en ausencia de tareas. 69
4.4. Transición de modo bajo consumo a normal para ejecutar una tarea de medición. 70
4.5. Consumo del dispositivo durante la transmisión de datos. 70
4.6. Consumo del dispositivo durante la adquisición del sensor DS18B20. 71
4.7. Distribución de consumo promedio durante una hora. 72
4.8. Nodo sensor montado con la sonda de temperatura del suelo, DS18B20. . . 73
4.9. Nodo sensor montado con la sonda de temperatura y humedad del aire, AM2302. 73
4.10. Nodo sensor montado con la sonda de humedad del suelo. 74
4.11. Número de muestras recibidas por cada nodo. 74
4.12. Datos de la sonda de temperatura del suelo. 75
4.13. Sonda de temperatura en el entorno de prueba. 75
4.14. Datos de temperatura y punto de rocío en el entorno donde se han realizado las pruebas. 76
4.15. Datos de la sonda de temperatura y humedad del aire. 76

4.16. Datos de la sonda de humedad del suelo. 77
4.17. Sonda de humedad en el entorno de prueba. 77
4.18. Módulo Global Positioning System (GPS) conectado al nodo sensor. 78
4.19. Módulo GPS conectado al nodo sensor. 79

5.1. Zona de cultivo agrícola en invernaderos en El Ejido, Almería. 81
5.2. Nodo sensor de temperatura y humedad del aire. 81
5.3. Localización de los nodos en el entorno de producción. 82
5.4. Mapa de cobertura. 82
5.5. Dashboard de visualización de los sensores del invernadero 1. 83
5.6. Dashboard de visualización de los sensores del invernadero 2. 83
5.7. Desglose en tareas del tiempo dedicado a su realización. 86

A.1. Legend Top. 95
A.2. Copper Top. 95
A.3. Solder Mask Top. 95
A.4. Legend Botton. 96
A.5. Copper Botton. 96
A.6. Solder Mask Botton. 96
A.7. Drill. 97
A.8. Piezas impresas en 3D del escudo de radiacion. 98

B.1. Configuración firewall. 100

Índice de tablas

3.1. Características de la placa Arduino Mini Pro. 32
3.2. Bandas Industrial, Scientific and Medical (ISM) compatibles con el uso de LoRa. 33
3.3. Características del módulo RFM95W con el chip de radio SX1276. 33
3.4. Características del sensor AM2302. 39
3.5. Características del sensor DS18B20. 40
3.6. Características del sensor de humedad capacitivo v1.2. 42
3.7. Signal-to-noise ratio (SNR) necesaria para demodular un mensaje en función de su Spreading Factor (SF). 47
3.8. Planes de frecuencia utilizadas en The Things Network en la banda EU863-870. 47
3.9. Versiones de compilación. 51
3.10. Comandos de configuración implementados. 52
3.11. Características de la antena. 54
3.12. Ejemplo de datos almacenados en una "measurement". 61

4.1. Tareas y consumo asociado 71
4.2. Características del Módulo GPS ATK-NEO-6M. 78

5.1. Bill of Materials (BOM) de la PCB del nodo sensor. 84
5.2. Componentes y coste del nodo de temperatura del suelo. 85
5.3. Componentes y coste del nodo de temperatura y humedad del aire. 85
5.4. Componentes y coste del gateway. 85
5.5. Coste material de la instalación realizada. 86

Acrónimos

3GPP 3rd Generation Partnership Project. 18

ABP Activation By Personalisation. 44

ABS Acrylonitrile Butadiene Styrene. 98

ADC Analog-to-digital converter. 42

ADR Adaptative Data Rate. 48, 79

API Application Programming Interface. 21, 62

AWS Amazon Web Services. 59

BOM Bill of Materials. 7, 84

BPSK Binary Phase-shift Keying. 19

Chirps Compressed High Intensity Radar Pulse. 45

CLI Command Line Interface. 52

CR coding rate. 47

CSS Chirp Spread Spectrum. 18

DIY Do It Yourself. 24

EDA Electronic Design Automation. 36

GCP Google Cloud Platform. 5, 59, 60

GPS Global Positioning System. 6, 78, 79, 82

GSM Global System for Mobile Communications. 18

HAL Hardware Abstraction Layer. 50

IoT Internet de las Cosas. 12–14, 16–19, 22, 24, 27, 62, 88

IPSO Internet Protocol for Smart Objects. 52

ISM Industrial, Scientific and Medical. 7, 18, 19, 32, 33

JSON JavaScript Object Notation. 26, 62–64

LED Light-emitting diode. 36

LMIC LoRaWAN MAC in C. 50, 51, 69

LPP Low Power Payload. 52, 59

LPWAN Low Power Wide Area Network. 17–19

LTE Long-Term Evolution. 18

LTS Long Term Support. 60

MAC Media Access Control. 27, 44, 49, 50

MQTT Message Queing Telemetry Transport.. 26, 62, 63, 88

NB-IoT Narrow Band IoT. 1, 4, 18–21

OTA Over-The-Air. 23

OTAA Over-The-Air Activation. 44, 45, 50, 52, 57, 58

PCB Printed Circuit Board. 4, 7, 33, 36–38, 42, 84

RSSI Received signal strength indication. 25, 61, 63, 79, 82

SF Spreading Factor. 7, 45–48, 79

SNR Signal-to-noise ratio. 7, 25, 46–48, 58, 61, 63, 79

SPI Serial Peripheral Interface. 34, 35

SQL Structured Query Language. 61

TDoA Time Difference of Arrival. 19

TLS Transport Layer Security. 26

TSDB Time Series Database. 27

TTN The Things Network. 2, 5, 25, 56, 57, 59, 88

VM Virtual Machine. 5, 59–61

VPC Virtual Private Cloud. 60

WDT Watchdog Timer. 68–70

Capítulo 1

INTRODUCCIÓN

El IoT (Internet de las Cosas) es un concepto que se basa en la inteconexión digital entre dispositivos, personas y la propia Internet permitiendo el intercambio de datos entre ellos. El objetivo es hacer que todos estos dispositivos se comuniquen entre sí y nos ofrezcan datos y detalles sobre su funcionamiento. Gracias a esta interconexión y conocimiento basado en datos, se busca que los objetos interactúen con elementos de la vida real y ganen independencia de los seres humanos.

Actualmente, con el crecimiento explosivo de las tecnologías de IoT, se puede encontrar un número cada vez mayor de aplicaciones en campos como la seguridad, la agricultura, el seguimiento de activos, las ciudades o los hogares inteligentes.

1.1. IoT en la agricultura

En la actual revolución tecnológica de la agricultura, el IoT juega un papel fundamental. Los dispositivos conectados permiten a los agricultores conocer el estado del medio, de la salud del cultivo, de la maquinaria en uso, de las condiciones de almacenamiento o del nivel de consumo de energía entre otros. La posibilidad de recoger, almacenar y procesar en tiempo real todos estos datos de tan distinta procedencia y cruzarlos automáticamente con mucha otra información relacionada con los precios de las materias primas, la evolución del mercado o las previsiones meteorológicas, convierten el IoT en algo imprescindible en el futuro de este sector.

Así, cada día es menos necesaria la intervención humana en los procesos agrícolas. Las máquinas y tecnologías que intervienen en los mismos son más autónomos que nunca y capaces de tomar decisiones precisas en cuestión de segundos. Esto se puede ver reflejado directamente en:

- **Mayor productividad y rentabilidad**. La monitorización y análisis en tiempo real permite la toma de decisiones basada en evidencias, logrando mejorar la producción al seleccionar mejor el tipo de cultivo, fechas y terreno de plantación o la cantidad exacta de agua y fertilizante que precisará. Cuanto mayor es el nivel de precisión de las previsiones, mayor probabilidad de sacar el máximo rendimiento a la explotación.

- **Medioambiente.** La posibilidad de monitorizar con precisión el estado del suelo, permite tanto la administración eficiente del agua como la detección de posibles fugas en las instalaciones, reduciendo drásticamente el desperdicio de recursos hídricos. De igual manera, la reducción en el uso de pesticidas aplicando solo las dosis necesarias, tiene un impacto positivo sobre las áreas de cultivo y la naturaleza.

- **Mejor control de catástrofes y plagas.** Mediante el uso de sensores y cámaras conectadas a Internet es posible realizar un mejor seguimiento y control de las plagas. Al detectar una población demasiado alta, se activa automáticamente un sistema de liberación de feromonas que altera los patrones de apareamiento de las plagas.

- **Mayor seguridad en los procesos.** El uso del IoT puede ir más allá de los procesos relacionados con el cultivo, también pueden ser aplicados en el área de la salud, conservación de los alimentos en el almacenamiento, transporte, etc... Actualmente se están utilizando diversos sensores y herramientas de seguimiento para la monitorización y control de factores como el tiempo de envío o la temperatura de almacenamiento, etc. que permiten garantizar la seguridad y la calidad alimentaria en todas los fases del proceso agrícola.

1.2. Problemática detectada y propuesta inicial

Ante la necesidad de implantar un sistema IoT para la monitorización de parámetros ambientales en un explotación agrícola, el primer paso lógico cosiste en acudir a soluciones comerciales ¿Para qué reinventar la rueda si no vamos a mejorarla?. Tras analizar diferentes soluciones ya existentes en el mercado se ha detectado cierta problemática común:

- **Reducida interoperabilidad** de los dispositivos. La mayoría de las empresas ofrecen sistemas completos y cerrados lo que imposibilita la conexión de otro tipo de dispositivos a sus plataformas o bien usar sus dispositivos finales (sensores o actuadores) en plataformas de terceros. En un paradigma donde la interoperabilidad y comunicación entre diferentes tipos de servicios es vital, encontrarse con sistemas cerrados no parece garantizar el éxito.

- **Matenibilidad futura**. Es un mercado en auge, con muchas empresas trabajando en esta dirección pero no hay dominantes en el mercado. El confiar un proyecto de tal envergadura a una empresa no asentada en el mercado, genera dudas en cuanto a la sostenibilidad a largo plazo ¿Qué ocurrirá si la empresa no tiene éxito?

- **Elevando coste** del sistema y mantenimiento que no justifica el servicio aportado. Muchas empresas ofrecen sus soluciones como un sistema a medida, justificando de esta forma un coste elevado.

Más allá de los sistemas comerciales, existen plataformas de hardware y software abierto mediante los que podemos montar nuestro propio sistema IoT. Con una simple búsqueda en Internet "IoT Arduino" encontraremos cientos de páginas en las que se nos indicará paso a paso cómo montar nuestros sistemas IoT. Aunque funcionales, estas soluciones no están optimizas ni son robustas para su uso extensivo en un entorno de producción.

Llegados a este punto, surge la idea de este TFM: crear una plataforma IoT basada en tecnologías abiertas que sea escalable, robusta y fácil de instalar en un entorno de producción. Además del diseño y desarrollo de cada uno de las partes que formarán el sistema IoT, se realizará un despliegue en una explotación agrícola, que permitirá validar en un entorno real el funcionamiento del sistema. Para ello se contará con la colaboración de la empresa Jiherflora S.L que ha ofrecido sus instalaciones para la realización de las pruebas.

El objetivo de que todos los elementos desarrollados sean publicados abiertamente, permitirá a otros desarrolladores utilizar este trabajo y sobre él aportar nuevas soluciones que cubran nuevos casos de uso.

1.3. Estructura del trabajo

A continuación, se exponen de forma resumida los contenidos que aparecen en los diferentes capítulos de este trabajo.

- En el **Capítulo 2** se realizará una revisión bibliográfica y tecnológica de los principales elementos involucrados en el sistema IoT que se va desarrollar. Esta información será la base para decidir qué elementos deben formar el sistema.
- En el **Capítulo 3** se expondrá en detalle el proceso de diseño e integración de todos los componentes del sistema.
- En el **Capítulo 4** se analizarán los resultados de funcionamiento del sistema, en condiciones controladas, con el objetivo de validar el diseño y mejorar aquellos aspecto que sean de interés.
- En el **Capítulo 5** se mostrarán los resultados del sistema una vez ha sido desplegado en el entorno de producción real. También se realizará un resumen del coste del económico del sistema.
- En el **Capítulo 6** se tratarán las conclusiones y posibles líneas futuras de investigación derivadas de este proyecto.

Capítulo 2

ESTADO DEL ARTE

La convergencia de tecnologías de comunicación inalámbricas, sistemas microelectrónicos y sensores, microservicios e Internet ha propiciado el desarrollo de los sistemas IoT. Para tener una visión global que nos permita enfocar el desarrollo de este TFM es necesario, en primer lugar, un conocimiento de las partes involucradas. El objetivo de este capítulo es realizar una revisión del estado tecnológico de los principales elementos involucrados en un sistema IoT.

2.1. Sistema IoT

Dentro del paradigma del Internet de las cosas, nos encontramos una gran variedad de tecnologías involucradas: sensores, microcontroladores, sistemas de captación y almacenamiento de energía, redes inalámbricas, protocolos de red, sistema de gestión de datos, machine learning, big data, etc. Bajo una visión global, un sistema IoT debe contener al menos los siguientes elementos:

- **Nodos finales**. Son los dispositivos encargados de interactuar con el entorno, ya sea mediante la recolección de información (sensores) o realizando acciones (actuadores).
- **Conectividad**. El hardware necesita una forma de transmitir toda esa información a los servidores de red, así como también un camino para recibir órdenes desde estos. En algunos sistemas IoT pueden existir también elementos intermedios entre los dispositivos y la plataforma de gestión, como gateways o router que se encargan de encaminar los paquetes de datos correctamente.
- **Plataforma de gestión**. Por lo general el software de un sistema IoT está en la nube y entre sus funciones está la de garantizar la correcta gestión de los dispositivos, asegurar la comunicación y ofrecer la interoperabilidad con otros servicios en red, por ejemplo para almacenar y visualizar los datos.
- **Interfaz de usuario**. A fin de que todo lo anterior tenga una utilidad real debe existir un método para que los usuarios interactúen con el sistema IoT, sean capaces de monitorearlo y tomar decisiones a partir de la información que ahí consiguen.

La creación un sistema completo IoT requiere conocer cuáles son las tecnologías existentes, saber integrarlas y poder desarrollar nuevas soluciones técnicas cuando éstas no existan.

2.2. Conectividad inalámbrica

Uno de los elementos clave de los sistemas IoT es la comunicación inalámbrica que permite la conexión e intercambio de datos entre dispositivos. En esta sección se hará un análisis del estado tecnológico de las redes de comunicación empleadas en dispositivos IoT.

Las aplicaciones de IoT tienen requisitos específicos, tales como largo alcance, baja transmisión de datos, bajo consumo de energía y bajo coste. Las tecnologías de radio de corto alcance ampliamente utilizadas, como ejemplo ZigBee o Bluetooth no están adaptadas a los escenarios que requieren una transmisión de largo alcance. Por otro lado, las soluciones basadas en las comunicaciones celulares: 2G, 3G y 4G proporcionan una mayor cobertura pero el consumo de energía es demasiado alto y tienen un coste operativo no despreciable.

Los requisitos de comunicación de las aplicaciones de IoT han impulsado el surgimiento de la tecnología LPWAN (Low Power Wide Area Network). La características que encontramos en este tipo de redes son:

- **Largo alcance**: Permite comunicaciones de 10 a 40 km en zonas rurales y de 1 a 5 km en zonas urbanas [1]. También puede permitir una comunicación de datos eficaz en lugares interiores y subterráneos.
- **Bajo consumo**: Optimizados para el consumo de energía, los transceptores LPWAN pueden funcionar con baterías pequeñas y económicas hasta 10 años [2].
- **Bajo coste**: Los protocolos simplificados y ligeros de LPWAN reducen la complejidad en el diseño del hardware y disminuyen los costes del dispositivo, encontrando transceptores de radio con un precio inferior a 2€ y un coste de mantenimiento de 1€ por dispositivo al año [3]. Su largo alcance combinado con una topología en estrella reduce el coste de la infraestructura. Por otro lado, el uso de bandas sin licencia reducen considerablemente los costes por el uso de la red.

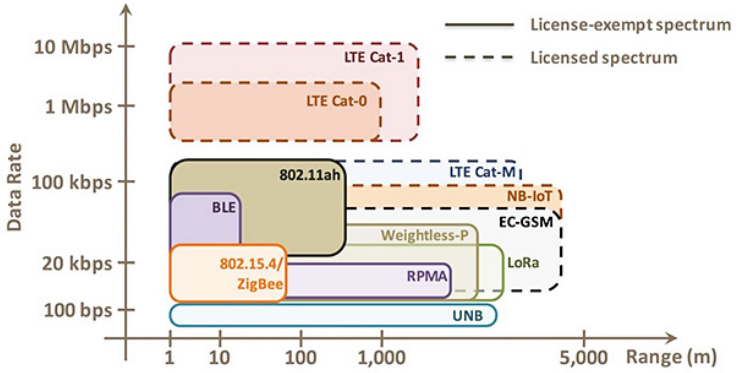


Figura 2.1: Velocidad de transmisión vs. alcance de diferentes tecnologías de radiocomunicación IoT.

Fuente Imagen: www.iotnet.eu

En resumen, la tecnología LPWAN es muy adecuada para aplicaciones de IoT que sólo necesitan transmitir pequeñas cantidades de datos a larga distancia. Para cubrir esta necesidad, han aparecido diferentes soluciones LPWAN tanto en el ancho de banda de frecuencias con licencia como sin licencia. Entre ellas, NB-IoT, LoRa y Sigfox son las principales tecnologías emergentes en este campo. En la Figura 2.1, se muestra una comparativa de la velocidad de transmisión y en alcance para diferentes tecnologías de comunicación IoT.

2.2.1. NB-IoT

NB-IoT es la iniciativa del 3GPP (3rd Generation Partnership Project), la organización que está detrás de la estandarización de los sistemas de tecnología móvil, para abordar las necesidades de conectividad de los dispositivos IoT. NB-IoT puede coexistir con GSM (Global System for Mobile Communications) y LTE (Long-Term Evolution) en las bandas de frecuencia autorizadas (por ejemplo, 700 MHz, 800 MHz y 900 MHz). NB-IoT ocupa un ancho de banda de frecuencia de 200 KHz, lo que corresponde a un bloque de recursos en la transmisión GSM y LTE [4].

Permite la conectividad de hasta 100.000 dispositivos finales por celda, para lo que emplea el acceso múltiple por división de frecuencia de una sola portadora (FDMA) en el enlace ascendente y el FDMA ortogonal (OFDMA) en el enlace descendente [4]. La velocidad de datos se limita a 200 kbps para el enlace descendente y a 20 kbps para el ascendente. El tamaño máximo de la carga útil para cada mensaje es de 1600 bytes. De acuerdo con el plan actual del 3GPP, la especificación NB-IoT se ampliará para incluir métodos de localización y servicios multi-cast que permitirían, por ejemplo, la actualización de software de los dispositivos finales.

El protocolo de comunicación de NB-IoT se basa en el protocolo de red LTE. De hecho, NB-IoT reduce las funcionalidades del protocolo LTE al mínimo para adaptarlo principalmente a los requisitos de bajo consumo. Es importante destacar que no es un protocolo de comunicación basado en IP como LTE-M (otra tecnología móvil LPWA asociada a las aplicaciones de IoT).

Actualmente se está probando y experimentando sólo en Europa, y no está ampliamente disponible.

2.2.2. LoRa/LoRaWAN

LoRa es el nombre de la tecnología inalámbrica basada en el tipo de modulación digital denominada CSS (Chirp Spread Spectrum), patentada por el fabricante Semtech, siendo el único titular para fabricar chips con licencia para LoRa.

LoRa utiliza las bandas ISM sin licencia, es decir, 868 MHz en Europa, 915 MHz en América del Norte y 433 MHz en Asia. Gracias al empleo de la técnica de comunicación de

espectro espandido (CSS) la señal resultante tiene bajos niveles de ruido, lo que permite una gran resistencia a las interferencias. LoRa utiliza diferentes factores de propagación, buscando un compromiso ente la tasa de datos y el alcance. Un mayor factor de dispersión permite un mayor alcance a expensas de una menor velocidad de datos y viceversa. La tasa de datos LoRa está entre 300 bps y 50 kbps dependiendo del factor de dispersión y el ancho de banda del canal. Los mensajes transmitidos utilizando diferentes factores de dispersión pueden ser recibidos simultáneamente por las estaciones base LoRa [5]. La longitud máxima de la carga útil de cada mensaje, tanto en enlace de subida como bajada, es de 243 bytes. Al utilizar la banda ISM en Europa esto implica un ciclo de servicio máximo del 1 % , que limita el volumen y la frecuencia del tráfico, así como la capacidad de la estación base para controlar la red y enviar mensajes hacia los dispositivos.

Por su parte, LoRaWAN es el protocolo de red de capa de enlace, que usa la tecnología de radio LoRa para comunicar y administrar dispositivos que utilizan LoRa. Cada mensaje transmitido por un dispositivo final es recibido por todas las estaciones base en el rango de alcance. Al explotar esta recepción redundante, la capa de red LoRaWAN mejora la proporción de mensajes recibidos con éxito. Las recepciones duplicadas resultantes se filtran en el servidor de red que también se encarga de validar la autenticación, enviar acuses de recibo al dispositivo final y enviar el mensaje al servidor de aplicación correspondiente. La recepción múltiple del mismo mensaje permite además localizar los dispositivos finales a partir de las diferencias de tiempo de llegada TDoA (Time Difference of Arrival), una funcionalidad que hace interesante el uso de LoRaWAN para la geolocalización de objetivos.

A diferencia de NB-IoT o Sigfox, para conectarse a una red LoRaWAn en la mayoría de los casos será necesario desplegar su propio gateway de red. Iniciativas como The Things Network pretenden desplegar una red global de gateways de forma colaborativa, creando una red pública ciudadana que puede ser usada de forma gratuita y abierta. También podemos encontrar operadores que ofrecen como servicio este tipo de red LPWAN, como por ejemplo Redexia.

2.2.3. Sigfox

Sigfox es un operador de red LPWAN que ofrece una solución end-to-end de conectividad IoT basada en su propia tecnología. Como servicio, ofrece una red global de estaciones bases capaces de recibir información de dispositivos así como el servicio web y back-end para la gestión de los dispositivos y su integración con otras aplicaciones.

Los dispositivos finales se conectan a estas estaciones base utilizando modulación BPSK (Binary Phase-shift Keying) en un portador de banda ultraestrecha (Ultra Narrow Band). Sigfox utiliza bandas ISM sin licencia (868 MHz en Europa, 915 MHz en América del Norte y 433 MHz en Asia). Al emplear la banda ultraestrecha, Sigfox utiliza el ancho de

banda de frecuencias de manera eficiente y experimenta niveles de ruido muy bajos, lo que conduce a un consumo de energía también muy bajo, una alta sensibilidad del receptor y un diseño de antena de bajo coste a expensas de una velocidad máxima de transmisión de 100 o 600 bps.

Permite una comunicación bidireccional con una significativa asimetría de enlace. Así, el número de mensajes en el enlace ascendente se limita a 140 mensajes por día y una longitud máxima del payload de 12 bytes. Sin embargo, el número de mensajes de enlace descendente está limitado a 4 por día, lo que significa que no se admite, a nivel de aplicación, el acuse de recibo de cada mensaje de enlace ascendente. La comunicación de enlace descendente sólo pueden ocurrir después de una comunicación de enlace ascendente y la carga útil máxima en este tipo de mensajes es de 8 bytes.

Para garantizar la fiabilidad de la comunicación del enlace ascendente, cada mensaje del dispositivo terminal se transmite varias veces (tres por defecto) por canales de frecuencia diferentes. Con este fin, en Europa, por ejemplo, la banda entre 868,180 MHz y 868,220 MHz se divide en 400 canales ortogonales de 100 Hz (entre ellos, 40 canales están reservados y no se utilizan) [3]. Como las estaciones base pueden recibir mensajes simultáneamente en todos los canales, el dispositivo final puede elegir aleatoriamente un canal de frecuencia para transmitir sus mensajes. Esto simplifica el diseño del dispositivo final y reduce su coste.

2.2.4. ¿Por qué LoRa/LoRaWAN ?

De las tres redes LPWAN presentadas en las secciones previas, LoRaWAN es la que mejor se adapta a los requisitos del sistema planteado.

	Sigfox	LoRaWAN	NB-IoT
Modulation	BPSK	CSS	QPSK
Frequency	Unlicensed ISM bands (868 MHz in Europe, 915 MHz in North America, and 433 MHz in Asia)	Unlicensed ISM bands (868 MHz in Europe, 915 MHz in North America, and 433 MHz in Asia)	Licensed LTE frequency bands
Bandwidth	100 Hz	250 kHz and 125 kHz	200 kHz
Maximum data rate	100 bps	50 kbps	200 kbps
Bidirectional	Limited / Half-duplex	Yes / Half-duplex	Yes / Half-duplex
Maximum messages/day	140 (UL), 4 (DL)	Unlimited	Unlimited
Maximum payload length	12 bytes (UL), 8 bytes (DL)	243 bytes	1600 bytes
Range	10 km (urban), 40 km (rural)	5 km (urban), 20 km (rural)	1 km (urban), 10 km (rural)
Interference immunity	Very high	Very high	Low
Authentication & encryption	Not supported	Yes (AES 128b)	Yes (LTE encryption)
Adaptive data rate	No	Yes	No
Handover	End-devices do not join a single base station	End-devices do not join a single base station	End-devices join a single base station
Localization	Yes (RSSI)	Yes (TDOA)	No (under specification)
Allow private network	No	Yes	No
Standardization	Sigfox company is collaborating with ETSI on the standardization of Sigfox-based network	LoRa-Alliance	3GPP

Figura 2.2: Comparativa de las tres soluciones LPWA: Sigfox, LoRaWAN y NB-IoT.

Fuente Imagen: [6]

Tanto el ancho de banda, como el rango de alcance y el máximo payload permitido, es compatible con la aplicación que vamos a desarrollar. Se reduce el coste de mantenimiento, ya que no es necesario el pago mensual por el uso de la red, algo que si ocurre con Sigfox o NB-IoT.

Pero sin duda, el aspecto más importante para la elección de esta tecnología es la posibilidad de configuración de su propio gateway y crear una red completamente separada y segura. De las tres tecnologías discutidas en esta sección, es la única que puede ser desplegada de forma autónoma, es decir, podemos dar cobertura a una zona por nuestros medios únicamente colocando una puerta de enlace (gateway). Esto es una gran ventaja ya que no dependemos de terceras empresas, ahorrando costes y asegurando su vida útil.

2.3. Estructura de red LoRaWAN

En una red LoRaWAN, los dispositivos finales, que disponen de uno o varios sensores o actuadores conectados a ellos, son los denominados nodos. Estos se comunican con los gateways, encargados de transmitir toda la información que reciben a un servidor de red. El servidor de aplicaciones recibe los datos del servidor de red y por medio de una interfaz de aplicación o API (Application Programming Interface) puede servir dar el control de los dispositivos al usuario. En la Figura 2.3 se muestra la estructura básica de una red LoRaWAN.

