



UNIVERSIDAD NACIONAL DEL ALTIPLANO
FACULTAD DE INGENIERÍA MECÁNICA ELÉCTRICA,
ELECTRÓNICA Y SISTEMAS
ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA



**“DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO
UTILIZANDO IoT PARA SUPERVISIÓN DE MEDIDORES
ELÉCTRICOS CON TECNOLOGÍA LoRa”**

TESIS

PRESENTADA POR:

NILTON EVER SONCCO CHARCA
FRANKLIN MIJAEL QUILCA SANDOVAL

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO ELECTRÓNICO

PUNO – PERÚ

2022



DEDICATORIA

A mis queridos padres Elvis Hilario Yunga Zabalaga y Rosa Sandoval Ruelas por su abnegada labor y su infinita bondad. a mis Hermanas: Paola y Rosario por su apoyo moral y A mis abuelos Cecilia y Aureliano por su motivación incondicional.

Franklin Mijael Quilca Sandoval



DEDICATORIA

A mis queridos padres Severo Eleuterio Soncco Soncco y María de Jesús Charca

Ccama por su abnegada labor y su infinita bondad. A mis Hermanos:

Heiner, Eliana y Lizbeth por su apoyo moral y cariño incondicional. A mis sobrinos

Gianela, Camila, Josue, Joaquin, Rodrigo y Antonela. A toda mi familia por su cariño

y apoyo moral y motivación incondicional.

Nilton Ever Soncco Charca



AGRADECIMIENTOS

Agradecemos a Dios por permitirnos cumplir con este objetivo anhelado a nuestras familias por la confianza y el gran apoyo recibido para culminarlo y a nuestros maestros por sus enseñanzas para desollarnos profesionalmente y habernos brindado todos sus conocimientos en nuestra etapa de estudiantes.

Franklin Mijael Quilca Sandoval

Nilton Ever Soncco Charca



ÍNDICE GENERAL

DEDICATORIA

DEDICATORIA

AGRADECIMIENTOS⁴

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ACRÓNIMOS

RESUMEN 13

ABSTRACT..... 14

CAPITULO I

INTRODUCCIÓN

1.1. PLANTEAMIENTO DEL PROBLEMA 15

1.2. HIPÓTESIS GENERAL..... 16

1.3. HIPÓTESIS ESPECÍFICAS 16

1.4. OBJETIVO GENERAL..... 16

1.5. OBJETIVOS ESPECÍFICOS 17

CAPITULO II

REVISIÓN DE LITERATURA

2.1. ANTECEDENTES DEL PROYECTO 18

2.2. MARCO TEORICO..... 23

2.3. TECNOLOGÍA LoRa..... 23

2.4. LoRaWAN Y SIMILARES 27

2.5. ARQUITECTURA DE RED 31

2.6. CAPAS DE REFERENCIA OSI..... 33

2.7. CAPA FISICA LORA..... 34



2.8. Capa MAC LoRaWAN	36
2.8.1. Clases LoRaWAN	37
2.8.2. Formatos de mensajes físicos	38
2.8.3. Formatos de mensajes MAC	40
2.8.4. Ventanas de recepción	40
2.8.5. Tipos de mensajes	40
2.8.6. Comandos MAC	42
2.8.7. Clase B ventana de recepción sincronizada	44
2.8.8. Clase C Nodos de escucha continua	46
2.8.9. Velocidad de datos adaptativa (ADR)	47
2.9. SEGURIDAD EN LPWAN.....	48
2.9.1. Adquisición y seguimiento de balizas	49
2.9.2. Activación por Personalización (ABP)	50
2.9.3. Simulación LoRaWAN	50
2.10. SERVIDOR DE RED LORAWAN.....	52
2.11. ENTORNO DE PROGRAMACIÓN.....	53
2.12. DATOS EN LA RED.....	54
2.13. PROTOCOLOS DE COMUNICACIÓN	56
2.14. ESP32.....	56
2.14.1. Especificaciones técnicas	58
2.15. CHIP LORA	61
2.15.1. LoRa SX1276	61
2.15.2. Sensor de Corriente	62
2.15.3. Sensor de Voltaje.....	63

CAPITULO III

MATERIALES Y MÉTODOS

3.1. MATERIALES	65
------------------------------	-----------



3.1.1. Hardware	65
3.1.2. Software.....	66
3.2. DISEÑO, NIVEL Y TIPO DE LA INVESTIGACIÓN	66
3.2.1. Diseño de la investigación.....	66
3.2.2. Nivel de la investigación	66
3.3. POBLACIÓN Y MUESTRA	67
3.4. UBICACIÓN Y DESCRIPCIÓN DE LA INVESTIGACIÓN.....	67
3.4.1. Ubicación.....	67
3.5. TÉCNICAS E INSTRUMENTOS DE RECOLECCIÓN DE DATOS	69
3.6. DISEÑO DEL PROTOTIPO	70
3.6.1. Desarrollo del hardware.....	71
3.6.2. Desarrollo del software.....	74
3.6.3. Desarrollo de la plataforma de supervisión	77
3.6.4. Desarrollo del sistema de comunicación	84

CAPITULO IV

RESULTADOS Y DISCUSIÓN

4.1. RESULTADOS DE LA PLATAFORMA DE SUPERVISIÓN	91
4.2. RESULTADOS DEL PROTOTIPO.....	93
V. CONCLUSIONES.....	102
VI. RECOMENDACIONES	103
VII. REFERENCIAS BIBLIOGRÁFICAS.....	104
ANEXOS.....	108

Área: Telecomunicaciones

Tema: redes móviles

FECHA DE SUSTENTACIÓN: 04 de enero de 2022



ÍNDICE DE FIGURAS

Figura 1:Preambulo y espectro de un paquete LoRa.	26
Figura 2:Estructura de marco LoRa.	27
Figura 3:Arquitectura LoRaWAN 30	30
Figura 4:Arquitectura LoRaWAN.	32
Figura 5:Referencia OSI de LoRaWAN 33	33
Figura 6:Ventana de RX y TX en LoRa 35	35
Figura 7:Estructura de marco LoRa..... 36	36
Figura 8:Clases LoRaWAN.	37
Figura 9:Formato de trama de enlace ascendente LoRaWAN 38	38
Figura 10:Formato de trama de enlace descendente de LoRa WAN..... 39	39
Figura 11:Ventana de recepción de enlace descendente..... 39	39
Figura 12:Formato de trama MAC.	40
Figura 13:Formato de trama de mensaje LoRaWAN NewChannelReq..... 44	44
Figura 14:Formato de marco de rango de datos LoRaWAN NewChannelReq..... 44	44
Figura 15:Formato de trama de mensaje LoRaWAN NewChannelAns..... 44	44
Figura 16:Ventana de recepción de clase B y temporización de balizas 46	46
Figura 17:Ventanilla de recepción clase C.	47
Figura 18:Enlace ADRReq.	48
Figura 19:Calculo NwkSKey y AppSKey 50	50
Figura 20:Área de cobertura ilustrada 51	51
Figura 21:Arquitectura de conexión para el entorno de desarrollo. 54	54
Figura 22:Entorno de supervisión..... 55	55
Figura 23:Modelo de comunicación con la nube..... 55	55
Figura 24:Protocolo de transferencia de hipertexto (HTTP). 56	56
Figura 25:Functional Block Diagram. 57	57
Figura 26:Pines de conexión esp32. 58	58
Figura 27:Pines de chip LoRa..... 62	62
Figura 28:Diagrama de bloque chip LoRa..... 62	62
Figura 29:Sensor de corriente..... 63	63
Figura 30:Sensor de voltaje. 64	64
Figura 31:Ubicación del nodo Tx y del servidor y nodo Rx 68	68



