Universidad Nacional de Loja

Facultad de la Energía, las Industrias y los Recursos

Naturales no Renovables

Maestría en Telecomunicaciones

“Diseño e implementación de un sistema para mantenimiento predictivo que permita detectar anomalías en un motor eléctrico mediante TinyML y LoRaWAN.”

Trabajo de Titulación previo, a la obtención del título de Magister en Telecomunicaciones.

AUTOR:

Ing. Juan Carlos Zaruma Villamarín

DIRECTOR:

Ing. Ronald Raúl Criollo Bonilla, Msig.

Loja – Ecuador

2024

Certificación

Loja, 26 de junio del 2024

Ing. Ronald Raúl Criollo Bonilla, Msig.
DIRECTOR DEL TRABAJO DE TITULACIÓN

CERTIFICO:

Que he revisado y orientado todo proceso de la elaboración del Trabajo de Titulación denominado: **Diseño e implementación de un sistema para mantenimiento predictivo que permita detectar anomalías en un motor eléctrico mediante TinyML y LoRaWAN,** previo a la obtención del título **de Magíster en Telecomunicaciones,** de la autoría del estudiante **Juan Carlos Zaruma Villamarín**, con **cédula de identidad N° 1104724263 ,** una vez que el trabajo cumple con todos los requisitos exigidos por la Universidad Nacional de Loja para el efecto, autorizo la presentación para la respectiva sustentación y defensa.

Firmado electrónicamente por:
RONALD RAUL CRIOLLO BONILLA

Ing. Ronald Raúl Criollo Bonilla, Msig.
DIRECTOR DEL TRABAJO DE TITULACIÓN

Autoría

Yo, Juan Carlos Zaruma Villamarín, declaro ser autor del presente Trabajo de Titulación y eximo expresamente a la Universidad Nacional de Loja y a sus representantes jurídicos de posibles reclamos y acciones legales, por el contenido del mismo. Adicionalmente acepto y autorizo a la Universidad Nacional de Loja la publicación de mi Trabajo de Titulación en el Repositorio Digital Institucional – Biblioteca Virtual.

Firma: 

Cédula de Identidad: 1104724263

Fecha: 26/06/2024

Correo electrónico: juan.zaruma@unl.edu.ec

Teléfono: 0939800718

Carta de autorización por parte del autor, para consulta, reproducción parcial o total y/o publicación electrónica de texto completo, del Trabajo de Titulación.

Yo, **Juan Carlos Zaruma Villamarín**, declaro ser autor del Trabajo de Titulación denominado: **Diseño e implementación de un sistema para mantenimiento predictivo que permita detectar anomalías en un motor eléctrico mediante TinyML y LoRaWAN**, como requisito para optar el título de **Magíster Telecomunicaciones**, autorizo al sistema Bibliotecario de la Universidad Nacional de Loja para que, con fines académicos, muestre la producción intelectual de la Universidad, a través de la visibilidad de su contenido de la siguiente manera en el Repositorio Institucional.

Los usuarios pueden consultar el contenido de este trabajo en el Repositorio Institucional, en las redes de información del país y del exterior con las cuales tenga convenio la Universidad.

La Universidad Nacional de Loja, no se responsabiliza por el plagio o copia del Trabajo de Titulación que realice un tercero.

Para constancia de esta autorización suscribo, en la ciudad de Loja, a los veintiséis días del mes de junio de dos mil veinticuatro.

Firma: 

Cédula de identidad: 1104724263
Dirección: Ciudadela Zamora
Correo Electrónico: juan.zaruma@unl.edu.ec
Teléfono: 0939800718

DATOS COMPLEMENTARIOS:
DIRECTOR DEL TRABAJO DE TITULACIÓN: Ing. Ronald Raúl Criollo Bonilla, Msig.

Dedicatoria

Dedico este trabajo de titulación a mi familia, amigos y compañeros que, de alguna manera, colaboraron para que esta meta profesional se hiciera realidad. En especial, dedico este logro a mis sobrinos, quienes son el mejor regalo que me han dado mis hermanos y el motor que me impulsa a asumir nuevos retos y a buscar ser una mejor persona y profesional cada día.

Juan Carlos Zaruma Villamarín

Agradecimiento

Agradezco a todas las personas que contribuyeron para que este objetivo personal se hiciera realidad. A mi familia y amigos, por su apoyo incondicional y sus palabras de aliento, que me motivaron a dar lo mejor de mí y a aprovechar al máximo esta nueva etapa de estudio. Extiendo mi gratitud a mis compañeros de maestría y a mis docentes de las distintas asignaturas del programa, por compartir sus conocimientos y experiencias, que fueron esenciales para mi desarrollo profesional.

A mi tutor el Ing. Ronald Raúl Criollo Bonilla, Msig., por sus consejos, su paciencia y su motivación constante, sin los cuales este trabajo no habría sido posible.

Agradezco infinitamente a todos.

Juan Carlos Zaruma Villamarín

Índice de Contenidos

Índice de Tablas:

Índice de Figuras:

Índice de Anexos:

1. Título

“Diseño e implementación de un sistema para mantenimiento predictivo que permita detectar anomalías en un motor eléctrico mediante TinyML y LoRaWAN.”

2. Resumen

El presente proyecto ofrece una propuesta de un sistema integral que detecta en tiempo real anomalías que se podrían presentar en el funcionamiento normal de un motor eléctrico. El prototipo utiliza como elemento principal un Arduino Nano 33 BLE Sense, cuyo software tiene incorporado un modelo de entrenamiento generado con TinyML en Edge Impulse que genera eventos de acuerdo a los diferentes estados que puede presentar el motor tales como: apagado, encendido y anomalía en caso de funcionamiento anormal. El prototipo cuenta además con un módulo Wio E5, que se conecta al Arduino y por medio de una red LoRaWAN envía el dato del estado del motor al Gateway SenseCAP M2 y este último lo transmite al servidor The Things Network.

Como parte complementaria del proyecto, se ha desarrollado una interfaz utilizando el stack MING. La primera parte está compuesta de un flujograma en NodeRED, en el cual, se implementó un cliente MQTT que se conecta al servidor de The Things Network. Cada vez que se recibe un mensaje, la información contenida es decodificada y organizada, para luego ser enviada y almacenada en InfluxDB. Con la información almacenada en la base de datos y con el objetivo de monitorear los datos y facilitar al operador la evaluación periódica del estado del motor y la detección de anomalías, se ha implementado un dashboard en Grafana que incluye paneles que permiten llevar un registro de datos, visualizar posibles eventos ocurridos y generar alarmas.

Este sistema unificado ofrece una solución completa para el mantenimiento predictivo de motores eléctricos, desde la recolección inicial de datos hasta la presentación visual y la toma de decisiones basada en la identificación de anomalías.

***Palabras claves**: Machine Learning, Arduino, Monitoreo Remoto, LoRaWAN, Node-RED, The Things Network.*

Abstract

The objective of this research project is to propose an integrated system that detects anomalies in real time during the normal operation of an electric motor. This prototype uses an Arduino Nano 33 BLE Sense as its main component with software incorporating a training model generated using TinyML on Edge Impulse. This model generates events corresponding to various motor states, such as off, on, and anomaly in case of abnormal operation. The prototype also includes a Wio E5 module, which connects to the Arduino and sends the motor's state data to the SenseCAP M2 Gateway via a LoRaWAN network. The gateway then transmits the data to The Things Network server.

As a supplementary part of the project, an interface has been developed using the MING stack. The first part consists of a flowchart in Node-RED, in which an MQTT client was implemented to connect to The Things Network server. Each time a message is received, the information is decoded, organized, and then sent to and stored in InfluxDB. With the stored data, and aiming to facilitate monitoring and periodic evaluation of the motor's status and anomaly detection, a dashboard has been implemented in Grafana. This dashboard includes panels for data tracking, visualization of possible events, and alarm generation.

This unified system offers a comprehensive solution for predictive maintenance of electric motors, from initial data collection to visual presentation and decision-making based on anomaly detection.

Keywords: *Machine Learning, Arduino, Remote Monitoring, LoRaWAN, Node-RED, The Things Network.*

3. Introducción

La información generada por dispositivos IoT puede ser empleada para entrenar algoritmos de Machine Learning, contribuyendo así a mejorar la capacidad de estos dispositivos para procesar y analizar datos en tiempo real, lo que resulta en una reducción significativa de la latencia. Paralelamente, los algoritmos TinyML buscan llevar la potencia del Machine Learning a dispositivos extremadamente pequeños, caracterizados por limitaciones significativas en capacidad de procesamiento, memoria y recursos energéticos, comunes en el entorno del internet de las cosas. Estos algoritmos están diseñados de manera óptima para ejecutar tareas complejas, como el reconocimiento de imágenes y audio, en entornos con recursos muy restringidos. La implementación exitosa de TinyML requiere conocimientos especializados en optimización de hardware y software, así como en ciencia de datos e Inteligencia Artificial.

La integración de TinyML en IoT representa la convergencia de tres tecnologías prometedoras: el Internet de las Cosas (IoT), las capacidades de aprendizaje automático y la miniaturización de dispositivos. Este enfoque permite la ejecución de tareas complejas con un consumo mínimo de energía. Estas herramientas innovadoras están habilitando a los investigadores para abordar una amplia variedad de problemas en diversos sectores. Este proyecto se centra en las aplicaciones de TinyML en la industria 4.0, específicamente en el ámbito del mantenimiento predictivo a través de un enfoque analítico. Utiliza datos históricos y en tiempo real para identificar anomalías en las cuales una máquina, en este caso un motor, puede presentar problemas permitiendo intervenciones anticipadas. En la práctica, esto implica que los sistemas embebidos de equipos industriales, capacitados por algoritmos de Machine Learning, podrían analizar datos en tiempo real y alertar sobre la necesidad de reparaciones preventivas, incluso en diferentes épocas del año o ubicaciones.

Objetivos

Objetivo general

Diseñar e implementar un sistema de mantenimiento predictivo para motores eléctricos con TinyML y comunicación LoRaWAN que integre MING para visualizar y almacenar datos.

Objetivos específicos

- Diseñar la arquitectura integral del sistema de mantenimiento predictivo definiendo la interconexión de TinyML, LoRaWAN y Stack MING.

- Generar un dataset de pruebas para evidenciar posibles anomalías y validar estado normal del motor.

- Desarrollar e implementar un modelo de TinyML en dispositivos embebidos para realizar análisis en tiempo real de los datos adquiridos.

- Configurar la red LoRaWAN para la transmisión eficiente de datos desde el Arduino Nano 33 BLE Sense hasta una estación base central.

- Configurar los cuadros de mando del Stack MING para visualizar y almacenar datos adquiridos empleando MQTT, InfluxDB, NodeRED y Grafana.

- Integrar todas las partes del sistema y realizar pruebas exhaustivas para garantizar su funcionamiento coherente.

4. Marco teórico

El presente proyecto de investigación se centra en el desarrollo de un sistema innovador de mantenimiento predictivo para motores eléctricos, empleando tecnologías avanzadas como TinyML y LoRaWAN. Este sistema no solo detecta anomalías en tiempo real mediante un modelo TinyML, sino que también incorpora la eficiente visualización y almacenamiento de datos a través de la plataforma MING (MQTT + InfluxDB + NodeRED + Grafana).

4.1.Introducción al uso de microcontroladores y TinyML

TensorFlow Lite es una extensión de TensorFlow diseñada para dispositivos móviles y embebidos con recursos limitados, como microcontroladores. Esta versión permite convertir modelos de TensorFlow a un tamaño adecuado para trabajar en el campo de TinyML (Machine Learning para dispositivos pequeños). TinyML implica la incorporación de Inteligencia Artificial en microcontroladores, adaptando los procesos para funcionar con menor capacidad de procesamiento, espacio reducido y un consumo de energía más bajo. En general, se busca obtener modelos que ocupen alrededor de 30 KB y que consuman energía en el orden de los miliwatios. Se pueden utilizar en aplicaciones como la detección y alerta temprana de fallos, detección de objetos, automatización de procesos y seguridad («¿Qué es TinyML?», 2022).

En resumen, TinyML ofrece varias ventajas significativas, entre las que se incluyen:

- El consumo de energía es bajo ya que no es necesario enviar la información, pues se ejecuta localmente en dispositivos pequeños.
- Trabajar con baja o nula cobertura de a internet, al procesar datos localmente en lugar de enviarlos a la nube. TinyML contribuye a mejorar la privacidad y seguridad de la información, ya que los datos sensibles no abandonan el dispositivo.
- Baja latencia ya que todo se ejecuta de forma local.
- Respuestas ante eventos nuevos que no hayan sido programados previamente.
- Es altamente escalable y adaptable a una variedad de aplicaciones, desde dispositivos portátiles hasta sistemas integrados.

4.1.1. Tarjetas electrónicas para Machine Learning

TensorFlow Lite es una herramienta diseñada para dispositivos móviles y embebidos, como los microcontroladores. Esta versión permite convertir proyectos de TensorFlow a un tamaño adecuado para trabajar con TinyML. TensorFlow Lite trabaja en microcontroladores de 32 bits. Los modelos compatibles son los que tienen la arquitectura de la serie ARM Cortex-M y también el ESP32 («TinyML», s. f.). A continuación, una lista de modelos compatibles con enlaces a sus páginas oficiales:

- Arduino (https://store-usa.arduino.cc/products/arduino-nano-33-ble-sense)
- SparkFun Edge (https://www.sparkfun.com/products/15170)
- Kit STM32F746(https://www.st.com/en/evaluation-tools/32f746gdiscovery.html)
- Kit TF Adafruit para microcontroladores (https://www.adafruit.com/product/4317)
- ESP32-DevKitC (https://www.espressif.com/en/products/devkits/esp32-devkitc)
- Wio: ATSAMD51(https://www.seeedstudio.com/Wio-Terminal-p-4509.html)
- Espressif ESP32-S3-EYE (https://www.adafruit.com/product/5955)
- Adafruit EdgeBadge (https://www.adafruit.com/product/4400)
- Sony Spresense (https://developer.sony.com/spresense)

4.1.2. Arduino Nano 33 BLE Sense

El Arduino Nano 33 BLE Sense es una placa de desarrollo de microcontroladores, diseñada por Arduino, que es pequeña y compacta, ideal para proyectos de IoT y wearables. El Arduino Nano 33 BLE Sense combina conectividad Bluetooth de baja energía (BLE) con una variedad de sensores integrados, lo que significa que tiene potencial para algunas aplicaciones de TinyML (*Arduino Tiny Machine Learning Kit*, s. f.). En la Figura 1, se puede apreciar el Arduino Tiny Machine Learning Kit, que utiliza el Arduino Nano 33 BLE Sense como principal componente y entre las principales características que presenta tenemos:

- Basado en el microcontrolador nRF52840 de ARM Cortex-M4F con Bluetooth 5.0 BLE y capacidades de bajo consumo.

- Incluye una variedad de sensores incorporados: sensor inercial (acelerómetro/giroscopio), sensor de micrófono, sensor de luz ambiental, sensor de proximidad, sensor de presión y temperatura, sensor de humedad y temperatura.
- Tiene 6 ejes de detección de movimiento (acelerómetro/giroscopio).
- Comunicación inalámbrica vía Bluetooth 5.0 BLE, antena PCB integrada.
- Entradas y salidas: 14 pines digitales (6 pueden ser usados como salidas PWM), 6 entradas analógicas, un LED RGB, 6 botones de capacitación.

Figura 1.
Arduino Tiny Machine Learning Kit

Fuente: Arduino Tiny Machine Learning Kit, s. f

El Arduino Nano 33 BLE Sense es perfecto para proyectos de IoT, captura de datos ambientales y desarrollo de aplicaciones interactivas, aprovechando sus capacidades integradas de sensores y conectividad Bluetooth de baja energía.

4.1.3. TensorFlow Lite y Edge Impulse Studio

Edge Impulse es una plataforma para desarrollar algoritmos de aprendizaje máquina enfocados a implementarse en sistemas embebidos como microcontroladores o computadoras con recursos reducidos cuenta con una interfaz muy amigable para el usuario y bastante intuitiva.

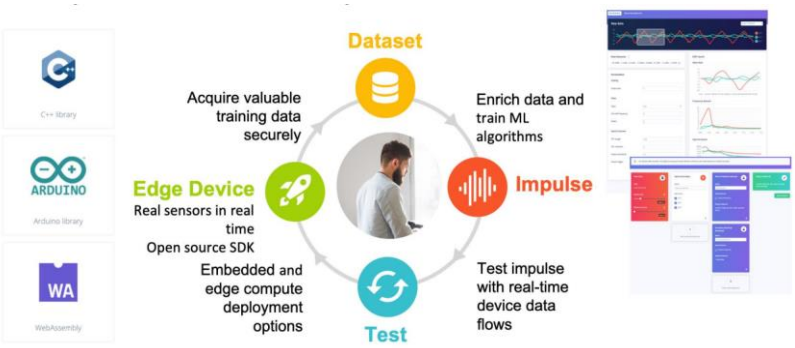
Edge Impulse es una plataforma en constante crecimiento y desarrollo cuenta con el respaldo de grandes fabricantes de semiconductores como, por ejemplo: ARM, ST Electronics, Microchip, Nordic Semiconductor y Arduino. Destaca especialmente la colaboración con TinyML y Hackster.io, ya que gracias a ellos se pueden encontrar algoritmos de aprendizaje automático bien documentados de numerosos proyectos desarrollados (*About Edge Impulse*, s. f.).

Algunas aplicaciones típicas de sistemas TinyML con aprendizaje automático utilizando Edge Impulse se encuentran en los sectores industrial, logístico y sanitario. Entre ellas se incluyen el mantenimiento predictivo, la supervisión y seguimiento de activos, y la detección de humanos y animales (Rodriguez, 2022).

En la Figura 2, podemos apreciar como Edge Impulse es especialmente adecuado para proyectos en los que se busque una solución rápida y fácil para incorporar capacidades de aprendizaje automático en dispositivos de borde y sistemas embebidos. Entre las características que ofrece tenemos una interfaz intuitiva, cuenta con bibliotecas y modelos preentrenados. El diseño para generar el dataset está preparado para funcionar con una amplia variedad de sensores y es compatible con diversas plataformas de hardware populares, como Arduino, Raspberry Pi y otras placas de desarrollo. Además, facilita el despliegue de modelos entrenados en dispositivos de borde, asegurando una ejecución eficiente incluso en entornos con recursos limitados.

Figura 2.

Edge Impulse para desarrolladores.



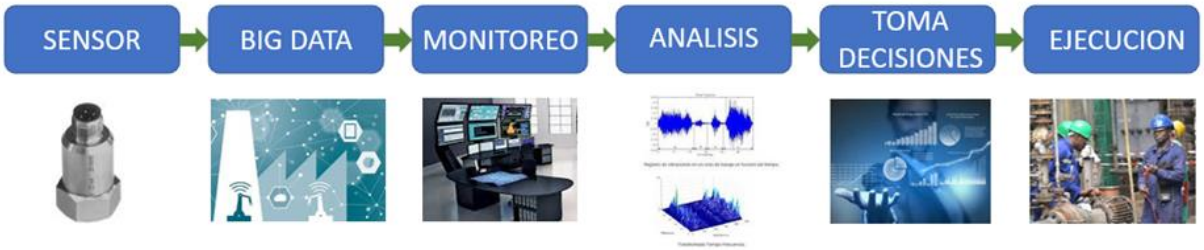
Fuente: Imagen tomada de https://cms.tinyml.org/wp-content/uploads/industry-news/tinyML_Talks-_Peter_Ing_210924.pdf

4.1.4. Industria 4.0: Mantenimiento preventivo

El mantenimiento en la industria 4.0 busca transformar los activos en equipos más inteligentes mediante el uso de sensores, big data, IoT y computación en la nube. En la Figura 3 podemos ver un ejemplo del modelo de mantenimiento y los pasos a seguir para llevarlo a cabo.

Figura 3.

Modelo de mantenimiento predictivo en la Industria 4.0.



Fuente: (¿En qué consiste el mantenimiento 4.0 y cuáles son sus beneficios?, s. f.)

El mantenimiento enfocado en la industria 4.0 busca implementar un servicio inteligente e innovador con beneficios (*¿En qué consiste el mantenimiento 4.0 y cuáles son sus beneficios?*, s. f.) tales como:

- Minimizar la interrupción de la producción por averías, ya que es posible identificar los elementos de la máquina que pueden dañarse antes de que ocurra un fallo catastrófico.
- Aumentar la satisfacción del cliente al reducir los problemas de calidad del producto debido a averías y evitar retrasos en la entrega de los productos.
- Optimizar el departamento de mantenimiento mediante la automatización de procesos para detectar anomalías a través de un sistema de monitoreo, permitiendo que el personal se enfoque en tareas estratégicas en lugar de desgastarse con múltiples dificultades operativas.
- Incrementa el índice de seguridad y reduce la incidencia de accidentes.

4.2.TinyML: principios de funcionamiento

El Machine Learning constituye una herramienta fundamental en el ámbito de la Inteligencia Artificial (IA) que permite a las máquinas aprender a partir de datos. Este aprendizaje

puede ser utilizado para predecir eventos, clasificar objetos y otros usos. En contraste con los programas convencionales, donde el programador conoce las variables que determinan la respuesta y limita las posibles respuestas, el Machine Learning permite a una máquina o computadora generar soluciones autónomamente basadas en experiencias de aprendizaje previas. («¿Qué es TinyML?», 2022)

TinyML es la abreviatura de Tiny Machine Learning y se refiere a la implementación del aprendizaje automático en componentes y dispositivos electrónicos de tamaño reducido, como microcontroladores, dispositivos IoT y sistemas integrados. El objetivo principal de TinyML es capacitar estos dispositivos para aprovechar y aplicar funcionalidades de aprendizaje automático de manera eficiente y efectiva en entornos donde los recursos computacionales son limitados. («¿Qué es TinyML?», 2022).

Debido a las limitadas capacidades de hardware de estos dispositivos embebidos, el aprendizaje automático en TinyML conlleva requisitos únicos. Dado que operan con estrictas restricciones de energía, cualquier capacidad de cómputo para tareas de aprendizaje automático debe optimizarse para minimizar el consumo. La principal característica de este enfoque radica en los recursos extremadamente limitados de estos microcontroladores, tanto en términos de memoria de trabajo como de potencia de procesamiento. En comparación con la mayoría de las aplicaciones tradicionales de aprendizaje automático, los dispositivos TinyML tienen una fracción de la memoria RAM disponible y una habilidad de cómputo menor. Estas limitaciones hardware imponen restricciones fundamentales en los modelos de aprendizaje automático que pueden ser desplegados eficientemente.

Las fortalezas de TinyML se centran en su capacidad para lograr la autonomía del sistema. Al procesar los datos de forma local, se elimina la dependencia de una conexión a Internet de alta velocidad. Al mismo tiempo, esto también reduce los tiempos de latencia ya que no resulta necesaria la transferencia de datos entre distintos dispositivos o sistemas. TinyML también tiene

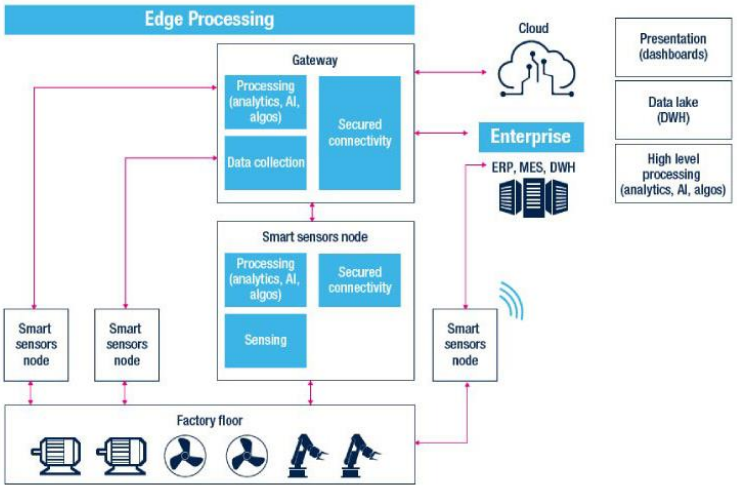
grandes exigencias en el ámbito de la privacidad y protección de datos. Debido a su eficiencia energética, TinyML puede operar en dispositivos alimentados por baterías («TinyML», s. f.).

4.2.1. Generación de dataset para el modelo de entrenamiento

El mantenimiento predictivo utiliza herramientas como el análisis estadístico y Machine Learning para predecir el estado del equipo en función de lo siguiente: detección de anomalías, algoritmos de clasificación y modelos predictivos. Por ejemplo, si una planta industrial puede tener varios motores, ventiladores y brazos robóticos utilizados en la fabricación de un producto. La empresa buscará minimizar el tiempo de inactividad de la maquinaria para maximizar la producción. Si estos equipos cuentan con sensores que pueden ser interpretados mediante aprendizaje automático y otras técnicas de análisis predictivo, es posible detectar cuándo una pieza de equipo se acerca a fallar. Un posible escenario se muestra en la Figura 4, donde los sensores instalados en los distintos componentes envían datos que son analizados por modelos de aprendizaje automático para identificar patrones de fallos. Esta información permite a los operadores realizar mantenimiento predictivo y reparar o reemplazar componentes antes de que ocurra una falla no planeada, minimizando así el tiempo de inactividad (*3 usos de tinyML en el borde*, s. f.).

Figura 4.

Sensores inteligentes que se utilizan para el mantenimiento predictivo.



Fuente: Imagen tomada de https://www.digikey.com/es/blog/3-uses-for-tinyml-at-the-edge

Integrar sensores inteligentes con microcontroladores de baja potencia que implementan TinyML permite habilitar una diversidad de aplicaciones prácticas. El mantenimiento predictivo posibilitado por TinyML puede volverse más proactivo y eficiente, potencialmente evitándole a una empresa costosas reparaciones de emergencia y permitiendo optimizar los planes de servicio de los equipos. Al detectar problemas mediante el análisis en tiempo real de datos de sensores, hechos que de otro modo podrían haber escalado hasta convertirse en fallas graves pueden ser abordados tempranamente, reduciendo así los tiempos de inactividad operativa. TinyML facilita la integración de inteligencia en el mantenimiento industrial a través del monitoreo remoto y el análisis en el borde, representando un cambio de un paradigma reactivo a uno proactivo.

4.2.2. Modelo para la detección de anomalías con TinyML

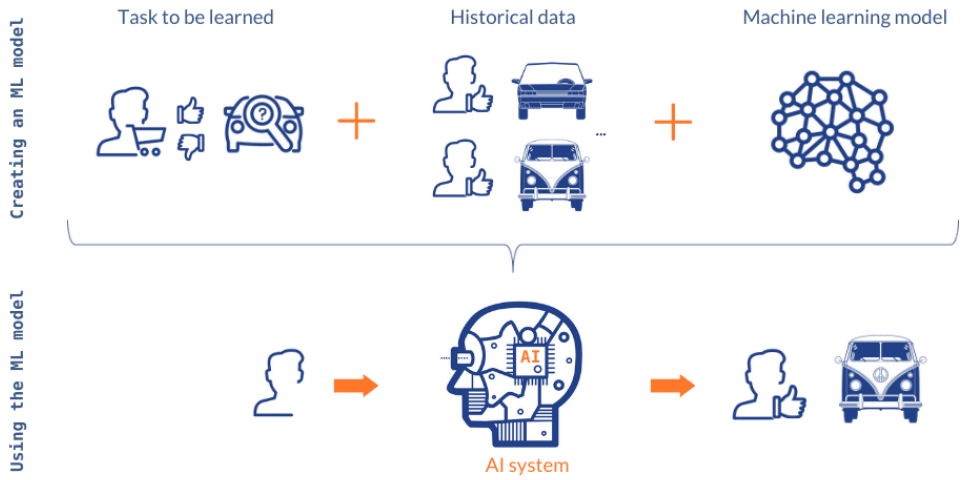
Un modelo de aprendizaje automático consiste en funciones matemáticas (es decir, un conjunto de reglas) que se supone que reflejan la conexión subyacente entre la variable objetivo y la variable de entrada. Los modelos de aprendizaje automático se desarrollan mediante el denominado entrenamiento de modelos. Durante el entrenamiento del modelo, proporcionamos a un modelo de aprendizaje automático datos existentes a partir de los cuales el modelo puede identificar y mostrar las relaciones relevantes entre las variables independientes y de salida.

Al utilizar Machine Learning, una máquina o computadora tiene la capacidad de responder a eventos generando soluciones de manera autónoma basadas en experiencias de aprendizaje previas. Por ejemplo, podemos identificar qué condiciones indican fallas en una máquina industrial y usar un sistema embebido que lea y compare esos datos con las condiciones establecidas para alertar sobre posibles fallas. Sin embargo, no todos los eventos son iguales, y siempre existe la posibilidad de que los factores varíen respecto a incidentes anteriores. («TinyML», s. f.).

