ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

REDES DE SENSORES INALÁMBRICOS PARA IOT
MONITOREO DEL NIVEL DE BATERÍA DE UN NODO LORAWAN

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN TELECOMUNICACIONES

LORENA CAROLINA TEPUD MARTÍNEZ
lorena.tepud@epn.edu.ec

DIRECTOR:

CARLOS ROBERTO EGAS ACOSTA, MSc.
carlos.egas@epn.edu.ec

DMQ, abril 2023

CERTIFICACIONES

Yo, LORENA CAROLINA TEPUD MARTÍNEZ declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

Lorena Carolina Tepud Martínez

Certifico que el presente trabajo de integración curricular fue desarrollado por LORENA CAROLINA TEPUD MARTÍNEZ, bajo mi supervisión.

Carlos Roberto Egas Acosta, MSc.

DIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como él (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

LORENA CAROLINA TEPUD MARTÍNEZ
CARLOS ROBERTO EGAS ACOSTA. MSc.

DEDICATORIA

Este trabajo de integración curricular está dedicado a mis padres Telmo Tepud y María Martínez, quienes con esfuerzo y dedicación han guiado mis pasos para formar a la persona que hoy en día soy.

A mis hermanos Sandra y Diego, quienes han apoyado mis decisiones y con un consejo o palabra de apoyo, me han dado fortaleza para seguir adelante en cada paso que he dado.

A mi familia y en especial a mis tías Sonia y Verónica, quienes con sus consejos y compañía me han ayudado a enfrentar las adversidades de la vida.

AGRADECIMIENTO

Me gustaría tener palabras que expresen mi gratitud hacia todas las personas que han participado directa e indirectamente en el desarrollo de este trabajo; sin embargo, haré lo posible por mencionar a todos:

Gratitud es lo que más le tengo a Dios, por cada bendición que ha llegado a mi vida, así como cada prueba, sin él nada de esto sería posible.

Mi agradecimiento especial para mis padres Telmo y María, quienes en gran medida son responsables de la persona que soy hoy en día.

A mis hermanos Sandra y Diego, quienes aportaron a mi crecimiento, cada uno con su forma de ser me ayudaron de manera significativa a ser mejor cada día.

A Elvis Bonilla, quien ha estado en esta etapa de manera incondicional para mí, por brindarme su apoyo y por compartir momentos nuevos e inolvidables.

A mi mejor amiga Viviana, quien desde los 12 años ha crecido conmigo en metas, sueños e ideales, cada consejo, cada opinión me han aportado para ser una mejor persona.

A mis amigos Nubia y Jonathan por apoyarme, ayudarme y aconsejarme en cada paso que di, gracias por las risas, las lágrimas, las malas noches, gracias por todas las aventuras que vivimos en esta etapa, siempre los llevaré en el corazón, gracias por su amistad.

ÍNDICE DE CONTENIDO

RESUMEN

En la actualidad la expansión de Internet de las cosas (IoT: Internet of things) ha llevado a la industria a la investigación de nuevas tecnologías de comunicación que tengan la capacidad de solventar sus necesidades, una de las tecnologías que logra resolver los problemas de transmisión de datos a largas distancias con bajo consumo de energía es la tecnología LPWAN y su estándar LoRaWAN, el cual presenta soluciones a esta problemática, a partir de los dispositivos finales de bajo coste y baja complejidad que pueden comunicarse de forma inalámbrica a grandes distancias como lo es la placa Wio-E5, siendo este dispositivo un nodo sensor alimentado por batería. Para poder monitorear el nivel de batería de diseño e implemento un prototipo, el cual se compone de un nodo LoRaWAN conectado a un Gateway LoRa y como aplicación final un dashboard ejecutado en node red, logrando visualizar los datos que arroja el sistema, controlando así el tiempo de duración de una batería dentro de una red LoRaWAN y tomar medidas oportunas cuando la batería esté muy baja. Al implementar el prototipo se obtuvieron resultados satisfactorios, detallando así sus ventajas. Este prototipo tiene utilidad en diferentes aplicaciones enfocadas al desarrollo de la tecnología.

Pablabras clave: IoT, LoRaWAN, monitorear, nivel de batería, prototipo.

ABSTRACT

Currently, the expansion of the Internet of Things (IoT) has led the industry to research new communication technologies that can meet their needs. One of the technologies that manages to solve the problem of transmitting data over long distances with low energy consumption is LPWAN technology and its LoRaWAN standard, which presents solutions to this problem using low-cost, low-complexity end devices that can communicate wirelessly over long distances, such as the Wio-E5 board, a battery-powered sensor node. In order to monitor the battery level, a prototype was designed and implemented, consisting of a LoRaWAN node connected to a LoRa gateway, and a final dashboard application executed in Node-RED, allowing for the visualization of the system's data, thus controlling the battery life within a LoRaWAN network and taking appropriate measures when the battery is low. The prototype yielded satisfactory results, detailing its advantages. This prototype is useful in different applications focused on technology development.

Keywords: IoT, LoRaWAN, monitoring, energy level, prototype.

1 INTRODUCCIÓN

En la actualidad la expansión de Internet de las cosas (IoT: Internet of things) como redes inteligentes, comunicaciones de sensores industriales, monitoreo de entornos remotos, agricultura de precisión, monitoreo de infraestructura, etc., ha llevado a la industria a la investigación de nuevas tecnologías de comunicación alternativas, que puedan ofrecer un funcionamiento de bajo consumo, debido a que los protocolos existentes son insuficientes en términos de cobertura y consumo de energía a grandes distancias, necesarias para satisfacer los requerimientos de IoT [1].

Las tecnologías de red de área amplia de baja potencia (LPWAN: Low Power Wide Area Network), tiene como objetivo resolver este tipo de problemas mediante su estándar red de área amplia de largo alcance (LoRaWAN: Long Range Wide Area Network), el cual presenta soluciones de bajo consumo de energía, dispositivos finales de bajo coste y baja complejidad que puedan comunicarse de forma inalámbrica a grandes distancias, dichos dispositivos son nodos sensores alimentados por batería [2].

Una característica de los nodos LoRaWAN es que tratan con baterías, las cuales deben tener una duración amplia para que puedan operar dentro de la red LoRa durante un tiempo prolongado. Dentro de este tiempo es importante controlar el uso de batería para determinar en qué punto estas baterías deben ser reemplazadas. La mayoría de nodos sensores LoRaWAN están diseñados con una vida útil de varios años, el tiempo de duración está determinado por el uso, la frecuencia de transmisión de datos, la distancia entre las puertas de enlace, el tipo de batería y el tipo de nodo sensor [3].

Uno de los principales problemas para atender los casos de uso IoT, es la batería, debido a que el funcionamiento intensivo de los nodos finales ubicados en un área geográfica más amplia sin influencia humana hace que el reemplazo regular de la batería sea impracticable. Por otro lado, para que la recopilación, transmisión y análisis de datos de sensores sea correcta, se debe aumentar mas el consumo de recursos de batería los cuales son limitados [3].

El presente Trabajo de Integración Curricular, desarrollará una aplicación que permita monitorear de forma remota la batería del nodo para finalmente visualizar los datos que arroje este sistema y así controlar los tiempos de duración de una batería dentro de una red LoRaWAN.

1.1 OBJETIVO GENERAL

Monitorear el nivel de batería de un nodo LoraWAN en entorno a una red punto-punto.

1.2 OBJETIVOS ESPECÍFICOS

1. Definir el fundamento teórico de las tecnologías Lora y LoraWAN en las redes de sensores inalámbricos.
2. Diseñar un prototipo de un sistema de monitoreo del nivel de batería de un nodo LoraWAN, dentro de una red LoRaWAN.
3. Implementar un prototipo de un sistema de monitoreo del nivel de batería de un nodo LoraWAN en una red LoRaWAN.
4. Presentar los resultados en base a la implementación del prototipo.

1.3 ALCANCE

El presente Trabajo de Integración Curricular tiene como objetivo implementar un sistema de monitoreo de nivel de batería de un nodo LoRaWAN, para lo cual se empezará por desarrollar una fase teórica en donde se estudiarán las necesidades que se deben solventar el estándar LoRaWAN como son, baja potencia, bajo consumo de la red y los dispositivos finales que puedan comunicarse de forma inalámbrica a grandes distancias, después del correspondiente estudio se seleccionara un nodo LoRaWAN que permita obtener el nivel de batería del nodo y pueda ser configurado para tal efecto. Uno de los principales enfoques es el estudio de herramientas con las cuales se debe configurar el nodo. Una vez realizado el estudio del nodo seleccionado se definirá las características mínimas del sistema que permitan monitorear el nivel de la batería del nodo. Luego de esto se diseñará el código que permita la obtención del nivel de la batería, así como su transmisión a la computadora donde se visualizara en modo texto, lo que nos permitirá comprobar el funcionamiento del prototipo, de igual manera se diseñara el software a implementar en la computadora que permita obtener la información del nivel de batería de un nodo LoRaWAN así como la presentación de este en pantalla en modo texto. Una vez realizado el diseño de los dos pro-

gramas se realizará la codificación de los programas en un lenguaje que será seleccionado para tal efecto. El lenguaje x en el nodo y el lenguaje y a ser utilizado en la computadora. La implementación de servidores LoRaWAN*, de la parte del diseño de la red implica la implementación de los servidores LoRaWAN necesarios para que el PC se comunique con los nodos. Finalmente, se entrará en una fase de análisis de resultados, donde se podrá visualizar los datos arrojados por el sistema y determinar cómo está funcionando el nodo y determinar la validez del sistema implementado para monitorear el nivel de la batería.

1.4 MARCO TEÓRICO

1.4.1 Definición de LPWAN

Actualmente, las tecnologías inalámbricas como las redes de área local inalámbricas (WLAN: Wireless Local Area Network) IEEE 802.11, Bluetooth IEEE 802.15.1 y redes de área personal inalámbricas (WPAN: Wireless Personal Area Network) IEEE 802.15.3, se utilizan para aplicaciones de sensores en entornos de corto alcance. WLAN y Bluetooth se diseñaron principalmente para la comunicación de datos de alta velocidad, mientras que ZigBee y WPAN se utilizan para aplicaciones de baja velocidad de datos en entornos locales.

Por el contrario, las redes celulares inalámbricas como 2G, 3G y 4G están diseñadas principalmente para comunicaciones de voz, datos, vídeo y a pesar de que pueden proporcionar una mayor cobertura, consumen demasiada energía del dispositivo. Todas estas tecnologías son inconsistentes para cumplir con las necesidades de las aplicaciones de IoT como largo alcance, baja velocidad de datos, bajo consumo de energía y rentabilidad, siendo esta la razón principal por la cual se ha impulsado el surgimiento de una nueva tecnología de comunicación inalámbrica: la red de área amplia de baja potencia (LPWAN: Low Power Wide Area Network) [4].

Las LPWAN son redes con recursos limitados y se caracterizan principalmente por una operación de batería de larga duración (es decir, más de 10 años de vida útil de la batería), alta capacidad y bajos costos de implementación y dispositivos, proporciona comunicación de largo alcance de hasta 10 a 40 km en zonas rurales y de 1 a 5 km en zonas urbanas [5].

Las soluciones principales patentadas y basadas en estándares disponibles en el mercado de la tecnología LPWAN son: Sigfox, LoRaWAN, NB-IoT y LTE-M.

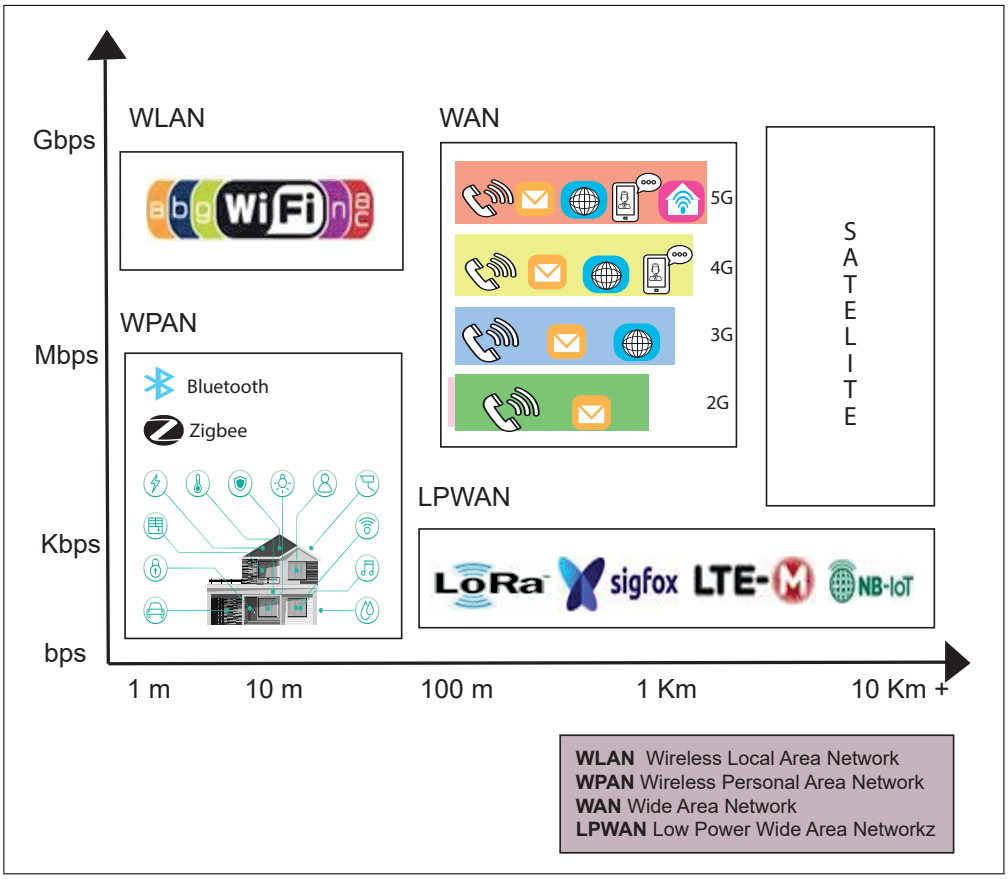

Figura 1.1: Tecnologías inalámbricas en función de la distancia y velocidad de trasmisión

1.4.2 Lora

La tecnología inalámbrica de largo alcance (LoRa:Long Range) es una tecnología de capa física, desarrollada por Semtech que modula las señales en las bandas de frecuencias para uso sin licencia y uso no comercial de radiofrecuencia electromagnética en áreas industriales, científicas y médicas (ISM: Industrial, Scientific and Medical), operado en bandas de frecuencia de 868 MHz y 433 MHz en Europa, 915 MHz en América del Norte y 433 MHz en Asia [6].

LoRa se basa en la modulación de espectro ensanchado chirp (CSS: Chirp Spread Spectrum), la cual es una modulación de frecuencia lineal de banda ancha, en la que la frecuencia de la portadora varía durante un período de tiempo determinado. LoRa utiliza conexión inalámbrica de un solo salto debido a que transmite los mensajes recibidos de los nodos sensores LoRa al servidor central a través de puertas de enlace. La velocidad de transmisión de datos admitida por LoRa varía de 300 bps a 50 kbps, según el factor de dispersión (SF: Spreading Factor) y la configuración del ancho de banda del canal. Las estaciones ba-

se LoRa pueden recibir mensajes transmitidos utilizando diferentes factores de dispersión [5]. Las transmisiones LoRa con los SF más altos amplían el rango, pero limitan la calidad de servicio y los SF del 7 al 12 permiten comunicaciones ortogonales, es decir, diferentes redes pueden hablar simultáneamente en la misma banda de frecuencia sin interferir. Permite un compromiso entre la tasa de datos y la cobertura, además de optimizar el rendimiento de la red con un ancho de banda [6].

LoRa Alliance ha desarrollado una capa de control de acceso medio llamada LoRaWAN, la cual incluye funcionalidades de red y capa superior. LoRaWAN presenta tres clases de dispositivos finales, los cuales cubren los diferentes requisitos de las distintas aplicaciones de internet de las cosas (IoT: Internet Of Things), siendo LoRa una de las mejores tecnologías para transmisiones de larga distancia y baja potencia [4].

1.4.3 LoRaWAN

Una red de área amplia de baja potencia (LoRaWAN: Low Power Wide Area Networks) es un estándar LPWAN abierto desarrollado por LoRa Alliance, el cual fue lanzado por primera vez en 2015. LoRaWAN es un protocolo de la capa de enlace de datos la cual proporciona un mecanismo de control de acceso al medio (MAC: Media Access Control) que está encargado de la arquitectura del sistema para la red y de la comunicación entre múltiples dispositivos y gateways, este protocolo corre encima de la capa física LoRa, la cual permite el enlace de comunicación de largo alcance [6].

LoRaWAN utiliza anchos de banda de 125, 250 o 500 kHz y transmite enlaces a través de varios kilómetros con velocidades de datos de hasta 5,5 kbps, siendo esto útil para distintas aplicaciones en las cuales se requiere de sensores que no dispongan de corriente eléctrica de red como es en el caso de lugares de poca cobertura. Así mismo, LoRaWAN solo admite la comunicación de un solo salto entre los nodos finales y los gateways ubicadas a varios kilómetros de distancia con un bajo consumo de energía, debido a este largo alcance las señales LoRaWAN pueden encontrar obstáculos o interferencias, lo que afecta la recepción de paquetes [1].

LoRaWAN define dos tipos básicos de dispositivos, el primer tipo es el nodo final, el cual es un dispositivo simple que envía información a los gateways mediante un enlace ascendente y recibe información del gateway a través de un enlace descendente, el segundo tipo es el gateway el cual es un dispositivo con una radio compleja que escucha simultáneamente en diferentes canales; en consecuencia, admiten miles de nodos al mismo tiempo. Los

gateways envían datos a un servidor principal mediante conexiones a Internet [1].

1.4.4 Clases de LoRaWAN

LoRaWAN tiene tres diferentes clases de dispositivos de punto final para cubrir las diferentes necesidades reflejadas en la amplia gama de aplicaciones. A continuación se describe cada una de ellas.

1.4.4.1 Clase A:

Todos los nodos deben implementar de manera obligatoria la clase A, donde los dispositivos envían información aleatoriamente al gateway utilizando el protocolo ALOHA para mantener la complejidad de la red lo más simple posible y maximizar el ahorro de energía [1]. En los dispositivos de Clase A, el enlace descendente (DL: Downlink) está controlado por el nodo final y los dispositivos no tienen una ventana de recepción de DL activa. Solo se activan dos ventanas de recepción de DL después se realiza una transmisión de enlace ascendente (UL: Uplink). Cualquier paquete que se transmita desde GW al nodo debe esperar el siguiente mensaje de UL. Por lo tanto, los dispositivos de Clase A son los más eficientes energéticamente [6].

1.4.4.2 Clase B:

La clase B es útil para aquellos dispositivos finales que reciben más información del gateway [1]. Para este caso, el gateway envía balizas de sincronización para abrir más ventanas de recepción con intervalos de tiempo predeterminados, aumentando así las posibilidades del enlace descendente. Debido a que los dispositivos de clase B necesitan abrir más ventanas, consumen más energía en comparación con los dispositivos de clase A [2].