Figura 32:Rango de cobertura teorico de LoRa del nodo	68
Figura 33:Ubicación del nodo Rx y Tx vista satelital.	69
Figura 34:diagrama de red y topologia de implementación LoRa.....	71
Figura 35:Diagrama de conexión de sensores a medidor eléctrico	72
Figura 36:Programación para sensor de corriente.	73
Figura 37:Programación para sensor de voltaje.....	74
Figura 38:Declaracion de Variables	75
Figura 39:Bucle de adquisicion de datos voltaje.	76
Figura 40:Bucle de adquisicion de datos corriente.....	76
Figura 41:Impresión de datos en el IDE Arduino.....	77
Figura 42:Crear cuenta de acceso a thingspeak.	77
Figura 43. Asignamos la localizacion del proyecto.....	78
Figura 44:Verificación de correo.....	79
Figura 45;Correo verificado.....	79
Figura 46:Asignación de una contraseña	80
Figura 47:Políticas y privacidad de la plataforma.	80
Figura 48:Mensaje de vinculacion de cuenta en ThingSpeak.....	81
Figura 49:Crear nuevo canal en ThingSpeak.....	81
Figura 50:Asignación de las variables corriente, voltage.....	82
Figura 51:Asignación de datos en el canal Thingspeak.....	83
Figura 52:Datos por visualizar en plataforma IoT Thingspeak.	83
Figura 53:Apikey para la conexión con el prototipo y la platafroma de supervisión.	84
Figura 54:Dispositivos TTGO LoRa32.	85
Figura 55:Esquemático de comunicación LoRa	86
Figura 56:Preferencias del IDE Arduino	87
Figura 57:Instalar la placa ESP32.....	87
Figura 58:Instalar las librerías para activar la pantalla OLED.	88
Figura 59:Instalar las librerías LoRa para transmitir datos.....	89
Figura 60:Configuración de red.....	89
Figura 61:Apikey de plataforma thingspeak.....	90
Figura 62:Supervisión del voltaje	91
Figura 63:Supervisión de la corriente.....	92
Figura 64:Configuración para obtener la obtener la potencia.....	93
Figura 65:Supervisión de la corriente.....	93



Figura 66:Esquemático de conexión.....	94
Figura 67:Declaración de librerías y pines de uso.	95
Figura 68: Inicialización de LoRa.	96
Figura 69:Programación para el envío de datos.....	96
Figura 70:Declaración de variable del modulo LoRa transfer.....	97
Figura 71:Recepción de paquetes LoRa	98
Figura 72:Inicialización de la comunicación I2C.	98
Figura 73:Envío de datos por I2C	99
Figura 74:Inicialización la comunicación I2C (maestro).	99
Figura 75:Recepción de datos por I2C.	100
Figura 76:Esquemático conexión del servidor.....	101



ÍNDICE DE TABLAS

Tabla 1:Frecuencia utilizable por continente.....	24
Tabla 2:Comparación de Sigofx, NB-IoT, LoRaWAN	31
Tabla 3:Valores LoRa SF.	35
Tabla 4:Valores Max y Min para las bandas ISM	35
Tabla 5:Comparación de clases LoRaWAN.....	38
Tabla 6:Tipos de mensajes LoRaWAN MAC	41
Tabla 7:Comandos MAC de LoRaWAN.....	42
Tabla 8:Frecuencias de los canales.	51
Tabla 9:Lista de materiales hardware	65
Tabla 10:Lista de materiales software.	66
Tabla 11:ubicación y elevación de los nodos Rx y Tx	67
Tabla 12:Técnicas e instrumentos de recolección de datos.	70



ÍNDICE DE ACRÓNIMOS

DL	:	Downlink
FFT	:	Fast Fourier Transformada
FEC	:	Tasa de código
FSCM	:	Modulación de chirrido por cambio de frecuencia
IP	:	Internet Protocol
UP	:	Uplink
IDE	:	Integrated Development Environment
OLED	:	Organic Light-Emitting Diode
GW	:	Puertas de enlace
DR	:	Tasa de datos
FSK	:	Frequency Shift Keying
ADR	:	Adaptive Data Rate
LoRaWAN	:	LoRa WIDE Area Networks Protocol
LoRa	:	Long Range
SSM	:	Spread Spectrum
IoT	:	Internet of Things
ISM	:	Instrumentation Scientific and Medical
OTA	:	Over The Air
RF	:	Radio Frequency
SPI	:	Serial Peripheral Interface
I2C	:	Inter Integrated Circuit
P2P	:	Per to Per
ABP	:	Activation By Personalization
SF	:	Spreading Factor
BW	:	Bandwidth
CSS	:	Chirp Spread Spectrum



RESUMEN

En la actualidad el sector eléctrico realiza la medición de consumo de la energía eléctrica utilizando el método de observación, esto podría generar mediciones erróneas, gastos y pérdidas en las empresas distribuidoras como también en los usuarios. En este proyecto se tiene como objetivo general, diseñar e implementar un prototipo utilizando IoT para la supervisión remota de medidores eléctricos basado en tecnología LoRa para la ciudad de Azángaro. Con este modelo de internet de las cosas se logró supervisar los medidores eléctricos de usuarios en la ciudad de Azángaro. En el desarrollo de este proyecto se consiguió determinar en tiempo real su consumo de energía potencial, basándose en el voltaje y amperaje y por consecuencia en la potencia. Se utilizó un prototipo con tecnología LoRa para la supervisión de varios medidores eléctricos. De esta manera se pudo determinar si algunos tienen un comportamiento anómalo, también se pudo detectar si el consumo de energía eléctrica es el adecuado para cada usuario. La implementación de un prototipo de monitoreo remoto de consumo de energía eléctrica basado en tecnología LoRa permitió obtener las mediciones en tiempo real a través de una plataforma tecnológica el cual es de gran utilidad para la población gracias a reportes que genera un servidor central. Se realizó una investigación de nivel exploratorio y de diseño experimental implicando aspectos como la gestión de energía, ahorro económico, el medio ambiente y medir el nivel de consumo de energía eléctrica de cada usuario. La implementación presentó resultados favorables para medir el voltaje, amperaje y potencia para su supervisión remota, los datos de estas variables fueron enviados satisfactoriamente sobre una comunicación LoRa punto a punto almacenándose en un servidor central que funciona sobre una arquitectura de internet de las cosas.

Palabras Clave: internet de las cosas, LoRa, medidores, energía eléctrica.



ABSTRACT

Currently, the electricity sector measures electricity consumption using the observation method, which could generate erroneous measurements, expenses and losses in the distribution companies as well as in the users. The general objective of this project is to design and implement a prototype using IoT for remote monitoring of electric meters based on LoRa technology for the city of Azángaro. With this Internet of Things model it was possible to supervise the electric meters of users in the city of Azángaro. In the development of this project it was possible to determine in real time their potential energy consumption, based on voltage and amperage and consequently on power. A prototype with LoRa technology was used to monitor several electric meters. In this way it was possible to determine if some of them have an anomalous behavior, also it was possible to detect if the electrical energy consumption is adequate for each user. The implementation of a prototype of remote monitoring of electricity consumption based on LoRa technology allowed to obtain measurements in real time through a technological platform which is very useful for the population thanks to reports generated by a central server. An exploratory and experimental design research was carried out involving aspects such as energy management, economic savings, the environment and measuring the level of electricity consumption of each user. The implementation presented favorable results to measure voltage, amperage and power for remote monitoring, the data of these variables were successfully sent over a LoRa point to point communication and stored in a central server that works on an internet of things architecture.

Keywords: internet of things, LoRa, meters, electric energy.