Por otra parte, si dicho sistema embebido tiene la capacidad de aprender de cada fallo que se produce, estará preparado para enfrentar nuevas situaciones y, al mismo tiempo, dispondrá de más opciones para hacer comparaciones. No es imperativo exponerse a diversas fallas para entrenar el sistema, ya que esto podría resultar contraproducente. En la figura 5, podemos ver un

ejemplo de cómo se puede crear un modelo de Machine Learning, partiendo de su entrenamiento, datos históricos aplicación del modelo y finalmente como se usa el modelo.

Figura 5.
Modelo de aprendizaje automático.

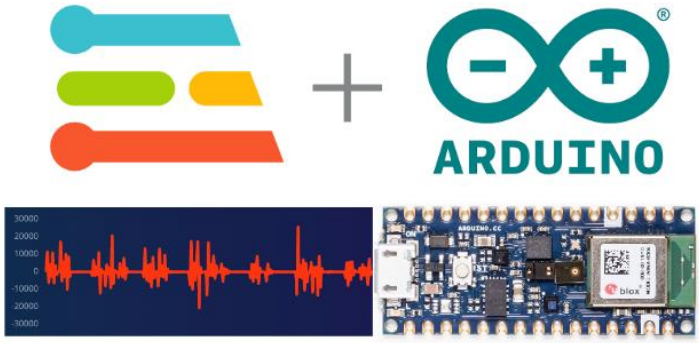


Fuente: Imagen tomada de https://www.alexanderthamm.com/es/data-science-glossary/modelo-de-aprendizaje-automatico-2/

4.2.3. Integración de TinyML y Arduino

La tendencia a ejecutar Machine Learning en microcontroladores, Figura 6, a veces se denomina Embedded ML o Tiny ML. TinyML tiene el potencial de crear pequeños dispositivos que pueden tomar decisiones inteligentes sin necesidad de enviar datos a la nube, lo cual es excelente desde la perspectiva de la eficiencia y la privacidad. Incluso potentes modelos de aprendizaje profundo (basados en redes neuronales artificiales) están llegando ahora a los microcontroladores. Con Edge Impulse ahora se puede recopilar rápidamente datos de sensores del mundo real, entrenar modelos de aprendizaje automático con estos datos en la nube y luego implementar el modelo en el Arduino (*Edge Impulse Brings TinyML to Millions of Arduino Developers*, 2020).

Figura 6.
Integración Edge Impulse con Arduino.



Fuente: (Edge Impulse Brings TinyML to Millions of Arduino Developers, 2020)

4.3.Introducción a LoRaWAN

LoRa (abreviatura de largo alcance - Long Range) es la tecnología con rápido crecimiento que atrae el interés de los investigadores porque permite la mejora de la robustez del sistema, una mayor tolerancia a interferencias y un menor consumo de recursos. LoRa es una tecnología que opera en un espectro de radiofrecuencia sin licencia. Es un protocolo de capa física que utiliza la modulación de espectro ensanchado, permitiendo la comunicación a larga distancia a cambio de un ancho de banda reducido. Además, emplea una forma de onda de banda estrecha con una frecuencia central para transmitir datos, lo que la hace resistente a las interferencias.

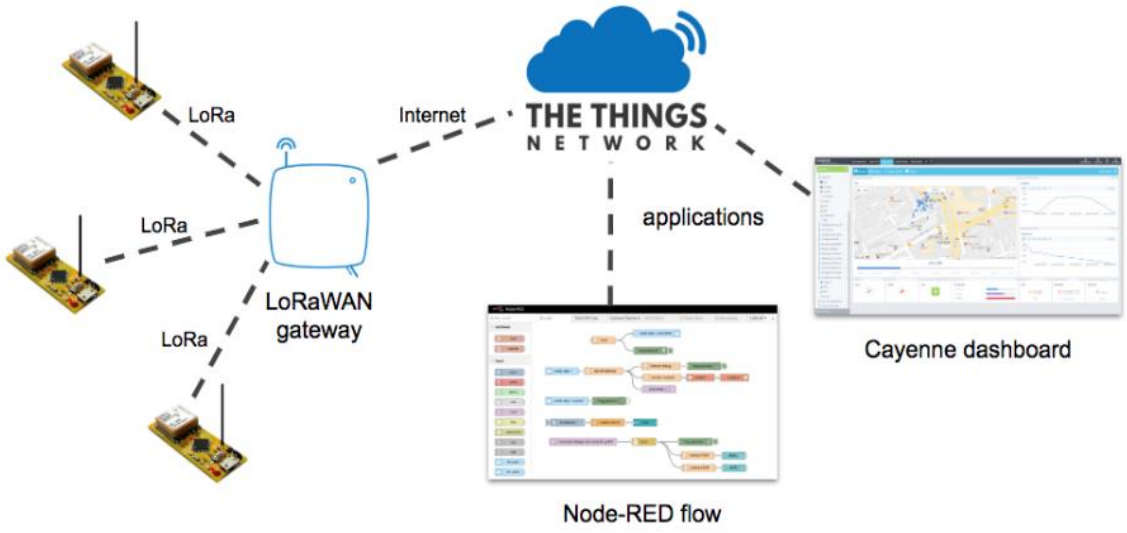
LPWAN (Low Power Wide Area Network) es un tipo de red de comunicación inalámbrica utilizada en el ámbito IoT. Esta tecnología está diseñada para dispositivos que requieren enviar pequeñas cantidades de datos a largas distancias. Aunque ofrece una velocidad de transmisión baja, compensa esta limitación al reducir el uso de recursos y optimizar el consumo de las baterías de los dispositivos.

LoRaWAN (Low Range Wide Area Network) es un protocolo abierto e inalámbrico de comunicaciones a nivel de red, diseñado para la transmisión de información entre dispositivos IoT.

El protocolo LoRaWAN es el resultado de la aplicación de la tecnología LoRa sobre las redes LPWAN. Se encuentra implementado en las capas de red y de enlace de datos, por encima de la capa física sobre la que funciona LoRa, y está orientado a redes de dispositivos IoT en los que se necesitan transmitir pequeños paquetes de información a larga distancia (*LoRaWAN y su aportación a las tecnologías IIoT | INCIBE-CERT | INCIBE*, s. f.).

LoRa/LoRaWAN ofrece funciones que soportan comunicaciones bidireccionales, económicas, móviles y seguras para aplicaciones de IoT, machine-to-machine (M2M), Smart Building, Smart City e industriales. Está optimizado para un consumo de energía reducido y diseñado para escalar desde una única pasarela hasta grandes redes globales con miles de millones de dispositivos. En la Figura 7, podemos observar un ejemplo sencillo de una red LoRaWAN implementada para enviar información a una interfaz: tres nodos envían datos a una pasarela (Gateway), la cual a su vez transmite la información a Internet, permitiendo su monitoreo desde cualquier lugar.

Figura 7.
Ejemplo de aplicación con LoRaWAN y red desplegada.



Fuente: (Ejemplo de aplicación con LoRaWAN y red desplegada)

El estándar de red LoRaWAN apunta a requerimientos característicos de IoT como son:

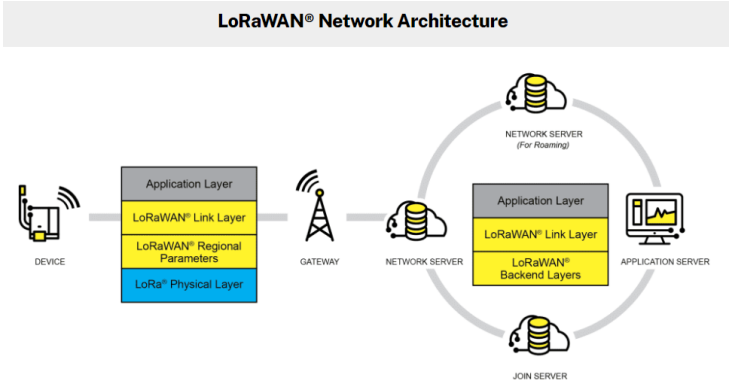
- Conexiones bidireccionales seguras mediante encriptación de extremo a extremo.
- Bajo consumo de energía, velocidades de datos reducidas, baja frecuencia de transmisión, movilidad y servicios de localización.
- Largo alcance de comunicación (10 - 20 km).
- Conexión de infinidad de sensores y equipos a redes públicas o privadas.
- Interoperabilidad de las diversas redes LoRaWAN en todo el mundo,

El estándar LoRaWAN facilita la interconexión de objetos inteligentes sin requerir instalaciones locales complejas, brindando gran libertad de uso tanto a los usuarios finales como a los desarrolladores y a las empresas interesadas en implementar su propia red para Internet de las Cosas (*Tecnología LoRA y LoRAWAN - Catsensors*, s. f.).

4.3.1. Arquitectura y protocolos de comunicación LoRaWAN

En la Figura 8 se puede observar un ejemplo de la arquitectura de red LoRaWAN, implementada en una topología de estrella. En esta configuración, las puertas de enlace transmiten mensajes entre los dispositivos finales y un servidor de red central. Las puertas de enlace están conectadas al servidor de red mediante conexiones IP estándar y actúan como un puente transparente, convirtiendo paquetes RF en paquetes IP y viceversa.

Figura 8.
Arquitectura de red LoRaWAN.



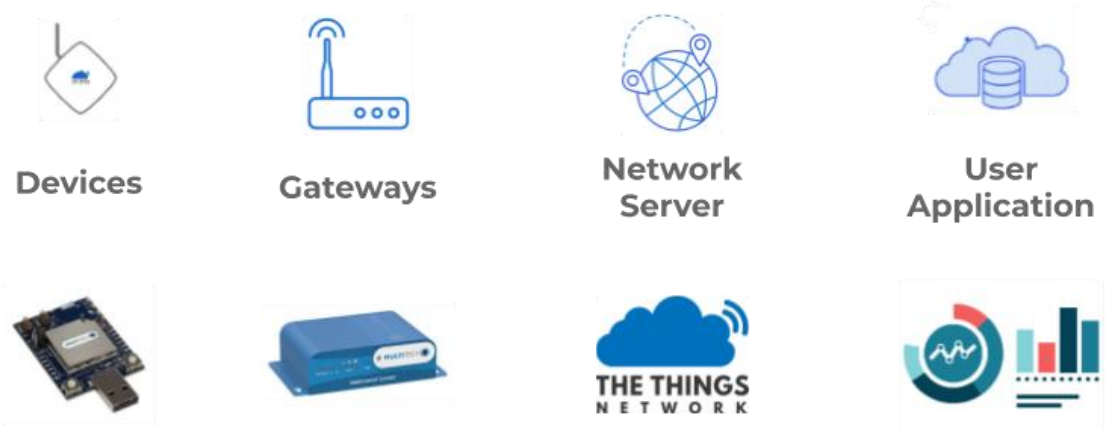
*Fuente: (*Qué es LoRa, cómo funciona y características principales - Venco Electrónica, s. f.*)*

La comunicación inalámbrica se beneficia de las características de largo alcance de la capa física LoRa, permitiendo un enlace de un solo salto entre el dispositivo final y una o varias puertas de enlace. Todos los modos admiten comunicación bidireccional y hay soporte para grupos de direccionamiento de multidifusión, lo que optimiza el uso del espectro durante tareas como actualizaciones de firmware por aire y otros mensajes de distribución masiva (*Qué es LoRa, cómo funciona y características principales - Venco Electrónica*, s. f.).

4.3.2. Configuración de un nodo en la red LoRaWAN

Los nodos transmiten mensajes a través del protocolo de radio LoRa, los cuales son recibidos por varios Gateway que reenvían las transmisiones. Cada Gateway está conectado a un router y cada router está conectado a uno o más brokers. Los brokers son la parte central de TTN, responsables de asignar dispositivos a aplicaciones, reenviar mensajes a la aplicación correcta y dirigir mensajes al router adecuado. El servidor de red se encarga de la funcionalidad específica de LoRaWAN, mientras que los usuarios gestionan los datos de las aplicaciones, los dispositivos y el cifrado (*Exploración a ecosistema LoRaWAN*, s. f.). En la Figura 9 se presenta un ejemplo de una red LoRaWAN con los principales elementos de manera general partiendo de un nodo, el Gateway, el servidor y las aplicaciones.

Figura 9.
Elementos de una red LoRaWAN.



Fuente: Imagen tomada de https://www.gotoiot.com/pages/articles/lorawan_exploration/content.html

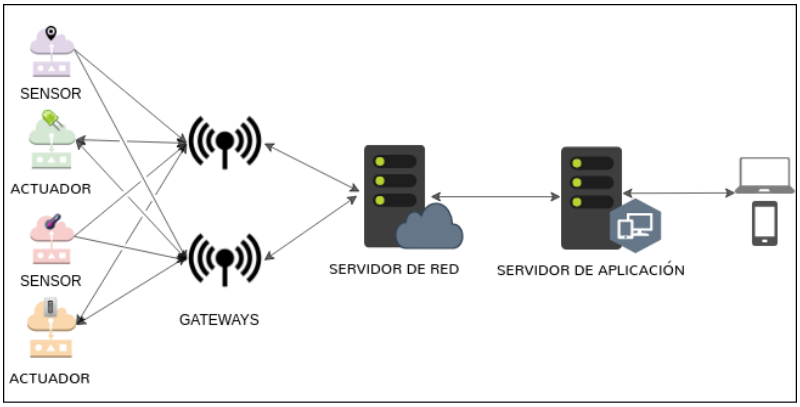
4.3.3. Establecimiento de la red con Gateway LoRaWAN

Una red LoRaWAN, como se muestra en la Figura 10, consta de cuatro componentes principales:

- ***Nodos finales:*** todos los dispositivos individuales que envían o reciben información utilizando la red LoRaWAN, como sensores, actuadores y geolocalizadores, se denominan nodos. Estos nodos se comunican con los Gateways mediante la modulación LoRa RF, permitiendo la comunicación a larga distancia.
- ***Pasarelas (Gateways):*** actúan como puente entre los nodos finales y el resto de la red. Recogen los datos de los nodos finales y los transmiten al servidor de red. Mientras que la comunicación entre los nodos finales y las pasarelas se realiza a través de LoRaWAN, las pasarelas se comunican con el servidor de red mediante protocolos de mayor ancho de banda, como WiFi, Ethernet o celular. Son los elementos más importantes de la red ya que sin ellos la comunicación no sería posible.
- ***Servidor de red:*** consolidan los datos de las pasarelas antes de cargarlos en el servidor de aplicaciones. Estos dispositivos se encargan de procesar y traducir la información recibida de los nodos.
- ***Servidor de aplicaciones:*** son los servidores encargados del tratamiento de los datos que se transmiten por la red. Son independientes de la red LoRaWAN.

Figura 10.

Ejemplo de elementos en red LoRaWAN.



Fuente: (Prototipo de red LoRaWAN para solución IoT, s. f.)

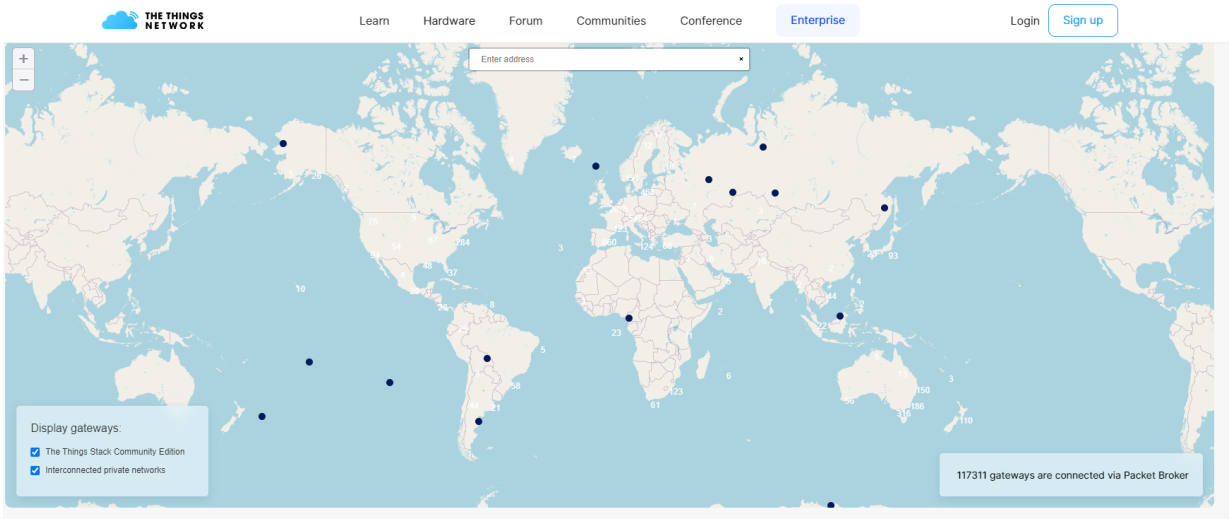
4.3.4. Aplicación LoRaWAN en The Things Network

The Things Network (TTN), es una red de IoT global, descentralizada, gratuita, abierta y en la que cualquier persona puede aportar sus conocimientos acerca de IoT con la finalidad de mejorar cada vez más la plataforma y para crear aplicaciones de bajo costo, seguras y escalables. The Things Network Library permite a las placas basadas en Arduino comunicarse a través de la plataforma The Things Network. (Network, s. f.)

El uso de redes públicas permite la creación rápida de pruebas de concepto sin necesidad de grandes inversiones. Si el proyecto demuestra ser viable, se puede mejorar la infraestructura y reducir los costos generales. Esta estrategia es ampliamente utilizada en Europa y Asia, y se observa un crecimiento constante en su adopción en países de América. Para obtener una visión más precisa, podemos consultar el mapa mundial de cobertura proporcionado por la LoRa Alliance, referenciado en la Figura 11.

Figura 11.

Mapa mundial de cobertura proporcionado por LoRA Alliance.



Fuente: Imagen tomada de https://www.thethingsnetwork.org/map

La visión de The Things Network (TTN) es llevar a cabo las funciones de conectividad de forma descentralizada y distribuida. En este enfoque, cualquier entidad interesada puede establecer su propia red y gestionar su participación, facilitando diversas formas de integración dentro de la

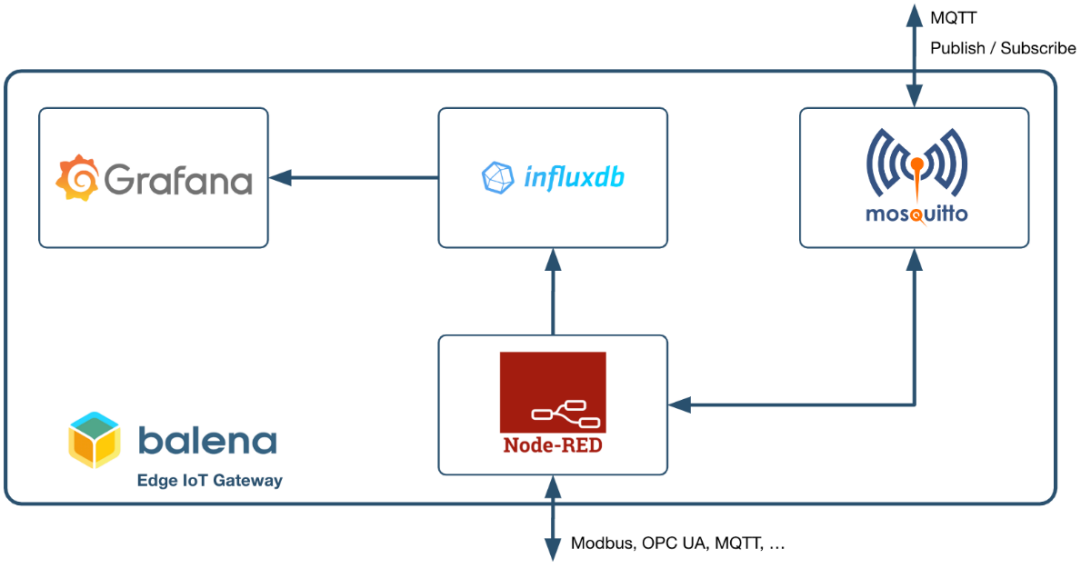
comunidad global. La red de TTN se extiende a más de 150 países, cuenta con más de 20,000 Gateways conectados capaces de atender a millones de dispositivos, y tiene aproximadamente 150,000 miembros.

4.4. Stack M.I.N.G. para visualización y almacenamiento

4.4.1. Introducción al stack MING

El stack MING es una poderosa combinación de tecnologías diseñadas para operar en el borde, permitiendo la recolección, procesamiento y visualización de datos en tiempo real provenientes de dispositivos IoT como PLC industriales o sensores LoRaWAN, que envían información al Servidor de Red LoRa. Estas tecnologías trabajan de manera sinérgica para formar una solución integral de extremo a extremo, facilitando el análisis y la visualización de datos en tiempo real (mpous, 2023). En la Figura 12 se puede apreciar una representación del stack MING y sus principales herramientas:

Figura 12.
Representación del stack MING.

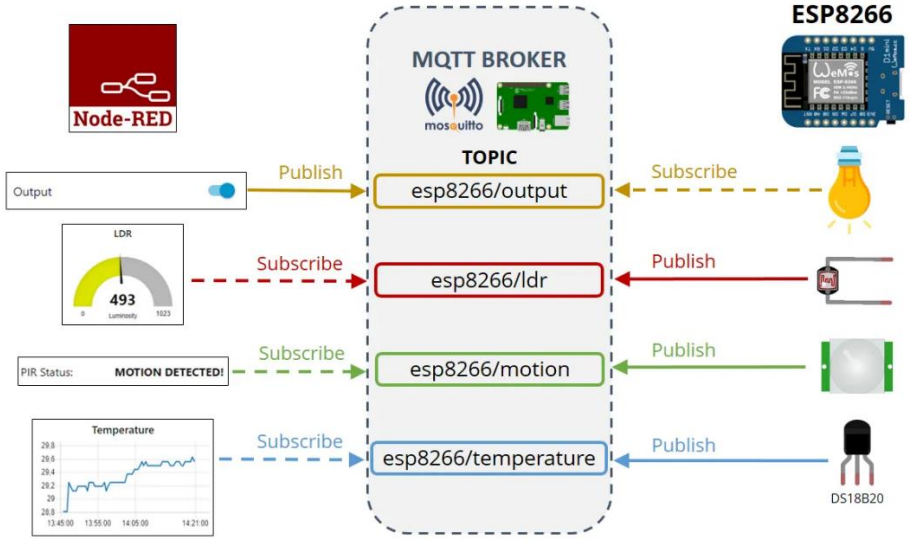


Fuente: Imagen tomada de https://blog.balena.io/ming-stack-mqtt-influxdb-nodered-grafana-balena/

En la Figura 13, podemos ver un ejemplo de la generación de datos por diferentes sensores o dispositivos como estos envían la información a un servidor MQTT y las aplicaciones que acceden a esos datos para monitoreo y gestión de los datos. A continuación, se describen brevemente cada uno de los servicios que se ejecutan en el stack MING:

- ***Mosquito:*** es un intermediario de mensajería MQTT para enviar y recibir datos desde dispositivos de Internet de las cosas.
- ***InfluxDB:*** es una base de datos de series temporales de código abierto y de alto rendimiento que está optimizada para almacenar y consultar grandes volúmenes de datos a altas velocidades.
- ***NodeRED:*** es una herramienta de código abierto que se puede utilizar para crear flujos para dispositivos IoT. Le permite conectar dispositivos, base de datos y servicios entre sí automatizando el intercambio de información entre ellos.
- ***Grafana:*** es una plataforma de código abierto para visualización y monitoreo de datos. Permite crear paneles que pueden mostrar datos en tiempo real de dispositivos y gestionar alertas que pueden notificarle sobre eventos o anomalías importantes. Puede crear paneles personalizados que se adaptan a necesidades específicas y que pueden ayudar a tomar decisiones.

Figura 13.
Ejemplo de intercambio de información por MQTT.



Fuente: Imagen tomada de https://randomnerdtutorials.com/esp8266-multisensor-shield-with-node-red/

5. Metodología

El presente trabajo de investigación se basará en una metodología con enfoque mixto, este enfoque se elige porque la investigación incorpora tanto elementos cualitativos como cuantitativos en la recopilación y análisis de información.

El análisis cuantitativo será esencial para la localización de posibles anomalías en el motor y mediante Machine Learning procesar los mismos a través de un algoritmo de cálculo basado en variables seleccionadas, lo que permite al Arduino ofrecer mejores respuestas y enviar los datos mediante LoRaWAN a la estación base. Por otra parte, el análisis cualitativo resultará importante para seleccionar las tarjetas electrónicas y sensores con los cuales se realizarán las pruebas para implementación del sistema propuesto, la selección se la realizará conforme a la aplicación, características técnicas, dispositivos de uso más común y que respondan de mejor manera a las limitaciones propuestas y definidas en los objetivos específicos para el proyecto de investigación elegido. La propuesta para desarrollar este proyecto de investigación se divide en cuatro etapas:

Etapa I: Diseño del prototipo. El diseño del prototipo consiste en elaborar una base plástica diseñada en Fusion 360 e impresa en 3D que pueda ser acoplada al motor, en su interior contiene los dispositivos electrónicos, como elemento principal un Arduino Nano 33 BLE, el mismo que gracias a los sensores internos que posee permite la adquisición de datos y en base a algoritmos de programación permite estimar estado del motor conforme la vibración producida por el movimiento del mismo, el prototipo integra además componentes electrónicos para comunicación LoRaWAN e indicadores leds para control de estados.

Etapa II: Modelo de aprendizaje en Edge Impulse. Mediante el uso la plataforma Edge Impulse se realiza la implementación de un modelo de aprendizaje automático entrenado para detectar patrones de comportamiento normal (motor encendido / motor apagado) y anomalías determinadas por movimientos fuera de lo normal del motor obteniendo como resultado el código base para el Arduino.

Etapa III: Configuración para envió de datos mediante el uso de dispositivos LoRaWAN. En esta etapa, se configura el módulo Wio – E5 para modificar el programa base obtenido en la etapa anterior y adaptarlo al prototipo como un nodo. Se realizan las configuraciones necesarias en el Gateway Multiplataforma LoRaWAN SenseCAP M2 para que pueda comunicarse con el nodo. Ambos dispositivos se registran en el servidor TTN (The Things Network) para subir información a Internet sobre los eventos detectados por el motor, generando mensajes mediante el protocolo MQTT por cada dato recibido.

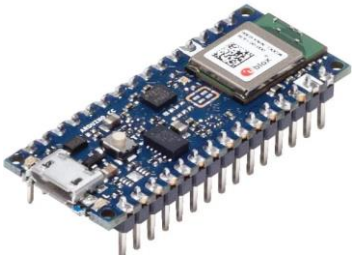
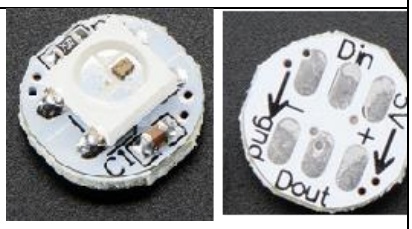
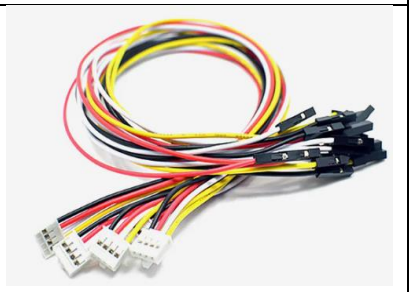
Etapa IV: Implementación del sistema integral de almacenamiento y visualización conocido como MING. En la etapa final, se utilizan los datos enviados con MQTT por el servidor TTN, se crea un cliente MQTT en NodeRED para decodificar y organizar la información recibida. La información útil se envía para ser almacenada en la base de datos de InfluxDB, y, finalmente se habilita un panel para visualización de información en Grafana que exhibe tendencias, alarmas y eventos relacionados con el funcionamiento del motor y las anomalías detectadas por el modelo TinyML configurado en el Arduino Nano 33 BLE. En esta etapa se evalúa el funcionamiento final del prototipo y se realiza un registro de los resultados obtenidos.

5.1.Etapa I: Diseño del prototipo

Tabla 1.