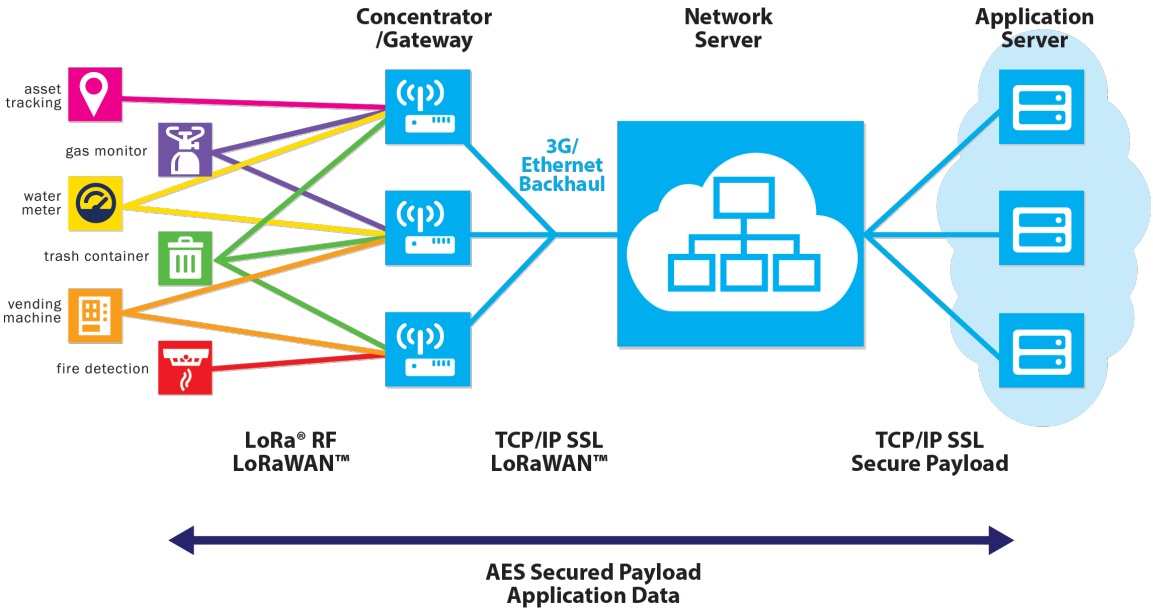


Figura 2.3: Estructura de una red LoRaWAN.

Fuente Imagen: https://medium.com/

2.3.1. Nodos finales

Los nodos finales son los elementos encargados de realizar las mediciones y transmitir la información y/o actuar sobre el medio en función de la información recibida.

Podemos encontrar dos tipos de soluciones a la hora de buscar nodos sensores con tecnología LoRaWAN:

- **Soluciones comerciales cerradas.** Los fabricantes de estos productos ofrecen una solución comercial destinada a emplearse junto con su propia plataforma. Al tratarse de soluciones cerradas solo podrán emplearse sensores ofrecidos por el fabricante y difícilmente sus dispositivos serán interoperables con otras plataformas. Al coste elevado de adquisición, habrá que añadir un coste de mantenimiento por el uso de las plataformas de datos.

- **Soluciones Open Source**. Son muy flexibles en cuanto a la capacidad para integrar sensores y conectar a diferentes redes y plataformas. El uso de estos elementos es recomendable para la etapa de prototipo, pero su coste y fiabilidad no los hacen adecuados para usar en entornos de producción.

En este contexto surge la necesidad de diseñar un nodo sensor que tenga lo mejor de ambos mundos: flexibilidad para la integración de diferentes tipos de sensores, diseño robusto para usar en un ambiente de producción y capacidad de integración en cualquier plataforma IoT que soporte los estándares del mercado. La flexibilidad para conectar diferentes tipos de sensores es algo que ya ofrecen los kits de desarrollo que podemos encontrar en el mercado, en su mayorías basados en la plataformas Arduino.

La empresa Dragino comercializa un shield, para acoplarlo directamente sobre la placa de desarrollo Arduino UNO. Este shield integra el transceptor de radio LoRa y tiene un coste aproximado de 25€, a lo que se sumaría el coste de la placa Arduino. Al tratarse de un kit de desarrollo, el conjunto carece de conexiones robustas para los sensores o actuadores a controlar.

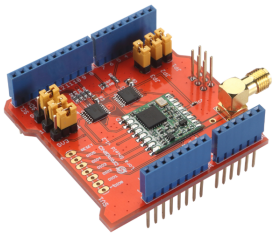


Figura 2.4: Shield Dragino LoRa

Fuente Imagen: https://tienda.bricogeek.com/

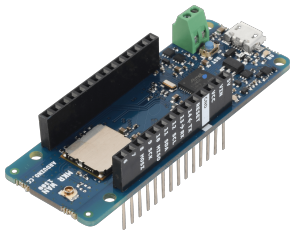


Figura 2.5: Arduino MKR WAN 1300.

Otra solución que podemos encontrar es la placa Arduino MKR WAN 1300, que integra conectividad LoRa junto con un microcontrolador de Atmel SAMD21. Tiene un coste de 40€ y al igual que con el shield de Dragino, nos encontramos que al tratarse de un kit genérico de desarrollo no tiene conexiones robustas para sensores.

Fuente Imagen: https://www.geekfactory.mx/

Otro ejemplo es el Waspmote de la empresa Libellium. Se trata de una plataforma altamente configurable tanto a nivel de conectividad como sensores. Dada su versatilidad es muy usada como plataforma de prototipado en investigación. La contrapartida de dicha versatilidad es su coste, superior a 200€.



Figura 2.6: WaspMote Libellium.

Fuente Imagen: https://www.libelium.com/

2.3.2. Gateways

Dentro de la arquitectura de red LoRaWAN, el gateway o puerta de enlace es el dispositivo hardware que recoge todos los mensajes LoRaWAN de los nodos finales y reenvían los paquetes de datos a un servidor de red centralizado (enlace ascendente) o enviar mensajes desde el servidor a los nodos finales (enlace descendente).

Los gateways y los nodos forman una red en estrella, ya que un gateway puede dar servicio a centenares de nodos finales . A su vez, un servidor de red central puede gestionar la conexión de uno o más gateways, formando una nueva red en estrella. Los gateways normalmente se conectan al servidor de red mediante un enlace cifrado sobre TCP/IP y pueden gestionar tanto mensajes de enlace ascendente como descendente hacia el nodo final.

Debido al diseño del protocolo, los datos de un nodo final LoRa pueden ser recibidos por múltiples puertas de enlace siempre que estén en su zona de alcance, por lo que será el servidor de red el encargado de gestionar la duplicidad de mensajes. El standard LoRaWAN también soporta mensajes de tipo broadcast [7], permitiendo, por ejemplo, la actualización de software OTA (Over-The-Air).

Un ejemplo de gateway comercial es el RAK7249. Diseñado para uso en exteriores, es una de las opciones LoRaWAN más configurable, manejable y escalable para aplicaciones industriales del IoT. Dispone de varias opciones de conectividad de red incluyendo 4G-LTE, 3G, 2G y Ethernet. Su grado de protección industrial es IP67/NEMA-6. El precio de este gateway ronda los 700€.



Figura 2.7: Gateway outdoor RAK7249.

Fuente Imagen: https://www.rakwireless.com



Figura 2.8: Gateway indoor LPS8.

También podemos encontrar modelos diseñados para uso en interiores a un coste más reducido. Por ejemplo el modelo LPS8 de la marca Dragino disponen de conectividad WiFi y Ethernet. Integra el concentrador SX1308 LoRa que proporciona 10 canales de demodulación paralelos lo que le permite manejar hasta 1000 nodos finales. Es totalmente compatible con LoRaWAN y su precio ronda los 150€.

Fuente Imagen: https://www.antratek.com

También se pueden construir gateways LoRaWAN DIY (Do It Yourself) utilizando, por ejemplo, una Raspberry Pi o cualquier otro SBC y una placa como la ic880a de la compañía alemana IMST. Otra solución maker que se está utilizando mucho últimamente es el RAK 831 + Raspberry Pi.

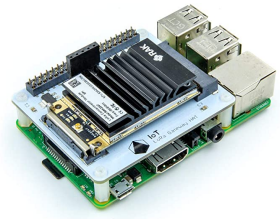


Figura 2.9: Raspberry PI - RAK831.

Fuente Imagen: https://www.cytron.io

2.3.3. Servidores de red

El servidor de red LoRaWAN es la parte central de la red LoRaWAN encontrándose en un punto intermedio entre los gateways y el servidor de aplicaciones. Sus principales objetivos son garantizar la seguridad, escalabilidad y fiabilidad del enrutamiento de datos a través de la red y servirlos a la capa de aplicación. Es el principal responsable de:

- Eliminar los mensajes duplicados cuando se reciben por múltiples puertas de enlace.
- Seleccionar la mejor puerta de enlace para los mensaje de enlace descendente, normalmente en función de la calidad de enlace, calculada a partir del RSSI (Received signal strength indication) y la SNR de los paquetes recibidos previamente.
- Enrutar los mensajes enviados por los gateways a la aplicación donde están registrados los dispositivos.

Existen diferentes plataformas y servicios que implementan el servidor de red LoRaWAN, tanto de pago por uso como servidores públicos gratuitos y de código abierto que pueden ser desplegados de forma autónoma para uso privado.

TTN es una iniciativa que tiene como objetivo la creación de una red abierta LoRaWAN con cobertura internacional. Los miembros de esta comunidad aportan su propia infraestructura (gateways) aumentando así la cobertura de la red que puede ser utilizada de forma abierta por cualquier persona. El servidor de red es gratuito, siempre y cuando permita a todos los nodos finales cercanos utilizar sus gateways para comunicarse con aplicaciones de terceros.



Es un servicio popular, especialmente para el desarrollo de prototipos o durante las pruebas iniciales, antes de un despliegue profesional. The Things Network no se recomienda para el despliegue comercial ya que no hay garantía de disponibilidad de los diferentes nodos de la red. The Things Industries es la oferta comercial de TTN. Este es un entorno privado en el que sus elementos de red no son compartidos con el resto de la comunidad de The Things Network.



Otro servidor de red popular es ChirpStack, antes conocido como LoRaServer. Es un software de código abierto desarrollado por Broccar. Es una solución económica, pero que requiere un cierto nivel de experiencia. La configuración no es simple, y requiere el mantenimiento y la gestión de la red, por lo que es un buena opción si se requiere el control absoluto de la red.

2.3.4. Servidores de aplicación

El servidor de aplicaciones maneja la capa de aplicación LoRaWAN, esto incluye el descifrado y la decodificación de datos enlace ascendente, también la gestión de los mensajes de enlace descendente así como su codificación y cifrado. Para ello, los servidores de aplicación se enlazan con los servidores de red para recibir el tráfico ascendente y enviar el tráfico descendente.

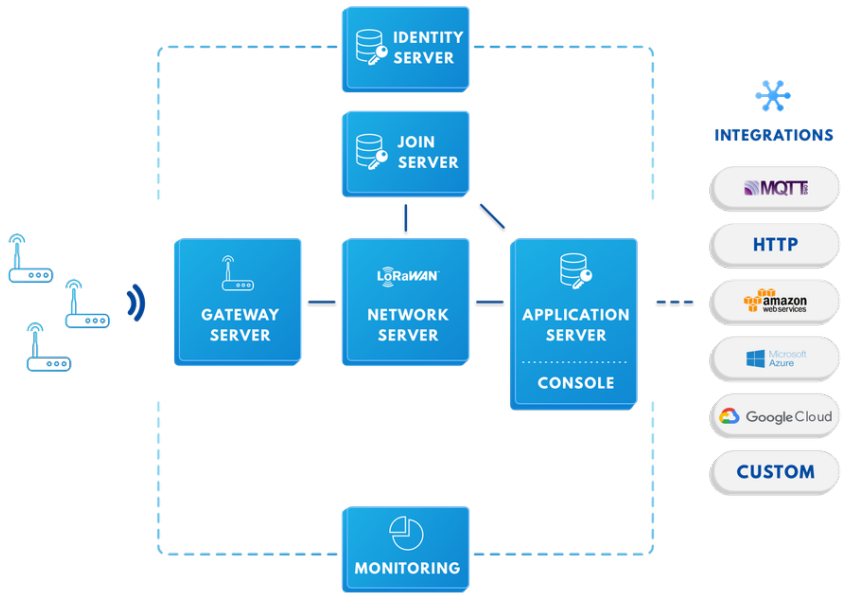


Figura 2.10: Arquitectura de microservicios del stack LoRaWAN implementado en The Things Network.

Fuente Imagen: https://www.thethingsnetwork.org

El servidor de aplicaciones puede decodificar y codificar la carga útil binaria enviada y recibida por los dispositivos finales. Esto permite trabajar con datos estructurados de streaming, como los objetos JSON que utilizan los webhooks MQTT y HTTP, y al mismo tiempo mantener un protocolo ligero en binario en la comunicación con los dispositivos finales.

Las aplicaciones pueden conectarse al servidor a través de múltiples protocolos y mecanismos, por ejemplo, intercambiando mensajes JSON (JavaScript Object Notation) sobre MQTT (Message Queing Telemetry Transport.) o webhooks. El propio servidor de aplicaciones alberga un servidor MQTT a través de TLS (Transport Layer Security), proporcionando confidencialidad de los mensajes intercambiados entre las aplicaciones y el servidor de aplicaciones.

2.4. Explotación de datos

En el nivel más alto de un sistema IoT nos encontramos las aplicaciones. Estas están destinadas a la explotación de los datos, bien sea mediante su almacenamiento, analítica y/o visualización.

Centraremos esta sección en el análisis de bases de datos y herramientas de visualización web especializadas en el tratamiento de series temporales.

2.4.1. Base de datos

Una base de datos de series temporales, TSDB (Time Series Database), es un sistema de software optimizado para almacenar y servir series temporales a través de pares asociados de tiempo y valor(es) [8].

Aunque es posible almacenar datos de series temporales en bases de datos genéricas, disponer de bases de datos específicas que utilizan la variable tiempo como índice clave permite una gestión más eficiente de los datos.

Una TSDB suele separar el conjunto de características fijas, denominadas etiquetas, del de sus valores dinámicos. Un ejemplo es el almacenamiento de la temperatura monitorizada por un sensor: las características fijas incluirían, por ejemplo, el nombre del dispositivo, el identificador, la MAC (Media Access Control), etc. y los valores dinámicos almacenarían el valor de temperatura y la marca de tiempo de cada instante. La separación tiene por objeto almacenar e indexar eficientemente los datos y mejorar la extracción para su posterior tratamiento.

Las TSDB varían considerablemente en sus características, pero la mayoría implementan funcionalidades para crear, leer, actualizar y eliminar los pares de valores de tiempo, así como las etiquetas asociadas. Además, en este tipo de base de datos es común encontrar funcionalidades para el cálculo de datos agregados, filtrado y análisis estadístico [9].

InfluxDB, Timescale, MongoDB, Prometheus, Druid, etc. son algunas de estas bases de datos que están optimizadas para trabajar con series temporales.

2.4.2. Visualización de datos

Los datos obtenidos de las series temporales proporcionan un valor significativo porque permiten analizar importantes métricas en tiempo real e históricas. La visualización de estas series temporales es la forma más directa para un análisis y comprensión de los datos, permitiendo la toma de decisiones basada en evidencias. Dependiendo de la naturaleza de los datos, estos pueden ser consultados y representados gráficamente usando diferentes paneles de visualización: gráficos de evolución temporal, histogramas, tablas, etc.

La visualización de series temporales se realiza de forma común mediante un gráfico x-y clásico, donde el eje horizontal se utiliza para trazar los incrementos de tiempo, mientras

que el eje vertical señala los valores de la variable que se está midiendo. Este tipo de gráficos permite a los usuarios finales identificar rápidamente una tendencia, detectar una anomalía o analizar cómo está cambiando una métrica a lo largo del tiempo.

El panel de control o dashboard en inglés, es la herramienta empleada para visualizar y presentar los datos de las series temporales en un formato significativo y fácil de entender.



Figura 2.11: Ejemplo de paneles de visualización implementados en Grafana.

Fuente Imagen: https://www.grafana.com

Existen herramientas como Chronograf o proyectos de código abierto como Grafana o Kibana que permiten, desde una interfaz web, la creación sencilla de paneles de control. Estas soluciones a menudo vienen con dashboards pre-construidos y permiten construir los gráficos a partir de múltiples fuentes de datos, incluidas bases de datos de series temporales como las que hemos enunciado en el apartado anterior.

Otras formas más personalizables para la construcción de paneles de visualización pasan por el uso de librerías de gráficos como Plotly, permitiendo integrar sus módulos en programas desarrollados en python o javascripts.

Una ventaja de emplear herramientas de visualización como Chronograf o Grafana es que permite la gestión de roles, por lo que el acceso o edición de los paneles de control estará limitado a usuarios y/o grupos.

2.5. Control de versiones Git

El desarrollo de software y hardware bajo herramientas de control de versiones permite gestionar ágilmente los cambios en el código fuente de los programas y poder revertirlos, así como facilitar el dasarrollo en paralelo cuando un equipo trabaja en el mismo proyecto. Existen infinidad de programas con esta finalidad siendo Git, Mercuarial o Subversion los más empleados.



En este proyecto se ha optado por el desarrollo bajo el sistema Git, cuyas principales características son:

- Mecanismo de almacenamiento de los elementos que deba gestionar (ej. archivos de texto, imágenes, documentación, etc.)
- Posibilidad de realizar cambios sobre los elementos almacenados (ej. modificaciones parciales, añadir, borrar, renombrar o mover elementos).
- Registro histórico de las acciones realizadas con cada elemento o conjunto de elementos, normalmente pudiendo volver o extraer un estado anterior.

El repositorio Git con el software desarrollado en este proyecto está alojado en la plataforma GitHub y puede ser consultado a través del siguiente enlace: https://github.com/AngelJMC/agriot-sensor-node

Capítulo 3

DISEÑO Y DESARROLLO

3.1. Nodos - elementos sensores

Como ya ha sido descrito en el Capítulo 2, los nodos finales son los elementos encargados de interactuar con el mundo físico. En esta sección se tratará el diseño electrónico del nodo sensor, así como la descripción técnica de los diferentes elementos que lo forman.

Los requisitos de diseño de partida, aun siendo genéricos, permiten establecer criterios a la hora de elegir los componentes y tomar las decisiones durante la etapa de diseño:

- Interfaces para conectar diferentes sensores con diferentes buses de comunicación: 1-Wire, I^2C.
- Rango de alimentación desde los 5V hasta los 24V en corriente continua.
- Bajo coste del nodo completo.
- Bajo consumo energético.
- Robustez del diseño para usar en un entorno de producción real.
- Diseño open source.

3.1.1. Unidad de control y procesamiento

Arduino Pro Mini es una placa basada en el microcontolador ATmega328P. Es la placa de desarrollo Arduino de menor coste y por su simplicidad y tamaño está destinada a su uso en aplicaciones finales.

El microcontrolador viene preprogramado con el bootloader de Arduino por lo que podemos cargarle el firmware a través de puerto serie. A diferencia de otros modelos de Arduino que pueden conectarse directamente al ordenador para ser programados, en esta placa es necesario emplear un adaptador FTDI para conectar el puerto serie del microcontrolador a un puerto USB. Además del microcontrolador ATmega328P, la placa incorpora un regulador de tensión, un oscilador de 8MHz y un botón de reset del módulo completo.

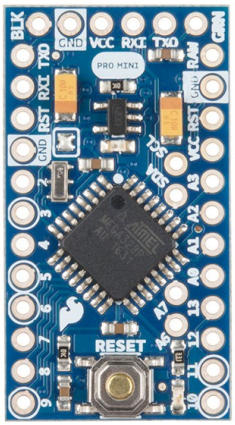


Figura 3.1: Arduino Pro Mini.

Microcontrolador:	ATmega328P
Tensión de funcionamiento:	3.3 V o 5 V (dependiento del modelo)
Pines digitales de E/S:	14 (6 de los cuales tienen salida PWM)
Pines de entrada analógica:	6
Intensidad máxima por E/S:	40 mA
Memoria Flash:	32 KB (2 KB para bootloader)
SRAM:	2 KB
EEPROM:	1024 bytes
Frecuencia de Reloj:	8 MHz (modelo de 3.3 V)
Buses de comunicaciones:	1 USART, 1 bus SPI, 1 bus I2C
Interrupciones externas:	2

Tabla 3.1: Características de la placa Arduino Mini Pro.

El ATmega328p es un microcontrolador de 8 bits, basado en tecnología AVR RISC. Al ejecutar instrucciones en un solo ciclo de reloj, el microcontrolador logra un rendimiento cercano a 1 MIPS por MHz, equilibrando el consumo de energía y la velocidad de procesamiento.

El modelo seleccionado para integrar en el diseño es el de 3,3 V, ya que podemos conectarlo directamente al módulo de comunicaciones LoRa sin necesidad de adaptar los niveles de tensión de las señales, reduciendo así el número de componentes y el coste. Además, el modelo de 3,3 V trabaja a una frecuencia menor, reduciendo el consumo energético del microcontrolador.

3.1.2. Transceptor de radio LoRa

Como ya ha sido presentado en el Capítulo 2, la comunicación inalámbrica de los nodos sensores se ha implementado usando una red LPWAN, concretamente un red LoRaWAN basada en la tecnología LoRa patentada por Sentech.

Sentech fabrica diferentes modelos de chips que incorporan la tecnología LoRa. La familia SX127x es la primera generación de módulo LoRa en el mercado y dentro de ésta nos encontramos tres modelos cuya diferencia fundamental es la banda de frecuencia de funcionamiento. La familia SX126x, sacada al mercado en 2019, ofrece mejoras tanto en el consumo energético como en la ganancia de transmisión y sensibilidad de recepción. De igual forma, dentro de la familia SX126x encontramos varios modelos adaptados a diferentes frecuencias de uso.

Para el uso del espectro electromagnético debemos ceñirnos a las frecuencias de libre uso conocidas como ISM. Son frecuencias internacionalmente libres para uso no comercial en aplicaciones de tipo industrial, científico y médico.

La Tabla 3.2 presenta las principales frecuencias ISM compatibles con los dispositivos LoRa propuestos.

Banda ISM (MHz)	Frecuencia mínima (MHz)	Frecuencia máxima (MHz)	Zonas geográficas
433	433.05	434.79	Europa, África, norte de Asia.
869	868	870	Europa, África, Asia y Oceanía
900	902	928	América continental

Tabla 3.2: Bandas ISM compatibles con el uso de LoRa.

Con el objetivo de facilitar el desarrollo del prototipo se ha empleado un módulo que contiene tanto el chip transceptor de radio como el front-end con los componentes electrónicos necesarios para su correcto funcionamiento. Aunque el transceptor SX1261 sería el más adecuando para la aplicación que estamos desarrollando, aún no existen en el mercado módulos de bajo coste que integren dicho chip. El módulo seleccionado es el RFM95W de la empresa HOPERF, que integra el transceptor SX1276 y gracias a su diseño puede ser soldado superficialmente sobre una PCB (Figura 3.2).



Figura 3.2: Módulo RFM95W.

Frecuencia:	868/915 MHZ
Modo de modulación:	LoRA/FSK/GFSK
Potencia de transmisión:	20 dBm
Sensibilidad de recepción:	-148 dBm en FSK
Tasa de transmisión:	18 bps 37,5 Kbps en LoRa
Corriente de corriente:	0,2 uA (sleep)
	120 mA (transmisión 20 dBm)
	10,3 mA (recepción)
Interfaz de datos:	SPI
Resistencia de la antena:	50 Ohm
Rango temperatura:	-20 °C - 70 °C
Tensión alimentación:	1,8 V - 3,7 V
Tamaño:	16*16 mm

Tabla 3.3: Características del módulo RFM95W con el chip de radio SX1276.

Aunque el chip de radio SX1276 es capaz de funcionar entre los 137 MHz y 1020 MHz configurándolo a través de los registros, em la práctica, el front-end analógico del módulo RFM95W está preparado para trabajar en la banda de 868/915MHz MHz.

El bus SPI (Serial Peripheral Interface) es el estándar de comunicación empleado para la transferencia de información entre el microcontrolador y el transceptor SX1276. Es un protocolo síncrono en el que todas las transmisiones son referidas a una señal de reloj común generada por el dispositivo maestro. Se pueden conectar diferentes dispositivos a la misma interfaz SPI, siendo el maestro el encargado de seleccionar el esclavo al que se quiere conectar asertando la entrada Slave Select (SS) del dispositivo correspondiente [10].

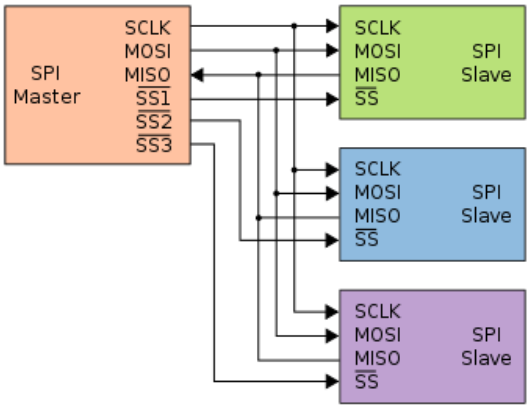


Figura 3.3: Diagrama de conexión del bus SPI.

Fuente Imagen: https://commons.wikimedia.org/

La sincronización y la transmisión de datos se realiza por medio de 4 señales:

- **SCLK** (Clock Signal): Es el pulso que marca la sincronización. Con cada pulso de este reloj, se lee o se envía un bit.
- **MOSI** (Master Output Slave Input): Salida de datos del maestro y entrada de datos al esclavo.
- **MISO** (Master Input Slave Output): Salida de datos del esclavo y entrada al maestro.
- **NSS** (Negative Slave Select): permite seleccionar el esclavo con el que se va a comunicar el dispositivo maestro.

A nivel lógico de sincronización y transmisión de datos, la comunicación SPI requiere la configuración de la polaridad del reloj (CPOL) y el bit fase del reloj (CPHA). Los dispositivos de radio LoRa SX1276 utilizan la configuración CPOL = 0 y CPHA = 0 que

se conoce como modo 0 SPI. La velocidad de la señal de reloj (SCLK) no debe superar los 10MHz.

Además del bus de comunicación SPI, el módulo RFM95W dispone de otros pines de entrada y salida que permiten interactuar con el microcontrolador para realizar funciones especiales. Por ejemplo, resetear el módulo desde el microcontrolador o avisar a este, mediante una interrupción, de la recepción de un paquete. En la Figura 3.4 se muestran los pines del módulo RFM95W que han sido conectados con entradas y salidas del microcontrolador.