CAPITULO I

INTRODUCCIÓN

Actualmente se vive una tendencia a nivel mundial en conseguir eficiencia en el consumo de energía eléctrica y, así como también, disminuir las emisiones de dióxido de carbono y gases contaminantes que se produce en el consumo de energía eléctrica. Pues siendo la energía eléctrica un servicio esencial para el crecimiento económico y la mejora del bienestar social. Se considera que mediante la supervisión de los medidores eléctricos basada en la tecnología LoRa lograr una óptima gestión de energía, así como optimizar el consumo, también se obtiene beneficios económicos, ya que no es necesario personal para la supervisión (comúnmente conocido como lectorador) del consumo de energía, ya que, este sistema enlazado con internet de las cosas muestra los eventos de consumo registrado por los medidores eléctricos, a su vez, todos los eventos son almacenados en un servidor centralizado para el monitoreo en tiempo real.

1.1. PLANTEAMIENTO DEL PROBLEMA

En la actualidad, la importancia que tiene energía eléctrica en la sociedad es imprescindible, ya que con ella viene el desarrollo de la persona en el mundo moderno la electricidad es el desarrollo que determina el progreso de las necesidades básicas de la sociedad más aún si podemos supervisarla y controlar. Actualmente los medidores eléctricos a nivel mundial son aparatos electromecánicos que se utilizan para medir el consumo de energía, utilizando bobinados de corriente y tensión, de acorde a la influencia de los campos magnéticos que generan. El principio del funcionamiento es igual a los motores de inducción basándose en la relación de las corrientes con los campos magnéticos generados. La supervisión de estos medidores se basa en la inspección visual. De esta misma manera se tiene los medidores en el distrito de Azángaro, provincia de



Azángaro y departamento de Puno. Si la supervisión de los medidores eléctricos se lleva de manera visual no se podrá determinar si algunos tienen un comportamiento anómalo, también no se podrá detectar si el consumo de energía eléctrica es el adecuado o no para cada usuario. El trabajo de investigación permite, implementar nuevas tecnologías de supervisión en medidores de energía eléctrica. La información nueva que aporta esta investigación es la implementación de un sistema de supervisión de tecnología de LoRa. El diseño e implementación del prototipo, permite supervisar los medidores eléctricos sin la necesidad de que un personal supervise el lugar, esta investigación permitirá que una nueva tecnología ingrese al mercado actual.

1.2. HIPÓTESIS GENERAL

El desarrollo de un prototipo utilizando IoT para supervisión de medidores eléctricos con tecnología LoRa permitirá mejorar el monitoreo de los medidores eléctricos en la ciudad de Azángaro.

1.3. HIPÓTESIS ESPECÍFICAS

- El diseño del prototipo con tecnología LoRa y utilizando IoT permitirá supervisar los medidores eléctricos.
- Es posible implementar el sistema de supervisión de medidores eléctricos basado en tecnología LoRa e internet de las cosas.

1.4. OBJETIVO GENERAL

Desarrollar un prototipo utilizando IoT para supervisión de medidores eléctricos con tecnología LoRa en la ciudad de Azángaro.



1.5. OBJETIVOS ESPECÍFICOS

- Diseñar un prototipo utilizando IoT para supervisión de medidores eléctricos con tecnología LoRa.
- Implementar un prototipo para supervisión de medidores eléctricos basado en tecnología LoRa e internet de las cosas.



CAPITULO II

REVISIÓN DE LITERATURA

2.1. ANTECEDENTES DEL PROYECTO

La presente tesis titulada “Diseño y estudio del uso de una red LPWAN (Low Power Wide Área Network) para la optimización de la medición, comunicación y corrección de errores, en el consumo de agua potable” de la Escuela Superior Politécnica del Litoral de Guayaquil, Ecuador; tuvo resultados en relación a los objetivos planteados; es decir, “los módulos LoRa tienen una cobertura adecuada e idónea para el área de la urbanización, la velocidad de transmisión y la cantidad de dispositivos a los cuales puede dar soporte sobrepasa la cantidad de casas censadas, el porcentaje de paquetes enviados sobrepasa el 62% indicando que la transmisión es óptima y el coste final del proyecto es de un valor bajo para poder ser implementado y así obtener ganancias estos resultados se lograron por la 4 fases que abarcó el investigador: la investigación, el desarrollo, la implementación y las pruebas” (Valarezo et al., 2017).

En su trabajo de investigación Jinchuña (2019) demostró que los sistemas de tele gestión para la iluminación o alumbrado con luminarias de tecnología LED ofrece soluciones de iluminación eficiente, optimizando la eficiencia energética reduciendo a su vez, costes de mantenimiento significativamente menor, incrementando la esperanza de vida útil de las luminarias, y así mismo mejorando la calidad del alumbrado público para las personas. Posteriormente, mediante el estudio de todas las variables que integran un sistema de iluminación, se plantea una solución de alumbrado con luminarias de tecnología LED tele gestionadas, se realiza el análisis del recambio de tecnología y finalmente se efectúa una comparación técnica-económica entre el sistema de iluminación actual versus el proyectado.



Flor (2019) en su investigación refiere que va a permitir el control y gestión de cada una de las luminarias LED con las que cuenta la zona, con el uso de tecnologías inalámbricas para conservar la estética del lugar y ofrecer movilidad al usuario final del sistema. Se entrega, el diseño de la red de conexión de los dispositivos IoT (luminarias), el diseño de la red de telecomunicaciones que soporta la interconexión de sensores y controladores hasta la plataforma de control y gestión. Esta propuesta es una iniciativa para convertir ciudades en sostenibles y llevarlas hacia un modelo de ciudades inteligentes.

Guaca, N., & López, M. (2020) en su artículo de investigación revisa aspectos del diseño del sistema IoT y realiza el análisis comparativo entre las diferentes tecnologías LPWAN disponibles para el desarrollo del proyecto.

Szalai et al., (2016) explica el resultado del desarrollo de un nuevo “dispositivo IoT el cual se puede instalar en luminarias de alumbrado público y permite la transferencia de datos independiente de la capa física (por ejemplo, PLC, Wifi, ZigBee u otra transmisión de RF). La funcionalidad principal de la capa de aplicación del protocolo sigue siendo, el control de iluminación, que se implementa mejor sobre la base de estándares existentes (como DALI), pero el sistema permite la implementación de otras tareas de comunicación basadas en otros protocolos de capa de aplicación”.

Kosari & Wentzloff (2019) En su artículo muestran el resultado del “enfoque de comunicación general que permite una velocidad de datos ajustable de 5-384 kbps con rangos de cobertura de decenas de kilómetros, lo que hace que la banda MURS sea una solución muy adecuada para aplicaciones LPWAN. También se presenta la implementación de hardware del método de transmisión propuesto, lo que demuestra una mejora de orden de magnitud en el consumo de energía del circuito de RF con respecto a las soluciones LPWAN existentes disponibles comercialmente”.



Según Petäjäjärvi (2017) en su artículo reporta los resultados obtenidos cuando se usa una potencia de transmisión de 14 dBm y el factor de dispersión más grande de 12 para la banda ISM de 868 MHz, se puede cubrir toda el área del campus con la tecnología LoRa. El índice de entrega de paquetes medidos fue del 96,7% sin acuses de recibo y retransmisiones.

Según Ferrer (2017) en su trabajo de investigación muestra como resultado “el desarrollo completo de la estación meteorológica, su funcionamiento basado en la impresión de los datos monitoreados en una página web con las variables a través de una tecnología de bajo consumo energético, con una herramienta que permita el estudio más detallado de los factores que afectan los cultivos”.

Dávila (2018) en su artículo explica sobre la implementación de una plataforma que incorpora LoRaWAN con el fin de monitorizar el nivel de gas de un tanque estacionario. Para cumplir el objetivo se realiza 5 pasos que se detallan en la investigación da como resultado que se tiene un “desarrollo tecnológico certificado con tres normas, cinco certificados de programas de cómputo y dos solicitudes de registro de diseño industrial”.