1.4.4.3 Clase C:

En la clase C, los nodos mantienen sus ventanas de recepción de DL siempre están activas y solo las cierran para enviar un paquete a través del enlace ascendente. Los nodos que admiten la clase C no implementan la clase B. Debido a que la clase C hace que los nodos reciban continuamente, estos dispositivos aumentan el consumo de energía, siendo estos los dispositivos menos eficientes [1]. Clase A y los dispositivos B tienen una latencia más alta en comparación con los nodos de clase C [2].

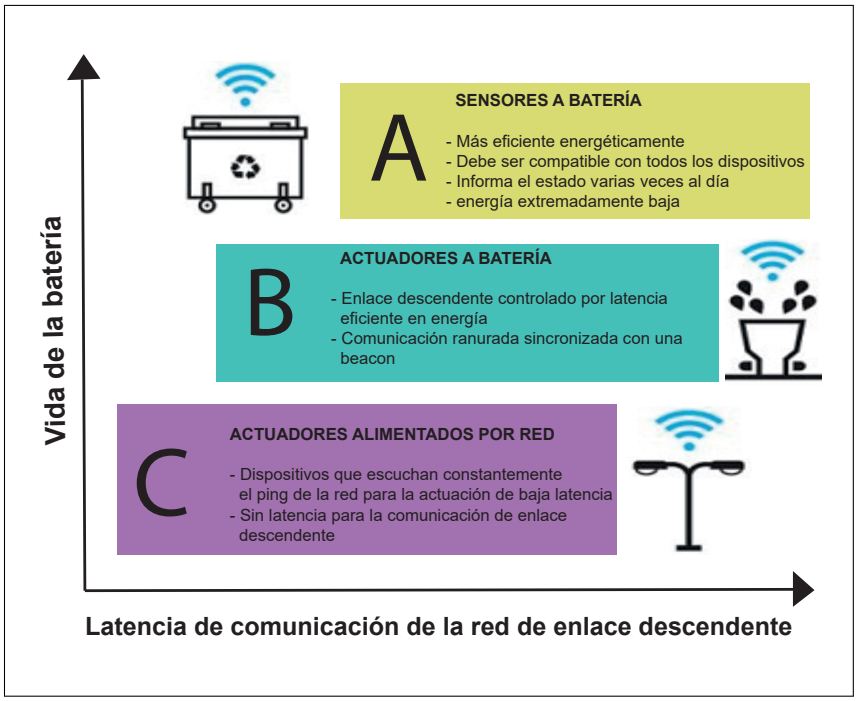

Figura 1.2: Clases de LoRaWAN en función de la latencia de comunicación de la red de enlace descendente vs. la vida de la batería [3].

1.4.5 Seguridad LoRaWAN

La seguridad del protocolo de red LoRaWAN se basa en IEEE 802.15.4, ampliándose mediante la utilización una capa de seguridad para la red y otra capa de seguridad para la aplicación [7]. La autenticidad del dispositivo final (nodo) en la red, es garantizada por la capa de seguridad de la red y para garantizar que el operador de la red no tenga acceso a los datos de la aplicación del usuario final, se emplea la capa de seguridad de aplicación; cada dispositivo final LoRaWAN está protegido por encriptación AES 128 por cuatro componentes:

1. **Dirección del dispositivo (DevAddr:Device Address)**: Un identificador de dispositivo de 32 bits, asignado por el servidor de red durante la activación del dispositivo.

2. **Aplicación EUI (AppEUI: App EUI)**: En las especificaciones de LoRaWAN anteriores a 1.1, JoinEUI se llamaba AppEUI [8]. Un identificador único de aplicación que es un espacio de direcciones IEEE UI64.

3. **Clave de sesión de red (NwkSKey: Network Session Key)** : Después de la activación, esta clave de cifrado se utiliza para proteger la comunicación entre el dispositivo final y el servidor de red [8].

4. **Clave de sesión de la aplicación (AppSKey: Application Session Key)**: Una clave de sesión de aplicación (claves AES-128) utilizada para proteger datos específicos de la aplicación.

Todo dispositivo final conectado a una red LoRaWAN admite dos mecanismos principales de activación: activación por personalización (ABP) o activación por aire (OTAA).

1.4.5.1 Activación por aire (OTAA: Over-the-Air Activation)

Antes de participar en intercambios de datos con el servidor de red, los dispositivos finales deben seguir un procedimiento de unión. Además, si un dispositivo final pierde la información de contexto de la sesión, debe pasar por un nuevo proceso de unión para poder participar en la comunicación de datos nuevamente.

Antes de iniciar el procedimiento de unión, se requiere que el dispositivo final se personalice con la siguiente información: un identificador de dispositivo final globalmente único (DevEUI), un identificador de aplicación (AppEUI) y una clave AES-128 (AppKey). Estos datos son necesarios para llevar a cabo el proceso de unión de manera exitosa [9].

Los dispositivos realizan un procedimiento de unión con la red, durante el cual se asigna una dirección de dispositivo dinámica y se negocian claves de seguridad con el dispositivo. Este método es el más seguro y preferido para conectarse a una red LoRaWAN, que ofrece mayor flexibilidad y escalabilidad que ABP [8].

1.4.5.2 Activación por Personalización (ABP: Activation by Personalization)

Es posible que los dispositivos finales se comuniquen directamente con el servidor de red utilizando claves previamente establecidas. Sin embargo, es esencial almacenar y proteger estas claves, ya que cualquier ataque de hardware puede comprometerlas y dar lugar a accesos no autorizados a la red. Aunque este método de comunicación pueda parecer más sencillo al omitir el procedimiento de unión, existen riesgos de seguridad que deben considerarse.

La metodología ABP también presenta ciertas desventajas, como la incapacidad de los dispositivos de cambiar de proveedor de red sin tener que actualizar manualmente las claves en cada dispositivo [8]. Además, este enfoque implica una conexión estricta de los nodos finales a un servidor de red específico, lo que impide que dichos dispositivos se beneficien de las funciones de roaming disponibles en LoRaWAN v1.1 [6].

1.4.6 Arquitectura de red LoRaWAN

La red LoRaWAN tiene una topología en estrella, donde los dispositivos finales solo pueden comunicarse con los gateways y no directamente entre sí [6]. La arquitectura la conforman cuatro componentes principales, gateways, nodos finales, servidores de red y servidor de aplicación. La comunicación de los nodos finales a través del gateway puede utilizar modulación LoRa o FSK con diferentes velocidades de datos y canales [7].

Los gateways solo están encargados de reenviar paquetes de datos sin procesar desde los nodos finales hacia el servidor de red, encapsulándolos en paquetes UDP/IP. El servidor de red es responsable de gestionar y enrutar los paquetes de datos de los gateways, auténtica, autoriza los nodos y envía los datos a la Aplicación final para su procesamiento y almacenamiento, si es necesario.

La comunicación termina en los servidores de aplicaciones que pueden ser propiedad por terceros. Se pueden conectar varias capas de aplicaciones a un único servidor de red [6].

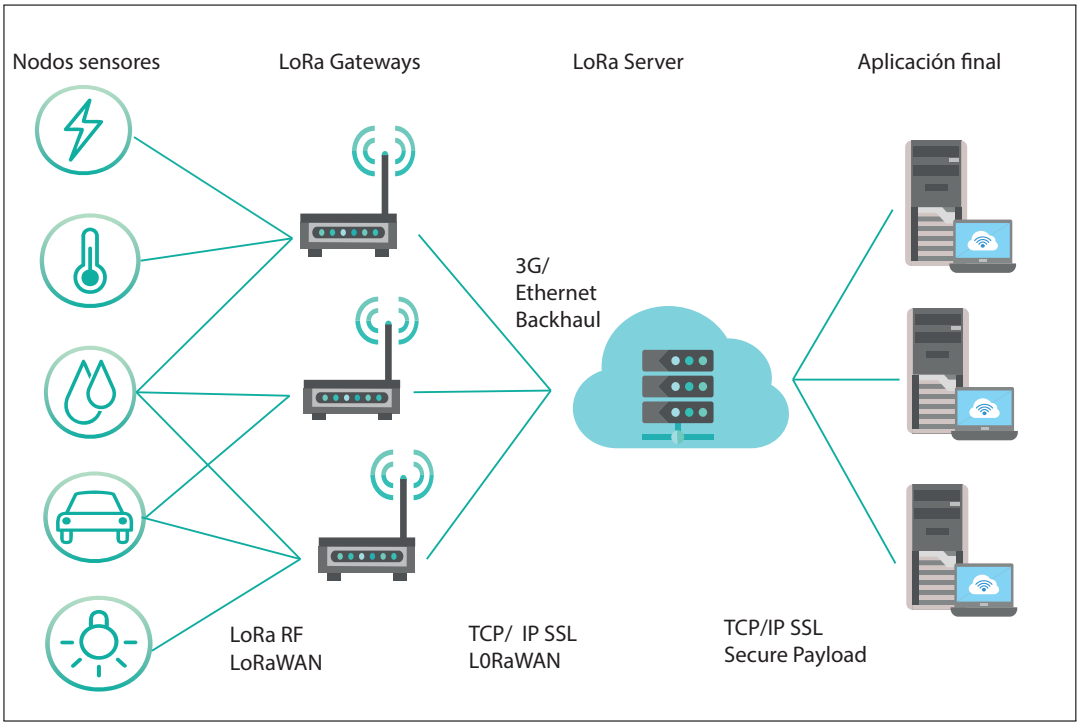

Figura 1.3: Arquitectura de una red LoRaWAN.

1.4.7 GW RG191

La figura 1.4 muestra el gateway Sentrius RG1xx LoRa-Enabled de Laird, que representa la última solución en términos de seguridad, escalabilidad y robustez en el ámbito de LoRa,

con el control de extremo a extremo de su red privada LoRaWAN. Además, incorpora un módulo certificado de Wireless Bridge de la serie 50, que es confiable y ha sido probado en el campo de Laird. Asimismo, ofrece una banda dual empresarial Wifi y Ethernet por cable, lo que permite una completa libertad de diseño.

El gateway Sentrius RG1xx LoRa-Enabled de Laird tiene diversas aplicaciones en el campo de la IoT, incluyendo el monitoreo ambiental, la automatización industrial, las ciudades inteligentes, etc. Eso gracias a que este dispositivo puede recopilar datos de sensores en tiempo real.

Figura 1.4: Gateway Sentrius RG191 [10].

1.4.7.1 Características técnicas del GW RG191

El gateway RG191 se basa en los diseños de chipset de Semtech SX1301/SX1257, lo que permite un rango LoRa de hasta diez millas. Además, cuenta con un software precargado LoRa Packet Forwarder, lo que lo hace ideal para redes IoT altamente escalables y flexibles. Por otro lado, el gateway Sentrius RG1xx funciona con los módulos certificados LoRa+BLE de la serie Sentrius RM1xx de Laird, lo que facilita su integración y uso inmediato. También es compatible con socios de LoRa y Cloud de terceros, así como con cualquier dispositivo cliente certificado por LoRaWAN [10].

GW RG191 cuenta con 8 canales LoRaWAN con una salida mayor a +27 dBm (cumple con el estándar). Para la administración de la configuración la información esta protegida, ya que tiene certificado TLS y autenticación basada en tokens.

1.4.8 The Things Network (TTN)

The Things Network ofrece una red global y abierta, así como herramientas de código abierto para crear aplicaciones IoT escalables, seguras y asequibles. En esta sección, se proporciona información general sobre la arquitectura de The Things Network, cómo administrarla a través de la CLI y la consola, y cómo implementar algunos o todos los componentes en servidores propios.

1.4.8.1 Arquitectura de la red

Los sistemas backend de The Things Network son responsables de dirigir los datos de IoT entre dispositivos y aplicaciones a través de gateways que actúan como puentes entre los protocolos de radio e Internet. Cuando los dispositivos admiten el stack de IP, estos gateways simplemente reenvían paquetes a Internet. Sin embargo, para protocolos como LoRaWAN, es necesario un enrutamiento y procesamiento específico antes de enviar mensajes a una aplicación. The Things Network actúa como intermediario entre los gateways y las aplicaciones, encargándose de los procesos de enrutamiento y procesamiento. Su enfoque es descentralizado y distribuido para establecer estas funciones de enrutamiento. Esta sección proporciona información general sobre la arquitectura de The Things Network, cómo administrarla a través de la CLI y la consola, y cómo implementar algunos o todos los componentes en servidores propios [11].

1.4.8.2 Función principal

Things Network ofrece un servidor de red LoRaWAN que es necesario debido al enfoque minimalista de los dispositivos y a la arquitectura centralizada de los operadores de telecomunicaciones. El protocolo LoRaWAN es considerado ïntensivo en la red", ya que la mayor parte de la lógica recae en los servidores de red. Para que LoRaWAN funcione en una infraestructura distribuida como la de The Things Network, se han agregado algunos pasos adicionales.

1. Algunas de las funciones asociadas con el gateway, como la programación y la gestión de la utilización de los gateways.

2. Se requieren funciones relacionadas con dispositivos que puedan administrar el estado de los dispositivos en la red.

3. Algunas funciones se encuentran relacionadas con las aplicaciones. Por ejemplo, los intermediarios y los controladores necesitan saber a qué servidor se debe reenviar el tráfico para una aplicación específica. Los controladores deben saber cómo interpretar datos binarios y conectarse con protocolos de capa superior, como AMQP y MQTT.

4. The Things Network es una red distribuida, por lo cual debería que haber una funcionalidad que permita esta distribución. La funcionalidad de detección de servicios ayuda a los componentes a determinar hacia dónde se debe enrutar el tráfico [11].

1.4.8.3 Seguridad de la red

Things Network es una red pública con un nivel de seguridad alta que permite un verdadero cifrado de extremo a extremo, mitigaciones contra varios ataques de intermediarios y compatibilidad con diferentes claves de cifrado de 128 bits para cada dispositivo final.

LoRaWAN cumple con el uso de comprobaciones de integridad de mensajes AES de 128 bits y cifrado de carga útil, las mismas que se encuentran totalmente encriptadas entre el nodo y el componente controlador del backend. Esto significa que puede optar por operar su propio controlador privado y tener un cifrado real de extremo a extremo. Los componentes del enrutador y el intermediario enrutan los datos en función de los metadatos públicos y no pueden descifrar la carga útil real [12].

1.4.8.4 Consola de red Things

Sus aplicaciones y dispositivos se pueden administrar a través de The Things Network Console. Esta es una interfaz de código cerrado sobre nuestra CLI de código abierto y las mismas API de nuestros componentes de servidor de código abierto que pueden usar sus aplicaciones.

Además de eso, proporciona una interfaz de usuario a nuestro Centro de operaciones de red de código cerrado para monitorear su uso de la red, así como integraciones de apuntar y hacer clic para plataformas de aplicaciones populares [13].

1.4.9 Arduino nano

El Arduino Nano es una placa electrónica de tamaño compacto y bajo costo que presenta alta compatibilidad con sensores y shields de Arduino, su facilidad de programación ayuda al desarrollo desde proyectos simples de electrónica hasta robótica avanzada y automatización del hogar. El Arduino Nano se basa en el ATmega328 (en Arduino Nano 3.x). A diferencia de otras placas, no tiene un conector de alimentación de CC y funciona con un cable USB Mini-B en lugar de uno estándar. Para su potencia, se puede utilizar una fuente de alimentación externa no regulada de 6-20 V en el pin 30 o una fuente de alimentación externa regulada de 5 V en el pin 27. Además, la placa tiene la capacidad de seleccionar automáticamente la fuente de alimentación con el voltaje más alto [14].

El Arduino Nano cuenta con diversas opciones para la comunicación con una computadora, otro Arduino u otros microcontroladores. El microcontrolador ATmega328 que utiliza el Arduino Nano admite la comunicación serial UART TTL (5V), la cual se encuentra disponible en los pines digitales 0 (RX) y 1 (TX) [14].

Tabla 1.1: Características del arduino nano

Tipo	**Característica**
Microcontrolador	ATmega328
Voltaje de funcionamiento	5 V
Voltaje de entrada	7-12V
Pines de entrada analógica	8
Pines de E/S digitales	22 (6 de los cuales son PWM)
El consumo de energía	19mA

1.4.10 Wio-E5 Development Kit

El kit de desarrollo de Wio-E5 es un conjunto compacto de herramientas para el desarrollo de alto rendimiento del Wio-E5 STM32WLE5J. Este kit incluye una placa de desarrollo Wio-E5, una antena (EU868/US915), un cable USB tipo C y un soporte para 2 pilas AA de 3V.

La placa de desarrollo Wio-E5 está equipada con el módulo Wio-E5 STM32WLE5JC, el cual permite el uso del protocolo LoRaWAN en la banda de frecuencia global. Además, ofrece entrada y salida de propósito general (GPIO: General Purpose Input-Output) que

son completamente compatibles con Wio-E5, lo que permite el soporte de varios protocolos e interfaces de datos, como RS-485, Grove, encabezados macho/hembra, entre otros.

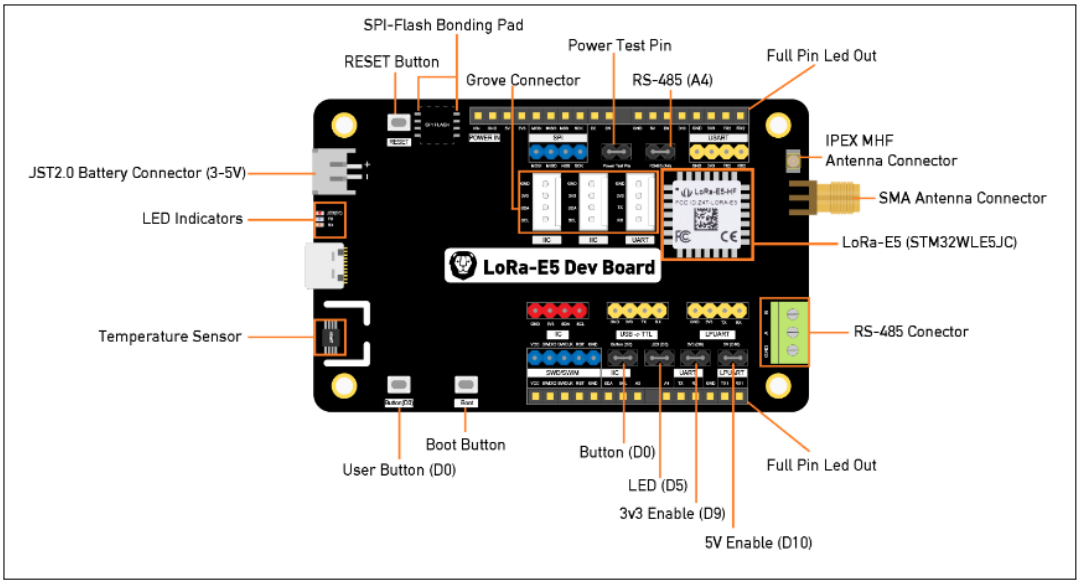

Figura 1.5: Descripción general del hardware [15].

1.4.10.1 Características técnicas de la placa Wio-E5

Esta placa de desarrollo es la primera combinación mundial de chip LoRa RF y MCU en un solo chip pequeño y está certificado por FCC y CE. Está alimentado por el núcleo ARM Cortex-M4 y el chip Semtech SX126X LoRa, es compatible con los protocolos LoRaWAN y LoRa en la frecuencia mundial y las modulaciones (G)FSK, BPSK, (G)MSK y LoRa.

