Escuela Superior Politécnica del Litoral

Facultad de Ingeniería en Electricidad y Computación

Estación meteorológica aplicando internet de las cosas con soporte en LoRaWAN para la agricultura inteligente.

Proyecto Integrador

Previo la obtención del Título de:

Ingeniero en Telecomunicaciones

Presentado por:

Alfonso Guido Lema Vistin

Andrés Xavier Cabrera Quizhpi

Guayaquil - Ecuador

Año: 2023

Dedicatoria

El presente proyecto lo dedico a mis queridos padres, el señor Gabriel Cabrera y la señora María Quizhpi. Sus consejos, dedicación y preocupación constante han asegurado que siempre tenga lo necesario para alcanzar mis metas. Este proyecto es el resultado de la educación y los valores que me han inculcado, y es a ellos a quienes se lo debo.

Andrés Xavier Cabrera Quizhpi.

El presente proyecto lo dedico a todas las personas que han creído en mí y han contribuido en mi formación profesional y personal. Por ellos estoy hoy en día donde estoy, desde Dios, mis padres, hermano, profesores, amigos e incluso enemigos. Como testimonio de ese apoyo les dedico este trabajo y esfuerzo.

Alfonso Guido Lema Vistin.

Agradecimientos

Nuestro más sincero agradecimiento a Lenin Morejón, por su confianza en nuestras habilidades. Sin su apoyo no hubiera sido posible este proyecto. Así también, extendemos el agradecimiento a nuestros docentes por guiarnos con su experiencia y conocimiento en la toma de decisiones para el avance de este proyecto.

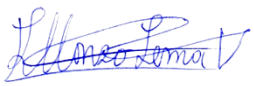
Declaración Expresa

Nosotros Andrés Xavier Cabrera Quizhpi, Alfonso Guido Lema Vistin acordamos y reconocemos que la titularidad total y exclusiva sobre los derechos patrimoniales de patente de invención, modelo de utilidad, diseño industrial, información no divulgada y cualquier otro derecho o tipo de Propiedad Intelectual que corresponda o pueda corresponder respecto de cualquier investigación, desarrollo tecnológico o invención realizada durante el desarrollo de su trabajo de titulación, incluyendo cualquier derecho de participación de beneficios o de valor sobre titularidad de derechos, pertenecerán de forma total, perpetua, exclusiva e indivisible a LA ESPOL, sin limitación de ningún tipo. Se deja además expresa constancia de que lo aquí establecido constituye un "previo acuerdo", así como de ser posible bajo la normativa vigente de transferencia o cesión a favor de la ESPOL de todo derecho o porcentaje de titularidad que pueda existir.

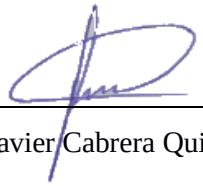
Sin perjuicio de lo anterior los alumnos firmantes de la presente declaración reciben en este acto una licencia de uso gratuita e intransferible de plazo indefinido para el uso no comercial de cualquier investigación, desarrollo tecnológico o invención realizada durante el desarrollo de su trabajo de titulación, sin perjuicio de lo cual deberán contar con una autorización previa expresa de la ESPOL para difundir públicamente el contenido de la investigación, desarrollo tecnológico o invención.

Así también autorizamos expresamente a que la ESPOL realice la comunicación pública de la obra o invento, por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual.

Guayaquil, 20 de septiembre de 2023.



Alfonso Guido Lema Vistin



Andrés Xavier Cabrera Quizhpi

Evaluadores

PhD. María Antonieta Álvarez Villanueva

Profesor de Materia

Ms.C. Verónica Alexandra Soto Vera

Tutor de proyecto

Resumen

El presente trabajo tiene como objetivo la implementación de una estación meteorológica automática (EMA) en ambiente agrícola, basada en una red de sensores de internet de las cosas (IoT). Para lograrlo, se establecieron 4 etapas para su construcción: alimentar el sistema, recopilar datos meteorológicos, transportarlos y visualizarlos en dispositivos digitales.

Por tal efecto, se utilizaron sensores meteorológicos de radiación, precipitación de lluvia, temperatura y humedad, todos compatibles con el microcontrolador programable, para procesar y recopilar datos. Se implementó un sistema de interconexión IoT en el microcontrolador, utilizando la tecnología Low Range Wide Area Network (LoRaWAN), para lo cual se estableció una arquitectura específica en el diseño de la EMA compatible con esta tecnología.

Para la comunicación, se empleó un módulo de comunicación inalámbrica compatible con la placa de desarrollo, el cual configuró el conjunto de sensores como el nodo meteorológico. Adicionalmente, se configuró un servidor virtual que permitió el almacenaje de datos empaquetados y codificados, siendo visible únicamente para el usuario específico. Estos datos se decodificaron y se integraron en plataformas de visualización en donde se determinó la confiabilidad de la EMA comparada con otros servicios.

Palabras clave: EMA, LoRaWAN, agricultura, ambiente.

Abstract

The aim of this research is to implement an Automatic Weather Station (EMA) in an agricultural setting using an Internet of Things (IoT) sensor network. To accomplish this goal, the project was divided into four key stages: powering the system, gathering meteorological data, transmitting the data, and displaying it on digital devices.

For this purpose, meteorological sensors were employed for measuring radiation, rainfall, temperature, and humidity. These sensors are fully compatible with the programmable microcontroller, enabling efficient data processing and collection. The IoT connectivity was established using Low Range Wide Area Network (LoRaWAN) technology, which required a tailored architectural design to seamlessly integrate with this technology.

To facilitate communication, a wireless communication module was employed, seamlessly configuring the sensor ensemble as the meteorological node. Furthermore, a virtual server was configured to securely store packaged and encrypted data, accessible exclusively to authorized users. These data sets were subsequently decoded and integrated into visualization platforms, which allowed for an assessment of the EMA's reliability compared to alternative services.

Key words: EMA, LoRaWAN, agriculture, environment.

Índice general

Abreviaturas

3G	Tercera generación.
API	Interfaz de programación de aplicaciones.
EMA	Estaciones meteorológicas automática.
EMC	Estaciones meteorológicas convencional.
ESPOL	Escuela Superior Politécnica del Litoral.
INANHI	Instituto Nacional de meteorología e Hidrología.
IoT	Internet of Things.
LoRa	Long Range.
LoRaWAN	Long Range Wide Area Network.
LTE	Long-term evolution.
OMM	Organización Meteorológica Mundial.
RF	Radio frequencies.
TTN	The Things Network.
TTNV3	The Things Networks V3.
TTS	The Things Stack.
UART	Universal Asynchronous Receiver-Transmitter.
LED	Light Emitting Diode.
ISM	Industrial, Científico y Médico.
ISP	Proveedor de servicios de internet.
TCP/IP	Transmission Control Protocol/Internet Protocol.
OTTA	Over-The-Air activation.
ABP	Activation By Personalization.
UDP/IP	datagram Protocol/Internet Protocol.
PHY	Physical Frame Number.

Simbología

bps	bits por segundo.
mil	milésima de pulgada.

Índice de figuras

Índice de tablas

Capítulo 1

1. Introducción

En Ecuador, la agricultura es un pilar fundamental para la economía de los ciudadanos, representa el 9,4% del Producto Interno Bruto (PIB) [1]; es la base de la seguridad alimentaria, produciendo el 95% de los bienes alimenticios consumidos en el país [2]; y, ayuda a la generación de empleo, con un 26,8% de la población económicamente activa (PEA) [3]. Por lo tanto, la agricultura interviene en la economía, brinda trabajo y asegura alimentos para la población ecuatoriana.

Dada su importancia económica, los productores buscan las predicciones meteorológicas para la toma de decisiones respecto a los cultivos. De esta manera, se sirven de la información que expone las estaciones meteorológicas para detectar variables como: temperatura, presión atmosférica, viento, radiación solar, humedad y precipitación [4].

En Ecuador, el Instituto Nacional de Meteorología e Hidrología (INAMHI) como organismo rector, coordinador y normalizador de la política nacional en temas de meteorología e hidrología, proporciona información oportuna y veraz sobre el tiempo, el clima y el agua a los diferentes sectores económicos y estratégicos, incluyendo el sector agrícola [5].

Para cumplir con su misión, requiere administrar una red de estaciones meteorológicas que cumplan los parámetros de la Organización Meteorológica Mundial (OMM), y ser suficientemente representativa, para que satisfaga la demanda de los usuarios internos y externos. De esta manera, la OMM menciona que las estaciones meteorológicas deben incluir mediciones de variables que caractericen el entorno físico de temperatura, radiación solar, precipitación, evaporización, humedad y velocidad de viento [6].

De hecho, el INAMHI cuenta con estaciones meteorológicas convencionales (EMC) para medir esas variables meteorológicas, en donde se requiere que un profesional se encargue de la recolección de datos en campo. Esto representa un problema, pues los datos están sujetos

a errores humanos al ser recopilados; y, generan un gasto de tiempo, ya que, deben desplazarse hasta tres veces al día [7].

Estudios refuerzan el problema del monitoreo meteorológico con EMC, pues algunas veces los operadores no pueden realizar la observación debido a la distancia (23%), fenómenos naturales (18%), motivos personales (20%) y el horario de observación (5%). Así también, existen estaciones meteorológicas e hidrológicas que deben ser reubicadas, debido a motivos de cambio de uso de suelo (5%), fenómenos extremos (15%) y cambio de predios (18%) [8].

Actualmente, con la presencia de instrumentos digitales, se ha facilitado el desarrollo de EMA, representando una solución a los problemas de recolección de datos meteorológicos, ya que no requieren de un profesional que recoja datos de manera presencial sino más bien de forma automática. Esta gran ventaja ha generado un mayor aumento en la migración de EMC a EMA [9]. No obstante, se sigue evidenciando que los sectores estratégicos, como el agrícola, aún persisten las EMC, representando una desventaja en competitividad.

En virtud de lo expuesto, se desarrolla el presente proyecto, el cual busca diseñar una EMA implementada con IoT para monitoreo agrícola ambiental, para brindar solución a la problemática de recopilación de variables: temperatura, humedad, radiación solar y precipitación.

1.1. Descripción del problema

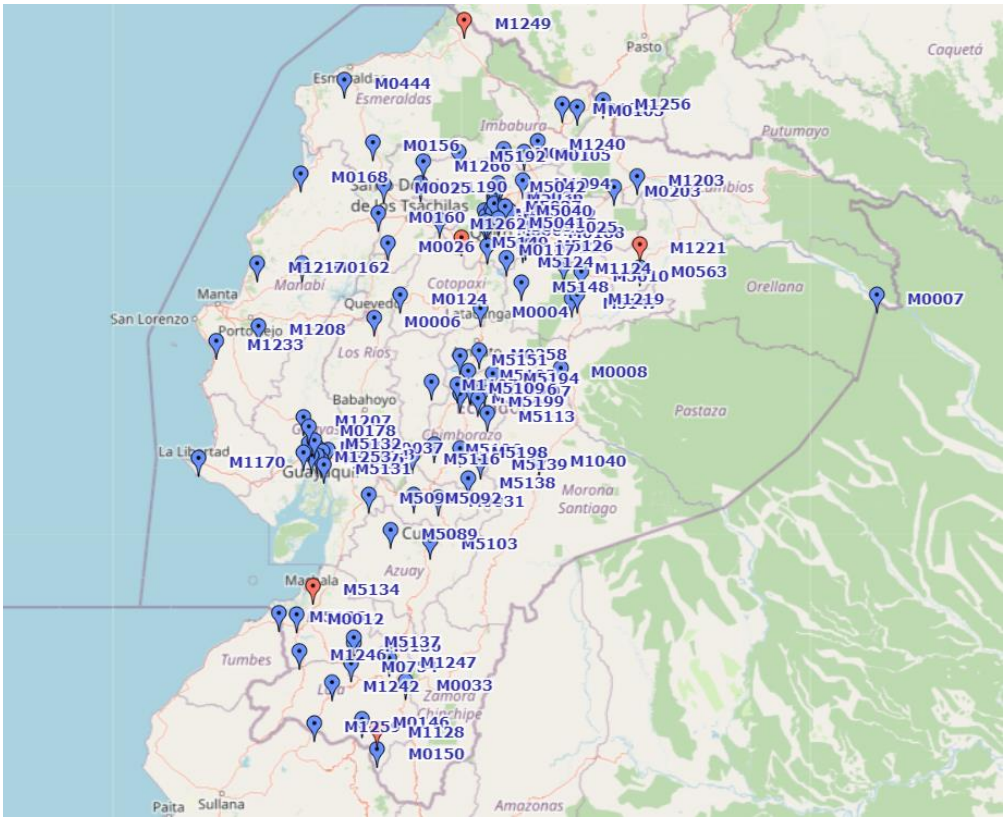
El cambio climático está afectando a varios sectores estratégicos, en especial, al agrícola. La variación global del clima en la Tierra genera que los patrones meteorológicos registrados en años pasados se alteren, lo cual complica la toma de decisiones en el sector agrícola. El predecir el clima permite aprovechar los períodos secos o lluviosos para la siembra, cultivo y cosecha de los productos agrícolas. Para tal efecto, Ecuador cuenta con entidades para la recolección, procesamiento y exposición de datos ambientales, para apoyar a la toma de decisiones en el sector agrícola mediante el monitoreo remoto [10].

El INAMHI se encarga de monitorear el clima, mediante 583 EMC y 170 EMA dentro del territorio ecuatoriano. En este sentido, es evidente que, en Ecuador se destinan recursos para la recopilación de datos por emplear EMC. Además, se pone en mesa la falta de EMA para el monitoreo de factores meteorológicos en tiempo real en los cultivos de empresas, lo cual afecta en la toma de decisiones, y repercute en su economía [7].

De manera más precisa, el INAMHI ha establecido una red de EMA en todo el territorio ecuatoriano. En la Figura 1 se presenta el mapa de las EMA en su plataforma oficial, donde las estaciones operativas están marcadas en azul y las no operativas en rojo. Actualmente, existen 5 estaciones que no están operativas, se ubican en: El Ingenio de Loja (M1128), Ministerio de Salud Machala de El Oro (M5134), San José Payamino de Orellana (M1221), Campamento de la Empresa de Pichincha (M5149), y Palesema de Esmeraldas (M1249).

Figura 1

Red de Estaciones Meteorológicas del Ecuador

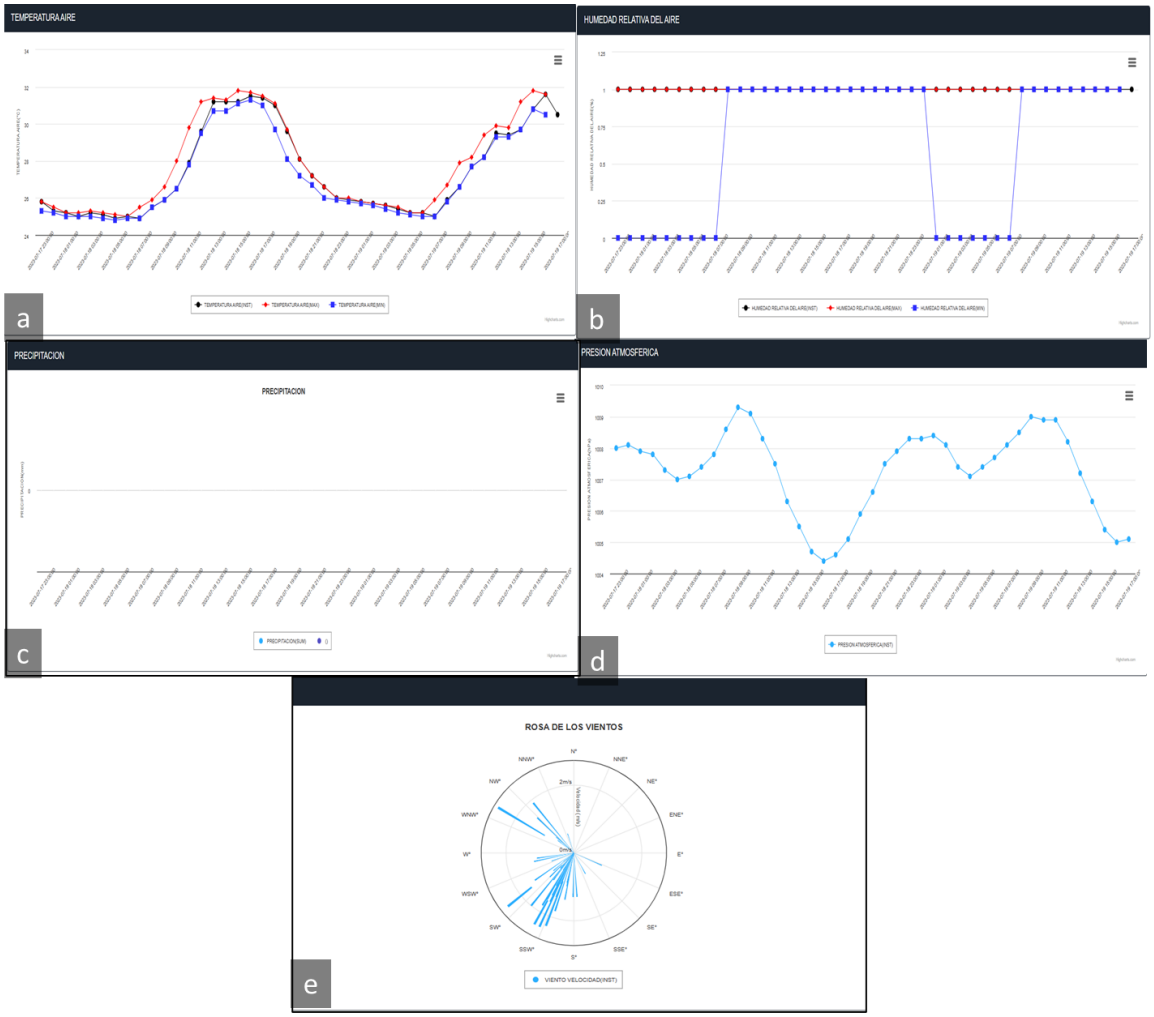


Nota. Tomado de INAMHI [11].

En este portal oficial, al seleccionar la estación de análisis ‘Ingenio Valdez’, se presentan datos instantáneos cada 24 horas, indicando, temperatura del aire (Figura 2a), humedad relativa del aire (Figura 2b), precipitación (Figura 2c), presión atmosférica (Figura 2d), y la rosa de los vientos para mostrar la velocidad y dirección del aire (Figura 2e). Es importante destacar que, aunque en el portal se puede visualizar todos los datos de la red meteorológica, tanto operativa como inactiva, no todos los datos de las variables están disponibles para otras estaciones. Esto sugiere posibles errores en la transmisión de datos al portal de visualización. [11].

Figura 2

Parámetros de la estación meteorológica Milagro (Ingenio Valdez)



Nota. Tomado de INAMHI [11].

La problemática descrita revela una serie de desafíos críticos en la gestión de datos meteorológicos en Ecuador, así como la limitada disponibilidad de datos en varias estaciones meteorológicas presentadas en el geoportal oficial del INAMHI, junto con la inoperatividad de 5 estaciones clave y las complicaciones en la recolección presencial de datos en áreas afectadas por fenómenos naturales extremos y debido a restricciones de horario, subrayan la necesidad de mejorar la gestión y el acceso a información meteorológica confiable para el sector agrícola.

1.2. Justificación del problema

La toma de decisiones en la industria agrícola es crucial para mantener la ventaja competitiva entre las empresas del sector y depende en gran medida de la calidad y de la información disponible. En ocasiones, la falta de datos precisos puede afectar los ciclos de producción [12].

Esta problemática de la confiabilidad de los datos se agrava con los efectos del cambio climático en el territorio. La Organización de las Naciones Unidas para la Agricultura y la Alimentación (FAO) señala que la agricultura es especialmente vulnerable a las variaciones climáticas. El aumento de la temperatura y los cambios en los patrones de lluvia pueden reducir la producción de cultivos incrementando la probabilidad de fracaso de las cosechas [10].

Otros autores concuerdan que los impactos del cambio climático afectan a los agricultores a pequeña escala, reduciendo el rendimiento de cultivos como el arroz, el maíz y el trigo, así como causando fenómenos climáticos impredecibles. Sin embargo, estas variaciones climáticas también pueden tener efectos beneficiosos en algunas regiones [13]. No obstante, estas variaciones de temperatura y lluvia son favorables para ciertas regiones. Por consiguiente, el estudio de estos parámetros favorece a la toma de decisiones frente al cambio climático, para conseguir beneficios económicos por el incremento en la producción.

En este sentido, se menciona a las EMA como equipos que proporcionan datos precisos sobre parámetros como temperatura, humedad, precipitaciones y radiación solar. Estas EMA

ofrecen ventajas significativas en la recolección de datos en comparación con las EMC, que requieren de un observador. Además, las EMA proporcionan datos en tiempo real y con mayor precisión, lo que beneficia la toma de decisiones en la agricultura frente a fenómenos ambientales [9].

Desde un punto de vista técnico, al realizarse una estación meteorológica autónoma utilizando IoT, se brinda un dispositivo resultado del conocimiento y experiencia de los investigadores en el campo de la electrónica y telecomunicaciones.

1.3. Objetivos

1.3.1. *Objetivo general*

Implementar una estación meteorológica automática basada en una red sensores IoT para facilitar la recolección, procesamiento y transmisión en tiempo real de datos meteorológicos en la finca ESPOL.

1.3.2. *Objetivos específicos*

- Realizar una investigación general de la literatura existente sobre las estaciones meteorológicas automáticas enfocándose en la identificación y evaluación del hardware adecuado y disponible para la implementación la red de sensores meteorológicos.
- Aplicar un sistema de interconexión IoT para los sensores meteorológicos.
- Implementar un sistema de comunicación basado en la tecnología LoRaWAN con el propósito de transmitir datos meteorológicos, incluyendo mediciones de temperatura, humedad y radiación solar al internet para su visualización en un dispositivo final.