Díaz(2019) en su artículo describe “líneas de investigación y desarrollo que forman parte de un proyecto integral destinado a proveer soluciones tecnológicas que aporten a escenarios de smart cities, áreas rurales, marítimas o alejadas de lo urbano donde la conectividad 3G o 4G resulta ineficaz/inaccesible y se presenta a LoRaWAN como red de soporte”.

Aguilar (2019) en su trabajo de investigación concluye “que el diseño de un nuevo sistema basado en IoT es más óptimo. El procedimiento para el diseño de la solución basada en IoT, se basó en el análisis comparativo de diferentes modelos de hardware y software, cálculos matemáticos y simulaciones de cobertura, simulaciones de la



aplicación web y evaluaciones económicas de ambas soluciones.” También determino utilizar dispositivos LoRaWAN pues ello facilita la transmisión desde las estaciones remotas hacia las estaciones de recepción, para luego enviarlas a un servidor de datos.

Córdoba (2017) en su trabajo de investigación afirma que se ofrecerá “información acerca de la contaminación ambiental, así como la de ayudar a la reforestación y mejora del hábitat de los animales autóctonos de la zona informando el estado real del parque. Por otro lado, se informará a los agentes forestales y excursionistas del estado de cada ruta. Utilizo la comunicación LoRaWAN, la cual dará servicio de comunicación entre nodo y Gateway, retransmitiendo información por 3G a Internet para poder visualizar la información desde cualquier lugar. De tal forma que exista un control absoluto del parque para prevención y actuación en caso de desastre”.

Burbano (2019) en su documento de investigación define que “los módulos LoRa que tienen grandes características por su bajo consumo de energía teniendo en cuenta la distancias que logra abarcar y su bajo costo, razón que los hacen una tecnología adecuada para cumplir con los propósitos del proyecto. Para lo cual se construyen 2 nodos sensores los cuales monitorean los siguientes parámetros: Conductividad eléctrica, pH, Temperatura. Por lo que se hace la adquisición, diseño y construcción de los sensores y los circuitos de acondicionamiento. También se realiza un nodo receptor que estará conectado a un computador y mostrará los datos mediante una interfaz gráfica realizada en Java”.

Freire (2019) en su trabajo de investigación se comprobó que “la comunicación con el módulo emisor que se encarga de obtener los datos del sensor de la bomba de extracción y posteriormente enviarlos al módulo receptor para los pozos petroleros, el cual recibe las medidas sensoriales tomadas.” También indica que el módulo receptor también estará conectado a paneles, los cuales permiten la visualización de los datos. El



prototipo tiene la capacidad de guardar la información transmitida en un servidor de base de datos, siempre en cuando exista conexión a internet dentro de la zona.

Rauh et al., (2018) en su artículo de investigación define “un conjunto de parámetros adecuados de las señales LPWAN y el modelo de interferencia, el cual propone un marco de procesamiento basado en el procesamiento de imágenes. Este marco permite un análisis de banda ancha de bandas de frecuencia exentas de licencia. Es capaz de Identificar parámetros de señal tales como longitud de acceso, velocidad de llegada, frecuencia central y ancho de banda. Estos valores obtenidos se pueden usar como parámetros de entrada del modelo de interferencia IEEE 802.15 u otros modelos de ocupación de canal. También presenta resultados de medición ejemplares que obtuvimos durante una prueba de campo como parte del llamado proyecto Betriebs-Adaptive Tracking Sensorsysteme (BATS) en Berlín, Alemania, y mostramos los conjuntos de parámetros derivados”.

El proyecto titulado “Análisis y diseño de una red de sensores en un parque natural” de Universidad Oberta de Cataluña; “utilizo la tecnología LoRaWAN como medio de comunicación, el mismo que dará servicio de comunicación entre nodos y Gateway, retransmitiendo información por 3G a Internet para poder visualizar la información desde cualquier lugar esto permitirá tener un control absoluto del parque para prevención y actuación en caso de desastre” (Cordoba, 2017).

La investigación titulada “Implementación De Una Red De Sensores Inalámbricos LPWAN Mediante Módulos LoRa Para El Monitoreo De La Calidad Del Agua En 2 Ríos” de la Universidad Distrital Francisco José De Caldas de Bogotá Colombia; Tuvo resultados la construcción de “2 nodos sensores los cuales monitorean los siguientes parámetros: Conductividad eléctrica, pH, Temperatura. Por lo que se hace la adquisición, diseño y construcción de los sensores y los circuitos de acondicionamiento. También se



realizará un nodo receptor que estará conectado a un computador y mostrará los datos mediante una interfaz gráfica realizada en Java, cabe recalcar que toda comunicación se realizó con los módulos LoRa por su robustez a interferencias y su máxima ventaja en largas comunicaciones” (Burbano, 2019).

2.2. MARCO TEORICO

2.3. TECNOLOGÍA LoRa

La comunicación LoRa es la capa física de radiofrecuencia que permite comunicarse a largas distancias, ha sido patentada por Semtech en 2014 y para su funcionamiento utiliza la banda ISM. Su modulación se basa en un espectro ensanchado y produce una señal de chirrido (se refiere a una señal de barrido, en donde la frecuencia cambia a una velocidad fija, ya sea constante o exponencial), donde todos los chirridos tendrán prácticamente el mismo tiempo de duración (Semtech, 2016).

El cambio de tiempo de cada chirrido en el lado del receptor se puede determinar después de la alineación del tiempo referencia entre receptor y transmisor por medio de detección de preámbulo. El cambio de hora de cada chirrido se determina multiplicando el chirrido mismo con un chirrido hacia abajo y encontrando el Fast Fourier Transformada (FFT) de la multiplicación. El máximo de la salida FFT indicará el cambio de tiempo del chirrido transmitido que en consecuencia determinará el símbolo digital transmitido. Si N es el chirrido longitud de un símbolo, entonces puede haber N posibles cambios cíclicos diferentes del chirrido base, y el valor del cambio cíclico se puede codificar usando $\log_2 N$ bits que da el factor de dispersión (SF) de la comunicación (Semtech, 2016)

Tabla 1: Frecuencia utilizable por continente.

CONTINENTE	BANDA
AMERICA	DESDE 902 MHZ HASTA 928 MHZ
EUROPA	DESDE 863 MHZ HASTA 870 MHZ
ASIA	DESDE 779 MHZ HASTA 787 MHZ

Nota. Esta tabla me muestra el ancho de banda de frecuencias ISM por continentes .

Para mejorar aún más la robustez contra el ruido y la interferencia de ráfagas, LoRa utiliza códigos intercalados y de corrección de errores de reenvío (FEC) con tasas de código de 4/5 a 4/8. La tasa de datos y la tasa de símbolos dependen del SF y del ancho de banda utilizado (Haxhibeqiri et al., 2018a).

La tasa de símbolos viene dada por la fórmula:

$$R_s = SF * \frac{BW}{2^{SF}} \quad (1)$$

Donde:

SF: factor de dispersión

BW: ancho de banda en Hz.

De la ecuación anterior, se ve que el tiempo de símbolo aumenta aumentando el SF, disminuyendo la tasa de símbolo.

La tasa de datos está dada por la fórmula:

$$R_s = SF * \frac{4}{\frac{4 + CR}{2^{SF}} * 100} * \frac{100}{BW} \quad (2)$$

Donde:

SF: es el factor de dispersión

CR: es la tasa de código

BW: es el ancho de banda en KHz.

Según Haxhibeqiri (2018) la tasa de datos en esta fórmula se expresa en bps. De la ecuación 2, se puede notar que al aumentar el SF, menos bits por símbolo será codificado, disminuyendo la tasa de datos.

La estructura del paquete en la capa física incluye un preámbulo, un encabezado opcional y los datos carga útil. El preámbulo se utiliza para sincronizar el receptor con el transmisor y puede tener una longitud desde 10 hasta 65.536 símbolos en total. La parte fija del preámbulo consta de cuatro símbolos, y el resto es programable con una longitud mínima de seis símbolos y una longitud máxima de 65.532. Según Seller y Sornin, el preámbulo comienza con una secuencia de símbolos upchirp constantes que es programable y ayuda a detectar el inicio del cuadro (Haxhibeqiri et al., 2018a).