Listado de componentes electrónicos utilizados en el desarrollo del prototipo

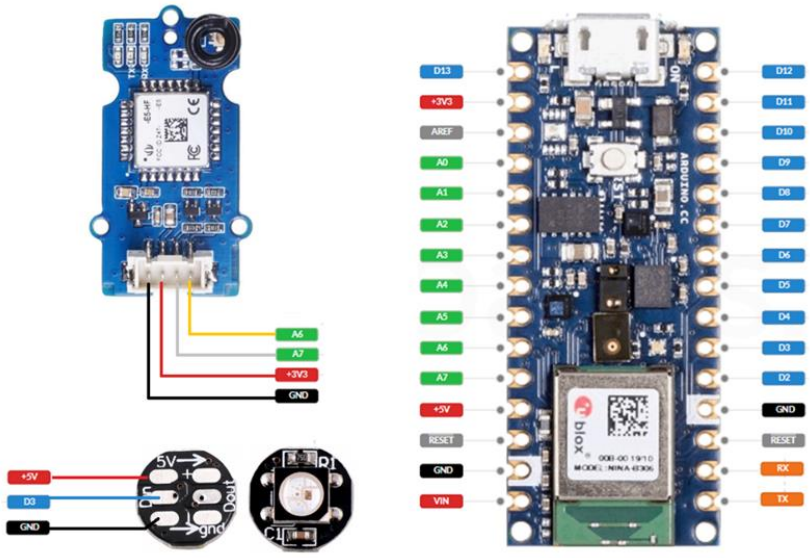
Item	Componente	Características principales	Referencia fotográfica
1	Motor DC Brushless	Motor de ventilador de pc, dimensiones 60*60*25, funciona con 5V 0.25 A. Motor elegido para determinar estados.	

2	Arduino Nano 33 BLE Sense	Se basa en el microcontrolador nRF52840. Se programa en Arduino IDE y para este proyecto se utiliza el sensor interno LSM9DS1(Módulo inercial iNEMO: acelerómetro, giroscopio y magnetómetro de 3 ejes). Referencia: https://store.arduino.cc/products/arduino-nano-33-ble-sense	
3	Grove - Wio-E5	Permite la configuración del Arduino como nodo para enviar datos al Gateway LoRaWAN con la frecuencia US915. Módulo de bajo consumo energético y fácil configuración. Referencia: https://wiki.seeedstudio.com/Grove_LoRa_E5_New_Version/	
4	Led Neopixel	Led colocado como accesorio al prototipo y configurado como indicador visual para detección de error y cambios de estados. Referencia: https://www.adafruit.com/product/1612	
5	Complementos para motor DC	Adaptador de 5V 1 A para poner en funcionamiento el motor, interruptor para encendido apagado y adaptador para ajuste de cable en prototipo. Tornillos y tuercas para fijar la base plástica al motor.	
6	Cables para conexión adicionales	Cables para conectar módulo Grove E5 y el Led Neopixel con el Arduino nano 33 BLE.	

Fuente: El autor

En la Tabla 1 se enlistan los componentes electrónicos que se utilizaron para el prototipo. Estos componentes se encuentran conectados de acuerdo con el esquema que se aprecia en la Figura 14, el módulo Wio E5 se conecta a los pines A6, A7, 3V3 y GND respectivamente mientras que el Led RGB neopixel se conecta a 5V, GND y al pin digital D3, esos pines son los que van a ser configurados mediante software para habilitar la comunicación LoRaWAN y el led indicador de estados.

Figura 14.
Conexiones realizadas entre Arduino Nano 33 BLE, módulo LoRaWAN y led indicador.



Fuente: El autor

El diseño para la estructura del prototipo y acople de componentes electrónicos con el motor de referencia es el que se propone en la Figura 15, en el mismo presenta un diseño modular se aprecia que consta de 4 partes:

- ***La tapa superior:*** diseñada con dimensiones específicas para adaptar fácilmente el módulo Wio E5, sirve como soporte del mismo y cuenta con aberturas para permitir la conexión con el Arduino Nano 33 BLE.

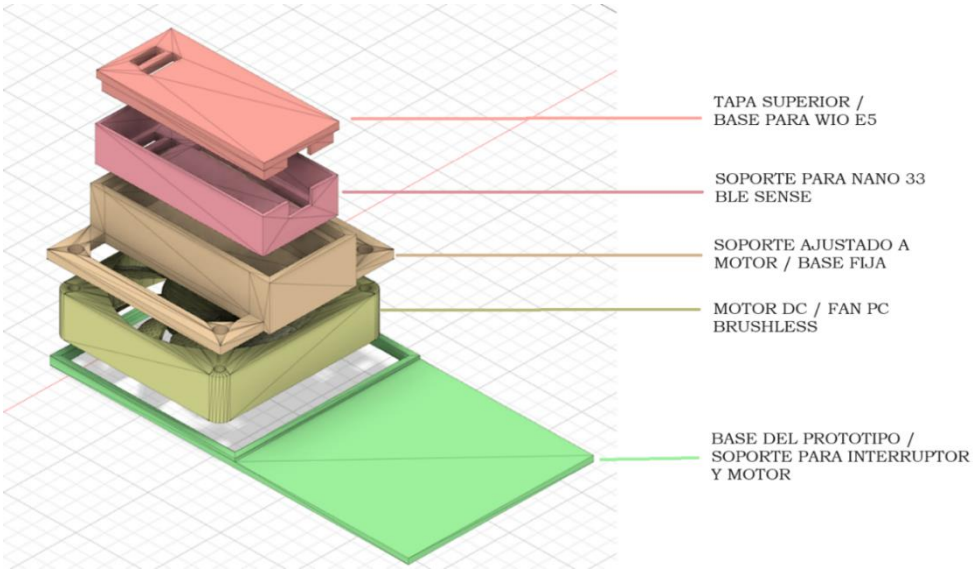
- ***Soporte para Arduino Nano 33 BLE:*** este soporte permite colocar al Arduino Nano 33 BLE sobre peinetas hembra a fin de desmotar si se requiere. Además, cuenta con

peinetas macho para conexión con Wio E5 y led neopixel. Las conexiones entre los diferentes componentes electrónicos se encuentran realizadas por la parte inferior. En conjunto con la tapa superior se pueden desmontar de la base ajustada al motor para ser utilizadas en otro motor o en otra aplicación de manera sencilla.

- ***Soporte ajustado a motor:*** esta base se debe ajustar con tornillos y tuercas al motor de manera que formen un solo conjunto y las vibraciones del motor también influyan en la base fija. Sobre esta base, se colocan el soporte para Arduino y la tapa superior con los respectivos componentes electrónicos. Además, cuenta con espacio para incluir una pequeña batería para hacer portátil el prototipo y en la parte posterior se coloca el Led Neopixel para tener referencia de estados.

- ***Base del prototipo:*** esta base cuenta con espacio para colocar el interruptor y el adaptador que alimentará al motor DC para ponerlo en funcionamiento. La base también restringe el movimiento del motor para asegurarnos que quede en una posición específica para adquirir datos y generar respuesta de estado con Edge Impulse.

Figura 15.
Prototipo realizado en Fusion 360.

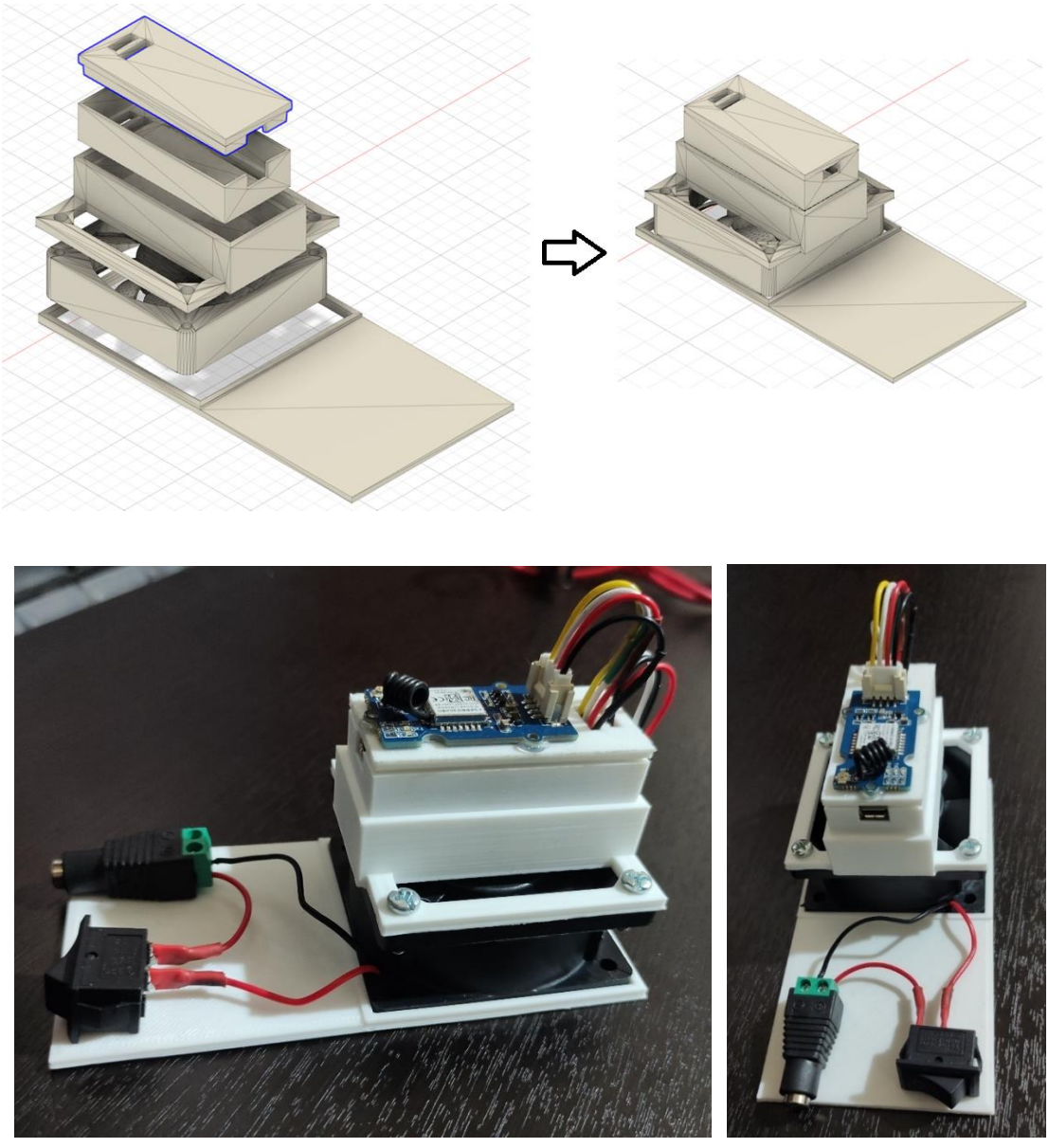


Fuente: El autor

En la Figura 16, se puede apreciar cómo queda el prototipo por ensamblar y ya ensamblado en una representación más realista realizada en Fusion 360 en comparación con el prototipo terminado con los componentes electrónicos ya colocados y listo para realizar pruebas de funcionamiento respectivas.

Figura 16.

Prototipo diseñado en Fusion 360 y prototipo ensamblado listo para pruebas de funcionamiento.



Fuente: El autor

5.2. Etapa II: Modelo de aprendizaje en Edge Impulse

El desarrollo de esta etapa incluye la habilitación del Arduino Nano 33 BLE para adquirir datos con Edge Impulse, generar el modelo de aprendizaje y finaliza al exportar la librería para Arduino, los pasos más importantes se describen a continuación:

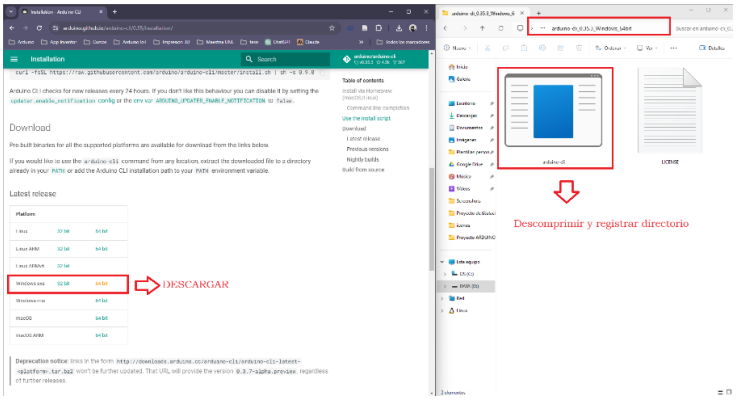
5.2.1. Instalar dependencias para adquirir datos de Arduino con Edge Impulse

Para adquirir las señales del sensor interno del Arduino Nano 33 BLE específicamente del acelerómetro interno que posee, debemos habilitar el mismo para que se puede conectar a la plataforma de Edge Impulse y tomar los datos que permitan entrenar el modelo de aprendizaje. El resultado genera una librería que se podrá modificar para adaptar al prototipo conforme se requiera. Los pasos principales para seguir son:

5.2.1.1. Instalación de Arduino CLI

Lo primero es descargar el CLI de la página web: https://arduino.github.io/arduino-cli/0.35/installation/, descomprimir el archivo descargado y registrar el directorio donde se encuentra el archivo, tal como se aprecia en la Figura 17.

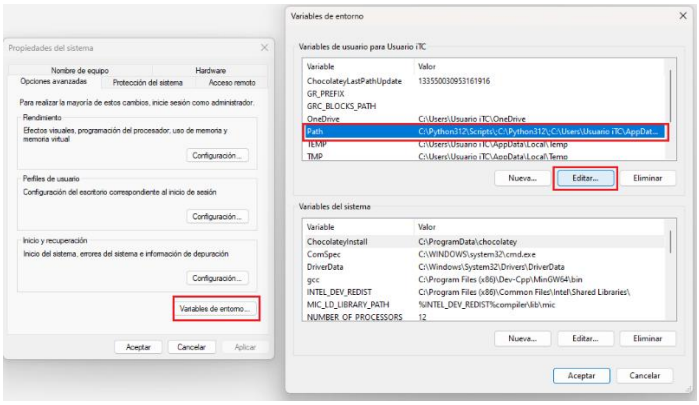
Figura 17.
Instalación del CLI para Arduino nano 33 BLE.



Fuente: El autor

En la Figura 18, se puede observar la configuración y modificación de las variables de entorno agregando un nuevo Path necesarias para habilitar el Arduino CLI.

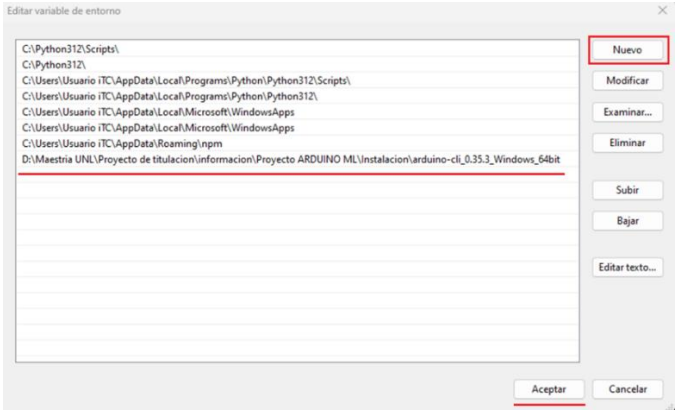
Figura 18.
Modificar variables de entorno para habilitar CLI.



Fuente: El autor

Verificamos el directorio donde se encuentra el archivo descargado previamente y lo registramos como nueva variable de entorno, en mi caso, el directorio registrado es D:\Maestria UNL\Proyecto de titulacion\informacion\Proyecto ARDUINO ML\Instalacion\arduino-cli_0.35.3_Windows_64bit, en la Figura 19 se aprecia el proceso realizado.

Figura 19.
Directorio para habilitar Path del CLI de Arduino.



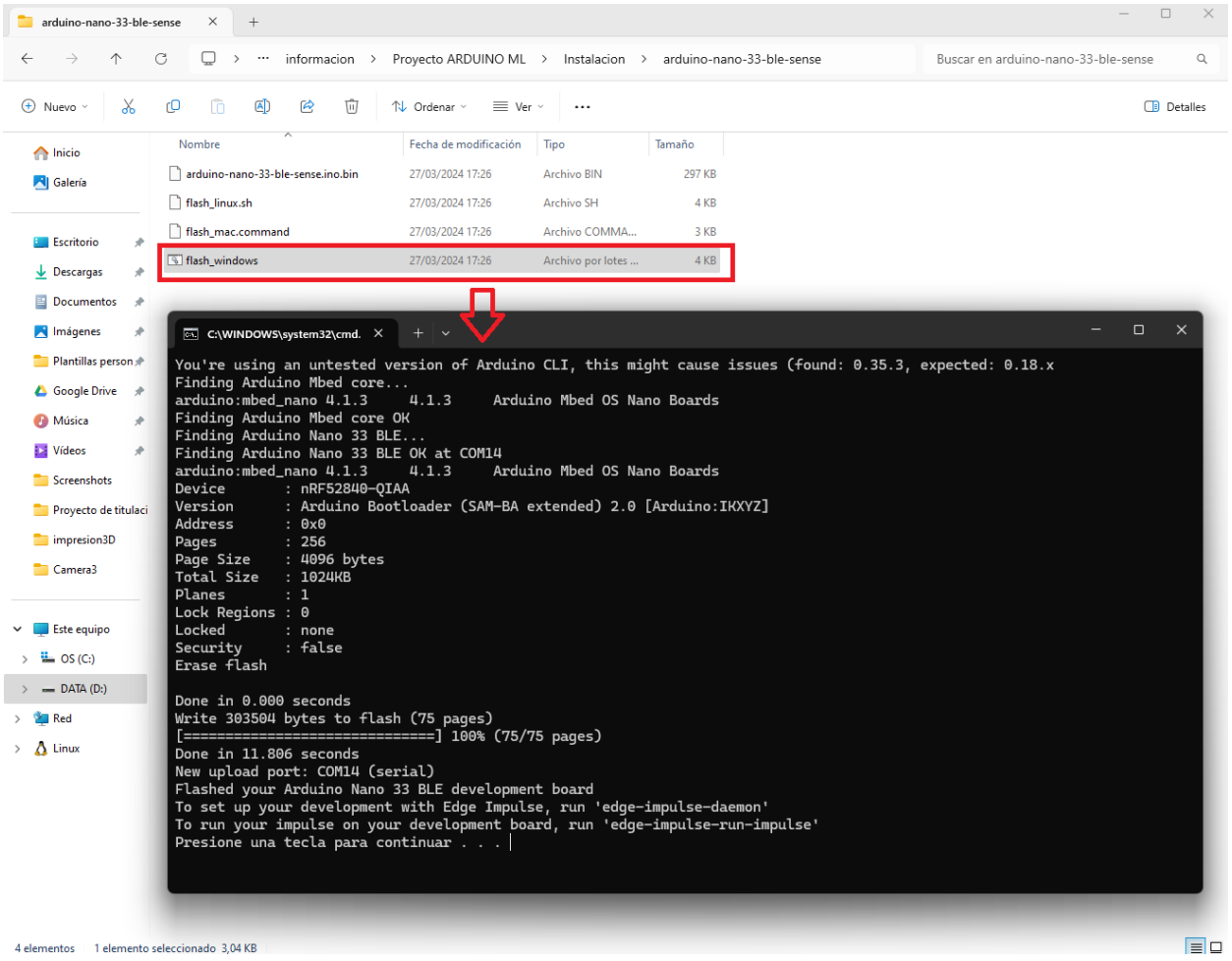
Fuente: El autor

5.2.1.2. Actualización del firmware

El siguiente paso, Figura 20, es actualizar el firmware sugerido por la plataforma Edge impulse para el Arduino nano 33 BLE con el fin que la plataforma reconozca al dispositivo y así poder realizar la adquisición de datos, para realizar este paso se debe ingresar al enlace: https://cdn.edgeimpulse.com/firmware/arduino-nano-33-ble-sense.zip, descargar el fichero comprimido y ejecutar el instalador correspondiente al sistema operativo, para este proyecto se utiliza la versión para Windows.

Figura 20.

Actualización del firmware en el Arduino nano 33 BLE mediante Arduino CLI.

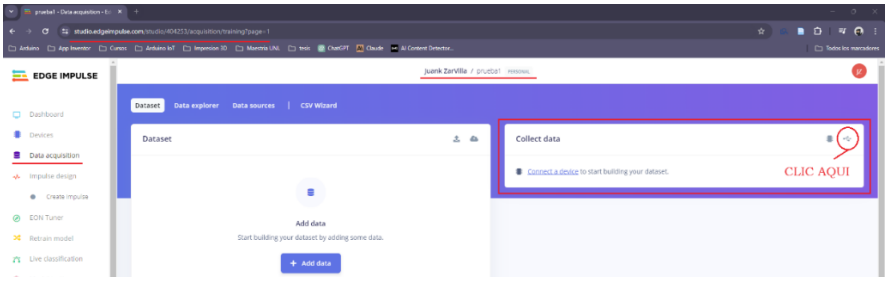


Fuente: El autor

5.2.1.3. Pruebas de comunicación entre Arduino – Edge Impulse

Como paso previo debemos acceder al navegador y estar registrados en la página de Edge Impulse, debemos tener activo un proyecto y en el apartado Data acquisition verificar que la tarjeta permita recolectar los datos y sea reconocida por la plataforma como se aprecia en la Figura 21.

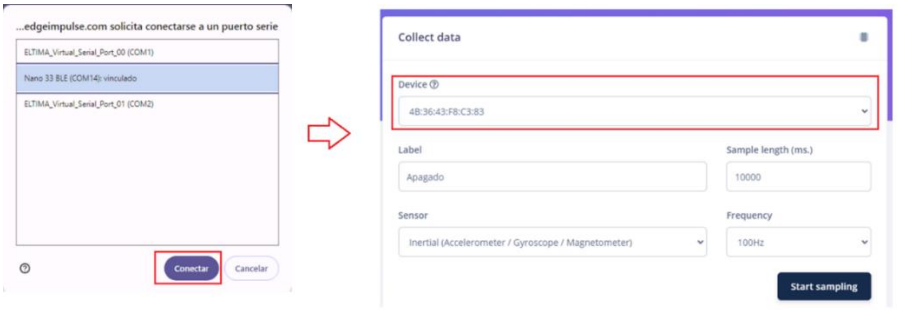
Figura 21.
Prueba de adquisición de datos entre Arduino y Edge Impulse.



Fuente: El autor

En el listado de puertos serie seleccionamos el que corresponde al Arduino Nano 33 BLE y procedemos a conectar el dispositivo. En el apartado Collect data, en caso de que no existan problemas o errores se debe reflejar el dispositivo reconocido, lo que haría falta es llenar los campos adicionales como nombre, sensores, tamaño de muestra y frecuencia para realizar un muestreo de prueba, tal como se aprecia en la Figura 22.

Figura 22.
Reconocer Arduino nano 33 BLE con Edge Impulse.



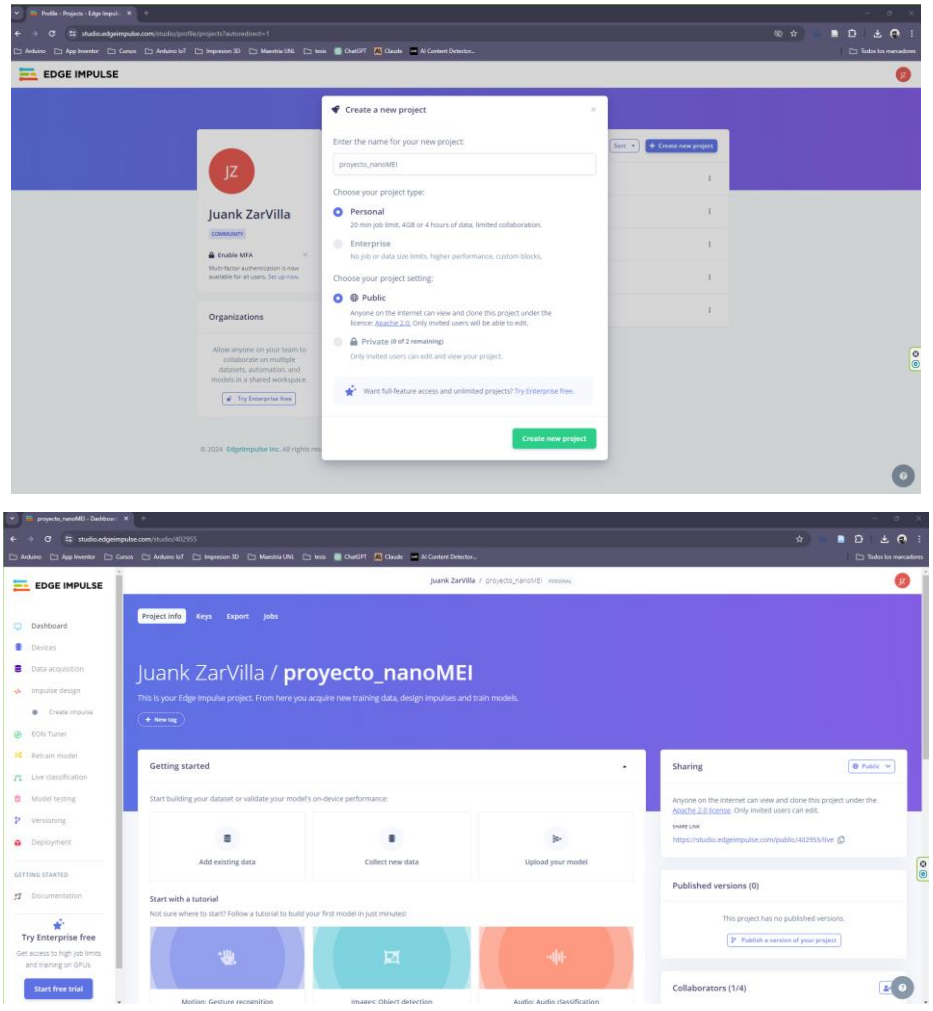
Fuente: El autor

5.2.2. Crear nuevo proyecto y adquirir datos del Arduino

5.2.2.1. Registro en plataforma Edge Impulse y creación de dataset

Para crear un nuevo proyecto previamente debemos registrarnos en la plataforma de Edge Impulse, una vez realizado eso, procedemos a crear un nuevo proyecto al que se ha denominado proyecto_nanoMEI, el tipo de proyecto es personal y público utilizando las herramientas de acceso libre que ofrece el entorno. El resultado de crear el proyecto se aprecia en la Figura 23.

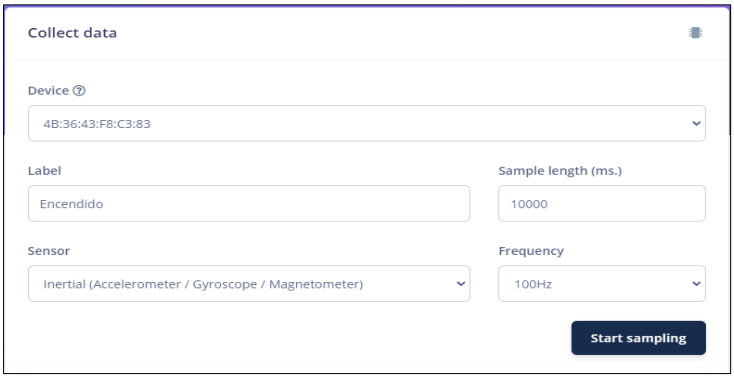
Figura 23.
Crear nuevo proyecto en Edge Impulse.

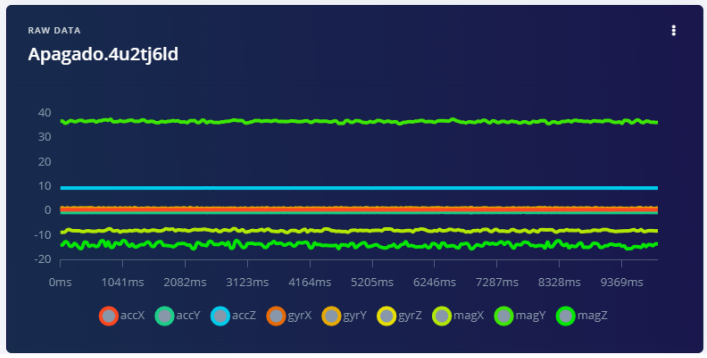
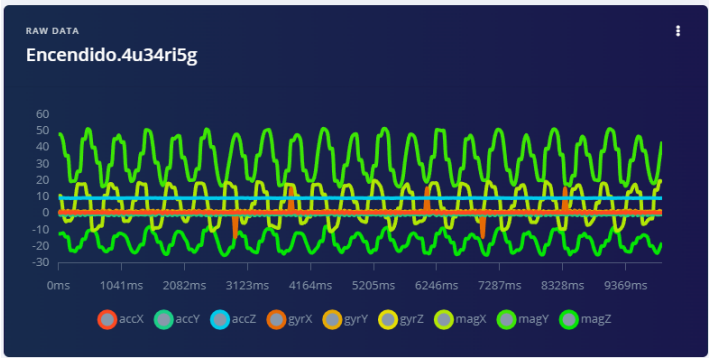


Fuente: El autor

Para crear un nuevo dataset definimos previamente dos estados para el motor Encendido y Apagado, se tomaron todas las muestras utilizando como base una longitud de 10000 ms, una frecuencia de 100 Hz y el Sensor Inercial, el detalle se adjunta en la Figura 24.