1.4. Marco teórico

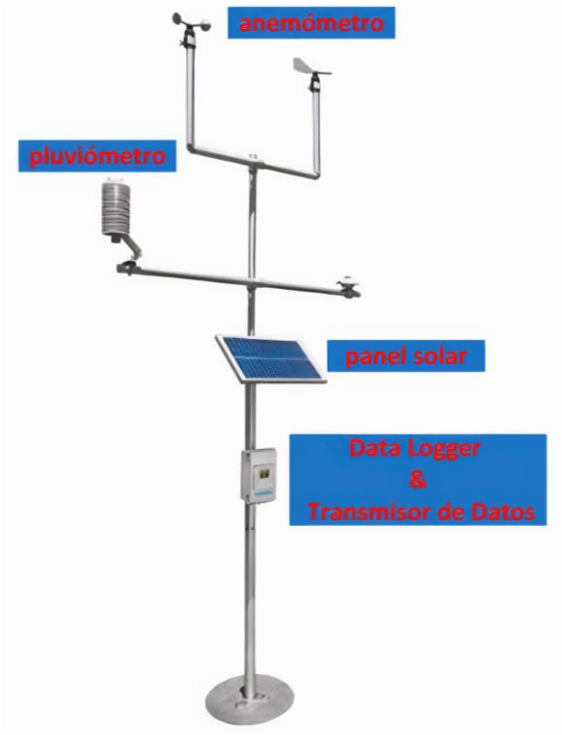
La presente sección proporciona conceptos fundamentales para la comprensión de la tecnología y variables a considerar para el desarrollo de la estación meteorológica del proyecto.

1.4.1. Estaciones meteorológicas automáticas (EMA)

Las EMA son equipos de registro de datos de forma continua y autónoma, conformada por un grupo de sensores que recogen información de parámetros ambientales en intervalos de tiempo menores a los obtenidos de manera manual. Se caracterizan por una construcción con elementos electrónicos, con registro de memoria sólida, instalación eléctrica y mecánica con varios sensores, los cuales son los responsables de la recopilación de los parámetros meteorológicos, limitados por la capacidad del dispositivo [14] (Figura 3).

Figura 3

Estación meteorológica Automática



Nota. Tomado de Darrera [15].

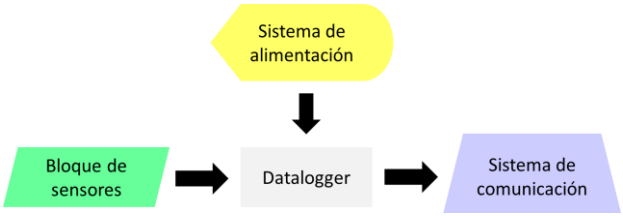
Las EMA están compuesta por 4 partes: datalogger, sensores, sistema de alimentación de energía, y sistema de comunicación y transmisión de datos (Figura 4). A continuación, se detallan los componentes:

- **Datalogger:** Es la unidad electrónica que se encarga del registro de manera sistemática [14].

- **Sensores:** Son los dispositivos electrónicos responsables de transformar un parámetro ambiental en un impulso eléctrico, el mismo que se registra, interpreta y muestra con un valor [14].
- **Sistema de alimentación de energía:** Es el módulo que permite brindar la energía necesaria para el funcionamiento del EMA. La energía eléctrica puede ser alterna o continua, a través de baterías según el voltaje que necesite el equipo. En los casos en donde la estación se ubique en lugares apartados, la energía se obtiene de paneles solares, procurando que el sistema de alimentación de energía sea el que satisfaga la necesidad de la demanda de energía total de la estación [14].
- **Sistema de comunicación y transmisión de datos:** Se encarga de gestionar los datos desde el *Datalogger* a un servidor, construyendo una base de datos histórica. El tipo de comunicación puede ser directa (computador – *Datalogger);* vía red telefónica, en donde se requiere de un módem interno junto al computador que se comunique con el módem externo del *Datalogger;* vía telefónica móvil, en donde se necesita de un módem; ondas de radio, en el que se requiere de antenas que cubran las condiciones topográficas del territorio; y, por satélite [14]. Así también, se puede realizar mediante tecnología bluetooth, con el cual se almacena los datos localmente en un smartphone o tableta [16].

Figura 4

Componentes de una EMA



Nota. Tomado de Darquea [14].

1.4.2. Estaciones meteorológicas convencionales (EMC)

Las EMC sirven para monitorear datos relacionados con el clima, en el que no se requiere de electricidad para su funcionamiento. Esto se debe a que se construyen con elementos mecánicos analógicos y químicos para registrar datos (Figura 5). Estas estaciones pueden ser usadas en el interior y al exterior [17]. Específicamente, estas estaciones se construyen con material inoxidable para evitar la corrosión de sus componentes [18]. Se distinguen de las EMA por necesitar un profesional que realice la tarea de observación e interpretación de datos, no ofrecen previsiones meteorológicas, se limita al registro actual; y su ubicación es de suma importancia para que las mediciones no sean alteradas por las condiciones climáticas [17].

Figura 5

Estación meteorológica Analógica

Nota. Tomado de Meteobenidorm [19].

1.4.3. Estaciones meteorológicas Automáticas con IoT

Las Estaciones Meteorológicas Automáticas con IoT se caracterizan por ser equipos que se componen de diversos sensores para medir las condiciones climáticas. Se caracterizan por una conectividad mediante tecnologías de comunicación [20]. De manera que, permite monitorear los parámetros ambientales en tiempo real, empleando internet para la transmisión

de datos. Usualmente, disponen de una consola de visualización para mostrar gráficas históricas. Esta consola emplea WIFI para la transferencia y almacenamiento de los datos. En caso de tener una página, los datos de la nube pueden exponerse mediante equipos digitales. No obstante, uno de los principales desafíos es su costo de adquisición. En la actualidad, en Ecuador, la incorporación de sensores adicionales aumenta el precio de estas estaciones, a diferencia del mercado extranjero, donde esta solución resulta ser más costosa [21] (Figura 6).

Figura 6

Estaciones meteorológicas Automáticas con IoT

Nota. Tomado de Tecno Industry [21].

1.4.4. Temperatura en la agricultura

La temperatura es una de las variables climatológicas con mayor incidencia tiene en la agricultura, su variación afecta a la cosecha y producción de los alimentos. Esto se debe a que la temperatura afecta de manera directa al crecimiento y desarrollo de muchas especies vegetales. Generalmente, la temperatura de los cultivos oscila entre los 18-25 °C, la cual permite que la planta crezca correctamente y brinde frutos [22].

1.4.5. Humedad en la agricultura

El parámetro de la humedad en la agricultura favorece a la formación correcta de plantas e incrementa el rendimiento de los cultivos, lo cual incrementa la producción e influye a la economía del propietario [22]. Una humedad demasiada alta provoca crecimiento débil, incremento de enfermedades de las hojas, deficiencia de nutrientes, aumento de enfermedades en las raíces, edemas y bordes quemados (gutación). En contraste, la humedad demasiada baja genera marchitamiento, plantas atrofiadas, tamaño más pequeño de hojas, puntas secas y quemadas, hojas rizadas e incremento de la infestación de arañuela roja [23].

1.4.6. Precipitación en la agricultura

La precipitación ayuda a satisfacer total o parcialmente las necesidades hídricas de las plantas. La precipitación afecta a la humedad del suelo, por lo que es necesario la medición de este parámetro para realizar planificaciones más acertadas. Los registros históricos de las precipitaciones permiten adelantarse a los eventos de sequía o muy lluviosos [24].

1.4.7. Radiación solar en la agricultura

La radiación impacta en procesos relacionados con la fotosíntesis, balances de agua, energía, crecimiento y desarrollo del cultivo. Los sensores se enfocan en la detección de la cantidad de luz solar y temperatura ambiente, así también, permite calcular la evaporación, lo que sirve para el inicio de los riegos. La radiación solar de la superficie de la tierra tiene una longitud de onda de 280 a 4000mm [25].

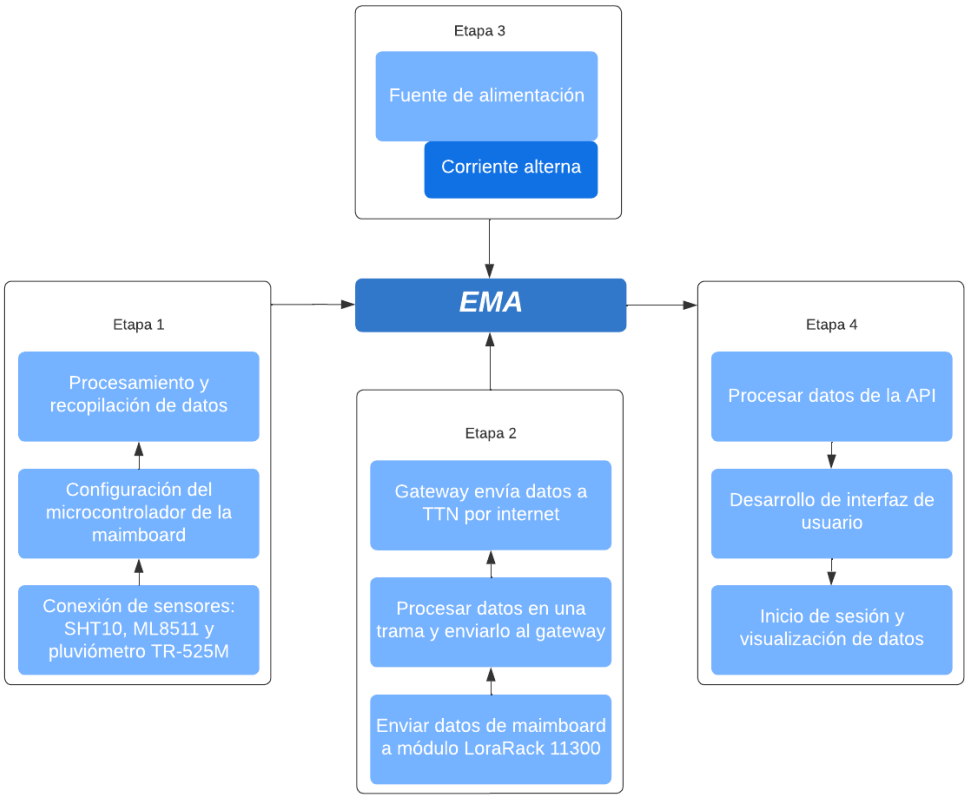
Capítulo 2

2. Metodología del diseño

El presente capítulo describe la metodología utilizada para la construcción de la EMA, el cual se realizó en cuatro etapas. En la primera se estableció la comunicación de los sensores con el microcontrolador, el cual se configuró para procesar y recopilar datos. La segunda etapa estableció el transporte de datos del microcontrolador al módulo de comunicación, para procesarlos y codificarlos en una trama hexadecimal para enviarlo a un Gateway, este a su vez lo transmite al servidor privado de The Things Network (TTN) usando la tecnología de comunicación LoRaWAN. La tercera etapa se centró en la instalación del sistema de suministro de energía a la EMA. Finalmente, en la cuarta etapa se realizó la capa de aplicación dedicada a la visualización de los datos en el dispositivo final, mediante integración de los datos a una Interfaz de programación de aplicaciones (API), donde se desarrolla una interfaz para la visualización de los datos meteorológicos. El flujograma de la Figura 7 resume la metodología.

Figura 7

Flujograma de la metodología del diseño de la estación meteorológica automática.



2.1. Hardware y sitio de implementación

En la presente sección se introduce el hardware para el diseño de la EMA con la tecnología LoRaWAN y el entorno que puede ser operado.

2.1.1. Microcontrolador, módulos y Gateway.

El hardware de este proyecto está diseñado para funcionar como un datalogger que sea compatible con sensores meteorológicos y que incorpore un sistema de comunicación eficaz. Para lograr esto, se ha empleado una placa de desarrollo y un módulo de comunicación inalámbrica, ambos coordinados a través de un Gateway. Adicionalmente, se han incluido módulos de entrada digital y analógica para asegurar la compatibilidad con una variedad de sensores. A continuación, se detallará más exhaustivamente cada componente del hardware.

Placa de desarrollo

En el Apéndice C se observa la placa de desarrollo, conocida también como mainboard o tarjeta madre, es el cerebro de la EMA. Está basada en Arduino Mega, utilizando el microcontrolador ATMEGA 2560, programable mediante el software de código abierto Arduino IDE. Las especificaciones técnicas se detallan en el Apéndice B, rescatando los protocolos de comunicación compatible con la mainboard y el número de estos puertos I2C y seriales disponibles, por lo cual la elección de sensores debe ser acorde a los puertos de comunicación.

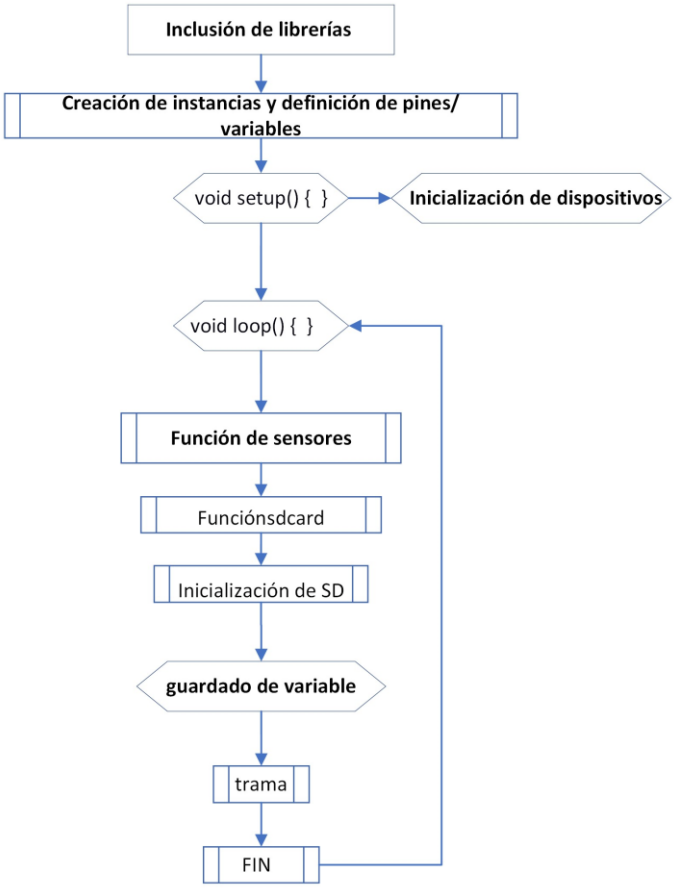
Con la comunicación entre los sensores y la mainboard, se configura la mainboard para que capture periódicamente los datos meteorológicos en intervalos de un minuto. Cumplido el minuto, se genera una trama de bits con los datos recopilados siguiendo un formato predefinido, que se almacena en una tarjeta MicroSD .

La interacción con los sensores perteneciente a la etapa 1 se logra incluyendo bibliotecas especializadas. El IDE realiza esta acción una vez. Para posteriormente iniciar la comunicación de acuerdo con su tipo de sensor.

El código que se ejecuta cada minuto es cíclico asegurando una captura y procesamiento constante de datos sin interrupciones. Este proceso se sintetiza en un flujograma que ilustra la secuencia de acciones llevadas a cabo en el sistema, presente en la Figura 8.

Figura 8

Flujograma de funcionamiento de la mainboard.



Módulo de entrada digital

En el Apéndice D se presenta el módulo de entrada digital, con capacidad de admitir hasta dos sensores digitales (X1 y X2), cada entrada está equipada de un pulsador generador de pulsos digitales para prueba. Los Leds indicadores en cada entrada confirman la recepción de pulsos. Además, el módulo dispone de terminales de voltaje (VCC) y tierra (GND) para la conexión de sensores que utilicen señales digitales y requieran ser adaptadas a la mainboard mediante ranuras ajustables.

Módulo de entrada analógica

En el Apéndice D se muestra el módulo de entrada analógica con capacidad para soportar hasta dos sensores analógicos (X1 y X2). Cada entrada dispone de 4 terminales: EN (habilitar), OUT (lectura), GND y VCC para la conexión con sensores que utilicen señales analógicas y requieran ser adaptadas a la mainboard mediante ranuras ajustables.

Módulo de comunicación LoRaWAN

El módulo de comunicación LoRaWAN permite cumplir la etapa 2, que consiste en procesar y empaquetar la trama de datos registrados por la mainboard, para la transmisión de forma inalámbrica a un servidor. El chip RAK11300 está basado en la tecnología Raspberry Pi RP2040 (Apéndice E), programable en el lenguaje Python. Adicionalmente, cuenta con un transceptor de radio Semtech SX1262, que garantiza la compatibilidad con LoRaWAN, siendo un protocolo de red que utiliza la tecnología Low Range (LoRa) en el ecosistema de radio frecuencia (RF). Utilizando las bandas industriales, científicas y médicas (ISM), llegando a tener un alcance máximo de 15 Km hasta la puerta de enlace (Gateway), con una antena receptora de ganancia adecuada.

Dado que el chip RAK11300 no es modular, se acopla el Wisblock RAK11300, el cual es compatible con Arduino mediante la interfaz Universal Asynchronous Receiver-Transmitter (UART). Para adaptarlo a la mainboard, se utilizó una tarjeta de circuito impreso (PCB). En el Apéndice F se muestra la combinación del chip RAK11300, el módulo Wisblock, y la PCB personalizada para la transmisión inalámbrica de datos en aplicaciones IoT. En el Apéndice G se muestran las especificaciones técnicas del módulo.

La ficha técnica del fabricante de la mainboard indica que se dispone de 4 puertos serie con capacidad para emplear UART. Esta característica justifica la interconexión IoT entre la placa de desarrollo y el módulo de comunicación para el diseño del nodo sensor [26],

permitiendo un acondicionamiento ajustable y programable mediante uno de puertos serie a una velocidad de transmisión de 9600 baudíos entre dispositivos.

Esta flexibilidad en la interconexión facilita la programación del módulo de comunicación en el mismo entorno de la mainboard, utilizando el software IDE, el cual requiere de uso de las librerías **‘**include <SPI.h>’ y ‘include <Arduino.h>’, establecidos por el fabricante [27]. Adicionalmente, se emplea el conjunto de librerías de LoRaWAN para cumplir con la función de comunicación inalámbrica como se muestra en la Figura 9.

Figura 9

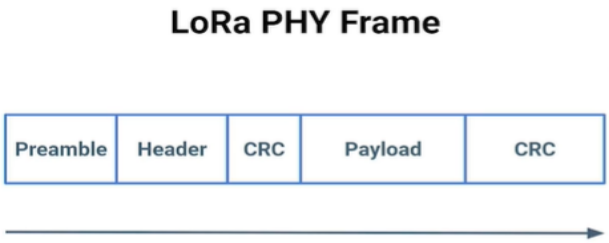
Librerías requeridas para programar en el módulo de comunicación RAK11300.

```
#include <Arduino.h>
#include "LoRaWan-Arduino.h" //http://librarymanager/All#SX126x
#include <SPI.h>
```

El código a programar en el módulo cuenta con las mismas funciones de la placa mainboard: ‘void setup()’ y ‘void loop()’. En ‘void setup()’ se inicializa la comunicación UART e invocan claves de seguridad para modos activación por aire (OTAA) y activación por personalización (ABP) necesarios para validar credenciales entre el chip RAK11300 y el servidor de la red LoRaWAN. En ‘void loop()’ se recogen los datos en formato decimal enviados por la placa de desarrollo a través del puerto serie. Estos datos se convierten a formato hexadecimal y se empaquetan en un bloque conocido como Payload. Adicionalmente, se incluyen otros bloques como ‘Preamble’, que permite la sincronización con el receptor, ‘Header’ como encabezado, y ‘CRC’ como delimitador. Estos bloques están definidos en este tipo de modulación para garantizar la transmisión inalámbrica de los datos. Este empaquetado se conoce como un mensaje PHY Frame, y su estructura por bloques se muestra en la Figura 10.

Figura 10

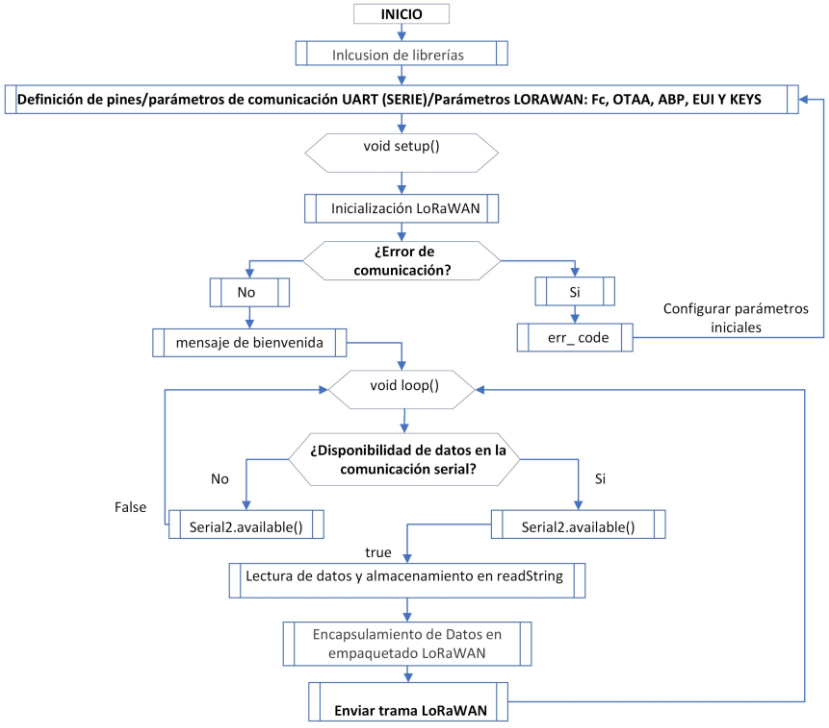
Estructura física del empaquetado de la trama de bytes.