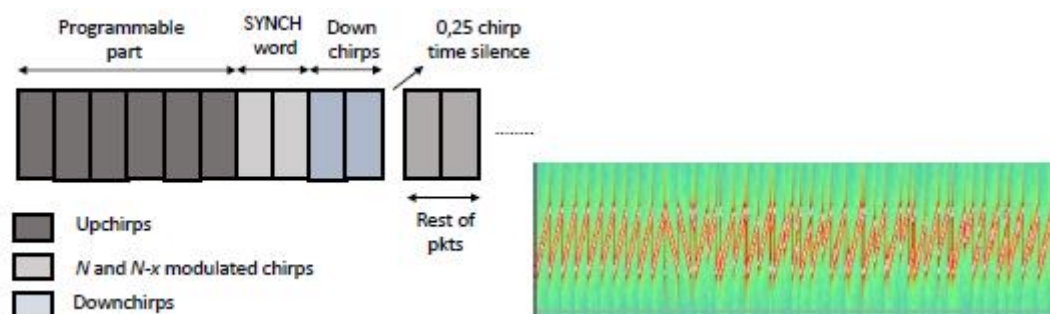
La parte programable es seguida por dos símbolos de chirrido que codifican la palabra de sincronización que se utiliza para la sincronización de fotogramas. Por lo general, los dos chirridos modularán valores opuestos, por ejemplo, x y $N-x$. Cuando el receptor detecta tres consecutivos chirridos, con el primero sin modular (upchirp), el segundo modulado con el primer valor x y el tercero modulado con el valor opuesto ($N-x$), se detecta un nuevo marco. (Haxhibeqiri et al., 2018a).

La palabra de sincronización también se puede usar para distinguir entre dispositivos de diferentes redes usando diferentes valores para cada uno la red. Junto a esto, el preámbulo termina con dos símbolos de chirrido descendente que se utilizan para la frecuencia sincronización. Después de los dos últimos símbolos, un tiempo de símbolo de 0,25 representa un tiempo de silencio utilizado para dejar el receptor alinea la hora. Opcionalmente, el final del preámbulo puede incluir otros dos no modulados chirridos de base que se utilizarán para sincronización final de tiempo y frecuencia. La estructura del preámbulo se muestra en la Figura 1, también se muestra la captura de espectro de un paquete LoRa donde los chirridos ascendentes al comienzo del preámbulo se notan

fácilmente. Tenga en cuenta que el transmisor y el receptor deben conocer el SF de antemano para detectar el preámbulo, ya que el tamaño del preámbulo se escala con SF y no hay preámbulo único para todos los SF (Haxhibeqiri et al., 2018a).

“Los factores de propagación están en un rango de (SF7 - SF12). Los factores de propagación y anchos de banda permitidos están definidos por las agencias reguladoras. En América del Norte, por ejemplo, especifica un ancho de banda de 500 kHz y factores de propagación de 7 a 10” (Haxhibeqiri et al., 2018a).

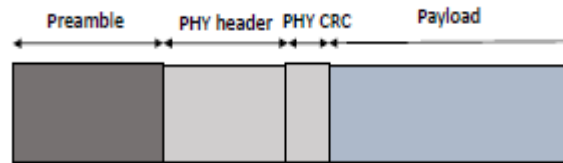
Figura 1: *Preámbulo y espectro de un paquete LoRa.*



Fuente: (Haxhibeqiri et al., 2018).

Opcionalmente, el preámbulo va seguido de un encabezado físico. En tal caso, el encabezado contiene la longitud de la carga útil en bytes, la tasa de código FEC de la carga útil y el encabezado CRC. El encabezado es siempre protegido con FEC con la tasa de código más alta de 4/8. Si estos tres parámetros son conocidos en avance, el encabezado se puede eliminar por completo. Esto disminuye el tiempo en el aire del paquete. En este caso, se aplica el mecanismo de encabezado implícito, donde los parámetros del encabezado se fijan de antemano en ambos extremos, el receptor y el lado del transmisor. La carga útil contiene la capa MAC de LoRaWAN, paquetes de control y paquetes de datos. Opcionalmente, la carga útil puede ir seguida de un CRC de carga útil. El marco de la estructura se muestra en la Figura 2 (Haxhibeqiri et al., 2018).

Figura 2: Estructura de marco LoRa.



Fuente: (Haxhibeqiri et al., 2018).

Como se dijo anteriormente, el mecanismo de modulación LoRa ha sido patentado y su implementación nunca se hace público. Se presenta la primera descripción matemática sobre cómo funciona la modulación LoRa. Los autores describieron la modulación LoRa como modulación de chirrido por cambio de frecuencia (FSCM) y dar el modelo matemático para ello. La capa física de LoRa opera en las bandas de frecuencia de 433, 868 o 915 MHz. En Europa, solo se pueden utilizar las bandas de 868 y 433 MHz. En la banda de 868 MHz hay tres canales de 125 kHz que es obligatorio implementar en todos los dispositivos finales. Hay otros cinco canales de 125 kHz (Haxhibeqiri et al., 2018).

En la sub banda de 867 MHz que se puede utilizar opcionalmente para la comunicación LoRa. En Europa debido a regulaciones de transmisión, cada transmisión en cualquiera de las sub bandas de 868 MHz y 867 MHz debe cumplir con un ciclo de trabajo de radio del 1 % o implementar una agilidad de frecuencia adaptable o escuchar antes de hablar. Mecanismo. Cuando se sigue la regulación del ciclo de trabajo, significa que, si la radio transmite durante 1 s, no puede transmitir durante los próximos 99 s, en América se utiliza la frecuencia de 915 MHz para realizar comunicaciones con la tecnología LoRa (Haxhibeqiri et al., 2018a).

2.4. LoRaWAN Y SIMILARES

La conectividad rentable a Internet es un tema esencial para los dispositivos integrados de bajo consumo que se dedican a una tarea específica. Las tecnologías de



comunicación inalámbrica convencionales son insuficientes cuando se considera cobertura (rango de comunicación), consumo de energía y costo. Objetivos de LPWAN es resolver estos problemas que pueden escalar y son adecuados para implementaciones a gran escala para fines de baja potencia (Haxhibeqiri et al., 2018a).

Según Haxhibeqiri (2018) se supone que las redes de área amplia de baja potencia funcionan a velocidades de datos bajos, para tener kilómetros en rango de cobertura desde zonas urbanas densas hasta regiones suburbanas. LoRaWAN, SigFox, NB-IoT y otras tecnologías de comunicación sub-GHz están brindando con éxito estas funcionalidades, sin embargo, LoRaWAN llamó la atención de organizaciones, comunidades e investigadores para convertir la tecnología LPWAN en la popularidad las LPWAN tienen alguna característica común que distingue a estas tecnologías dentro de redes de comunicación tradicionales:

- Baja potencia, la red y los dispositivos finales deben consumir poca energía.
- El costo de comunicación, implementación y administración es un tema esencial, ya que, en una gran cantidad, el costo de despliegue sería significativamente barato.
- Desde los dispositivos hasta las aplicaciones, todo el ecosistema necesita un sólido mecanismo de seguridad.
- La localización integrada es una ventaja cuando se consideran implementaciones en interiores.
- El despliegue de redes en áreas urbanas densas hace que las redes de radio se bloqueen dentro de las mismas o adyacentes. La modulación robusta (resistencia a la interferencia) es imprescindible.
- Al final del día, los nodos generan datos y esto debe manejarse adecuadamente.