La placa de desarrollo Wio-E5 también tiene varias interfaces, desarrolladas para desbloquear la funcionalidad completa del módulo Wio-E5, la placa de desarrollo Wio-E5 ha sacado 28 pines completos de Wio-E5 y proporciona interfaces optimizadas que incluyen conectores Grove, terminal RS-485, encabezados de pines macho/hembra para que puedan conectar sensores y módulos con diferentes conectores y protocolos de datos.

Así mismo, la placa se puede alimentar fácilmente conectando el portapilas con 2 pilas AA, lo que permite un uso temporal cuando no dispone de una fuente de alimentación externa. Es una placa fácil de usar para pruebas sencillas y creación rápida de prototipos [15].

Tabla 1.2: Características de Wio-E5 Development Kit

Tipo	Característica
Core	32-bit Arm Cortex-M4 CPU, to 48MHz
Interfaces	UART*3, I2C*1, ADC(12-bit)*11, SPI*1, GPIO*6
Modulación	LoRa, (G)FSK, (G)MSK, BPSK
Potencia	Soporta 1.8V -3.6V
Salida de Potencia de RF	Hasta +20.8 dBm a 3.3V

1.4.11 Node Red

Node-RED es una plataforma de programación de código abierto que utiliza una interfaz visual para conectar nodos que representan funciones o tareas. Si bien se utiliza principalmente para la integración de sistemas IoT, también se puede utilizar para desarrollar aplicaciones web, móviles y de escritorio [16].

1.4.11.1 Edición de flujo basada en navegador

La plataforma de programación visual de código abierto Node-RED cuenta con un editor de flujo basado en el navegador que simplifica la interconexión de diferentes procesos mediante una amplia variedad de nodos disponibles en su paleta. Los flujos pueden ejecutarse en tiempo real con solo un clic, y cuenta con un editor de texto mejorado que permite la creación de funciones en JavaScript dentro de la plataforma [16].

1.4.11.2 Basado en Node.js

El entorno de ejecución de Node-RED se basa en Node.js, siendo así más eficiente y adecuado para ser utilizado en el borde de la red en hardware de bajo costo como Raspberry Pi, así como en la nube. La plataforma cuenta con más de 225,000 módulos disponibles en el repositorio de paquetes del nodo, lo que hace que sea fácil ampliar la gama de nodos disponibles para agregar nuevas funcionalidades. [16].

2 METODOLOGÍA

Fig 2.1 permite observar el diagrama de red del funcionamiento final una vez se haya concluido la implementación y configuración de los elementos. En el mismo diagrama se puede evidenciar cuatro partes bien señaladas, estas partes constan de un sensor de señal, Wio-E5 Development Kit, el cual se comunica mediante LoRa RF con un gateway Sentrius RG191 y este utiliza como una red inalámbrica el Wifi conectándose así a la consola TTN en la cual se configura un servidor de red, el cual mediante (MQTT: Message Queuing Telemetry Transport) hace su conexión con los dispositivos finales, en este caso node red en el cual se diseñó un dashboard el cual muestra diagramas de la cantidad de la batería de nodo.

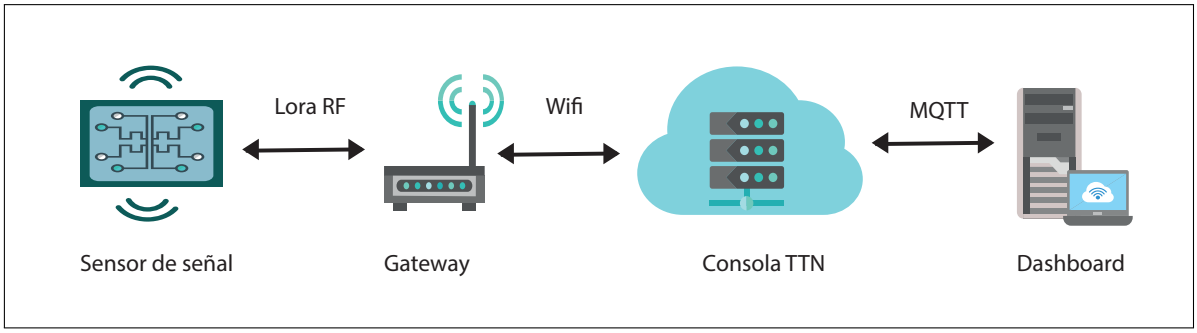

Figura 2.1: Esquemático de la red LoRaWAN a implementar

2.1 PROBLEMÁTICA DE LA IMPLEMENTACIÓN

Para realizar la medición de los valores de la batería se hace uso de la hoja de comandos AT en la cual se usa el comando BAT para medir el parámetro de batería, este comando debería proporcionar el valor real del estado actual de la batería, sin embargo, estos comandos no proporcionan dichos valores. Los comandos AT setean el umbral de funcionamiento, es decir, el valor más pequeño que pueden generar los mismos.

El objetivo de esta implementación es medir el nivel de batería que está ingresando al nodo, es decir, poder ver si la batería está decayendo en el tiempo. En la figura 2.3 se puede observar una parte del esquemático de la placa en el cual el punto A es el valor de la batería requerido a medir, pero al cual no se tiene acceso directo con los comandos AT, estos comandos como tal nos da el valor del punto B, siendo este el valor del VCC el cual siempre nos va a dar 3.26 porque este es el valor que da el regulador de voltaje por el cual pasa [17].

Como solución ante este problema es considerar el uso de un Arduino nano el cual obtiene el voltaje del punto A siento este el valor real que está ingresando al nodo.

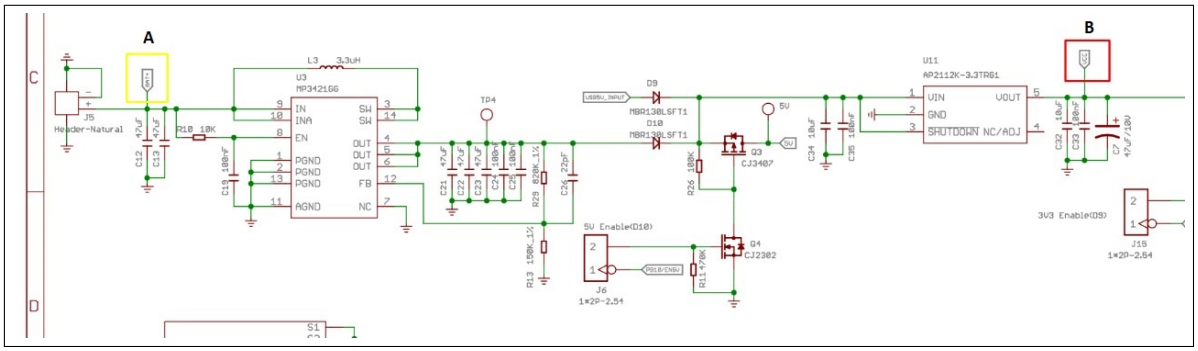

Figura 2.2: Parte del esquemático de la placa Wio-E5 en la cual se indica la problemática de la implementación [18].

2.2 REQUISITOS DE CONFIGURACIÓN DE LA RED

2.2.1 Conexión del RG1xx

El modelo del gateway RG191 viene con una fuente de poder de 12v y 2A, un enrutador y un cable Ethernet. Para instalar las antenas, se debe conectar la antena LoRa al puerto correspondiente a 868/915MHz y las dos antenas Wifi a los puertos de 2.4/5.5GHz. Para la configuración de la banda de frecuencia se utiliza el valor de 915 MHz, siendo esta banda compatible con el nodo Wio-E5.

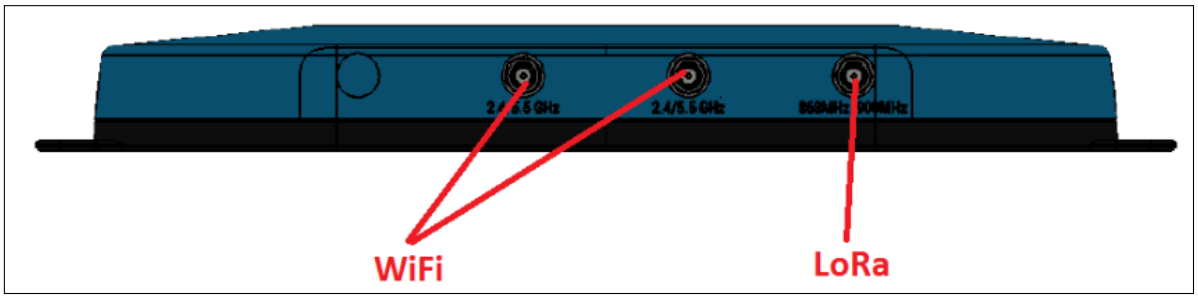

Figura 2.3: Conexión del RG1xx[10].

2.2.1.1 Factor de Spreading

La tecnología LoRa cuenta con 6 niveles de dispersión, desde SF7 hasta SF12, que afectan varios aspectos del sistema. Estos factores, conocidos como factores de propagación, tienen un impacto en la velocidad de transmisión de datos, el tiempo que los datos pasan en el aire, la duración de la batería y la sensibilidad del receptor.

Se ha optado por utilizar un factor de Spreading de 10 chirps debido a su conformidad con los requisitos para la banda de frecuencia US915 y es completamente compatible con las funciones del gateway empleado. Además, gracias a su excelente desempeño, es el factor determinado que utiliza el nodo.

2.3 COMPONENTES IMPLEMENTADOS

2.3.1 Arduino Nano

El Arduino Nano presenta un diseño amigable con dimensiones pequeñas, haciéndolo manipulable ante la implementación con el nodo LoRa-Ed Dev Board. El Arduino Nano tienen pines que permiten una fácil conexión a una placa de pruebas y cuenta con un conector USB Mini-B. [19].

El hardware Arduino tiene incorporado soporte para la comunicación serial en los pines 0 y 1 los cuales van al ordenador a través de una conexión USB. El soporte de origen pasa a través de un componente de hardware (integrado en el chip) llamado UART. Este hardware permite el chip ATmega recibir la comunicación serie incluso mientras trabaja en otras tareas, siempre que haya espacio en la memoria intermedia serie de 64 bytes [19].

2.3.1.1 SoftwareSerial

Para realizar la implementación se hará uso de la biblioteca SoftwareSerial debido a que permite la comunicación serial con los pines digitales de una placa Arduino, utilizando software para replicar la funcionalidad. Es posible tener múltiples puertos seriales de software con velocidades de hasta 115200 bps [20].

Para la implementación del código se realizaron varias funciones que ayudan a capturar los valores requeridos.

2.3.1.2 Función enviarDatos

Esta función intercambia mensajes del Arduino Nano al Wio-E5 mientras se imprime en el monitor serial de arduino el mensaje que será enviado al nodo LoRa, posteriormente se captura la respuesta enviada del Wio-E5 al Arduino Nano mostrándose la respuesta en el monitor serie.

```
//---------Funcion enviarDatos---------
void enviarDatos(String msj){
  Serial.write("Tx:");
  Serial.println(msj);
  node_L5.println(msj);
  delay(2000);
  while(node_L5.available()){
    char inChar = (char)node_L5.read();
    inputString += inChar;
  }
  Serial.print(inputString);
  inputString = "";
}
//-------------------------------------
```

Código 2.1: Función para enviar datos

2.3.1.3 Función inicializarNodeL5

Esta función hace uso de la función enviarDatos para setear las configuraciones requeridas para la conexión correcta entre el nodo LoRa y el gateway. Estas configuraciones son la banda de frecuencias, el spreading factor (SF), el número de canales habilitados, el método de activación, etc.

```
//----Funcion inicializarNodeL5--
void inicializarNodeL5(){
  delay(500);
  Serial.println("Iniciando");
  enviarDatos("AT");
  enviarDatos("AT+DR=US915");
  enviarDatos("AT+CH=NUM,8-15");
  enviarDatos("AT+MODE=LWOTAA");
  }
//------------------------------
```

Código 2.2: Función de inicialización del nodo

2.3.1.4 Función conectNodeL5

Esta función establece la conexión entre el nodo LoRa y el gateway con ayuda del comando AT+JOIN. Así mismo cuenta con un tiempo de guarda para establecer la conexión y así recibir el acuse de acceso, de esta manera lo muestra por pantalla y elimina el contenido del string leído.

```
1  //-----------Funcion conectNodeL5----------
2  void conectNodeL5(){
3    Serial.write("Tx:AT+JOIN");
4    node_L5.println("AT+JOIN");
5    delay(20000);
6    while(node_L5.available()){
7      //Serial.write(node_L5.read());
8      char inChar = (char)node_L5.read();
9      inputString += inChar;
10   }
11   Serial.print(inputString);
12   inputString = "";
13   }
14 //----------------------------------------
```

Código 2.3: Función para la conexión del Wio-E5

2.3.1.5 Función obtenerBat

Esta función captura el VDD del nodo, al inicio se envía un mensaje piloto para obtener un acuse de recibo que tiene diferentes datos, entre estos el VCC. Después se realiza la captura del string de respuesta y se separa el valor de VDD del resto de la cadena, este valor se guarda en la variable nuevoV de tipo String, todo este proceso se documenta en varios mensajes que se imprimen en el monitor serie de arduino.

```
1  //----------------------Funcion obtenerBat-----------------------------
2  void obtenerBat(){
3    inputString="";
4    Serial.println("\nTx:AT+VDD");
5    node_L5.println("AT+VDD");
6    delay(2000);
7    while(node_L5.available()){
```

```
8       char inChar = (char)node_L5.read();
9       inputString += inChar;
10    }
11    Serial.println(inputString);
12    int pos=inputString.indexOf("VDD");
13    if (pos>=0) {
14     Serial.print("se ha detectado la palabra VDD en la posicion ");
15     Serial.println(pos);
16     nuevoV=inputString.substring(pos+4, pos+9);
17     Serial.println(nuevoV);
18      }
19 }
20 //-----------------------------------------------------------------
```

Código 2.4: Función para obtener la batería del nodo

2.3.1.6 Función stringHex

Esta función convierte las cadenas de texto (String) que se ingresan en su argumento de entrada a cadenas hexadecimales (Hex). Es decir, convierte cada carácter de la cadena en un valor hexadecimal y muestra tanto el texto original como la cadena hexadecimal generada a través del monitor serie de Arduino. Al final, la función se encarga de enviar el mensaje final convertido utilizando el comando AT+MSGHEX.

```
1  //------------Funcion string_Hex----------------
2  void string_Hex(String s1){
3  int hex_dec;
4
5  Serial.println();
6  Serial.print("string: ");
7  Serial.println(s1);
8  Serial.println("hexval: ");
9
10 for (const auto &item : s1) {
11   hex_dec = int(item);
12   char hexaDeciNum[100];
13   int i = 0;
14
15   while (hex_dec != 0) {
16     int temp = 0;
```

```
17      temp = hex_dec % 16;
18      if (temp < 10) {
19        hexaDeciNum[i] = temp + 48;
20        i++;
21      }
22      else {
23        hexaDeciNum[i] = temp + 55;
24        i++;
25      }
26      hex_dec = hex_dec / 16;
27    }//54 50 2D 4C 69 6E 6B 5F 44 42 30
28    for (int j = i - 1; j >= 0; j--){
29      // hexaDeciNum[j] = "0x" + hexaDeciNum[j];
30      output += hexaDeciNum[j];
31    }
32  }
33  output="AT+MSGHEX="+output;
34  Serial.print(output);
35  node_L5.println(output);
36  Serial.println("");
37  }
38  //---------------------------------------------
```

Código 2.5: Función convierte de String a hexadecimales

2.3.1.7 Función msjGW5

Esta función esta compuesta por 4 partes, primero se llama a la función obtenerBat para obtener el VDD del nodo, segundo se compone un string que contenga tanto la etiqueta como el valor de VDD en formato JSON, tercero se llama a la función string_Hex ingresando la cadena de caracteres, que fue elaborada en el paso anterior, como argumento de entrada, finalmente se limpia todas las variables globales para la siguiente iteración.

```
1  //------------------------------Funcion msjGW-----------------------------
2  void msjGW(){
3    obtenerBat();
4    Serial.println("");
5    Serial.println("--------------------");
6    Serial.println("Resultado");
7    String res="{\"bat\":"+nuevoV+",\"vcc\":"+a0+"}";
8    Serial.println(res);
```

```
9   Serial.println("--------------------");
10  Serial.println("");
11
12  Serial.println("");
13  Serial.println("--------------------");
14  Serial.println("Resultado");
15  string_Hex(res);
16  //Serial.println(res);
17  Serial.println("--------------------");
18  Serial.println("");
19  delay(5000);
20  output="";
21
22  while(node_L5.available()){
23    node_L5.read();           //Funci n para limpiar el buffer
24  }
25  inputString = "";
26  }
27 //------------------------------------------------------------------------
```

Código 2.6: Función para enviar mensaje al gateway

En la figura 1 se observa la ejecución del código implementado anteriormente.

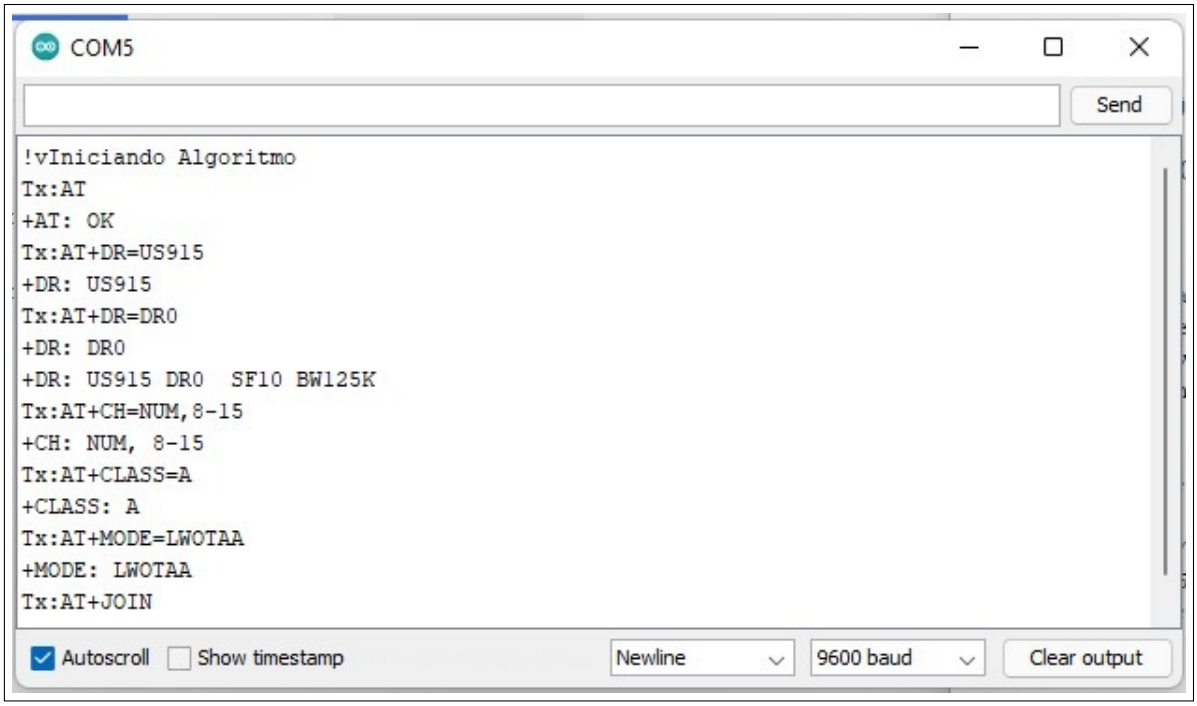

Figura 2.4: Ejecución del código

2.3.2 Wio-E5 Development Kit

Wio-E5 es un chip LoRaWAN con MCU por lo cual existen tres formas principales de utilizar la placa de desarrollo Wio-E5:

1. Desarrollar una aplicación de usuario usando SDK
2. Conectar la placa de desarrollo Wio-E5 a otra placa base a través de UART y controlar mediante comandos AT
3. Conectar la placa de desarrollo Wio-E5 a la PC a través de USB y controlar mediante comandos AT

Se utilizara la tercera opción debido a que se acopla mejor con la implementación sé que se va a realizar, dentro de este proceso hay una función integrada de USB a UART a bordo, simplemente puede conectar la placa de desarrollo Wio-E5 a su PC con un cable USB tipo C y usar el software de comunicación en serie para enviar comandos AT y leer datos de la placa [15].