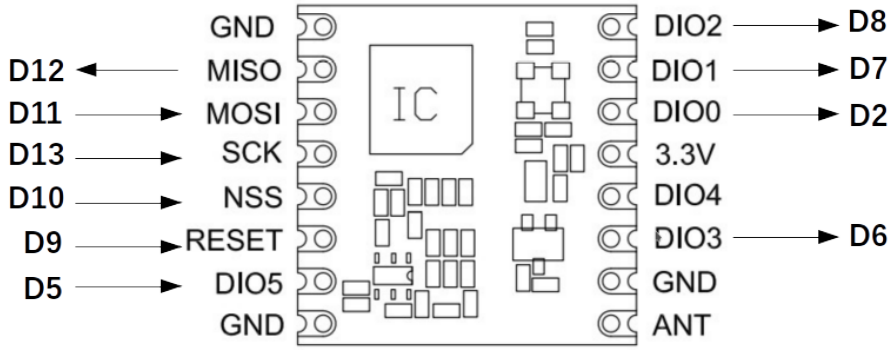


Figura 3.4: Diagrama conexión RFM95W con el microcontrolador.

Fuente Imagen: RFM95W Datasheet

3.1.3. Convertidor DC-DC

A partir de los componentes electrónicos que se han montado en el nodo sensor, un requisito de diseño es que la tensión de funcionamiento sea de 3,3 V. Como sería muy restrictivo y poco versátil limitar la tensión de alimentación del nodo a 3,3 V, se ha integrado un convertidor DC/DC Buck de potencia, que permite una entrada entre 4,75 V y 23 V para proporcionar una salida fija de 3,3 V. Esta tecnología para reducir tensión es más eficiente que el empleo de un regulador, aunque algo más compleja al necesitar un circuito de conmutación. Por suerte, existen en el mercado circuitos integrados a un coste reducido que realizan esa función y añadiendo unos pocos elementos pasivos se consigue unos resultados muy buenos. En este proyecto se ha utilizado el dispositivo MP2307 integrado en un módulo soldable en superficie. En la Figura 3.5 se muestra el esquema del circuito DC/DC implementado.

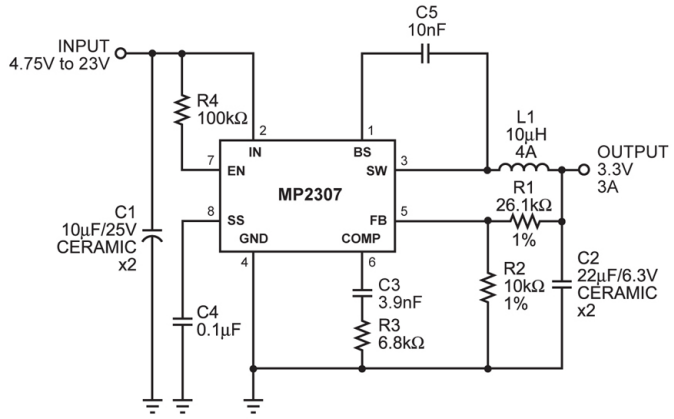


Figura 3.5: Convertidor DC/DC Buck.

Fuente Imagen: https://www.monolithicpower.com/

3.1.4. Interfaz de usuario

A nivel hardware se ha implementado una interfaz de usuario reducida destinada al testeo de los dispositivos tras su configuración. Esta interfaz consta de un pulsador y un LED (Light-emitting diode). Al activar el pulsador el sistema fuerza una adquisición de los sensores y su posterior transmisión inalámbrica. Esta funcionalidad permite testear el funcionamiento y la conexión del dispositivo una vez ha sido instalado, sin necesidad de esperar el tiempo entre muestra y muestra configurado en el dispositivo. El encendido y posterior apagado del led indica que la adquisición y transmisión ha sido correcta.

Aun siendo de gran ayuda, para aplicaciones de muy bajo consumo, se debe evitar activar el LED en cada envío de datos para reducir al máximo la energía consumida por el dispositivo.

3.1.5. Diseño de la PCB

El proceso de diseño se ha realizado con el software KiCAD. Es un paquete de software libre para la automatización del diseño electrónico EDA (Electronic Design Automation). Integra en el mismo entorno diferentes utilidades para la realización de todas las etapas del diseño: captura esquemática, diseño de la PCB, generación/visualización de archivos Gerbers.

Una vez se han definido todos los elementos que se van a montar en la PCB, se ha procedido al diseño del esquemático en el que se establecen las conexiones entre todos los elementos (ver Apéndice A.1). Posteriormente se realiza el diseño de la PCB, mediante el posicionamiento de los componentes y el trazado de las pistas de conexión.

Por último, tras verificar el diseño se han generado los Gerbers empleados para la fabricación de la PCB, Apéndice A.2. La Figura 3.6 muestra un render en 3D de la PCB realiza con KiCAD.

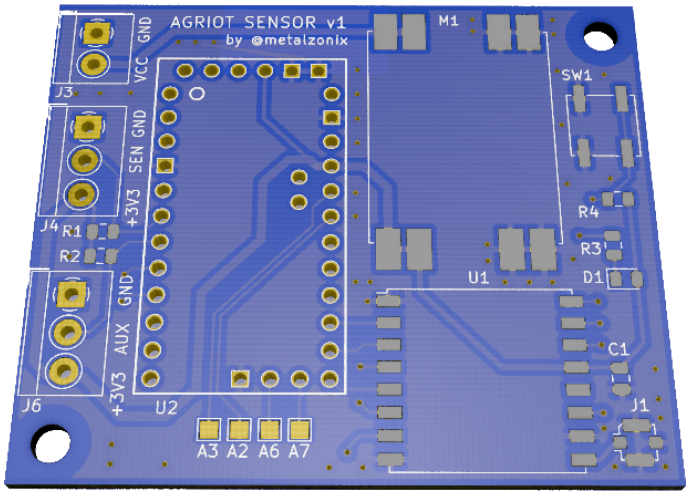


Figura 3.6: Render de la PCB del nodo sensor generado con KiCAD.

El resultado de la PCB, una vez fabricada, se muestra en la Figura 3.7.[1]

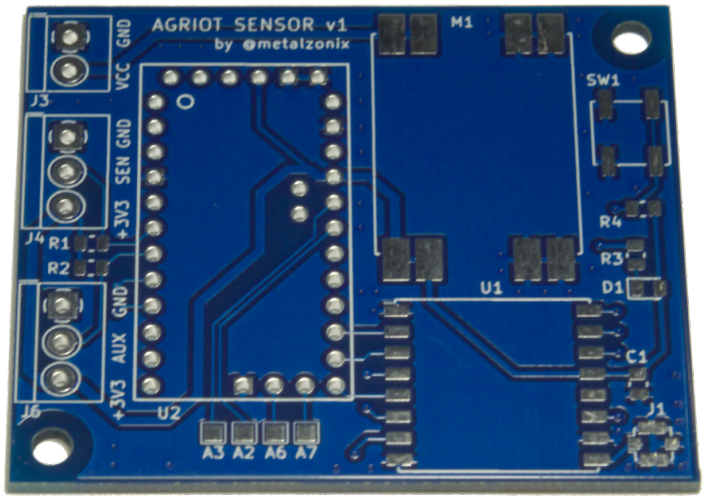


Figura 3.7: Imagen real de la PCB del nodo sensor sin componentes.

[1]Las PCBs han sido fabricadas por la empresa JLCPCB. Ésta ofrece un servicio para la fabricación de pocas unidades para prototipos (sin panelizar) a un precio muy asequible. Además, desde la propia página web se pueden configurar diferentes aspectos de la PCB, como el color, el espesor, el material de acabado, etc.

En la Figura 3.8 podemos ver la PCB del nodo sensor fabricado. La imagen de la izquierda muestra el lado superior de la PCB con todos los componentes montados, y en la derecha el lado inferior.

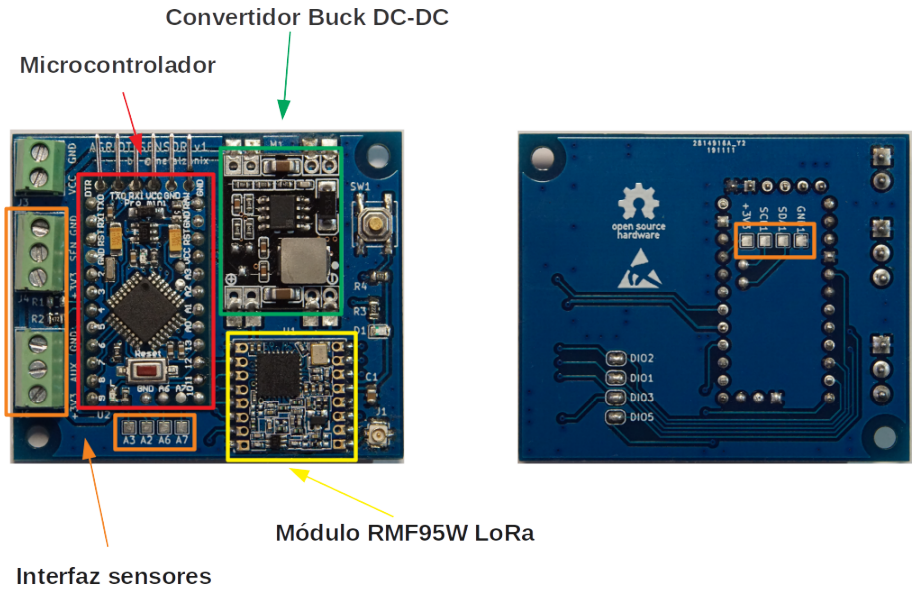


Figura 3.8: Imagen real del nodo sensor ya montado con todos sus componentes.

3.1.6. Sensores

En esta sección veremos las características de los diferentes sensores que han sido probados e integrados con el nodo sensor.

3.1.6.1. Humedad y temperatura ambiente - AM2302

El AM2302 es un sensor digital precalibrado de temperatura y humedad. También conocido como DHT22 podemos encontrarlo en dos formatos, uno pensando para ser soldado sobre una PCB y otro, de mayor tamaño, para conectarse por cable. Sin llegar a ser un sensor de alta precisión, tiene unas características aceptables para emplearlo en proyectos reales de monitorización, que requieran una precisión media.

Internamente está compuesto por un sensor capacitivo de humedad y un termistor para medir la temperatura del aire circundante, además dispone de un procesador interno que realiza la adquisición y proporciona la medición mediante una señal digital. Cada sensor es calibrado en fábrica, guardando los coeficiente de compensación en la memoria interna, los datos devueltos por el sensor son compensados con los datos de calibración.

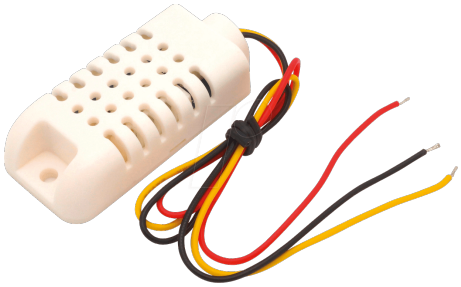


Figura 3.9: Sensor AM2302/DHT22 con conexionado.

Fuente Imagen: https://cdn-reichelt.de

Además de las características comentadas, su pequeño tamaño, bajo consumo y la larga distancia de transmisión de datos por el bus (100 m) permiten que el AM2302 se adapte a una gran variedad de aplicaciones, incluyendo la monitorización agrícola.

Tensión de alimentación:	3,3 V - 6 V
Consumo de corriente :	2,5 mA (durante conversión)
Interfaz :	1-Wire (digital)
Rango de temperatura:	-40 °C 125 °C
Precisión temperatura a 25°C:	+-0,5 °C
Resolución temperatura:	0,1 °C (8-bit)
Rango de humedad:	0 % RH hasta los 100 % RH
Precisión humedad a 50 %:	2-5 % RH
Resolución humedad:	0,1 % RH
Frecuencia de muestreo:	0,5 Hz
Tamaño:	16*16 mm

Tabla 3.4: Características del sensor AM2302.

El sensor utiliza su propio sistema de comunicación bidireccional mediante un único cable, empleando señales temporizadas. Está especialmente diseñado para este sensor y no es compatible con el bus 1-Wire desarrollado por la empresa Maxim/Dallas, ya que no implementa el protocolo tal cual está definido Maxim.

Cuando el microcontrolador envía la señal de inicio, el sensor cambia su estado de standby a running. Un vez se finalizada la trama de inicio, el sensor responde transmitiendo 40 bits de datos, que corresponden con 2 bytes para el valor humedad, 2 bytes para el valor temperatura y 1 byte final para la comprobación de errores (CRC). Una vez que los datos han sido transmitidos el sensor vuelve a stanby [11].

3.1.6.2. Temperatura del suelo - DS18B20

El DS18B20 es un sensor de temperaturas digital fabricado por la compañía Maxim Integrated. Se comunica a través del bus 1-Wire para la comunicación con el microcontrolador. Cada DS18B20 tiene un código serial único de 64 bits, que permite que varios DS18B20 funcionen en el mismo bus, además de una comunicación robusta de larga distancia (50 m) sin que se deteriore la señal [12].

Una de las ventajas del DS18B20 es que se comercializa tanto en un integrado TO-92 como en forma de sonda impermeable, lo que permite realizar mediciones de temperatura en líquidos y gases (Figura 3.10).



Figura 3.10: Sensor DS18B20 en sus dos formatos de encapsulado.

Fuente Imagen: https://www.robotics.org

El protocolo 1-Wire permite que todos los sensores conectados al bus se alimenten a través de la línea de datos. Para ello, disponen de un condensador que almacena energía mientras la línea de datos está en HIGH. Este modo se denomina "modo parásito". En caso de no usar el modo parásito, los dispositivos deberán ser alimentados a una tensión entre 3,0 V y 5,5 V. En la Tabla 3.5 se muestran en detalle las características eléctricas del sensor DS18B20.

Tensión de alimentación:	3 V - 5,5 V
Consumo de corriente :	2,5 mA
Interfaz :	1-Wire (digital)
Rango de temperaturas:	-55 ºC - 12 5ºC
Error (-10ºC a 85ºC):	+-0,5 ºC
Error (-55ºC a 125ºC):	+-2ºC
Máxima Resolución:	0,0625 ºC (12-bit)

Tabla 3.5: Características del sensor DS18B20.

Para la comunicación entre el sensor y el microcontrolador se puede emplear cualquier pin digital, sin necesidad de que este se pueda configurar con interrupciones. En cualquier

caso, el bus 1-Wire requiere una resistencia de pull-up de 4k7 Ohmios entre Vcc y DQ para que funcione correctamente (Figura 3.11)

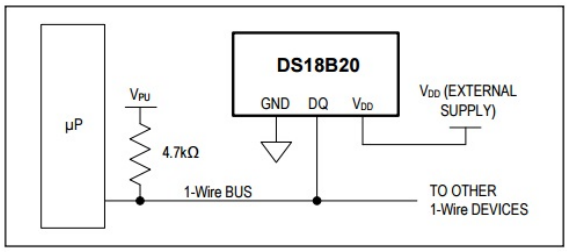


Figura 3.11: Diagrama de conexión DS1B20 con el microcontrolador.

Fuente Imagen: [12]

3.1.6.3. Humedad del suelo

La medición precisa del contenido de agua del suelo es esencial para las aplicaciones en la agronomía y la botánica, donde el riego excesivo o insuficiente del suelo puede dar lugar a la ineficacia o al desperdicio de recursos. Esta variable ambiental puede ser medida usando diferentes técnicas: gravimétricas, nucleares, electromagnéticas, tensiométricas e higrométricas [13].



Figura 3.12: Sensor capacitivo de humedad del suelo.

Fuente Imagen: https://boutique.semageek.com/

El contenido de humedad del suelo puede determinarse por su efecto sobre la constante dieléctrica del medio, midiendo la capacidad entre dos electrodos implantados en el suelo: en suelos secos la permitividad relativa del medio está entre 2-6 y la del agua tiene un valor aproximado de 80 [14]. Así, cuando la humedad del suelo es predominantemente en forma de agua (por ejemplo, en suelos arenosos), la constante dieléctrica es directamente proporcional al contenido de humedad. Normalmente se da a la sonda una excitación de

frecuencia para permitir la medición de la constante dieléctrica. La lectura de la sonda no es lineal con el contenido de agua y está influida por el tipo de suelo y su temperatura, por lo tanto, se requiere una calibración dependiente del tipo de suelo

El sensor empleado para la medición de la humedad del suelo se denomina "Capacitive Soil Moisture Sensor v1.2" y se muestra en la Figura 3.12. Éste, mide los niveles de humedad por medio de una detección capacitiva en lugar de una resistiva como otros sensores de bajo coste que podemos encontrar en el mercado. Está hecho de material resistente a la corrosión, lo que le da una excelente vida útil. Las características eléctricas son recogidas en la Tabla 3.6

Tensión de alimentación:	3,3 V - 5,5 V
Consumo de corriente:	5 mA
Tipo de señal:	analógica

Tabla 3.6: Características del sensor de humedad capacitivo v1.2.

Al tratarse de un sensor analógico, la señal del sensor que es transmitida hasta el microcontrolador, se verá afectada por la longitud de cable y tendrá poca inmunidad ante ruidos electroctromagnéticos. Con el objetivo de mejorar la robustez del sensor, se ha incluido un ADC (Analog-to-digital converter) de 12 bits con comunicación I^2C (Inter-Integrated Circuit), en una PCB anexa al sensor. De esta forma, la señal del sensor es convertida y transmitida hacia el microcontrolador de forma digital. Además, al usar un ADC de 12 bits, la resolución será mayor que usando el ADC de 10 bits del microcontrolador.

El bus I^2C es un estándar que facilita la comunicación entre microcontroladores, memorias, sensores y otros dispositivos con cierto nivel de "inteligencia", y que solo requiere de dos líneas de señal y una línea de masa. Fue diseñado a este efecto por Philips y permite el intercambio de información entre muchos dispositivos a una velocidad aceptable, de unos 100 Kbits por segundo, aunque hay casos especiales en los que el reloj llega hasta los 3,4 MHz [10, cap. 8].

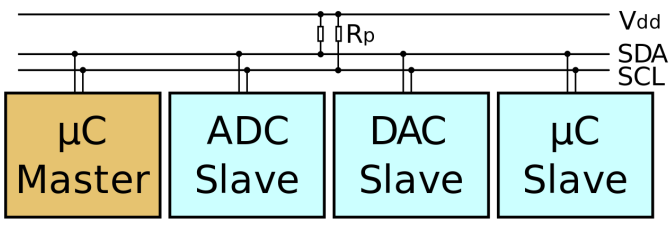


Figura 3.13: Diagrama de conexión del bus I^2C.

Fuente Imagen: https://www.wikipedia.org

La metodología de comunicación de datos del bus I^2C es en serie y sincrónica. Una de las señales del bus marca el tiempo (pulsos de reloj) y la otra se utiliza para intercambiar datos. La dos señales de del bus son:

- **SCL** (Serial Clock) es la línea de los pulsos de reloj que sincronizan el sistema.
- **SDA** (Serial Data) es la línea de transferencia de datos entre los dispositivos.

Las líneas SDA y SCL son del tipo drenador abierto. Se deben polarizar en estado alto (conectando a la alimentación por medio de resistencias "pull-up") para que el funcionamiento del bus sea correcto (ver Figura 3.13). El valor de las resistencias de polarización no es muy crítico, y puede ir desde 1k8 a 47k ohmios. Un valor menor de resistencia incrementa el consumo de los integrados pero disminuye la sensibilidad al ruido y mejora el tiempo de los flancos de subida y bajada de las señales.

3.1.7. Comunicaciones LoRaWAN

La especificación LoRaWAN define tres tipos de dispositivos. Todos los dispositivos LoRaWAN deben implementar la clase A, mientras que la clase B y la clase C son extensiones de la especificación de los dispositivos de clase A (ver Figura 3.14).

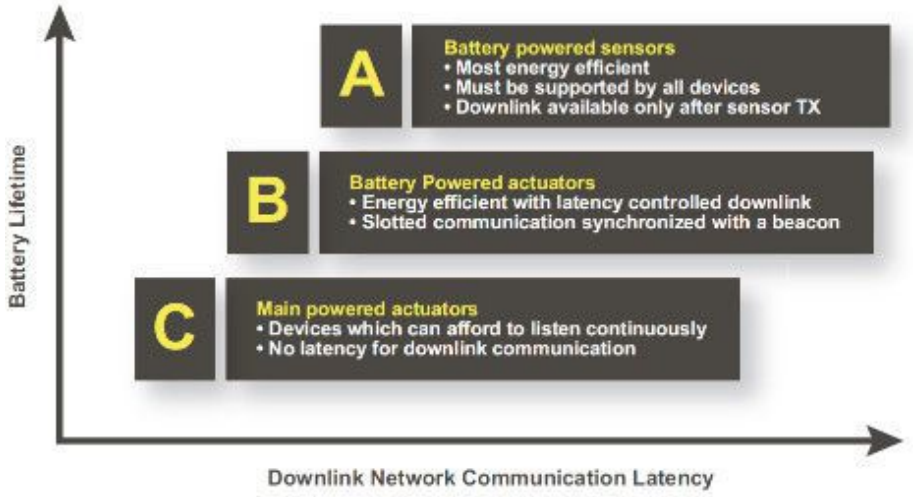


Figura 3.14: Relación entre la latencia y la duración de la batería para distintas clases de nodos.

Fuente Imagen: https://alfaiot.com/

- Dispositivos finales bidireccionales (clase A). Permiten comunicaciones bidireccionales en las que la transmisión del enlace ascendente de cada dispositivo final va seguida de dos ventanas cortas de recepción de enlace descendente. Este modo de

funcionamiento es el de menor consumo energético ya que la recepción sólo está activa durante un corto periodo tras el envío de un mensaje de enlace ascendente. Es el modo de funcionamiento programado en el nodo sensor.

- Dispositivos finales bidireccionales con lotes de recepción programados (clase B). Amplían la clase A al añadir ventanas de recepción en horarios programados para los mensajes de enlace descendente del servidor. Para abrir ventanas de recepción a la hora programada, los dispositivos finales reciben un beacon de sincronización con la hora de la estación base.

- Dispositivos finales bidireccionales con ranuras de recepción máximas (clase C). Los dispositivos finales de clase C tienen ventanas de recepción casi continuamente abiertas, y sólo se cierran cuando transmiten a expensas de un consumo excesivo de energía.

Desde el punto de vista de la seguridad, las comunicaciones están protegidas mediante claves AES128 tanto para cifrado como para firma de mensajes, confiriendo cierto grado de confidencialidad e integridad. El desacoplo entre los servidores de red y aplicación motiva el empleo de dos claves AES128: una para proteger los metadatos de la capa MAC (clave de sesión de red, o NwkSKey), y otra para proteger los payloads de aplicación (clave de sesión de aplicación, o AppSKey).

El proceso de registro del dispositivo en la red se denomina activación, y tiene implicaciones en la gestión de estas claves. LoRaWAN ofrece dos métodos de activación:

- **OTAA (Over-The-Air Activation)**. En OTAA, el nodo final envía un paquete MAC de tipo "Join Request", el cual contiene el DevEUI [2] del dispositivo y el AppEUI[3] del servidor de aplicación con el que quiere comunicarse. Si la solicitud es aceptada, la red responde con un mensaje de tipo "Join Accept" cuyo contenido está cifrado con la AppKey, y del cual se obtienen las claves de sesión NwkSKey y AppSKey. El dispositivo puede almacenar esas claves y seguir usándolas para comunicarse. Si se pierden o la red decide caducarlas, el dispositivo debe volver a unirse para generar nuevas claves. En la Figura 3.15 se muestra el proceso de activación OTAA de forma gráfica.

- **ABP (Activation By Personalisation)**. Al emplear este modo, los nodos cuentan con la DevAddr, NwkSKey y AppSkey preprogramada. Debido a que esta información ya existe, los nodos pueden comunicarse con la red directamente sin intercambiar mensajes de Join.

[2]Identificador único extendido de 64 bits del dispositivo final. Es asignado por el fabricante, o el propietario del dispositivo.

[3] Identificador único extendido de 64 bits de la aplicación. Es generado por el servidor de aplicaciones LoRaWAN.

El modo OTAA ofrece una mayor seguridad y permite cambiar el dispositivo de red sin necesidad de ser reprogramado, por estas dos razones es el modo implementando en el nodo sensor.

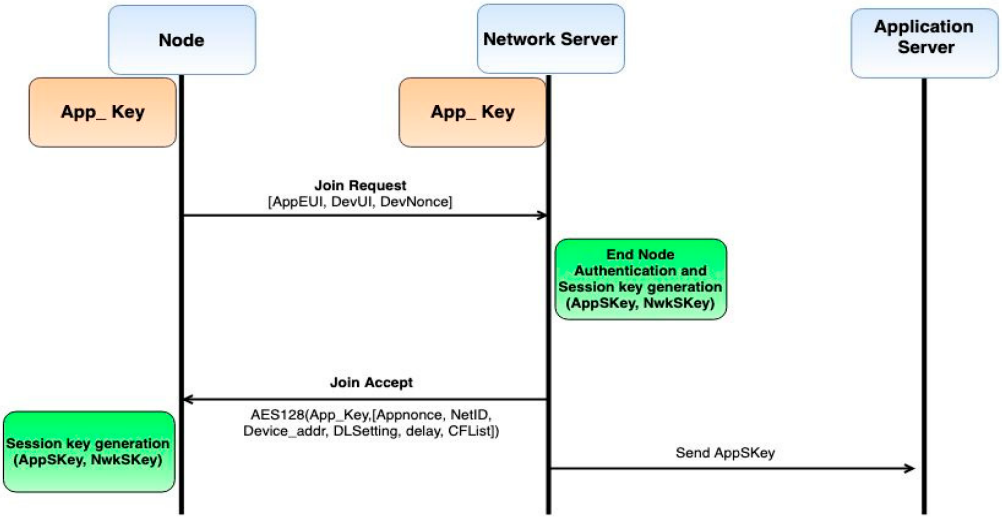


Figura 3.15: Protocolo de activación OTAA.

Fuente Imagen: https://www.mdpi.com/

En el protocolo LoRa nos encontramos tres parámetros claves en la transmisión de datos: potencia de transmisión, ancho de banda y factor de dispersión.

A mayor **potencia de transmisión**, también será mayor el alcance de la señal y el consumo de energía del dispositivo. La regulación de la UE (Unión Europea) limita a 14 dBm la potencia máxima con la que se puede transmitir en la banda de 868 Mhz, por lo que tendremos un límite legal que no se debe superar.

La tasa de transmisión depende del **ancho de banda** y del **factor de dispersión** empleado. LoRaWAN puede utilizar canales con un ancho de banda de 125 kHz, 250 kHz o 500 kHz, dependiendo de la región o del plan de frecuencias (ver Tabla 3.8). El SF es establecido por el dispositivo y permite controlar el número de Chirps (Compressed High Intensity Radar Pulse) enviados por segundo. Un SF más bajo significa que se envían más Chirps por segundo, por lo tanto se pueden codificar más datos por segundo aumentando la tasa de transmisión. Por otro lado, el envío de la misma cantidad de datos con un SF más alto requiere un mayor tiempo de transmisión, lo que implica que el transceptor está activo más tiempo y consumiendo más energía. El protocolo LoRa define un rango de SF entre 7 y 12, pudiendo apreciar en la Figura 3.16 una comparativa del tiempo de transmisión para los diferentes SF permitidos.