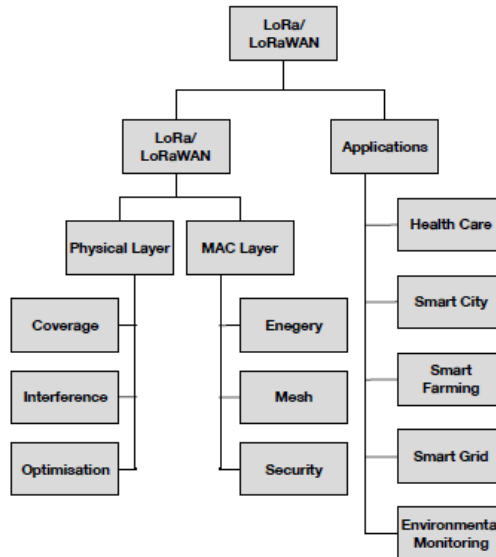
Surgieron varias LPWAN en bandas con y sin licencia, LoRaWAN, Sigfox y NB-IoT.

Estas son tecnologías vitales ampliamente desplegadas. Sigfox es una red de banda ultra estrecha patentada y operada por Sigfox, ocupa 100 Hz y opera en frecuencias de 433 MHz, 868 MHz y 915 MHz dependiendo de la zona geográfica regiones. La tecnología tiene restricciones estrictas sobre el número de paquetes (140 por día) y el paquete tamaño a enviar (12 bytes). Además, LoRaWAN tiene políticas similares para evitar la congestión de la red. Permitiendo que los canales estén disponibles solo 1% del ciclo de trabajo para EU 868MHz. NB-IoT es una tecnología de banda ultra estrecha desarrollada por el grupo 3GPP que se puede adoptar en GSM y redes LTE. Ocupa 200 MHz de ancho de banda y puede alcanzar hasta 200 kbps de transmisión de datos. En comparación con LoRaWAN, Sigfox y NB-IoT son operados por redes globales que limitan despliegues de las redes privadas.(M. A. Ertürk et al., 2019).

Cuando las características de LPWAN se considera que LoRaWAN se convierte en un fuerte candidato para casos de usos científicos e industriales. LoRaWAN es un protocolo abierto propuesto por LoRa Alliance que habilita la capa MAC para la red. En este estudio, presentaremos los aspectos tecnológicos de LoRaWAN y los estudios de vanguardia. Cumplido sobre LoRaWAN hasta hoy. El protocolo LoRaWAN se lanzó por primera vez en 2015 y tenía algunas revisiones, las diferencias en el protocolo se han destacado en la subsección correspondiente. Los estudios se clasifican en dos secciones principales; tecnología de red y las aplicaciones, cada sección tiene sus propias sub ramas. la mayoría de los el trabajo se centra en las pruebas de cobertura y aplicabilidad de la tecnología para diferentes campos de grandes escalar aplicaciones de ciudades inteligentes a activos de seguimiento personal. Sin embargo, aún existen lagunas en la literatura donde los investigadores pueden abordar estos problemas, como la

comunicación posterior a un desastre, descentralizada redes, administración de redes y tasa de datos adaptativa (M. A. Ertürk et al., 2019).

Figura 3: *Arquitectura LoRaWAN*



Fuente: Ertürk (2019, p. 2).

LoRa es una modulación de RF y corresponde a la capa física en el modelo de referencia OSI. Mientras que LoRaWAN es un estándar de capa MAC que coordina el medio. (M. A. Ertürk et al., 2019).

Tabla 2: Comparación de Sigfox, NB-IoT, LoRaWAN

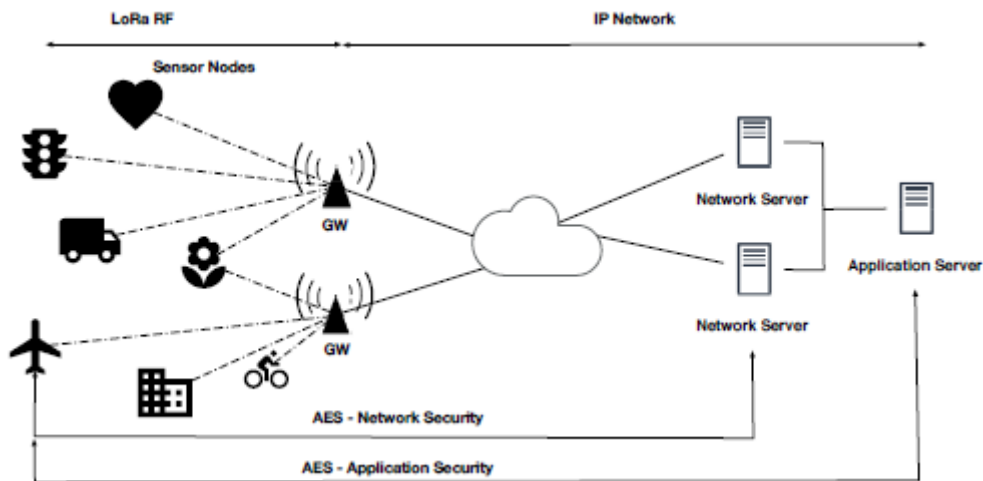
	Sigfox	NB-IoT	LoRaWAN	ZigBee
Standards	Sigfox	3GPP	LoRa Alliance	IEEE 802.15.4
Modulation	BPSK	QPSK	CSS	DSSS, QPSK
Frequencies	ISM: 433 MHz, 868 MHz, 915 MHz	Licensed under LTE	ISM: 433 MHz, 868 MHz, 915 MHz	ISM: 868 MHz, 2.4 GHz
Coverage	10–40 km	2–20 km	1–10 km	10–100 m
Bandwidth	100 Hz	200 kHz	1–10 km	2 MHz
TX Limit	140 Packets per Day	Unlimited	Duty Cycle Lim.	Unlimited
Max Data Rate	100 bps	200 kbps	50 kbps	250 kbs at 2.4 GHz
Private Deployments	No	No	Yes	Yes
Energy Consumption	Low	Low	Low	Low
Security	Low	High	High	High

Fuente Ertürk (2019, p. 3).

2.5. ARQUITECTURA DE RED

La topología de red de LoRaWAN se considera como estrella de estrellas, el sistema tiene tres componentes principales, servidores de red, puertas de enlace (GW) y nodos finales. Los nodos finales se comunican con el servidor de red (o servidor de datos) a través de GW y La comunicación de nodo a GW puede ser modulación LoRa o FSK, cabe mencionar que esta última utiliza topología punto a punto con diferentes velocidades de datos y canales. Los servidores de red administran los GW a través de la tecnología IP estándar y las tramas de datos enviadas a través de nodos finales, recibidos por GW y enrutados a través del servidor de red. Una visión general de la arquitectura LoRaWAN se presenta en la Figura 4 (M. A. Ertürk et al., 2019).

Figura 4: *Arquitectura LoRaWAN.*



Fuente: Ertürk (2019, p. 3)

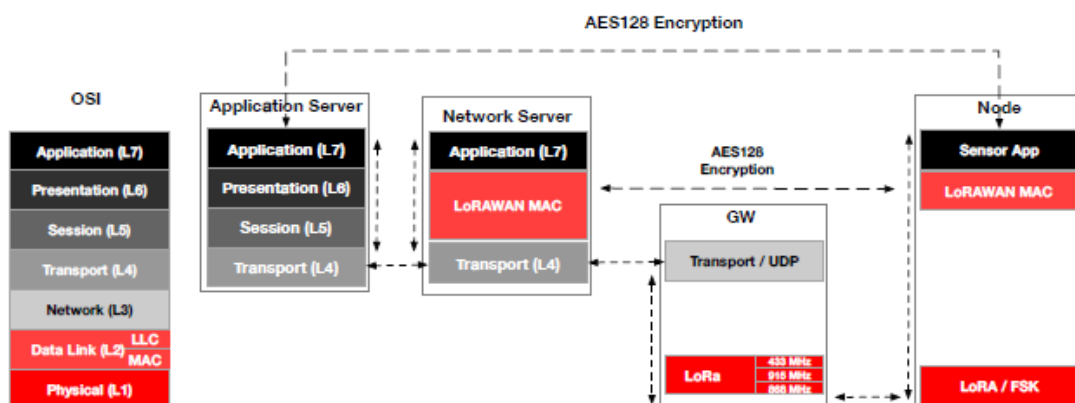
LoRaWAN es un protocolo de capa MAC y tiene como objetivo resolver problemas de gestión del medio y congestión en la red. Cualquier nodo que utilice el protocolo LoRaWAN puede beneficiarse de las siguientes funciones proporcionadas por la norma (M. A. Ertürk et al., 2019).