Las características más relevantes de Wio-E5 Development Kit es su consumo de energía ultrabajo y alto rendimiento, las pruebas sencillas y creación rápida de prototipos, los GPIO completos condujeron a interfaces enriquecidas, que incluyen RS-485, Grove, etc., la compatibilidad con el plan de frecuencia global LoRaWAN y LoRa y finalmente el rango de transmisión de larga distancia a 10 km (distancia ideal en áreas abiertas).

En la figura 2.5 se observa la conexión realizada entre el Arduino nano y el nodo LoRa-E5 Dev Board, los cuatro pines de conexión son:

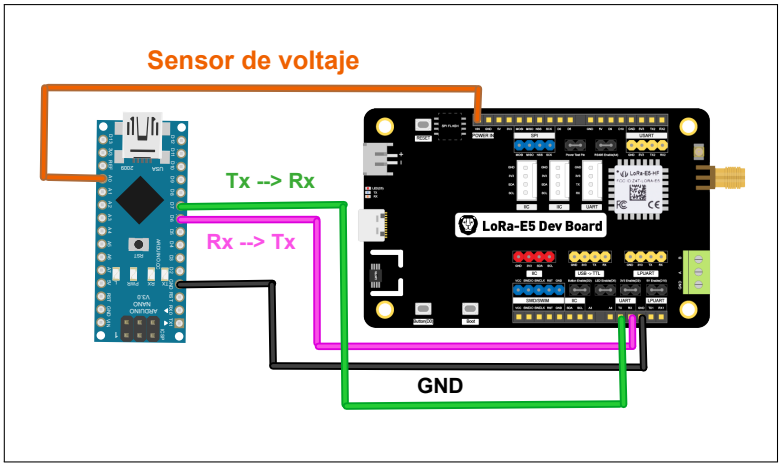

Figura 2.5: Conexión de los pines entre el Arduino y la el Wio-E5

Tabla 2.1: Características relevantes para la configuración del Wio-E5

Tipo	Características
Suministro de voltaje	3-5V (batería) / 5V (USB tipo C)
Voltaje de salida	ES 3V3 / 5V
Salida de potencia	Hasta +20,8dBm a 3,3V
Frecuencia	UE868/ US915/ AU915 /AS923
Corriente	Corriente de suspensión del módulo LoRa-E5 tan baja como 2.1uA (modo WOR)

2.3.2.1 Pines a conectar

En la tabla 2.2 se indica los pines UART a conectar para realizar la comunicación serial con el Arduino, el pin VIN tiene como función actuar como el sensor de voltaje con el cual se obtendrá el valor real que está ingresando al nodo.

Tabla 2.2: Pines conectados entre la Arduino nano y el Wio-E5

Cable	Arduino Uno	Nodo Lora-E5
	Pin GND	Pin GND
	Pin D6	UART RX
	Pin D7	UART TX
	Pin A0	VIN

2.3.3 Comandos AT

Los comandos AT son una serie de comandos que inician siempre con las letras AT, estos comandos son enviados por medio de un puerto serie por medio de un Arduino o Conversor USB Serial para enviar los comandos desde la PC. Así mismo se hace uso de estos comandos cuando se requiere probar si la conexión del módulo está bien, este es un comando ficticio al igual que otros comandos AT comunes [18].

2.3.3.1 Comandos de configuración

- **AT**

  Se utiliza para comprobar si la conexión del módulo está correcta. Este es un comando de prueba [18].

- ✧ **Comando:** AT
- ✧ **Retorno:** +AT: OK

❒ **ID**

Verifica el ID del módulo LoRaWAN o cambiar el ID. ID permite obtener las diferentes credenciales del nodo [18]. Leer formato de identificación:

- ✧ **Comando:** AT+ID
- ✧ **Retorno:**

  +ID: DevAddr 26:0C:CB:0C

  +ID: DevEui, 2C:F7:F1:20:32:30:4D:E1

  +ID: AppEui, 80:00:00:00:00:00:00:06

❒ **DR**

Para establecer la tasa de datos del módem LoRaWAN AT, se debe utilizar DRx definido por LoRaWAN [18].

- ✧ **Comando:** AT+DR=US915
- ✧ **Retorno:** +DR: US91
- ✧ **Comando:** AT+DR=DR0
- ✧ **Retorno:**

  +DR: DR0

  +DR: US915 DR0 SF10 BW125K

❒ **CH**

Permite habilitar los primeros 15 canales de comunicación. Si se activan todos los canales disponibles en la banda utilizada, se puede llegar a tener un máximo de 96 canales [18].

- ✧ **Comando:** AT+CH=NUM,0-15
- ✧ **Retorno:** +CH: NUM, 0-15

❒ **CLASS**

Este comando podría cambiar el modo de operación del módem LoRaWAN a una clase diferente (A/B/C). Por defecto, el módem LoRaWAN trabaja en modo clase A cuando se enciende, pero el usuario puede cambiar manualmente a las clases B o C según sus necesidades [18].

- ✧ **Comando:** AT+CLASS=A
- ✧ **Retorno:** +CLASS: A

❒ **MODE**

Este comando permite elegir el modo de operación, que incluyen los modos LWABP, LWOTAA y TEST. Es importante tener en cuenta que el módem LoRaWAN solo puede trabajar en un modo a la vez [18].

- ✧ **Comando:** AT+MODE=LWOTAA:
- ✧ **Retorno:** +MODE: LWOTAA

❒ **JOIN**

Cuando el modo OTAA está activado, se puede utilizar el comando JOIN para conectarse a una red previamente conocida [18].

- ✧ **Comando:** AT+JOIN
- ✧ **Retorno:**

  +JOIN: Start

  +JOIN: NORMAL

  +JOIN: Network joined

  +JOIN: NetID 000013 DevAddr 26:0C:CB:0C

  +JOIN: Done

2.3.4 Comandos para obtención y envío de datos

❒ **BAT**

Establece el valor del nivel de batería, este valor está disponible de 0 $\sim$ 255, el cual siempre medirá 3.26 V porque es el valor del VCC [18].

- ✧ **Comando:** AT+LW=BAT
- ✧ **Retorno:** +LW: BAT, 255

❒ **VDD**

Se obtiene el voltaje de suministro, valor devuelto en la unidad 0.01V [18].

- ✧ **Comando:** AT+VDD
- ✧ **Retorno:** +VDD: 3.36V

❒ **MSGHEX**

Este comando se utiliza para transmitir una trama en formato hexadecimal que no requiere una confirmación por parte del servidor [18].

✧ **Comando:** AT+MSGHEX="Mensaje en hexadecimal"

✧ **Retorno:**

+MSGHEX: Start

+MSGHEX: Done

2.3.5 The Thing of Networks

Esta aplicación web se utiliza para registrar aplicaciones, dispositivos finales o gateways, su función principal es monitorear el tráfico de la red o configurar opciones relacionadas con la red [21].

2.3.5.1 Consola

Para registrar una aplicación o gateway se realiza el siguiente procedimiento:

1. Ingresar a la página web https://www.thethingsnetwork.org/ e iniciar sesión con el perfil ya creado anteriormente y seleccionar la opción de **Console**.

Figura 2.6: Ingreso a la consola TTN

2. Seleccionar el país en el que se encuentra ubicado y seleccionar un clúster de las 3 opciones que se indican en la figura 2.26, en este caso es North America 1:

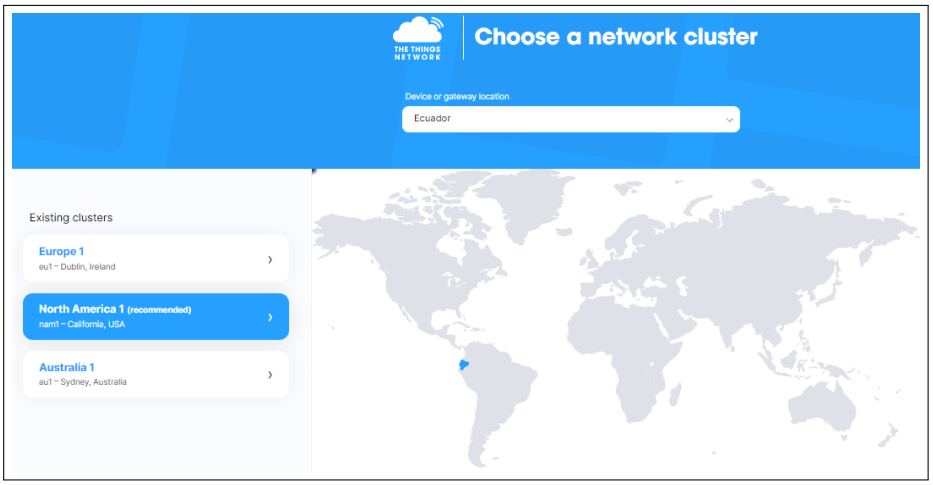

Figura 2.7: Selección del país y cluster correspondiente

3. Se tienen las opciones de configurar los gateways y las aplicaciones, en este caso se seleccionará la opción **Create an application**. Se debe tomar en cuenta que cada gateway puede ser registrado una sola vez en TTN, para volver hacer uso del mismo gateway se lo debe eliminar de la cuenta en la cual está ingresado, en el caso de las aplicaciones se puede registrar varios nodos terminales, así mismo pueden estar registrados en varias aplicaciones diferentes.

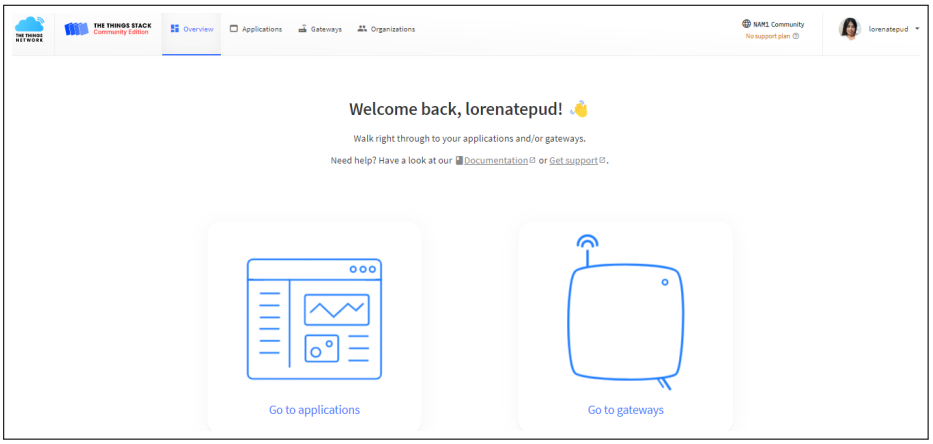

Figura 2.8: Selección de la creación de aplicación o gateway

2.3.5.2 Registro del GW RG191

Para establecer el funcionamiento en TTN: The Thing of Networks se debe registrar dos procesos, el primero es el gateway y el segundo es el nodo de la red privada LoRaWAN, de esta manera se obtendrá una comunicación exitosa.

El procedimiento para registrar el gateway es el siguiente:

1. En la figura 2.8 seleccionar la opción **Go to gateways** y en la nueva ventana se muestran los gateways registrados, así como la opción de registrar nuevos gateways.

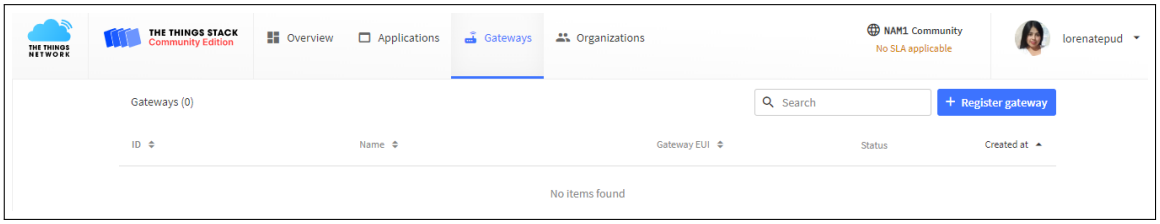

Figura 2.9: Listado de gateways

2. Hacer clic en **Register Gateway** para registrar un nuevo gateway, se abrirá una nueva ventana en la cual pide ingresar el **Gateway EUI**, este valor se encuentra en la etiqueta de información del gateway con el nombre de DevEUI como se indica en la figura 2.11 y dar clic en **Confirm**.

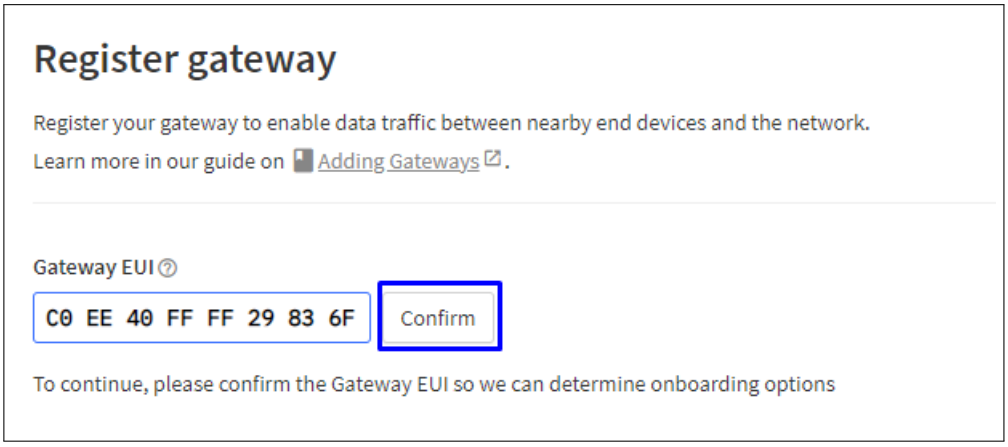

Figura 2.10: DevEUI valor hexadecimal de 8 bytes

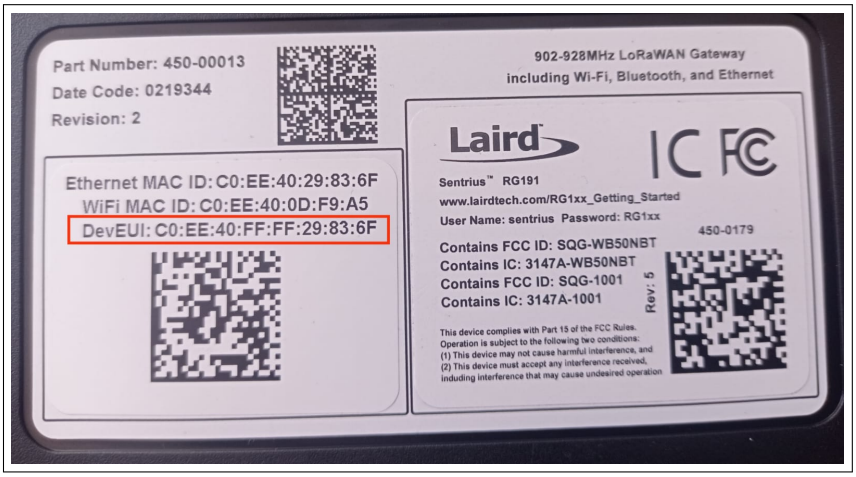

Figura 2.11: DevEUI valor hexadecimal de 8 bytes

3. Después de confirmar el Gateway EUI se abre una ventana donde se debe llenar la información del Gateway ID y el plan de frecuencia, estos campos son obligatorios, el Gateway name es un campo opcional, finalmente se da clic en **Register gateway** como se muestra en la figura 2.12.

Figura 2.12: Registro del Gateway

2.3.5.3 Crear una aplicación

1. En la figura 2.8 seleccionar la opción **Go to applications** y en la nueva ventana se muestran las aplicaciones creadas. Para crear una nueva aplicación, hacer clic en **Create application**.

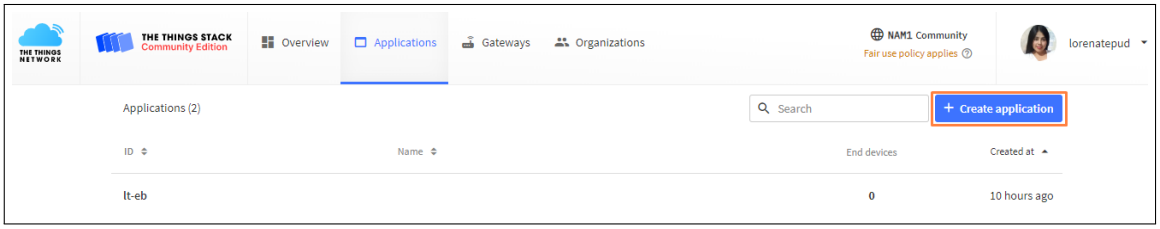

Figura 2.13: Lista y creaación de aplicaciones

2. Se abrirá una nueva ventana en donde se debe llenar campos obligatorios como la información de un identificador único para la aplicación (Application ID), mientras que los campos de nombre y descripción son opcionales. Se debe tomar en cuenta que si elcampo de nombre de la aplicación (Aplication name) se deja en blanco, este utilizará el mismo nombre que el Application ID.