Figura 24.
Parámetros predefinidos para generar Dataset.

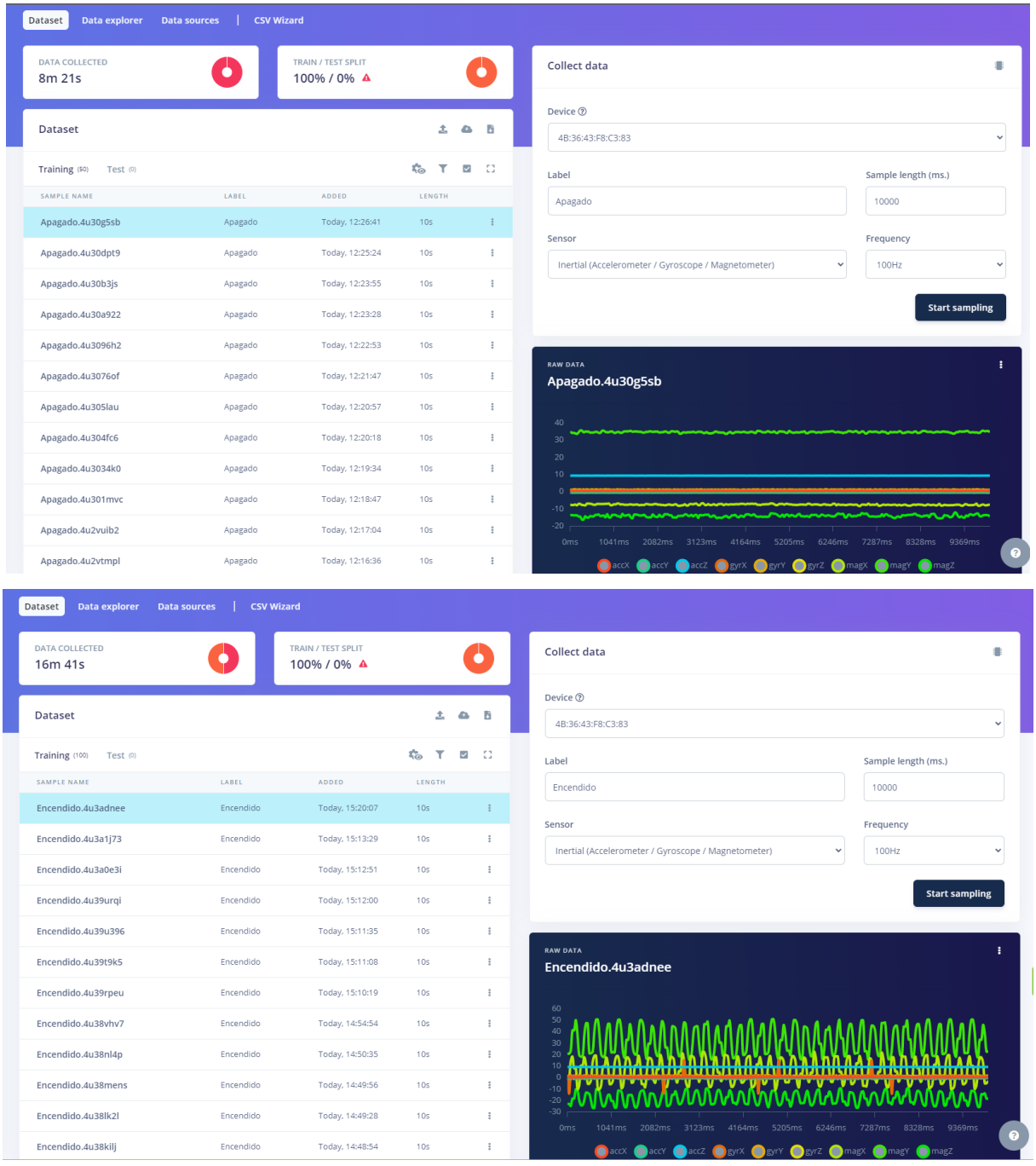


ETIQUETA	TIEMPO MUESTRAS	OBSERVACIÓN
Apagado (Motor detenido - 0V)	Para el tiempo de muestro la referencia es de 0m 0s - 8m 21s con un total de 50 muestras	
Encendido (Motor en funcionamiento - 5V)	Para el tiempo de muestro la referencia es de 8m 21s – 16m 41s con un total de 50 muestras	

Fuente: El autor

En la Figura 25, se observa el resultado de generación del dataset para estados Encendido / Apagado, con un tiempo total de 16 minutos 41 segundos y 100 muestras obtenidas.

Figura 25.
Parámetros predefinidos Encendido / Apagado para generar Dataset.



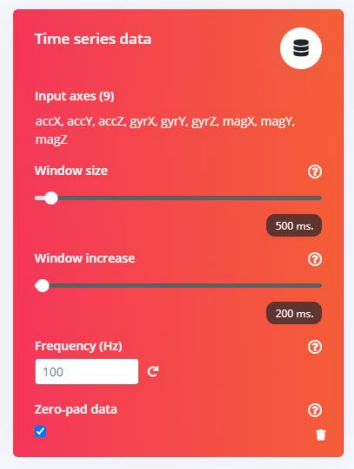
Fuente: El autor

5.3.Diseño del impulso con Edge Impulse

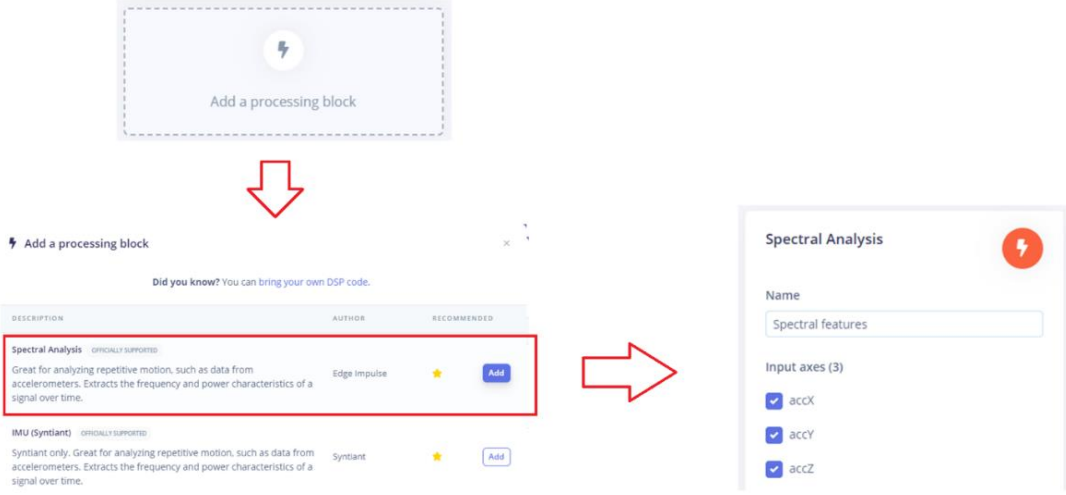
5.3.1. Crear el impulso

A partir del dataset de 100 muestras adquiridas procedemos a crear el impulso para este caso definimos el tamaño de ventana en 500 ms con un incremento de ventana de 200 ms. Agregamos un bloque de procesamiento del tipo análisis espectral y elegimos las variables que corresponden al acelerómetro interno del Arduino Nano 33 BLE, resultado en la Figura 26.

Figura 26.
Creación de impulso: configuración tamaño ventana y frecuencia.



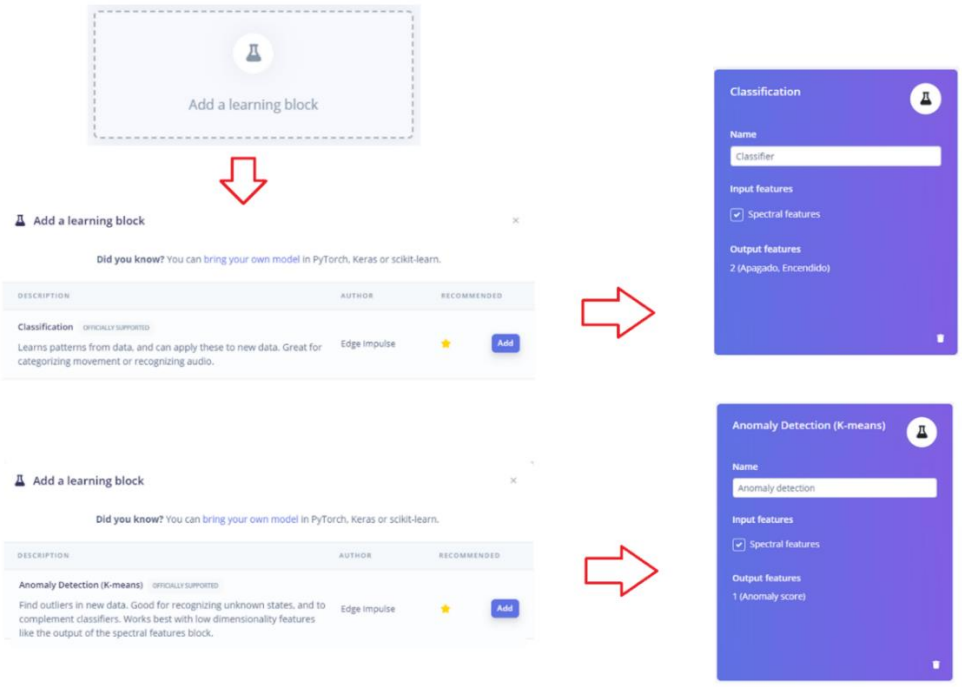
Creación de impulso: agregar bloque de procesamiento para análisis espectral.



Fuente: El autor

Agregamos dos bloques para entrenamiento, uno para clasificación que considere los estados Apagado y Encendido y el segundo bloque de entrenamiento para detección de anomalías, el resultado se aprecia en la Figura 27.

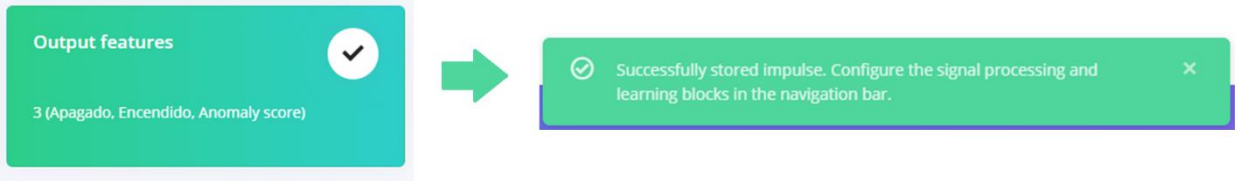
Figura 27.
Creación de impulso: agregar bloque de procesamiento clasificación y detectar anomalías.



Fuente: El autor

Verificamos que todo este correcto y procedemos a guardar el impulso creado, el resultado de realizar este proceso correctamente se aprecia en la Figura 28.

Figura 28.
Parámetros predefinidos para generar Dataset.



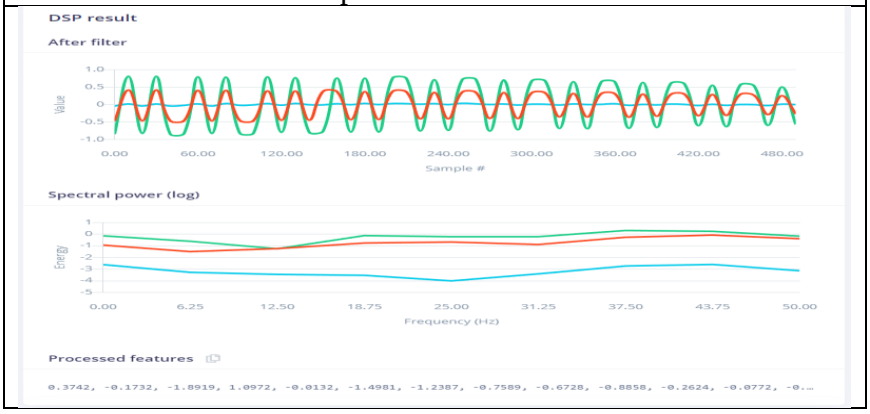
Fuente: El autor

5.3.2. Características especiales

Para este apartado podemos realizar una comparación entre las muestras del dataset que corresponden a los estados Apagado y Encendido correspondientes al funcionamiento normal del motor. Las medidas y gráficas obtenidas para el Apagado son bastante cercanas a 0 mientras que cuando el motor se enciende como producto de las ligeras vibraciones del motor y del prototipo se aprecian que se presentan medidas cuya amplitud oscila entre -1 y 1, resultados expuestos en la Tabla 2. Los valores correspondientes a los parámetros que nos ofrece Edge Impulse por defecto no fueron modificados y procedemos a grabar y generar respuestas.

Tabla 2.
Características principales de las señales respectivas para estados Encendido / Apagado

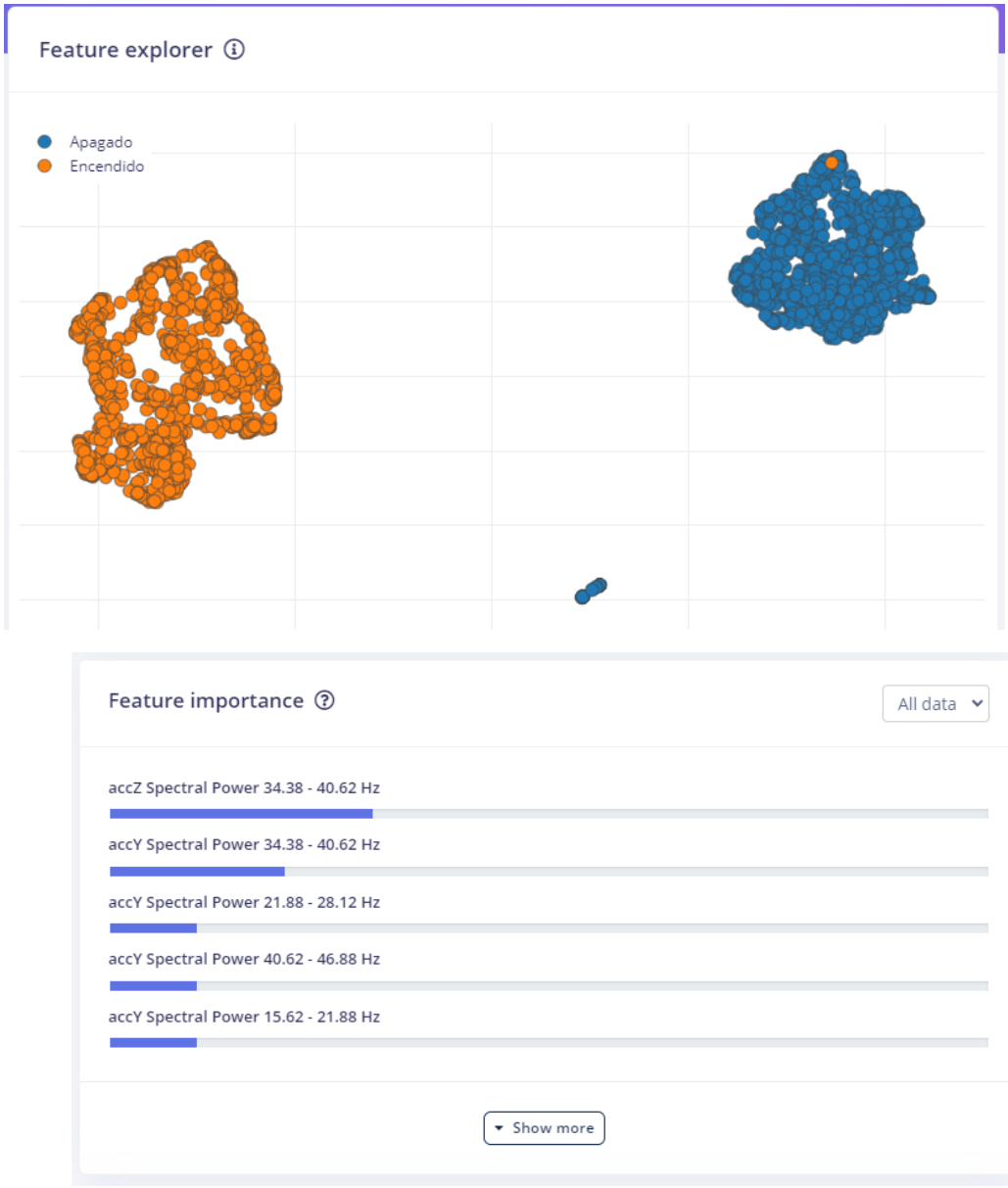
Promedio para muestras Apagado

Promedio para muestras Encendido


Fuente: El autor

En la Figura 29, podemos observar los diferentes comportamientos de las muestras obtenidas en el modelo de entrenamiento, así como las principales variables del sensor que se pueden ocupar como referencia para generar la librería para el Arduino.

Figura 29.
Características especiales a tomar como referencia para modelo de aprendizaje.



Fuente: El autor.

5.3.3. Clasificador ENCENDIDO / APAGADO

El modelo de entrenamiento obtenido en base al dataset generado refleja una precisión de 100%. tal como se aprecia en la Figura 30, para clasificar los valores en base a los estados Encendido y Apagado. En la exploración de la data se puede apreciar de color verde y color amarillo claramente separadas e identificados los estados y finalmente podemos apreciar como resumen que el comportamiento del modelo sobre el dispositivo propone una respuesta rápida y no consume recursos significativos.

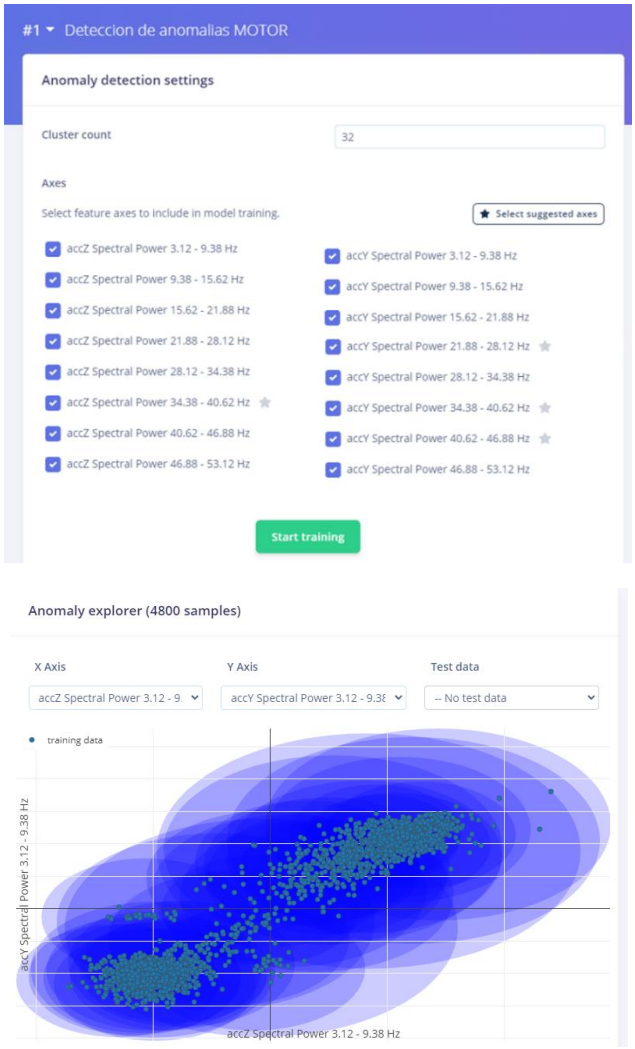
Figura 30.
Modelo de comportamiento generado en base a Dataset.

Fuente: El autor

5.3.4. Detección de Anomalías

Para la configuración de detección de Anomalías para este caso con el prototipo propuesto se seleccionaron el eje Y y el eje Z para los valores a obtener del acelerómetro. Una vez que se inicia el entrenamiento, se puede apreciar en la Figura 31, el explorador de Anomalías en el que se evalúan las muestras que no se clasificarían como estados ni Encendido ni Apagado y corresponden a variaciones significativas sobre los ejes seleccionados detectados como anomalías.

Figura 31.
Detección de anomalías: configuración y explorador gráfico.

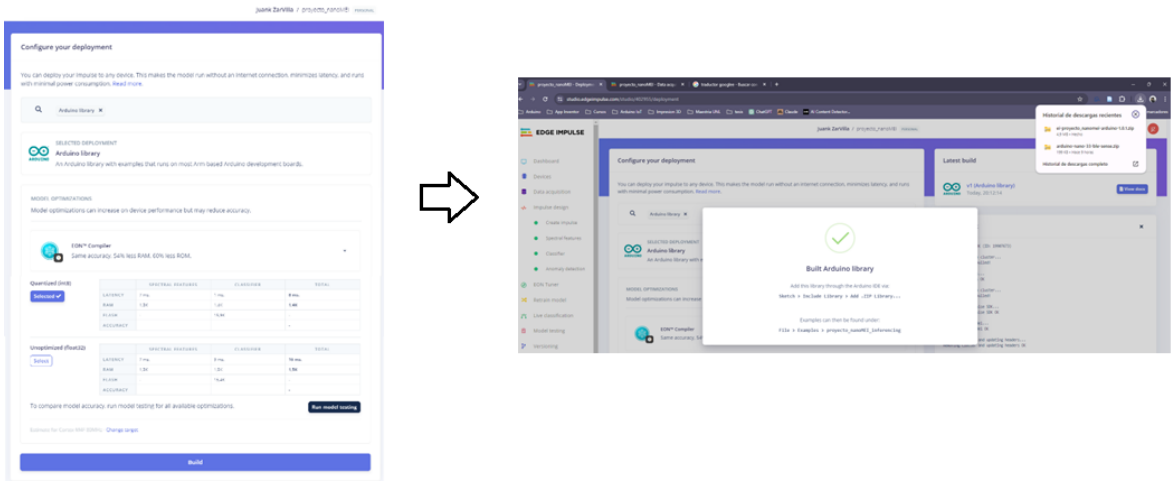


Fuente: El autor

5.3.5. Despliegue de modelo de entrenamiento con librería para Arduino IDE

Una vez culminado el modelo de aprendizaje se procede a generar la librería para Arduino IDE la misma que será compatible con el Arduino nano 33 BLE, en la Figura 32 se adjuntan capturas de la generación de la librería que será el código fuente base para clasificar estados del motor del prototipo.

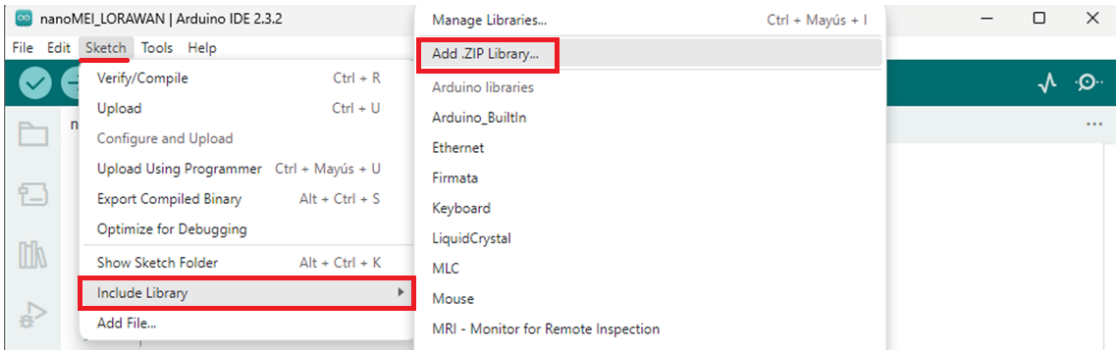
Figura 32.
Parámetros predefinidos para generar Dataset.



Fuente: El autor

En la Figura 33, se indica la librería que se debe instalar de nombre proyecto_nanoMEI_inferencing.zip generada por la plataforma de Edge Impulse.

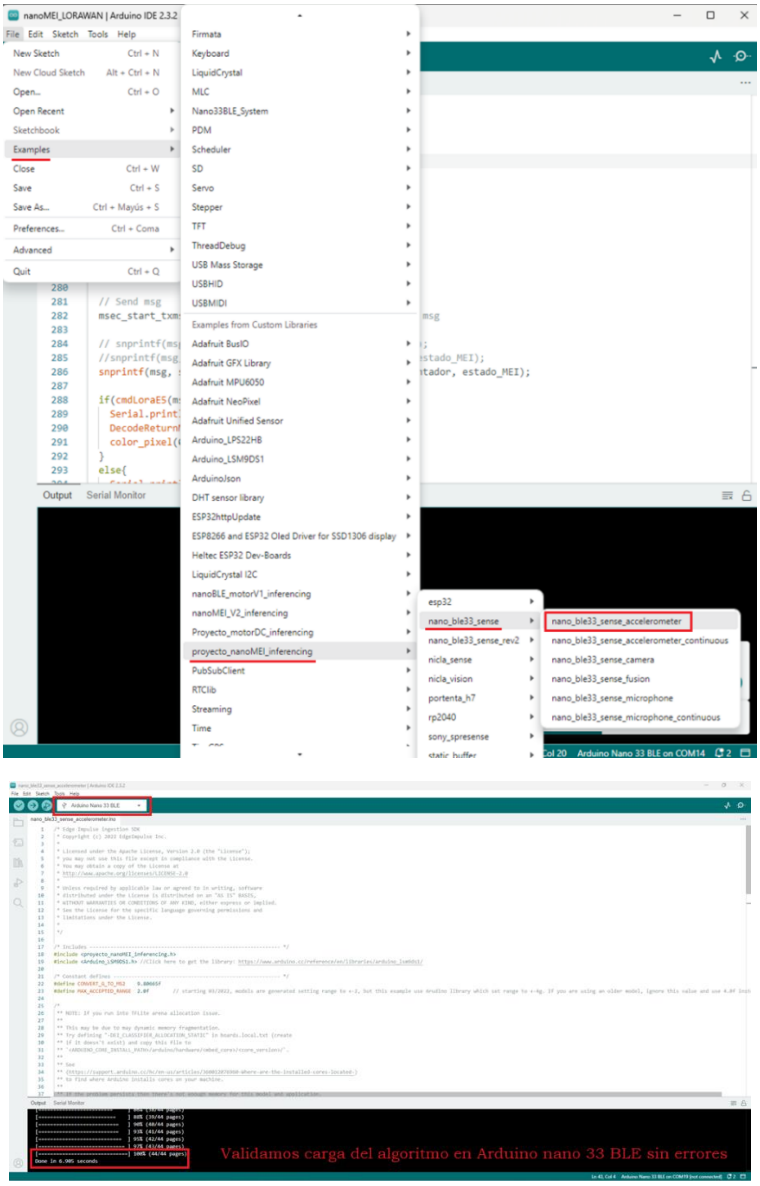
Figura 33.
Parámetros predefinidos para generar Dataset.



Fuente: El autor

En la Figura 34, se procede a seleccionar un ejemplo de la librería previamente instalada y con el prototipo conectado al puerto USB y reconocido el Arduino Nano 33 BLE procedemos a cargar el ejemplo nano_ble33_sense_accelerometer.ino que será el ejemplo de partida sobre el cual se realizaran las modificaciones necesarias para envío de datos por LoRaWAN hacia el servidor TTN.

Figura 34.
Subida de código ejemplo al Arduino nano 33 BLE.



Fuente: El autor

En la Tabla 3, se adjunta captura de las pruebas realizadas con el código base del ejemplo cargado en el Arduino Nano 33 BLE con el fin de validar que realice la clasificación de estados Apagado, Encendido y Anomalía de acuerdo a la interpretación de datos obtenidos por el prototipo.

Tabla 3.