El comando que se emplea en el código para enviar el PHY Frame de forma inalámbrica es 'send_lora_frame', el cual será recibido por el servidor que contenga las credenciales OTAA y ABP. Este PHY Frame se decodifica en el lado del servidor para visualizar los datos meteorológicos. El proceso detallado del acondicionamiento se muestra en la Figura 11.

Figura 11

Configuración del módulo de comunicación LoRaWAN



Gateway LoRaWAN

El Gateway es el dispositivo que se conecta de forma inalámbrica con el nodo sensor para direccionar el PHY Frame hacia la red de internet. Para enviarlo al servidor, se utilizan un

conjunto de reglas que permiten a los equipos comunicarse con las redes a través de internet [28]. Siendo Transmission Control Protocol/Internet Protocol (TCP/IP) el protocolo de red utilizado por los proveedores de servicios de internet (ISP). Los paquetes que llegan al servidor utilizan el protocolo User datagram Protocol/Internet Protocol (UDP/IP) elegido por su velocidad y eficiencia en la transmisión, a pesar de no garantizar la entrega absoluta de paquetes.

En el presente proyecto, se configuró el servidor The Things Network (TTN) como ruta de llegada del Gateway el cual se muestra en el Apéndice H, siendo de la familia Laird Connectivity Sentrius™ RG191 LoRa-Enabled Gateways, para la comunicación con los nodos.

2.1.2. Sensores

Los sensores que se exponen fueron seleccionados por la relación costo-beneficio de los inversores, tomando en cuenta los estándares que propone la OMM [29]. Estos fueron: SHT10 para temperatura y humedad, ML8511 para radiación solar y el pluviómetro TR-525M para precipitación de lluvia. La precisión de los sensores se detalla en el Apéndice A. Su ilustración se muestra en el Apéndice I.

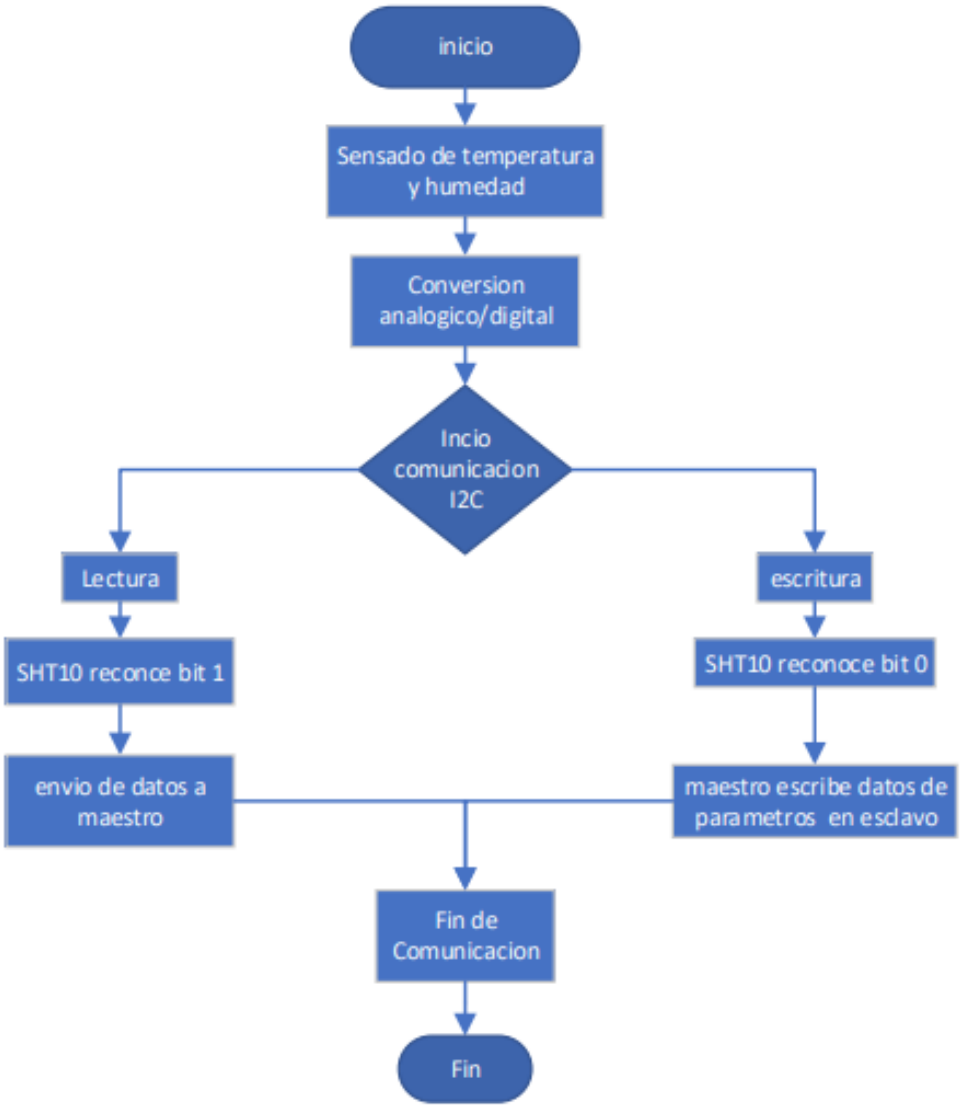
Sensor SHT10

El SHT10 es un sensor de temperatura y humedad, apto para diseño exterior, protegido por una malla metálica que permite exclusivamente el paso del aire. Por ende, la temperatura pasa al sensor sin alterar su valor. De tal manera que, se logra una larga durabilidad y precisión [30]. En este sistema, la placa de desarrollo inicia la comunicación enviando una señal como bit de lectura (1) o escritura (0). Una vez reconocida la señal, se inicia la transferencia de datos, ya sea para obtener lecturas del sensor o para la configuración de parámetros.

Los datos recogidos por el SHT10 se digitalizan usando el protocolo de comunicación I2C entre Arduino y el sensor, a través de la librería AHT10. Este protocolo utiliza los cables SDA y SCL en un sistema maestro-esclavo. El funcionamiento se resume en la Figura 12.

Figura 12

Flujograma de funcionamiento del sensor de temperatura

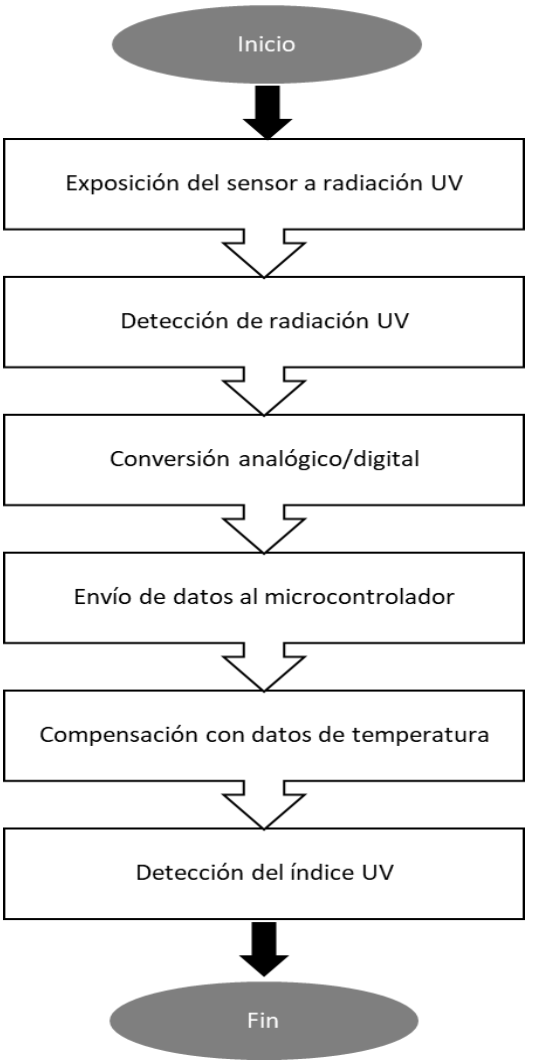


Sensor ML8511

El módulo ML8511 es un sensor de luz ultravioleta (UV) que proporciona una señal de voltaje analógica dependiendo de la cantidad de luz UV. La comunicación con el microcontrolador emplea un pin analógico y digital. El pin analógico recibe datos; el digital controla el umbral y envía mensaje al microcontrolador [31]. En la parte física, este sensor requiere de ciertas medidas de protección para garantizar lecturas precisas en condiciones específicas. Su funcionamiento detallado se muestra en la Figura 13.

Figura 13

Funcionalidad del sensor de radiación solar

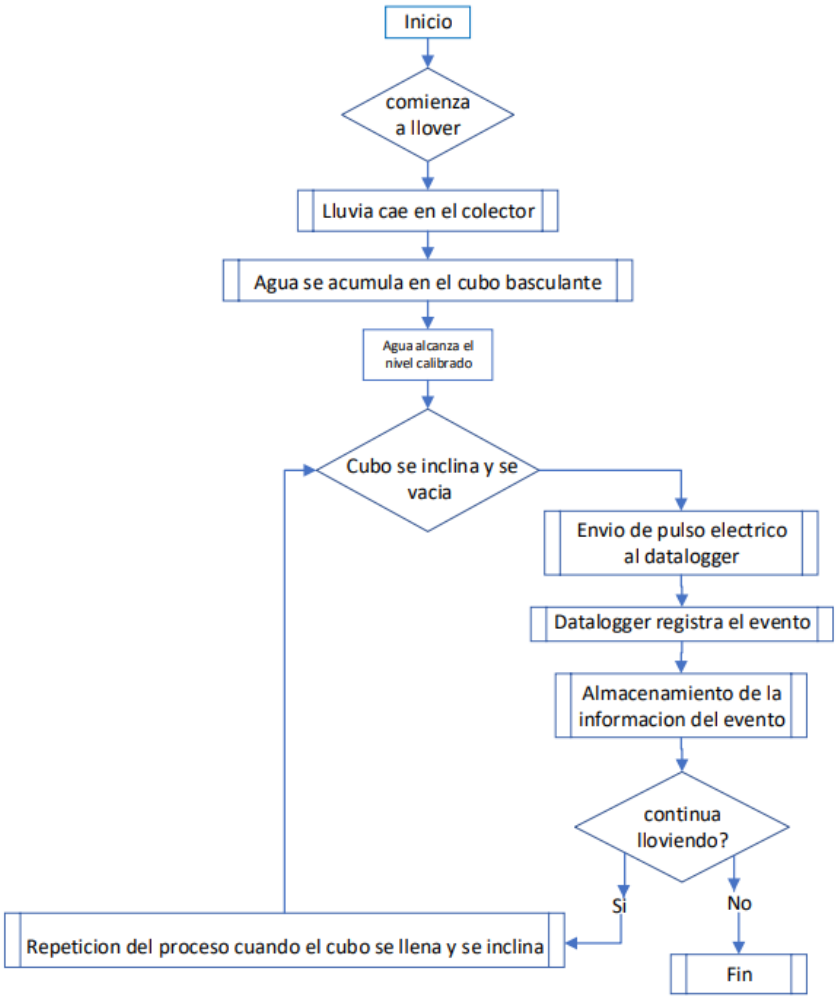


Pluviómetro TR-525M

El pluviómetro TR-525M es un dispositivo mecánico de medición de precipitaciones líquidas, con un estilo de balde de volteo remoto. Para llevar a cabo su función, presenta una abertura en la parte superior del instrumento, para capturar la lluvia, la cual fluye al balde basculante. Conforme el agua se acumula, el balde basculante se llena al punto de volcarse, provocando el cierre momentáneo de un interruptor. Este evento genera una señal eléctrica, que se traduce en un dato de la precipitación de lluvia, el funcionamiento detallado en la Figura 14 explica el flujograma del funcionamiento.

Figura 14

Flujograma del funcionamiento del pluviómetro



En este trabajo, se emplea un balde de volteo remoto con la capacidad de almacenar 4,73 ml aproximadamente, con una resolución de 0,1ml de precipitación por inclinación. Cuando el cubo basculante se inclina, un imán activa un interruptor que envía un pulso eléctrico. Este pulso es registrado por el datalogger [32]. El datalogger se reemplaza por programación en la tarjeta madre, lo que garantiza que se cumplan las funciones mencionadas previamente.

2.1.3. Sitio de implementación y pruebas

En la presente sección se detallan los requerimientos que debe cumplir el entorno para implementar la EMA con la tecnología de comunicación LoRaWAN.

Sitio de implementación.

El sitio de implementación debe contar con un enrutador con conexión a internet y suministro de energía eléctrica (Figura 15). Adicionalmente, la EMA diseñada debe cumplir con el requisito de la instalación de equipos meteorológicos que deben estar ubicados en un sitio despejado y sin obstrucciones [33].

Figura 15

Elementos a considerar para el sitio de implementación.

Pruebas

Las pruebas realizadas para evaluar la fiabilidad de la EMA instalada en el sitio de implementación deben comparar sus datos con otros servicios meteorológicos confiables. Estos servicios incluyen tantos sistemas satelitales y EMAs de entes privados o públicos con el fin de determinar si hay similitud o diferencia significativa en los datos.

Los servicios meteorológicos satelitales ofrecen amplia gama de datos meteorológicos, los cuales son registrados por satélites geoestacionarios. Entre estos servicios se destaca a AccuWeather como fuente confiable para obtener datos meteorológicos. Sin embargo, es importante señalar que, aunque AccuWeather puede ofrecer una alta precisión en ciertos aspectos meteorológicos a nivel global o regional, su precisión puede no ser tan alta en lugares específicos o regiones particulares [34].

En el caso de uso de otras EMAs para la comparación, se utiliza como recurso el servicio de las condiciones actuales del tiempo proporcionados por la INHAMI, como se definen en la Figura 1, en la Red de Estaciones Meteorológicas del Ecuador [11], sin embargo, para mejor precisión se debe hacer la comparación con la estación más cercana al sitio de pruebas.

2.2. Diseño del sistema de comunicación de la estación meteorológica

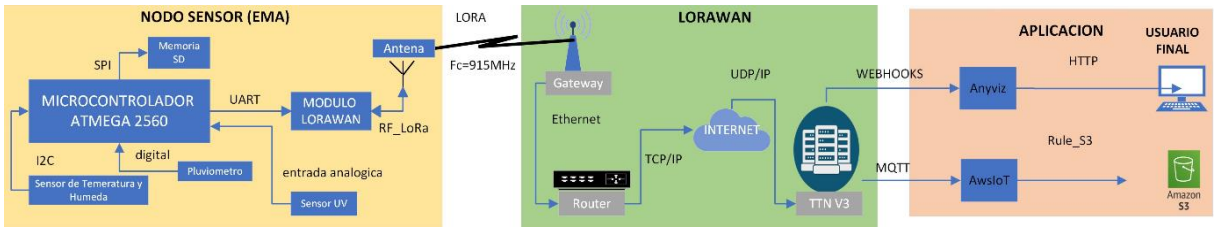
El diseño del sistema de comunicación LoRaWAN se centra en una arquitectura de transporte y de aplicación que permite al nodo sensor, descrito en la sección anterior dedicado al hardware, transmitir datos meteorológicos en intervalos de un minuto. El objetivo es hacer que estos datos sean accesibles al usuario final desde cualquier dispositivo. Los detalles de la tecnología empleada y del diseño se explican en la presente sección.

2.2.1. Diseño del sistema de comunicación

La capa de transporte utiliza la tecnología LoRaWAN para manejar la información meteorológica recolectada por los sensores a través de la placa de desarrollo, que está equipada con el microcontrolador ATMEGA 2560. Esta tecnología emplea la modulación LoRa y envía los datos al aire en forma de PHY Frame, utilizando una antena transmisora. El tamaño máximo del PHY Frame es de 242 bytes. En la Figura 16, se muestra el esquema del sistema de comunicación. Aquí se destaca el Gateway como el intermediario entre la capa de transporte y la red. Este Gateway se conecta a la red mediante una conexión Ethernet con el router para enviar el PHY Frame a la nube. El destino final es un servidor específico de TTN, donde se aplican procesos de integración para decodificar el Payload. Posteriormente, se procede con la integración del Payload decodificado con el objetivo de visualizarlo en un sitio web. Se emplean Webhooks y AWSIoT para almacenar estos datos en una base de datos de forma permanente.

Figura 16

Diagrama de bloques: Nodo sensor (EMA), LoRaWAN, Aplicación



2.2.2. Modulación LoRa

LoRa es una tecnología de modulación de largo alcance y bajo consumo, ideal para aplicaciones de IoT. Utiliza un espectro ensanchado para transmitir datos encapsulados como señales de radiofrecuencia, haciendo uso de las bandas ISM con una frecuencia central de 915 MHz. Esta tecnología puede superar obstáculos y penetrar edificios, casas y árboles, garantizando una comunicación confiable entre el nodo y el Gateway. Por lo tanto, es especialmente útil en aplicaciones que requieren una amplia cobertura y una larga duración de la batería, como en las Estaciones Meteorológicas Automáticas (EMA).

No obstante, en entornos desafiantes es crucial considerar diversos aspectos que afectan la calidad de la comunicación en el canal inalámbrico. Elementos como el factor de esparcimiento o Spreading Factor (SF), el nivel de potencia de transmisión y la tasa de bits influyen en la transmisión y recepción de datos. La elección adecuada de estos parámetros permite optimizar la eficacia de la comunicación en condiciones adversas, maximizando la confiabilidad y minimizando la interferencia. Estos aspectos técnicos son de importancia primordial en el diseño y configuración de la red LoRa para garantizar una comunicación sólida y coherente en el entorno de las EMA.

2.2.3. Servidor The Things Networks (TTN)

The Things Network (TTN) actúa como un respaldo temporal para el procesamiento de datos a través de sistemas backend, facilitando la renderización de información IoT en un

entorno de servicios en la nube, comúnmente conocido como Cloud Computing. Sin embargo, en abril de 2021, este servicio transfirió sus operaciones a The Things Stack (TTS), que representa una evolución de The Things Network en su versión 3 (TTN V3). Esta migración permitió una arquitectura más escalable para la red LoRaWAN, beneficiando tanto a usuarios que buscan soluciones empresariales como a aquellos que necesitan herramientas para pruebas y evaluaciones. La transición a TTS introdujo nuevas API para plantillas de integración que ofrecen oportunidades adicionales.

Se utilizó el servicio The Things Stack (Cloud), que ofrece la posibilidad de tener un servidor privado gratuito con un límite de hasta 10 nodos o dispositivos finales registrados y Gateway, donde se registra el Gateway como el módulo de comunicación. TTS actúa como el punto de entrada para los datos procesados, respaldados en formato JSON con detalles esenciales sobre la comunicación, calidad de la información y la carga útil codificada. Esta información se actualiza cada minuto según la configuración de la mainboard. Sin embargo, es importante mencionar que este respaldo no es permanente, ya que TTS reinicia estos datos en intervalos predefinidos. Por lo tanto, se recurrió a integraciones adicionales para asegurar la conservación de los datos meteorológicos y su visualización en un frontend.

2.2.4. Integración de datos y visualización

Este apartado es el perteneciente a la etapa 4, destinado al registro de nodos, donde se establece el parámetro ID, una pequeña descripción y los parámetros del dispositivo. Para la activación del nodo, se emplean las “claves de sesión” proporcionadas por el servidor de red para encriptar los paquetes enviados, de esa manera se identifica el propietario del nodo LoRa. En TTS se emplean las siguientes aplicaciones:

• MQTT: Este protocolo de mensajería ligero se basa en el principio de publicador-suscriptor. Los sensores de la EMA actúan como publicadores, mientras que cualquier cliente MQTT, en un servidor local o cualquier frontend, actúa como suscriptor [35].

• Webhooks: son una forma de automatización en la que una aplicación o servicio puede enviar datos en tiempo real a otra aplicación o servicio a través de solicitudes HTTP/HTTPS [36].

• AWSIoT: Envía datos directamente a AWSIoT a través del protocolo MQTT como un widget de Amazon Web Services (AWS) para una arquitectura de servicios en la nube escalables donde no se puede invertir en servidores locales, clave para una solución full Stack [37]. Para el caso de dispositivos IoT, como la EMA. La integración con AWSIoT asegura que los datos recopilados se procesen y almacenen de manera eficiente y segura [38].

2.3. Consideraciones éticas, legales y de seguridad

En la presente sección se detalla los criterios de consideraciones legales a considerar para la implementación de EMA como el uso del espectro y la privacidad de los datos.

Uso de bandas libres (ISM)

Las bandas ISM establecidas entre las frecuencias 902-928 MHz (frecuencia central 915 MHz) [39], está destinada para el uso sin fines de lucro a equipos e instalaciones para fines industriales, médicos, domésticos o similares [40]. Entre ellas se incluye LoRaWAN, por tal motivo, no se requiere de licencias para el uso de esta frecuencia.

Privacidad de datos

En Ecuador se garantiza el ejercicio del derecho a la protección de datos personales, contenidos en cualquier soporte, automatizados o no [41], para tal efecto, se emplea la plataforma TTN como servidor. De tal manera que, el acceso a la información queda bajo la aprobación por parte de los cooperadores que registraron el Gateway. Así también, se toma en consideración las Normas Internacionales ISO/IEC 27000, para la gestión de la seguridad de los datos, de manera que, se plantea mejorar de manera continua y aplicar las buenas prácticas en seguridad [42].

Capítulo 3

3. Diseño de arquitectura

En presente capitulo se muestra el resultado de las pruebas de implementación de la EMA en un terreno agrícola privado del sector Durán (Apéndice L), donde se solventaron cada una de las etapas.

En la etapa 1, se utilizaron los tipos de comunicación especificados en el apartado 2.1.1 sobre microcontroladores, módulos y Gateway. El resultado del acondicionamiento generó información de 76 Bytes que se guardan en la tarjeta MicroSD de 32 GB. Posteriormente, se utiliza la comunicación UART entre el módulo y la placa de desarrollo ajustados en 9600 baudios para que los datos digitalizados llegan por puerto serial.