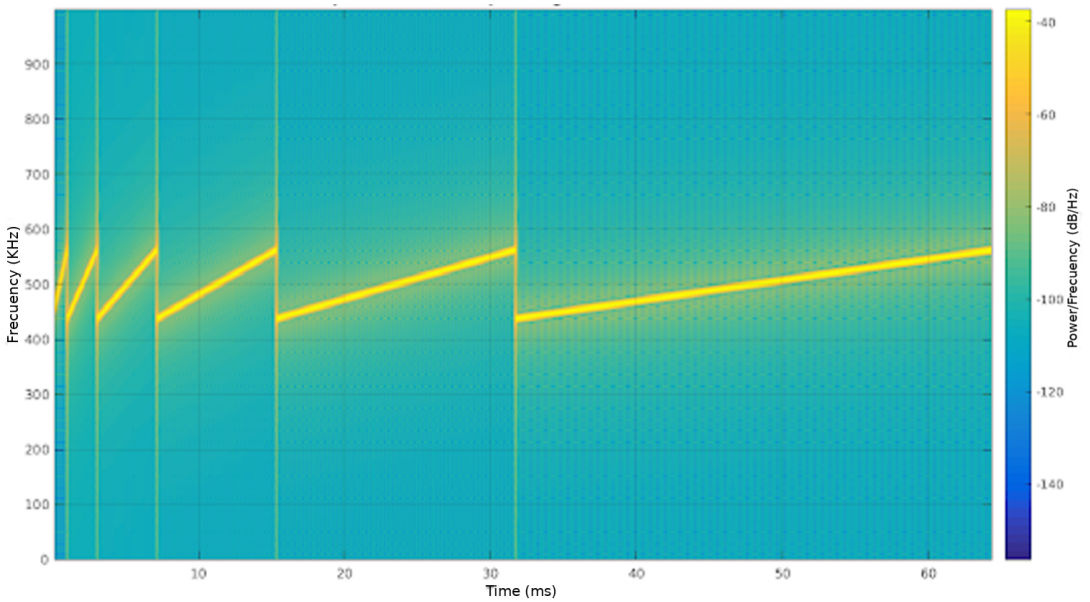


Figura 3.16: Chirps con distinto factor de dispersión.

Fuente Imagen: https://www.sghoslya.com/p/lora-is-chirp-spread-spectrum.html

Al incrementar el SF se consigue un tiempo de transmisión (airtime) más prolongado, dando al receptor más oportunidades de muestrear la señal, lo que se traduce en una mayor sensibilidad e inmunidad frente a ruido. También implica que se pueda recibir la señal más lejos, por lo que se obtiene una mejor cobertura sin necesidad de aumentar la potencia de transmisión. Por contra, se reduce la tasa de transmisión (bitrate) y se aumenta el consumo de energía del dispositivo (ver Figura 3.17).

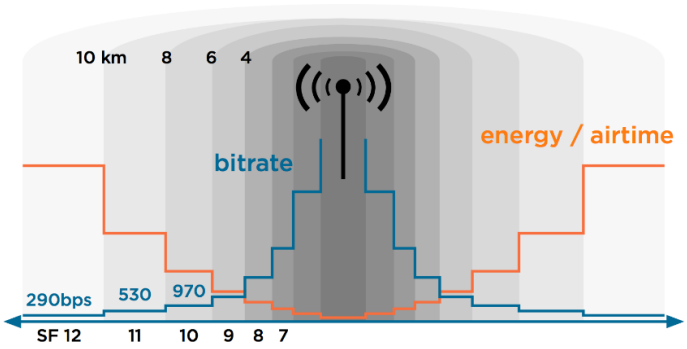


Figura 3.17: Relación distancia vs bitrate vs airtime.

Fuente Imagen: https://lora-developers.semtech.com/

Teóricamente, cada paso en SF duplica el tiempo para transmitir la misma cantidad de datos y se correlaciona con unos 2,5dB extra de ganancia de enlace. En la Tabla 3.7 se muestra la SNR necesaria para demodular un mensaje en función de su SF.

SF	Chirps/símbolo	SNR mínima
6	64	-5 dB
7	128	-7.5 dB
8	256	-10 dB
9	512	-12.5 dB
10	1024	-15 dB
11	2048	-17.5 dB
12	4096	-20 dB

Tabla 3.7: SNR necesaria para demodular un mensaje en función de su SF.

Otro parámetro importante en el protocolo LoRa es el **CR (coding rate)** que nos indica la forma de codificar para la corrección de errores. Según la técnica especificada, añade símbolos de control para saber si los datos son correctos o no e incluso poder corregir los valores incorrectos. Este parámetro puede tomar valores entre 1 y 4.

Mediante la ecuación 3.1 podemos obtener la tasa de transmisión efectiva en bits por segundo (bps) en función de los parámetros establecidos: factor dispersión (SF), ancho de banda (BW) y ratio de codificación (CR).

$$R_b = SF * \frac{\left[\frac{4}{4+CR}\right]}{\left[\frac{2^{SF}}{BW}\right]} * 1000 \tag{3.1}$$

Frecuencia	SF y BW	Tipo de enlace
868.1	SF7BW125 a SF12BW125	Uplink/Downlink (RX1)
868.3	SF7BW125 a SF12BW125 and SF7BW250	Uplink/Downlink (RX1)
868.5	SF7BW125 a SF12BW125	Uplink/Downlink (RX1)
867.1	SF7BW125 a SF12BW125	Uplink/Downlink (RX1)
867.3	SF7BW125 a SF12BW125	Uplink/Downlink (RX1)
867.5	SF7BW125 a SF12BW125	Uplink/Downlink (RX1)
867.7	SF7BW125 a SF12BW125	Uplink/Downlink (RX1)
867.9	SF7BW125 a SF12BW125	Uplink/Downlink (RX1)
868.8	FSK	Uplink/Downlink (RX2)
869.525	SF9BW125 (RX2)	Downlink (RX2)

Tabla 3.8: Planes de frecuencia utilizadas en The Things Network en la banda EU863-870.

En la Tabla 3.8 se muestran la diferentes frecuencias definidas en The Things Network para ser empleadas en Europa en la banda EU868-870. Además de la frecuencia,

podemos ver el ancho de banda de cada canal, el SF y si se trata de un canal de enlace ascendente/descendente.

El **ADR (Adaptative Data Rate)** es un mecanismo para ajustar la tasa transmisión con el objetivo de reducir el uso de la red y por tanto el consumo de energético. Éste debe emplearse siempre que un dispositivo final tenga condiciones estables de comunicación, lo que implica que, generalmente, solo se use para dispositivos estáticos. Los dispositivos finales deciden qué SF deben emplear apoyándose en la información de calidad de enlace aportada por la red.

Para determinar la velocidad de datos óptima, la red necesita algunas mediciones (mensajes de enlace ascendente). Estas mediciones contienen el contador de tramas, la SNR y el número de gateways que recibieron cada mensaje de enlace ascendente. Para cada una de estas mediciones, se toma la SNR del mejor gateway y se calcula el llamado "margen", que es la SNR medida menos la SNR requerida para demodular un mensaje dado el SF (ver Tabla 3.7). Este margen se utiliza para determinar cuánto podemos aumentar la tasa de datos o disminuir la potencia de transmisión.

Por ejemplo, cuando la red recibe un mensaje con SF12 y un SNR de 5 dB, ese mensaje tiene un margen de 25 dB (hasta los -20 dB). Esto supone un desperdicio de energía y tiempo de uso del canal. Si aumentáramos la tasa de datos a SF7 todavía tendríamos un margen de 12,5 dB (hasta -7,5 dB), pudiendo incluso reducir la potencia de transmisión para ahorrar aún más energía y causar menos interferencias.

En la Figura 3.18 se muestras los tiempos de transmisión para un mensaje de 22 bytes transmitido con diferente SF. También podemos ver el número máximo de mensajes a transmitir durante un día para cumplir con un ciclo de uso del 1 % máximo del espectro de radiofrecuencia en Europa.

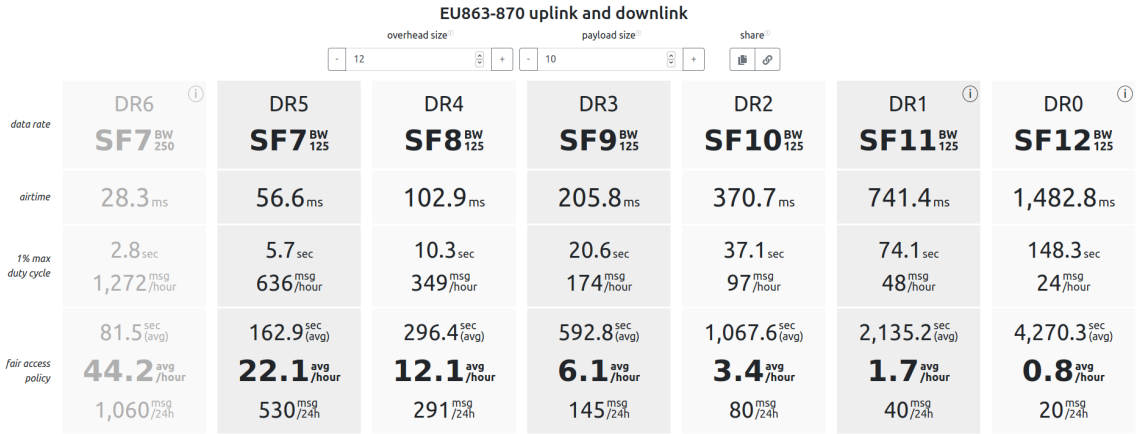


Figura 3.18: Tiempo de uso máximo del canal según el tamaño del mensaje y el SF.

Fuente Imagen: https://lora-developers.semtech.com/

3.1.8. Firmware

El desarrollo del firmware se ha realizado con el IDE Visual Studio Code y el plugin Platformio a través del cual tenemos soporte para usar el framework de Arduino. Para el control de versiones de código se ha usado Git, a través de la plataforma online Github. También se ha empleado Travis-CI, que es un sistema de integración continua y gratuito para proyectos de código abierto. Travis-CI se integra con GitHub y automáticamente ejecuta el pipeline definido tras cada push al servidor. Gracias a este sistema comprobamos que cualquier cambio hecho en el código es compilado correctamente.

El uso de librerías de código abierto ha permitido implementar un gran número de funcionalidades sin necesidad de desarrollar todo el código desde cero, un trabajo que hubiera incrementado el tiempo de desarrollo exponencialmente. Algunas de las librerías están integradas por defecto en el framework de Aduino como por ejemplo EEPROM, Wire o SPI, que son librerías de bajo nivel para gestionar las comunicaciones o acceder a la memoria EEPROM del microcontrolador. Otras librerías, como por ejemplo LMIC-Arduino, implementan la gestión de la capa MAC así como la comunicación con el módulo de radio RFM95W.

Las librerías que no están incluidas por defecto en el framework de Arduino, se han obtenido de repositorios Git públicos y han sido añadidas al proyecto en forma de submódulos Git. De esta manera, tendremos identificada de forma inequívoca la versión de cada librería que se ha usado y, de forma automática, es descargada e incluida en el proyecto. Si una librería es actualiza por sus creadores, es fácil traer esos cambios al proyecto. En la Lista 3.1 se enumeran todas las librerías que han sido empleadas para el desarrollo de este proyecto.

```
1  |-- <ArduinoUniqueID> 1.0.9
2  |-- <EEPROM> 2.0
3  |-- <AsyncSerial> 1.0.0
4  |-- <LMIC-Arduino> 1.5.0+arduino-3
5  |----- <SPI> 1.0
6  |-- <SDISerial>
7  |-- <CayenneLPP> 1.1.0
8  |----- <ArduinoJson> 6.15.2
9  |-- <Wire> 1.0
10 |-- <DHT sensor library> 1.3.10
11 |----- <Adafruit Unified Sensor> 1.1.4
12 |-- <OneWire> 2.3.5
```

Lista 3.1: Librerías de código libre empleadas en el proyecto.

3.1.8.1. Comunicación de radio

La librería Arduino LMIC (LoRaWAN MAC in C) de IBM es una implementación en C para la plataforma Arduino de la especificación LoRaWAN1.0.2 para dispositivos finales. Es compatible con las variantes EU-868, US-915, AU-915, AS-923, KR-920 e IN-866 de la especificación y puede gestionar dispositivos de clase A y clase B. La librería se encarga de controlar el transceptor de radio de Sentech y gestionar la capa MAC.

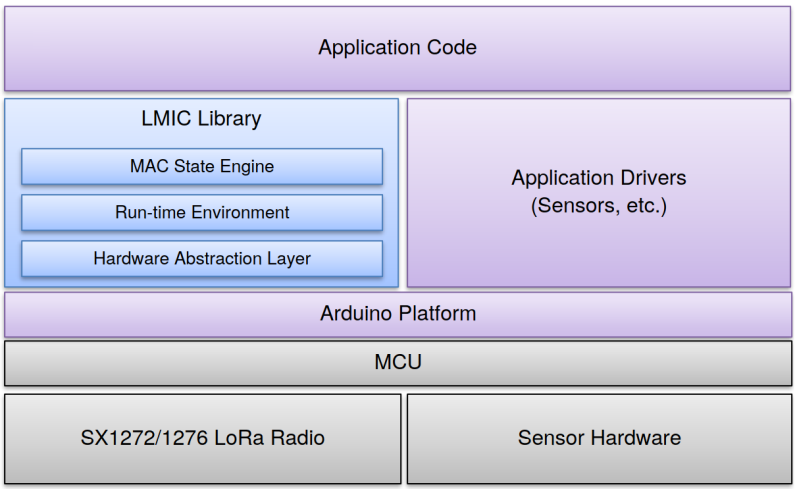


Figura 3.19: Capas de abstracción de la librería LMIC.

Fuente Imagen: LMIC Technical Specification

La librería ha sido diseñada para satisfacer con precisión las limitaciones de tiempo del protocolo MAC e incluso considerar posibles desviaciones de reloj en los cálculos de tiempo. Desde la capa de aplicación se puede acceder y configurar toda la funcionalidad a través de un simple modelo de programación basado en eventos y sin necesidad de lidiar con detalles específicos de la plataforma, logrando así un nivel de abstracción de capas que simplifica el desarrollo del código (ver Figura 3.19). Se proporciona una HAL (Hardware Abstraction Layer) de Arduino que permite una fácil integración con la mayoría de las variantes de Arduino, incluido Arduino Pro Mini. Algunas funciones soportadas por LMIC:

- Envío de paquetes de enlace ascendente, teniendo en cuenta el ciclo de trabajo.
- Cifrado y comprobación de la integridad del mensaje.
- Frecuencias personalizadas y ajustes de la tasa de datos.
- Activación por aire OTAA.
- Recepción de paquetes de enlace descendente en las ventanas RX1 y RX2.
- Procesamiento de comandos MAC.

3.1.8.2. Entorno en tiempo de ejecución

Aprovechando que la librería LMIC implementa un entorno en tiempo de ejecución, el código implementado en la capa de aplicación ha sido desarrollado bajo este modelo de programación.

Todo el código ejecutado a nivel de aplicación está contenido en los llamados trabajos "jobs" que se ejecutan en el bucle principal mediante la función **os_runloop_once()**. Cada trabajo invoca a una función en la que se implementan las acciones, por ejemplo, adquirir una medición de un sensor. Además, entre las acciones debe estar la de programar la ejecución de los siguientes trabajos, por ejemplo, medir de nuevo el sensor después de 1s, o transmitir los datos del sensor.

El instante en el que se ejecuta cada trabajo es gestionado por el entorno en tiempo de ejecución. Para la gestión de los trabajos se requiere una estructura de control adicional, osjob_t, que identifica el trabajo y almacena la información de contexto. Los trabajos no deben ser de larga duración para garantizar un funcionamiento correcto.

A nivel de aplicación se han programado cuatro trabajos de ejecución:

- **os_send()**: Realizar los envíos y comprobar el estado de la capa MAC de radio.
- **sensors_update()**: Adquirir una medida del sensor y codificar los datos.
- **cli_update()**: Comprobar la recepción de comandos por el puerto serie y procesarlos.
- **os_checkSwitch()**: Comprobar de forma periódica el estado del pulsador.

3.1.8.3. Manejo del sensor

Con el objetivo de integrar diferentes sensores en el firmware y al mismo tiempo reducir el impacto en la memoria Flash, se han creado diferentes "sabores" de compilación. Así, mediante compilación condicional el preprocesador de C decidirá qué código incluir en cada versión. En la Tabla 3.9 se muestran las tres versiones distintas que se pueden compilar en función del tipo de sensor conectado al nodo.

Flag de compilación	Descripción
SENS_DHT22	Sensor de temperatura y humedad DHT22.
SENS_DS	Sensor de temperatura suelo.
SENS_SOIL_MOINSTURE	Sensor de humedad suelo.

Tabla 3.9: Versiones de compilación.

Desde el IDE de programación Visual Studio se puede seleccionar el flag de la compilación a realizar, y en el caso de no tener ninguno seleccionado se realizará la compilación

de todos los versiones de firmware. Para facilitar la estructuración del código, para cada sensor se ha creado su propia implementación contenida en un fichero .c. Todas las implementaciones tiene dos funciones comunes, una de inicialización **sensor_ini()** y otra de actualización **sensor_update()** que es invocada de forma periódica por el sistema de tiempo de ejecución.

La función de actualización realiza la adquisición del sensor correspondiente, convierte los datos si es necesario y los codifica en el payload del mensaje LoRaWAN. Se ha empleado el protocolo Cayenne LPP (Low Power Payload) para codificar las mediciones ya que permite reducir el tamaño del payload, teniendo siempre identificado el tipo de sensor y canal del que se ha obtenido la medida. Los tipos de datos se ajustan a las Directrices de Objetos Inteligentes de la Alianza IPSO (Internet Protocol for Smart Objects), que identifican cada tipo de datos con un "Object ID".

3.1.8.4. Interfaz de línea de comandos

Se ha implementado una CLI (Command Line Interface) para la configuración de parámetros del nodo sensor. La interacción con la línea de comandos se realiza por medio del puerto serie, el protocolo de comunicación envía y transmite directamente la información en ASCII. Es un protocolo simple orientado a acceso a registros, para ello se envía el identificador del registro seguido de ":" y el valor a configurar, el final del comando se marca con el carácter final de línea "\n". Para la lectura simplemente se envía el identificador de registro y el dispositivo responde con el valor.

Los tres comandos implementados están relacionados con la configuración del dispositivo en una red LoRaWAN y se recogen en la Tabla 3.10.

Nombre	ID	Tipo	Descripción
APPEUI	E	R/W	Identificador único de la aplicación final.
APPKEY	K	R/W	Usado en OTAA para generar las claves de sensión.
DEVEUI	D	R	Identificador único del dispositivo.

Tabla 3.10: Comandos de configuración implementados.

A nivel de implementación, el módulo denominado *cli* se encarga de comprobar la recepción de comandos por el puerto serie, validarlos, extraer la información y transmitir la respuesta de nuevo por el puerto serie.

Los valores de la configuración son guardados en la EEPROM del microcontrolador, de forma que persiste tras el apagado o reprogramación del dispositivo.

3.1.8.5. Interfaz de dispositivo

En esta ejecución se comprueba el estado del botón, y en caso de detectar su pulsación se fuerza una lectura del sensor y envío del mensaje. Para evitar saturar la capa de radio y ocupar más tiempo del permitido, se comprueba que entre pulsación y pulsación ha pasado un tiempo mínimo establecido en 10 segundos.

3.2. Despligue de red LoRaWAN - TheThingsNetwork

En esta sección se describirá el proceso montaje y configuración de un Gateway LoRaWAN y su registro en la red pública The Things Newtwork. También se mostrará como registrar los dispositivos finales en esta red.

3.2.1. Construcción y registro del Gateway.

Como se ha comentado en la Sección 2.3.2, en el mercado podemos encontrar una gran variedad de gateways compatibles con LoRa. Para este desarrollo se ha optado por usar el concentrador iC880A-SPI (ver Figura 3.20). Éste es capaz de recibir hasta 8 paquetes enviados simultáneamente con diferentes factores de dispersión y también en diferentes canales.

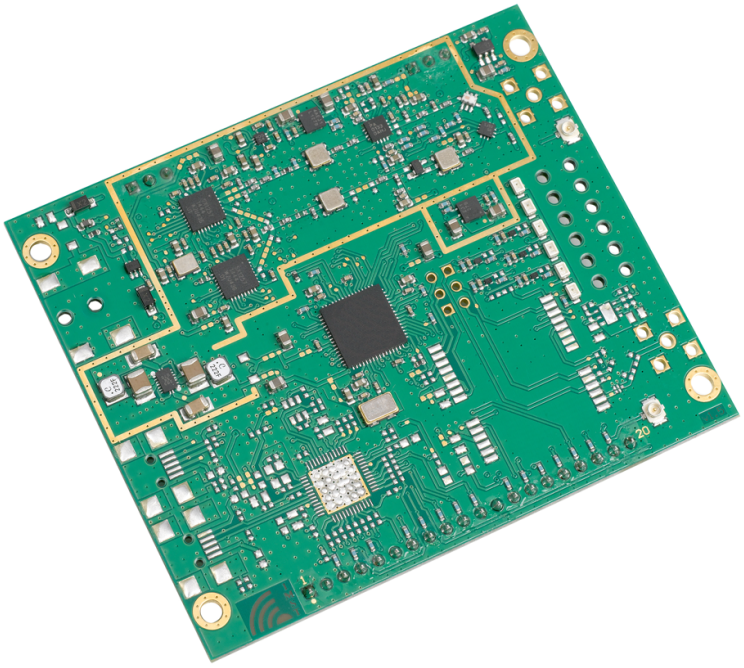


Figura 3.20: Concentrador LoRa iC880A-SPI.

Fuente Imagen: https://shop.imst.de

La placa iC880A-SPI necesita conectarse a un PC o un MCU que realice la función de host, que por la características de este proyecto se realiza por una Raspberry Pi 3. La conexión física entre el concentrador C880A-SPI con la placa Raspberry Pi, se ha realizado mediante una placa de adaptación (ver Figura 3.21) de forma que estos tres elementos quedan apilados uno sobre otro creando una conexión robusta entre ellos. Con esta combinación de elementos se puede construir un gateway LoRaWAN con unas características equivalentes a uno comercial.

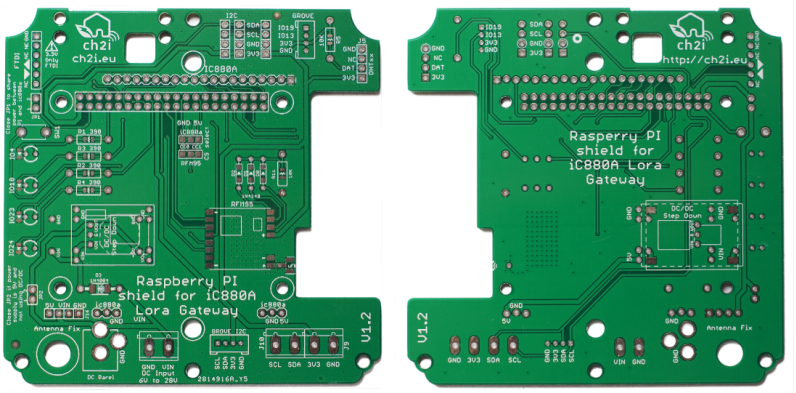


Figura 3.21: Placa de adaptación ic880A para Raspberry Pi.

La placa de adaptación también permite la integración de sensores aunque esta posibilidad no ha sido explotada. Los esquemáticos y gerbers para la fabricación de esta PCB han sido obtenidos del repositorio https://github.com/ch2i/iC880A-Raspberry-PI.

Para maximizar el alcance de la red, la antena del gateway se va a localizar en el exterior de la instalación. Ha sido necesario emplear un antena diseña para ambientes exteriores y con un cable de 2m para cubrir la distancia entre en gateway el lugar de instalación de la antena. En la Tabla 3.11 se recogen las principales características de la antena.

Rango Frec.:	824 a 960MHz
Polarización:	Lineal
Ganancia:	3dBi
V.S.W.R:	< 2.0
Impedancia:	50Ω
Cable:	RG174
Conector:	SMA macho

Tabla 3.11: Características de la antena.

La tres placas, una vez correctamente conectadas, han sido alojadas en una caja de conexión estanca de medidas 30x20x10cm, tal y como se muestra en la Figura 3.22. Desde el exterior de la caja entran tres cables: alimentación de 5V, conexión de datos Ethernet y adaptador coaxial de IPX a SMA para la conexión de la antena.



Figura 3.22: Vista superior del gateway.

Una vez realizado el montaje electrónico del gateway, se ha procedido a la configuración del software. En primer lugar es neceario descargar y grabar en una tarjeta SD el sistema operativo Rasberry Pi OS Lite, en concreto se ha empleado la versión publicada el 2020-08-20. La instrucciones detalladas para configurar el software del gateway puede ser consultadas en https://github.com/ttn-zh/ic880a-gateway/wiki. En resumen, los pasos a realizar son:

1. Conectar por SSH a la placa Raspberry.
2. Habilitar el bus SPI desde las opciones de configuración de raspi-config.

```
$ sudo raspi-config
```

3. Configurar la localización y la zona horaria.

```
$ sudo dpkg-reconfigure locales
$ sudo dpkg-reconfigure tzdata
```

4. Actualizar SO e instalar Git.

```
$ sudo apt-get update
$ sudo apt-get upgrade
$ sudo apt-get install git
```

5. Clonar el repositorio con el software del gateway e instalar:

```
$ git clone -b spi https://github.com/ttn-zh/ic880a-gateway.git ~/
    ic880a-gateway
$ cd ~/ic880a-gateway
$ sudo ./install.sh spi
```

Al finalizar la instalación se imprimirá en pantalla el EUI del gateway formado por 8 bytes en hexadecimal (por ejemplo B827EBFFFE7B80CD). Este identificar será empleado para registrar el dispositivo en la plataforma The Things Network.

Iniciando sesión en thethingsnetwork.org, nos dirigimos al apartado "Gateways" y registramos el dispositivo. Para ello, es necesario introducir el Gateway EUI obtenido en la configuración, establecer el plan de frecuencia de Europa e indicar el uso del "Semtech packet forwader" tal y como se muestra en la Figura 3.23.

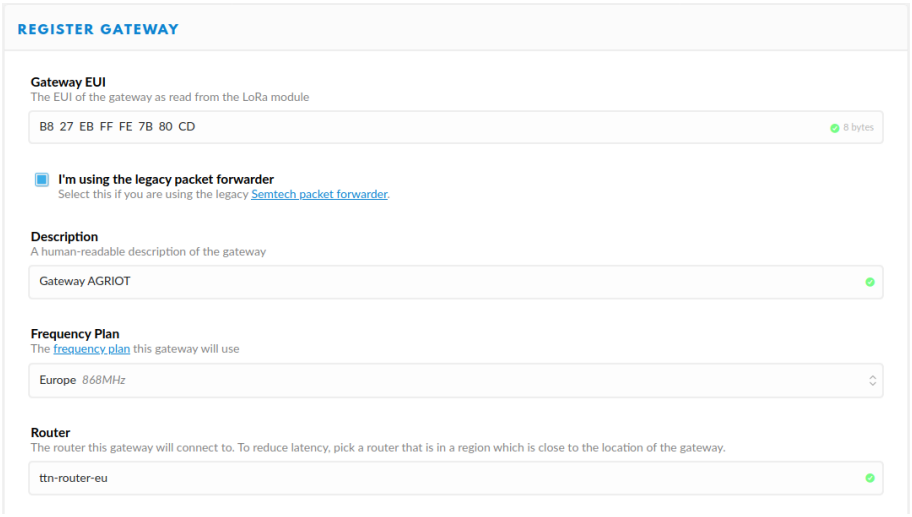


Figura 3.23: Registro del gateway en la red TTN.