Resultados de ejecución algoritmo de ejemplo descargado desde Edge Impulse

Estado evaluado	**Resultado obtenido monitor serial programa ejemplo librería Edge Impulse**
Apagado (motor detenido)	Starting inferencing in 2 seconds... Sampling... Predictions (DSP: 27 ms., Classification: 0 ms., Anomaly: 6 ms.): Apagado: 0.99609 Encendido: 0.00000 anomaly score: -0.783 Starting inferencing in 2 seconds... Sampling... Predictions (DSP: 27 ms., Classification: 0 ms., Anomaly: 6 ms.): Apagado: 0.99609 Encendido: 0.00000 anomaly score: -0.428
Encendido (motor en funcionamiento)	Starting inferencing in 2 seconds... Sampling... Predictions (DSP: 27 ms., Classification: 0 ms., Anomaly: 6 ms.): Apagado: 0.00000 Encendido: 0.99609 anomaly score: -0.012 Starting inferencing in 2 seconds... Sampling... Predictions (DSP: 27 ms., Classification: 0 ms., Anomaly: 6 ms.): Apagado: 0.00000 Encendido: 0.99609 anomaly score: -0.273
Anomalías (provocar movimiento anormal sobre el motor)	Starting inferencing in 2 seconds... Sampling... Predictions (DSP: 27 ms., Classification: 0 ms., Anomaly: 6 ms.): Apagado: 0.00391 Encendido: 0.99609 anomaly score: 1.192 Starting inferencing in 2 seconds... Sampling... Predictions (DSP: 27 ms., Classification: 0 ms., Anomaly: 6 ms.): Apagado: 0.00391 Encendido: 0.99609 anomaly score: 1.636

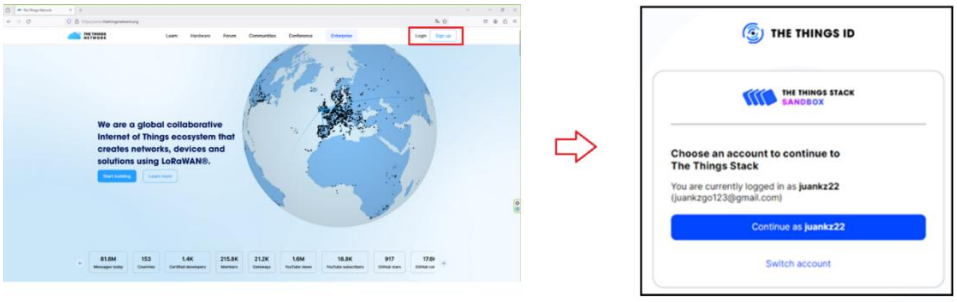
Fuente: El autor

5.4.Etapa III: Configuración para envió de datos mediante el uso de dispositivos LoRaWAN

5.4.1. Registro de dispositivos en The Things Network (TTN)

Previo a registrar dispositivos en la plataforma de TTN debemos ingresar a la página web oficial y crear una cuenta en la misma: https://www.thethingsnetwork.org/, una vez registrados accedemos al modo consola. Ejemplo de realizar el registro en TTN se aprecia en la Figura 35.

Figura 35.
Página principal para registro en plataforma TTN.

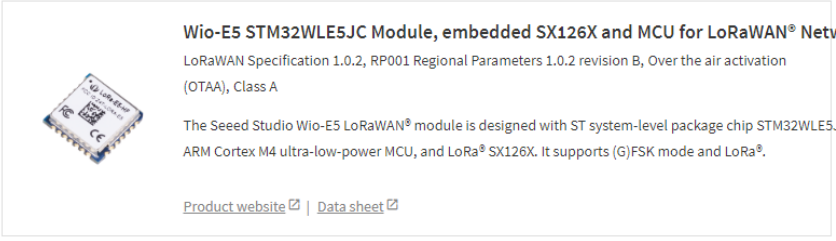


Fuente: El autor

5.4.1.1.Registro y configuración de aplicaciones – nodo en The Things Network

Para el registro de nodos y configuración de aplicaciones en la plataforma de TTN se utiliza como referencia el módulo LoRaWAN Wio – E5, las características principales se pueden observar en la plataforma de TTN, representado en la Figura 36, destacando la especificación LoRaWAN, parámetros regionales y frecuencia.

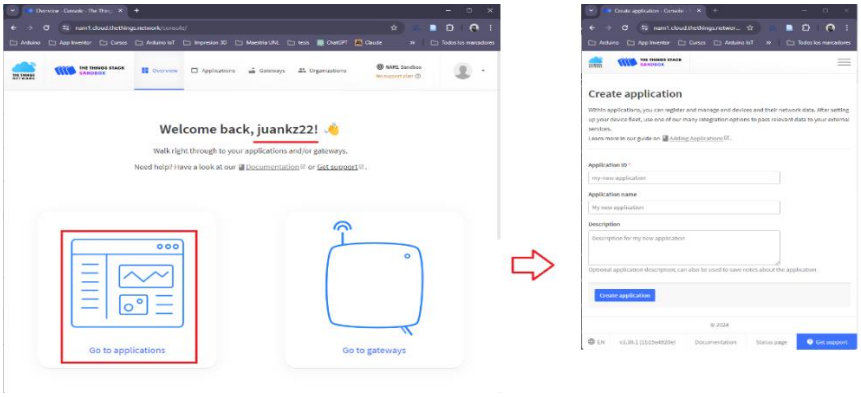
Figura 36.
Referencia para datos del módulo Wio – E5.



Fuente: El autor

Para crear las aplicaciones debemos estar registrados en la plataforma de TTN, accedemos a la consola, seleccionamos aplicaciones y crear una nueva. Procedemos a registrar los datos requeridos tal como se aprecia en la Figura 37.

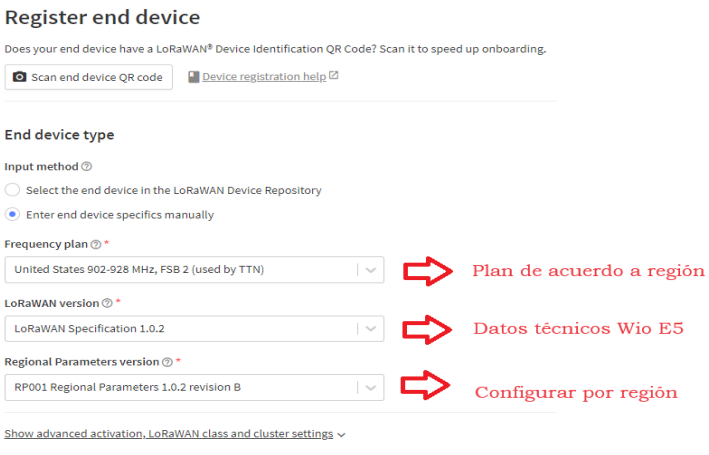
Figura 37.
Registro de aplicaciones en TTN.



Fuente: El autor

Con la referencia de la Figura 36 configuramos los parámetros del Wio – E5, registrando la frecuencia de operación, versión LoRaWAN, parámetros regionales tal como se aprecia en Figura 38.

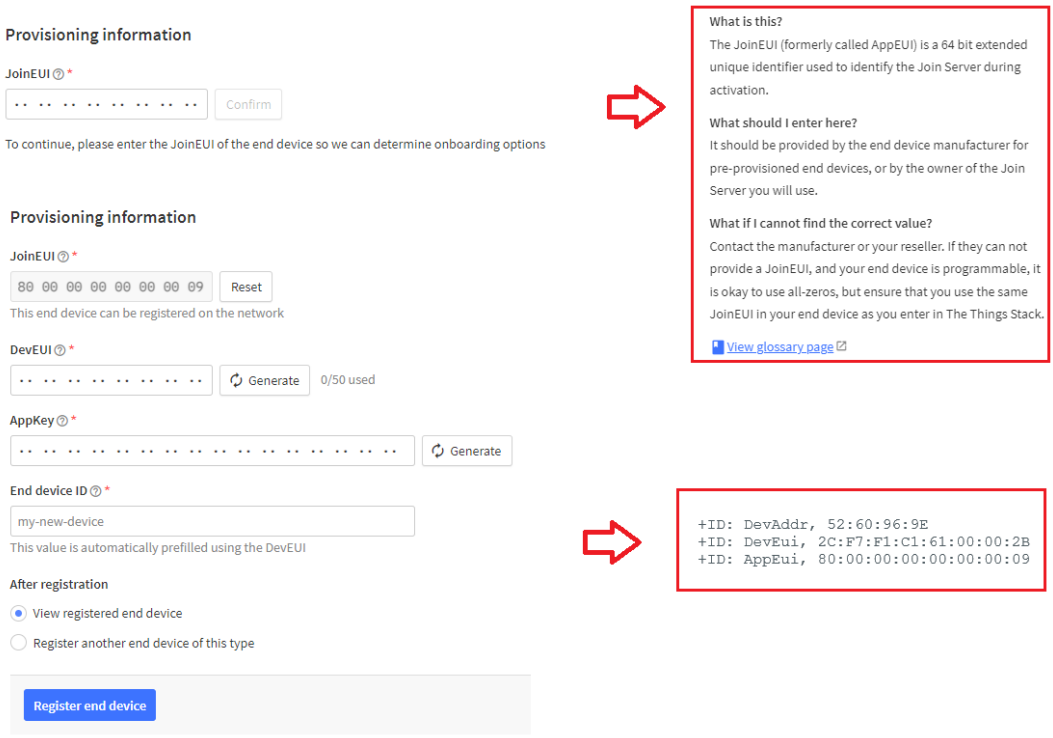
Figura 38.
Parámetros predefinidos para registrar módulo Wio – E5 en TTN.



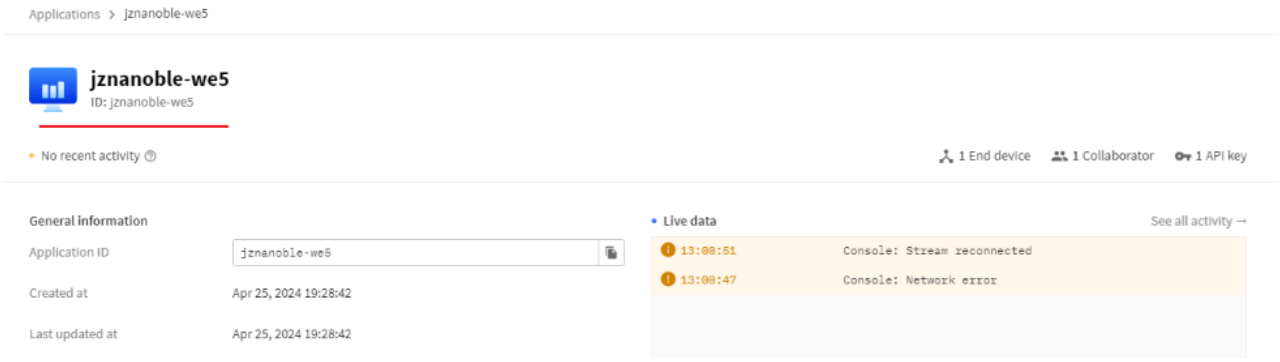
Fuente: El autor

Mediante comunicación serial entre el Arduino Nano 33 BLE y el módulo Wio-E5 realizamos consultas por comandos AT para identificar los parámetros JoinEUI (AppEui) y el DevEui para completar el registro del dispositivo en el servidor TTN, el resultado se aprecia en la Figura 39.

Figura 39.
Información necesaria para registro de dispositivo en TTN.



Al finalizar el registro del dispositivo en la plataforma de Edge Impulse debemos obtener como resultado que la aplicación ya se encuentra registrada en la plataforma a la espera de los datos.

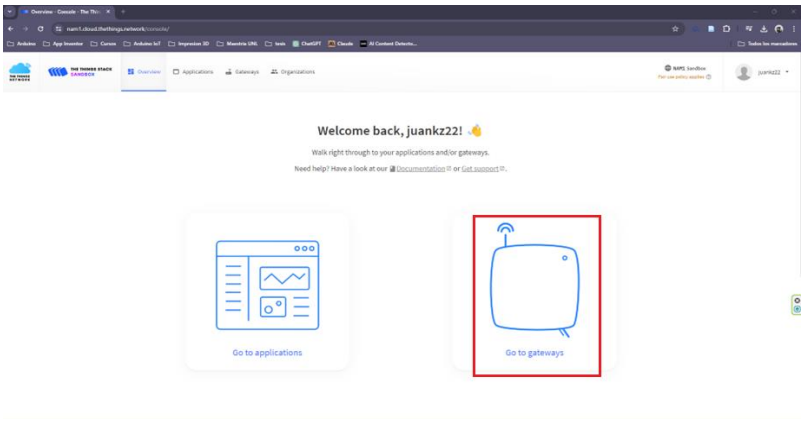


Fuente: El autor

5.4.1.2.Registro y configuración de Gateway en The Things Network

Para realizar el registro del Gateway Multiplataforma M2 – US915 en TTN debemos acceder al modo consola de la plataforma, Figura 40, y seleccionar la opción ***Go to gateways***, de manera simultánea podemos configurar el Gateway para obtener la información que solicita el registro del mismo.

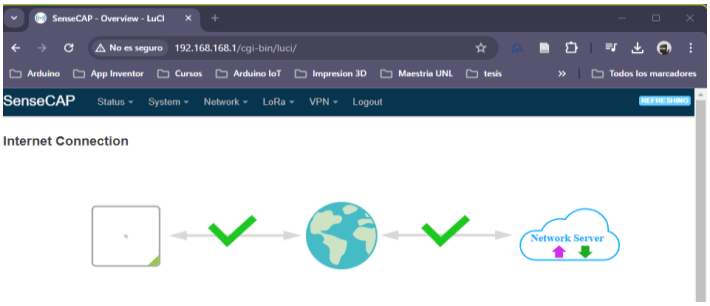
Figura 40.
Acceso a registro del Gateway en la plataforma TTN.



Fuente: El autor (Referencia: https://nam1.cloud.thethings.network/console/)

Conforme sugiere el fabricante del Gateway lo primero es habilitar el servicio de internet para el dispositivo, en este caso se lo realizó configurando los parámetros de red para que se conecte de manera inalámbrica a una red privada por Wifi, se verifica acceso en la Figura 41.

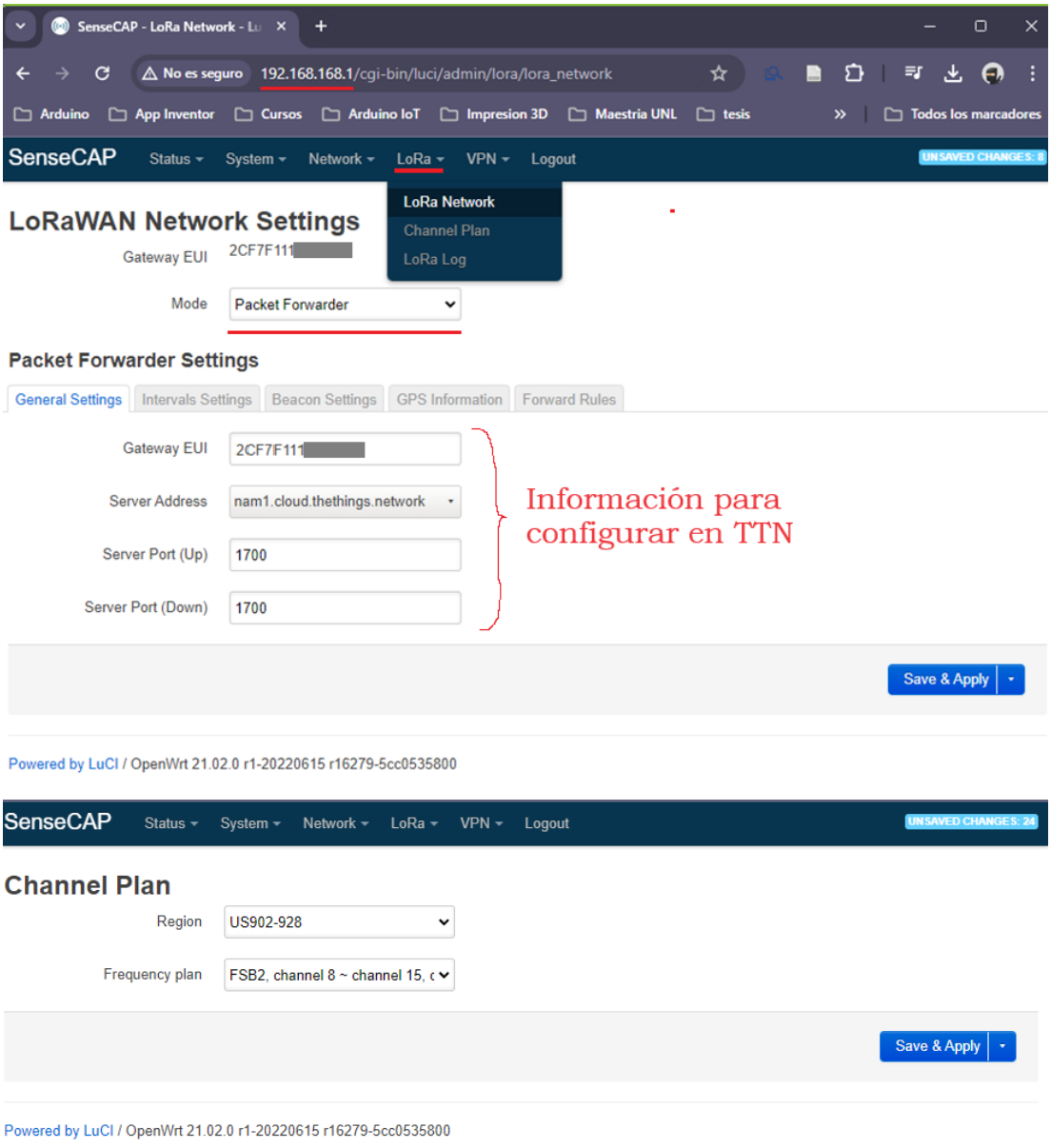
Figura 41.
Verificación de acceso a internet del Gateway M2.



Fuente: El autor

Para habilitar la comunicación de la red LoRa debemos configurar los parámetros que se aprecian en la Figura 42, los datos a registrar son dirección de servidor, puertos para subida y descarga de datos, información del dispositivo, el plan de canal a configurar por Región y la frecuencia principal que será utilizada para establecer comunicación con dispositivos o nodos.

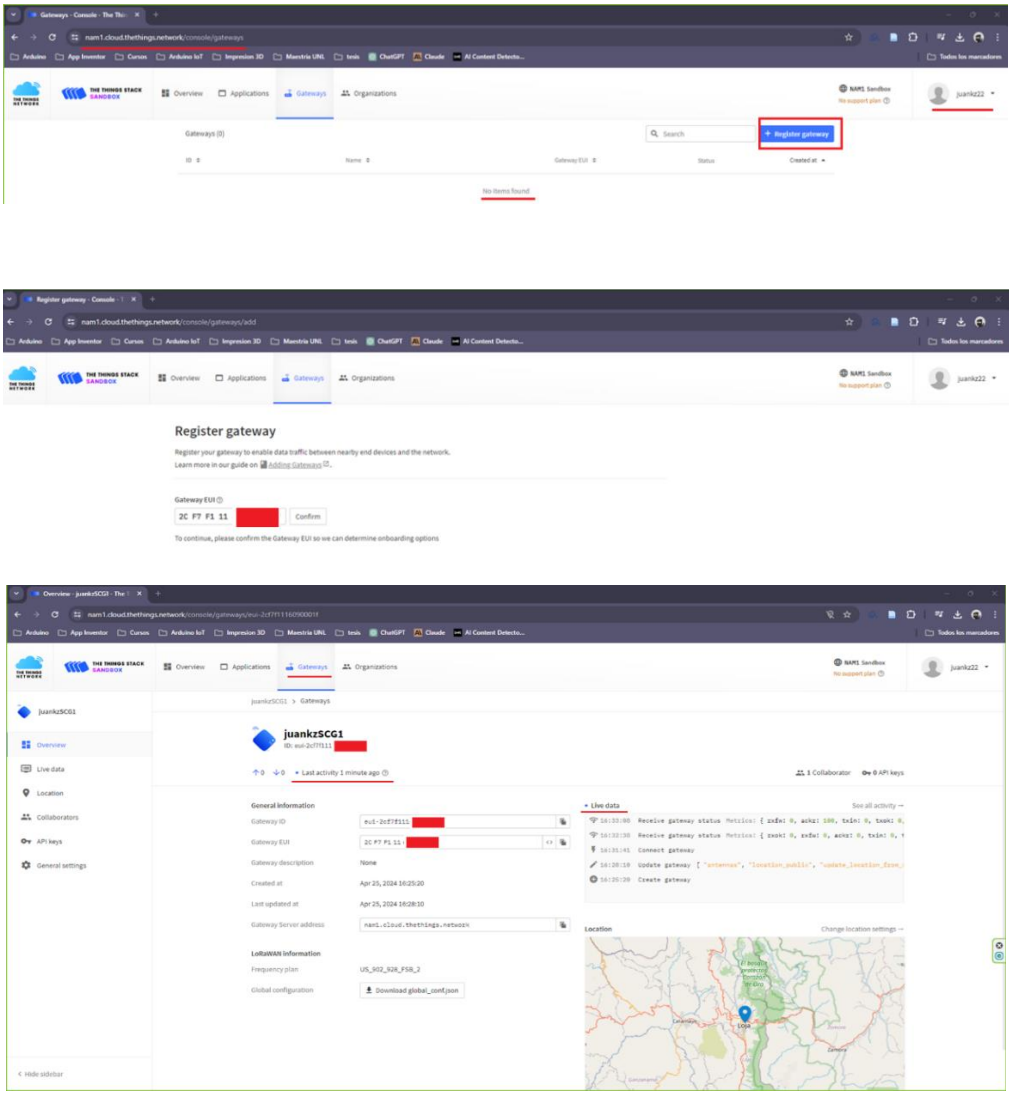
Figura 42.
Configuración para habilitar red LoRaWAN.



Fuente: El autor

Una vez configurado el Gateway para que tenga acceso a Internet y los parámetros para conectividad con LoRa, procedemos a registrar el Gateway en TTN. El dato requerido es el EUI que es único para cada dispositivo y se lo puede obtener físicamente del dispositivo que viene como etiqueta incluida en la parte inferior o en su defecto por medio del paso anterior, ese dato se debe registrar en la plataforma de TTN, así como la frecuencia del plan para la región, un nombre y el ID con el que será reconocido el Gateway en TTN, al completar el registro se puede ver el resultado en la Figura 43.

Figura 43.
Registro del Gateway M2 en la plataforma TTN.



Fuente: El autor

5.4.2. Algoritmo de programación para Arduino Nano 33 BLE

Una vez realizadas esas configuraciones y guardados los cambios esperamos un momento y el Gateway ya debe estar activo y en línea en la plataforma TTN. Las principales secciones del código con el que se programó el Arduino Nano 33 BLE se explican a continuación:

a) Iniciamos declarando librerías y variables necesarias para funcionamiento

```
//Incluimos las librerias necesarias para que funcionen los leds neopixel y envio de datos
#include <Adafruit_NeoPixel.h>
#include <Arduino.h>
#include <string.h>
#include <stdint.h>
#include <stdlib.h>
#include <proyecto_nanoMEI_inferencing.h>
#include <Arduino_LSM9DS1.h>

```

Las variables declaradas incluyen pines de conexión para Led Neopixel colores y tiempos de encendido apagado del led, puertos de conexión del módulo Wio - E5 y variables para almacenar lectura de estado de Edge Impulse y envío de datos a TTN. Así como el retardo a esperar por cada lectura de datos realizado.

b) En la función setup () configuramos el led neopixel como indicador visual de eventos, habilitamos el puerto serial para comunicación con el módulo Wio – E5 por comandos AT y el monitor serial para revisar cambios de estado en Arduino IDE, validamos que exista comunicación y lectura del acelerómetro interno de nano BLE.

```
/*******************************************************************************
*                          SUBRUTINA DE CONFIGURACION                          *
*******************************************************************************/
void setup()
{
    // Configuracion para leds neopixel
    leds_neopixel.begin();
    leds_neopixel.setBrightness(255);
    leds_neopixel.show();
    color_pixel(0, apagado, retardo);    // Inicializamos los leds apagados

    // Configuramos puertos de comunicacion serial
    Serial.begin(115200);
    LoraSerial.begin(9600);
    Serial.println("Inicio de configuracion modulo LORAWAN E5\r\n");

    iniciaLORAWAN();        // Configuracion de modulo E5 por comandos AT
    verificaLORAWAN();      // Verificamos comunicacion establecida E5 a GATEWAY

    // Verificamos comunicacion entre en nano 33 BLE y pc para lectura del IMU
    while (!Serial);
    Serial.println("Edge Impulse Inferencing Demo");

    if (!IMU.begin()) {
        ei_printf("Failed to initialize IMU!\r\n");
    }
    else {
        ei_printf("IMU initialized\r\n");
    }

    if (EI_CLASSIFIER_RAW_SAMPLES_PER_FRAME != 3) {
        ei_printf("ERR: EI_CLASSIFIER_RAW_SAMPLES_PER_FRAME should be equal to 3 (the 3 sensor axes)\n");
        return;
    }
}
```

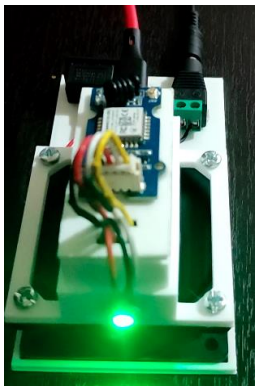
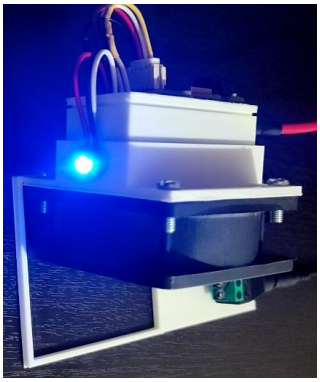
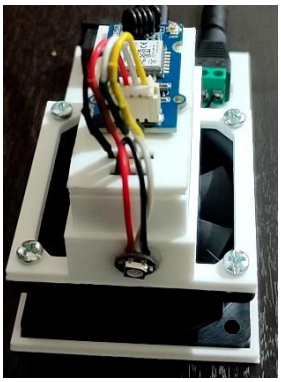
c) En el programa principal la función loop() esta resumida de la siguiente manera:

```
/*****************************************************************************************
*                                    SUBRUTINA PRINCIPAL                                 *
*****************************************************************************************/
void loop() {
  leerDATOS();
  enviarDATOS();
  esperaDATOS();
}
```

La función ***leerDATOS(),*** tiene como base el algoritmo generado por la plataforma Edge Impulse para determinar el estado del motor una vez realizada la lectura del acelerómetro interno del Arduino Nano 33 BLE, en base al valor del clasificador y uso de preguntas se establece el valor de la variable estado_MEI como resultado al finalizar la ejecución de la función. La función ***enviarDATOS(),*** permite una vez que se conoce el valor de la variable estado_MEI, y el módulo Wio – E5 no presente problemas de comunicación, realizar él envió al Gateway y registrar el dato a la vez en la plataforma TTN. La función ***esperaDATOS(),*** es una función cuya función es generar un retardo para nueva lectura de datos, el tiempo estimado de lectura de nuevo dato está configurado a 30 segundos. En la Tabla 4, se puede observar los diferentes cambios de estados y las respuestas que presenta el Led indicador frente a cada cambio.

Tabla 4.

Estados del prototipo considerados como principales.