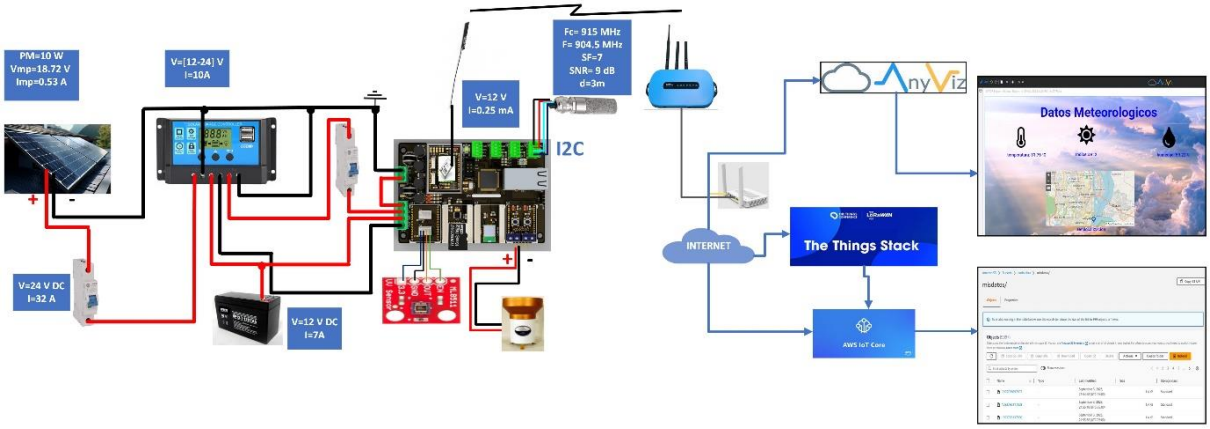
En la etapa 2, se programó el módulo de comunicación, en donde se codificó la información meteorológica en un Payload de 48 Bytes, el Gateway se implementó en una distancia de 3 metros con el nodo, dando como resultado una rápida calidad de envió de datos según el SF=7. Dicho Payload se decodificó en TTS donde se obtuvo datos meteorológicos de radiación solar, temperatura y humedad.

En la etapa 3, se trabajó con un proceso autosustentable del cual aprovecha la energía calorífica del sol, para esto se usa un panel solar de potencia 10 W de corriente corto-circuitada de 0,57 A. Adicionalmente, de una batería de 12 V a 7 A para energizar continuamente la EMA. Para direccionar la carga se usó un módulo para panel solar de10 A. Se usó un tablero de categoría IP65 para protección del nodo en el ambiente exterior de su uso.

En la etapa 4, se utilizó integración por Webhooks mediante la plataforma AnyViz y AWSIoT para respaldar la información. Todos estos procesos y conexiones se detallan en el diagrama del diseño implementado en la Figura 17.

Figura 17

Diagrama del diseño implementado



3.1. Dimensionamiento y cálculos de autosustentabilidad.

Se optó por un sistema autosuficiente para la EMA usando energía solar, basado en datos de 2021 de la NASA. Con las coordenadas de la estación, se determinó que la orientación horizontal del panel solar es la más eficaz, generando un máximo de 3,79 horas pico de radiación solar, según la Tabla 1.

Tabla 1

Datos mensuales de radiación de horas solar pico

Meses del año	Irradiación solar en posición horizontal	Irradiación solar latitud de -15 inclinación	Irradiación solar latitud de +15 inclinación	Irradiación solar en posición vertical
Enero	3,77	3,86	3,46	1,38
Febrero	3,74	3,75	3,54	1,44
Marzo	3,94	3,88	3,83	1,48
Abril	4,05	3,88	4,06	1,86
Mayo	3,69	3,46	3,81	2,06
Junio	3,43	3,16	3,59	2,09
Julio	3,45	3,2	3,61	2,05
Agosto	3,72	3,51	3,81	1,86
Septiembre	3,95	3,84	3,9	1,49
Octubre	3,84	3,83	3,66	1,35
Noviembre	4	4,09	3,69	1,39
Diciembre	3,92	4,05	3,55	1,35
Promedio	3,79	3,71	3,71	1,65

Para el diseño y construcción de la estación meteorológica, se procedió a obtener el consumo energético requerido para su funcionamiento. Para realizar esto, se calculó los valores individuales de consumo de energía de cada nodo.

El módulo RAK11300 genera dos etapas en un minuto: transmisión para datos y estado en reposo, debido a eso se procedió a obtener el consumo de energía requerida, dando un valor de 0,0138 Wh, según se observa en la Tabla 2.

Tabla 2

Consumo de energía requerida del módulo RAK11300

Potencia de transmisión(W)	tiempo de transmisión(h)	Potencia en reposo(W)	tiempo en reposo(h)	Consumo (Wh)
0,0886	0,0009	0,0137	0,9999	0,0138

Ya que la EMA funciona durante las 24 horas del día, se procedió al cálculo respectivo y resultó en un consumo energético total de 36,4179 Wh, como se observa en la Tabla 3.

Tabla 3

Consumo energético total de la EMA

Nodo	Potencia (W)	Horas de uso	Consumo Energético (Wh)
RAK11300	0,0138	24	0,3303
SHT10	0,0030	24	0,0720
ML8511	0,0007	24	0,0156
Placa de desarrollo	1,5	24	36
Suma total			36,4179

En la siguiente Tabla 4, se utilizó las siguientes ecuaciones para determinar resultados de potencia pico, número de panel solar a implementar, cantidad de batería, controlador de carga y el dimensionamiento del cableado.

Tabla 4

Ecuaciones Utilizadas para los cálculos.

No.	Ecuaciones
ecuación (1)	$$\boldsymbol{Wp = \frac{Consumo\ energ\acute{e}tico\ total}{Hora\ solar\ pico}}$$
ecuación (2)	$$\boldsymbol{Cantidad\ de\ M\acute{o}dulos = \frac{Wp}{Potencia\ del\ panel\ solar}}$$
ecuación (3)	$$\boldsymbol{Amperaje\ Hora = \frac{Consumo\ energ\acute{e}tico\ total\ \times d\acute{i}a}{voltaje \times \%\ de\ descarga}}$$
ecuación (4)	$$\boldsymbol{Cantidad\ de\ bater\acute{i}as = \frac{Amperaje\ Hora}{Amperaje\ Hora\ de\ bater\acute{i}a}}$$
ecuación (5)	$$\boldsymbol{Regulador\ [A] = cortocircuitada\ \times\ Cantidad\ de\ M\acute{o}dulos}$$
ecuación (6)	$$\boldsymbol{Intesidad = Regulador\ [A] \times 1,25 \times 1,25}$$
ecuación (7)	$$\boldsymbol{Secci\acute{o}n = 2 \times \frac{Longitud \times Intensidad}{conductividad \times caida\ de\ tensi\acute{o}n}}$$

Cálculo de potencia pico

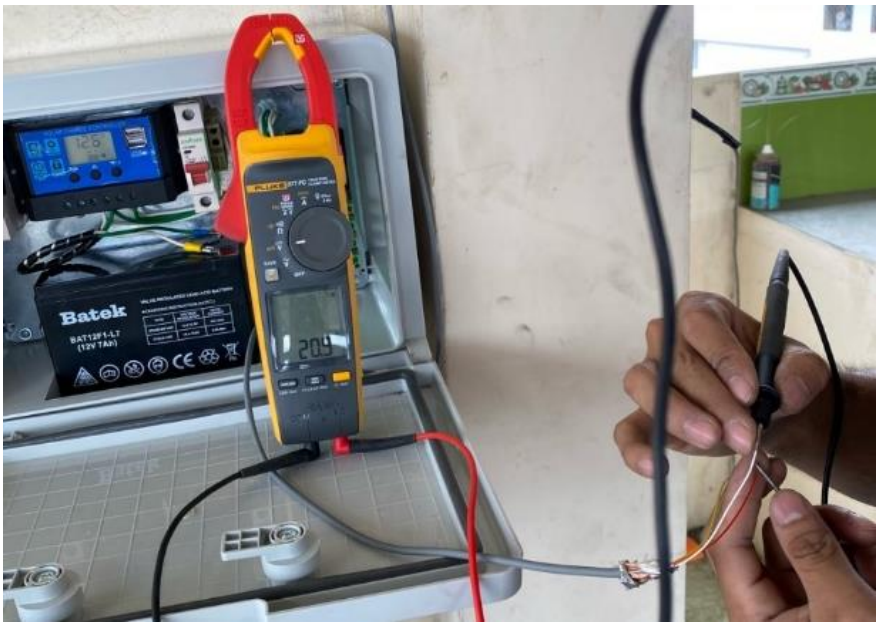
Como resultado de obtener los valores de consumo energético total y hora solar pico, se procedió a utilizar la ecuación (1), donde se tiene un valor de potencia pico de 9,52 W.

Cálculo del panel solar

Para determinar la cantidad de paneles solares, se seleccionó un panel del mercado con especificaciones de 10 W de potencia, 18 V de voltaje y 0,57 A de corriente. Durante las pruebas realizadas en Durán, el multímetro obtuvo un valor de voltaje en salida de 20,9 V en condiciones reales, como se observa en la Figura 18. Usando la ecuación (2), se encontró que el panel solar con estas características es suficiente para suministrar de energía al nodo meteorológico y ser completamente autosuficiente.

Figura 18

Pruebas del panel solar en el nodo



Cálculo de batería

Para asegurar un funcionamiento continuo de 24 horas, la EMA estuvo equipada con una batería que siempre es recargada mediante el panel solar. Según la ecuación (3), y considerando un 50% de descarga, se requirieron 4,046 Ah para mantener la autosustentabilidad del sistema. Al evaluar baterías en el mercado con capacidades de 12 V y 7 Ah, y aplicando la ecuación (4), se determinó que una sola batería fue suficiente.

Cálculo de controlador de carga

Al haber utilizado paneles solares, se necesitó un controlador para redirigir la carga al mainboard y la batería. Según la ecuación (5), se requirió un mínimo de 0,57 A. Dado que los controladores comerciales empezaban con 10 A, se optó por una de esas especificaciones.

Cálculo del cableado

Se aplicó la ecuación (6) para dimensionar el cableado, obteniendo una intensidad de 0,89 A del panel al controlador. Mediante la ecuación (7), se determinó una sección de 0,204 mm^2, que se identificó como correspondiente a un cable 24 AWG. En el Apéndice J se mostró

un case IP65 que contiene la batería de 12 V a 7 A, un panel solar, la mainboard, y dos breakers de 2 A para evitar corrientes altas que dañarían los circuitos.

3.2. Solución sistema IoT con los sensores meteorológicos

La presente sección enfoca los resultados de la vinculación de los sensores meteorológicos con la placa de desarrollo, así como el análisis de la trama, las pruebas del pluviómetro para la evaluación de precisión. Por último, se aborda la interconexión con el módulo de comunicación.

3.2.1. Almacenamiento y placa de desarrollo

El peso de cada trama fue de 76 bytes por minuto, conociendo este dato se procedió a calcular el tiempo requerido para llenar una MicroSD. Su propósito es guardar todos los datos censados por la EMA. Se hizo proyecciones con distintas tarjetas MicroSD, esto se visualiza en la Tabla 5. Por tal motivo se optó por escoger el almacenamiento de 32GB, optimizando costos y recursos de hasta 110 años de uso.

El Apéndice K muestra el diseño de la placa de desarrollo y sus módulos ensamblados, con resultados visualizables en una pantalla LCD sirviendo para validar los datos en la MicroSD. Además, se analizó el formato de la trama: Fecha, hora, longitud, latitud, temperatura, humedad, radiación solar, pulso del pluviómetro, voltaje y corriente respectivamente, visualizado en la Figura 19. El código se detalla en el Apéndice M.

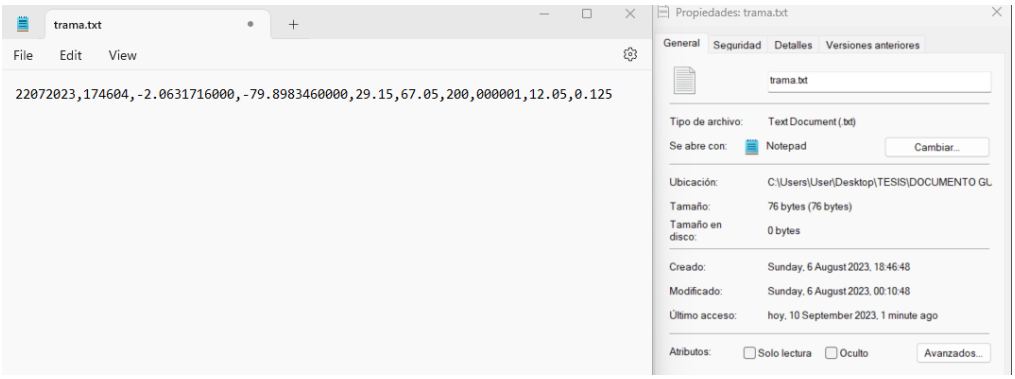
Tabla 5

Tiempo a completar distintos almacenamientos en la ranura SD

Capacidad	Tiempo
32 GB	110 años, 221 días, 16 horas, 0 minutos, 0 segundos
64 GB	216 años, 153 días, 4 horas, 0 minutos, 0 segundos
128 GB	433 años, 51 días, 8 horas, 0 minutos, 0 segundos

Figura 19

Trama de datos meteorológicos en placa de desarrollo



3.2.2. Pruebas del pluviómetro

Se evaluó el pluviómetro debido a sus años previos de uso, mediante 20 pruebas de precisión usando un sistema de goteo de agua para conocer el volumen necesario que genera un volcamiento, obteniendo un promedio de 4,46 ml. Conforme a las especificaciones del fabricante, dictamina que 4,73ml de agua genera un volcamiento, respecto a esta comparativa se calculó el porcentaje de error promedio, obteniendo 5,71%, detallado en la Tabla 6.

Tabla 6

Resultados de las pruebas del pluviómetro

Experimento	Volumen de agua (ml)	Porcentaje de error (%)
1	4,48	5,28
2	4,64	1,90
3	4,56	3,59
4	4,64	1,90
5	4,32	8,67
6	4,24	10,36
7	4,50	4,86
8	4,50	4,86
9	4,40	6,98
10	4,41	6,77
11	4,48	5,28
12	4,40	6,98
13	4,43	6,34
14	4,54	4,01
15	4,48	5,28
16	4,32	8,67
17	4,48	5,28
18	4,56	3,59

19	4,48	5,28
20	4,40	6,98
Promedio	4,46	5,71

3.2.3. Módulo de comunicación

En el Apéndice N se presenta el código para la conexión entre la placa de desarrollo y el módulo RAK11300. Utilizando pines UART_TX y UART_RX entre ambas placas, necesarios para enlazar con el puerto Serial2 de la mainboard, como se muestra en la Figura 20, se especifica el apartado inicial del código donde se definieron estos parámetros. Parte de los parámetros OTAA para la conectividad con el servidor de TSS de LoRaWAN se especifican en la Figura 21.

Los parámetros LoRa, incluida la frecuencia US915 y la conversión de datos a hexadecimal comprimieron la información de 76 bytes a 48 bytes.

Figura 20

Parámetros iniciales del módulo de comunicación

```
bool doOTAA = true;   // OTAA is used by default.
#define SCHED_MAX_EVENT_DATA_SIZE APP_TIMER_SCHED_EVENT_DATA_SIZE /**< Maximum size of scheduler events. */
#define SCHED_QUEUE_SIZE 60                     /**< Maximum number of events in the scheduler queue. */
#define LORAWAN_DATERATE DR_2                   /*LoRaMac datarates definition, from DR_0 to DR_5*/
#define LORAWAN_TX_POWER TX_POWER_5             /*LoRaMac tx power definition, from TX_POWER_0 to TX_POWER_15*/
#define JOINREQ_NBTRIALS 3                      /**< Number of trials for the join request. */
DeviceClass_t g_CurrentClass = CLASS_C;         /* class definition*/
LoRaMacRegion_t g_CurrentRegion = LORAMAC_REGION_US915;    /* Region:EU868*/
lmh_confirm g_CurrentConfirm = LMH_UNCONFIRMED_MSG;         /* confirm/unconfirm packet definition*/
uint8_t gAppPort = LORAWAN_APP_PORT;                      /* data port*/

/**@brief Structure containing LoRaWan parameters, needed for lmh_init()
*/
static lmh_param_t g_lora_param_init = {LORAWAN_ADR_ON, LORAWAN_DATERATE, LORAWAN_PUBLIC_NETWORK, JOINREQ_NBTRIALS, LORAWAN_TX_POWER, LORAWAN_DUTYCYCLE_OFF};

// Foward declaration
static void lorawan_has_joined_handler(void);
static void lorawan_join_failed_handler(void);
static void lorawan_rx_handler(lmh_app_data_t *app_data);
static void lorawan_confirm_class_handler(DeviceClass_t Class);
static void send_lora_frame(void);
void lorawan_unconf_finished(void);
```

Figura 21

Credenciales de identificación entre servidor y módulo RAK11300

```
//OTAA keys !!!! KEYS ARE MSB !!!!
uint8_t nodeDeviceEUI[8] = {0xAC, 0x1F, 0x09, 0xFF, 0xFE, 0x06, 0xB8, 0x86};  //AC1F09FFFE06B886
uint8_t nodeAppEUI[8] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
uint8_t nodeAppKey[16] = {0x6E, 0x72, 0xF7, 0x44, 0xB3, 0xAD, 0x90, 0xAA, 0x86, 0x92, 0x11, 0x14, 0xE3, 0x11, 0x66, 0xC4};//6E72F744B3AD90AA86921114E31166C4

// ABP keys
uint32_t nodeDevAddr = 0x260116F8;
uint8_t nodeNwsKey[16] = {0x7E, 0xAC, 0xE2, 0x55, 0xB8, 0xA5, 0xE2, 0x69, 0x91, 0x51, 0x96, 0x06, 0x47, 0x56, 0x9D, 0x23};
uint8_t nodeAppsKey[16] = {0xFB, 0xAC, 0xB6, 0x47, 0xF3, 0x58, 0x45, 0xC7, 0x50, 0x7D, 0xBF, 0x16, 0x8B, 0xA8, 0xC1, 0x7C};
```

3.3. Solución de recepción e integración de datos meteorológicos

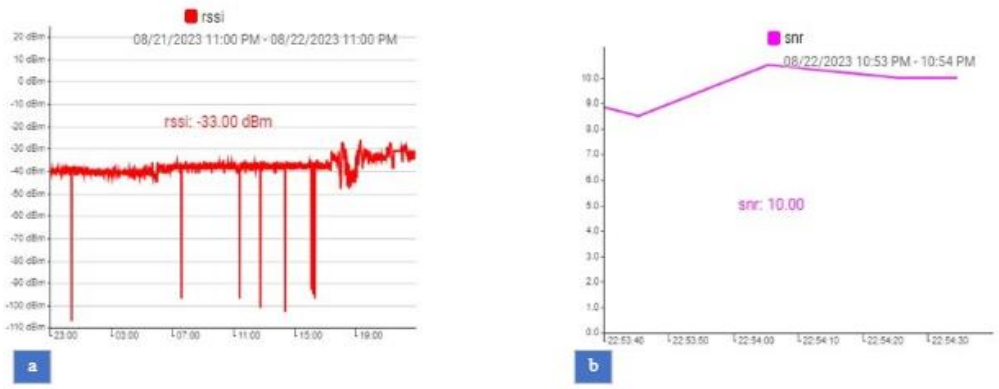
Los resultados de las pruebas relacionadas con la capa de transporte, específicamente entre el nodo y el Gateway, así como los resultados de la capa de aplicación producto de la integración del servidor TTS con Webhooks y AWSIoT, se detallan en la siguiente sección.

3.3.1. Pruebas de recepción

Las pruebas de recepción se realizaron mediante los datos enviados desde el módulo LoRaWAN hacia el servidor de TTS, las cuales se visualizan en la Figura 22.

Figura 22

Análisis de señal recibida



Nota. a) Potencia de recepción. b) Potencia de relación señal-ruido.

El módulo de comunicación se programó para enviar parámetros como potencia de recepción (RSSI) y relación señal ruido (SNR) sin la necesidad de codificar. Se observaron estos datos en TTS, y AnyViz, destacando valores medios de –33 dBm en RSSI y 10 dB para el SNR. Se generó un código en TTS en formato JSON cada vez que llegaban datos del módulo de comunicación (Apéndice O). Con una frecuencia de operación de 90,45 MHz y un SF de 7, se priorizó la velocidad de transmisión en vez de la resistencia de señal en el ambiente, en un entorno de prueba controlado a 3 metros de distancia entre el Gateway y el nodo. El tiempo de

vuelo fue de 0,066816 segundos, justificando el SF. Además, una tasa de codificación de 4/5 proporcionó un buen equilibrio entre eficiencia y resiliencia.

3.3.2. Pruebas de integración.

En la presente sección, se describen con más detalle las configuraciones realizadas para la obtención de datos en plataformas que utilizan servicios de Webhooks y AWSIoT. Para ello, se llevaron a cabo configuraciones desde el lado de TTS hasta los servicios de integración. En este contexto, se empleó AnyViz para Webhooks y S3 Budget para AWSIoT.