3.2.2. Registro de dispositivos en la red TTN

Con el objetivo de habilitar la comunicación de los nodos con el gateway, en primer lugar es necesario crear una aplicación. Para ello, desde el apartado "Applications" de la plataforma TTN, se introduce el ID y descripción de la aplicación tal y como se muestra en la Figura 3.24.

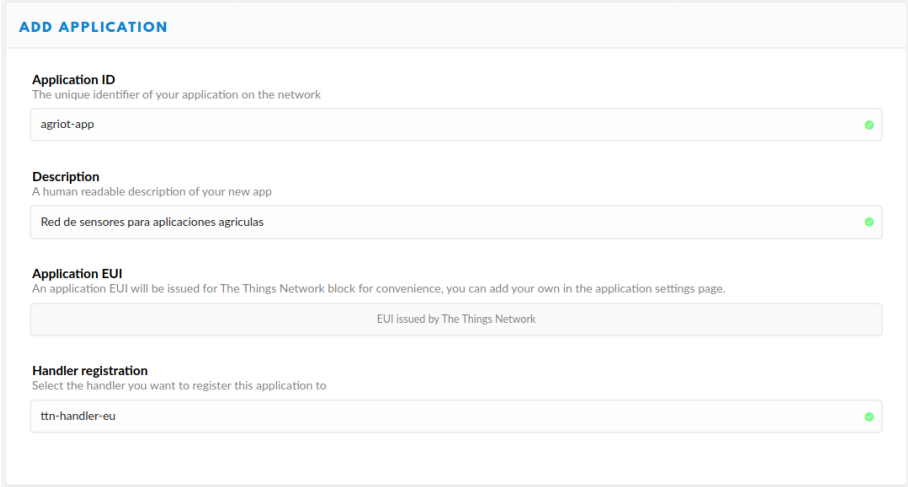


Figura 3.24: Registro de la aplicación en la red TTN.

El siguiente paso consiste en registrar el nodo sensor en la aplicación ya creada. El nodo se identifica mediante el Device ID, que debe ser único dentro de la aplicación, y el Device EUI, que es el identificador único de dispositivo (ver Sección 3.1.8.4). Además se debe seleccionar OTAA como método de activación.

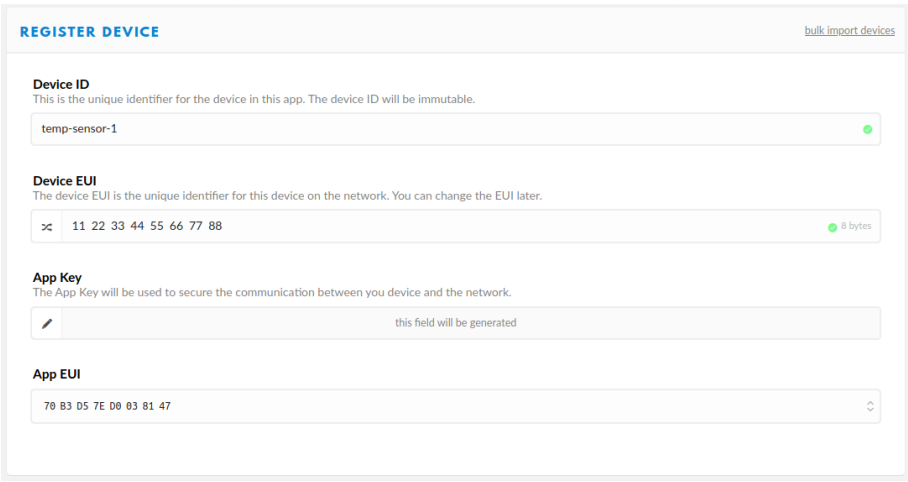


Figura 3.25: Registro de dispositivos en la red TTN.

Al realizar el registro se obtiene el **App EUI** y el **App KEY** (ver Figura 3.26), que son empleados para que el dispositivo pueda autenticarse en la aplicación, para ellos introducidos en el puerto serie del dispositivo los comandos E y K con sus respectivos identificadores.

- Application Eui: **E:70B3D57ED0038147**
- Application Key: **K:275EE0F36489D6AE679FE4827C140EBF**

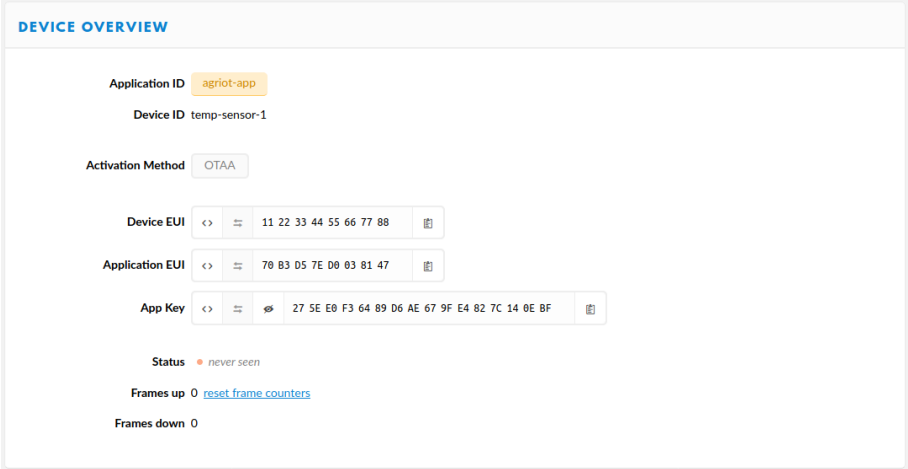


Figura 3.26: Información de activación del dispositivo.

En este punto, el dispositivo configurado ya es capaz de unirse a la red mediante el procedimiento de activación OTAA. Al recibir el mensaje de aceptación de la red, el nodo se encuentra listo para poder enviar o recibir paquetes desde o hacia el gateway.

En la Figura 3.27 se muestra el intercambio de mensajes entre el nodo y el servidor de red. El primer mensaje enviado por el nodo es para autenticarse en la red, posteriormente envía un mensaje sin payload y con acuse de recibo que es contestado por el servidor de red con un mensaje de enlace descendente. A partir del SNR de este mensaje, el nodo ajusta el factor de dispersión de los siguiente mensajes a transmitir (Tasa de Datos Adaptativa). Una vez se ha realizado este proceso, el nodo comienza la transmisión periódica de los datos del sensor.

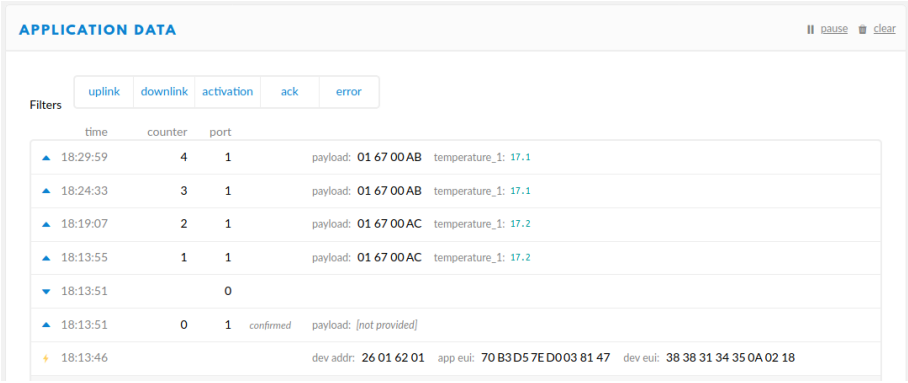


Figura 3.27: Intercambio de mensajes entre el nodo y el servidor de red.

Es necesario mencionar que al estar usando los servidores de The Things Network, el gateway necesita tener acceso a Internet ya que es en estos servidores donde se realiza la activación de los dispositivos registrados en una aplicación.

Para interpretar correctamente el payload desde la propia aplicación de The Things Networlk, en la configuración se debe indicar que el formato enviado por los dispositivos emplea Cayenne LPP (Figura 3.28).

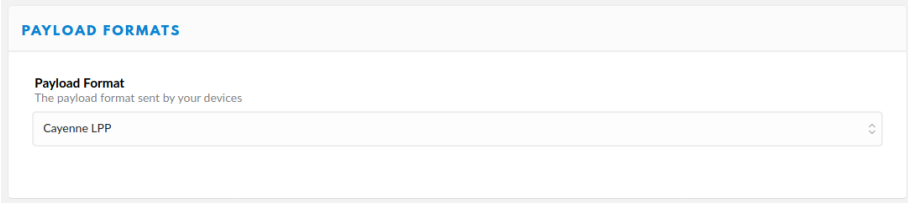


Figura 3.28: Formato del payload.

3.3. Plataforma

Como hemos comentado anteriormente, los servidores de TTN no almacenan los datos. Eso significa que necesitamos un método de extracción de datos, almacenamiento y visualización que nos permita su consulta y puesta en valor.

En esta sección se analizaran los diferentes elementos software que se han implementado a nivel de plataforma para poder presentar los datos de los sensores de forma visual.

3.3.1. Servidor - Google Cloud Platform

La ejecución de todo el software de plataforma se realiza en un servidor en red. Al tratarse de un sistema que, inicialmente, requiere de pocos recursos se ha optado por el uso de servidor virtual en vez de un servidor dedicado. Un servidor virtual es una partición dentro de un servidor físico dedicado, el cual permite compartir recursos entre diferentes máquinas virtuales. Por consiguiente, cada máquina virtual actúa como un ordenador independiente, sobre el que podremos instalar un sistema operativo con sus correspondientes aplicaciones y configuraciones.

Este paradigma permite desplegar aplicaciones webs a un costes muy bajos ya que no requiere tener un máquina física dedicada al 100 % para ejecutar aplicaciones que requieren muy pocos recursos. Otro factor clave es la escalabilidad, al permitir aumentar los recursos destinados al servidor virtual según se incrementan los necesidades.

Servicios como GCP, AWS (Amazon Web Services) o Microsoft Azure son algunas de las plataformas más empleadas a nivel mundial para el despliegue de soluciones web.

Google denomina **instancia** a una VM alojada en su infraestructura. Existen diferentes tipos de instancias en función del conjunto de recursos hardware asociados: memoria RAM del sistema, número de CPUs virtuales y capacidad del disco persistente. En la Tabla 3.29 podemos encontrar las categorías de tipos de instancias disponibles en GCP en función del uso.

E2 de uso general	N2, N2D y N1 de uso general	M2 y M1 Con optimización de memoria	C2 optimizada para procesamiento
Procesamiento diario a un costo menor	Equilibrio de precio y rendimiento en una amplia gama de formas de VM	Cargas de trabajo con memoria ultraalta	Rendimiento ultraalto para cargas de trabajo de procesamiento intensivo
• Servicio web • Entrega de aplicaciones • Aplicaciones para oficinas administrativas • Bases de datos pequeñas y medianas • Microservicios • Computadoras de escritorio virtuales • Entornos de desarrollo	• Servicio web • Entrega de aplicaciones • Aplicaciones para oficinas administrativas • Bases de datos medianas y grandes • Caché • Transmisión multimedia	• Bases de datos en memoria grandes, como SAP HANA • Análisis en memoria	• HPC • Automatización de diseño electrónico (EDA) • Videojuegos • Aplicaciones con un solo subproceso

Figura 3.29: Tipos de VM de GCP para diferentes cargas de trabajo.

Fuente Imagen: https://cloud.google.com/compute/docs/machine-types

Se ha optado por una máquina de tipo N1, en concreto la *f1-micro* que dispone de una única CPU virtual, que puede ocupar hasta el 20 % del tiempo de la CPU física. La memoria RAM es de 0,6GB y 10GB máximo de disco duro persistente. El sistema operativo elegido para despliegue de las aplicaciones es Ubuntu, en concreto la versión 20.04 LTS (Long Term Support). El coste de mantenimiento de esta máquina virtual oscila entre los 3-4€ mensuales.

Una vez creada la instancia, es importante asignar un IP estática para poder acceder al servidor sin necesidad de consultar la IP asignada en cada momento. Para ello escribimos Red de VPC (Virtual Private Cloud) en el buscador y entraremos en la sección del menú lateral izquierdo "Direcciones IP externas". Seleccionaremos nuestra instancia y cambiaremos su tipo a estática. Le ponemos un nombre identificativo a esta IP y aceptamos. En la Figura 3.30 se muestra el resultado tras configurar la IP externa estática.



Figura 3.30: Configuración estática IP de la instancia en GCP.

3.3.2. Almacenamiento de datos - InfluxDB

Como se ha comentado anteriormente, la base de datos elegida para el almacenamiento de los datos generados por los nodos ha sido InfluxDB. Es un servicio de código abierto que podemos instalar sin ningún coste en la VM.

El proceso de instalación y configuración ha sido extensamente detallado en el Apéndice B.2. Para entender el funcionamiento y estructura de datos de InfluxDB, a continuación se exponen algunos conceptos claves.

Con el servicio InfluxDB se ha creado una base de datos **"database"** llamada *agriot-app*. Dentro de la base de datos aparece el término **"measurement"**, que es conceptualmente similar a una tabla en una base de datos SQL (Structured Query Language). Actúa como un contenedor para las etiquetas **"tags"**, los campos **"fields"** y la columna de tiempo **"timestamp"**. El nombre de la measurement es un string que describe los datos que se almacenan en los campos asociados.

Al tratar con una base de datos de series temporales, todos los datos en InfluxDB tienen una columna de tiempo, que almacena el **timestamp** con la fecha y la hora en formato RFC3339 UTC.

Dentro de los datos almacenados encontramos dos grupos: **fields** y **tags**. Ambos son incluidos en cada nueva observación insertada pero tiene un tratamiento muy distinto.

- **Fields**, son obligatorios y contienen el valor de una variable observada en un tiempo determinado. En nuestro caso, será el valor del sensor (temperatura, humedad, etc...) e información asociada a la comunicación con el nodo (SNR y RSSI). Los valores de campos pueden ser strings, floats, integers o booleanos.

- **Tags**, son opcionales y su valor se almacena como cadena de texto. Es una buena idea hacer uso de ellas porque, a diferencia de los campos, las etiquetas están indexadas. Esto significa que las consultas sobre ellas son más rápidas por lo son ideales para almacenar metadatos de uso común. En nuestro caso, emplearemos las etiquetas para guardar el nombre del dispositivo (device_id) y el número de serie (hardware_serial).

time	temp	hum	snr	rssi	device_id	hardware_serial
2020-10-18T00:00:00Z	18.1		-51	5	temp-sensor-1	1605729039501
2020-10-18T00:01:00Z	17.5		-49	6	temp-sensor-1	1605729039501
2020-10-18T00:01:00Z		57.3	-48	5	hum-sensor-1	1605729039501
2020-10-18T00:01:00Z		57.4	-50	5	hum-sensor-2	1605729039320

Tabla 3.12: Ejemplo de datos almacenados en una "measurement".

Al tratarse de una base de datos no estructurada, pueden ser mezcladas observaciones con diferentes campos y etiquetas. Es decir, en la misma measurement, pueden existir datos de un sensor de temperatura y de un sensor de humedad. En la Tabla 3.12 se muestra un ejemplo de la estructura de datos en una measurement de InfluxDB.

3.3.3. Conexión con TTN - Node-red

La comunicación entre el servidor y la plataforma The Things Network se realiza mediante el protocolo MQTT. Es un protocolo de red abierto y ligero creado para la transmisión de datos orientado a sistemas de IoT. Funciona sobre TCP/IP siguiendo el patrón publicador-suscriptor, es decir, el cliente se suscribe a aquellos mensajes publicados por el servidor que son de interés. The Things Network dispone de su propio servidor MQTT para que podamos extraer fácilmente los datos enviados por los dispositivos [15].

Por otro lado, Node-RED es una herramienta de programación para conectar entre sí dispositivos hardware, diferentes API y servicios online. Es de código libre, está construida bajo el entorno Node.js y se encuadra en la familia de herramientas de programación basadas en flujos. Ofrece un modo de programación visual accesible desde el navegador pero también permite programar en javascript para el desarrollo de funciones específicas que no existan en forma de nodos. En el Apéndice B.1 puede ser consultado el proceso de instalación y configuración de Node-RED en el servidor.

En la Figura 3.31 se muestra la implementación realizada en Node-RED. El primer nodo "ttn-mqtt" realiza la conexión con el broker MQTT de The Things Network y se subscribe a los mensajes publicados en el topic. Para subscribirse al topic, es necesario autenticarse con las credenciales asociadas a la aplicación creada en The Things Network.

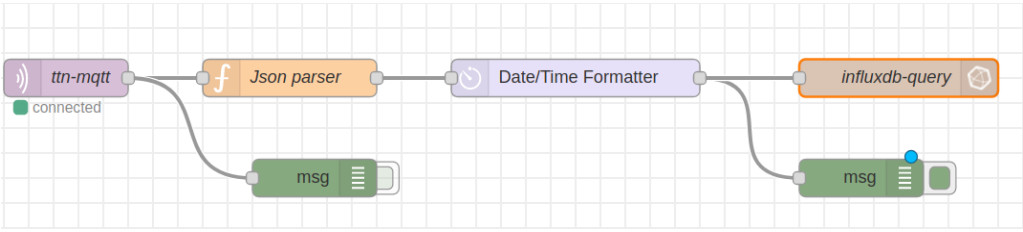


Figura 3.31: Flujo implementado en Node-RED

En la Lista 3.2 se muestra un ejemplo un ejemplo de un mensaje JSON devuelto por el broker MQTT de The Things Network. En su contenido podemos encontrar tanto datos del nodo sensor como metadatos asociados a la comunicación entre el nodo y el gateway.

La información localizada en el campo payload del JSON es la relacionada con el nodo sensor, de aquí nos interesa extraer el **app_id**, **dev_id**, **hardware_serial** y el **payload_fields**.

De los metadatos del JSON nos interesa extraer el campo **timestamp**, el **rssi** y el **snr** asociado al gateway que ha recibido los datos. En el caso de que varios gateways hayan recibido el mismo mensaje, nos quedaremos con la información del primer gateway que ha recibido el mensaje.

```
{"topic":"agriot-app/devices/temp-sensor-1/up",
  "payload":{
      "app_id":"agriot-app",
      "dev_id":"temp-sensor-1",
      "hardware_serial":"008592623E127EFA",
      "port":1,
      "counter":7380,
      "payload_raw":"AWi+",
      "payload_fields":{
          "temperature_1":19.5
      },
      "metadata":{
          "time":"2020-11-18T20:04:33.212172148Z",
          "frequency":868.5,
          "modulation":"LORA",
          "data_rate":"SF7BW125",
          "airtime":51456000,
          "coding_rate":"4/5",
          "gateways":[{
              "gtw_id":"eui-b827ebfffe942f9a",
              "timestamp":304165331,
              "time":"",
              "channel":2,
              "rssi":-47,
              "snr":8.2,
              "rf_chain":0}
              ]
          }
      }
  }
```

Lista 3.2: Mensaje en formato JSON recibido del broker MQTT de The Things Network.

Los dos siguientes nodos "Json parser" y "Data/Time Formatter" de la Figura 3.31 se encargan de extraer la información de interés del mensaje MQTT recibido. Además del valor medido por el sensor, en cada observación introducida en la base de datos se añade el RSSI, SNR de la señal recibida por el gateway. Para estructurar los datos el identificador de dispositivo y el hardware serial también son incorporados como etiquetas en cada nueva observación. Por último también se extrae la marca temporal de recepción del mensaje y

se convierte a formato epoch. En la Lista 3.3 se muestra el JSON resultante con el formato adecuado para insertar en InfluxDB.

```
{"measurement":"agriot-app",
  "payload":[{
    "temp":19.5,
    "rssi":-129,
    "snr":-8.2,
    "time":"1605729039501"
  },
  {
    "device_id":"temp-sensor-1",
    "hardware_serial":"008592623E127EFA"
  }]
}
```

Lista 3.3: JSON formateado para inyectar en InfluxDB.

Una vez extraída y formateada la información, el nodo "influxdb-query" (Figura 3.31) se encarga de inyectar cada nueva observación en la base de datos InfluxDB.

3.3.4. Visualización de datos - Grafana

Grafana es la plataforma web elegida para la visualización de los datos almacenados en la base de datos. Al igual que el resto de herramientas elegidas en este proyecto, es Open Source y puede ser descargada e instalada en el servidor sin ningún coste. Además de la creación de dashboards en los que visualizar los datos de diferentes formas, permite la gestión de roles de usuario, así como la creación de organizaciones en las que agrupar a los usuarios.

En el Apéndice B.3 se ha detallado el proceso de instalación y configuración de este servicio. Una vez realizado este proceso, se accede a Grafana vía web, bien desde la IP del servidor o desde el nombre de dominio asociado.

Una vez dentro, el primer paso consiste en configurar la fuente de datos, es decir, registrar la base de datos InfluxDB para que sea accesible desde Grafana (Figura 3.32). Como ambos servicios están en la misma máquina virtual, podemos emplear la dirección localhost para acceder a InfluxDB y el puerto 8086. Será necesario indicar el nombre de la base de datos, "agriot-app" así como el usuario de lectura y contraseña que se ha creado para esta base de datos (consultar Apéndice B.2).

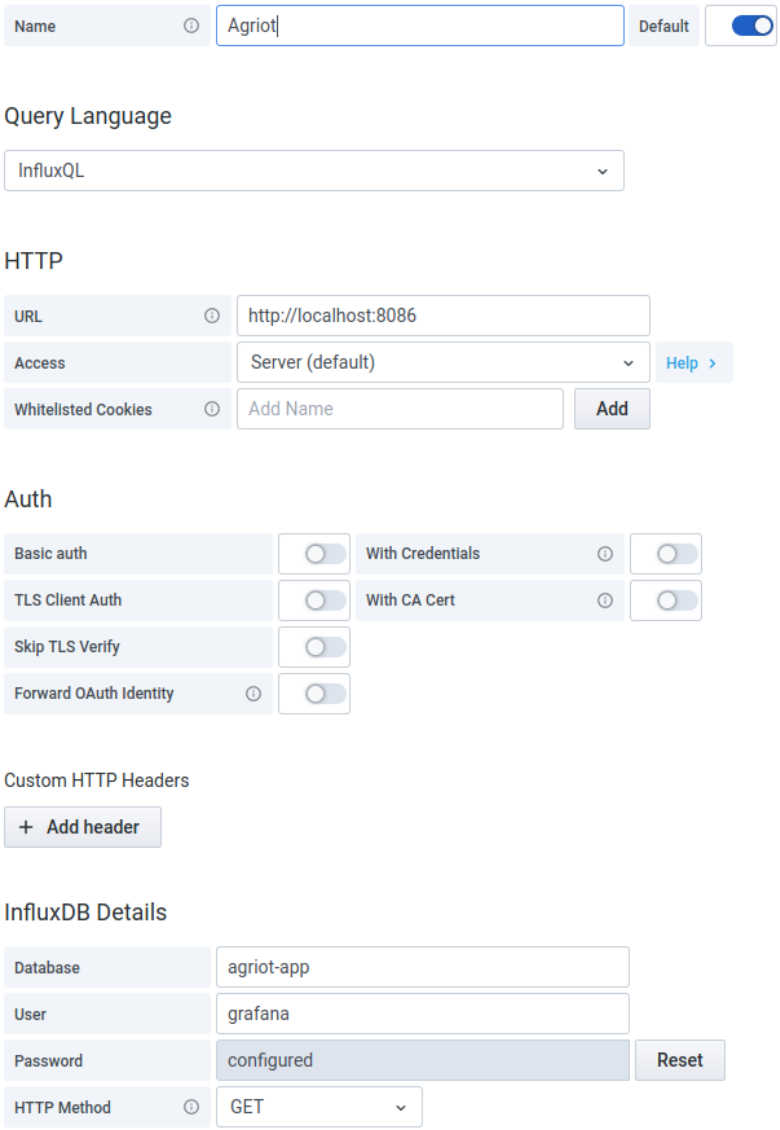


Figura 3.32: Panel de configuración bases de datos en Grafana.

Una vez añadida la fuente de datos, desde el apartado *Dashboards-Manage* podemos crear un panel de visualización de datos para un sensor o conjunto de sensores.

Desde este panel de configuración podemos generar la query que se hace a InfluxDB para obtener los datos. En la query se indica el nombre de la measurement, el Device ID o hardware serial (tags) del dispositivo deseado, así como el nombre del campo datos. También se indica el intervalo de tiempo y el tipo de agregado de datos que se va a obtener (mediana, media, máximo, mínimo, etc ...)

```
1 SELECT mean("temperature_1") FROM "agriot-app" WHERE ("device_id" = 'soil-
    temp-1') AND $timeFilter GROUP BY time($__interval) fill(null)
```

En este ejemplo de query se indica que el nombre de la measurement a consultar es "agriot-app", y se obtienen los datos de temperatura promediados del dispositivo cuyo device ID es "soil-temp-1". Gráficamente, la query es creada desde el panel de configuración tal y como se muestra en la Figura 3.33.

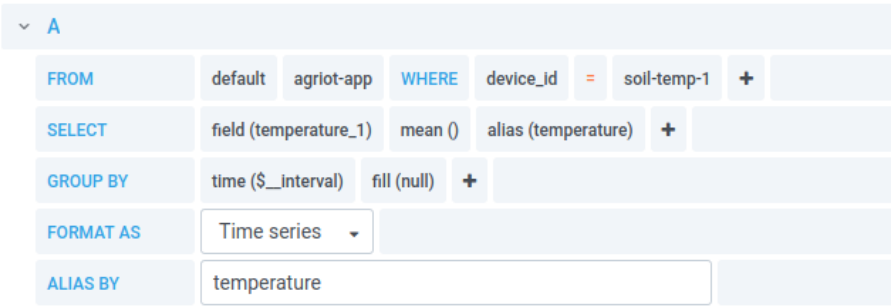


Figura 3.33: Creación de una query desde el panel de configuración.

En una misma query se pueden obtener los datos de diferentes sensores y campos de datos al mismo tiempo, siendo representados en la misma gráfica todos los datos.

Desde el panel de configuración, también se realiza la configuración visual, es decir,la selección del tipo de gráfica en la que se muestran los datos así como la configuración de los ejes, leyenda, umbrales y otros parámetros que puede ser modificados en las gráficas.