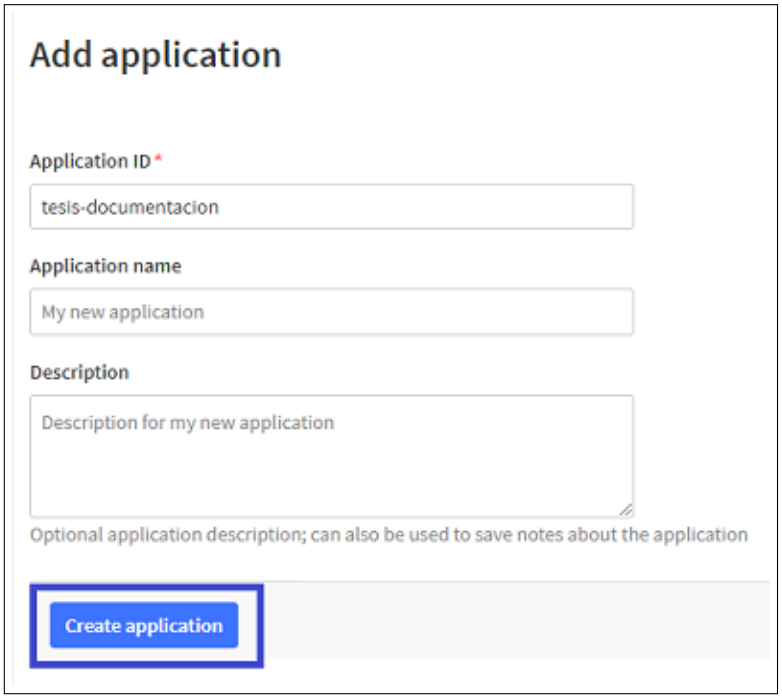

Figura 2.14: Creación de aplicación

3. Dar clic en **Register end device** para agregar los nodos terminales y así empezar con el envío de datos.

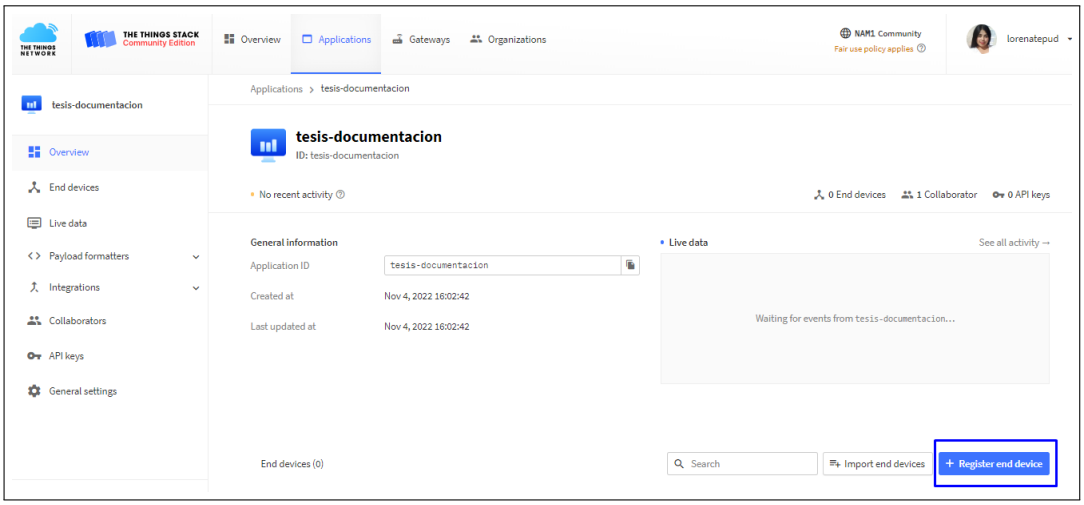

Figura 2.15: Aplicación creada

4. En la nueva ventana llenar los campos obligatorios como el plan de frecuencias, la versión de LoRaWAN y la versión de parámetros regionales con la información que se muestra en la figura 2.16.

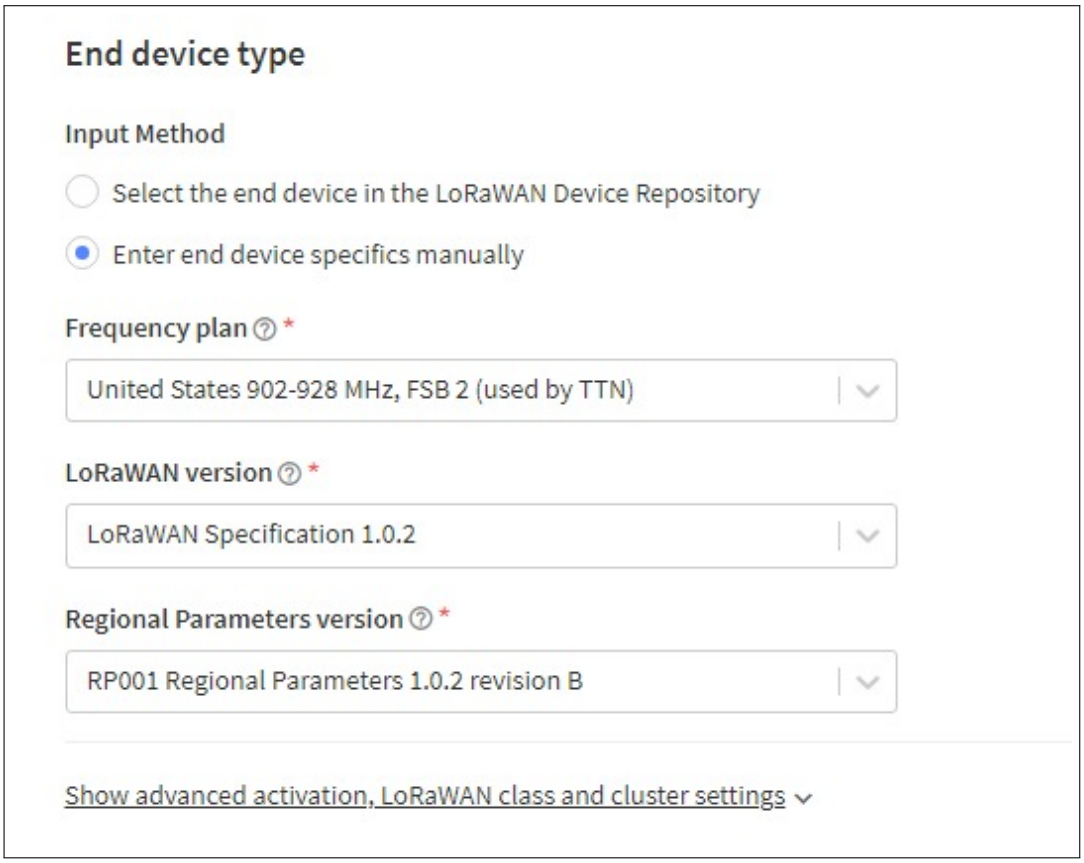

Figura 2.16: Selección de la creación de aplicación

5. Para llenar el campo de **Provisioning information** y posteriormente los campos de DevEUI, AppEUI y AppKey se debe hacer uso del comando **AT+ID** en la consola serial de la placa Wio-E5, el ejecutarlo se obtendrá el resultado de la figura 2.17.

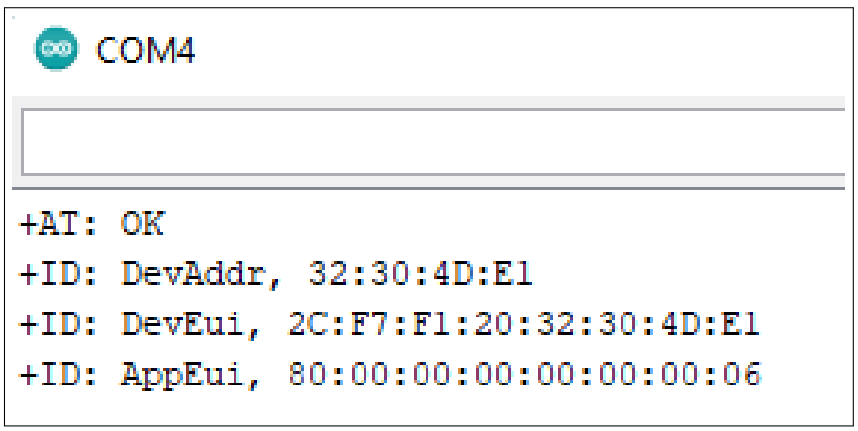

Figura 2.17: Resultado del comando AT+ID

6. Los campos DevEUI, AppEUI y AppKey son llenados con los valores obtenidos anteriormente. Al llenar el campo de DevEUI automáticamente se llenará el campo de ID del dispositivo final (End device ID). Finalmente, hacer clic en **Register end device**.

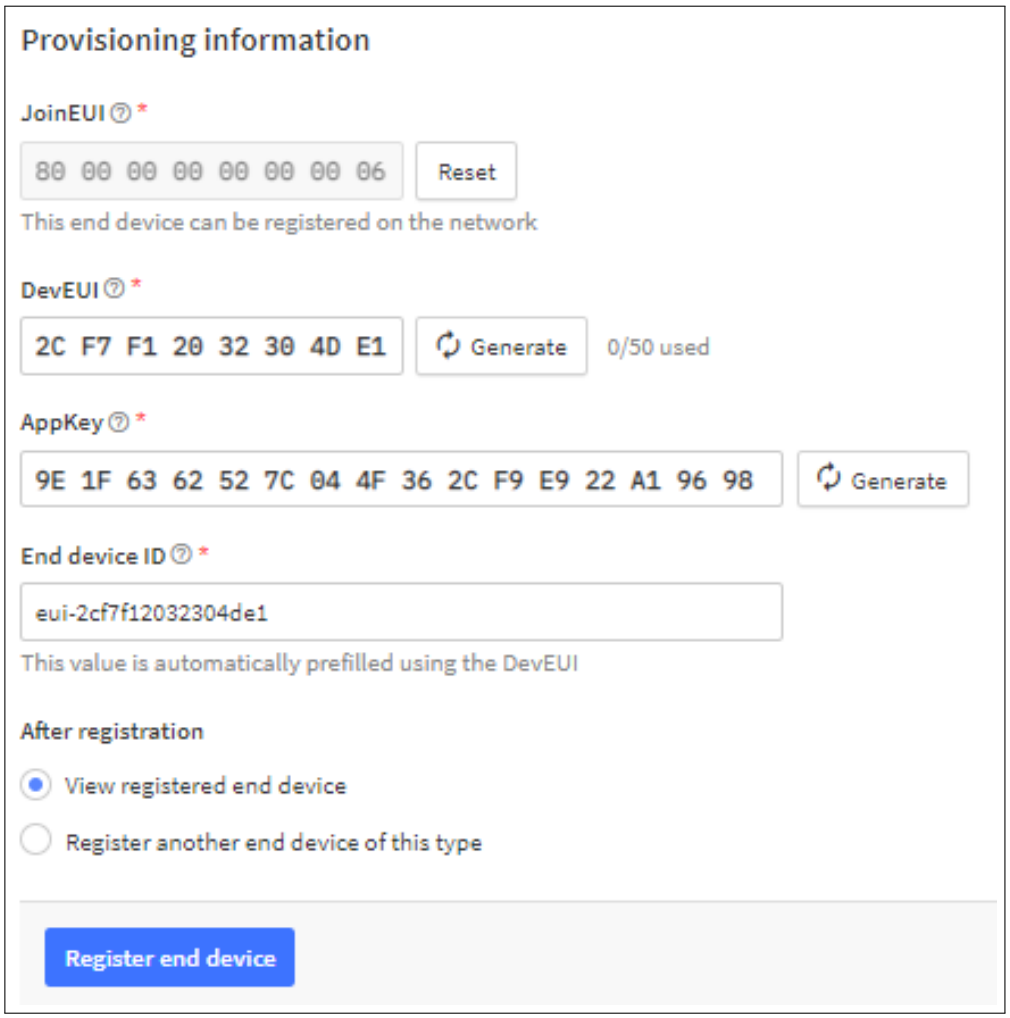

Figura 2.18: Selección de la creación de aplicación

7. Para el correcto funcionamiento se requiere realizar dos configuraciones adicionales:

   a) Para los mensajes de llegada al gateway se debe generar un decodificador y Json Parser.

   b) Para comunicarse con la aplicación desarrollada en Node-Red se debe habilitar un servidor MQTT a través de las integraciones.

   Ambas configuraciones se pueden realizar accediendo a las opciones de Payload formatters → Uplink e Integrations → MQTT, que se encuentra en la parte izquierda del servidor.

2.3.5.4 Uplink

1. Se da clic en Uplink y la opción de formatter type se debe seleccionar la opción de Custom Javascript formatter, el cual es el decodificador utilizado por defecto, el mismo indica en pantalla los datos obtenidos por un parser que lee un String y entrega un Json.

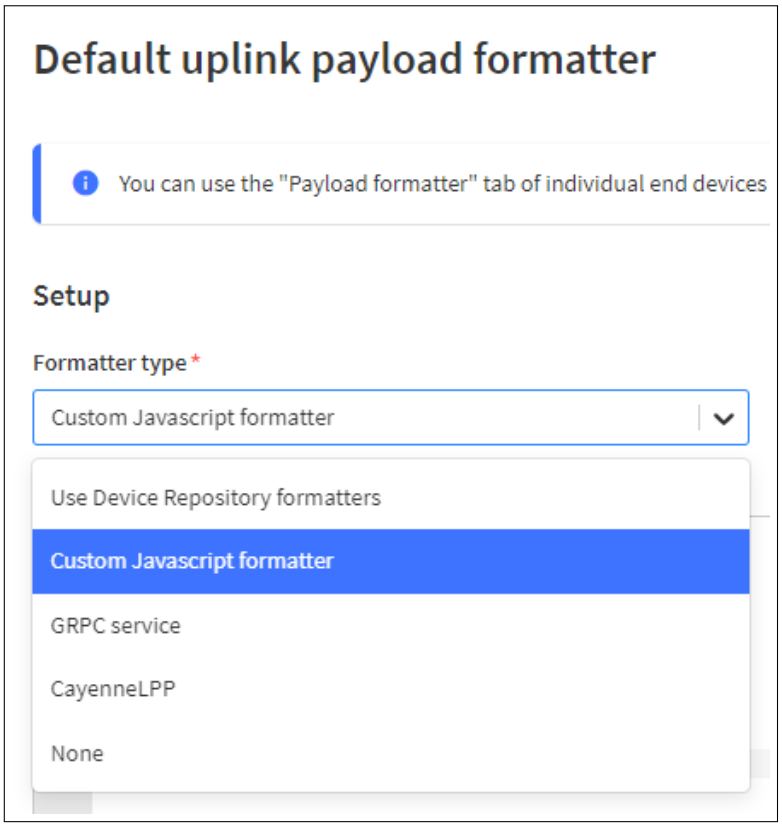

Figura 2.19: Configuración del uplink formatter

```
function Decoder(bytes, port) {
  var deco = {};
if (port==8){
 return {
      Values: JSON.parse(String.fromCharCode.apply(null, bytes)) //Json
    parser
  };
  }
  return deco;
}
```

Código 2.7: Uplink

2.3.6 Node Red

Se ha optado por utilizar Node-Red como la aplicación final debido a su fácil programación con flujos y la capacidad de crear un panel de monitoreo dinámico a partir de los datos recibidos de un servidor MQTT. También se pueden utilizar bibliotecas de Node JS y modificar la interfaz de usuario de la aplicación con codificación en Angular JS.

Node-Red se puede instalar en diferentes sistemas operativos, como Windows, macOS o Linux. Así mismo se puede encontrar instrucciones de instalación en la página oficial de Node-Red. En este caso, para el funcionamiento de la aplicación se ha levantado un servidor local Node-red en Windows.

Para instalar node red en Windows se realiza el procedimiento indicado en [22]. Al finalizar con la descarga e instalación se abre una terminal o línea de comandos y se escribe el comando node-red. Si todo el procedimiento se realizó correctamente, deberá aparecer un mensaje que indica que Node-RED se está ejecutando, así como se indica en la figura 2.20:

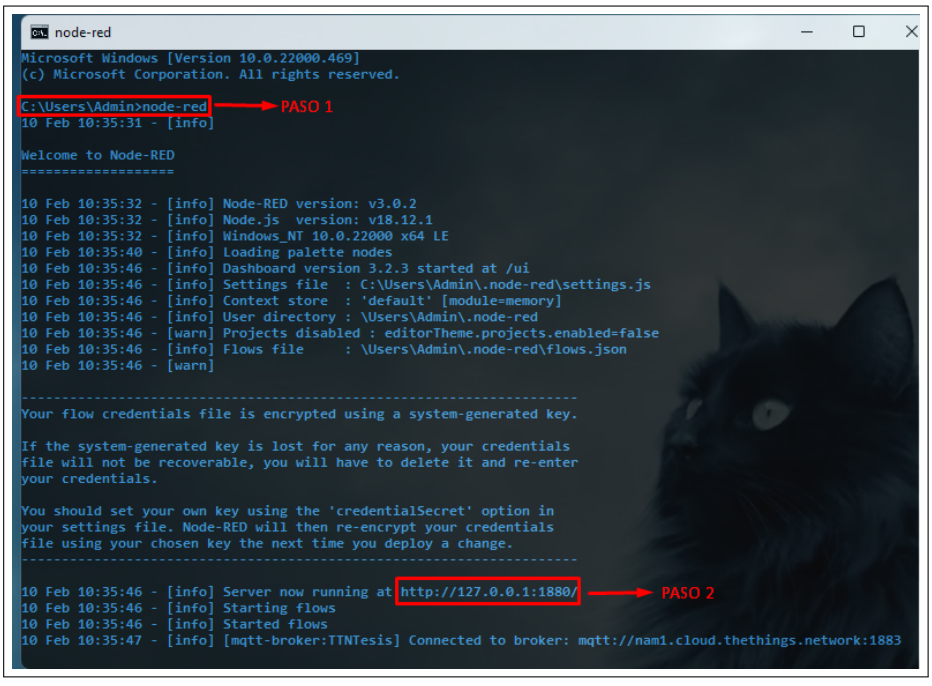

Figura 2.20: Inicialización de Node Red en cmd de Windows

Para abrir el editor de flujos en Node-Red, después de iniciar su ejecución, se debe abrir un navegador y acceder a la dirección local en el puerto 1880. Esto puede hacerse ingresando a la dirección URL 127.0.0.1:1880.

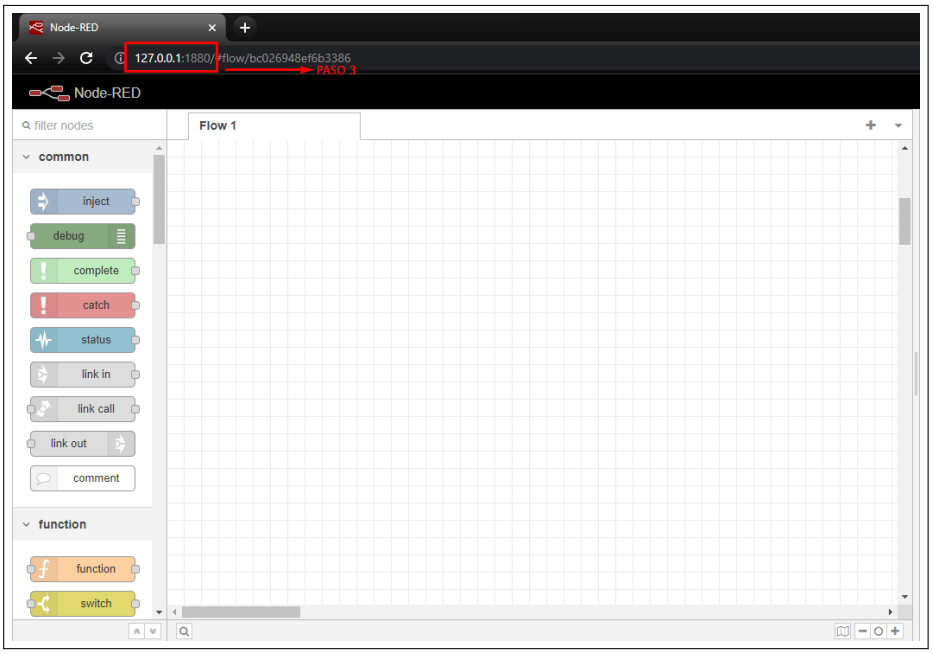

Figura 2.21: PASO 3:

❒ Instalar la paleta node-red dashboard:

1. Ingresar a opciones en la barra superior derecho a lado de Deploy.
2. Seleccionar Manage Palette.
3. Hacer clic en Palette → Nodes → en filter nodes *node-red-dashboard*
4. Dar clic en install

❒ En la página de inicio del editor de Node-RED, existe una paleta de nodos en la parte izquierda. Para crear un flujo se puede arrastrar y soltar nodos desde la paleta al área de trabajo principal.