The Things Stack (TTS)

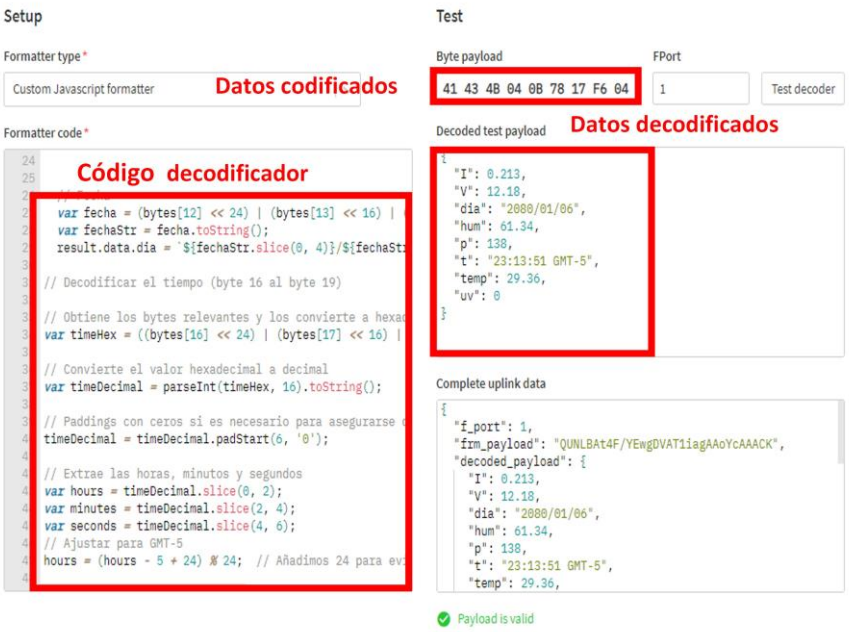
El end device, conocido como 'espol-tesis-pluviometro', se registró previamente en la aplicación 'Prometeo Energy' en el servidor del cliente 'prometeolab' el cual se evidencia en la dirección del navegador, visualizado en la Figura 23. Los datos llegaron en tramas de 48 bytes de forma inalámbricas cada minuto como una señal de subida (Uplink) con el registro de la hora de llegada del último dato. Se evidencia los datos decodificados como los parámetros meteorológicos de la temperatura, humedad, pulsos pluviómetro y radiación UV. También abarcó lo valores de corriente, voltaje, fecha y hora, conforme a las especificaciones del cliente. La decodificación es detallada en la Figura 24, donde se decodifica una Payload aleatoria que confirma los datos antes mencionados. El código decodificador se incluyó en el Apéndice P.

Figura 23

Trama recibida en TTS

Figura 24

Código de decodificación de Payload



Webhooks-AnyViz

Los datos decodificados de la Payload se enviaron a AnyViz mediante Webhooks, utilizando un Proyect ID asignado al usuario. Esta integración permite la visualización de datos en tiempo real para el cliente, como se evidencia en la Figura 25. La plataforma final muestra los parámetros: temperatura, humedad y promedio de radiación UV, además de la ubicación de la EMA. Los parámetros de calidad de señal también se presentaron en AnyViz, como se detalló previamente en la Figura 22.

Figura 25

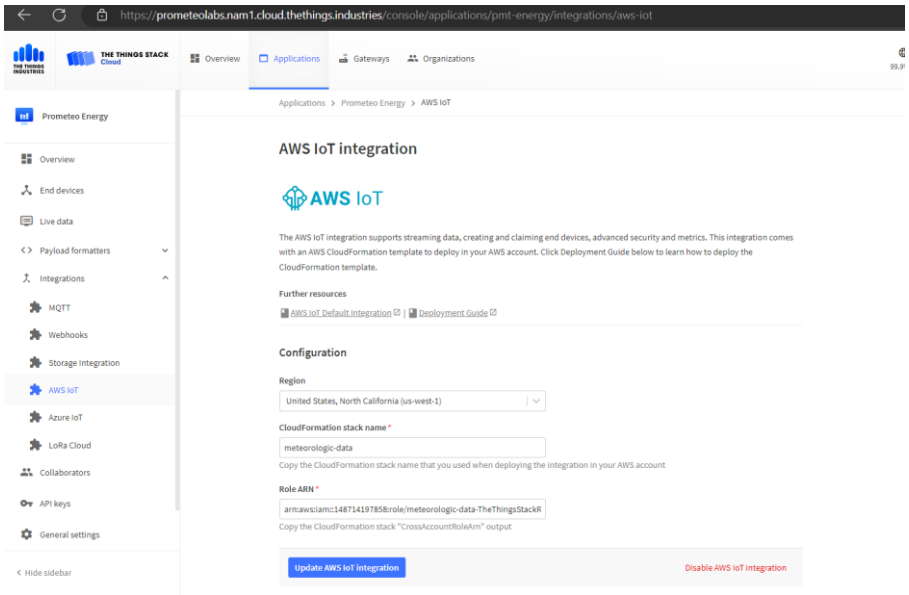
Plataforma final del Webhooks en AnyViz

AWSIoT-S3 Budget

El servidor de operaciones, denominado ‘meteorologic-data’, se registró en AWSIoT en el norte de california. La arquitectura del diseño en AWS se almacena en un Stack de Cloud Formation, como se muestra en la Figura 26, donde se llenaron campos de autentificación entre ambas plataformas.

Figura 26

Configuración de la integración por AWSIoT

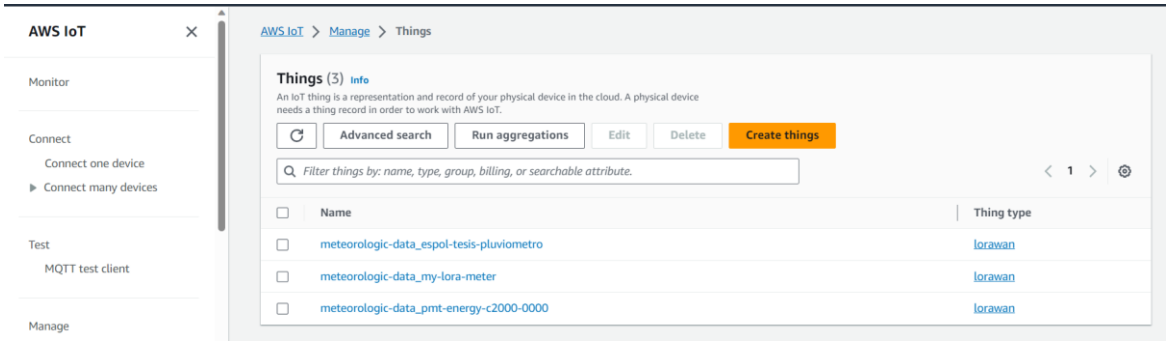


IoTCore utilizó información de Cloud Formation para configurar de manera específica el end device ‘meteorologic-data_espol-tesis-pluviometro’. A través de filtros y acciones

diseñadas, se implementa una regla S3 que permite el almacenamiento en tiempo real de los datos recibidos desde TTS, como se muestra en la Figura 27.

Figura 27

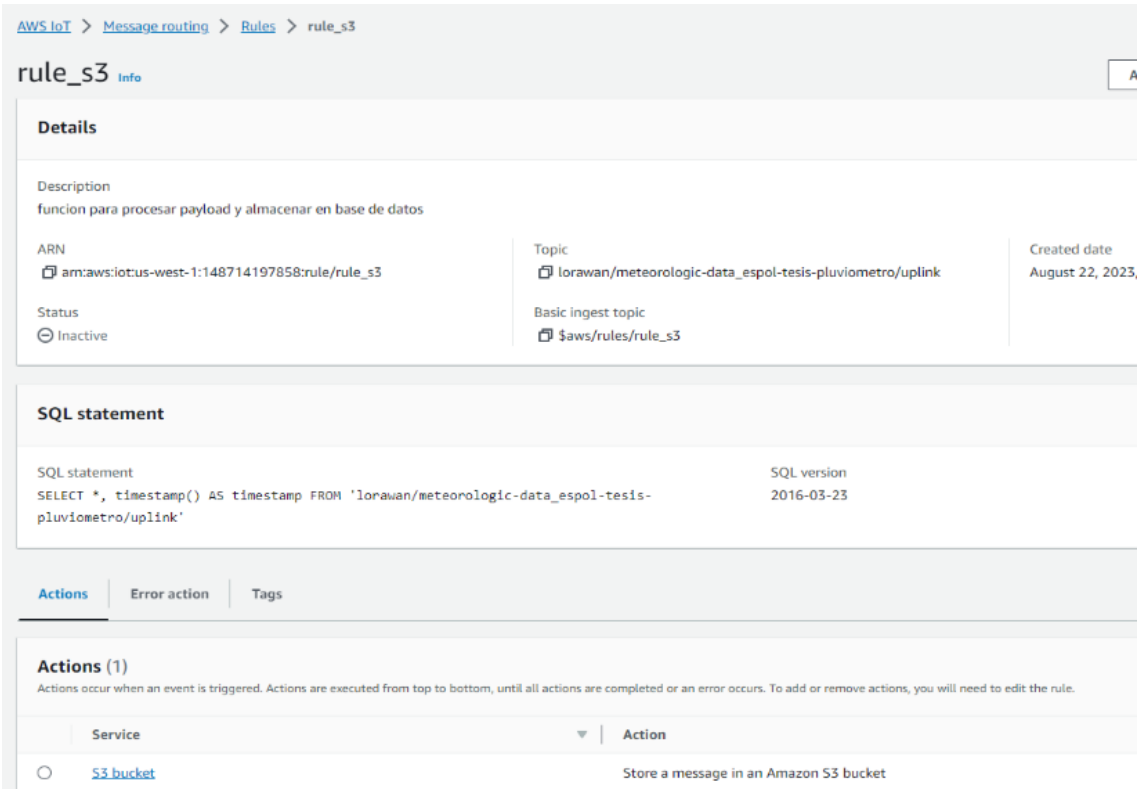
Configuración de la integración por AWSIoT



La regla S3 para el end device seleccionado es detallada en la Figura 28. En este contexto, se filtraron los datos mediante SQL para almacenar exclusivamente el código proveniente de TTS, permitiendo así un análisis focalizado de la información inalámbrica.

Figura 28

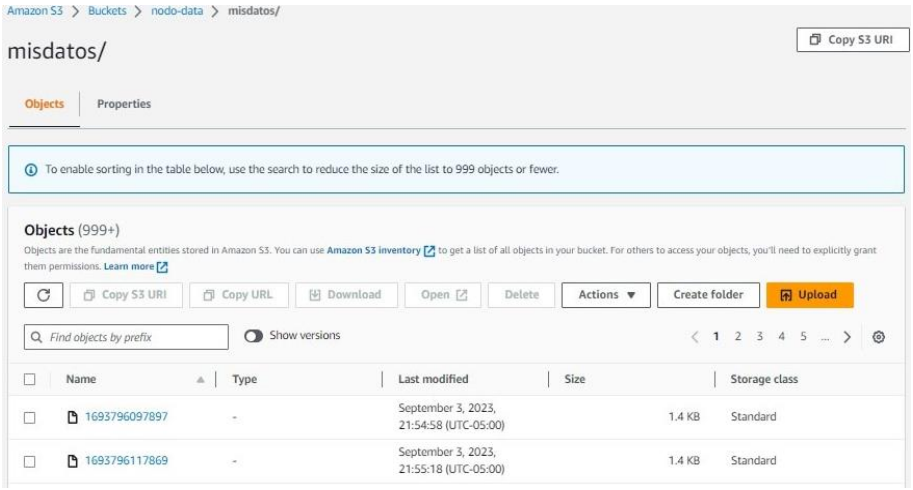
Regla S3



La Figura 29 muestra los elementos almacenados en el servidor S3 en el Bucket ‘misdatos’, que corresponden a los datos recopilados del nodo y se actualizan en intervalos de un minuto, cada dato guardado tiene un peso de 1,4 KB debido a que se almacena todo el código mostrado en el Apéndice O.

Figura 29

Datos guardados



3.4. Validación de datos reales

Para evaluar la precisión de la EMA, se compararon sus datos con los proporcionados por AccuWeather. A lo largo de dos días y en distintos horarios, se efectuaron ocho pruebas enfocadas en sensores de temperatura, humedad y radiación ultravioleta a excepción de la precipitación de lluvia por la carencia de lluvia en el territorio de implementación. La Tabla 7 muestra los resultados específicos para el sensor de temperatura, revelando un porcentaje promedio de error del 3,01%.

Tabla 7

Comparación de datos de temperatura

Fecha y Hora	Temperatura estación automática (°C)	Temperatura(www.accuweather.com) (°C)	Porcentaje de error (%)
14/08/23 - 23:00	29,86	28	6,64

14/08/23 - 14:00	29,84	31	3,74
14/08/23 - 19:00	25,95	25	3,80
15/08/23 - 00:00	25,25	25	1,00
15/08/23 - 07:00	24,46	24	1,92
15/08/23 - 10:00	29,76	31	4,00
15/08/23 - 00:30	32,86	32	2,69
15/08/23 - 15:45	30,91	31	0,29
Promedio	24,88	28,38	3,01

La Tabla 8 proporciona datos de humedad tomadas en ocho distintas instancias. Cada fila en la tabla es un momento específico de medición, donde el error porcentual promedio se mantuvo en 2,61%, debido a que la temperatura y humedad se registran por el mismo sensor SHT10 ambos parámetros contaron con cifras significativas que aportaron en el aumento del porcentaje de error incluso en datos de cifra decimal tal como se evidencia en la temperatura registrada el día 15 de agosto del 2023 a las 7:00 donde la temperatura del sensor fue de 24,46 °C y la de AccuWeather fue de 24 °C el cual no incluye cifras significativas aportando así un error de 1,92%.

Tabla 8

Comparación de datos de humedad

Fecha y Hora	Humedad estación meteorológica (%)	Humedad (www.accuweather.com) (%)	Porcentaje de error (%)
14/08/23 - 23:00	62,5	60	4,17
14/08/23 - 14:00	68,44	70	2,23
14/08/23 - 19:00	84,45	86	1,80
15/08/23 - 00:00	87,4	92	5
15/08/23 - 07:00	88,71	89	0,33
15/08/23 - 10:00	69	71	2,82
15/08/23 - 00:30	65,41	67	2,37
15/08/23 - 15:45	62,32	61	2,16
Promedio	73,53	74,50	2,61

La Tabla 9 exhibe el índice de radiación solar con un porcentaje de error del 0%. Sugiere que, al tratarse de una medición que da un valor entero, se prescindió de cualquier fracción decimal o centesimal en las lecturas para cada estación meteorológica.

Tabla 9

Sensor de radiación solar ML8511

Fecha y Hora	Niveles de radiación solar (Estación meteorológica)	Nivel de radiación (www.accuweather.com)	Porcentaje de error (%)
14/08/23 - 11:00	0	0	0
14/08/23 - 14:00	3	3	0
14/08/23 - 19:00	0	0	0
15/08/23 - 00:00	0	0	0
15/08/23 - 07:00	2	2	0
15/08/23 - 10:00	2	2	0
15/08/23 - 12:30	3	3	0
15/08/23 - 15:45	4	4	0
Promedio	1,75	1,75	0

3.5. Costos

En la Tabla 10 presenta un análisis detallado del costo total estimado invertido en la implementación de la EMA, el cual fue de $6236,88 dólares americanos. Es crucial subrayar que este monto incluye el valor de la placa de desarrollo, un componente que tiene costo de $5000 dólares americanos debido que se encuentra licenciada por el cliente. Adicionalmente, el pluviómetro fue el segundo componente más costoso, con un valor de $800 dólares americanos. Este equipo fue prestado por la institución para el desarrollo del presente proyecto. En este presupuesto, también se deben considerar los costos postpagos mensuales por los servicios en AWS, los cuales abarcan seis funciones. El total facturado por el mes en el que se realizaron las pruebas fue de $0,35 dólares americanos. Este costo podría variar, ya sea aumentando o disminuyendo, en función de la frecuencia de uso de las funciones. El precio actual justificó el almacenamiento de datos de 1,4 KB en el Bucket denominado 'misdatos'.

Tabla 10

Costos detallados para la implementación de la estación meteorológica.

Ítem	Cantidad	Precio Unitario ($)	Precio Total ($)
Pluviómetro	1	800	800
Sensor de Temperatura y Humedad	1	45	45
Sensor de Radiación Ultravioleta	1	14	14
Módulos: RAK 11300, Wisblock RAK11300, PCB entrada analógica, entrada digital,	1	250	250
Placa de Desarrollo incluido GPS, ranura MicroSD, módulo Ethernet,	1	5000	5000
Tarjeta microSD 32GB	1	20	20
Gateway LoRaWAN	1	265	265
Panel Solar	1	20,16	20,16
Breaker	2	2,4	4,8
Batería	1	17,37	17,37
Cable Nº 24AWG (6 metros)	6	0,9	5,4
Cable Nº 18AWG (1 metro)	1	0,9	0,9
Case IP65	1	32,3	32,3
Tubo Termocontraíble (1 metro)	1	0,7	0,7
Regulador de Voltaje-Corriente	1	12,3	12,3
Prensa Stopped de 3/4	1	2,5	2,5
Servicios de AWS (mensual): Cloudwatch, IoT, Simple Storage Service, Data Trandfer, Lambda, Secrets Manager.	6	NA	0,35
Total			6236,88

Capítulo 4

4. Conclusiones y recomendaciones

4.1. Conclusiones

- Este estudio ha logrado demostrar la viabilidad y eficiencia de la estación meteorológica implementada con tecnología IoT en sus datos meteorológicos obtenidos por los sensores seleccionados y probados: SHT10 para temperatura y humedad, y ML8511 para el sensor de radiación Ultravioleta, de acuerdo a los estándares de la Organización Meteorológica Mundial (OMM): ±0,5°C de precisión para el sensor de temperatura, el cual coincide con los ± 0,5 °C del estándar; 4,5% de error para la humedad, el cual está por debajo de los 5% del estándar; y 10% de error para el sensor de radiación ultravioleta, el cual coincide con el 10% de error que exige la organización. Con este cumplimiento del estándar, en la comparativa de datos con el portal meteorológico AccuWeather se obtuvo porcentajes de error del 3,01% para temperatura, 2,61% para la humedad y 0% para la radiación solar.
- Se logró reemplazar el datalogger del pluviómetro TR-525M con programación de interpretación del volcamiento mediante pulsos eléctricos en la placa de desarrollo. Sin embargo, es relevante señalar que la falta de lluvia durante el período de estudio limitó la validación completa de esta función. No obstante, se realizaron pruebas de simulación en la capacidad de almacenamiento por volcamiento, dando un promedio de 4,46 ml, en comparación con los 4,73 ml establecidos por la ficha técnica, con un porcentaje de error del 5,71%.
- Se obtuvo un sistema IoT materializado a través de la integración de la placa de desarrollo y el módulo de comunicación RAK11300, Esta unión se realizó utilizando una interfaz UART, lo cual simplificó la programación al permitir un único entorno de software para ambos componentes y los sensores asociados el cual fue Arduino IDE. Esta configuración habilitó diversas funcionalidades: la placa de desarrollo se encargó

de recolectar y almacenar datos provenientes de tres diferentes sensores en una tarjeta MicroSD de 32 GB, proporcionando una solución de respaldo de datos en caso de fallas en la transmisión dando un estimado de hasta 110 años continuo de guardado. Por su parte, el módulo de comunicación tuvo la función de recibir los datos de los tres sensores a través del puerto serial, formando así un nodo sensor unificado apto para la comunicación inalámbrica.

- Se logró trasmitir información de 76 bytes cada minuto a través del canal inalámbrico con una frecuencia de 90,45 MHz, utilizando tecnología LoRaWAN en la estación meteorológica. Los parámetros de calidad de señal mostraron una potencia de recepción de -33 dBm, relación señal ruido de 10 dB y Spread factor de 7. Respaldando una eficiencia en la transmisión de datos con tiempo de vuelo de 0,066816 segundos en el entorno de pruebas, donde las condiciones para la transmisión fueron con línea de vista a 3 metros de distancia entre el Gateway y el nodo meteorológico. Validando así el uso de la tecnología LoRaWAN como una solución LPWAN para la transmisión de pequeños datos a largas distancias con un bajo consumo de energía, la cual para este tipo de aplicación fue de 36,42 Wh.
- Se logró desarrollar una plataforma interactiva para la visualización de datos meteorológicos destinada a usuarios finales. Esta se diseñó utilizando AnyViz y se integró con el servidor The Things Network mediante Webhooks. Además, se implementó una integración con AWSIoT, lo cual permitió el almacenamiento del historial de datos en un Bucket de S3.

4.2. Recomendaciones

- La escalabilidad para incorporar más sensores meteorológicos es factible, gracias a la disponibilidad de 3 entradas I2C, una entrada analógica y una entrada digital en el sistema. No obstante, se aconseja adicionar un segundo panel solar con las mismas

especificaciones y estar conectado en serie con el otro panel solar. De esta forma se conserva la misma corriente para todo el sistema con el aumento de tensión para satisfacer las nuevas demandas energéticas de la estación meteorológica.

- Debido a las especificaciones técnicas que indican instalar las estaciones meteorológicas en espacios abiertos sin obstrucciones, brindan una gran ventaja en la transmisión de datos meteorológicos de forma inalámbrica debido a que el medio se moldea a un modelo de programación de espacios libres incluso con cortos obstáculos. Sin embargo, se recomienda realizar esta experimentación en lugares rurales donde puedan existir obstrucciones en el trayecto de radio frecuencia para determinar si la carga útil de este proyecto pueda ser enviada con los mismos componentes en aquellos entornos.
- Se recomienda hacer pruebas en época invernal para analizar el desempeño del sensor de pluviómetro para ambientes reales.

Referencias

[1] Banco Mundial, «Agricultura, valor agregado (% del PIB) - Ecuador,» 2021. [En línea]. Available: https://datos.bancomundial.org/indicador/NV.AGR.TOTL.Zs?end=2021&locations=EC&start=1960&view=chart. [Último acceso: 16 junio 2023].

[2] Diario Primicias, «La inversión en el sector agrícola representa una apuesta para el futuro,» 18 marzo 2022. [En línea]. Available: https://www.primicias.ec/noticias/patrocinado/la-inversion-en-el-sector-agricola-representa-una-apuesta-para-el-futuro/. [Último acceso: 17 junio 2023].

[3] J. Fiallo, «Importancia del Sector Agrícola en una Economía Dolarizada,» Trabajo de titulación, Universidad San Francisco de Quito, 2023.