Capítulo 4

VALIDACIÓN Y CARACTERIZACIÓN DEL SISTEMA

En este tema se expondrán las diferentes pruebas que se han realizado sobre el sistema, con el objetivo de validar y caracterizar el comportamiento de sus diferentes componentes. El consumo energético de los nodos sensores, el alcance máximo de comunicación o la fiabilidad de los sensores son características que van a ser evaluadas.

4.1. Consumo energético

Para caracterizar el consumo de energía se ha eliminado el convertidor buck ya que este elemento está pensado para ser empleado en dispositivos alimentados de forma continua por cable y una tensión superior a 5 V.

La herramienta empleada para medir la corriente y el consumo del dispositivo es el kit Power Profiler desarrollado por la empresa Nordic Semiconductor. La placa de medición de consumo funciona junto con el kit de desarrollo nRF51DK que permite monitorizar el consumo en tiempo real desde un ordenador. Esta herramienta permite alimentar directamente el dispositivo a 3.3 V mientras monitoriza la corriente consumida por el sensor.

En primer lugar vamos a realizar una medición del consumo sin realizar ningún tipo de optimización, es decir, el nodo está siempre encendido y en ningún momento entra en bajo consumo. En la Figura 4.1, se observa que su consumo promedio, sin medir ni transmitir datos, es de 4.71 mA. Es un consumo muy elevado si pensamos en un dispositivo de bajo consumo conectado a una batería.

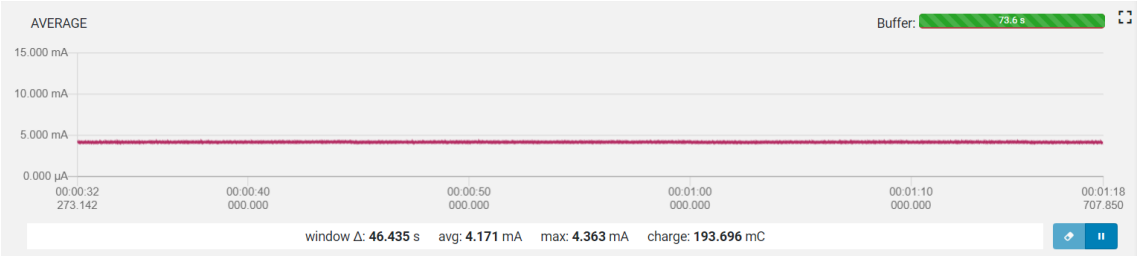


Figura 4.1: Consumo del dispositivo sin recurrir a modos de bajo potencia.

Para reducir el consumo es necesario poner el microcontrolador en un modo de baja potencia durante los periodos de espera, es decir, siempre que no esté realizando una medición o transmitiendo/recibiendo datos. En la Figura 4.2 se muestran todos los modos de bajo consumo disponibles en el microcontrolador Atmega328P.

El modo Power-down es el que apaga el mayor número de periféricos del microcontrolador y, por lo tanto, es el que logra un menor consumo. El microcontrolador permanece en modo sleep durante un tiempo configurable por el WDT (Watchdog Timer), que es la fuente empleada para despertarlo. El tiempo máximo de configuración del WDT es de 8s, esto implica que forzosamente el microcontrolador deberá despertar pasado ese tiempo, posteriormente puede volver a esta sleep si no hay ninguna tarea a la que atender.

Recordemos que en el microcontrolador estamos empleando un entorno en tiempo de ejecución (Sección 3.1.8.2) sobre el que se planifican las tareas o trabajos: adquisición del sensor, transmisión, etc. Para la correcta temporización de las tareas es importante tener una referencia que permita contar tiempo (ticks de sistema). De este modo, cuando el microcontrolador está despierto, esta referencia es generada por el Timer2, sin embargo, como se observa en la Figura 4.2 en el modo Power-down dicho timer está desactivado.

Para usar el modo Power-down ha sido necesario modificar la librería LMIC para que los ticks de sistema sean incrementados de forma proporcional al tiempo que el microcontrolador está en sleep, es decir, el tiempo que se ha configurado para el WDT. Una vez despierta, se actualizan los ticks de sistema y se ejecuta el entorno de tiempo de ejecución, si no hay ninguna tarea a la que atender en los próximos 8 segundos, el microcontrolador vuelve a modo Power-down.

	Active Clock Domains					Oscillators		Wake-up Sources							
Sleep Mode	clk_{CPU}	clk_{FLASH}	clk_{IO}	clk_{ADC}	clk_{ASY}	Main Clock Source Enabled	Timer Oscillator Enabled	INT1, INT0 and Pin Change	TWI Address Match	Timer2	SPM/EEPROM Ready	ADC	WDT	Other/O	Software BOD Disable
Idle			X	X	X	X	X(2)	X	X	X	X	X	X	X	
ADC noise Reduction				X	X	X	X(2)	X(3)	X	X(2)	X	X	X		
Power-down								X(3)	X				X		X
Power-save					X		X(2)	X(3)	X	X			X		X
Standby(1)						X		X(3)	X				X		X
Extended Standby					X(2)	X	X(2)	X(3)	X	X			X		X

Figura 4.2: Modos bajo consumo disponibles en el microcontrolador atmega328P.

En la Figura 4.3 se puede observar el consumo cuando el sistema está en modo Power-down. Los picos de corriente corresponden al momento de despertado del microcontrolador debido al WDT, en este caso configurado para que sea cada 8 segundos. Debido a la corta duración del evento, la herramienta de consumo no es capaz de muestrearlo correctamente, devolviendo un valor variable de consumo.

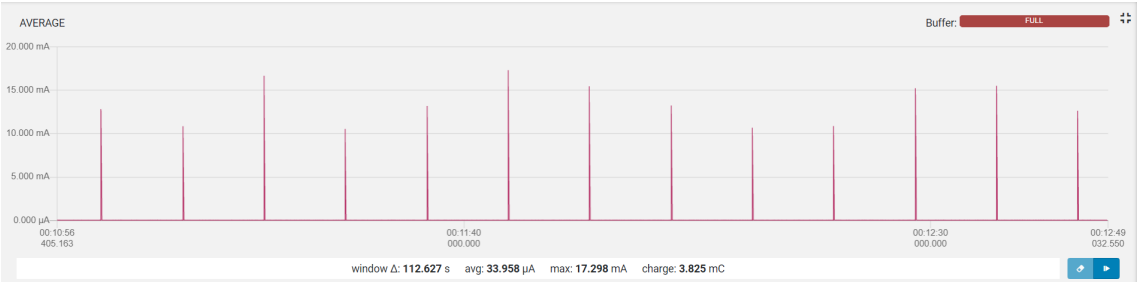


Figura 4.3: Consumo base del dispositivo en ausencia de tareas.

Para reducir aún más el consumo, evitando tiempos de espera activa, se ha implementado una configuración dinámica del tiempo en Power-down. Para ello, el WDT se configura con valores de 1, 2, 4 u 8 segundos en función del tiempo restante para la ejecución de la siguiente tarea. Es decir, si la próxima tarea se ejecuta en 11,5 segundos, primero se realizará un periodo en Power-down de 8 segundos, seguido de 2 segundos y otro de 1 segundo, los 0,5 segundos restantes antes de la ejecución de la tarea será una espera en modo activo.

En la Figura 4.4 se observa cómo al aproximarse a la ejecución de la tarea de medición, en el minuto 10:40, el tiempo en Power-down se va reduciendo, de los 8 segundos iniciales, pasa a 4 segundos y posteriormente a 2 segundos antes de entrar en modo normal. En la Figura 4.6 se muestra el evento de medición con más detalle.

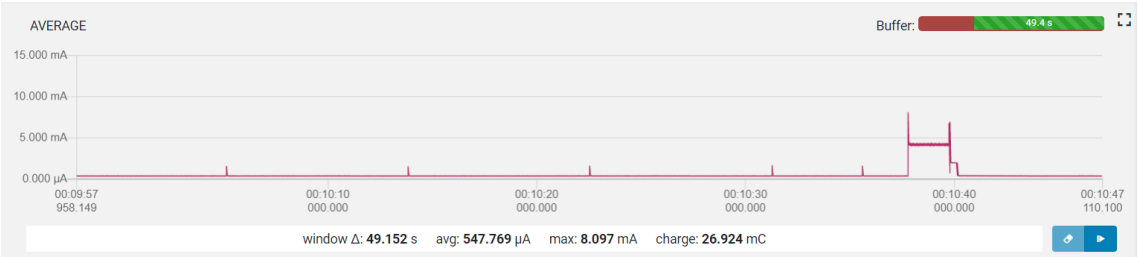


Figura 4.4: Transición de modo bajo consumo a normal para ejecutar una tarea de medición.

Vamos a evaluar también las tareas de alto consumo en el dispositivo, y a partir de estos datos, evaluar el consumo medio en una hora. En la Figura 4.5 se ha capturado la tarea de transmisión seguido de las dos ventanas de recepción del protocolo LoRaWAN. En un tiempo de 2,281 s el consumo promedio ha sido de 7,314 mA.

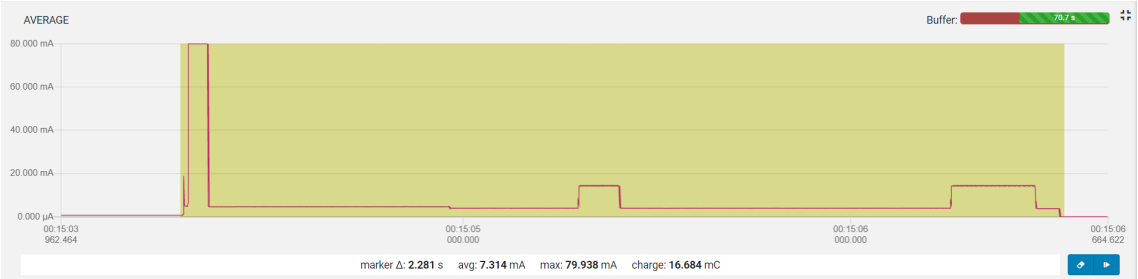


Figura 4.5: Consumo del dispositivo durante la transmisión de datos.

Por otro lado, en la Figura 4.6 se ha capturado un periodo de medición del sensor DS18B20, en este caso durante un tiempo de 2,296 s el consumo promedio ha sido de 3,854 mA.

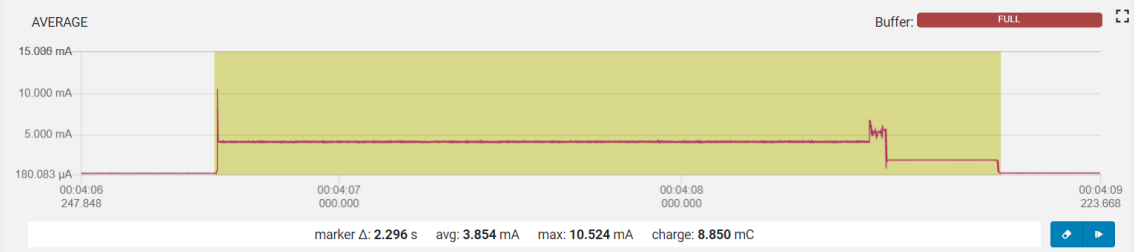


Figura 4.6: Consumo del dispositivo durante la adquisición del sensor DS18B20.

En la Tabla 4.1 se recogen los datos de consumo y duración de cada tarea, a partir de los cuales se ha realizado la estimación para calcular el consumo promedio del dispositivo durante una hora.

La tarea encargada de la comunicación LoRa es la que tiene un consumo puntual mayor, llegando a los 80 mA en el momento de la transmisión. Por otro lado, su corta duración y baja frecuencia de ocurrencia hace que su consumo acumulado en una hora sea del mismo orden que el consumo del sensor y el consumo en modo Power-down (Figura 4.7).

Nombre de la tarea	Tareas por hora	Duración de la tarea	Consumo de la tarea	Duración por hora	Carga eléctrica
LoRa Tx/Rx	12	2,8 s	7,314 mA	33,6 s	0,068 mAh
Adquisición sensor	12	2,3 s	3,854 mA	27,6 s	0,030 mAh
Espera			0,034 mA	3511,2 s	0,033 mAh
Total				3600 s	**0,131 mAh**

Tabla 4.1: Tareas y consumo asociado

El dispositivo alimentado a 3,3 V tiene un consumo instantáneo promedio de 0,131 mA, lo que en términos de potencia implica 0,432 mW.

Una pila de tipo AAA tiene una capacidad de entre 1,3 y 1,8 Wh. Tomando un valor intermedio de 1,5 Wh, idealmente el dispositivo podría funcionar durante 3472 h. Considerando las pérdidas en los elementos para convertir los niveles de tensión y también las pérdidas por autodescarga de la propia pila, podríamos alcanzar un duración de 4 meses de autonomía.

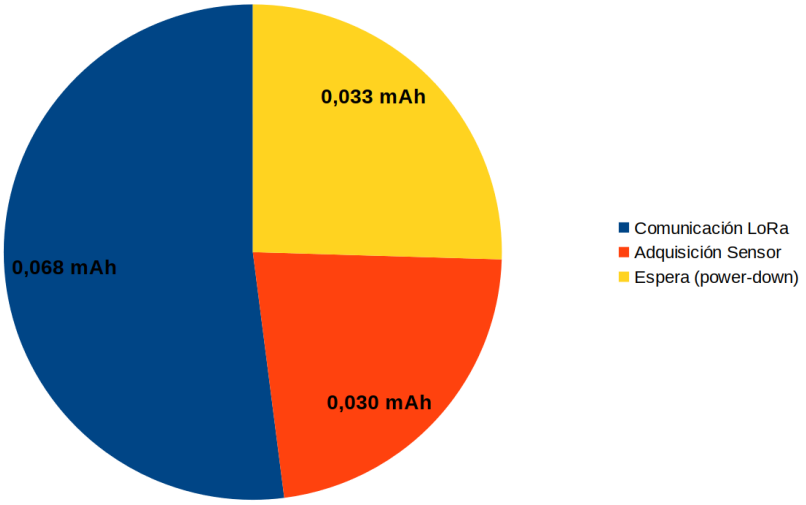


Figura 4.7: Distribución de consumo promedio durante una hora.

4.2. Pruebas de nodos sensores

Como paso previo al despliegue de los dispositivo en el entorno de producción, se ha realizado la validación del sistema en unas condiciones de operación lo más cercanas a la reales. Para ello se han montado tres nodos sensores, uno de cada tipo, y se han mantenido funcionando durante una semana. El objetivo, una vez analizados los resultados, será validar los siguientes puntos:

- Recepción continuada de datos y con un intervalo constante entre muestras consecutivas.
- Sentido físico de los datos recibidos y coherencia con el tiempo real durante los días de la prueba.
- Funcionamiento adecuado de todos los elementos del sistema: gateway, base de datos, visualizador, etc.

La electrónica de cada nodo sensor se ha introducido de un caja eléctrica de dimensiones 12x8x5 cm, con protección IP65. A través de cuatro prensaestopas con que cuenta la caja y que aseguran la estanqueidad del dispositivo entra un cable de alimentación y otro para la comunicación con el correspondiente sensor. Por otro lado, la antena se monta sobre la caja pero en el lado exterior. El gateway se ha situación a una distancia de 3 m, suficiente para que la transmisión inalámbrica de los datos no suponga ningún problema en esta prueba. Además, se han configurado los nodos para que el periodo de subida entre muestra y muestra sea de 3 minutos, un tiempo menor al empleado en la prueba de consumo que fue de 5 minutos.

En las Figura 4.8, 4.9 y 4.10, se muestran los tres nodos con los que se ha realizado la validación.



Figura 4.8: Nodo sensor montado con la sonda de temperatura del suelo, DS18B20.

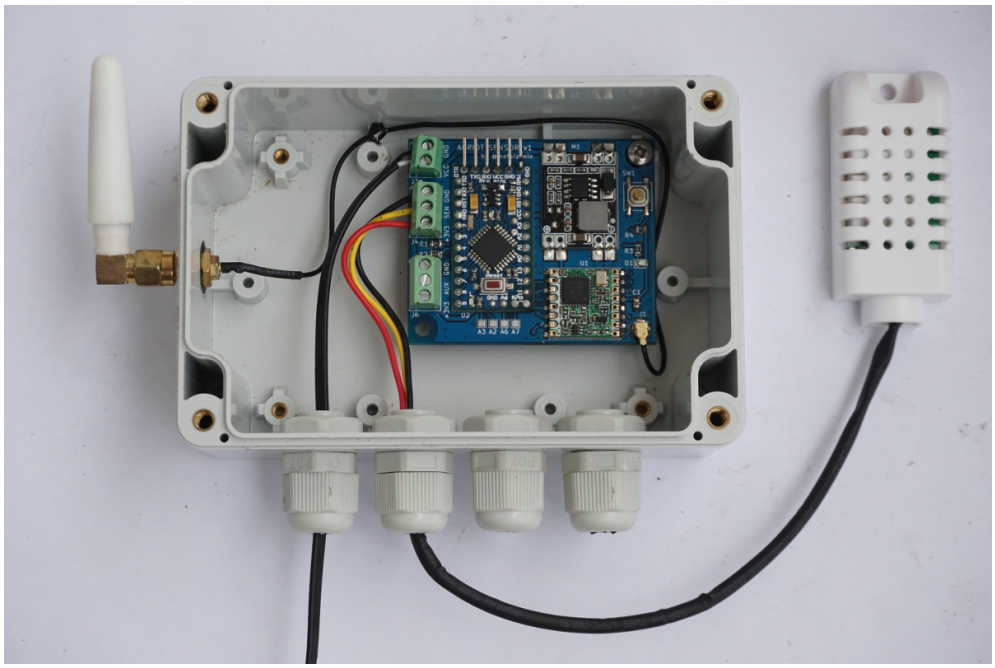


Figura 4.9: Nodo sensor montado con la sonda de temperatura y humedad del aire, AM2302.

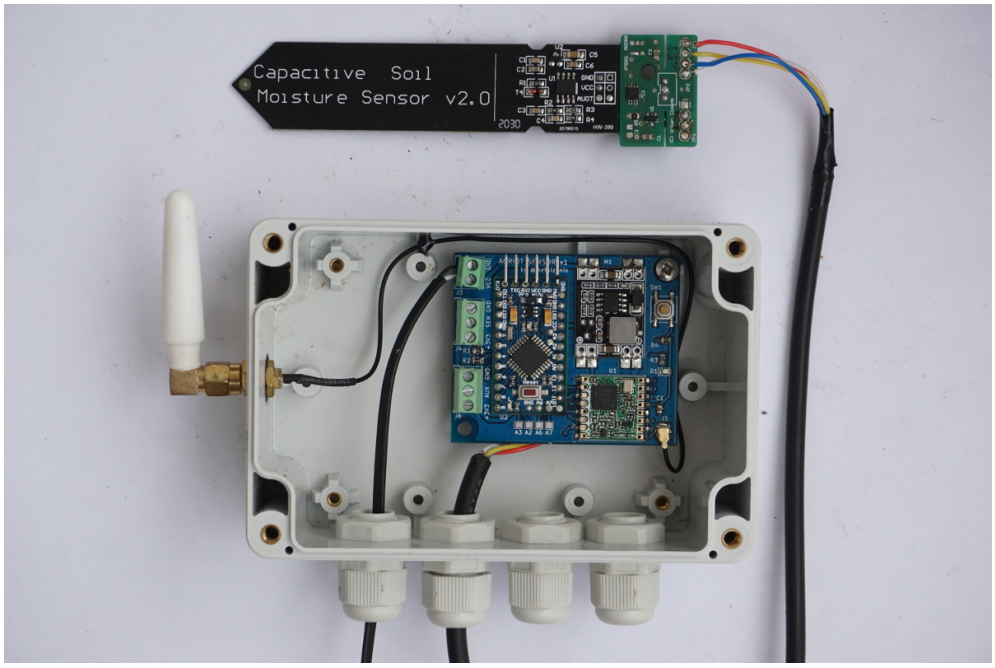


Figura 4.10: Nodo sensor montado con la sonda de humedad del suelo.

En la Figura 4.11 se representan el número de muestras totales recibidas en la base de datos por cada nodo entre los 7 días de duración de la prueba. En intervalo de subida configurado, de 3 minutos, no se ha considerado el tiempo de adquisición del sensor ni el tiempo de transmisión de los datos, tampoco el error debido a precisión del oscilador externo que usa el microcontrolador como fuente de reloj. Por esta razón, al analizar el tiempo real promedio entre muestras consecutivas, recibidas en la base de datos, es de 3 minutos y 12 segundos.

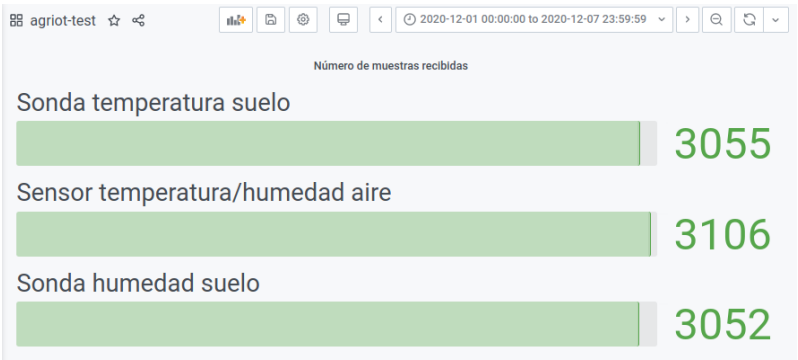


Figura 4.11: Número de muestras recibidas por cada nodo.

Conociendo el intervalo de recepción, podemos calcular el número de muestras totales a recibir durante 7 días: $60\ m * 24\ h * 7\ d/3,2\ m = 3150$. Para el peor de los casos, el nodo con la sonda de humedad del suelo, se han recibido 3052 muestras, es decir, el 97,8 % de

las muestras esperadas. Para el sensor de temperatura y humedad del aire se ha recibido correctamente el 98,6 %.

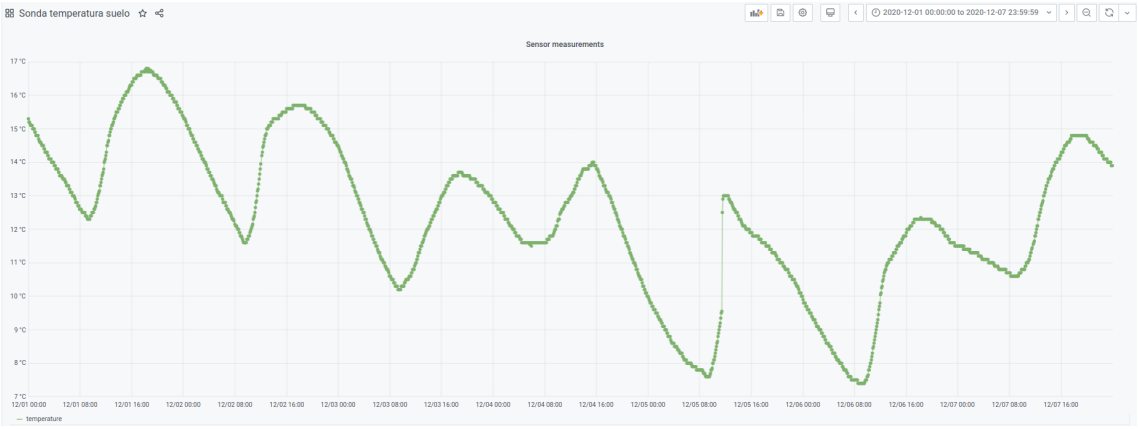


Figura 4.12: Datos de la sonda de temperatura del suelo.

En la Figura 4.12 se muestran los valores medidos por la sonda de temperatura del suelo durante los 7 días de la prueba. Podemos observar cómo durante la mañana del día 5, se produce una subida brusca de la temperatura que coincide con el momento de riego del suelo donde estaba localizado el sensor (Figura 4.13).

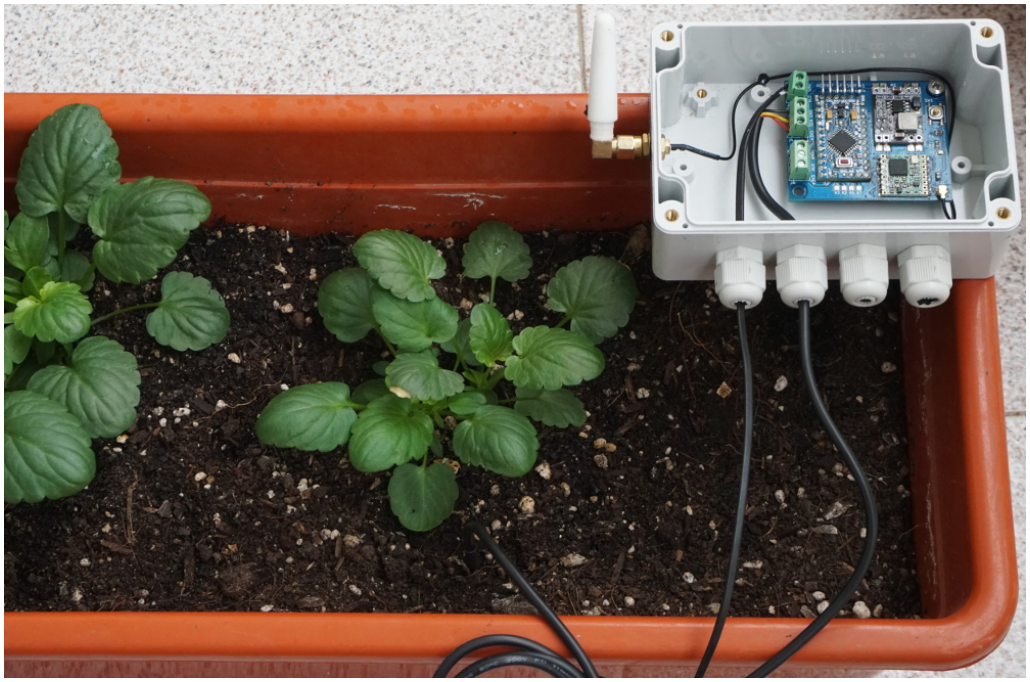


Figura 4.13: Sonda de temperatura en el entorno de prueba.

A excepción del momento donde se ha realizado el riego, el resto de la señal presenta una evolución constante y suave, los valores medidos están entre los rango esperados y su dinámica está correlacionada con la temperatura ambiente obtenida de una estación meteorológica cercana (Figura 4.14).

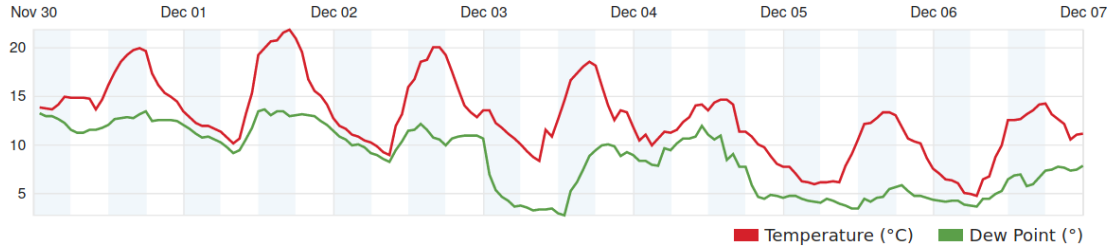


Figura 4.14: Datos de temperatura y punto de rocío en el entorno donde se han realizado las pruebas.