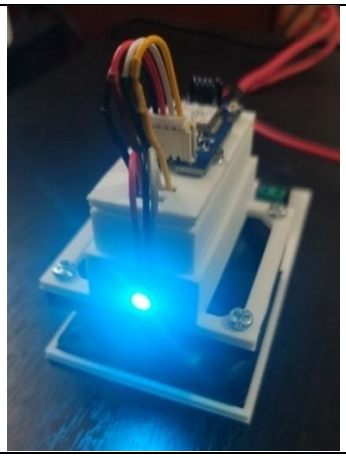
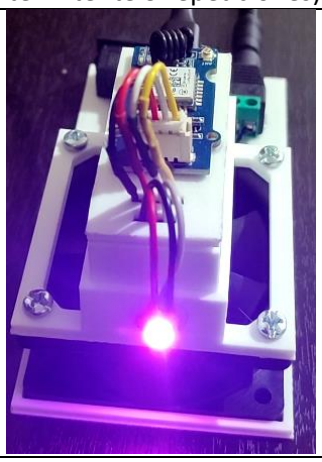
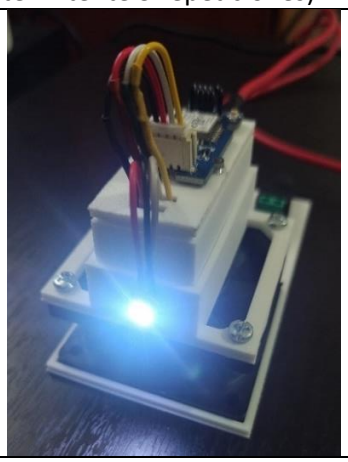
ESTADO 0	**ESTADO 1**	**ESTADO 2**	**INDICADOR APAGADO**
Apagado	Encendido	Anomalía	
Valor predefinido para enviar a TTN = 0	Valor predefinido para enviar a TTN = 1	Valor predefinido para enviar a TTN = 2	Estado de espera para lectura y envió de datos a TTN.
Led indicador rojo	Led indicador verde	Led indicador azul	
			

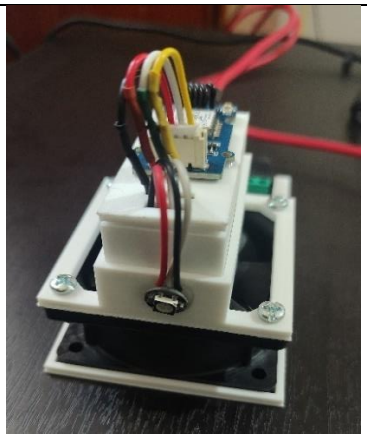
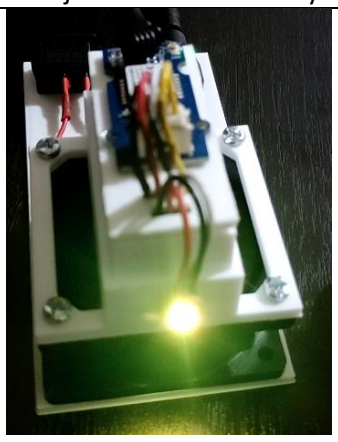
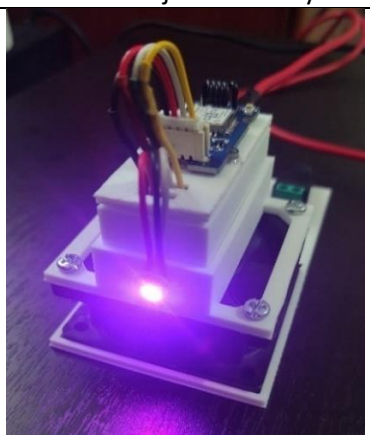
Fuente: Elaborada por el autor

En la Tabla 5 se agregaron imágenes con las respuestas preconfiguradas y que se incluye como parte del código programado en el Arduino Nano 33 BLE para un control de errores. El control de errores ofrece una respuesta para el caso de que la comunicación con el módulo Wio-E5 no sea detectado, cuando el prototipo no se pueda comunicar con el Gateway o si existe problemas con el envío de mensajes obtenidos de la predicción con el algoritmo de Edge Impulse.

Tabla 5.

Especificaciones para el led indicador ante posibles cambios de estado.

INDICADOR DE ENCENDIDO	ERROR MODULO WIO E5	ERROR CON GATEWAY M2
Indica inicio de configuración de módulo Wio - E5	Error de comunicación con modulo Wio - E5	Error de comunicación con Gateway M2
Led indicador celeste	Led indicador violeta (intermitente 3 repeticiones)	Led indicador blanco (intermitente 3 repeticiones)
		

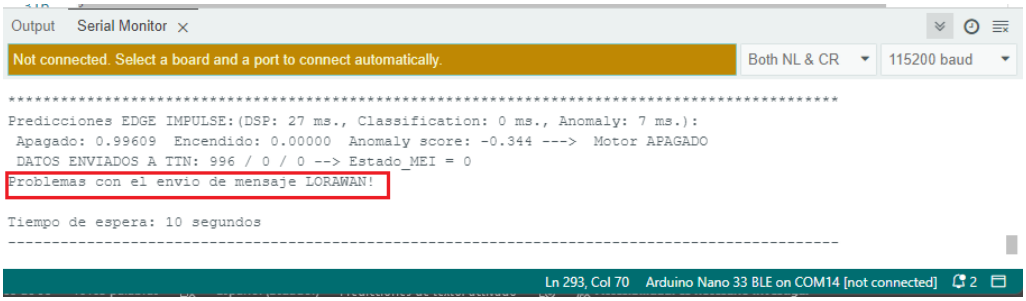
ESPERA DE DATOS	**ENVIO CORRECTO A TTN**	**ERROR ENVIO DATOS A TTN**
Led apagado a la espera de nueva lectura de datos.	Led color amarillo indica mensaje enviado a Gateway.	Led color violeta indica no enviado mensaje a Gateway.
		

Fuente: Elaborada por el autor

Como resultado se presenta la Figura 44, en la misma se aprecia que de presentarse un problema en la comunicación entre el Arduino y el módulo por el momento en el código se encuentra comentada la sección que permite visualizar el detalle en monitor serial pero está claramente identificada, por el momento como mejor opción se dejó configurado el prototipo de tal manera que se realice la predicción del estado del motor y luego del mismo en monitor serial se coloca un mensaje indicando si el mensaje se envió o existe problemas con el envío de datos al Gateway y por ende no se van a transmitir a TTN y al aplicativo de visualización.

Figura 44.

Mensaje en monitor serial indicando problemas al enviar datos por LoRaWAN.

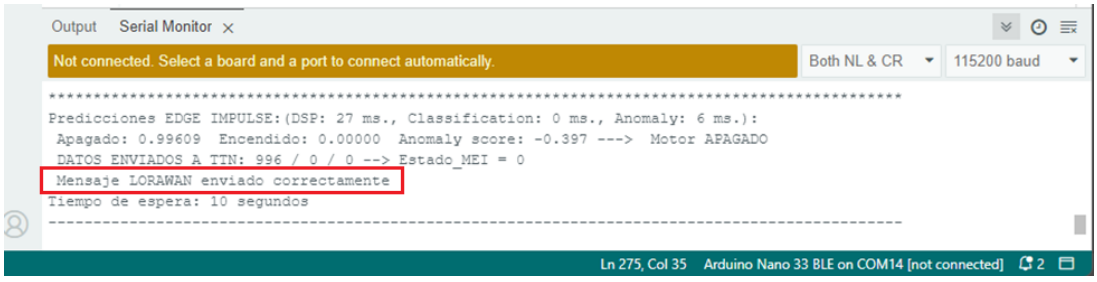


Fuente: El autor

Como resultado del funcionamiento normal del prototipo, Figura 45, se obtiene en el monitor serial un mensaje con las predicciones entregadas por el algoritmo de Edge Impulse, los valores de clasificación para cada estado del motor y confirmación del mensaje enviado a TTN y el tiempo de espera configurado para nueva lectura de datos.

Figura 45.

Mensaje en monitor serial indicando funcionamiento correcto al enviar datos por LoRaWAN



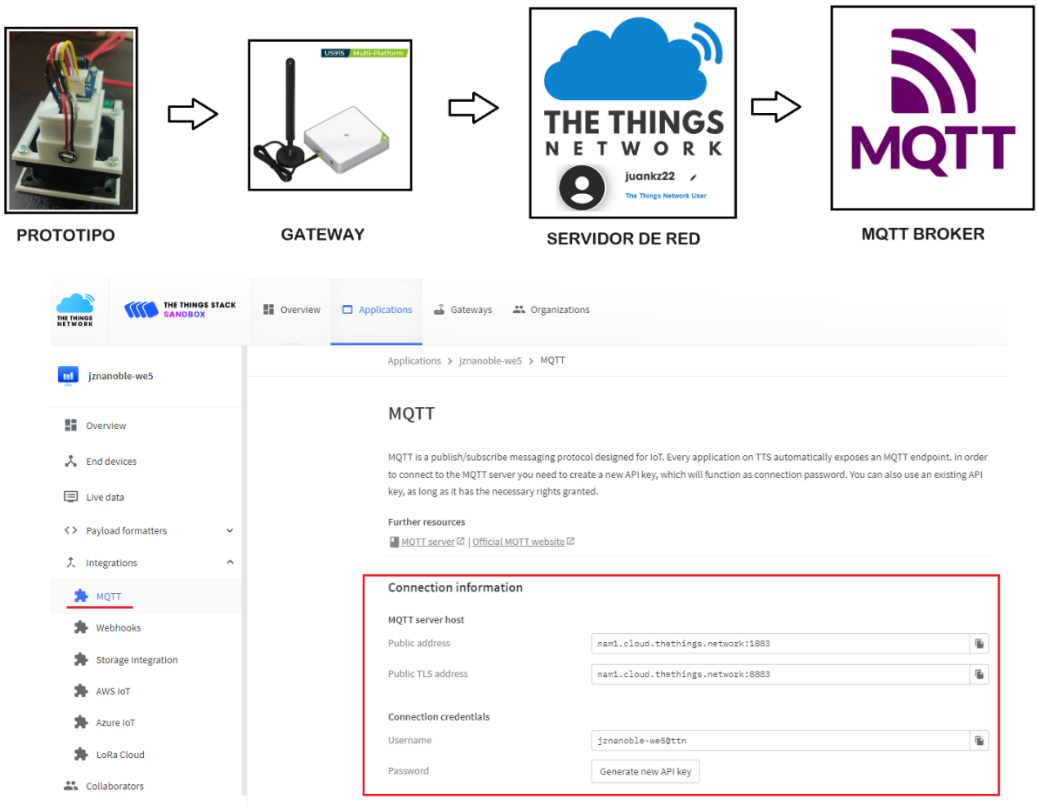
Fuente: El autor

5.5.Etapa IV: Implementación del sistema integral de almacenamiento y visualización conocido como MING

5.5.1. Integración de MQTT en TTN para envío de datos a Stack MING

Conforme el esquema presentado en la Figura 46, una vez que el prototipo envía el dato por LoRaWAN al Gateway y la información llega al servidor The Things Network (TTN), todo el proceso desarrollado en las etapas previas. Lo primero a realizar es identificar credenciales y direcciones IP del servidor MQTT que tiene habilitado TTN para aplicaciones IoT. Para conectarse al servidor MQTT, necesitamos crear una nueva clave API, que funcionará como contraseña de conexión, así como conocer la dirección del servidor nam1.cloud.thethings.network:1883 y el nombre de usuario jznanoble-we5@ttn.

Figura 46.
Representación de la secuencia a seguir para registro de datos.

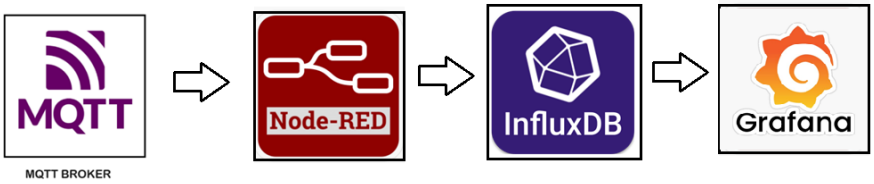


Fuente: El autor

5.5.2. Registro de datos MQTT en Node-RED

Los datos generados por el servidor TTN con MQTT para poder ser visualizados en Grafana previamente van a ser recibidos y organizados en Node-RED. Estos datos se envían a una base de datos en InfluxDB y finalmente se podrán visualizar en Grafana, este orden se requiere seguir para llegar desde el servidor TTN a Grafana tal como se aprecia en la Figura 47.

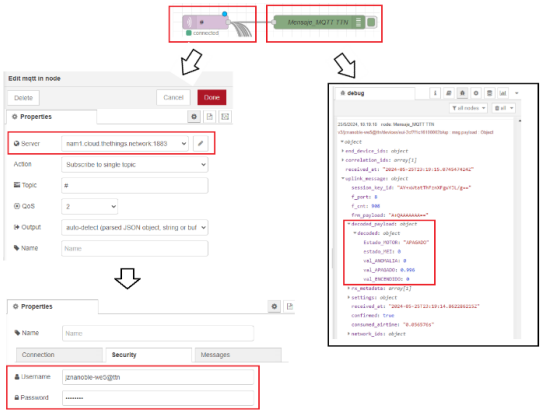
Figura 47.
Parámetros predefinidos para generar Dataset.



Fuente: El autor

En NodeRED lo primero que se realizó fue utilizar un nodo como entrada MQTT y configurarlo con los datos del servidor de TTN: la dirección IP y las credenciales para validar conectividad y se agrega un nodo DEBUG con la finalidad de verificar que los mensajes llegan. El mensaje de entrada contiene múltiple información, para este caso el objeto principal a utiliza es decoded_payload del cual obtendremos los datos independientes para enviar a InfluxDB y Grafana, el desarrollo de este paso se aprecia en la Figura 48.

Figura 48.
Prueba de recepción de datos enviados por TTN y recibidos en NodeRED.

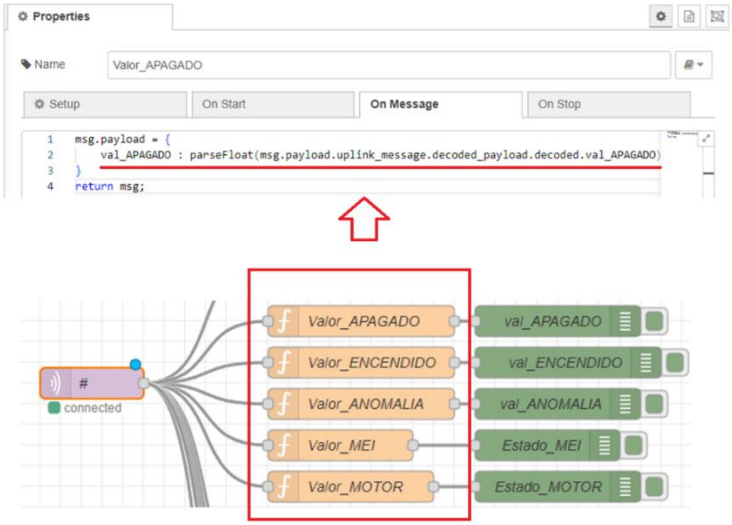


Fuente: El autor

Los datos correspondientes a las diferentes variables se obtienen de extraer la información útil, Figura 49, por medio de la línea de código descrita a continuación y que contiene el nombre de la variable y dato a obtener respectivamente. La sintaxis a seguir es:

```
val_dato: parseFloat(msg.payload.uplink_message.decoded_payload.decoded.DATO)
```

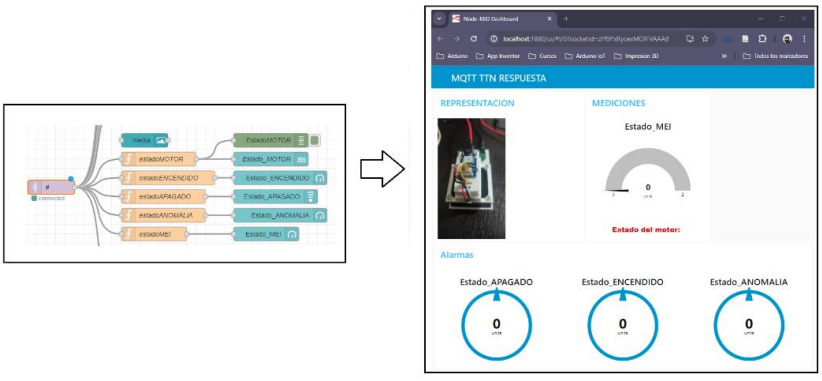
Figura 49.
Configuración en Node RED para extraer valores del mensaje recibido por MQTT.



Fuente: El autor

Se configuró tal y como se aprecia en la Figura 50 un dashboard en NodeRED con el fin de visualizar los valores recibidos en cada mensaje que llega desde TTN y es enviado por MQTT.

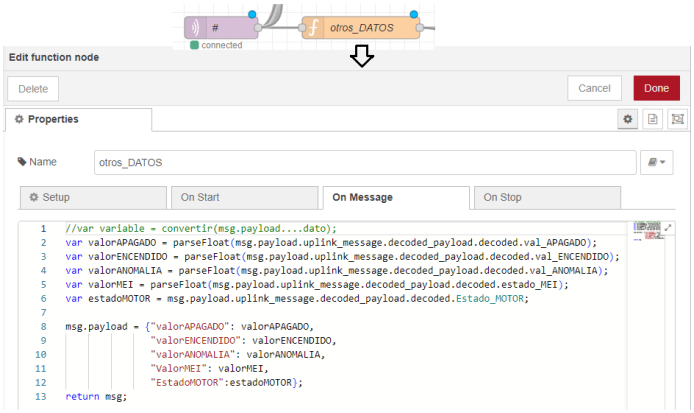
Figura 50.
Dashboard en NodeRED para visualizar datos recibidos.



Fuente: El autor

Para almacenar los datos en InfluxDB se utilizó un nodo función conectado al nodo de entrada MQTT, Figura 51, configuramos las propiedades para extraer la información del mensaje original de entrada y organizarla para que sea enviada a la base de datos agregando una etiqueta a cada valor a almacenar.

Figura 51.
Parámetros predefinidos para generar Dataset.



Fuente: El autor

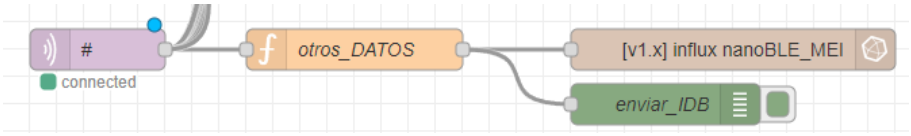
Agregamos un nodo de salida de InfluxDB como se aprecia en la Figura 52 y configuramos el servidor, puerto, le asignamos un nombre a la base de datos que vamos a crear y a la medición que queremos almacenar. En este caso, el servidor es local, la base de datos se llama datosMEI y la medición nanoBLE_MEI.

Figura 52.
Configuración en NodeRED para enviar datos a Influx.

Fuente: El autor

Para validar a nivel de debug se coloca un bloque para visualizar los datos enviados a Influx y el formato en él se realiza el envío. El flujograma resultante se puede ver en la Figura 53.

Figura 53.
Flujograma para envío de datos de NodeRED a Influx.

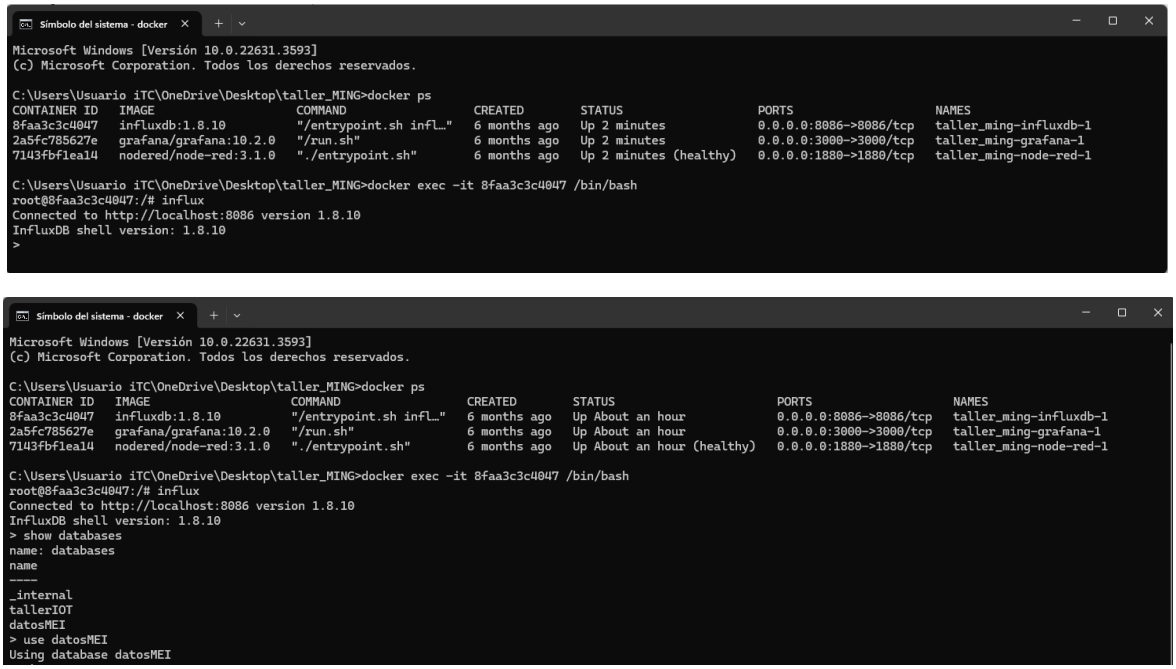


Fuente: El autor

5.5.3. Registro de datos en InfluxDB

El primer paso es acceder a Influx por medio del terminal ejecutado sobre el directorio del Docker preinstalado, eso lo podemos ver en las Figura 54 - 55. Desde el terminal creamos la base de datos de nombre datosMEI y accedemos a la información almacenada para verificar los datos.

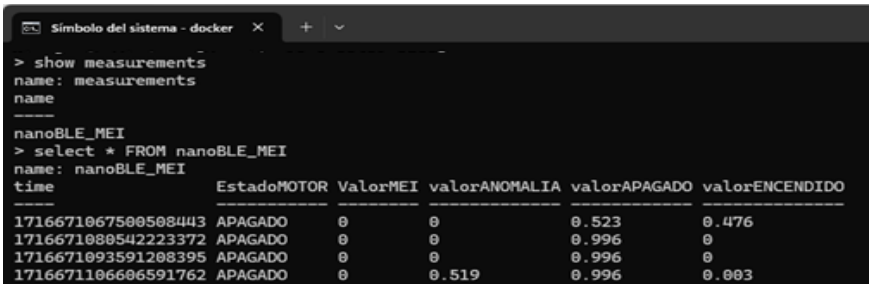
Figura 54.
Parámetros predefinidos para generar Dataset.



Fuente: El autor

Desde el terminal visualizamos las mediciones que existen en la base de datos datosMEI, que en este caso corresponde a nanoBLE_MEI que fue nombrada así en NodeRED y procedemos a visualizar las mediciones obtenidas para validar los datos almacenados y el formato que tienen los datos con sus respectivas etiquetas que serán utilizadas en Grafana para extraer y visualizar el dato esperado.

Figura 55.
Datos almacenados en Influx.



Fuente: El autor

5.5.4. Visualización de datos en Grafana

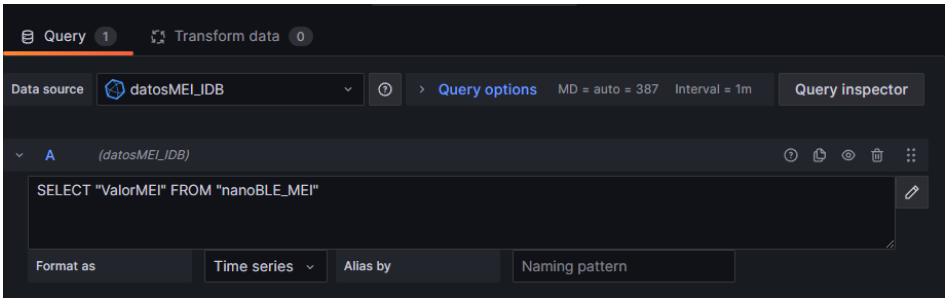
Para la visualización de los datos en Grafana, lo primero es crear un nuevo dashboard asociado a la base de datos de InfluxDB, que previamente se creó y lleva por nombre datosMEI. El primer panel consta de cuatro paneles independientes cuya función es visualizar el valor de la variable estado_MEI (puede ser 0:apagado, 1:encendido, 2:anomalía), y tres paneles tipo bar gauge que son los encargados de visualizar el valor conforme nos entrega como resultado el algoritmo de Edge Impulse programado en el Arduino Nano 33 BLE, resultado en la Figura 56.

Figura 56.
Panel en Grafana para datos del estado del motor.

Fuente: El autor

La configuración de los tres paneles es similar, para en la fuente de datos de acuerdo con la variable y dato requerido escribimos SELECT "***Variable***" FROM "***mediciones***", teniendo como variables posibles ValorMEI, valorENCENDIDO, valorAPAGADO, valorANOMALIA y como parámetro de mediciones ***nanoBLE_MEI***. Resultado se aprecia en la Figura 57.

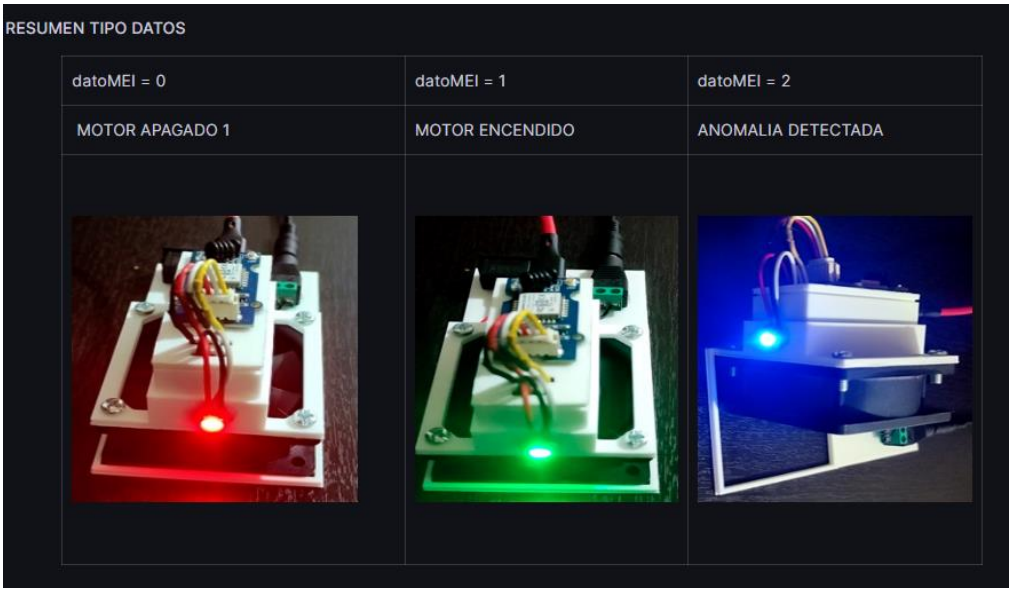
Figura 57.
Ejemplo de configuración de panel de visualización de datos.



Fuente: El autor

Se incluye un panel de tipo Texto, en el que a manera informativa se detallan ejemplos de visualización de los 3 posibles estados que el motor puede presentar y han sido considerados. Resultado se aprecia en la Figura 58.

Figura 58.
Panel informativo tipo texto colocado en dashboard en Grafana.



Fuente: El autor

En la Figura 59, se aprecia el resultado de incluir un panel de tipo Histograma y un panel de tipo Time Series, para visualizar la variación de los estados del motor conforme se ha puesto en marcha y se han realizado el registro de datos del prototipo.

Figura 59.

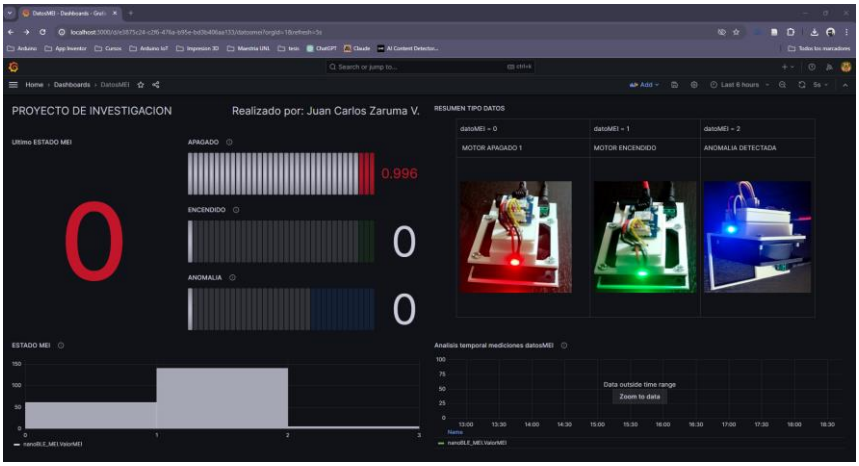
Panel tipo histograma y time series colocados en dashboard en Grafana.

Fuente: El autor

El dashboard realizado en Grafana finalizado se puede apreciar en la Figura 60, podemos distinguir cada panel descrito anteriormente y el acceso se lo hace mediante red local habilitada sobre el docker mediante la IP http://localhost:3000/

Figura 60.

Dashboard final realizada en Grafana.