[4] Picón y Aristega, «Aplicación de Bases de Datos en la Meteorología Agrícola para el Cultivo de Banano,» Tesis de grado, Universidad Estatal de Milagro, 2019.

[5] Instituto Nacional de Meteorología e Hidrología, «Anuario Meteorológico 2013,» Quito, 2017.

[6] World Meteorological Organization, «Guide to Agricultural Meteorological Practices (GAMP),» World Meteorological Organization , Geneva, 2018.

[7] C. Loyola y H. Salazar, «Correlación estadística de estaciones meteorológicas convencionales y automáticas durante el periodo 2014-2021.,» Universidad Técnica de Cotopaxi (UTC), Latacunga, 2022.

[8] Bravo y Rosas, «Análisis y diseño de un modelo de gestión de la información hidrometeorológica para los procesos desconcentrados del Instituto Nacional de

Meteorología e Hidrología - INAMHI.,» Tesis de pregrado, Facultad de Ingeniería, Ciencias Físicas y Matemáticas, Universidad Central del Ecuador, 2015.

[9] Balladares y López, «Diseño e implementación de un prototipo de estación meteorológica agrícola autosustentable para el monitoreo de parámetros ambientales en cultivos de cacao mediante Raspberry PI,» Tesis de grado, Universidad Politécnica Salesiana, 2021.

[10] Nelson, Rosegrant, Koo, Robertson, Sulser, Zhu, Claudia, Msangi, Palazzo, Batka, Magalhaes, Valmonte, Ewing y Lee, «Cambio climático. El impacto en la agricultura y los costos de adaptación,» International Food Policy Research Institute, 2009.

[11] Instituto Nacional de Meteorología e Hidrología, «Red de estaciones automáticas hidrometereológicas. Condiciones actuales del tiempo para las últimas 24 horas,» junio 2023. [En línea]. Available: http://186.42.174.236/InamhiEmas/.

[12] López, López, Banguela y Suárez, «Principales análisis para la toma de decisiones en las producciones agropecuarias,» *Revista Universidad y Sociedad*, vol. 13, nº 3, pp. 233-242, 2021.

[13] Viguera, Martínez, Donatti, Harvey y Alpizar, «Impactos del cambio climático en la agricultura de Centroamérica, estrategias de mitigación y adaptación,» Proyecto CASCADA. Conservación Internacional. Centro Agrónomo Tropical de Investigación y Enseñanza (CATIE), Costa Rica, 2017.

[14] J. Darquea, «Diseño y construcción de una estación meteorológica autónoma e inalámbrica basada en hardware embebido que soporte entornos gráficos de programación,» Trabajo de Titulación, Facultad de Informática y Electrónica. Escuela Superior Politécnica de Chimborazo, 2020.

[15] Darrera, «Estación Meteorológica Automática (EMA),» 2023. [En línea]. Available: https://www.darrera.com/wp/es/producto/3r-aws100-estacion-meteorologica-automatica-ema/. [Último acceso: 2 junio 2023].

[16] Météo Shopping, «Estación meteorológica portátil Skywatch BL-1000,» 2023. [En línea]. Available: https://www.meteo-shopping.com/es/anemometro/672-estacion-meteorologica-portatil-skywatch-bluetooth.html. [Último acceso: 13 agosto 2023].

[17] Guías Prácticas, «Estaciones meteorológicas analógicas,» 2020. [En línea]. Available: https://www.guiaspracticas.com/estaciones-meteorologicas/estacion-meteorologica-analogica.

[18] TekoMeteo, 2020. [En línea]. Available: https://tekmeteo.com/estacion-meteorologica-analogica#:~:text=Las%20estaciones%20meteorol%C3%B3gicas%20anal%C3%B3gicas%20sirven,la%20madera%20o%20el%20acero.

[19] Meteobenidorm, «El tiempo en benidorm,» 25 octubre 2019. [En línea]. Available: http://meteobenidorm.blogspot.com/2019/10/tiempo-analogico.html.

[20] Khomp, «Estación Meteorológica LoRa,» 2023. [En línea]. Available: https://www.khomp.com/iot/es/produto/estacion-meteorologica/.

[21] Tecno Industry, 2023. [En línea]. Available: https://tecnoindustry.com/productos/estacion_meteorologica_wifi/.

[22] G. Chacón, «Equipos y sensores de la red de agrometeorología INIA,» Boletín Inia No. 415, 2023.

[23] S. Parent, «¿Cómo influye la humedad en la calidad de los cultivos?,» 27 junio 2023. [En línea]. Available: https://www.pthorticulture.com/es/centro-de-formacion/como-influye-la-humedad-en-la-calidad-de-los-cultivos/.

[24] Agro Tecnología Tropical, «¿Cual es la Importancia de medir las precipitaciones?,» 2023. [En línea]. Available: https://www.agro-tecnologia-tropical.com/Importancia_precipitacion.php#:~:text=En%20pocas%20palabras%20las%20precipitaciones,necesidades%20h%C3%ADdricas%20de%20las%20plantas..

[25] Díaz, «Efecto de la Radiación en el Desarrollo Fenológico, Rendimiento y Calidad en Policultivo: Chile, Jitomate, Maíz, Frijol y Amaranto en condiciones de Invernadero.,» Universidad de Querétaro. Tesis MC. Querétaro, 2012.

[26] Arduino, «Arduino MEGA 2560 REV3,» 2023. [En línea]. Available: https://docs.arduino.cc/hardware/mega-2560. [Último acceso: 23 agosto 2023].

[27] Rakwireless, «RAK11300 WisDuo LPWAN Module,» 2023. [En línea]. Available: https://docs.rakwireless.com/Product-Categories/WisDuo/RAK11300-Module/Overview/?_gl=1%2a8czuoh%2a_ga%2aMTQ1NDIxMTE3OS4xNjg3OTg2NDA2%2a_ga_9MEJ39NSTZ%2aMTY5MjAxNzEzMi43LjEuMTY5MjAyMDc1Mi42MC4wLjA. [Último acceso: 23 agosto 2023].

[28] M. Fernández, «Redes de datos. Protocolo TCP/IP,» Grado en Gestión y Administración Pública, Facultad de Ciencias Sociales y de la Comunicación, Universidad de Cádiz, 2020.

[29] World Meteorological Organization, «Guide to Agricultural Meteorologicas Practices,» WNO-No.134, 2010.

[30] BricoGeek, «Sensor de temperatura y humedad SHT10 (Acero Inox),» 2023. [En línea]. Available: https://tienda.bricogeek.com/sensores-temperatura/762-sensor-de-temperatura-y-humedad-sht10-acero-inox.html.

[31] Naylamp, «Módulo de sensor de luz ultravioleta (UV) ML8511,» 2023. [En línea]. Available: https://naylampmechatronics.com/sensores-luz-y-sonido/169-modulo-sensor-de-luz-ultravioleta-uv-ml8511.html. [Último acceso: 15 junio 2023].

[32] Ambimet, «Sensor de lluvia TR-525 M,» 2023. [En línea]. Available: https://www.ambimet-instrumentacion.cl/producto/sensor-de-lluvia-tr-525-m/. [Último acceso: 16 junio 2023].

[33] Instituto de Hidrología, Meteorología y Estudios Ambientales, «Guía para el emplazamiento de las estaciones meteorológicas,» Subdirección de Meteorología, 2021.

[34] Muhammad Afif Pratama1 y Tati Harihayati Mardzuki2, «DEVELOPMENT OF SMART APPLICATION FOR RANCAUPAS CAMPING GROUND USING THE API ACCUWEATHER AND GYROSCOPE SENSOR ON ANDROID SMARTPHONE».

[35] The Things Network, «Data API (MQTT),» 2023. [En línea]. Available: https://www.thethingsnetwork.org/docs/applications/mqtt/. [Último acceso: 23 agosto 2023].

[36] The Things Industries, «Webhooks,» 2023. [En línea]. Available: https://www.thethingsindustries.com/docs/integrations/webhooks/creating-webhooks/. [Último acceso: 23 agosto 2023].

[37] The Things Network, «AWS IoT,» 2023. [En línea]. Available: https://www.thethingsnetwork.org/docs/applications/aws/. [Último acceso: 23 agosto 2023].

[38] AWS Amazon, «AWS IoT Analytics,» 2023. [En línea]. Available: https://aws.amazon.com/es/iot-analytics/. [Último acceso: 23 agosto 2023].

[39] LoraWan, «Frequency Plans by Country,» 2023. [En línea]. Available: https://www.thethingsnetwork.org/docs/lorawan/frequencies-by-country/. [Último acceso: 29 agosto 2023].

[40] Agencia de Regulación y Control de las Telecomunicaciones, «Plan Nacional de Frecuencias,» Dirección Técnica de Regulación del Espectro Radioeléctrico, 2021.

[41] Ley Orgánica de Protección de la Privacidad y los Datos Personales, «Registro Oficial Suplemento 459,» 2021.

[42] Intedya, «ISO 27000 y el conjunto de estándares de Seguridad de la Información,» 2015. [En línea]. Available: https://www.intedya.com/internacional/757/noticia-iso-27000-y-el-conjuntode-estandares-de-seguridad-de-la-informacion.html#:~:text=ISO%2027000%20es%20un%20conjunto,la%20Seguridad%20de%20la%20Informaci%C3%B3n..

[43] Texas Electronics, «TR-525 Series Rainfall Sensors User's Manual,» Texas, 2015.

[44] AliExpress, «RAK Wireless Official Store,» 2023. [En línea]. Available: https://es.aliexpress.com/i/1005004219500366.html?gatewayAdapt=Msite2Pc.

[45] Mouser Electronics, «Laird Connectivity Sentrius RG1xx LoRa-Enabled Gateways,» 2023. [En línea]. Available: https://www.mouser.ec/new/laird-connectivity/laird-sentrius-rg1-lora-gateway/. [Último acceso: 23 agosto 2023].

[46] FunDino, «ML8511 UVA/UVB Sensor Module,» 2023. [En línea]. Available: https://funduinoshop.com/en/electronic-modules/sensors/light-color/805/ml8511-uva/uvb-sensor-module/.

[47] Licor, «Support: Biomet System (Sutron),» 2023. [En línea]. Available: https://www.licor.com/env/support/Biomet/topics/precipitation-gauge-tr525m.html.

Apéndices

Apéndice A

Característica de los sensores

Característica		SHT10	ML8511	TR-525M
Parámetro		Temperatura y humedad	Luz ultravioleta UV	Precipitación
Precisión		± 0.5 °C 0 ~ 100% RH, precisión 4.5%	15%-10%	1.0% hasta 50 mm por hora
Rango de medición		- 40 a + 125°C 0-100% RH	280nm – 390 nm	0.10 mm por vuelco (4.73mL)
Voltaje de operación	Mínimo	2.4V	3.3V	30VDC a2 A
	Máximo	5V	5.25V	115VAC a1 A
Consumo de corriente		80 µA (medida) 0.2 µA (standby)	300 µA	-
Consumo de energía		3mW	~0.65mW	-
Interfaz de comunicación		I2C	Análogo	Salida de pulso
Resistente a la intemperie		Sí	Requiere protección	Sí
Tiempo de respuesta		8 segundos	-	Cierre del interruptor (135ms) Rebote máximo tiempo de establecimiento (0.75ms)
Terminales		Verde = GND Amarillo = Reloj Azul = Señal	VIN (Voltaje), GND (Ground), OUT/EN (En producción/Habilitado)	Digital
Precio		$40,00	$20,00	$900,00

Nota: Característica de los sensores basados en el OMM [29] y Texas Electronics [43].

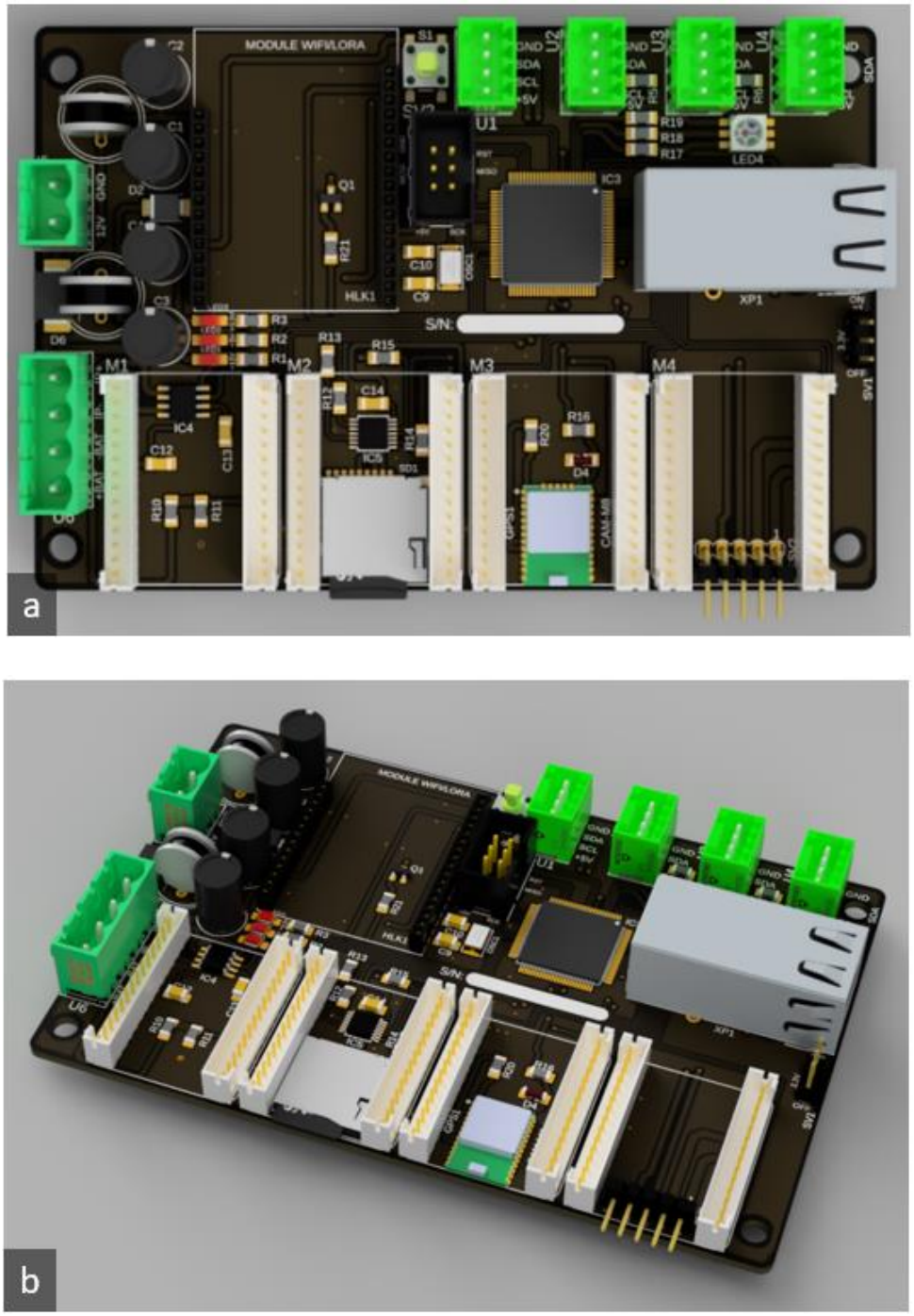
Apéndice B

Especificaciones técnicas de la placa de desarrollo

Especificación	Descripción
Fabricante	No aplica
Modelo	Mainboard para desarrollo
Microcontrolador	ATmega 2560
GPS	Quectel L96
Módulo TCP/IP	Lantronix XPort
Lector de memoria	MicroSD
Medidor de Corriente	Máximo 5Amp
Medidor de Voltaje	Máximo 12V
Módulo de comunicación inalámbrica	Slot x1 (Wifi, GSM, Radio, LoRaWAN o Bluetooth)
Módulos seriales	Slot x4
Puerto I2C	x4
Puerto serial	Para LOG o depuración
Indicador LED	RGB, controlado por PWM
Alimentación	12V

Apéndice C

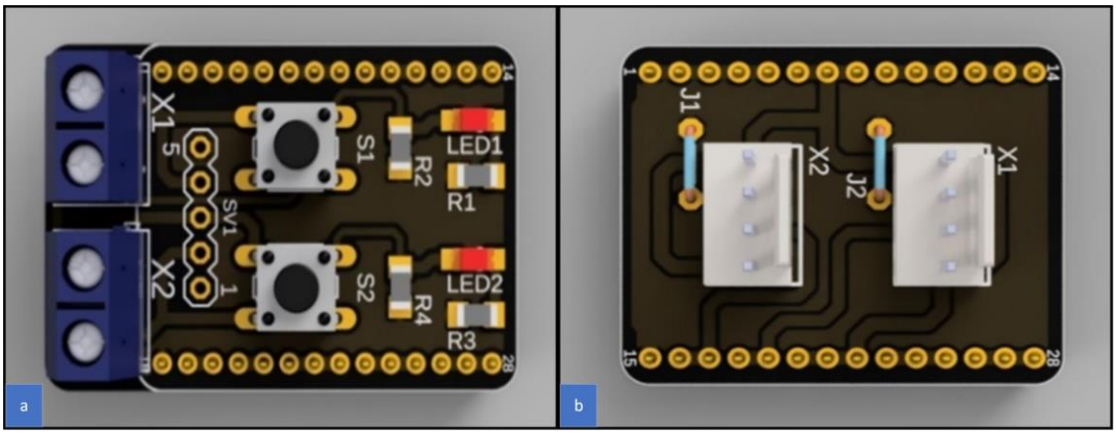
Placa de desarrollo



Nota. a) vista superior de la placa de desarrollo y b) vista lateral de la placa.

Apéndice D

Módulo de entradas digitales y analógicos



Nota. a) Módulo para sensores digitales. b) Módulo para sensores analógicos.

Apéndice E

Procesador de comunicación LoRaRAK11300

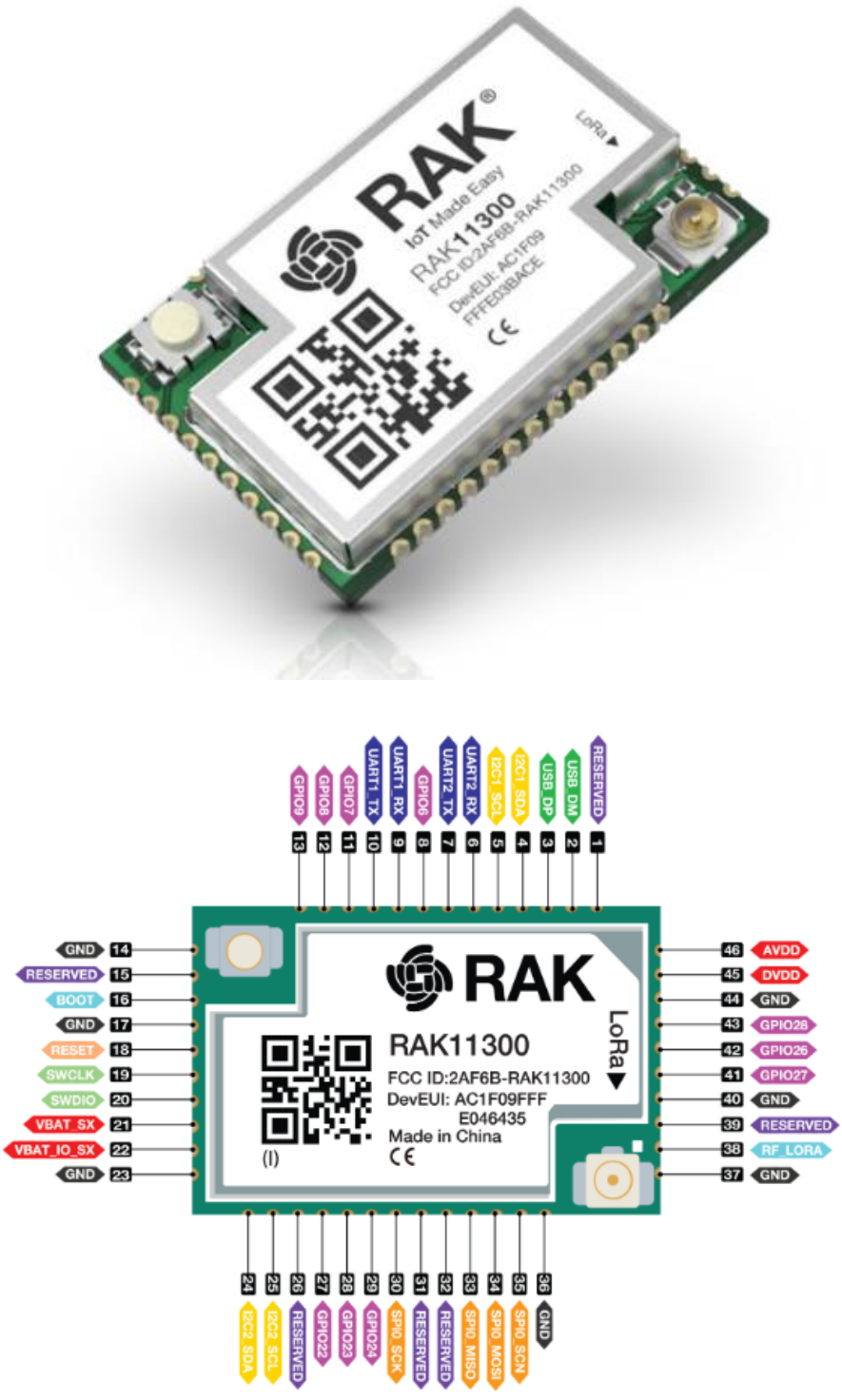
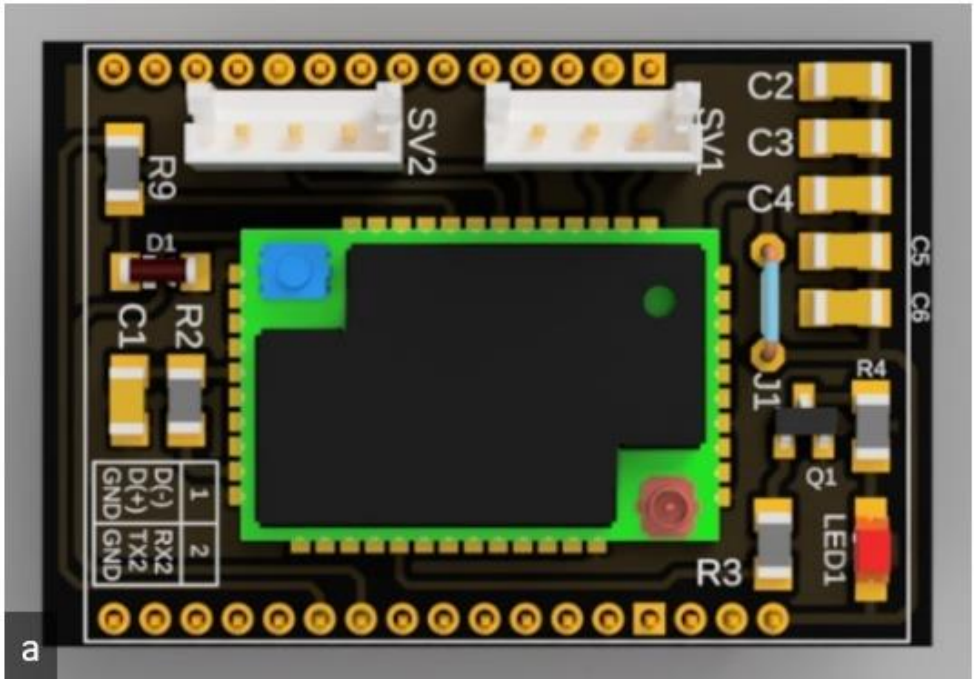