❒ Para conectar los nodos, únicamente se hace clic en el punto de conexión del nodo y se arrastra el cable hasta el punto de conexión del siguiente nodo.

❒ Para configurar los nodos se hace doble clic en un nodo para abrir su configuración y así modificar sus propiedades.

❒ Una vez que se ha creado y configurado los nodos del flujo, hacer clic en el botón de "Deploy"que se encuentra en la esquina superior derecha del editor para ejecutar el flujo.

En este caso se desarrollará en tres fases: conexión, filtrado de datos y presentación de datos.

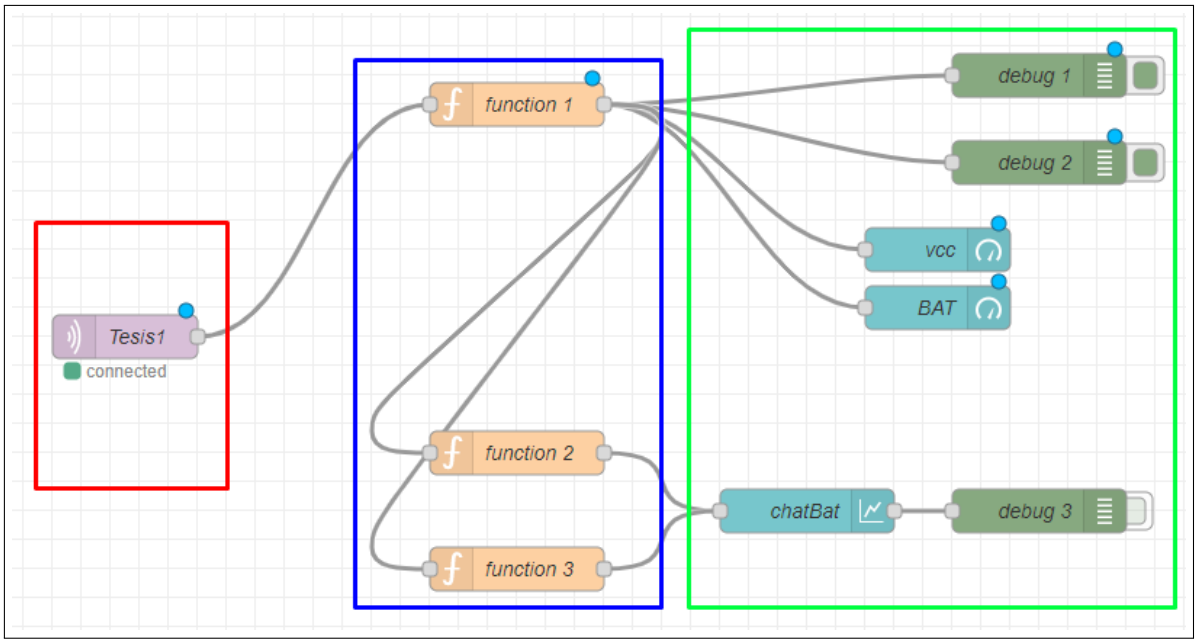

Figura 2.22: Hoja de trabajo de Node red

2.3.6.1 Fase de conexión

Se configura la conexión al servidor, para esto se especifica el servidor nam1.cloud.thethings.network en el puerto 1883, en la pestaña de seguridad se ingresa el usuario y la API Key definidas para la integración Mqtt configurada previamente

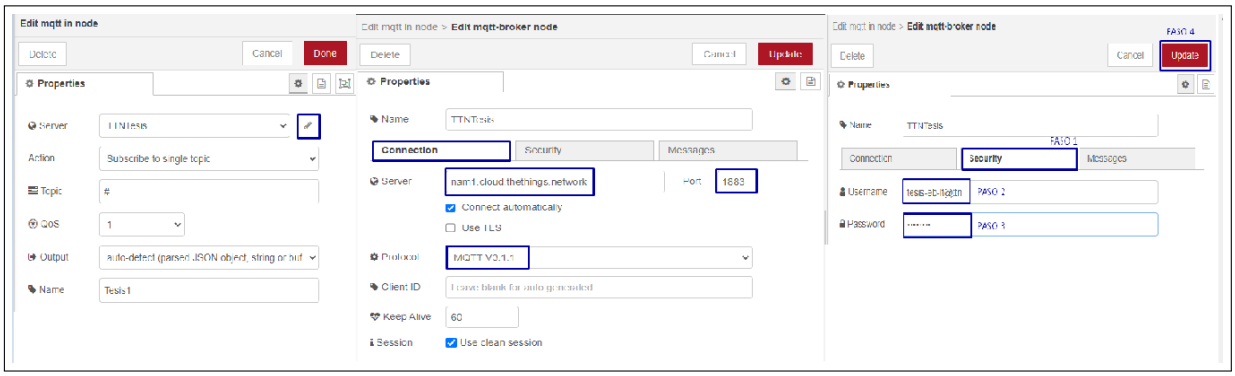

Figura 2.23: Configuración de la entrada de datos por MQTT

2.3.6.2 Fase de filtrado de datos

Para la creación de esta fase de hace uso de la creación de tres funciones implementadas en JavaScript. El formato de mensaje de entrada de las funciones es de tipo JSON. Las funciones son las siguientes:

Esta función accede a los mensajes que contienen datos codificados y a los mensajes de control se forma individual.

```
if (msg.payload.uplink_message.decoded_payload) {
    return msg;
}
```

Código 2.8: Función GetMessage

Esta función extrae la información del dato llamado Bat del mensaje codificado que fue enviado desde el gateway.

```
msg.payload = msg.payload.uplink_message.decoded_payload.Values.bat;
msg.topic = "Bat";

return msg;
```

Código 2.9: Función GetBat

Esta función ingresa a los metadatos que son enviados desde el gateway y captura el dato de VCC.

```
msg.payload = msg.payload.uplink_message.decoded_payload.Values.vcc;
msg.topic = "vcc";

return msg;
```

Código 2.10: Función GetVcc

2.3.6.3 Presentación de datos

Para la presentación de datos se hace uso de la paleta de nodos que se encuentran en la parte izquierda de la interfaz de usuario, cada nodo realizan una función específica y son de diferentes tipos como de entrada, salida, comunicación, entre otros. El procedimiento para la creación de un Dashboard es el siguiente:

1. En la parte derecha de la interfaz de usuario, junto a la opción de configuración se encuentra una opción que al dar clic despliega varias opciones.

2. Seccionar dashboard.

3. Seleccionar la pestaña de Layout que es una de las tres pestañas que ofrece esta opción de Dashboard, las otras dos son: Site, Theme.

4. Seleccionar la opción $+\mathbf{Tab}$ y añadir un nombre para el mismo. Para este caso el nombre seleccionado es Dashboard BAT.

5. Crear un nuevo grupo para lo cual se coloca el cursor sobre la opción $+\mathbf{group}$. En este caso serán creados dos grupos: BAT y Función del tiempo.

6. Para editar la información de cada grupo se ingresa a $\mathbf{edit}$, aquí se abre una nueva ventana en la cual se debe ingresar el nombre del grupo y el tab al que va a pertenecer, opcionalmente se puede editar el ancho predeterminado en la opción $\mathbf{Width}$.

7. Seguir este mismo procedimiento para cada grupo de widgets que se vaya a mostrar en el dashboard.

8. Para guardar las configuraciones dar clic en $\mathbf{Deploy}$.

Para el desarrollo del Dashboard final se han establecido 2 grupos principales en los que se mostrará la información, los mismos son Batería y Función del tiempo.

❒ **Grupo Función del tiempo**

Para la creación de este grupo se utiliza un nodo tipo chart dentro del cual se debe realizar la configuración del grupo al que va a pertenecer, el tamaño, el tipo (es la traza de los valores de entrada en un gráfico, el mismo puede ser de líneas basadas en el tiempo, de barras (vertical u horizontal) o un gráfico circular Cada valor de msg.payload de entrada se convertirá en un número). El gráfico se escalará automáticamente a cualquier valor recibido, donde el eje X define una ventana de tiempo o un número máximo de puntos y el eje y establece valores máximos y mínimos de la batería [16].

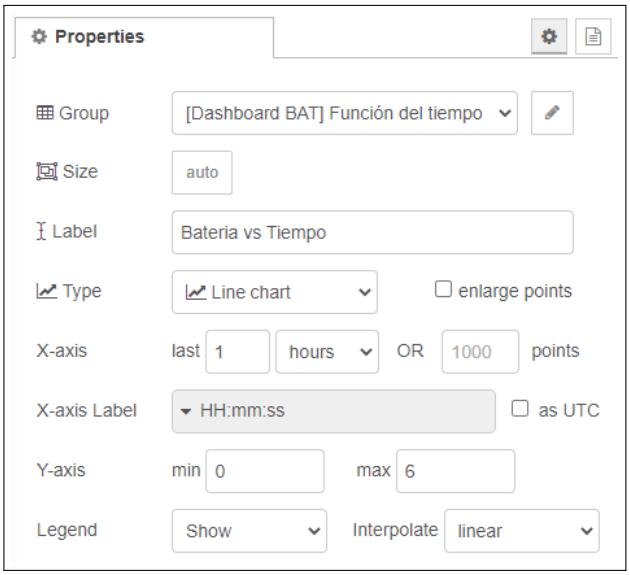

Figura 2.24: Configuración de nodo chatBat

❒ Grupo Batería

En este grupo se agregan dos nodos del tipo gauge (indicador), dentro de esta configuración se busca en msg.payload un valor numérico y se formatea de acuerdo con el formato de valor (Value formtat) definido, que luego se puede formatear usando filtros angulares. Se pueden especificar los colores de cada uno de los 3 sectores, los colores deben especificarse en formato hexadecimal (rrggbb) [16]. Para este caso se utilizó estos nodos para representar el valor del BAT (valor real que está ingresando al nodo) y el VCC (valor de 3.26 V).

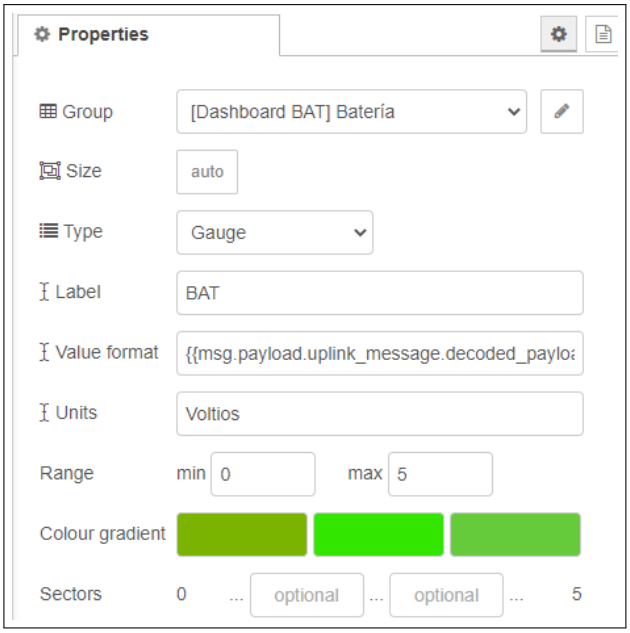

Figura 2.25: Configuración de nodo VCC

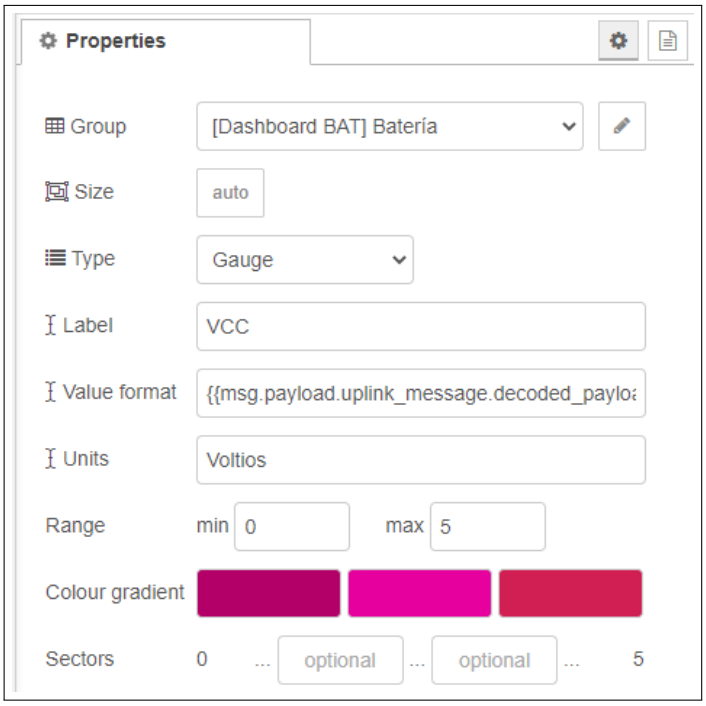

Figura 2.26: Configuración de nodo VCC

En la figura 2.27 se observa el resultado esperado, el cual es un Dashboard con los dos grupos definidos anteriormente, el primero de ellos muestra un gráfico de líneas basado en el tiempo del comportamiento de BAT en la última hora, y los otros dos gráficos representan el valor actual del BAT y del VCC. Para ingresar al dashboard implementado se debe acceder al navegador web y se coloca en la barra de url 127.0.0.1:1880/ui.

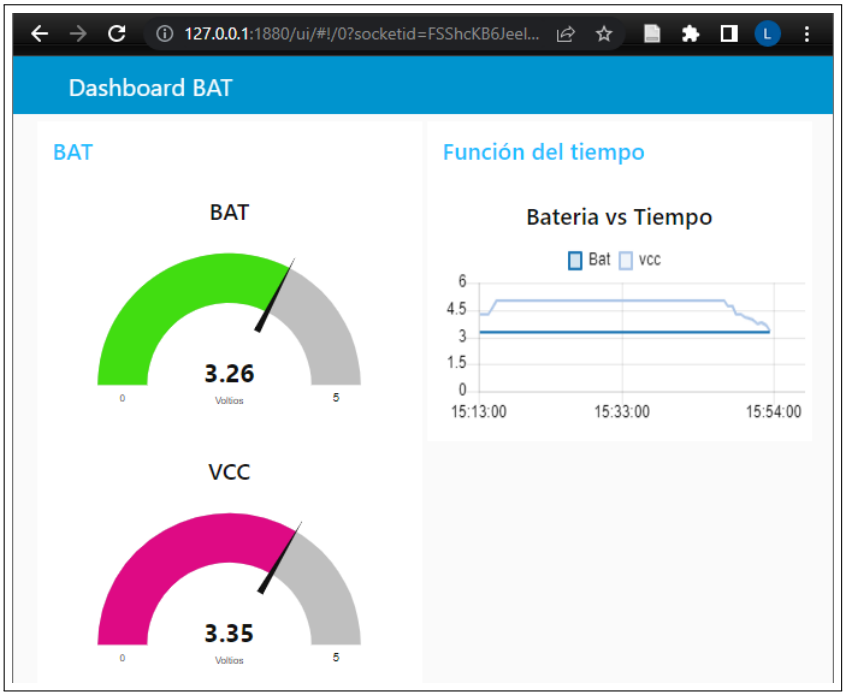

Figura 2.27: Dashboard implementado

2.3.7 Esquemático electrónico del prototipo

Después de haber implementado un prototipo funcional en software, se requiere realizar la implementación física de este nodo sensor, para lo cual, se ha decidido usar los siguientes materiales:

- ❒ Una baquelita perforada
- ❒ Zócalos ubicados específicamente para adaptarse a la forma de las dos tarjetas usadas
- ❒ Un interruptor
- ❒ Una entrada de batería de dos pines
- ❒ Una porta pilas AA
- ❒ Dos borneras de dos pines
- ❒ Segmentos de cable unifilar

En la figura 2.28 se muestra el esquemático del prototipo a implementarse, en el mismo se establece la conexión entre las borneras y los pines A1 y D9 del Arduino, además se muestra la conexión entre el puerto serial del Arduino con el de la Wio-E5. Finalmente, se muestra que el Arduino como la Wio-E5 son alimentadas desde la misma batería.

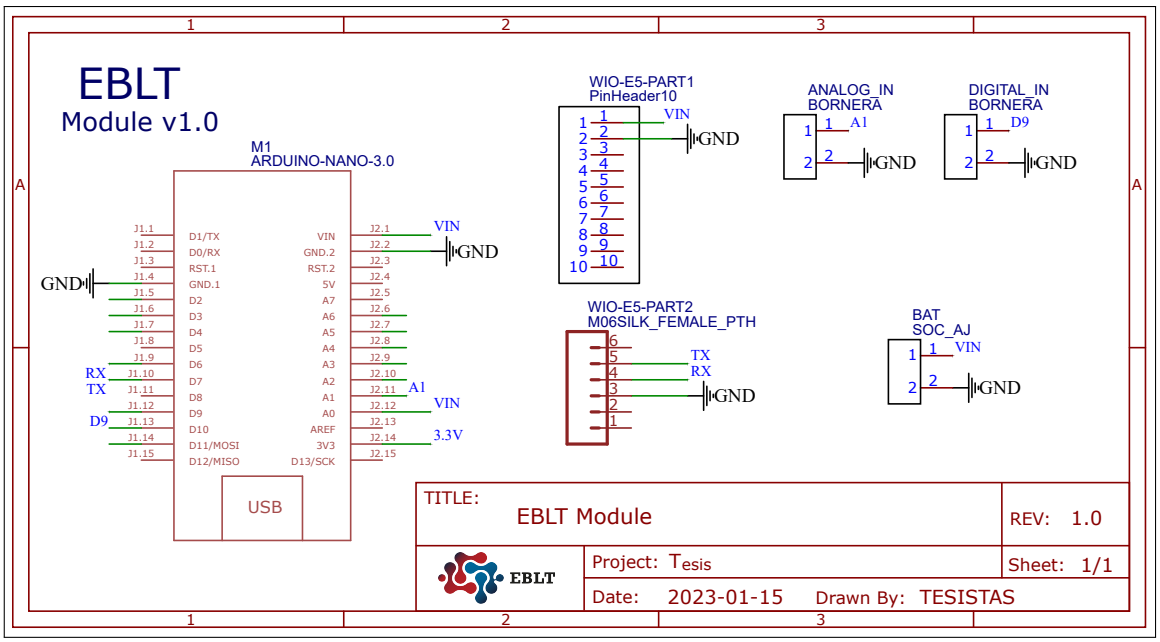

Figura 2.28: Esquemático electrónico del prototipo

3 RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

En el presente capítulo se muestra la implementación física del prototipo realizado, así como los resultados obtenidos en diferentes escenarios de funcionamiento, para lo que se utilizará una fuente variable para saber que valores de voltaje están alimentando al nodo y que valores de voltaje mide el sensor. Luego, se presentarán los resultados tanto en el ámbito de aplicación como a través del monitor serial.