Por otro lado, en la Figura 4.15 se muestran los resultados obtenidos por el sensor AM2302, temperatura (verde) y humedad (amarillo). Podemos observar cómo durante las horas del medio días, la temperatura sube de forma brusca, próxima a los 30 ºC el día 1/12. Si comparamos con los datos de temperatura de una estación meteorológica cercana, Figura 4.14, se observa que ese día la temperatura máxima es de 23 ºC. Esta diferencia se debe a que el nodo sensor recibe la radiación solar de forma directa, produciendo un calentamiento en el material que hace incrementar la temperatura local del sensor.

Para evitar que el sensor se vea afectado por la radiación directa del sol se ha añadido un escudo de radiación que, además, lo protege de la entrada de agua. Consultar el Apéndice A.3 para más información.



Figura 4.15: Datos de la sonda de temperatura y humedad del aire.

Por último, en la Figura 4.16 se muestran los resultados de la sonda de humedad del suelo. Durante el periodo de la prueba, el suelo donde se localiza el sensor (Figura 4.17) no ha sido regado en ningún momento. Sin embargo, en la gráfica se observa un incremento y decremento diario del valor de la humedad, mostrando una correlación muy alta con la temperatura ambiente medida. En vista de estos resultados poco fiables se ha decido descartar el uso de este sensor en el entorno de producción.

Podría ser tema interesante estudiar en detalle la señal del sensor y tratar de mejorar su selectividad mediante técnicas de compensación, sin embargo es algo que se escapa del alcance de este TFM y que podría ser abordado en futuros trabajos.

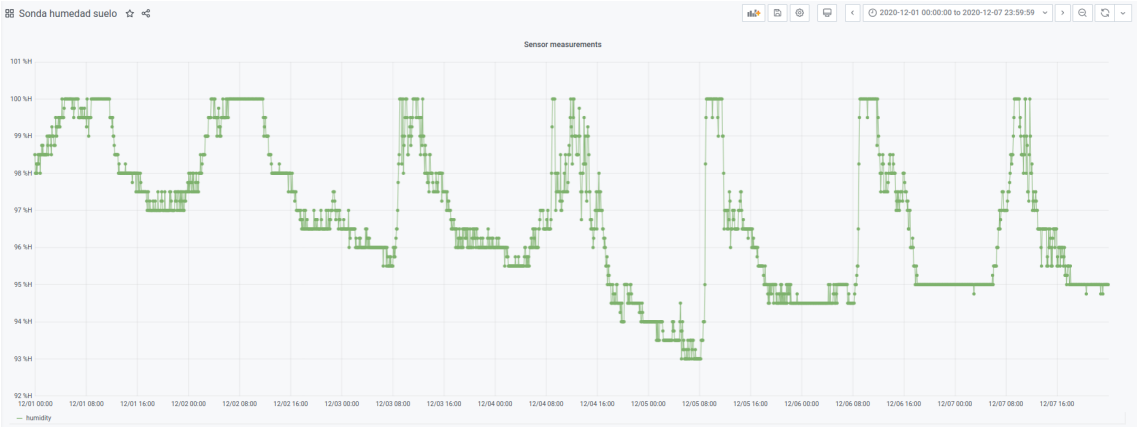


Figura 4.16: Datos de la sonda de humedad del suelo.

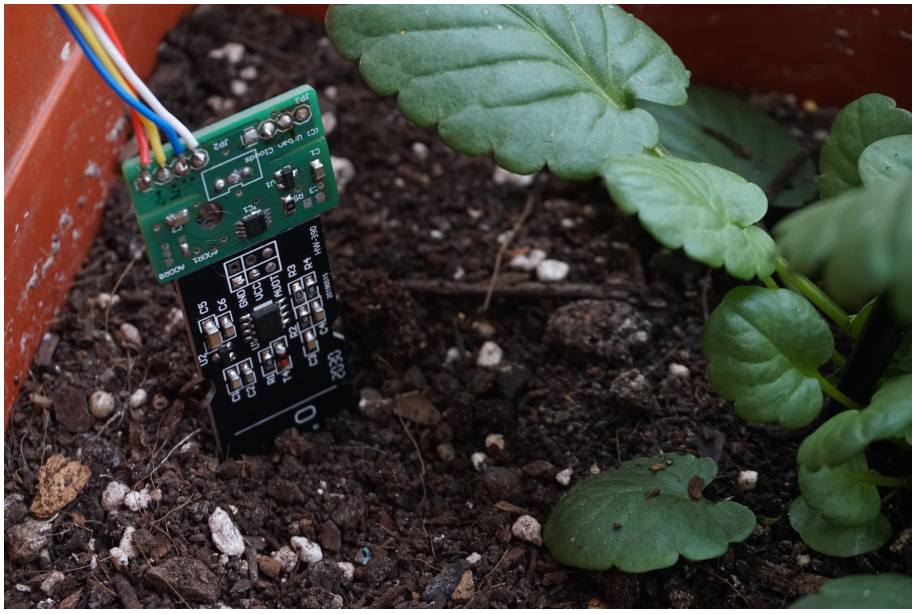


Figura 4.17: Sonda de humedad en el entorno de prueba.

4.3. Cobertura de la red

Otro aspecto importante para la validación del sistema es la caracterización empírica del rango de alcance de la red. Con este objetivo se ha creado un nodo al que se han conectado un receptor GPS, de forma que al transmitir su localización usando LoRaWAN podemos establecer de forma sencilla cuál es el rango real de cobertura del gateway.

El dispositivo GPS empleado es el U-blox Neo 6M montado sobre el módulo ATK-NEO-6M-v2.3, cuyas características se muestras en la Tabla 4.2.

Tensión de alimentación:	3,3 V - 5 V
Consumo de corriente :	45 mA (durante recepción)
Interfaz :	Serial UART
Sensibilidad de recepción:	-161 dBm
Frecuencia de refresco:	5 Hz max.
Protocolo:	NMEA / UBX Binary
Tiempo de arranque:	Cold Start: 27 s min.
	Warm Start: 27 s
	Hot Start: 1 s
Rango de temperatura:	-40 °C - 85 °C
Tamaño:	25,5*31 mm

Tabla 4.2: Características del Módulo GPS ATK-NEO-6M.

El módulo GPS se comunica por una interfaz UART. Dado que el microcontrolador ATmega328p solo dispone de un periférico UART que ya ha sido empleado para la interfaz de línea de comandos (Sección 3.1.8.4), se ha utilizado un puerto serie emulado por software. Mediante la librería SoftwareSerial del framework Arduino, podemos usar los pines 3 y 4 como puerto serie software.

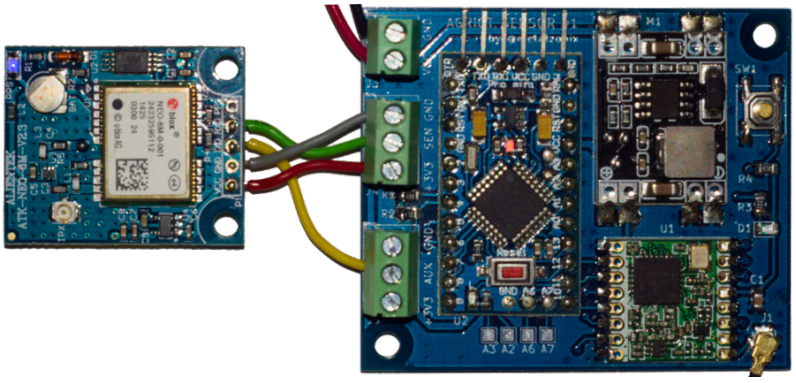


Figura 4.18: Módulo GPS conectado al nodo sensor.

Los datos son transmitidos, recibimos y almacenados en la base de datos empleando la misma infraestructura que es empleada por el resto de nodos sensores. Para visualizar los datos de geolocalización en Grafana, se ha instalado el plugin TrackMap Panel, que nos permite marcar sobre un mapa el histórico de localizaciones así como ver en cada punto otros valores asociados, en nuestro caso RSSI y SNR de la señal. En la Figura 4.19 se muestra un recorrido realizado a pie y portando el nodo.



Figura 4.19: Módulo GPS conectado al nodo sensor.

El máximo alcance ha sido de 700 m, al tratarse de un entorno urbano y estar el gateway en el interior de una vivienda. Es un rango que será sustancialmente mayor en campo abierto y con visibilidad directa del gateway.

Con esta prueba se ha podido validar también el mecanismo de Tasa de Datos Adaptativa (ADR). Según aumenta la distancia con el gateway y disminuye el RSSI, automáticamente el nodo sensor aumenta el factor de dispersión (SF) de la transmisión para evitar un disminución drástica del SNR.

Capítulo 5

DESPLIEGUE Y COSTE DEL SISTEMA

5.1. Despliegue del sistema

El despliegue del sistema se ha realizado en tres invernaderos contiguos, que forman parte de la misma explotación agrícola, y cubren una superficie aproximada de 30.000 m^2 localizados en El Ejido, provincia de Almería.

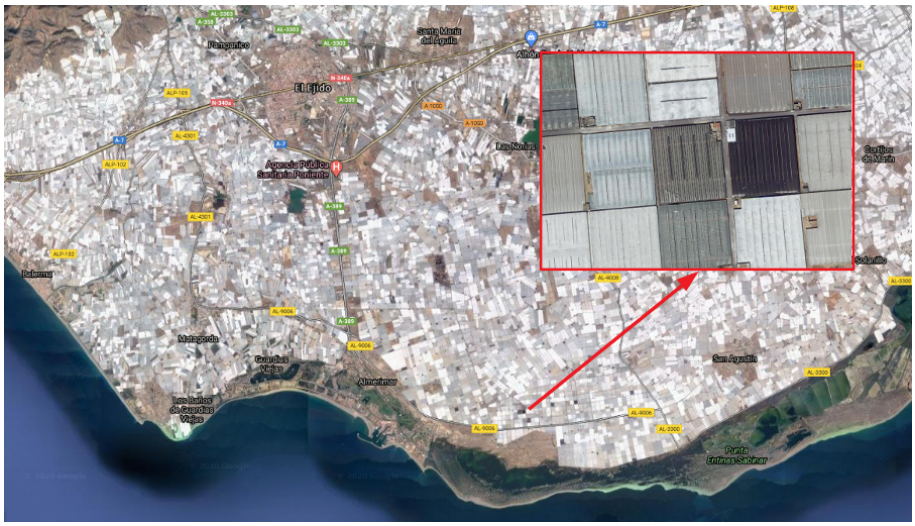


Figura 5.1: Zona de cultivo agrícola en invernaderos en El Ejido, Almería.

En total se han instalados 9 nodos sensores, 4 destinados a medir la temperatura del suelo y 5 para medir la temperatura y humedad del ambiente. En la Figura 5.2 se muestra la distribución que se ha realizado de los dispositivos.

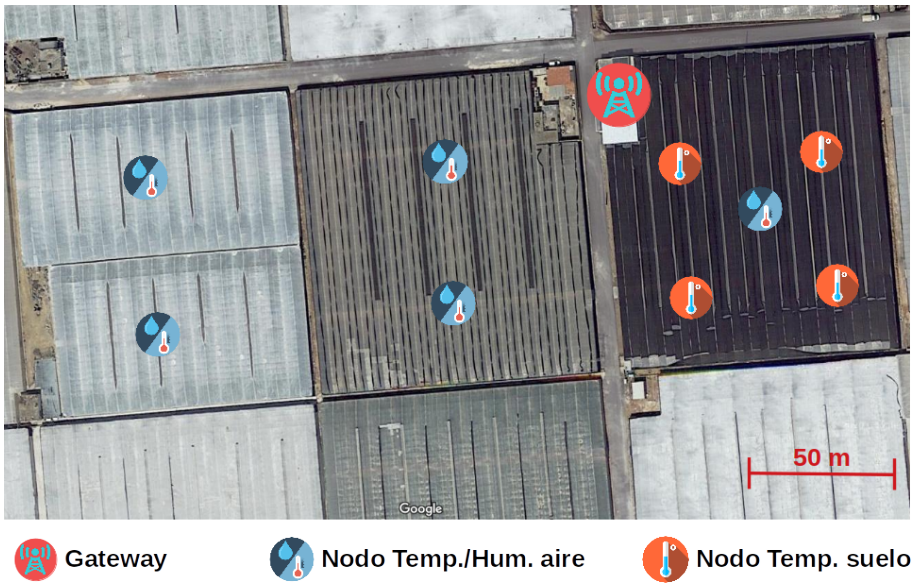


Figura 5.2: Nodo sensor de temperatura y humedad del aire.

Para identificar los invernaderos, éstos se han numerado del 1 al 3 empezando por el invernadero que se muestra en el lado derecho de la Figura 5.2.

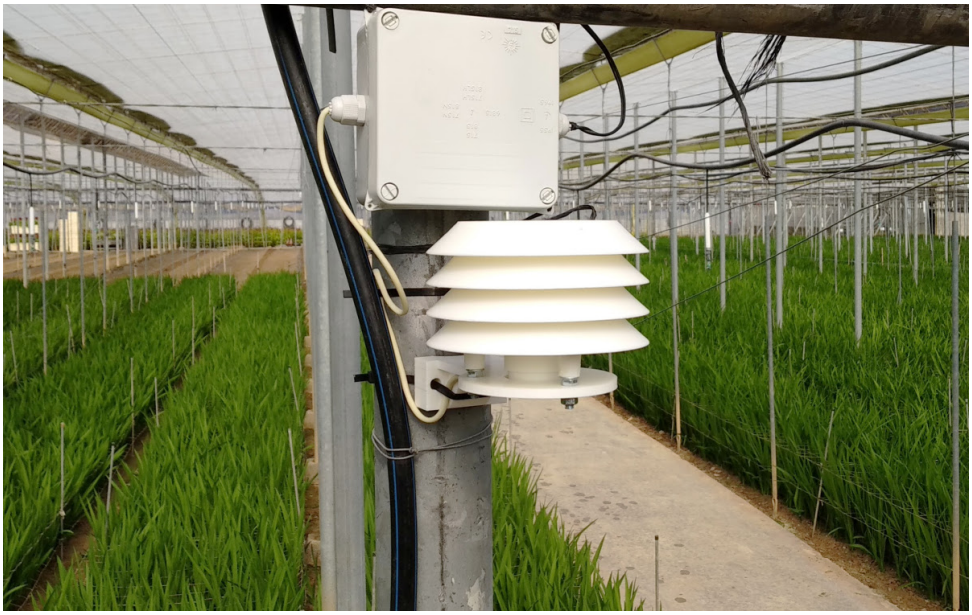


Figura 5.3: Localización de los nodos en el entorno de producción.

5.1.1. Mapa de cobertura

Se ha empleado el tracker GPS expuesto en la Sección 4.3 para realizar un mapa de cobertura de la instalación. El resultado se muestra en la Figura 5.4, en la que podemos observar cómo la zona de interés queda ampliamente cubierta por el gateway, con un RSSI siempre superior a -115 dBm. Además, el máximo alcance logrado ha sido de 2 km muy superior a los 250 m necesarios para cubrir la instalación.

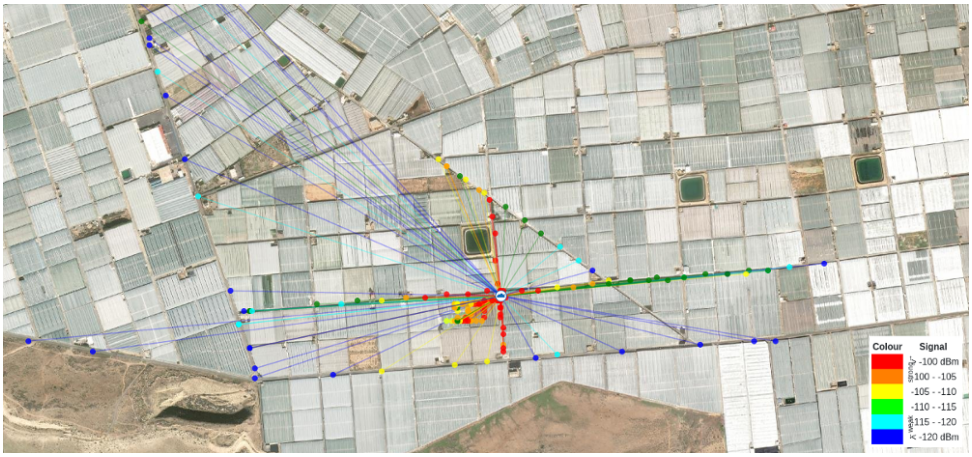


Figura 5.4: Mapa de cobertura.

5.1.2. Dashboard de visualización.

Para cada invernadero se ha creado un dashboard en el que se muestran los datos de los sensores, se ha incluido un dial para mostrar el último dato recibido y un gráfica temporal en la que se puede ver la evolución de las mediciones durante el tiempo seleccionado.



Figura 5.5: Dashboard de visualización de los sensores del invernadero 1.

En la Figura 5.5 se muestra el dashboard creado para representar los datos de los sensores instalados en el invernadero 1, es decir, 4 sondas de temperatura del suelo y 1 sensor de temperatura y humedad del aire. Se han representado los datos recogidos durante una semana de funcionamiento. Por otra parte, En la Figura 5.6 se muestra el dashboard creado para el invernadero 2, en el que solo se han instalado 2 sensores para medir la temperatura y humedad del aire. El mismo tipo de dashboard ha sido empleado para el invernadero 3.

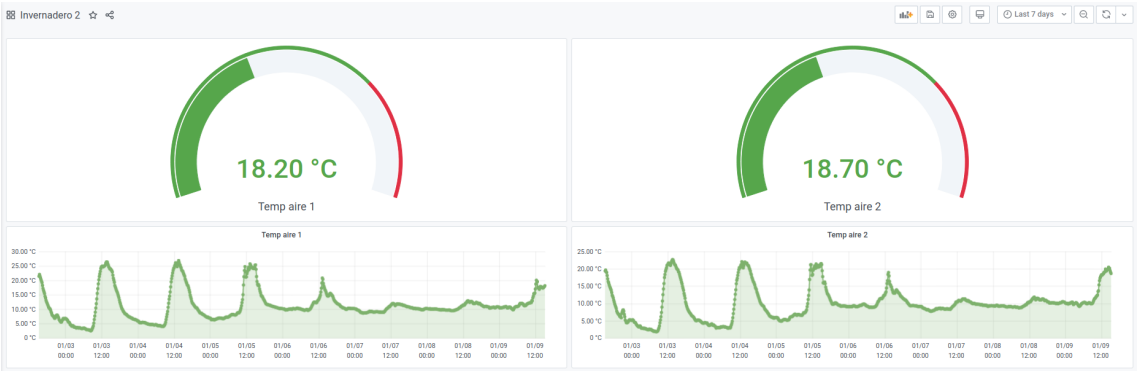


Figura 5.6: Dashboard de visualización de los sensores del invernadero 2.

5.2. Coste del desarrollo

Uno de los objetivos del proyecto, que se ha tenido en cuenta durante todo el proceso de desarrollo, ha sido el de reducir el coste del sistema al mínimo. Para lograr este objetivo, el realizar un diseño simple y ajustado a las necesidades ha sido clave, permitiendo cumplir con la funcionalidades deseadas, sin reducir la calidad.

Con el objetivo de conocer el coste final, por una lado se ha realizado una evaluación del coste material de todos los elementos que conforman el sistema instalado. Por otro lado también se ha evaluado el coste en horas de todas las tareas realizadas en desarrollo del sistema.

5.2.1. Coste material del sistema

En la Tabla 5.1 se muestra el BOM de la PCB diseña para los nodos sensores. Además del coste de todos los componentes que se montan en PCB, se ha incluido el coste fabricación y ensamblado de la PCB.

PCB + componentes			
Descripción	Cantidad	Coste unitario	Coste
LED	1	0,093	0,09 €
DC-DC_Buck_Step_Module	1	0,910	0,91 €
Resistor 2.2kR	1	0,009	0,01 €
SW_DIP_x01	1	0,032	0,03 €
Resistor 4.7k	3	0,009	0,03 €
Screw_Terminal_01x03	2	0,136	0,27 €
Screw_Terminal_01x02	1	0,113	0,11 €
ARDUINO_Pro-MiniALL	1	1,780	1,78 €
Conn_Coaxial	1	0,154	0,15 €
RFM95W-868S2	1	3,670	3,67 €
Capacitor 4.7uF/6.3v	1	0,012	0,01 €
PCB Nodo	1	0,79	0,79 €
Ensamblaje PCB	1	2,5	2,50 €
		Total	**10,36 €**

Tabla 5.1: BOM de la PCB del nodo sensor.

En la Tabla 5.2 y 5.3 se refleja el coste total de la PCB, con el nombre de "PCB Ensamblada + componentes".

En la Tabla 5.2 se han listado todos los componentes que forman un nodo sensor de temperatura del suelo así como el coste asociado a cada componente y el coste total del nodo. Este mismo esquema se ha repetido para el nodo sensor de temperatura y humedad del aire y para el gateway, recogidos en la Tabla 5.3 y 5.4, respectivamente.

Nodo Sensor temperatura suelo			
Descripción	Cantidad	Coste unitario	Coste
PCB Ensamblada + componentes	1	10,362	10,36 €
Antena 868 MHz 3dBi + Cable SMA/u.FL	1	2,086	2,09 €
Caja estanca IP65	1	1,15	1,15 €
Prensaestopas M16 IP68	2	0,9	1,80 €
Tornillos M3	2	0,04	0,08 €
Sensor DS18B20	1	1,92	1,92 €
		Total	**17,40 €**

Tabla 5.2: Componentes y coste del nodo de temperatura del suelo.

Nodo Sensor temperatura y humedad del aire			
Descripción	Cantidad	Coste unitario	Coste
PCB Ensamblada + componentes	1	10,362	10,36 €
Antena 868 MHz 3dBi + Cable SMA/u.FL	1	2,086	2,09 €
Caja estanca IP65	1	1,15	1,15 €
Prensaestopas M16 IP68	2	0,9	1,80 €
Tornillos M3	2	0,04	0,08 €
Sensor AM2302B	1	2,98	2,98 €
Escudo de radiacción 3D	1	24	24,00 €
		Total	**42,46 €**

Tabla 5.3: Componentes y coste del nodo de temperatura y humedad del aire.

Se puede observar cómo el coste del nodo de temperatura y humedad del aire es considerablemente superior al de temperatura del suelo, ésta diferencia se debe al empleo del escudo de radiación para proteger el sensor, que aun siendo impreso en 3D es el componente de mayor coste empleado en el nodo.

Gateway			
Descripción	Cantidad	Coste unitario	Coste
Raspberry Pi Model B	1	39,000	39,00 €
Placa Adaptación	1	6,000	6,00 €
Concentrador iC880A-SPI	1	119,000	119,00 €
Fuente alimentación 5V 2A	1	5,500	5,50 €
Cabler Ethernet	1	2,000	2,00 €
Caja registro eléctrico	1	2,350	2,35 €
Antena externa 3dBi	1	3,860	3,86 €
Cable SMA/u.FL	1	1,200	1,20 €
		Total	**178,91 €**

Tabla 5.4: Componentes y coste del gateway.

Como se muestra en la Tabla 5.5, el coste material del sistema instalado asciende a un total de 460,79 €.

Instalación			
Descripción	Cantidad	Coste unitario	Coste
Gateway	1	178,91 €	178,91 €
Nodo Sensor Temp. suelo	4	17,40 €	69,59 €
Nodo Sensor Temp/Hum. aire	5	42,46 €	212,29 €
		Total	**460,79 €**

Tabla 5.5: Coste material de la instalación realizada.

5.2.2. Coste del desarrollo

En total se han dedicado 228 h a la investigación y desarrollo del sistema. Para este cálculo no se ha tenido en cuenta el tiempo de redacción de la memoria ni la búsqueda de documentación asociada. En la Figura 5.7 se desglosan las tareas principales junto con el tiempo invertido para cada una de ellas. Las tareas en tonalidad azul son las relacionados con el desarrollo de los nodos y gateway, en tonalidad roja las relacionas con la plataforma y en amarillo el tiempo empleado para pruebas y validación global del sistema.

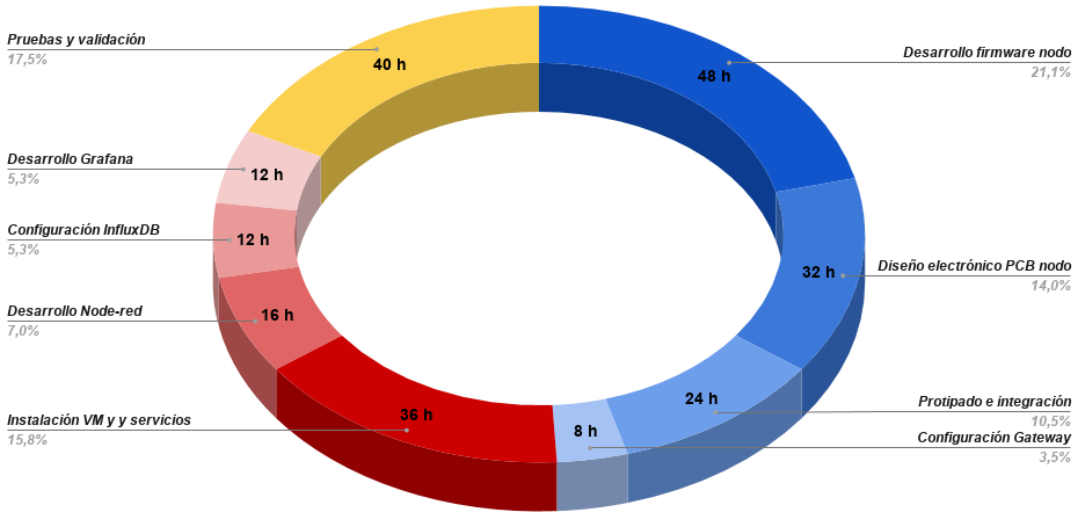


Figura 5.7: Desglose en tareas del tiempo dedicado a su realización.

Capítulo 6

CONCLUSIONES Y LÍNEAS FUTURAS

En este capítulo se muestra un resumen de los objetivos cumplidos en este proyecto así como posibles líneas futuras de investigación.

Se recuerda que el objetivo inicial definido en el Capítulo 1 era crear una plataforma IoT basada en tecnologías abiertas, de bajo coste, escalable, robusta y fácil de instalar en un entorno de producción. Con estas premisas y tras realizar un análisis del estado tecnológico actual de los sistemas IoT se ha podido definir, con buen criterio, las líneas básica del diseño. Este estudio también nos ha permitido conocer en detalle los diferentes elementos que forman un sistema IoT, así como la interrelación entre ellos.

Tras el desarrollo y despliegue del sistema en un entorno de producción, podemos resumir los aspecto claves en los siguientes puntos:

- **Nodos finales**: Por su importancia en el sistema y posibilidad de mejora, ha sido el elemento al que más tiempo se le ha dedicado para su desarrollo. Gracias a un diseño optimizado de la electrónica y el firmware se ha logrado un nodo de muy bajo coste, robusto y con flexibilidad para conectar otros sensores no contemplados en este proyecto.