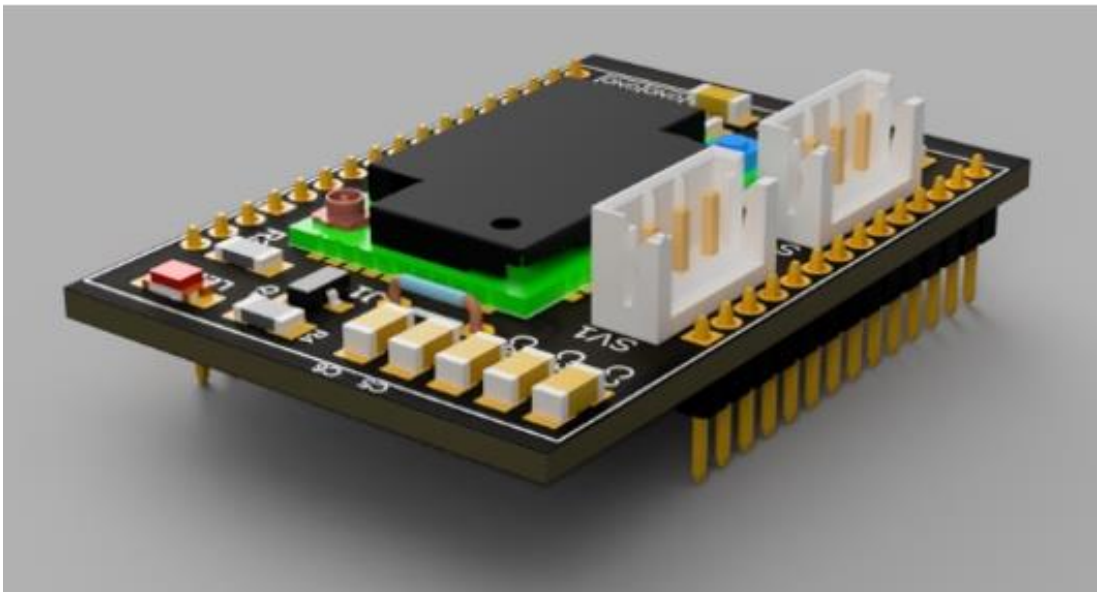
Figure 2: RAK11300 Module Pinout

Nota. Tomado de AliExpress [44].

Apéndice F

Módulo de comunicación LoRaWAN





Apéndice G

Especificaciones técnicas del módulo de comunicación

Característica	Especificacion
Tecnología de Comunicación	LoRaWAN
Procesador	Raspberry Pi RP2040 (ARM Cortex-M0+ Dual Core, 133Mhz)
Frecuencia	868/915 MHz (US915 para América)
Alcance	Hasta 15 km en áreas rurales, 5 km en áreas urbanas con antena optimizada
Transceptor LoRa	Semtech SX1262
Interfaz	UART, USB, I2C, GPIO
Programación	C/C++, MicroPython, CircuitPython (IDE de Arduino, PlatformIO, Thonny IDE)
Consumo de Energía	24,6 mA (TX), 3,8 mA (reposo)
Temperatura de Operación	-40°C a +85°C (75 °C con modulo WisBlock Core)
Voltaje de Operación	2,0V – 3,6V
Dimensiones	20 x 30 mm (con módulo WisBlock Core)
Compatibilidad con Arduino	Compatible con placas Arduino mediante módulos WisBlock Core
Activacion de red LoRaWAN	OTAA (Over-the-Air Activation) y ABP (Activation By Personalization)

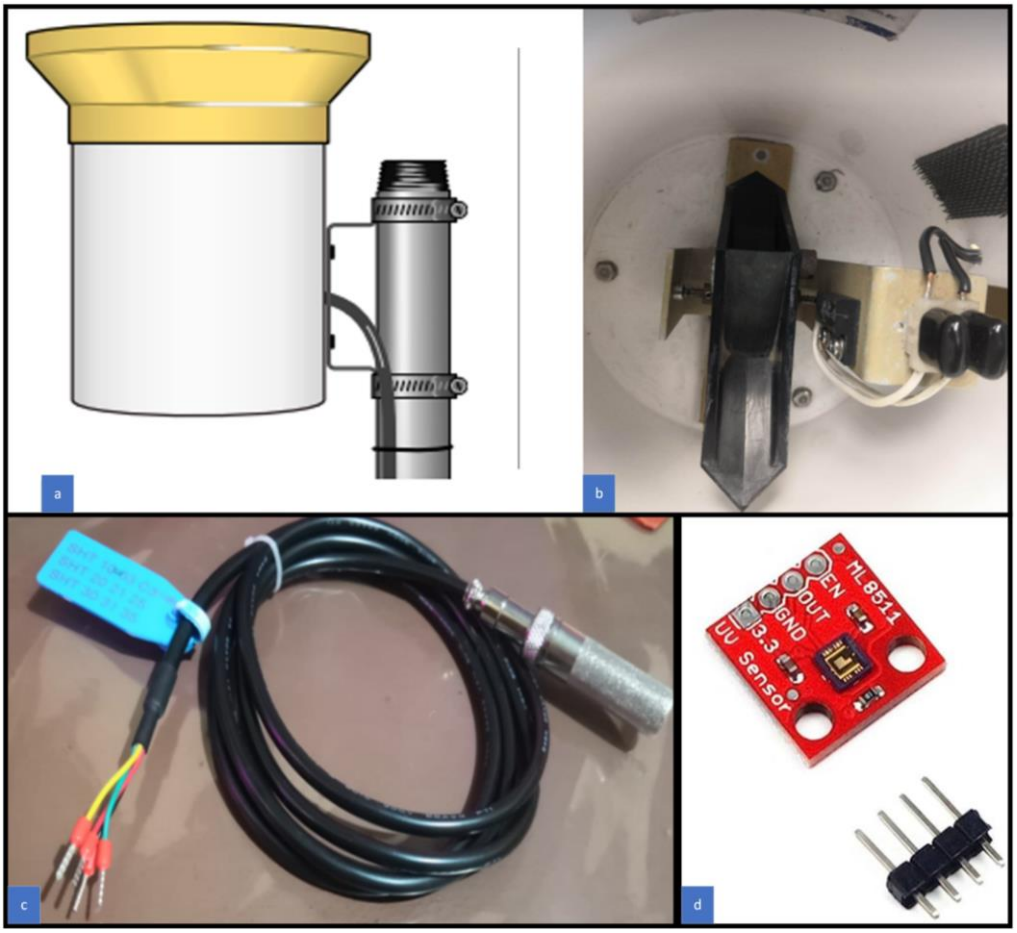
Apéndice H

Gateway Sentrius RG191

Nota. Tomado de Mouser Electronics [45]

Apéndice I

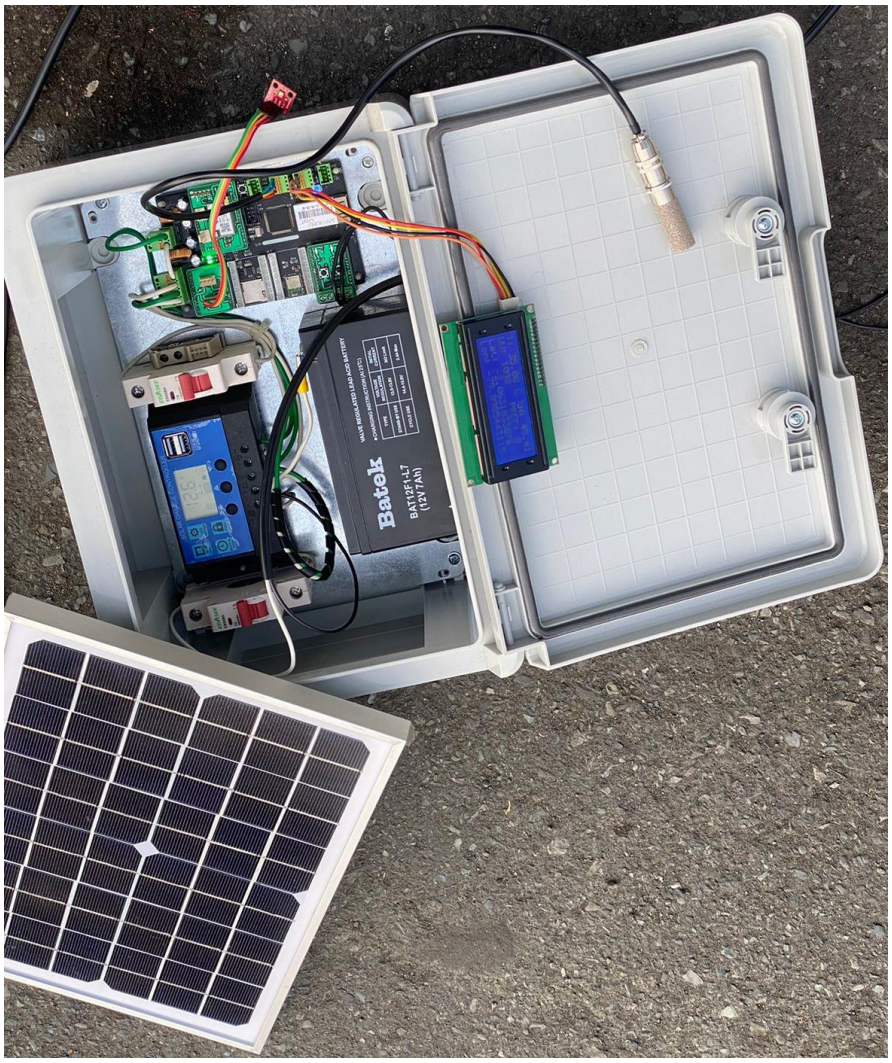
Sensor meteorológicos empleados.



Nota. Vista externa (a) e interna (b) del pluviómetro TR-525M. Tomado de Licor [47]. c) Sensor de temperatura y humedad SHT10. d) Sensor de radiación solar ML8511. Tomado de FunDino [46].

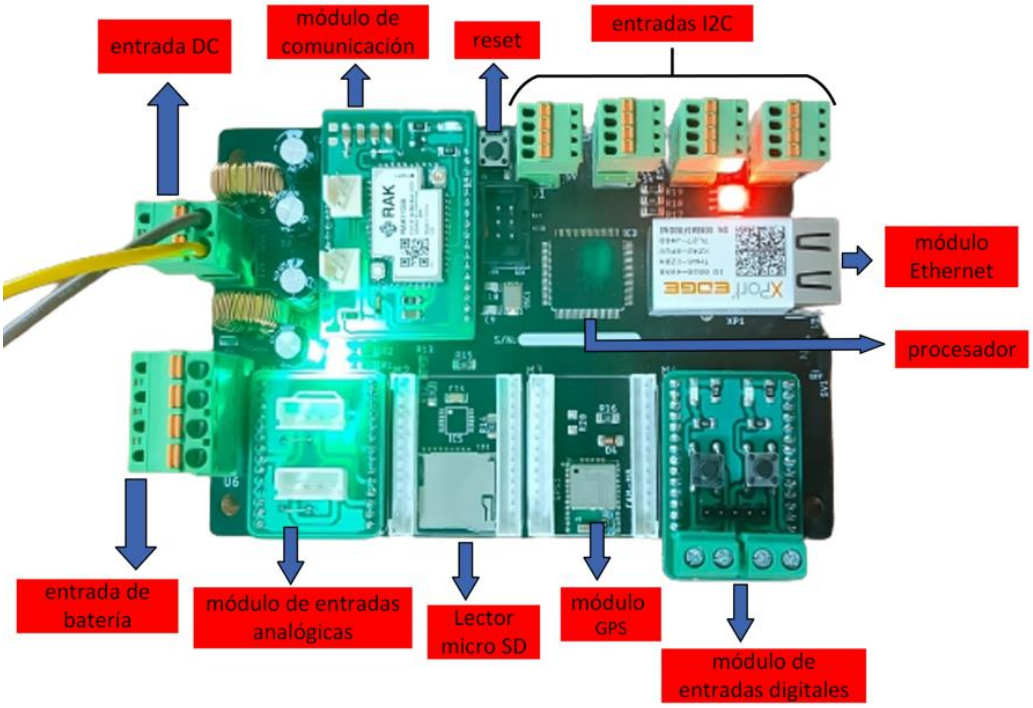
Apéndice J

Prueba del Sistema, almacenamiento y transmisión del EMA en case IP65



Apéndice K

Diseño y prueba de la placa de desarrollo



Apéndice L

Prototipo de EMA en sitio

Prototipo de Estacion meteorologica Automatica

Apéndice M

Código de la tarjeta de desarrollo

```
#include <LiquidCrystal_I2C.h>
#include <TinyGPS++.h>
#include <SD.h>
#include <SPI.h>
#include <Wire.h>
#include <AHT10.h>
AHT10Class AHT10;
TinyGPSPlus gps;
LiquidCrystal_I2C lcd(0x27, 20, 4);
int red = 2;
int green = 3;
int blue = 5;
int pulso1 = 18;
int pulso2 = 19;
int Ledazul = 5;
int Ledverde = 3;
int Ledrojo = 2;
int estadopulso1 = 0;
int estadopulso2 = 0;
int contador1 = 0;
int contador2 = 0;
int sensorUV = 14;
int indice = 0;
float I;
float uvIntensity;
float outputVoltage;
int uvLevel;
float radiacion;
float voltuv;
int intradiacion;
float mulradiacion;
float temperature;
float humedad;
float corriente;
float mulcorriente;
float multemp;
float mulhum;
float mulvol;
uint32_t intvol;
uint32_t intemp;
uint32_t inthum;
uint32_t intcorriente;
int voltaje = A0;
int Sensorcorriente = A1;
float voltajeSensor;
int valor = 0;
float vsense;
float Vsensor;
//float I;
float vreal;
float K=0.185; //resolucion del sensor en Voltios/Amperio para sensor de 5A
const int chipSelect = 53;
const int timeThreshold = 250;
volatile int ISRCounter = 0;
int counter = 0;
long startTime = 0;
String strLat;
String strLng;
String strYear;
String strMonth;
String strDay;
String strHour;
String strMinute;
String strSecond;
String strDate;
```

```
String strTime;
String strCounter;
String strRadiacion;
long lat, lng;

void setup() {
 Serial.begin(9600); // UART para comunnicacion con LANTRONIX
 Serial3.begin(9600);  // UART para comunicacion con GPS
 Serial2.begin(9600); // UART para comunicacion con modulo LoRa RAK11300
 Wire.begin();
 if(AHT10.begin(eAHT10Address_Low))
 pinMode(pulso1, INPUT_PULLUP);
 pinMode(pulso2, INPUT_PULLUP);
 pinMode (red, OUTPUT);
 pinMode (green, OUTPUT);
 pinMode (blue, OUTPUT);
 attachInterrupt(digitalPinToInterrupt(pulso1), debounceCount, RISING);
 //attachInterrupt(digitalPinToInterrupt(pulso2), conteo2, RISING);
 pinMode(Ledverde, OUTPUT);
 pinMode(Ledazul, OUTPUT);
 pinMode(Ledrojo, OUTPUT);
 lcd.init();
 lcd.backlight();

 pinMode(chipSelect,OUTPUT);
 delay(1500);
 if (!SD.begin(chipSelect))
 {
  delay(1500);
  return;
 }
 delay(1000);
}

float get_corriente(int n_muestras)
{
 float voltajeSensor;
 float corriente=0;
 for(int i=0;i<n_muestras;i++)
 {
  voltajeSensor = analogRead(A1) * (5.03 / 1023.0);////lectura del sensor
  corriente=corriente+(voltajeSensor-2.5)/K; //Ecuación  para obtener la corriente
 }
 corriente=corriente/n_muestras;
 return(corriente);
}
 void debounceCount()
{
 if (millis() - startTime > timeThreshold)
 {
  ISRCounter++;
  startTime = millis();
 }
}

 void sdcard() {
 String dataString = "";
//  dataString += String("AÑO");
//  dataString +=",";
  dataString += String(gps.date.year());
  //dataString +=",";

//  dataString += String("MES");
//  dataString +=",";
  dataString += String(gps.date.month());
  //dataString +=",";

//  dataString += String("FECHA");
//  dataString +=",";
  dataString += String(gps.date.day());
  dataString +=",";
```

```
//  dataString += String("HORA");
//  dataString +=",";
  dataString += String(gps.time.hour());
  //dataString +=",";

//  dataString += String("MINUTOS");
//  dataString +=",";
  dataString += String(gps.time.minute());
  //dataString +=",";

//  dataString += String("SEGUNDOS");
//  dataString +=",";
  dataString += String(gps.time.second());
  dataString +=",";

//  dataString += String("LATITUD");
//  dataString +=",";
  dataString += String(gps.location.lat(), 10);
  dataString +=",";

//  dataString += String("LONGITUD");
//  dataString +=",";
  dataString += String(gps.location.lng(), 10);
  dataString +=",";

//  dataString += String("Temperatura");
//  dataString +=",";
  dataString += String(temperature);
  dataString +=",";

//  dataString += String("Humedad");
//  dataString +=",";
  dataString += String(humedad);
  dataString +=",";

//  dataString += String("IRUV");
//  dataString +=",";
  dataString += String(indice);
  dataString +=",";

//  dataString += String("Contador");
//  dataString +=",";
  dataString += String(counter);
  dataString +=",";

  dataString += String("12.5");
  dataString +=",";

  dataString += String("0.125");
  //dataString +=",";

  File dafile = SD.open("datalog.txt",FILE_WRITE);
  if (dafile) {
    //dafile.print(",");
    dafile.println(dataString);
    dafile.close();
    Serial.println(dataString);
  }
  else {
    Serial.println("error al abrir datalog.txt");
  }
}

//  void conteo1(){
//    contador1 = contador1 + 1;
//    Serial.print("contador1 es: ");
//    Serial.println(contador1);
//    delay(500);
//  }
```

```
 void IRUV()
  {
    int uvLevel = averageAnalogRead(sensorUV);
    float outputVoltage = 3.3 * uvLevel/1024;
    float uvIntensity = mapfloat(outputVoltage, 0.99, 2.9, 0.0, 15.0);

 Serial.print("UV Intensity: ");
 Serial.print(uvIntensity);
 Serial.print(" mW/cm^2");
 Serial.println();
  }

 int averageAnalogRead(int pinToRead)
{
 byte numberOfReadings = 8;
 unsigned int runningValue = 0;

 for(int x = 0 ; x < numberOfReadings ; x++)
  runningValue += analogRead(pinToRead);
 runningValue /= numberOfReadings;

 return(runningValue);

}

float mapfloat(float x, float in_min, float in_max, float out_min, float out_max)
{
 return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;
}


 void leer_voltaje(){
 valor = analogRead(voltaje);
 vsense = (valor*5.0)/1023.0;
 vreal = (vsense*106.0)/30.4;
 }
 void temphum(){
  temperature = AHT10.GetTemperature();
  humedad = AHT10.GetHumidity();
  }

 void leegps (){
    while (Serial3.available())     // check for gps data
 {
  if (gps.encode(Serial3.read()))  // encode gps data
  {

    strLat = String(gps.location.lat() * 100000);
    strLat.replace(".","");

    if(strLat.indexOf("-") >= 0)
    {
      strLat.replace("-","");
      strLat = PadLeft(strLat,"0",12);
      strLat = "S" + strLat;
    }
    else
    {
      strLat = PadLeft(strLat,"0",12);
      strLat = "N" + strLat;
    }

    strLng = String(gps.location.lng() * 100000);
    strLng.replace(".","");

    if(strLng.indexOf("-") >= 0)
    {
      strLng.replace("-","");
      strLng = PadLeft(strLng,"0",12);
      strLng = "W" + strLng;
    }
```

```
    else
    {
      strLng = PadLeft(strLng,"0",12);
      strLng = "E" + strLng;
    }
    Serial.println("   ");
    Serial.print("LAT: ");
    Serial.println(gps.location.lat(), 10);
    Serial.print("LONG: ");
    Serial.println(gps.location.lng(), 10);

    Serial.print("Date: ");
    Serial.print(gps.date.day()); Serial.print("/");
    Serial.print(gps.date.month()); Serial.print("/");
    Serial.println(gps.date.year());

    Serial.print("Hour: ");
    Serial.print(gps.time.hour()); Serial.print(":");
    Serial.print(gps.time.minute()); Serial.print(":");
    Serial.println(gps.time.second());


    strYear = String(gps.date.year());
    strYear = PadLeft(strYear, "0", 4);

    strMonth = String(gps.date.month());
    strMonth = PadLeft(strMonth, "0", 2);

    strDay = String(gps.date.day());
    strDay = PadLeft(strDay, "0", 2);

    strDate = strYear + strMonth + strDay;

    strHour = String(gps.time.hour());
    strHour = PadLeft(strHour, "0", 2);

    strMinute = String(gps.time.minute());
    strMinute = PadLeft(strMinute, "0", 2);

    strSecond = String(gps.time.second());
    strSecond = PadLeft(strSecond, "0", 2);

    strTime = strHour + strMinute + strSecond;
   }
  }
}

void mostrar(){
    lcd.setCursor(3, 0);
    lcd.print(temperature);
    lcd.setCursor(15, 0);
    lcd.print(humedad);
    lcd.setCursor(4, 1);
    lcd.print(uvIntensity);
    lcd.setCursor(17, 1);
    lcd.print(counter);
    lcd.setCursor(5, 2);
    lcd.print(gps.location.lat(), 10);
    lcd.setCursor(5, 3);
    lcd.print(gps.location.lng(), 10);

}

String PadLeft(String txt, String character, int len)
{
  String result = "";
  if (len > 0 && (txt.length() < len))
  {
    for (int i = 0; i < len; i++)
    {
      if (txt.length() == len) break;
```

```
    else
    {
     result = character + txt;
     txt = result;
    }
   }
  }
  else if (len > 0 && txt.length() > len) txt = txt.substring(0, len);
  return txt;
}

void loop() {

  lcd.setCursor(0, 0);
  lcd.print("T: ");
  lcd.setCursor(11, 0);
  lcd.print("%H: ");
  lcd.setCursor(9, 0);
  lcd.print("C");
  lcd.setCursor(0, 1);
  lcd.print("UV:");
  lcd.setCursor(11, 1);
  lcd.print("PREC:");
  lcd.setCursor(0, 2);
  lcd.print("Lat: ");
  lcd.setCursor(0, 3);
  lcd.print("Lon: ");

  digitalWrite(blue, HIGH);
  delay(300);
  digitalWrite(blue, LOW);


  leer_voltaje();
  leegps();
  //sdcard();
    if (counter != ISRCounter)
    {
      counter = ISRCounter;
      Serial.println(counter);
    }
    IRUV();
    temphum();
    float I=get_corriente(200);//obtenemos la corriente promedio de 200 muestras
    mulcorriente = I*1000;
    intcorriente = int(mulcorriente);
    mulradiacion = uvIntensity*1000;
    intradiacion = int (mulradiacion);
    multemp = temperature*100;
    intemp = int(multemp);
    mulhum = humedad*100;
    inthum = int(mulhum);
    mulvol = vreal*100;
    intvol = int(mulvol);
    String strTemp = String(intemp);
    strTemp = PadLeft(strTemp,"0",8);
    String strHum = String(inthum);
    strHum = PadLeft(strHum,"0",8);
    String strVol = String(intvol);
    strVol = PadLeft(strVol,"0",8);
    String strCorriente = String(intcorriente);
    strCorriente = PadLeft(strCorriente,"0",8);
    strCounter = String(counter);
    strCounter = PadLeft(strCounter,"0",6);
    strRadiacion = String(intradiacion);
    strRadiacion = PadLeft(strRadiacion,"0",6);

    Serial.print("Temperatura =  ");
    Serial.print(temperature);
    Serial.println("  C");
    Serial.print("Humedad =  ");
```

```
    Serial.print(humedad);
    Serial.println("  %");
    Serial.print("El conteo de precipitacion es: ");
    Serial.println(counter);
    Serial.print("el voltaje es:  ");
    Serial.println(vreal);
    Serial.print("la corriente es:  ");
    Serial.println(I,3);
    Serial.print(" radiacion:  ");
    Serial.println(radiacion);
    Serial.println("----------------------------");
    mostrar();
    sdcard();

// envia cadena al Serial
    Serial.print("dato a enviar: ");
    Serial.print(strTemp);
    Serial.print(strHum);
    Serial.print(strVol);
    Serial.print(strCorriente);
    Serial.print(strLat);
    Serial.print(strLng);
    Serial.print(strDate);
    Serial.print(strTime);
    Serial.print(strCounter);
    Serial.print(strRadiacion);
// datos a nviar al modulo LoRa
    Serial2.print(strTemp);
    Serial2.print(strHum);
    Serial2.print(strVol);
    Serial2.print(strCorriente);
//    Serial2.print(strLat);
//    Serial2.print(strLng);
    Serial2.print(strDate);
    Serial2.print(strTime);
    Serial2.print(strCounter);
    Serial2.print(strRadiacion);
//    Serial2.print(strRadiacion);


}
```