3.1 RESULTADOS

3.1.1 Implementación del prototipo

En la figura 2.28 se puede observar el diseño del circuito electrónico que permite conectar una placa Arduino Nano con la Wio-E5, el mismo fue implementado en una baquelita perforada. Se procura que el prototipo ocupe la menor cantidad de espacio para que el mismo sea presentado en una carcasa diseñada e impresa en 3D. En la figura 3.10 se puede observar el resultado final.

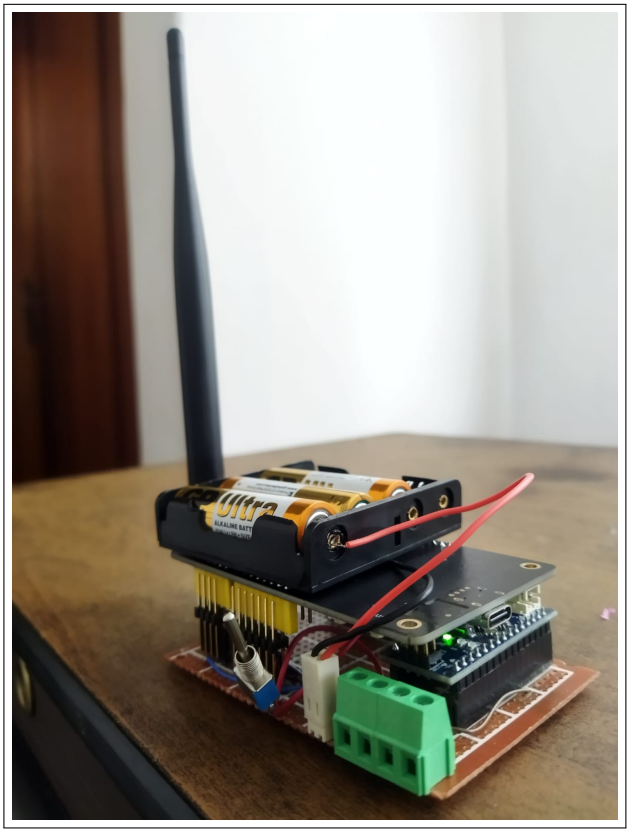

Figura 3.1: Prototipo implementado

Para tener una mejor protección del prototipo se ha decidido crear una carcasa en 3D con la ayuda de la herramienta Tinkercad. Esta carcasa está diseñada a las medidas exactas del prototipo, las cuales son: longitud de 99 mm, un ancho de 73 mm y una altura de 53 mm. De esta manera solo se exponen los puertos para un sensor digital y un sensor analógico.

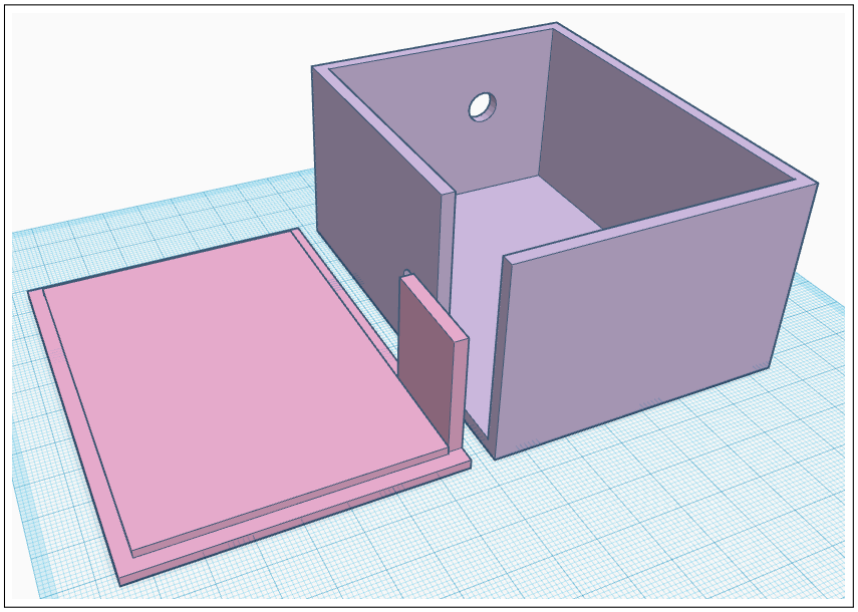

Figura 3.2: Estructura diseñada en Tinkercad

Después de imprimir la estructura en 3D, se procede a colocar el prototipo dentro de la carcasa y se cierra de tal manera que el prototipo queda bien asegurado, así mismo para su mejor manipulación del encendido y apagado se tiene al descubierto el interruptor. En la figura 3.10 se puede observar el prototipo final.

Figura 3.3: Prototipo implementado

3.1.2 Funcionamiento del sensor y monitoreo por el puerto serial

Como ya se mencionó anteriormente en las funciones implementadas en Arduino Nano, esta placa permite supervisar el funcionamiento del nodo mediante el puerto serial. Así, todos los comandos utilizados pueden ser mostrados en la pantalla, las mediciones realizadas por el nodo sensor, el mensaje a enviar y el mensaje en formato hexadecimal para su posterior conversión a formato JSON.

```
Iniciando
Tx:AT
+AT: OK
Tx:AT+DR=US915
+DR: US915
Tx:AT+DR=DR0
+DR: DR0
+DR: US915 DR0  SF10 BW125K
Tx:AT+CH=NUM,0-15
+CH: NUM, 0-15
Tx:AT+CLASS=A
+CLASS: A
Tx:AT+MODE=LWOTAA
+MODE: LWOTAA
Tx:AT+JOIN+JOIN: Start
+JOIN: NORMAL
+JOIN: Network joined
Tx:AT+VDD

se ha detectado la palabra VDD en la posicion 16
 3.26
Tx:AT+MSG
XWIN1, RSSI -8, SNR 13.0

se ha detectado la palabra RSSI en la posicion 7
 -8
--------------------
Resultado
{"bat": 3.26,"rssi": -8,"vcc":4.32}
--------------------
--------------------
Resultado

string: {"bat": 3.26,"rssi": -8,"vcc":4.32}
hexval:
AT+MSGHEX=7B22626174223A20332E32362C2272737369223A202D382C22766363223A342E33327D
```

Figura 3.4: Primera iteración y Monitoreo por el puerto serial del nodo en su inicialización

3.1.3 Funcionamiento del nodo sensor

En esta sección se definirá un escenario de prueba, obteniendo distintos resultados con los cuales se realiza un análisis de contraste entre los datos extraídos del voltaje que alimenta al nodo y los valores que mide el sensor.

3.1.3.1 Escenario de prueba

Para realizar las pruebas se hizo uso de una fuente variable para saber que valores de voltaje están alimentando al nodo al mismo tiempo que se mide que valores de voltaje del sensor. Se considera además que el nodo sensor se ha configurado para funcionar como clase A y con SF de 10, adicional a esto la banda de frecuencias definida es la US915 con frecuencia central en los 915MHz.

3.1.4 Resultados obtenidos

Los resultados fueron obtenidos a partir de las mediciones realizadas en el nodo sensor Arduino y en la fuente variable. Los datos medidos están disponibles en el Anexo 2, en total se recopilaron 37 datos y con ayuda de la herramienta de Excel se graficaron los mismos, como se muestra en la figura 3.5.

Figura 3.5: Valores medidos y valores obtenidos del Arduino

3.1.5 Error porcentual y error porcentual medio

Para calcular el error porcentual se hace uso de la ecuación 3.1 y para el error porcentual medio la ecuación 3.2, estas dos medidas se determinan a partir de los 37 valores obtenidos en la medición de muestras que se indica en el anexo 2. En las figuras 3.6 y 3.7 se puede observar el resultado de las dos medidas.

$$E\,\% = \frac{|\mathbf{V_{medido}} - \mathbf{V_{arduino}}|}{\mathbf{V_{medido}}} \times \mathbf{100}\,\% \tag{3.1}$$

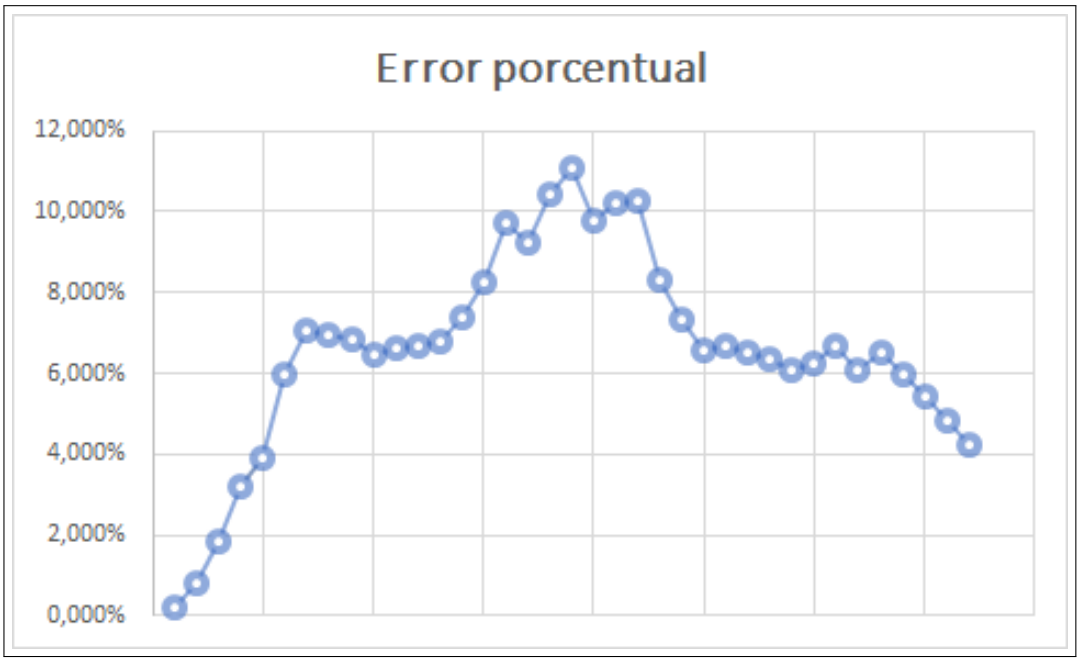

Figura 3.6: Error porcentual

Luego de analizar los errores porcentuales, se puede concluir que el sensor tiene un rendimiento excelente, ya que la mayoría de los errores individuales son menores al 12 %. Además, al calcular el valor promedio del error porcentual medio, se obtiene un valor cercano al 7 % como se observa en la figura 3.7, lo que garantiza que las lecturas del sensor del nodo son precisas. Con la certeza de la precisión de las lecturas, se procede a verificar que las destinas funcionalidades de monitoreo que fueron agregadas a la aplicación desarrollada funcionen adecuadamente.

$$E\,\%_{medio} = \frac{\sum_1^n E\,\%}{n} \tag{3.2}$$

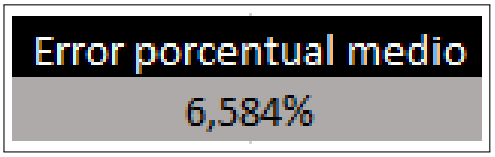

Figura 3.7: Error porcentual medio

3.1.6 Funcionamiento de la aplicación de monitoreo

Al desarrollar la aplicación de monitoreo, es importante mostrar los resultados de manera clara y concisa para que el usuario pueda comprender y utilizar los datos generados. La forma en que se presentan los resultados es de una forma fácil de entender y visualmente atractiva, para esto se han incluido tres gráficos que mejoran la comprensión y seguimiento de los datos presentados en el panel de control. Estos gráficos son:

- **Batería vs. Tiempo:** Es un histórico de los dos valores obtenidos de BAT y VCC, a medida que el tiempo transcurre el histórico va graficando los valores que va obteniendo.
- **BAT:** Es el valor de la batería obtenido mediante el comando AT, VDD.
- **VCC:** Es el valor que se obtiene del sensor implementado en el pin A0 del Arduino.
- Adicional a las gráficas se implementa un mensaje de alerta, el cual aparece en una ventana emergente cuando la batería es menor 2.85V.

Para observar de mejor manera el funcionamiento de la aplicación de monitoreo se hace captura de tres tiempos diferentes, de esta manera se pueda apreciar como la aplicación va tomando los distintos valores y como los va almacenando.

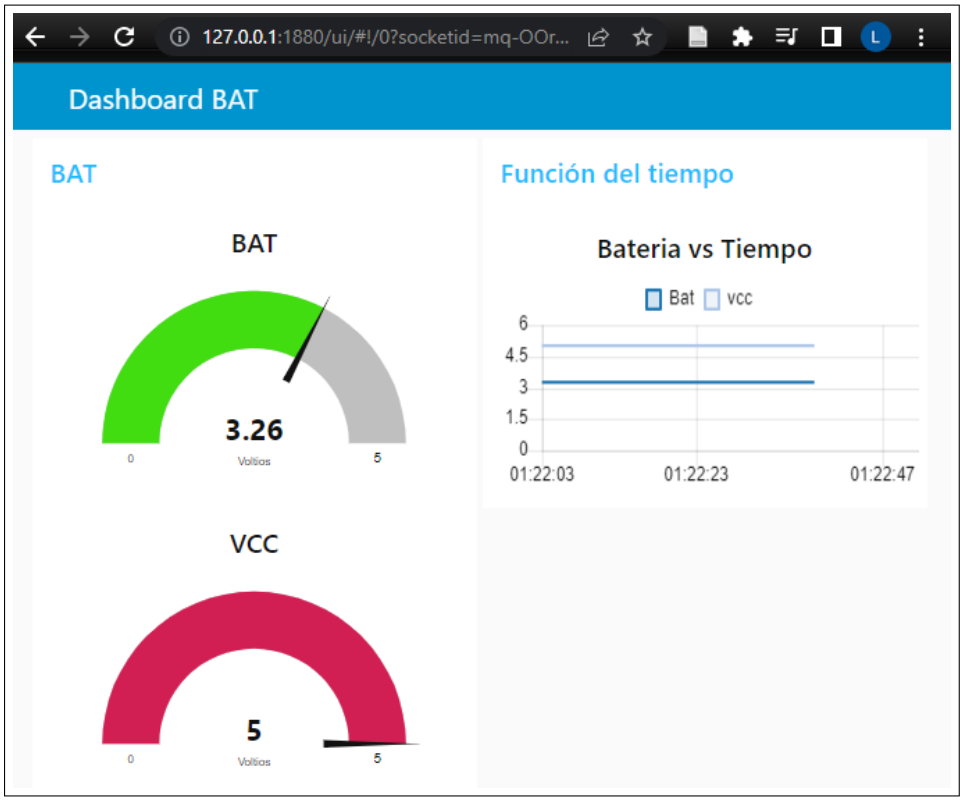

Figura 3.8: Nivel de batería en 44s

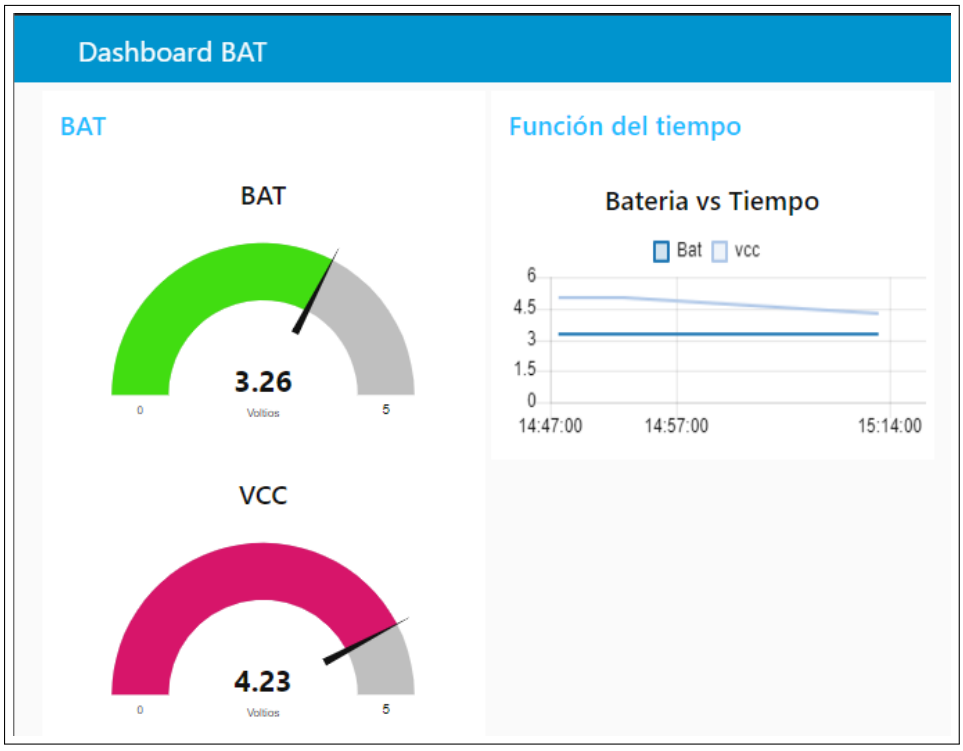

Figura 3.9: Nivel de batería en 27 minutos

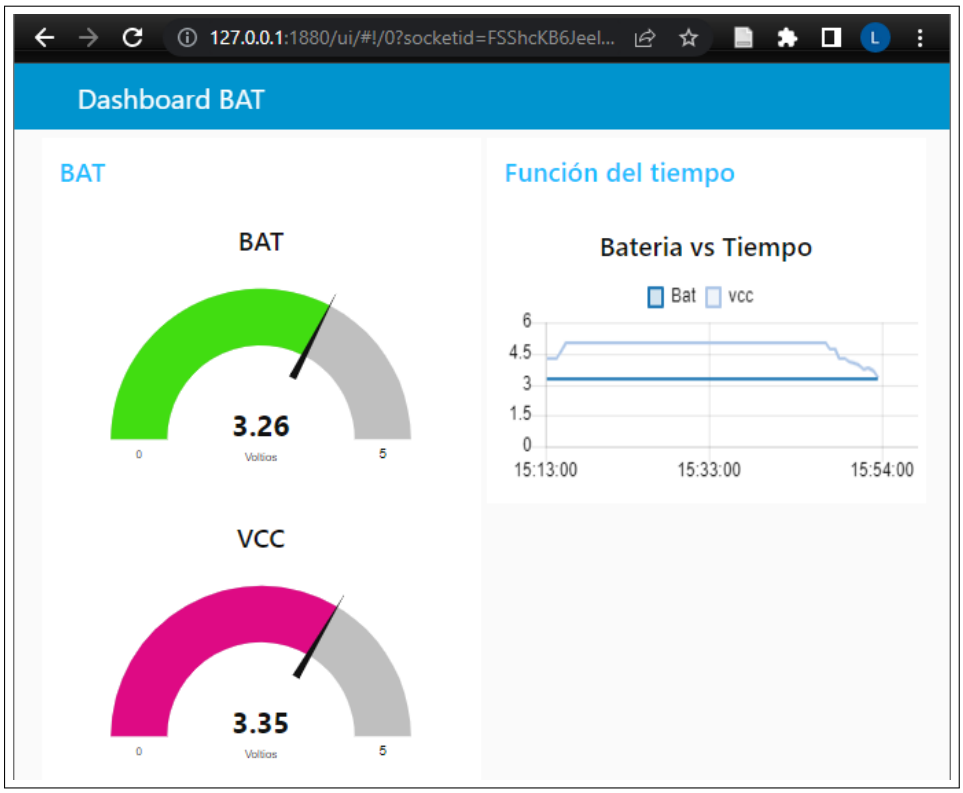

Figura 3.10: Nivel de batería en 41 minutos

3.1.7 Sistema de alerta

Este sistema de alerta temprana se lo implemento con el objetivo de mejorar la visualización del sistema y de evitar la desconexión del mismo, para esto se ha incorporado un mensaje de alerta que se muestra únicamente en los casos en que se detecten valores excesivamente bajos de batería o cuando la batería se ha agotado y el nodo se ha apagado. Así mismo, este mensaje se dispara cuanto las medidas de batería del nodo son menores a 2.85V, como se muestra en la figura 3.12.