- Gestión de canales
- Eficiencia energética
- Velocidad de datos adaptativa
- Seguridad
- Geolocalización sin GPS

2.6. CAPAS DE REFERENCIA OSI

El diseño de red en capas proporciona una comunicación fluida entre los diferentes elementos de la red. Las capas LoRaWAN se pueden asignar a la referencia OSI para comprender mejor la tecnología subyacente como se presenta en la Figura 5. Las implementaciones de red LoRaWAN genéricas consisten en múltiples nodos finales, uno o más puertas de enlace y al menos un servidor de red para ejecutar una red completa. Las redes LoRaWAN se distinguen por la comunicación TCP/IP clásica con GW y nodos finales. Los servidores de red son servicios de aplicaciones simples que operan sobre la capa de transporte; sin embargo, todas las funcionalidades de la capa MAC de toda la red están controladas por el servidor de red (M. A. Ertürk et al., 2019).

Figura 5: Referencia OSI de LoRaWAN



Fuente: Ertürk (2019, p. 3).

LoRa es una tecnología de capa física que opera en L1, y la función principal es transmitir Datos de la capa de aplicación al medio. En referencia del modelo OSI, la capa de enlace de datos, coincide con el protocolo LoRaWAN, que define el acceso seguro al medio y estrategias de gestión de nodos finales. Por lo general, los nodos finales realizan una tarea específica simple para minimizar consumo de energía por lo tanto, los nodos finales realizan funcionalidades L1, L2 y L7 en la referencia OSI. Los GW son los enlaces



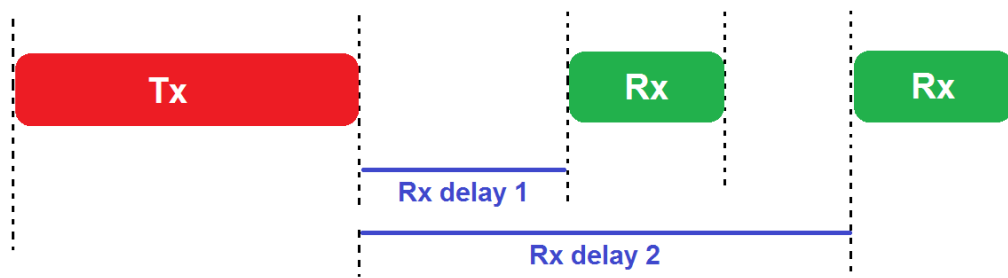
entre el nodo final y el servidor de red. El servidor de red es el factor principal que controla la medio (M. A. Ertürk et al., 2019).

LoRaWAN tiene como objetivo apoyar a los dispositivos finales que funcionan con una sola batería para un periodo de tiempo prolongado, a través de una pérdida de acoplamiento máxima de 155dB, y una cobertura fiable y descarga eficiente de archivo (M. Ertürk et al., 2019).

2.7. CAPA FISICA LORA

Long Range (LoRa) es una tecnología de capa física introducida por Semtech y propiedad intelectual, los derechos de modulación pertenecen a la empresa. La modulación LoRa extiende los principios tradicionales de Spread Spectrum para reducir la cantidad de energía necesaria para transmitir bits a través del canal. La tasa de datos (DR) en la comunicación alcanzable se puede calcular a partir del ancho de banda BW (Hz), el factor de dispersión (SF) y la tasa de codificación (CR). La modulación LoRa tiene un mejor rendimiento que las técnicas de salto de frecuencia en la gestión interferencias La modulación puede tolerar interferencias de niveles de potencia arbitrarios hasta el 30% de la longitud de símbolo con menos de 6 dB de degradación de la sensibilidad. SF es la variable clave que asegura la calidad del servicio. Cuando se utiliza el rango inferior de valores SF, la tasa de datos es muy alta y el tiempo de aire es bajo. Los SF más altos amplían el rango, pero limitan la calidad de servicio. Los SF del 7 al 12 permiten comunicaciones ortogonales, es decir, diferentes redes pueden hablar simultáneamente en la misma banda de frecuencia sin interferir. La Tabla 3 presenta los valores de SF contra Chirridos/Símbolo y Demodulación SNR. La modulación SF LoRa se basa en la representación de cada bit de información por información multi-chirps (Kurniawan, Building Wireless Sensor Networks with ESP32 LoRa, 2019).

Figura 6: Ventana de RX y TX en LoRa



Fuente: Ertürk (2019, p. 3)

Tabla 3: Valores LoRa SF.

Spreading Factor	Chirps/Symbol	Demodulation SNR (dB)
7	128	-7.5
8	256	-10
9	512	-12.5
10	1024	-15
11	2048	-17.5
12	4096	-20

Fuente: Ertürk (2019, p. 3)

Tabla 4: Valores Max y Min para las bandas ISM

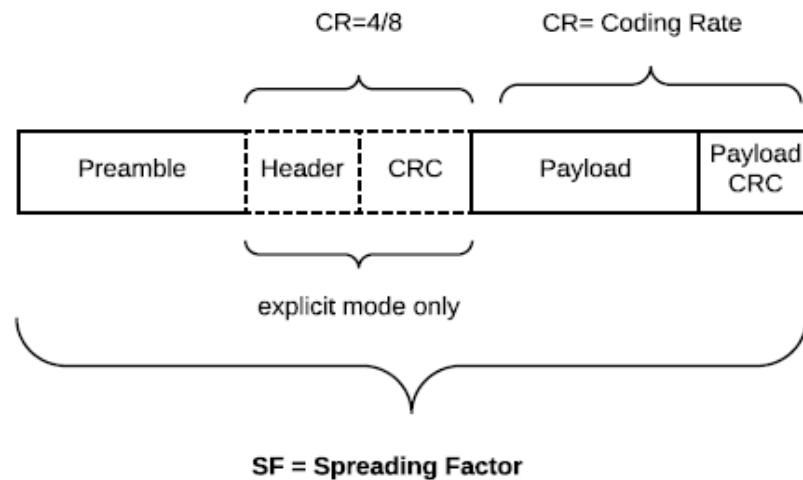
Parámetros	Mín	Máx
Ciclo de trabajo: Esta implementada en algunas bandas de frecuencia	1%	No limit
Vel de datos y tamaño de fotograma en 868 MHz de UE	250 bits/s; 59 octetos	50000 bits/s; 250 octetos
Vel de datos y tamaño de fotograma en 915 MHz de US	980 bits/s; 19 octetos	21900 bits/s ; 250 octetos

Fuente: (Kurniawan, Building Wireless Sensor Networks with ESP32 LoRa, 2019)

El formato de trama LoRa puede ser implícito o explícito, donde el paquete explícito incluye un encabezado corto que contiene información sobre los bytes, CRC y

la tasa de codificación utilizada en la trama. El formato de cuadro LoRa es dada en la Figura 7. En la mayoría de los casos, la tasa de codificación y el CRC son conocidos y no cambian, no es necesario para especificar los valores, este escenario se denomina modo de encabezado implícito.