Apéndice N

Código del módulo de comunicación

Código del modulo de comunicacion:

```
‘#include <Arduino.h>
#include "LoRaWan-Arduino.h" //http://librarymanager/All#SX126x
#include <SPI.h>

#include <stdio.h>

#include "mbed.h"
#include "rtos.h"
String readString="";
int temperature;

using namespace std::chrono_literals;
using namespace std::chrono;

bool doOTAA = true;   // OTAA is used by default.
#define SCHED_MAX_EVENT_DATA_SIZE APP_TIMER_SCHED_EVENT_DATA_SIZE /**< Maximum size of scheduler events. */
#define SCHED_QUEUE_SIZE 60                                                                                                    /**<
Maximum number of events in the scheduler queue. */
#define LORAWAN_DATERATE DR_2
/*LoRaMac datarates definition, from DR_0 to DR_5*/
#define LORAWAN_TX_POWER TX_POWER_5                                                                                    /*LoRaMac tx
power definition, from TX_POWER_0 to TX_POWER_15*/
#define JOINREQ_NBTRIALS 3                                                                                                     /**<
Number of trials for the join request. */
DeviceClass_t g_CurrentClass = CLASS_C;                                                    /* class definition*/
LoRaMacRegion_t g_CurrentRegion = LORAMAC_REGION_US915;   /* Region:EU868*/
lmh_confirm g_CurrentConfirm = LMH_UNCONFIRMED_MSG;                                  /* confirm/unconfirm packet
definition*/
uint8_t gAppPort = LORAWAN_APP_PORT;                                                                  /* data port*/

/**@brief Structure containing LoRaWan parameters, needed for lmh_init()
*/
static lmh_param_t g_lora_param_init = {LORAWAN_ADR_ON, LORAWAN_DATERATE, LORAWAN_PUBLIC_NETWORK,
JOINREQ_NBTRIALS, LORAWAN_TX_POWER, LORAWAN_DUTYCYCLE_OFF};

// Foward declaration
static void lorawan_has_joined_handler(void);
static void lorawan_join_failed_handler(void);
static void lorawan_rx_handler(lmh_app_data_t *app_data);
static void lorawan_confirm_class_handler(DeviceClass_t Class);
static void send_lora_frame(void);
void lorawan_unconf_finished(void);
void lorawan_conf_finished(bool result);

/**@brief Structure containing LoRaWan callback functions, needed for lmh_init()
*/
static lmh_callback_t g_lora_callbacks = {BoardGetBatteryLevel, BoardGetUniqueId, BoardGetRandomSeed,
                                          lorawan_rx_handler, lorawan_has_joined_handler,
                                          lorawan_confirm_class_handler, lorawan_join_failed_handler,
                                          lorawan_unconf_finished, lorawan_conf_finished
                                         };
//OTAA keys !!!! KEYS ARE MSB !!!!
uint8_t nodeDeviceEUI[8] = {0xAC, 0x1F, 0x09, 0xFF, 0xFE, 0x06, 0xB8, 0x86};  //AC1F09FFFE06B886
uint8_t nodeAppEUI[8] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
uint8_t nodeAppKey[16] = {0x6E, 0x72, 0xF7, 0x44, 0xB3, 0xAD, 0x90, 0xAA, 0x86, 0x92, 0x11, 0x14, 0xE3, 0x11, 0x66,
0xC4};//6E72F744B3AD90AA86921114E31166C4

// ABP keys
uint32_t nodeDevAddr = 0x260116F8;
uint8_t nodeNwsKey[16] = {0x7E, 0xAC, 0xE2, 0x55, 0xB8, 0xA5, 0xE2, 0x69, 0x91, 0x51, 0x96, 0x06, 0x47, 0x56, 0x9D, 0x23};
uint8_t nodeAppsKey[16] = {0xFB, 0xAC, 0xB6, 0x47, 0xF3, 0x58, 0x45, 0xC7, 0x50, 0x7D, 0xBF, 0x16, 0x8B, 0xA8, 0xC1, 0x7C};

// Private defination
#define LORAWAN_APP_DATA_BUFF_SIZE 64                  /**< buffer size of the data to be transmitted. */
```

```
#define LORAWAN_APP_INTERVAL 20000                  /**< Defines for user timer, the application data transmission interval. 20s, value in [ms]. */
static uint8_t m_lora_app_data_buffer[LORAWAN_APP_DATA_BUFF_SIZE];          //< Lora user application data buffer.
static lmh_app_data_t m_lora_app_data = {m_lora_app_data_buffer, 0, 0, 0, 0}; //< Lora user application data structure.

mbed::Ticker appTimer;
void tx_lora_periodic_handler(void);

static uint32_t count = 0;
static uint32_t count_fail = 0;

bool send_now = false;

void setup()
{
 pinMode(23, OUTPUT);
 digitalWrite(23, LOW);
 Serial2.begin(9600);

 // Initialize Serial for debug output
 time_t timeout = millis();
 Serial.begin(115200);
 while (!Serial)
 {
  if ((millis() - timeout) < 5000)
  {
   delay(100);
  }
  else
  {
   break;
  }
 }
 // Initialize LoRa chip.
 lora_rak11300_init();

 Serial.println("=====================================");
 Serial.println("Welcome to RAK11300 LoRaWan!!!");
 if (doOTAA)
 {
  Serial.println("Type: OTAA");
 }
 else
 {
  Serial.println("Type: ABP");
 }

 switch (g_CurrentRegion)
 {
  case LORAMAC_REGION_AS923:
   Serial.println("Region: AS923");
   break;
  case LORAMAC_REGION_AU915:
   Serial.println("Region: AU915");
   break;
  case LORAMAC_REGION_CN470:
   Serial.println("Region: CN470");
   break;
  case LORAMAC_REGION_CN779:
   Serial.println("Region: CN779");
   break;
  case LORAMAC_REGION_EU433:
   Serial.println("Region: EU433");
   break;
  case LORAMAC_REGION_IN865:
   Serial.println("Region: IN865");
   break;
  case LORAMAC_REGION_EU868:
   Serial.println("Region: EU868");
   break;
  case LORAMAC_REGION_KR920:
```

```
    Serial.println("Region: KR920");
    break;
   case LORAMAC_REGION_US915:
    Serial.println("Region: US915");
    break;
   case LORAMAC_REGION_RU864:
    Serial.println("Region: RU864");
    break;
   case LORAMAC_REGION_AS923_2:
    Serial.println("Region: AS923-2");
    break;
   case LORAMAC_REGION_AS923_3:
    Serial.println("Region: AS923-3");
    break;
   case LORAMAC_REGION_AS923_4:
    Serial.println("Region: AS923-4");
    break;
  }
  Serial.println("=====================================");

  // Setup the EUIs and Keys
  if (doOTAA)
  {
   lmh_setDevEui(nodeDeviceEUI);
   lmh_setAppEui(nodeAppEUI);
   lmh_setAppKey(nodeAppKey);
  }
  else
  {
   lmh_setNwkSKey(nodeNwsKey);
   lmh_setAppSKey(nodeAppsKey);
   lmh_setDevAddr(nodeDevAddr);
  }

  // Initialize LoRaWan
  uint32_t err_code = lmh_init(&g_lora_callbacks, g_lora_param_init, doOTAA, g_CurrentClass, g_CurrentRegion);
  if (err_code != 0)
  {
   Serial.printf("lmh_init failed - %d\n", err_code);
   return;
  }

  // Start Join procedure
  lmh_join();
}

void loop()
{
  // Every LORAWAN_APP_INTERVAL milliseconds send_now will be set
  // true by the application timer and collects and sends the data
  while(Serial2.available()) // Esperando a que haya una comunicación serial
   {
    delay(3);
    char c = Serial2.read();
    readString += c; //Se está almacenando caracter por caracter en readString
   }
    if(readString.length()>0){
    temperature = (readString.toInt());
    }
    readString = "";
  if (send_now)
  {
    send_now = false;
    send_lora_frame();
  }

}

/**@brief LoRa function for handling HasJoined event.
*/
void lorawan_has_joined_handler(void)
```

```
{
 if (doOTAA == true)
 {
  Serial.println("OTAA Mode, Network Joined!");
  digitalWrite(23, HIGH);
  delay(200);
  digitalWrite(23, LOW);
 }
 else
 {
  Serial.println("ABP Mode");
 }

 lmh_error_status ret = lmh_class_request(g_CurrentClass);
 if (ret == LMH_SUCCESS)
 {
  delay(1000);
  // Start the application timer. Time has to be in microseconds
  appTimer.attach(tx_lora_periodic_handler, (std::chrono::microseconds)(LORAWAN_APP_INTERVAL * 1000));
 }
}
/**@brief LoRa function for handling OTAA join failed
*/
static void lorawan_join_failed_handler(void)
{
 Serial.println("OTAA join failed!");
 Serial.println("Check your EUI's and Keys's!");
 Serial.println("Check if a Gateway is in range!");
}
/**@brief Function for handling LoRaWan received data from Gateway

  @param[in] app_data  Pointer to rx data
*/

void lorawan_rx_handler(lmh_app_data_t *app_data)
{
 Serial.printf("LoRa Packet received on port %d, size:%d, rssi:%d, snr:%d, data:%s\n",
           app_data->port, app_data->buffsize, app_data->rssi, app_data->snr, app_data->buffer);
 uint8_t relay_command = *app_data->buffer;
 if(relay_command==01)
 {
  Serial.println("LED on");
  digitalWrite(23, HIGH);
  }
 else
 {
  Serial.println("LED off");
  digitalWrite(23, LOW);

 }
}

void lorawan_confirm_class_handler(DeviceClass_t Class)
{
 Serial.printf("switch to class %c done\n", "ABC"[Class]);
 // Informs the server that switch has occurred ASAP
 m_lora_app_data.buffsize = 0;
 m_lora_app_data.port = gAppPort;
 lmh_send(&m_lora_app_data, g_CurrentConfirm);
}

void lorawan_unconf_finished(void)
{
 Serial.println("TX finished");
}

void lorawan_conf_finished(bool result)
{
 Serial.printf("Confirmed TX %s\n", result ? "success" : "failed");
}
```

```
void send_lora_frame(void)
{
 if (lmh_join_status_get() != LMH_SET)
 {
  //Not joined, try again later
  return;
 }
 digitalWrite(23, HIGH);
 uint32_t i = 0;
 memset(m_lora_app_data.buffer, 0, LORAWAN_APP_DATA_BUFF_SIZE);
 m_lora_app_data.port = gAppPort;
 m_lora_app_data.buffer[i++] = 'A';
 m_lora_app_data.buffer[i++] = 'C';
 m_lora_app_data.buffer[i++] = 'K';
 m_lora_app_data.buffer[i++] = 0x04;

 m_lora_app_data.buffer[i++] = ((temperature & 0xFF000000) >> 24);
 m_lora_app_data.buffer[i++] = ((temperature & 0x00FF0000) >> 16);
 m_lora_app_data.buffer[i++] = ((temperature & 0x0000FF00) >> 8);
 m_lora_app_data.buffer[i++] = temperature & 0x000000FF;

 //m_lora_app_data.buffer[i++] = (temperature>>8)&0x0FF;

 m_lora_app_data.buffsize = i;
 delay(10);
 digitalWrite(23, LOW);
 lmh_error_status error = lmh_send(&m_lora_app_data, g_CurrentConfirm);

 if (error == LMH_SUCCESS)
 {
  count++;
  Serial.printf("lmh_send ok count %d\n", count);
 }
 else
 {
  count_fail++;
  Serial.printf("lmh_send fail count %d\n", count_fail);
 }
}

/**@brief Function for handling user timerout event.
*/
void tx_lora_periodic_handler(void)
{
 appTimer.attach(tx_lora_periodic_handler, (std::chrono::microseconds)(LORAWAN_APP_INTERVAL * 1000));
 Serial.println("Sending frame now...");
 // This is a timer interrupt, do not do lengthy things here. Signal the loop() instead
 send_now = true;
}'
```

Apéndice O

trama en tts en formato JSON


```
{
 "end_device_ids": {
  "device_id": "espol-tesis-pluviometro",
  "application_ids": {
   "application_id": "pmt-energy"
  },
  "dev_eui": "AC1F09FFFE06B886",
  "join_eui": "0000000000000000",
  "dev_addr": "27FE09CB"
 },
 "correlation_ids": [
  "as:up:01H9GVVRRVB4P084M8XQGVZ2KH",
  "gs:conn:01H9F15FDEFVESP88XS6PE12X7",
  "gs:up:host:01H9F15FHJTPN1PW575TJVS941",
  "gs:uplink:01H9GVVRJCYA1P7QRG1YXN4PKE",
  "ns:uplink:01H9GVVRJDSB91DTYT4F8VN99S",
  "rpc:/ttn.lorawan.v3.GsNs/HandleUplink:01H9GVVRJD557MPQ8W0BCYEZR3",
  "rpc:/ttn.lorawan.v3.NsAs/HandleUplink:01H9GVVRRTZ7FNW2MW5FAR7C82"
 ],
 "received_at": "2023-09-04T19:49:09.018892401Z",
 "uplink_message": {
  "session_key_id": "AYpfGm3E9F7yy7XX9NuOXQ==",
  "f_port": 2,
  "f_cnt": 2212,
  "frm_payload": "QUNLBAx2FvkEwgD1",
  "decoded_payload": {
   "A": "A",
   "C": "C",
   "K": "K",
   "corriente": 0.245,
   "delimiter": 4,
   "humedad": 58.81,
   "temperatura": 31.9,
   "voltaje": 12.18
  },
  "rx_metadata": [
   {
    "gateway_ids": {
     "gateway_id": "laird-indoor-isconet",
     "eui": "C0EE40FFFF297E97"
    },
    "timestamp": 1954867203,
    "rssi": -42,
    "channel_rssi": -42,
    "snr": 9,
    "uplink_token": "CiIKIAoUbGFpcmQtaW5kb29yLWlzY29uZXQSCMDuQP//KX6XEIPQk6QHGgwItOnYpwYQrIimgwMguJepufKODg==",
    "channel_index": 3,
    "received_at": "2023-09-04T19:49:08.740553390Z"
   }
  ],
  "settings": {
   "data_rate": {
    "lora": {
     "bandwidth": 125000,
     "spreading_factor": 7,
     "coding_rate": "4/5"
    }
   },
   "frequency": "904500000",
   "timestamp": 1954867203
  },
  "received_at": "2023-09-04T19:49:08.813380336Z",
  "consumed_airtime": "0.066816s",
  "network_ids": {
   "net_id": "000013",
```

```
    "tenant_id": "prometeolabs",
    "cluster_id": "nam1",
    "cluster_address": "nam1.cloud.thethings.industries",
    "tenant_address": "prometeolabs.nam1.cloud.thethings.industries"
   }
  },
  "timestamp": 1693856949454
}
```

Apéndice P

Código decodificador de Paylod.

```
function decodeUplink(input) {
 var bytes = input.bytes;
 var result = {
  data: {},
  warnings: [],
  errors: []
 };

 // Temperatura
 var temp = (bytes[4] << 8) | bytes[5];
 result.data.temp = temp / 100;

 // Humedad
 var hum = (bytes[6] << 8) | bytes[7];
 result.data.hum = hum / 100;

 // Voltaje
 var volt = (bytes[8] << 8) | bytes[9];
 result.data.V = volt / 100;

 // Corriente
 var curr = (bytes[10] << 8) | bytes[11];
 result.data.I = curr / 1000;


 // Fecha
 var fecha = (bytes[12] << 24) | (bytes[13] << 16) | (bytes[14] << 8) | bytes[15];
 var fechaStr = fecha.toString();
 result.data.dia = `${fechaStr.slice(0, 4)}/${fechaStr.slice(4, 6)}/${fechaStr.slice(6, 8)}`;

// Decodificar el tiempo (byte 16 al byte 19)

// Obtiene los bytes relevantes y los convierte a hexadecimal
var timeHex = ((bytes[16] << 24) | (bytes[17] << 16) | (bytes[18] << 8) | bytes[19]).toString(16);

// Convierte el valor hexadecimal a decimal
var timeDecimal = parseInt(timeHex, 16).toString();

// Paddings con ceros si es necesario para asegurarse de que tenga 6 dígitos
timeDecimal = timeDecimal.padStart(6, '0');

// Extrae las horas, minutos y segundos
var hours = timeDecimal.slice(0, 2);
var minutes = timeDecimal.slice(2, 4);
var seconds = timeDecimal.slice(4, 6);
// Ajustar para GMT-5
hours = (hours - 5 + 24) % 24;  // Añadimos 24 para evitar números negativos

// Crea una cadena de tiempo en el formato HH:MM:SS
var time = `${hours}:${minutes}:${seconds} GMT-5`;

// Almacena el tiempo en el objeto 'result'
result.data.t = time;

 // Decodificar pulso
 var pulso = (bytes[20] << 24) | (bytes[21] << 16) | (bytes[22] << 8) | bytes[23];
 result.data.p = pulso;

  // Decodificar UV
 var uvHex = ((bytes[24] << 24) | (bytes[25] << 16) | (bytes[26] << 8) | bytes[27]).toString(16);
 var uvDecimal = parseInt(uvHex, 16);
 result.data.uv = uvDecimal / 1000;

 return result;
}
```