- **Conectividad**: La decisión de desplegar una red LoRaWAN se ha visto como la más acertada para este caso de uso. Como se ha mostrado en el Capítulo 5, el rango de alcance cumple perfectamente con las expectativas, y al tratarse de un red propia no requiere de pago a ningún operador por su uso, lo que implica reducir costes. La tasa de transmisión de datos también se adapta perfectamente a nuestras necesidades actuales y futuras, permitiendo aumentar aún más la frecuencia de envío de paquetes o el uso de mensajes de enlace descendente para controlar el dispositivo. Tanto los módulos LoRa del nodo sensor como el gateway se han mostrado robustos durante las pruebas, teniendo ambos un coste contenido dentro del sistema.

- **Plataforma de gestión**: El uso de la plataforma TheThingsNetwork para la gestión de los dispositivo se ha mostrado eficaz y ha permitido un acceso rápido y simple a los datos enviados por los nodos sensores. Gracias al uso de protocolos abiertos como MQTT, hemos podido integrar TTN con la plataforma desplegada en nuestro propio servidor, que permite el almacenamiento de los datos y acceso posterior bajo demanda.

- **Interfaz de usuario**: El uso de servicios como Grafana o InfluxDB nos han permitido montar un plataforma escalable. Mediante Grafana y su conexión a InfluxDB podemos construir paneles de visualización avanzados, en los que mostrar la información de un modo transparente para el usuario final. Gracias al control de usuarios podemos personalizar los dashboards para cada usuario o grupo, así como gestionar el acceso a la base de dato. Los servicios empleados son de código abierto y gratuitos, por lo que el único pago a realizar es el mantenimiento de la infraestructura

cloud de la máquina virtual. Gracias al servicio cloud podemos escalar el sistema de plataforma en función de su uso, manteniendo siempre un coste lo más reducido posible.

Al tener como objetivo el despliegue en un entorno real de producción, se ha enfrentado al reto de diseñar un producto robusto y de fácil instalación. También, gracias al feedback real de los usuarios se ha podido comprobar su utilidad y mejorar de forma continua el sistema durante su desarrollo.

Ha quedado patente cómo es totalmente factible desarrollar un proyecto de esta envergadura empleando en su mayoría herramientas y servicios de código abierto, por ejemplo KiCad (diseño electrónico), Visual Studio Code (entorno de programación), Arduino (framework de programación), Git (control de versiones), The Thins Network Stack (servidor de red LoRaWAN), Grafana (visualizador de datos) o InfluxDB (base de datos). El uso gratuito de estas herramientas permite reducir considerablemente el coste de desarrollo y de mantenimiento del sistema. Por otro lado, el hecho de publicar abiertamente los desarrollos realizados en este proyecto, permitirá a otras personas o instituciones utilizar este trabajo, modificarlo, y sobre él aportar nuevas soluciones que cubran nuevos casos de uso.

Al romper las barreras económicas, este tipo de sistemas de monitorización pueden llegar a más explotaciones agrícolas, ayudando a los agricultores a generar un mayor conocimiento de su cultivo y que, a su vez tendrá una repercusión directa en la mejora de la productividad y reducción de los recursos empleados.

6.1. Líneas futuras

Este trabajo se han centrado en la monitorización de variables en entornos agrícolas. Sin embargo, el sistema y los nodos también podrían emplearse para el control de actuadores en el campo, por ejemplo, válvulas de riego, sistemas de ventilación, etc. De esta forma, mediante la implementación de un sistema de control sería posible mantener de forma automática los parámetros ambientales óptimos para el desarrollo del cultivo.

Además de su uso en entornos agrícolas, el sistema desarrollado podría ser adaptado para aplicaciones de monitorización en ganadería, flota de vehículos, recursos hídricos, ciudades inteligentes, etc. bastaría con integrar nuevos sensores que extiendan la funcionalidad para monitorizar otras variables hasta ahora no exploradas. También puede ser empleada como una plataforma de investigación y prototipado rápido por su facilidad para modificar e integrar nuevas funcionalidades.

Otro campo de exploración abierto con este proyecto, es el de investigación y desarrollo de un sensor de humedad del suelo. Vistos los malos resultados obtenidos del sensor de bajo coste y el precio tan elevado de los sensores comerciales, sería interesante abrir un línea de desarrollo en ese campo.

Bibliografía

[1] M. Centenaro, L. Vangelista, A. Zanella, and M. Zorzi, "Long-range communications in unlicensed bands: the rising stars in the IoT and smart city scenarios," *IEEE Wireless Communications*, vol. 23, no. 5, pp. 60–67, 2016.

[2] T. Attia, M. Heusse, B. Tourancheau, and A. Duda, "Experimental characterization of LoRaWAN link quality," in *2019 IEEE Global Communications Conference, GLOBECOM 2019 - Proceedings*, no. Section II, IEEE, 2019.

[3] U. Raza, P. Kulkarni, and M. Sooriyabandara, "Low Power Wide Area Networks: An Overview," *IEEE Communications Surveys Tutorials*, vol. 19, no. 2, pp. 855–873, 2017.

[4] Y. P. Wang, X. Lin, A. Adhikary, A. Grövlen, Y. Sui, Y. Blankenship, J. Bergman, and H. S. Razaghi, "A Primer on 3GPP Narrowband Internet of Things," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 117–123, 2017.

[5] K. Mikhaylov, Juha Petaejaejaervi, and T. Haenninen, "Analysis of Capacity and Scalability of the LoRa Low Power Wide Area Network Technology, European Wireless 2016 - 22th European Wireless Conference," in *European Wireless 2016, European Wireless 2016 - 22th European Wireless Conference*, p. 6, VDE VERLAG GMBH;, 2016.

[6] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer, "A comparative study of LPWAN technologies for large-scale IoT deployment," *ICT Express*, vol. 5, no. 1, pp. 1–7, 2019.

[7] A. M. Cardenas, M. Zennaro, M. K. Nakamura Pinto, M. Rainone, E. Pietrosemoli, and P. Manzoni, "A LoRa enabled sustainable messaging system for isolated communities," *ACM International Conference Proceeding Series*, pp. 118–123, 2018.

[8] A. Mueen, E. Keogh, Q. Zhu, S. Cash, and B. Westover, "Exact discovery of time series motifs," *Society for Industrial and Applied Mathematics - 9th SIAM International Conference on Data Mining 2009, Proceedings in Applied Mathematics*, vol. 1, pp. 469–480, 2009.

[9] I. Inc, "Time series database (TSDB) explained."

[10] J. Catsoulis, *Designing embedded hardware*. .º'Reilly Media, Inc.", 2005.

[11] Aosong Electronics Co., "DHT22 Temperature and Humidity Sensor Datasheet," 2003.

[12] M. integrated, "DS18B20 Programmable Resolution 1-Wire Digital Thermometer DS18B20 Programmable Resolution 1-Wire Digital Thermometer Absolute Maximum Ratings," vol. 92, pp. 1–20, 2019.

[13] J. Zazueta, F. S., Xin, "Soil Moisture Sensors," *Florida Cooperative Extension Service*, vol. 30, no. 1, pp. 69–83, 1994.

[14] A. Martinez and A. P. Byrnes, "Modeling dielectric-constant values of geologic materials: An aid to ground-penetrating radar data collection and interpretation," *Current Research in Earth Sciences*, vol. 247, no. 1, pp. 1–16, 2001.

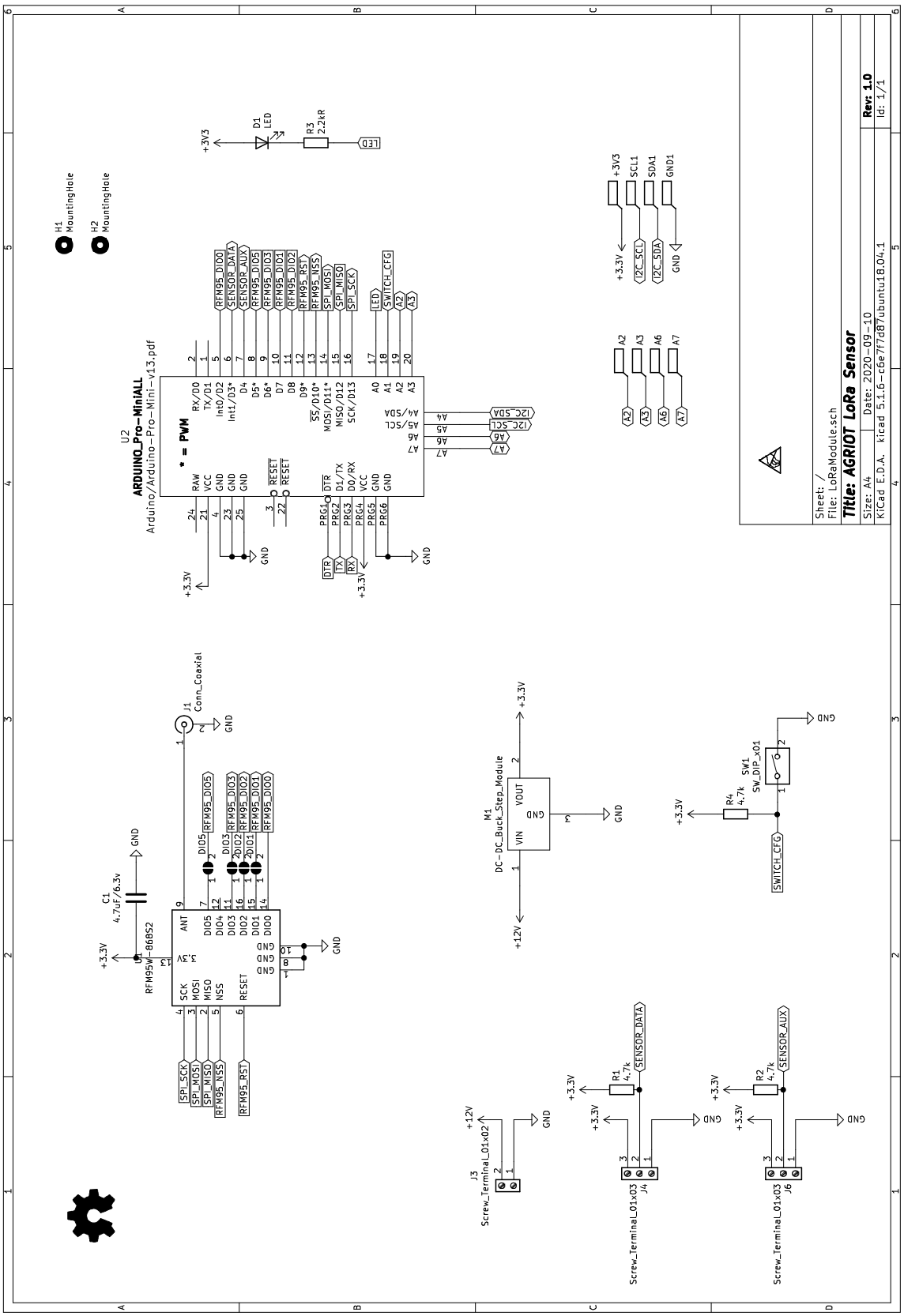
[15] C. Rattanapoka, S. Chanthakit, A. Chimchai, and A. Sookkeaw, "An MQTT-based IoT Cloud Platform with Flow Design by Node-RED," *RI2C 2019 - 2019 Research, Invention, and Innovation Congress*, no. June, 2019.

[16] B. LAB, "How to deploy a node red environment on a gcp instance," 2018.

Apéndice A

Apéndice - Diseño Hardware

A.1. Esquemático nodo LoRa



A.2. Gerbers

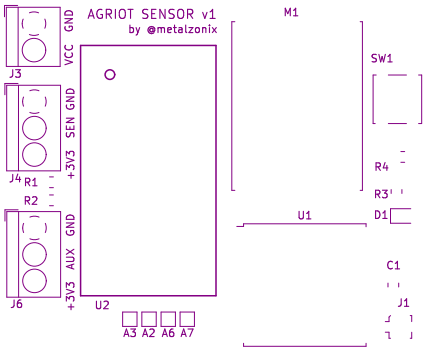


Figura A.1: Legend Top.

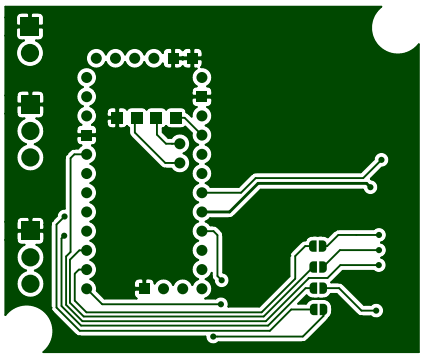


Figura A.2: Copper Top.

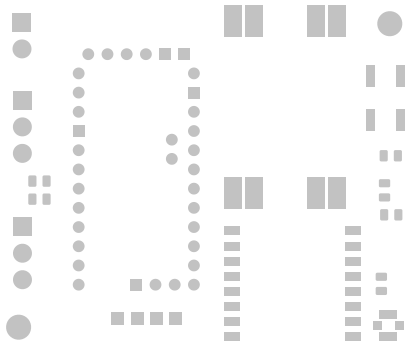


Figura A.3: Solder Mask Top.

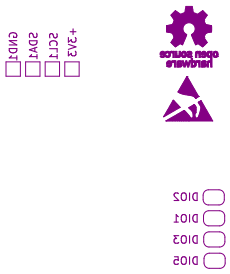


Figura A.4: Legend Botton.

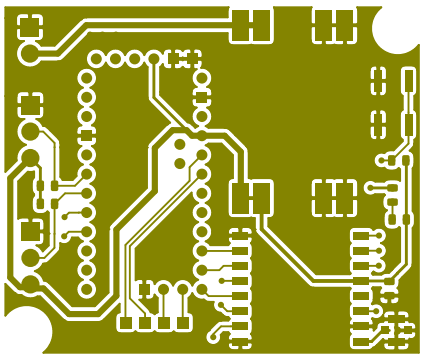


Figura A.5: Copper Botton.

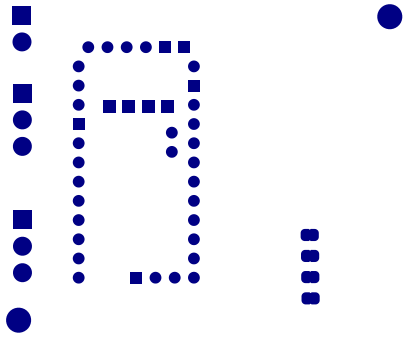


Figura A.6: Solder Mask Botton.

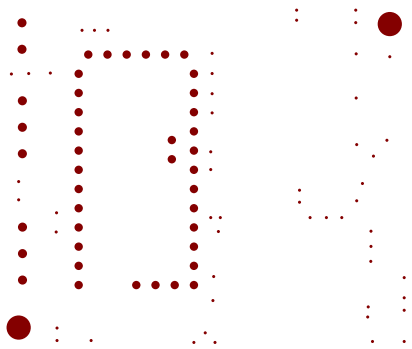


Figura A.7: Drill.

A.3. Escudo de radiación de temperatura

Tal y como se indicó en el Capítulo 4, el sensor de humedad y temperatura ambiente se ha introducido en el interior de un escudo de radiación. Esta protección mecánica permite que el aire del ambiente llegue al sensor pero no se vea afectado directamente por la influencia de la radiación solar o la entrada de agua. El escudo de radiación ha sido impreso en 3D con material plástico ABS (Acrylonitrile Butadiene Styrene) de color blanco, reduciendo así la absorción de radiación solar y el calentamiento de la cobertura. En la Figura A.8 se muestran las piezas que forman el escudo de radiación, obtenidas de la plataforma Thingiverse https://www.thingiverse.com/thing:1067700.

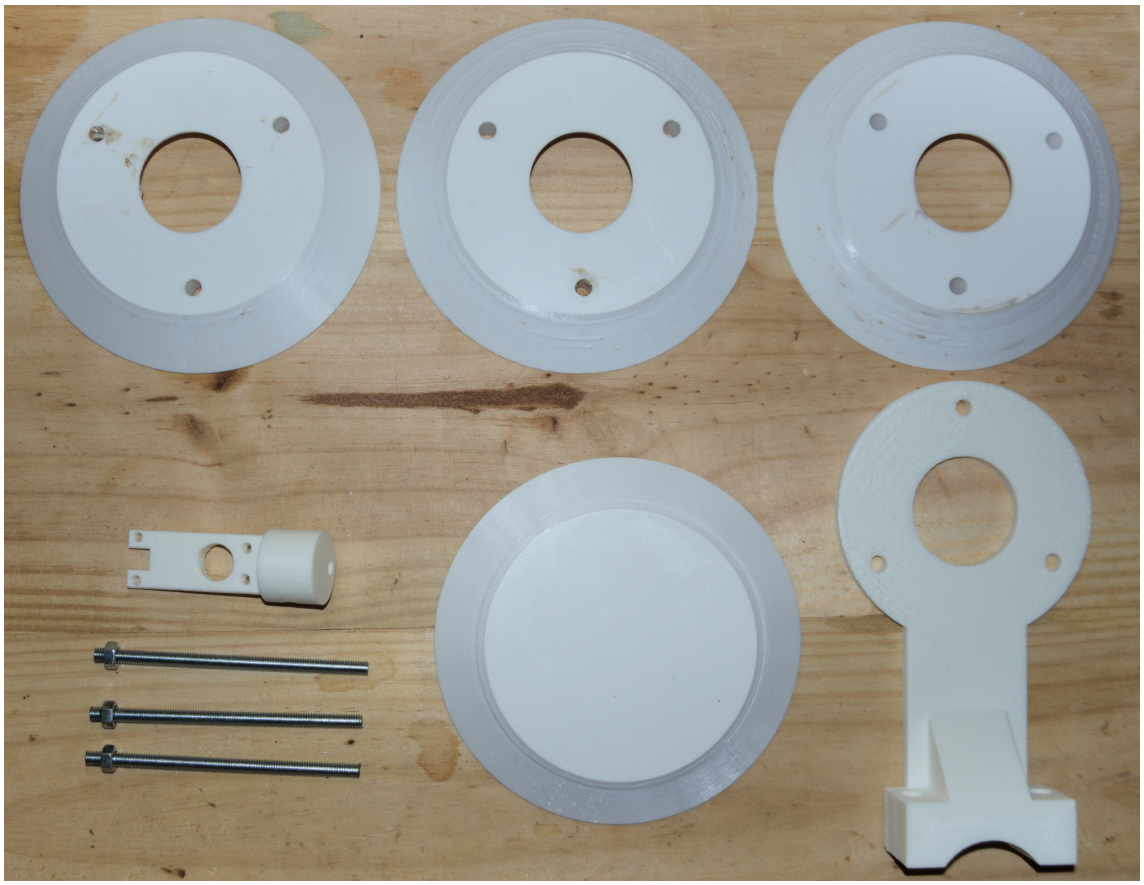


Figura A.8: Piezas impresas en 3D del escudo de radiacion.

Apéndice B

Apéndice - Software de plataforma

B.1. Instalación y configuración del servicio NODE-RED

En primer lugar es necesario abrir el puerto TCP 1880, ya que es el usado por Node-RED, la interfaz web de desarrollo.

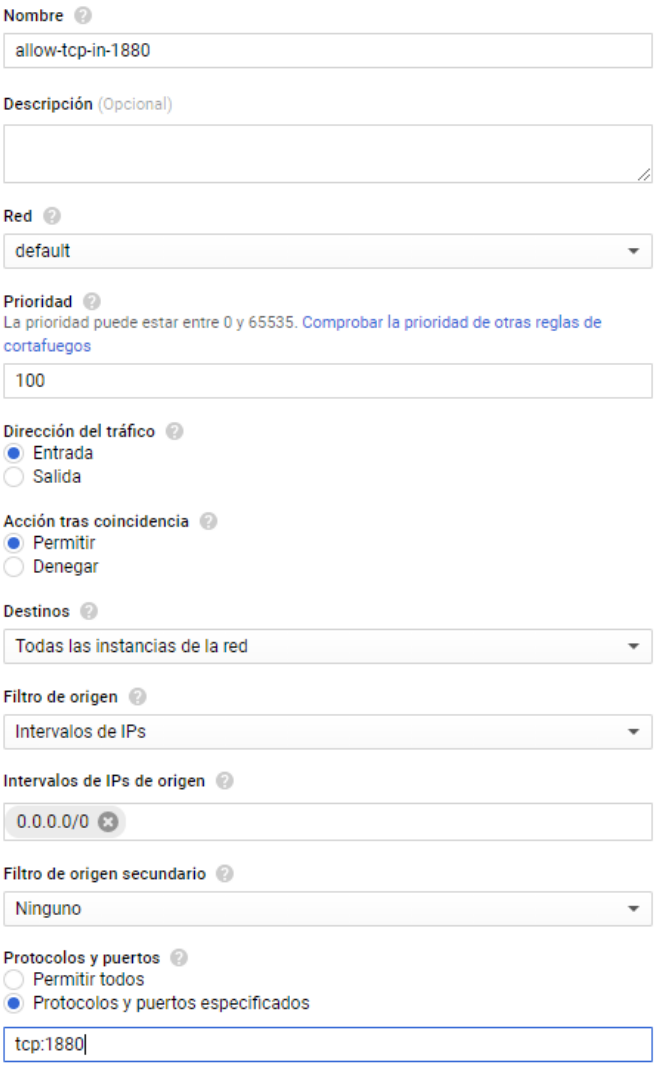


Figura B.1: Configuración firewall.

El siguiente paso es instalar nodejs, en este caso la última versión disponible en el momento de escribir estas líneas:

```
~$ sudo apt-get install build-essential
~$ curl -sL https://deb.nodesource.com/setup_15.x | sudo -E bash -
~$ sudo apt-get install -y nodejs
```

Instalar Node-RED y la herramienta de administración, las librería de nodos para conectar con la base de datos InfluxDB y otra para facilitar la conversión de formatos de

fechas:

```
~$ sudo npm install -g --unsafe-perm node-red
~$ sudo npm install -g node-red-admin
~$ sudo npm install -g node-red-contrib-influxdb node-red-contrib-moment
```

El siguiente paso, de vital importancia, es proteger el acceso al servicio Node-RED:

```
~$ node-red-admin hash-pw
```

Lo que nos preguntará por la contraseña con la que deseamos proteger el servidor. Tras escribirla y pulsar intro, nos devolverá una secuencia hash con la contraseña encriptada, que deberemos introducir en el archivo de configuración de Node-RED:

```
~$ sudo nano ~/.node-red/settings.js
```

y dentro de este archivo, se sustituye el USUARIO y HAST_GENERADO:

```
adminAuth: {
  type: "credentials",
  users: [{
    username: "USUARIO",
    password: "HASH_GENERADO",
    permissions: "*"
  }]
},

// To password protect the node-defined HTTP endpoints (httpNodeRoot), or
// the static content (httpStatic), the following properties can be used.
// The pass field is a bcrypt hash of the password.
// See http://nodered.org/docs/security.html#generating-the-password-hash
httpNodeAuth: {user:"USUARIO",pass:"HASH_GENERADO"},
httpStaticAuth: {user:"USUARIO",pass:"HASH_GENERADO"},
```

Por último, será interesante que el servidor cargue automáticamente Node-RED cuando se reinicie:

```
~$ sudo nano /etc/systemd/system/node-red.service
```

donde copiaremos el siguiente texto:

```
[Unit]
Description=Node-RED
After=syslog.target network.target

[Service]
ExecStart=/usr/bin/node-red
Restart=on-failure
KillSignal=SIGINT

# log output to syslog as 'node-red'
SyslogIdentifier=node-red
```

```
StandardOutput=syslog

# non-root user to run as
WorkingDirectory=/home/TUUSUARIO/
User=TUUSUARIO
Group=TUUSUARIO

[Install]
WantedBy=multi-user.target
```

B.2. Instalación y configuración de InfluxDB

Añadir el repositorio InfluxData al gestor de paquetes de Ubuntu para su instalación:

```
~$ curl -s https://repos.influxdata.com/influxdb.key | sudo apt-key add -
source /etc/lsb-release
~$ echo "deb https://repos.influxdata.com/${DISTRIB_ID,,} ${DISTRIB_CODENAME
    } stable" | sudo tee /etc/apt/sources.list.d/influxdb.list
```

Instalar e iniciar el servicio InfluxDB:

```
~$ sudo apt-get update && sudo apt-get install influxdb
~$ sudo service influxdb start
```

Con el objetivo de mejorar la seguridad de la instalación, se ha habilitado la autenticación en el servidor InfluxDB y se ha creado un usuario con privilegios de sólo lectura que será usado para acceder a los datos desde Grafana.

Todos los parámetros de configuración de InfluxDB están contenidos en el archivo (/etc/influxdb/influxdb.conf). La mayoría de los ajustes están comentados, lo que implica la aplicación de la configuración predeterminada. El primer paso consiste en habilitar la autenticación en la configuración de InfluxDB [16]:

```
~$ nano /etc/influxdb/influxdb.conf
```

Cambiar el valor de la variable auth-enabled por "true":

```
[http]
enabled = true
bind-address = ":8086"
auth-enabled = true
```

Reiniciar el servicio InfluxDB:

```
~$ sudo service influx restart
```

Cuando se habilita la autenticación y no existe al menos un usuario administrador, InfluxDB no permite ejecutar ninguna consulta desde su CLI, excepto la creación de un usuario administrador.

```
~$ influx
> create user admin with password 'admin1234' with all privileges
> quit
```

A partir de ahora es necesario autenticarse ingresando usuario (-username) y contraseña (-password) como parámetros al cliente "influx". Una vez verificado el proceso de autenticación, se crea la base de datos en la que se van a guardar las observaciones de los sensores, cuyo nombre es agriot:

```
~$ influx -username admin -password ''
password:
> create database agriot
> show databases
name: databases
name
----
_internal
agriot
```

Una vez creado el usuario administrador, es posible crear un usuario de sólo lectura y dar acceso a este usuario a la base de datos "agriot":

```
> create user grafana with password 'grafana1234'
> grant read on "agriot" to "grafana"
> quit
```

Configurar el inicio automática de InfluxDB en el arranque del sistema:

```
~$ sudo systemctl enable influxdb.service
```

B.3. Instalación y configuración de Grafana

Instalar dependencias y añadir el repositorio de Grafana para versiones estables:

```
~$ sudo apt-get install -y apt-transport-https
~$ sudo apt-get install -y software-properties-common wget
~$ wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -
~$ echo "deb https://packages.grafana.com/oss/deb stable main" | sudo tee -a
    /etc/apt/sources.list.d/grafana.list
```

Instalar Grafana:

```
~$ sudo apt-get update
~$ sudo apt-get install grafana
```

Iniciar el proceso del servidor de Grafana:

```
~$ sudo systemctl daemon-reload
~$ sudo systemctl start grafana-server
~$ sudo systemctl status grafana-server
```

Configurar el inicio automática de Grafana en el arranque del sistema:

```
~$ sudo systemctl enable grafana-server.service
```