Fuente: El autor

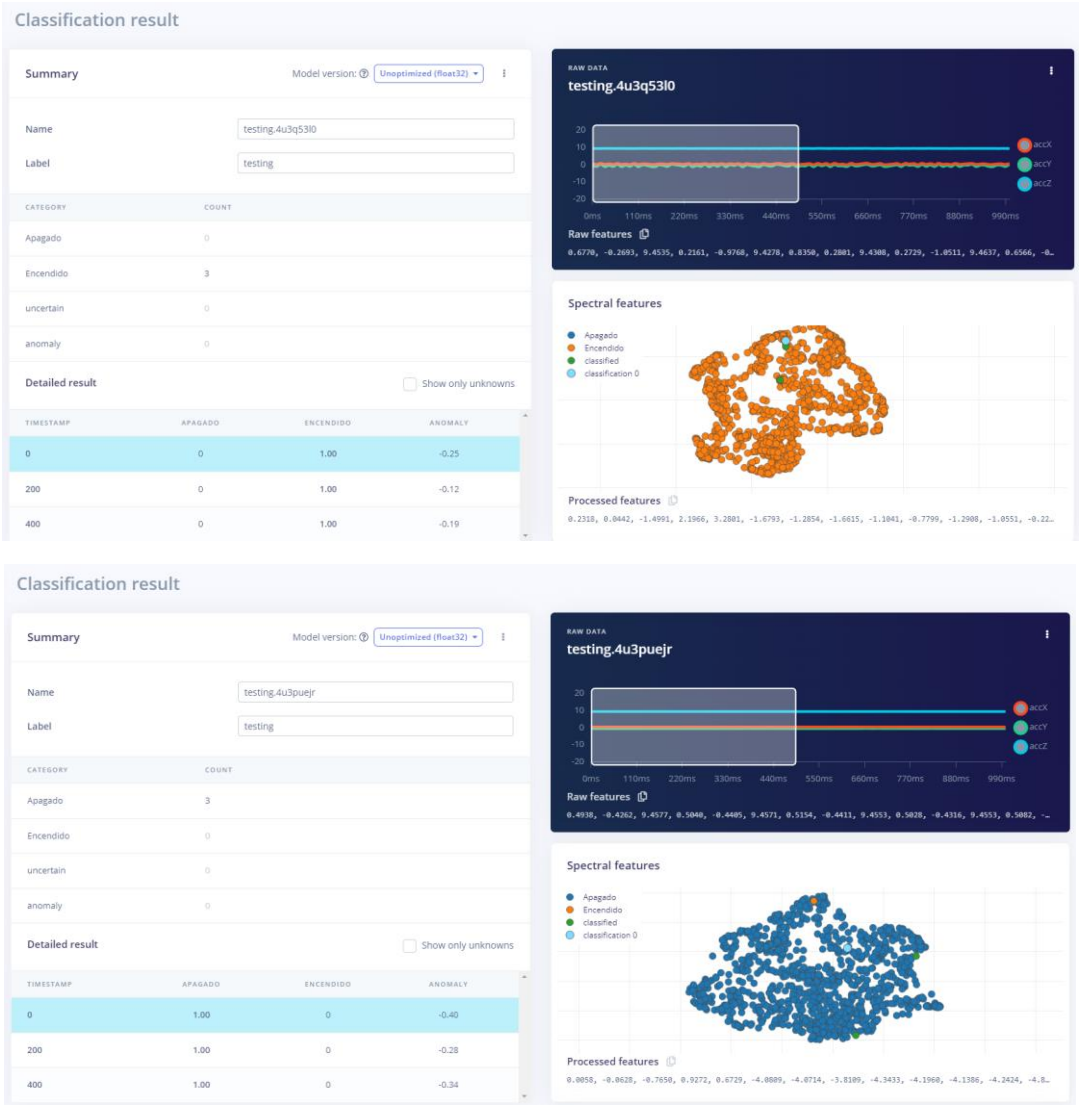
6. Resultados

Los resultados del funcionamiento del prototipo se adjuntan a continuación:

- El modelo de aprendizaje resultante en la plataforma Edge Impulse al realizar un test de prueba de clasificación con un tiempo de muestra de 1000 ms a una frecuencia de 100 Hz, identifica con certeza entre el estado encendido y apagado. Resultado se aprecia en la Figura 61.

Figura 61.

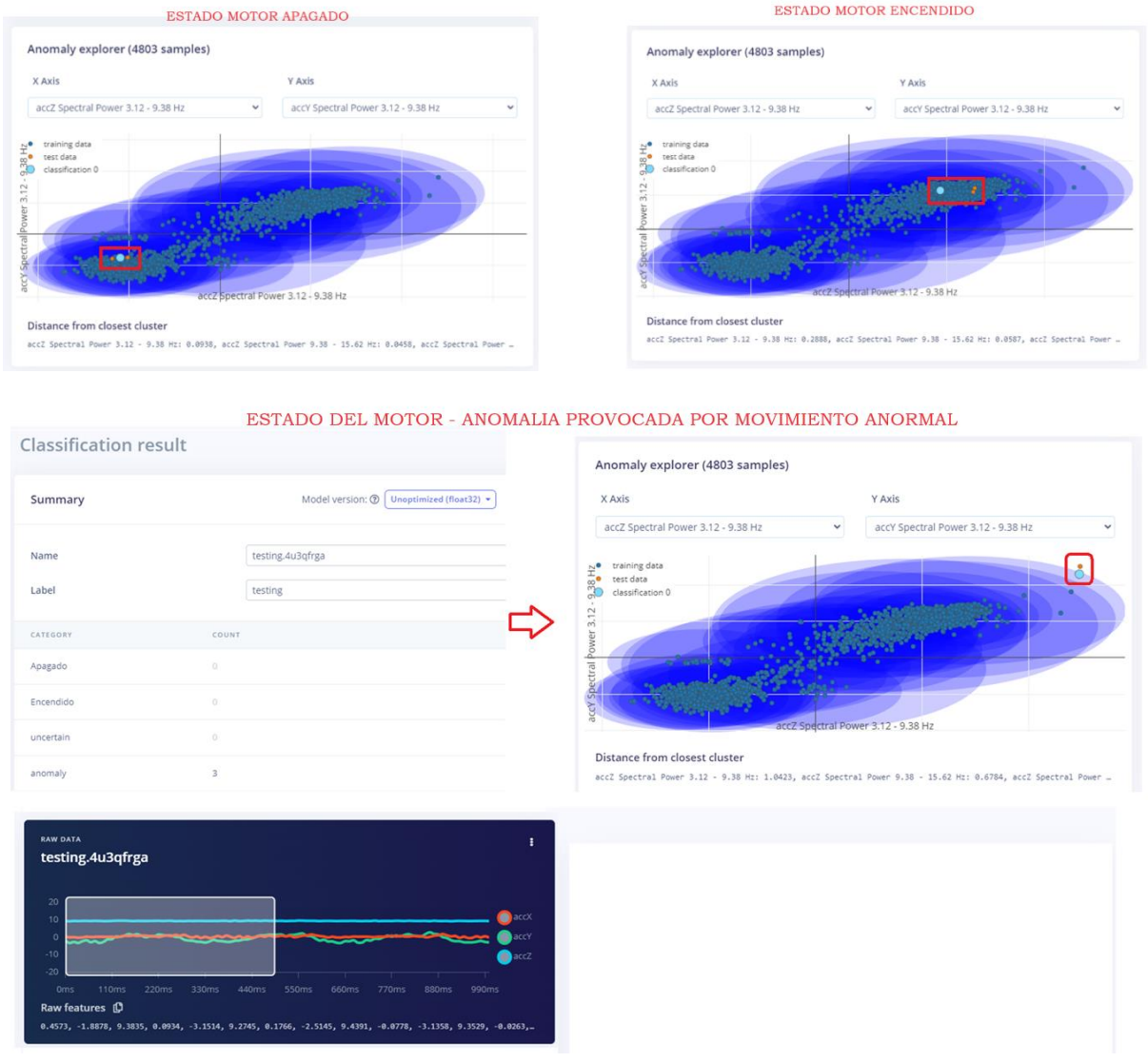
Pruebas de clasificación de estados en plataforma Edge Impulse.



Fuente: El autor

- La detección de anomalías configurada para este caso responde a eventos anormales en el funcionamiento del motor provocados al ejecutar movimientos sobre el eje Y y eje Z para los ejes de referencia que tiene el sensor LSM9DS1 incorporado en el Arduino nano 33 BLE. En la Figura 62, se coloca un ejemplo de las pruebas realizadas y la respuesta que ofrece el modelo de entrenamiento de Edge Impulse al clasificar estados y al detectar alguna anomalía.

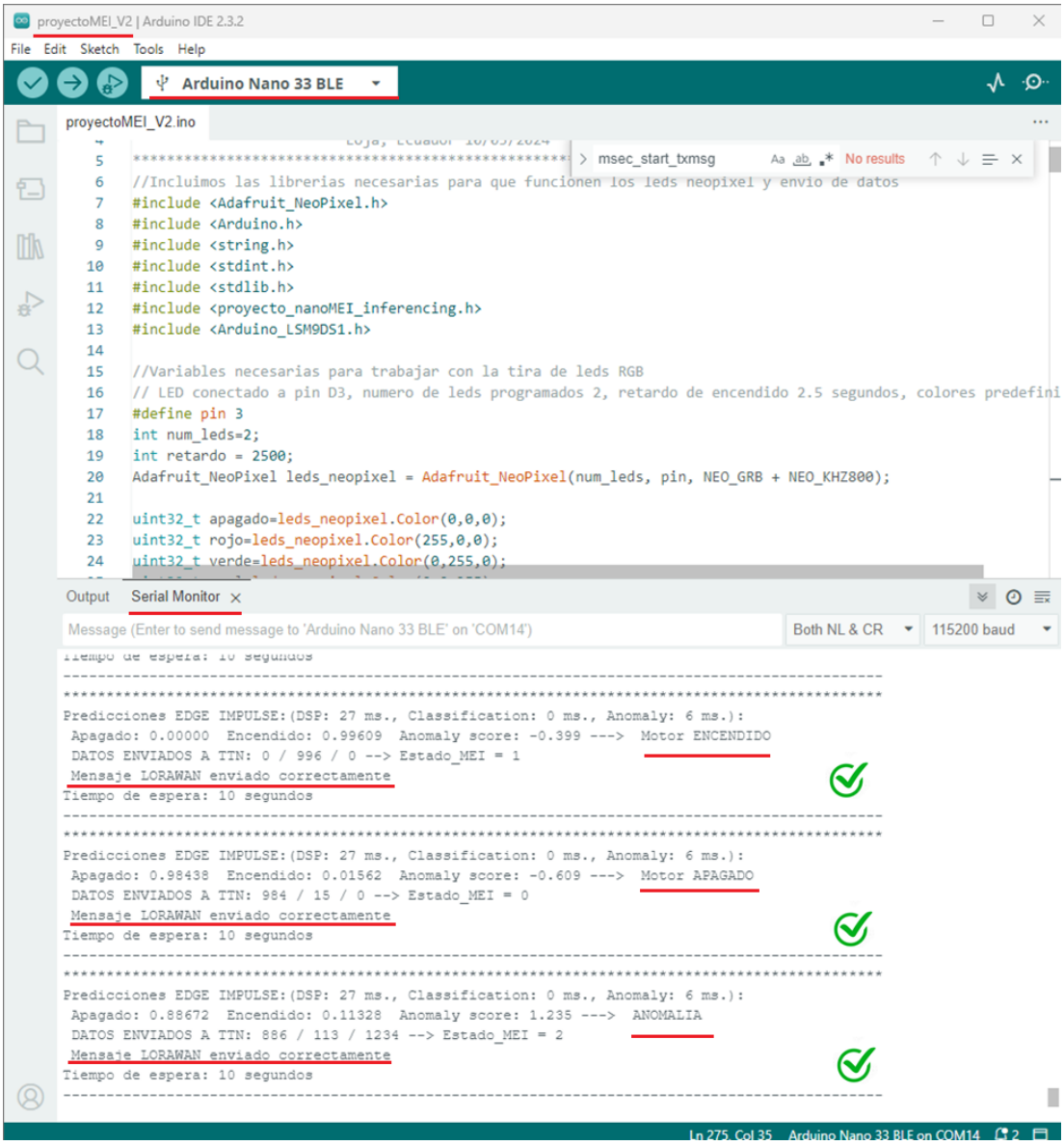
Figura 62.
Detección de anomalías en pruebas realizadas en Edge Impulse.



Fuente: El autor

- En la figura 63, se adjunta una captura del código final cargado en el Arduino Nano 33 BLE y al ejecutar el mismo en el monitor serial se aprecia que se realiza la clasificación de estados, detección de anomalías y confirmación de envío de mensaje a TTN en caso de que no exista problema de comunicación con Gateway LoRAWAN o con el módulo Wio – E5.

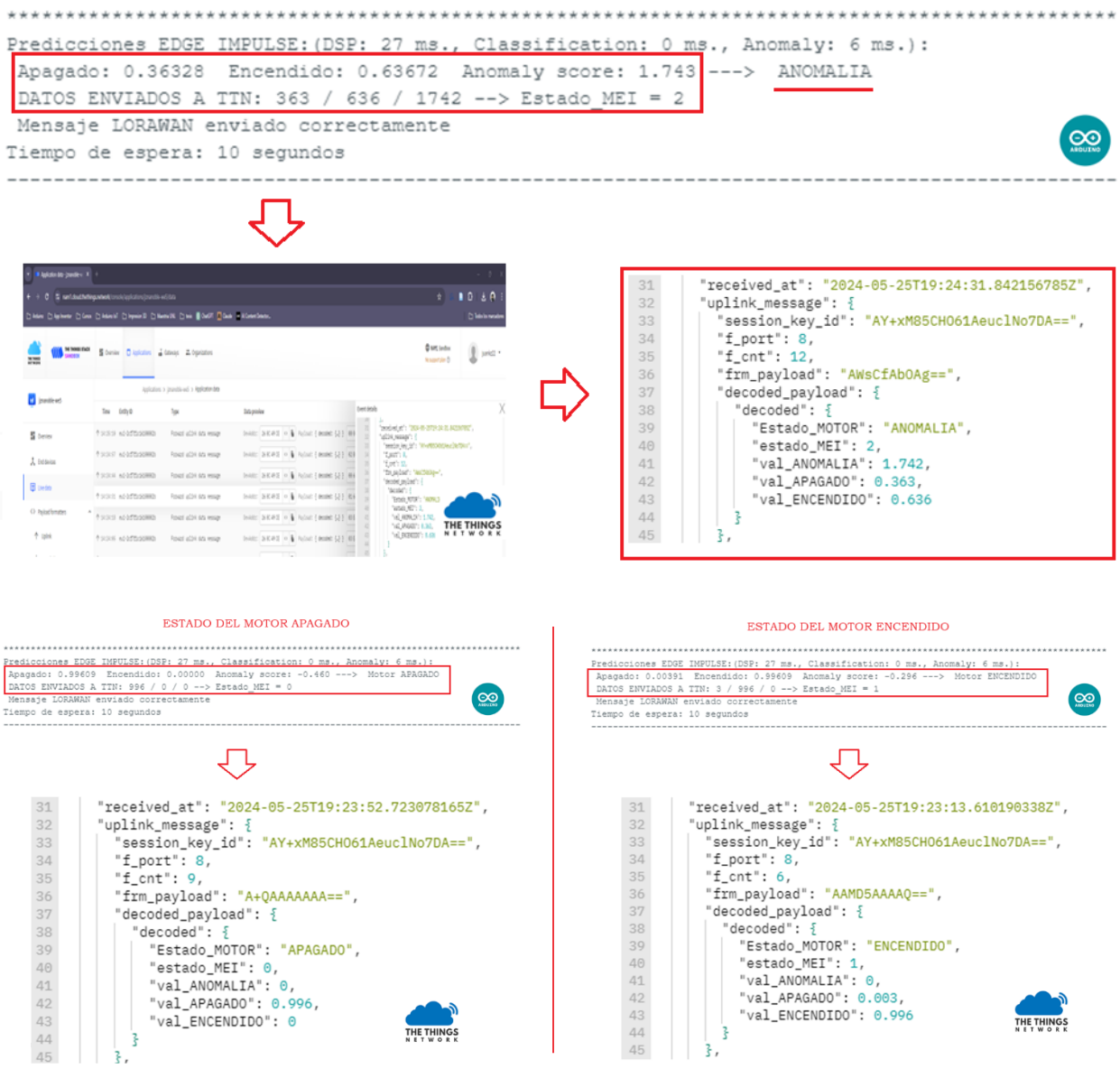
Figura 63.
Pruebas de funcionamiento en monitor serial Arduino IDE para proyectoMEI_V2.ino.



Fuente: El autor

- En la Figura 64, se adjunta se adjunta un ejemplo de los resultados obtenidos para el envío de datos desde el Arduino hacia la plataforma The Things Network, el mensaje es decodificado para obtener la carga útil deseada y enviar esa información por MQTT hacia NodeRED.

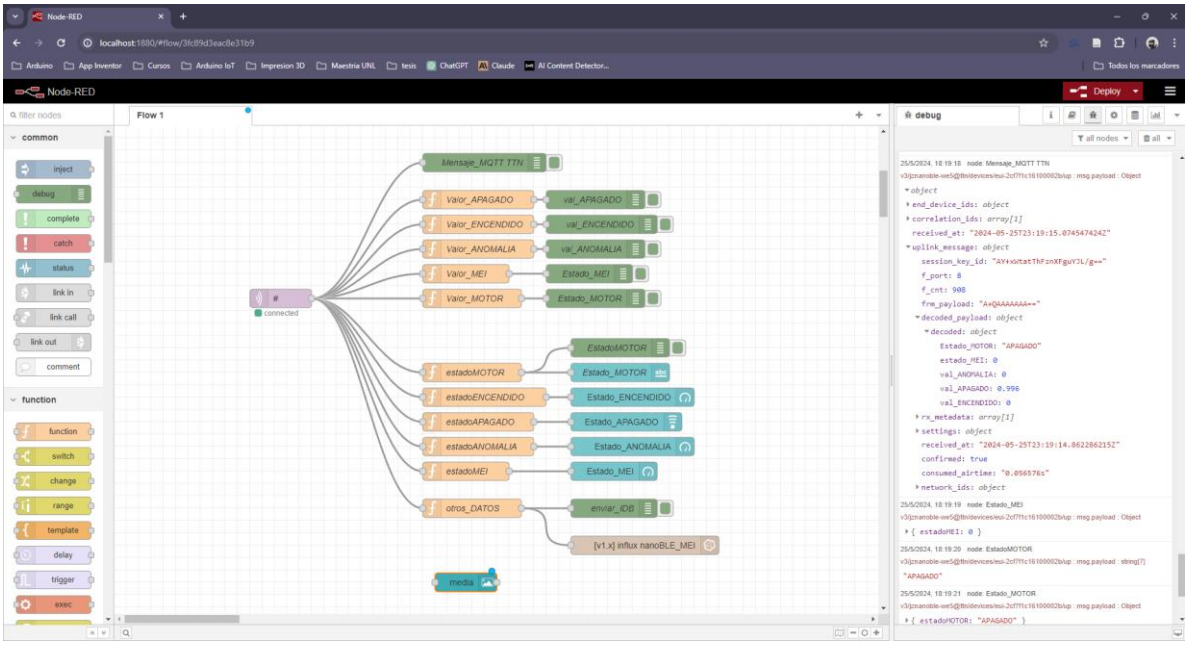
Figura 64.
Resultados de envío de datos de Arduino a TTN.

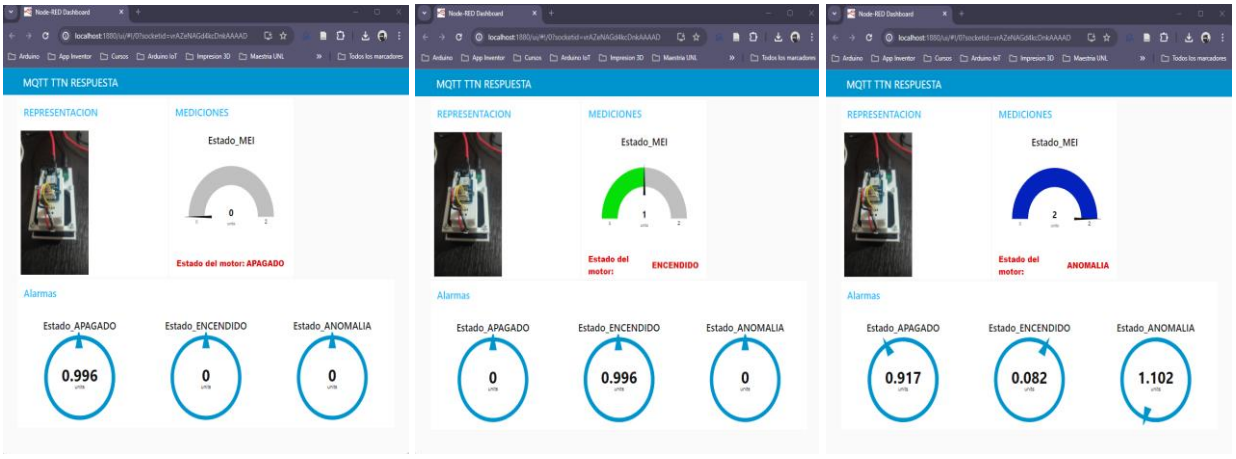


Fuente: El autor

- En la figura 65 se coloca una captura de NodeRED del flujograma final con el debug activo para validar que llega la información de TTN se reorganiza y se envía a Influx, además se colocan imágenes de los cambios de estado representados en el dashboard de NodeRED.

Figura 65.
Flujograma y dashboard finales realizados en NodeRED.

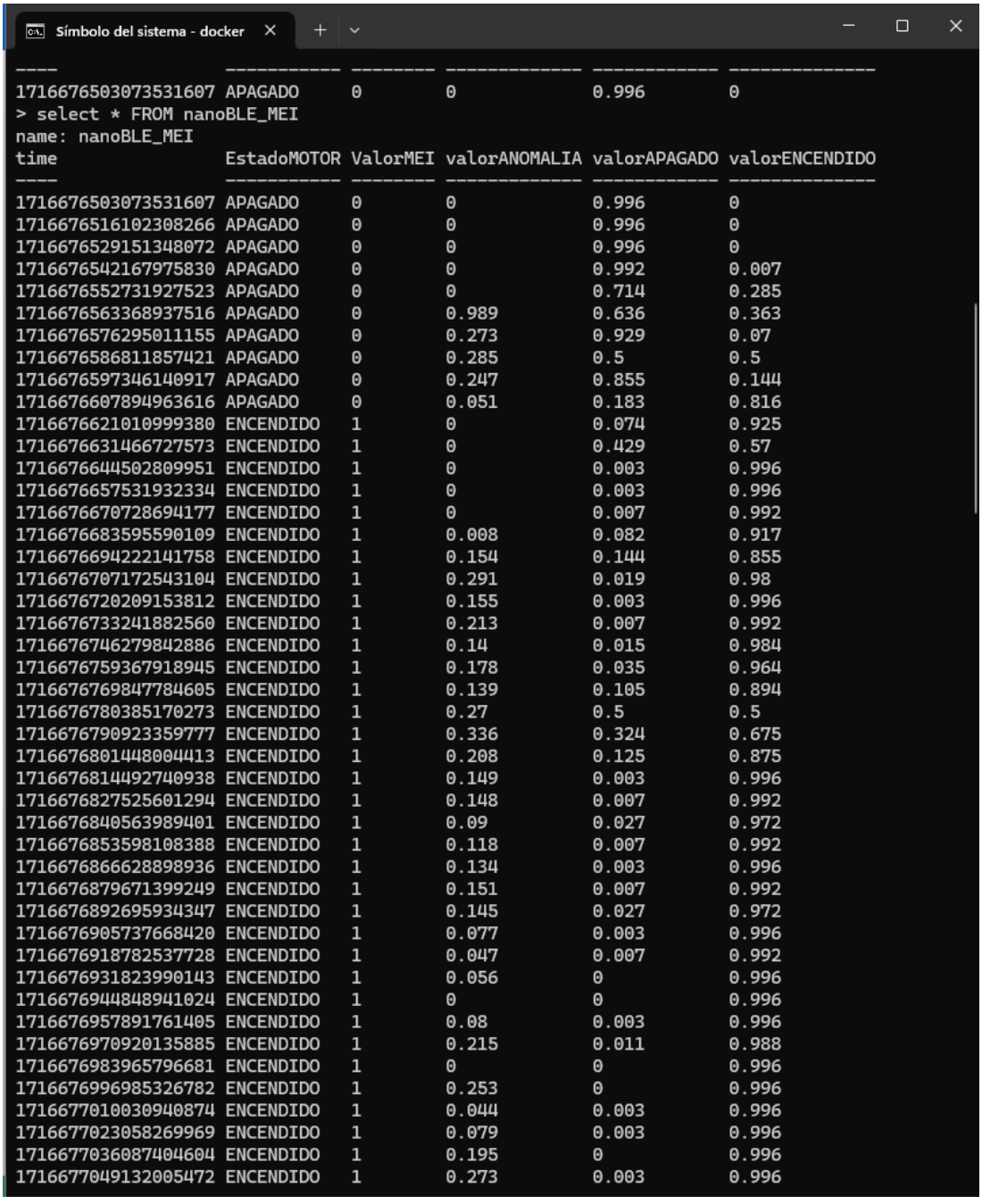




Fuente: El autor

- En la figura 66, se coloca una captura de las mediciones almacenadas en la base de datos nanoBLE_MEI y el tipo de dato que se coloca para cada registro almacenado.

Figura 66.
Captura de resultante de mediciones almacenadas en Influx.



Fuente: El autor

- En la figura 67, se aprecia el resultado final de la visualización de estados del motor en Grafana cuando el motor se encuentra Apagado, cuando está Encendido y al momento de detectar posibles Anomalías.

Figura 67.
Dashboard final realizada en Grafana.

Fuente: El autor

7. Discusión

En términos generales, el prototipo para la detección de anomalías se realizó con éxito. Sin embargo, al ser un modelo didáctico, presenta ciertas limitaciones inherentes al prototipo. La principal limitación es que solo cuenta con dos estados del motor: Apagado y Encendido. En el estado Apagado, el motor no está alimentado por la fuente, mientras que, en el estado Encendido, el motor se conecta directamente a la fuente de voltaje, proporcionando un valor constante de 5V y 1A. No es posible variar la velocidad, las revoluciones por minuto ni la dirección de giro.

Sin embargo, el sensor interno correspondiente al acelerómetro del Arduino Nano 33 BLE Sense capta señales que se distinguen con relativa facilidad cuando el motor está encendido o apagado. Esto permite generar un dataset para el modelo de aprendizaje obtenido en la plataforma de Edge Impulse, el cual presenta una precisión del 100% en la clasificación de estados debido a la mínima cantidad de estados y su clara definición.

En cuanto al registro y envío de datos por LoRaWAN, resulta complicado encontrar dispositivos que soporten esta tecnología en tiendas de electrónica locales. Esto implica que, para replicar el experimento o realizar uno similar, es necesario importar los componentes. La plataforma The Things Network facilita el despliegue y gestión de redes IoT de largo alcance y bajo consumo, promoviendo una infraestructura abierta y colaborativa que soporta la creciente demanda de soluciones conectadas en diversos sectores industriales y comunitarios.

Realizar este proyecto fue un desafío profesional debido a sus múltiples etapas. Estas incluyeron la generación de datos, la creación de un prototipo físico, el desarrollo de un modelo de aprendizaje con Edge Impulse, y la habilitación de la comunicación LoRaWAN entre un nodo, el Gateway y la plataforma de TTN. Además, gracias al uso del protocolo MQTT, se integró un Docker con NodeRED, InfluxDB y Grafana para desarrollar una interfaz de monitoreo remoto. Esta interfaz permite visualizar el estado actual del motor y detectar anomalías generadas por las vibraciones no esperadas del mismo, convirtiéndose en una herramienta final que al replicarla en un entorno industrial permita aplicar mantenimiento preventivo con bases de industria 4.0.

8. Conclusiones

Para esta investigación, se seleccionó un motor de ventilador como elemento principal, ya que sus vibraciones al estar en funcionamiento permitieron detectar y predecir su estado. Esto es aplicable en la industria 4.0, ya que el dispositivo se puede replicar y adaptar a motores semiindustriales para la detección de anomalías en su funcionamiento mediante las vibraciones emitidas. El LED indicador facilita al operador comprender de manera sencilla la respuesta del prototipo frente a determinados eventos contemplados en el código.

El prototipo desarrollado en este proyecto de investigación facilita la comprensión de las distintas etapas del uso de TinyML y ofrece un medio para validar la detección de anomalías en un motor. Esto resulta útil para identificar movimientos anómalos, tanto cuando el motor está encendido como cuando está apagado. Se evaluó con éxito la detección de anomalías utilizando el algoritmo generado por la plataforma Edge Impulse y la clasificación en los estados de Encendido y Apagado. El prototipo respondió de manera eficiente durante las pruebas realizadas.

La comunicación a través de LoRaWAN ofrece varias ventajas para este proyecto, como el uso de un espectro libre, la capacidad de alcanzar distancias considerablemente grandes y la baja latencia en la transmisión periódica de información. El uso de la red global de The Things Network (TTN) ofrece la infraestructura necesaria para que los dispositivos se comuniquen entre sí y con aplicaciones en Internet mediante la tecnología LoRaWAN.