El template utilizado para la configuración del mensaje de alerta es el siguiente:

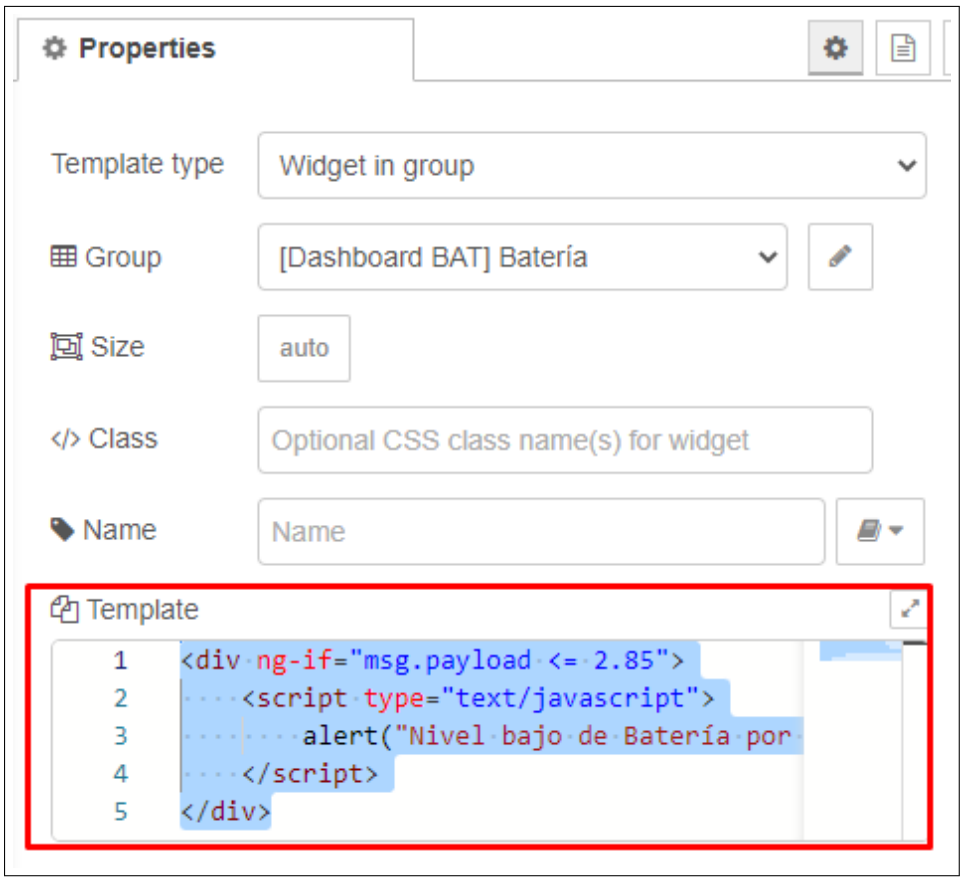

Figura 3.11: Configuración del mensaje de alerta

Figura 3.12: Mensaje de alerta

3.2 CONCLUSIONES

- En el presente trabajo, se realizó el monitoreo del nivel de batería en un nodo LoRaWAN dentro de una red punto-punto. Para este fin, se realizó el diseño e implementación de un sensor de voltaje. Adicionalmente, se utilizó una fuente variable para saber que valores de voltaje están alimentando al nodo y que valores de voltaje mide el sensor. De esta forma se logró realizar una comparativa entre estos dos valores a fin de llevar a cabo el monitoreo del nivel de batería en un Dashboard en el cual se puede visualizar estos valores.

- Mediante la previa investigación al diseño del nodo sensor, se pudo establecer de forma correcta la configuración principal de la red. Esta configuración incluye aspectos como la banda de frecuencia, ancho de banda, factor de spreading, canales de comunicación y el método de activación. Todo esto permitió diseñar un nodo sensor eficiente y confiable que cumple con los requerimientos del sistema de monitoreo LoRaWAN.

- La red diseñada cumple con las especificaciones de una red LoRaWAN funcional. Se ha logrado integrar el módulo transceiver LoRa con un gateway LoRaWAN para transmitir los datos al servidor de red TTN. Además, se ha desarrollado un dashboard en Node-Red para visualizar e interpretar los datos de forma fácil y eficiente. A través de este sistema, se puede monitorear y supervisar el nivel de batería del nodo en tiempo real, lo que permite tomar medidas preventivas en caso de que el nivel de energía disminuya y se necesite reemplazar la batería.

- Basándonos en las configuraciones previas del gateway usado en trabajos de integración curricular anteriores, la unica conexión posible es el modo de velocidad DR0 con un factor spreading de 10 y un ancho de banda de 125 kHz, ofreciendo así el mejor rendimiento para el prototipo diseñado. Basándonos en la hoja de datos se pudo determinar que no fue necesario realizar pruebas en los distintos modos, ya que los resultados de las métricas relevantes para este trabajo son similares entre ellos.

- La elección de utilizar Angular.JS en Node-Red fue una decisión acertada, ya que permitió una fácil implementación del código y una mayor facilidad en el monitoreo del sistema. Esto mejoró la experiencia del usuario al proporcionar una visualización clara y concisa de los datos recopilados por el sistema.

3.3 RECOMENDACIONES

- El importante analizar el esquemático de la placa Wio-E5 Development Kit para tener una idea clara del funcionamiento interno de la misma, en este caso se determinó que el nivel de batería que se requería medir no era posible debido a que el punto de medida se encuentra después de un regulador de voltaje entonces siempre que se obtenga el valor con el comando AT se va a obtener un valor de 3.26 V que es el valor que mide el regulador de voltaje, entonces al utilizar un Arduino se obtiene el voltaje real que está entrando al nodo.

- El desarrollo de estos nodos es continua y rápida, por lo cual se debe utilizar la última versión de la documentación, la misma se genera en tiempos relativamente cortos, por lo cual es indispensable definir el método por el cual se va a obtener los resultados, ya sea programando la tarjeta directamente o usando los comandos AT.

- Cada nodo tiene su hoja de datos, por lo cual es importante asimilar la información proporcionada como es la lista de comandos AT y sus parámetros, ya que el uso adecuado de estos documentos puede evitar retrasos e inconvenientes, como los experimentados al intentar utilizar el comando BAT fuera del modo TEST de la Wio-E5. El problema se presentó porque el comando solo proporciona el valor VDD de la placa y ese valor siempre es 3.26 V, por lo cual no nos ayuda a controlar los niveles de batería del mismo.

4 REFERENCIAS BIBLIOGRÁFICAS

[1] A. Osorio, M. Calle, J. D. Soto y J. E. Candelo-Becerra, «Routing in LoRaWAN: Overview and Challenges,» *IEEE Communications Magazine*, vol. 58, n.º 6, págs. 72-76, jun. de 2020, Conference Name: IEEE Communications Magazine, ISSN: 1558-1896. DOI: 10.1109/MCOM.001.2000053.

[2] J. Haxhibeqiri, E. De Poorter, I. Moerman y J. Hoebeke, «A Survey of LoRaWAN for IoT: From Technology to Application,» en, *Sensors*, vol. 18, n.º 11, pág. 3995, nov. de 2018, ISSN: 1424-8220. DOI: 10.3390/s18113995. dirección: http://www.mdpi.com/1424-8220/18/11/3995 (visitado 22-07-2022).

[3] L. A. T. M. Workgroup, *What is LoRaWAN?* Nov. de 2015.

[4] B. S. Chaudhari, M. Zennaro y S. Borkar, «LPWAN Technologies: Emerging Application Characteristics, Requirements, and Design Considerations,» en, *Future Internet*, vol. 12, n.º 3, pág. 46, mar. de 2020, ISSN: 1999-5903. DOI: 10.3390/fi12030046. dirección: https://www.mdpi.com/1999-5903/12/3/46 (visitado 03-10-2022).

[5] K. Mekki, E. Bajic, F. Chaxel y F. Meyer, «A comparative study of LPWAN technologies for large-scale IoT deployment,» en, *ICT Express*, vol. 5, n.º 1, págs. 1-7, mar. de 2019, ISSN: 2405-9595. DOI: 10.1016/j.icte.2017.12.005. dirección: https://www.sciencedirect.com/science/article/pii/S2405959517302953 (visitado 03-10-2022).

[6] M. A. Ertürk, M. A. Aydın, M. T. Büyükakkaşlar y H. Evirgen, «A Survey on LoRaWAN Architecture, Protocol and Technologies,» en, *Future Internet*, vol. 11, n.º 10, pág. 216, oct. de 2019, ISSN: 1999-5903. DOI: 10.3390/fi11100216. dirección: https://www.mdpi.com/1999-5903/11/10/216 (visitado 22-07-2022).

[7] M. A. M. Quimbita y C. E. P. Salvador, «Evaluación de pasarela LoRa/LoRaWAN en entornos urbanos,»

[8] «Reference.» (), dirección: https://www.thethingsindustries.com/docs/reference/ (visitado 28-12-2022).

[9] «End device activation,» The Things Network. Section: lorawan. (), dirección: https://www.thethingsnetwork.org/docs/lorawan/end-device-activation/ (visitado 29-12-2022).

[10] «Laird RG1xx,» The Things Network. (), dirección: https://www.thethingsnetwork.org/docs/gateways/laird/ (visitado 29-12-2022).

[11] «Network Architecture,» The Things Network. (), dirección: https://www.thethingsnetwork.org/docs/network/architecture/ (visitado 29-01-2023).

[12] «Network Security,» The Things Network. (), dirección: https://www.thethingsnetwork.org/docs/network/security/ (visitado 31-01-2023).

[13] «The Things Network Console,» The Things Network. (), dirección: https://www.thethingsnetwork.org/docs/network/console/ (visitado 31-01-2023).

[14] «Arduino Nano,» Arduino Official Store. (), dirección: https://store.arduino.cc/products/arduino-nano (visitado 08-01-2023).

[15] «Wio-E5 Development Kit - Seeed Wiki.» (), dirección: https://wiki.seeedstudio.com/LoRa_E5_Dev_Board/ (visitado 05-01-2023).

[16] «Nodo-RED.» (), dirección: https://nodered.org/ (visitado 07-02-2023).

[17] ShaoZongPan, *LoRa-E5 Dev Board v1.0*, 2 de abr. de 2021. dirección: https://files.seeedstudio.com/products/113990934/LoRa-E5%5C%20Dev%5C%20Board%5C%20v1.0.pdf (visitado 05-01-2023).

[18] S. Technology, *LoRa-E5 AT Command Specification*. dirección: https://files.seeedstudio.com/products/317990687/res/LoRa-E5%5C%20AT%5C%20Command%5C%20Specification_V1.0%5C%20 (visitado 11-01-2023).

[19] «Nano | Arduino Documentation.» (), dirección: https://docs.arduino.cc/hardware/nano (visitado 06-01-2023).

[20] «SoftwareSerial Library | Arduino Documentation.» (), dirección: https://docs.arduino.cc/learn/built-in-libraries/software-serial (visitado 06-01-2023).

[21] «Console.» (), dirección: https://www.thethingsindustries.com/docs/getting-started/console/ (visitado 12-01-2023).

[22] O. M. F. Alzate. «Instalar node red en windows,» http://codigoelectronica.com. (), dirección: http://codigoelectronica.com/blog/instalar-node-red-en-windows (visitado 10-02-2023).

5 ANEXOS

ANEXO 1

```
#include<SoftwareSerial.h>
SoftwareSerial node_L5(7, 6); //
//--------Funciones----------
void inicializarNodeL5();
void enviarDatos(String msj);
void conectNodeL5();
void string_Hex(String s1);
void obtenerBat();
//---------------------------

//------Variables globales---
unsigned long inicio = 0;
String inputString = "";
String output;
String nuevoV;
float a0;
//---------------------------

void setup() {
  Serial.begin(9600);
  node_L5.begin(9600);
  inicializarNodeL5();
  conectNodeL5();
}

void loop() {
  a0=analogRead(A0)*0.004888;
   unsigned long nowT = millis();
  if (nowT - inicio >= 35000) {
    inicio = nowT;
    //enviarDatos("AT+LW=BAT");
    msjGW();
  }
}
//---------------Funcion inicializarNodeL5-----------------------------
void inicializarNodeL5(){
```

```
37    delay(500);
38    Serial.println("Iniciando");
39    enviarDatos("AT");
40    enviarDatos("AT+DR=US915");
41    enviarDatos("AT+CH=NUM,8-15");
42    enviarDatos("AT+MODE=LWOTAA");
43    }
44  //-------------------------------------------------------------------------
45
46  //----------------------Funcion conectNodeL5-----------------------------
47  //-------------------------------------------------------------------------
48  void conectNodeL5(){
49    Serial.write("Tx:AT+JOIN");
50    node_L5.println("AT+JOIN");
51    delay(20000);
52    while(node_L5.available()){
53      //Serial.write(node_L5.read());
54      char inChar = (char)node_L5.read();
55      inputString += inChar;
56    }
57    Serial.print(inputString);
58    inputString = "";
59    }
60  //-------------------------------------------------------------------------
61
62
63  //------------------Funcion enviarDatos--------------------------------
64  //-------------------------------------------------------------------------
65  void enviarDatos(String msj){
66    Serial.write("Tx:");
67    Serial.println(msj);
68    node_L5.println(msj);
69    delay(2000);
70    while(node_L5.available()){
71      char inChar = (char)node_L5.read();
72      inputString += inChar;
73    }
74    Serial.print(inputString);
75    inputString = "";
76  }
77  //-------------------------------------------------------------------------
78
```

```
79
80
81
82 //------------------------------Funcion msjGW-----------------------------
83 //------------------------------------------------------------------------
84 void msjGW(){
85   obtenerBat();
86   Serial.println("");
87   Serial.println("--------------------");
88   Serial.println("Resultado");
89   String res="{\"bat\":"+nuevoV+",\"vcc\":"+a0+"}";
90   Serial.println(res);
91   Serial.println("--------------------");
92   Serial.println("");
93
94   Serial.println("");
95   Serial.println("--------------------");
96   Serial.println("Resultado");
97   string_Hex(res);
98   //Serial.println(res);
99   Serial.println("--------------------");
100  Serial.println("");
101  delay(5000);
102  output="";
103
104  while(node_L5.available()){
105    node_L5.read();           //Funci n para limpiar el buffer
106  }
107  inputString = "";
108  }
109 //------------------------------------------------------------------------
110
111
112 //-------------------------Funcion string_Hex-----------------------------
113 //------------------------------------------------------------------------
114 void string_Hex(String s1){
115 int hex_dec;
116
117 Serial.println();
118 Serial.print("string: ");
119 Serial.println(s1);
120 Serial.println("hexval: ");
```

```
121
122 for (const auto &item : s1) {
123   hex_dec = int(item);
124   char hexaDeciNum[100];
125   int i = 0;
126
127   while (hex_dec != 0) {
128     int temp = 0;
129     temp = hex_dec % 16;
130     if (temp < 10) {
131       hexaDeciNum[i] = temp + 48;
132       i++;
133     }
134     else {
135       hexaDeciNum[i] = temp + 55;
136       i++;
137     }
138     hex_dec = hex_dec / 16;
139   }//54 50 2D 4C 69 6E 6B 5F 44 42 30
140   for (int j = i - 1; j >= 0; j--){
141     // hexaDeciNum[j] = "0x" + hexaDeciNum[j];
142     output += hexaDeciNum[j];
143   }
144 }
145 output="AT+MSGHEX="+output;
146 Serial.print(output);
147 node_L5.println(output);
148 Serial.println("");
149 }
150 //--------------------------------------------------------------------
151
152
153 //----------------------Funcion obtenerBat-----------------------------
154 //--------------------------------------------------------------------
155 void obtenerBat(){
156   inputString="";
157   Serial.println("\nTx:AT+VDD");
158   node_L5.println("AT+VDD");
159   delay(2000);
160   while(node_L5.available()){
161     char inChar = (char)node_L5.read();
162     inputString += inChar;
```

```
163    }
164    Serial.println(inputString);
165    int pos=inputString.indexOf("VDD");
166    if (pos>=0) {
167     Serial.print("se ha detectado la palabra VDD en la posicion ");
168     Serial.println(pos);
169     nuevoV=inputString.substring(pos+4, pos+9);
170     Serial.println(nuevoV);
171      }
172 }
173 //---------------------------------------------------------------------
```

Código 1: Código de Arduino completo

ANEXO 2

N muestra	Vm	Varduino	Porcentaje
1	5.01	5.00	0.200%
2	4.94	4.98	0.810%
3	4.82	4.91	1.867%
4	4.70	4.85	3.191%
5	4.63	4.81	3.888%
6	4.51	4.78	5.987%
7	4.40	4.71	7.045%
8	4.31	4.61	6.961%
9	4.23	4.52	6.856%
10	4.18	4.45	6.459%
11	4.08	4.35	6.618%
12	4.03	4.30	6.700%
13	3.97	4.24	6.801%
14	3.92	4.21	7.398%
15	3.88	4.20	8.247%
16	3.81	4.18	9.711%
17	3.78	4.13	9.259%
18	3.73	4.12	10.456%
19	3.70	4.11	11.081%
20	3.68	4.04	9.783%
21	3.63	4.00	10.193%
22	3.61	3.98	10.249%
23	3.60	3.90	8.333%
24	3.55	3.81	7.324%
25	3.50	3.73	6.571%
26	3.45	3.68	6.667%
27	3.38	3.60	6.509%
28	3.30	3.51	6.364%
29	3.28	3.48	6.098%
30	3.20	3.40	6.250%
31	3.14	3.35	6.688%
32	3.12	3.31	6.090%
33	3.08	3.28	6.494%
34	3.02	3.20	5.960%
35	2.95	3.11	5.424%
36	2.90	3.04	4.828%
37	2.82	2.94	4.255%

Error porcentual medio
6.584%

Figura 1: Cálculo de errores en excel