El uso de la tecnología de contenedores ha revolucionado la forma en que se desarrollan, despliegan y gestionan las aplicaciones, proporcionando una solución eficiente y escalable para el análisis y visualización de resultados. Como parte complementaria del proyecto, se verificó la integración de diversos componentes de código abierto. Esto incluye el envío de información por MQTT desde un servidor en The Things Network (TTN) hacia un cliente en Node-RED. Los datos se almacenaron en InfluxDB y se visualizaron a través de una interfaz diseñada en Grafana. Este flujo facilita el monitoreo del motor y la visualización de sus posibles estados de manera eficaz.

9. Recomendaciones

Para garantizar el óptimo funcionamiento del prototipo se recomienda en caso de replicar el experimento descrito a lo largo de este proyecto, trabajar en condiciones similares como por ejemplo utilizar el mismo motor o con características similares, el modelo de aprendizaje fue entrenado en base al Arduino Nano 33 BLE Sense versión 1 por lo que se recomienda utilizar la misma tarjeta electrónica como elemento base. En caso, que se pretenda trabajar con una versión portátil se recomienda realizar mejoras al prototipo como incorporar una batería y cargador para que funcione sin necesidad de tener conectado un adaptador.

Para el caso de la comunicación con LoRaWAN, debemos asegurarnos que el Gateway tenga conexión permanente al servicio de Internet para enviar información al servidor TTN. Debemos tener conocimiento previo en comunicación LoRa y las frecuencias en las que puede operar este tipo de comunicación de acuerdo con la zona geográfica en la se realicen las pruebas. Ambos dispositivos tanto el prototipo como el Gateway deben contar con tecnologías compatibles entre ellos y debemos considerar el alcance para que puedan intercambiar información.

Se recomienda tener conocimientos previos sobre programación en Arduino IDE, uso de tarjetas electrónicas como Arduino, conocimiento en diseño de prototipado electrónico en general. Para la parte complementaria del proyecto que hace énfasis en la interfaz de monitoreo, se recomienda tener conocimiento en lo que comprende el stack MING, sobre uso de flujogramas en NodeRED, almacenar datos en InfluxDB y realizar un dashboard en Grafana.

10. Bibliografía

3 usos de tinyML en el borde. (s. f.). DigiKey. Recuperado 2 de diciembre de 2023, de https://www.digikey.com/es/blog/3-uses-for-tinyml-at-the-edge

About Edge Impulse. (s. f.). Recuperado 14 de diciembre de 2023, de https://edgeimpulse.com/about

Arduino Tiny Machine Learning Kit. (s. f.). Arduino Online Shop. Recuperado 9 de diciembre de 2023, de https://store-usa.arduino.cc/products/arduino-tiny-machine-learning-kit

Edge Impulse Brings TinyML to Millions of Arduino Developers. (2020, junio 26). https://www.edgeimpulse.com/blog/edge-impulse-brings-ml-to-arduino

¿En qué consiste el mantenimiento 4.0 y cuáles son sus beneficios? (s. f.). Recuperado 14 de diciembre de 2023, de https://es.linkedin.com/pulse/en-qu%C3%A9-consiste-el-mantenimiento-40-y-cu%C3%A1les-son-sus-beneficios-sas

Exploración a ecosistema LoRaWAN. (s. f.). Exploración a ecosistema LoRaWAN. Recuperado 23 de junio de 2024, de https://www.gotoiot.com/pages/articles/lorawan_exploration/content.html

LoRaWAN y su aportación a las tecnologías IIoT | INCIBE-CERT | INCIBE. (s. f.). Recuperado 7 de diciembre de 2023, de https://www.incibe.es/incibe-cert/blog/lorawan-y-su-aportacion-las-tecnologias-iiot

mpous. (2023, mayo 15). *Use the MING stack to accelerate your IoT application development*. Balena Blog. https://blog.balena.io/ming-stack-mqtt-influxdb-nodered-grafana-balena/

Network, T. T. (s. f.). *The Things Network*. The Things Network. Recuperado 15 de diciembre de 2023, de https://www.thethingsnetwork.org/map

Qué es LoRa, cómo funciona y características principales—Venco Electrónica. (s. f.). Recuperado 9 de diciembre de 2023, de https://www.vencoel.com/que-es-lora-como-funciona-y-caracteristicas-principales/

¿Qué es TinyML? (2022, marzo 2). *TodoMaker*. https://todomaker.com/blog/que-es-tinyml/

Rodriguez, A. (2022, marzo 23). *La implantación de los sistemas TinyML*. www.diarioelectronicohoy.com. https://www.diarioelectronicohoy.com/la-implantacion-de-los-sistemas-tinyml/

Tecnología LoRA y LoRAWAN - Catsensors. (s. f.). Recuperado 23 de junio de 2024, de https://www.catsensors.com/es/lorawan/tecnologia-lora-y-lorawan

TinyML. (s. f.). *Alexander Thamm GmbH*. Recuperado 2 de diciembre de 2023, de https://www.alexanderthamm.com/es/data-science-glossary/tinyml/

11. Anexos

Anexo 1. Algoritmo realizado para decodificar datos en plataforma TTN.

```
// Codigo para decodificar envío de datos proyecto MEI

// REFERENCIA: https://www.thethingsindustries.com/docs/integrations/payload-
formatters/javascript/

function decodeUplink(input) {

	var data = {};

	bindata = data2bits(input.bytes);

	val_APAGADO = precisionRound(bitShift(16)*0.001, 3);
	val_ENCENDIDO = precisionRound(bitShift(16)*0.001, 3);
	val_ANOMALIA = precisionRound(bitShift(16)*0.001, 3);
	estado_MEI = bitShift(8);

	switch(estado_MEI) {
		case 0	:				// Motor DC APAGADO
			Estado_MOTOR = "APAGADO";
			break;
		case 1	:				// Motor DC ENCENDIDO
			Estado_MOTOR = "ENCENDIDO";
			break;
		case 2	:				// Deteccion ANOMALIA
			Estado_MOTOR = "ANOMALIA";
			break;
	}

	data.decoded = {
		"val_APAGADO": val_APAGADO,
		"val_ENCENDIDO": val_ENCENDIDO,
		"val_ANOMALIA": val_ANOMALIA,
		"estado_MEI": estado_MEI,
		"Estado_MOTOR": Estado_MOTOR
	};

	return {
		data: data,
		warnings: [],
		errors: []
	};
}
```

Anexo 2. Hoja de datos técnicos de componentes electrónicos utilizados

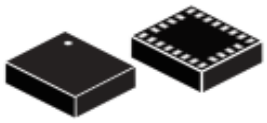
Acelerómetro de 3 Ejes XYZ LSM9DS1. Referencia:
https://www.digikey.de/htmldatasheets/production/1639232/0/0/1/lsm9ds1-datasheet.html

life.augmented

LSM9DS1

iNEMO inertial module: 3D accelerometer, 3D gyroscope, 3D magnetometer

Datasheet - production data



LGA-24L (3.5x3x1.0 mm)

Applications

- Indoor navigation
- Smart user interfaces
- Advanced gesture recognition
- Gaming and virtual reality input devices
- Display/map orientation and browsing

Features

- 3 acceleration channels, 3 angular rate channels, 3 magnetic field channels
- ±2/±4/±8/±16 *g* linear acceleration full scale
- ±4/±8/±12/±16 gauss magnetic full scale
- ±245/±500/±2000 dps angular rate full scale
- 16-bit data output
- SPI / I^2C serial interfaces
- Analog supply voltage 1.9 V to 3.6 V
- "Always-on" eco power mode down to 1.9 mA
- Programmable interrupt generators
- Embedded temperature sensor
- Embedded FIFO
- Position and motion detection functions
- Click/double-click recognition
- Intelligent power saving for handheld devices
- ECOPACK®, RoHS and "Green" compliant

Description

The LSM9DS1 is a system-in-package featuring a 3D digital linear acceleration sensor, a 3D digital angular rate sensor, and a 3D digital magnetic sensor.

The LSM9DS1 has a linear acceleration full scale of ±2*g*/±4*g*/±8/±16 *g*, a magnetic field full scale of ±4/±8/±12/±16 gauss and an angular rate of ±245/±500/±2000 dps.

The LSM9DS1 includes an I^2C serial bus interface supporting standard and fast mode (100 kHz and 400 kHz) and an SPI serial standard interface.

Magnetic, accelerometer and gyroscope sensing can be enabled or set in power-down mode separately for smart power management.

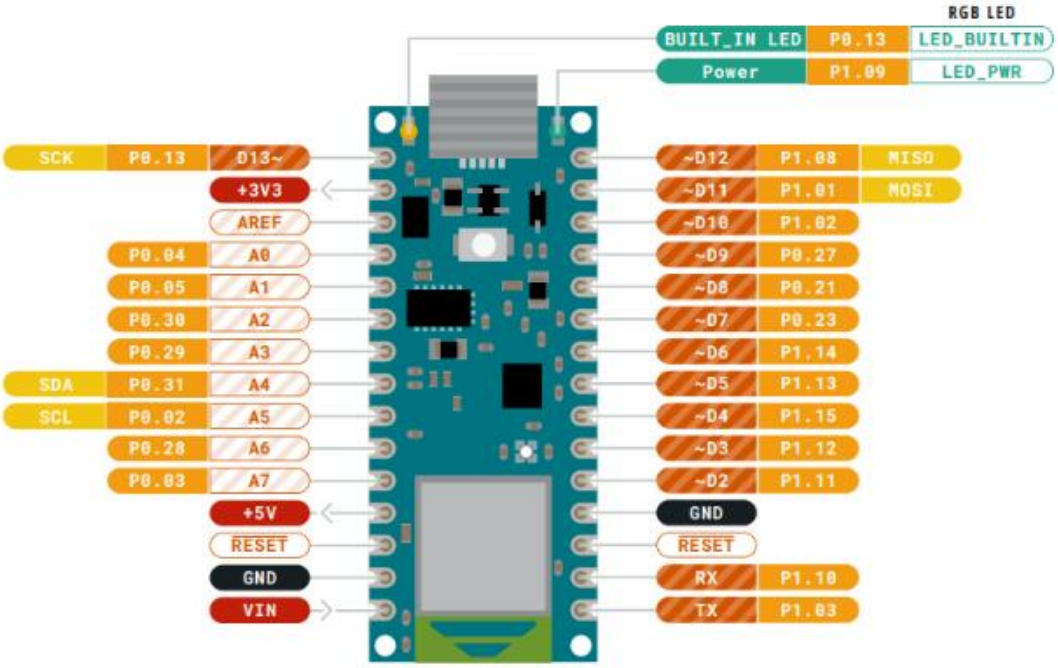
The LSM9DS1 is available in a plastic land grid array package (LGA) and it is guaranteed to operate over an extended temperature range from -40 °C to +85 °C.

Table 1. Device summary

Part number	Temperature range [°C]	Package	Packing
LSM9DS1	-40 to +85	LGA-24L	Tray
LSM9DS1TR	-40 to +85	LGA-24L	Tape and reel

Arduino Nano 33 BLE Sense V1.0 Pinout. Referencia:
https://docs.arduino.cc/resources/datasheets/ABX00031-datasheet.pdf

4 Connector Pinouts



Pinout

4.1 USB

Pin	Function	Type	Description
1	VUSB	Power	Power Supply Input. If board is powered via VUSB from header this is an Output **(1)**
2	D-	Differential	USB differential data -
3	D+	Differential	USB differential data +
4	ID	Analog	Selects Host/Device functionality
5	GND	Power	Power Ground

Arduino Nano 33 BLE Sense V1.0 Pinout. Referencia:
https://docs.arduino.cc/resources/datasheets/ABX00031-datasheet.pdf

4.2 Headers

The board exposes two 15 pin connectors which can either be assembled with pin headers or soldered through castellated vias.

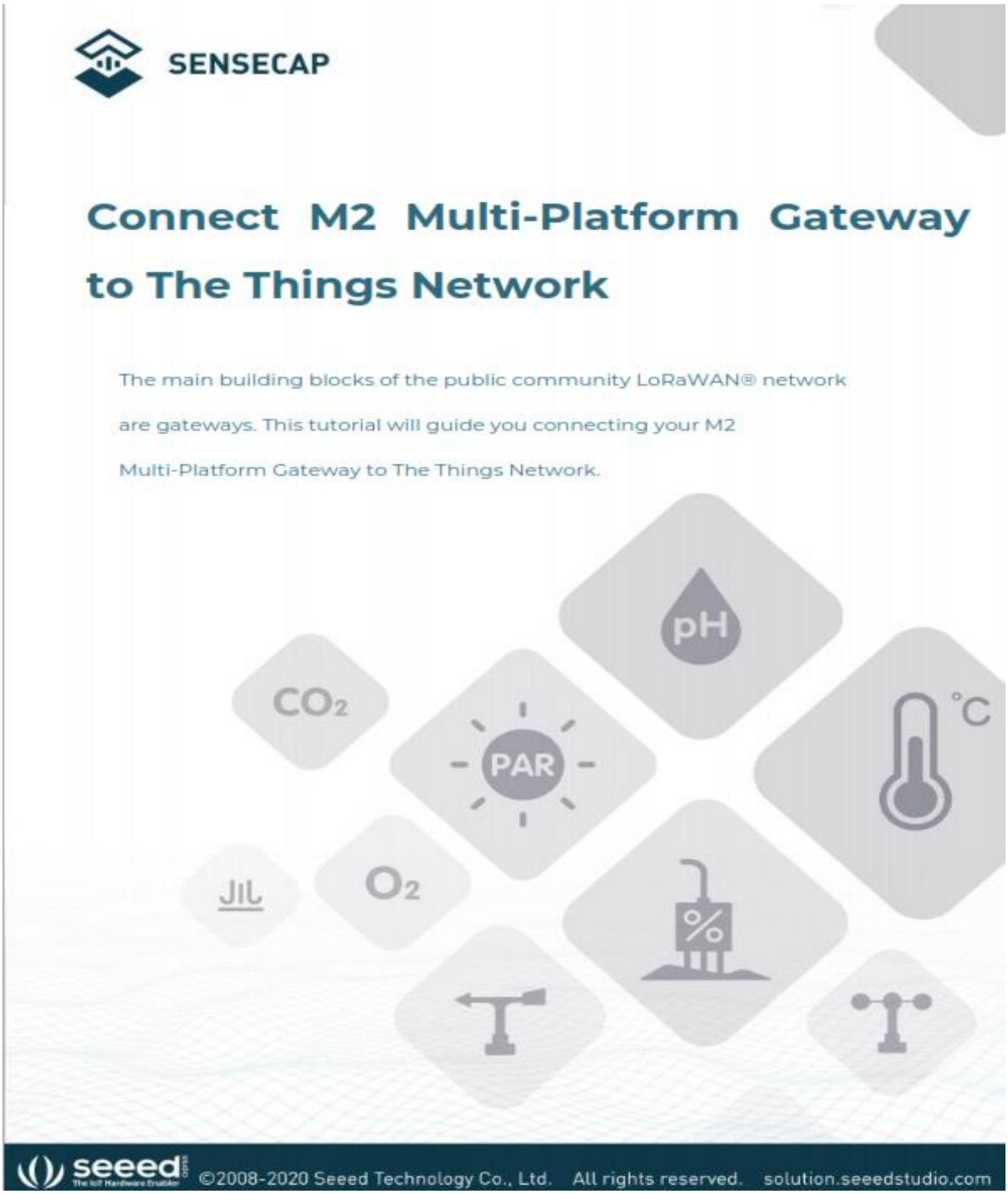
Pin	Function	Type	Description
1	D13	Digital	GPIO
2	+3V3	Power Out	Internally generated power output to external devices
3	AREF	Analog	Analog Reference; can be used as GPIO
4	A0/DAC0	Analog	ADC in/DAC out; can be used as GPIO
5	A1	Analog	ADC in; can be used as GPIO
6	A2	Analog	ADC in; can be used as GPIO
7	A3	Analog	ADC in; can be used as GPIO
8	A4/SDA	Analog	ADC in; I2C SDA; Can be used as GPIO **(1)**
9	A5/SCL	Analog	ADC in; I2C SCL; Can be used as GPIO **(1)**
10	A6	Analog	ADC in; can be used as GPIO
11	A7	Analog	ADC in; can be used as GPIO
12	VUSB	Power In/Out	Normally NC; can be connected to VUSB pin of the USB connector by shorting a jumper
13	RST	Digital In	Active low reset input (duplicate of pin 18)
14	GND	Power	Power Ground
15	VIN	Power In	Vin Power input
16	TX	Digital	USART TX; can be used as GPIO
17	RX	Digital	USART RX; can be used as GPIO
18	RST	Digital	Active low reset input (duplicate of pin 13)
19	GND	Power	Power Ground
20	D2	Digital	GPIO
21	D3/PWM	Digital	GPIO; can be used as PWM
22	D4	Digital	GPIO
23	D5/PWM	Digital	GPIO; can be used as PWM
24	D6/PWM	Digital	GPIO, can be used as PWM
25	D7	Digital	GPIO
26	D8	Digital	GPIO
27	D9/PWM	Digital	GPIO; can be used as PWM
28	D10/PWM	Digital	GPIO; can be used as PWM
29	D11/MOSI	Digital	SPI MOSI; can be used as GPIO
30	D12/MISO	Digital	SPI MISO; can be used as GPIO

4.3 Debug

On the bottom side of the board, under the communication module, debug signals are arranged as 3x2 test pads with 100 mil pitch with pin 4 removed. Pin 1 is depicted in Figure 3 – Connector Positions

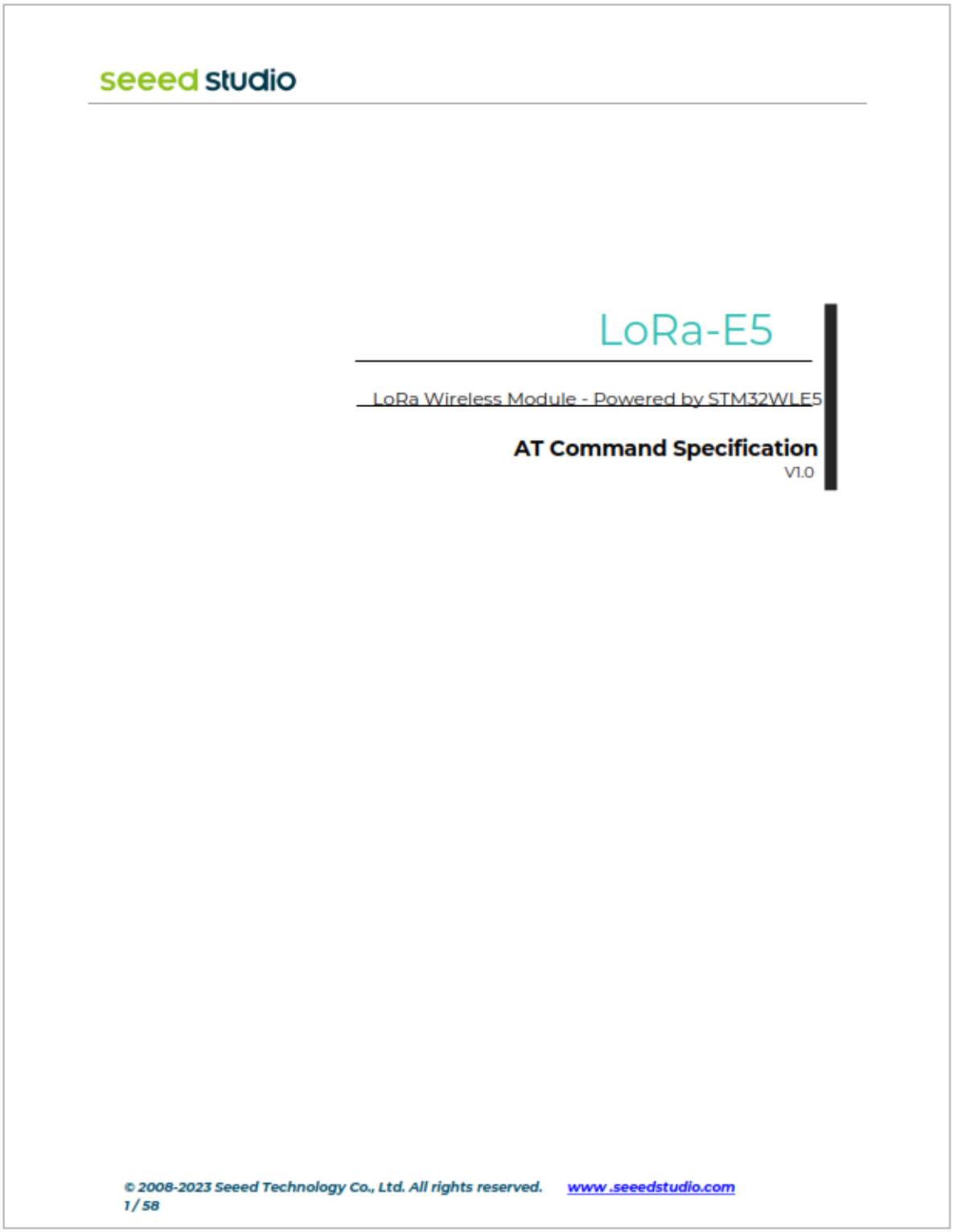
Pin	Function	Type	Description
1	+3V3	Power Out	Internally generated power output to be used as voltage reference
2	SWD	Digital	nRF52480 Single Wire Debug Data
3	SWCLK	Digital In	nRF52480 Single Wire Debug Clock
5	GND	Power	Power Ground
6	RST	Digital In	Active low reset input

SENSECAP M2 Muliplataform Gateway connect to TTN. Referencia:
https://wiki.seeedstudio.com/Network/SenseCAP_Network/SenseCAP_M2_Multi_Platform/Tutorial/Connect-M2-Multi-Platform-Gateway-to-The-Things-Network/



SenseCAP M2 Muli Platform Gateway & SenseCAP Sensors. Referencia:
https://files.seeedstudio.com/products/SenseCAP/M2_Multi-Platform_Gateway/Quick_Start_for_SenseCAP_Gateway_&_Sensors.pdf

Módulo Grove – Wio E5. Referencia:
https://wiki.seeedstudio.com/Grove_LoRa_E5_New_Version/



Módulo Grove – Wio E5 Datasheet. Referencia:

https://files.seeedstudio.com/products/317990687/res/LoRa-E5%20module%20datasheet_V1.0.pdf

seeed studio

4 Commands

Command	Description
AT	Test command
FDEFAULT	Factory data reset
RESET	Software reset
DFU	Force bootloader to enter dfu mode
LOWPOWER	Enter sleep mode
VER	Version[Major.Minor.Patch]
MSG	LoRaWAN unconfirmed data
MSGHEX	LoRaWAN unconfirmed data in hex
CMSG	LoRaWAN confirmed data
CMSGHEX	LoRaWAN confirmed data in hex
PMSG	LoRaWAN proprietary
PMSGHEX	LoRaWAN proprietary in hex
CH	LoRaWAN channel frequency
DR	LoRaWAN datarate
ADR	LoRaWAN ADR control
REPT	Unconfirmed message repetition
RETRY	Confirmed message retry
POWER	LoRaWAN TX power
RXWIN2	LoRaWAN RX window2
RXWIN1	LoRaWAN RX window1
PORT	LoRaWAN communication port
MODE	LWABP, LWOTAA, TEST
ID	LoRaWAN DevAddr/DevEui/AppEui
KEY	Set NWKSKEY/APPSKEY/APPKEY
CLASS	Choose LoRaWAN modem class(A/B/C)
JOIN	LoRaWAN OTAA JOIN
LW THLD)	LoRaWAN misc configuration (CDR, ULDL, NET, DC, MC,
BEACON	LoRaWAN Class B utilities
TEST	Send test serious command
UART	UART configure
DELAY	RX window delay
VDD	Get VDD
RTC	RTC time get/set
EEPROM	Write/Read EEPROM
WDT	Watchdog control
TEMP	Get Temperature
LOG	Log DEBUG/INFO/WARN/ERROR/FATAL/PANIC/QUIET

Table 4-1 Command List

 www.seeedstudio.com
22 / 58

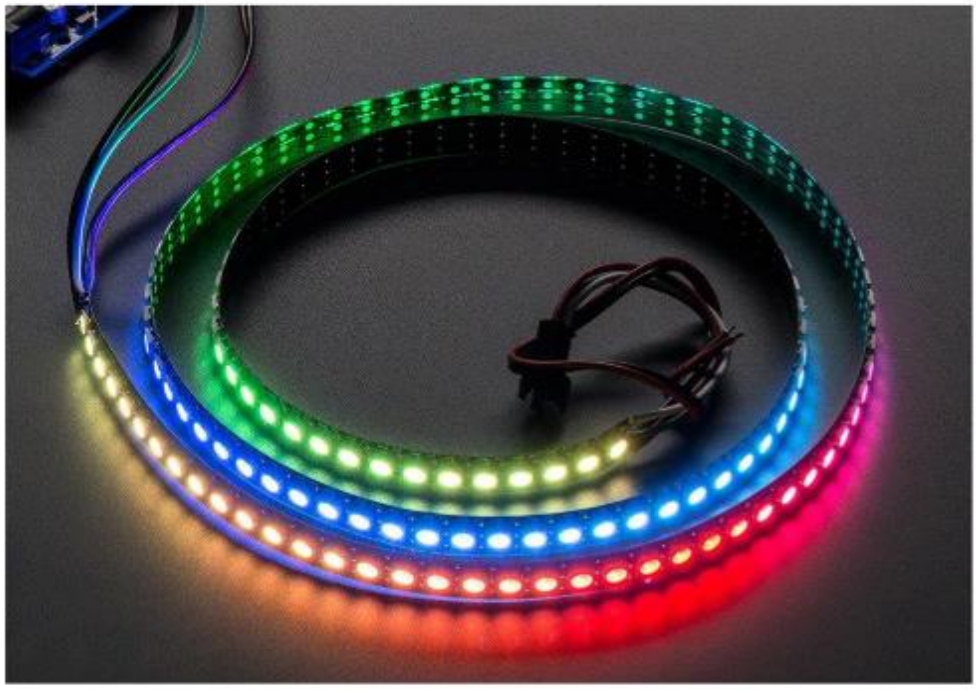
LED NEOPIXEL WS2812B. Referencia:

https://static6.arrow.com/aropdfconversion/175d93e2459dd52100714dbab770547b6930def5/datasheet_1506finalnologo.pdf

adafruit

Adafruit NeoPixel Digital RGB LED Strip 144 LED - 1m Black - BLACK

PRODUCT ID: 1506



DESCRIPTION

We crammed **ALL THE NEOPIXELS** into this strip! An unbelievable 144 individually-controllable LED pixels on a flexible PCB. It's completely out of control and ready for you to blink. This strip has a black mask, and an extra heavy flex PCB.

These LED strips are even more fun and glowy. There are **144 RGB LEDs per meter**, and you can control each LED individually! Yes, that's right, this is the digitally-addressable type of LED strip. You can set the color of each LED's red, green and blue component with 8-bit PWM precision (so 24-bit color per pixel). The LEDs are controlled by shift-registers that are chained up and down the strip so you can shorten or lengthen the strip. Only 1 digital out pin is

Anexo 3. Certificado de la traducción del Resumen al Idioma Inglés.

THE CANADIAN HOUSE CENTER

"Make today so awesome, yesterday gets jealous."

Loja, 25 de junio del 2024

CERTIFICADO DE TRADUCCIÓN

A quien corresponda,

Yo, José Geovanny Jiménez Balcázar, traductor oficial del The Canadian House Center, instituto privado especializado en la enseñanza del inglés como lengua extranjera y centro de traducción autorizado y acreditado por el Consejo Nacional de la Judicatura del Ecuador bajo la licencia profesional número 12282677, certifico que el trabajo de titulación previo a la obtención del título de Magíster en Telecomunicaciones, titulado **"Diseño e implementación de un sistema para mantenimiento predictivo que permita detectar anomalías en un motor eléctrico mediante TinyML y LoRaWAN."**, realizado por *Juan Carlos Zarumo Villamarín* portador de la cédula de identidad ecuatoriana *1104724263*, de la Universidad Nacional de Loja, ha sido traducido de buena fe del español al inglés en la institución antes mencionada, y es una traducción fiel y exacta del documento original según mi leal saber y entender.

El portador puede hacer uso de este certificado y del documento traducido para cualquier fin legal que consideren oportuno.

Lic. José Geovanny Jiménez Balcázar
TRADUCTOR OFICIAL
THE CANADIAN HOUSE CENTER
Email: chcloja@gmail.com
Tel: +593 (0)7 258 5435

CHC MATRÍZ: Venezuela 19 - 77 e/ José María Peña y Av. Pío Jaramillo Alvarado
CHC CENTRO: Miguel Riofrío 14 - 35 entre Bolívar y Sucre
2565257 (Chc Centro) - 2585435 (Chc Matríz)

www.thecanadianhousecenter.com
/CHCLoja