Figura 7: Estructura de marco LoRa



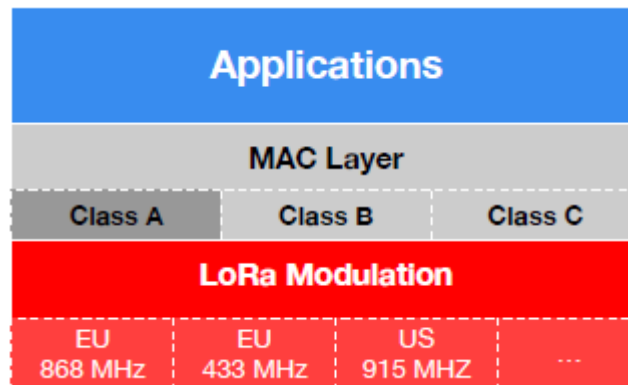
Fuente: (Kurniawan, Building Wireless Sensor Networks with ESP32 LoRa, 2019)

2.8. CAPA MAC LoRaWAN

Según Ertürk (2019) LoRaWAN es un protocolo de capa dos para proporcionar velocidad de datos adaptativa (ADR) a dispositivos finales LoRa, Funcionalidades de Seguridad, Geolocalización sin GPS, Acceso a Canales y Ahorro de Energía. LoRaWAN es introducido por la alianza LoRa. En la versión 1.1, LoRaWAN admite roaming y agrega más características de seguridad a la Clase B. Se especifican parámetros regionales por un documento complementario. LoRa y otras tecnologías de banda estrecha tienen que ser eficientes, por lo que los gastos generales costarán más energía o latencia. Para superar este problema, LoRaWAN utiliza un sistema bastante estrategia simple de gestión de canales para mantener la rentabilidad de los dispositivos finales. LoRaWAN adopta el mecanismo ACK para simplificar el control de acceso al medio. El protocolo de capa

MAC define una clase obligatoria y dos opcionales para diferentes casos de uso posibles y una descripción general, las representaciones de esta clase se presentan en la Figura 8.

Figura 8: Clases LoRaWAN.



Fuente: (M. Ertürk et al., 2019).

2.8.1. Clases LoRaWAN

LoRaWAN tiene tres estrategias principales de acceso al canal agrupadas como Clase A, B y C. La Clase A es obligatoria y todos los dispositivos finales deben implementar la especificación de protocolo definida. La tabla 5 presenta una breve información sobre las clases. LoRaWAN adopta el acceso aleatorio tipo ALOHA para mantener la complejidad de la red lo más simple posible y maximizar el ahorro de energía. En los dispositivos de Clase A, el enlace descendente (DL) está controlado por el nodo final y los dispositivos no tienen una ventana de recepción de DL activa. Solo se activan dos ventanas de recepción de DL después de se realizó una transmisión de enlace ascendente (UL). Cualquier paquete que se transmita desde GW al nodo debe esperar el siguiente mensaje de UL. Por lo tanto, los dispositivos de Clase A son los más eficientes energéticamente (M. Ertürk et al., 2019).

Tabla 5: *Comparación de clases LoRaWAN.*

Clases	Consumo de batería	Descripcion
A	Más eficiente energéticamente	Debe ser compatible con todos los nodos finales. DL después de TX Slotted communication
B	Eficiente con DL controlada	synchronized with beacon frames Los dispositivos escuchan continuamente. DL sin latencia.
C	Menos eficiente	

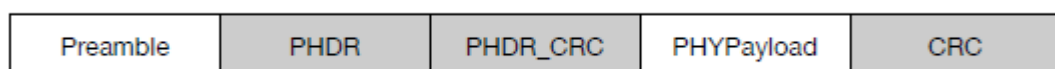
Fuente: (Blokdyk, LPWAN A Complete Guide, 2019).

Los dispositivos de clase B controlan las ventanas de recepción con intervalos de tiempo predeterminados y los dispositivos finales abren ventanas DL solo en horarios programados. Los temporizadores de nodo final son tramas Beacon controladas por la puerta de enlace. Los dispositivos de Clase C son los dispositivos menos eficientes porque las ventanas de recepción de DL siempre están activas. Los dispositivos de clase A y B tienen una latencia más alta en comparación con los nodos de clase C (Blokdyk, LPWAN A Complete Guide, 2019).

2.8.2. Formatos de mensajes físicos

Según Ertürk (2019) los mensajes de UL se transmiten desde los nodos finales al medio y son recibidos por todos los GW dentro del alcance. El servidor de red elimina los marcos duplicados y se procesa el paquete único. Los mensajes UL utilizan el modo explícito en la modulación LoRa.

Figura 9: *Formato de trama de enlace ascendente LoRaWAN*



Fuente: (M. Ertürk et al., 2019).

Los mensajes DL se envían desde el servidor de red a los dispositivos a través de GW. Los mensajes DL son enviados a un único nodo final. LoRaWAN es un protocolo de un solo salto y no admite enrutamiento ni Entrega de paquetes multisalto. El formato de trama DL se muestra en la Figura 10 (M. Ertürk et al., 2019).

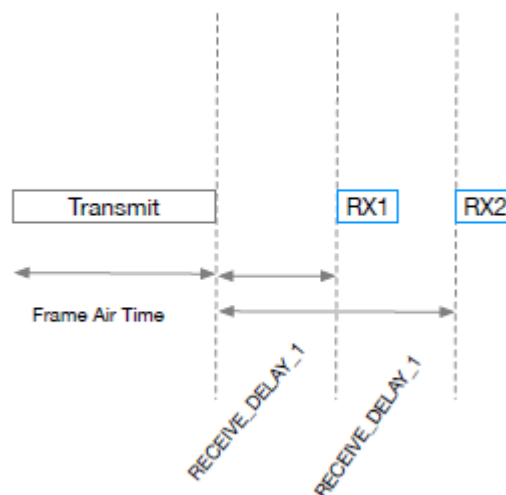
Figura 10: *Formato de trama de enlace descendente de LoRa WAN.*



Fuente: (M. Ertürk et al., 2019).

Las ventanas de recepción son un control de acceso de asignación de fechas iniciado a través de un mensaje UL de los nodos finales. La primera ventana de recepción de DL RX1 se abre hasta RECEIVE_DELAY_1 segundos (+/- 20 microsegundos) utilizando el mismo canal que se muestra en la Figura 11. RX2 tiene una tasa de datos configurable y frecuencia a través de Comandos MAC (M. A. Ertürk et al., 2019).

Figura 11: *Ventana de recepción de enlace descendente.*

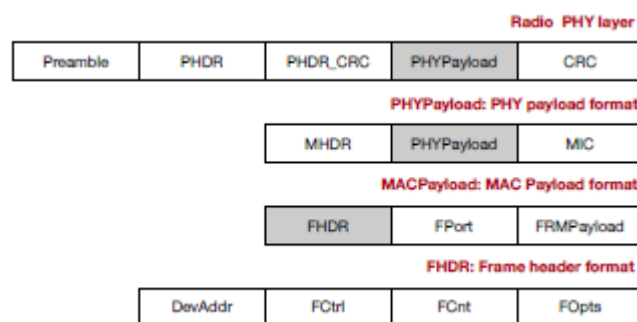


Fuente: Ertürk (2019, p. 12).

2.8.3. Formatos de mensajes MAC

LoRa reconoce entre mensajes de enlace ascendente UL y descendente DL. Todos los mensajes de enlace UL Y DL llevan una carga útil PHY (Payload) que comienza con un encabezado MAC de un solo octeto (MHDR), inmediatamente después de una carga útil MAC (MAC payload), y terminando con un código de integridad de mensaje de 4 octetos (MIC) (Gomez, 2017).

Figura 12: Formato de trama MAC.



Fuente: (Gomez, 2017).

2.8.4. Ventanas de recepción

Después de cada transmisión de enlace ascendente, el dispositivo final abre dos ventanas de recepción cortas. Las horas de inicio de la ventana de recepción se definen tomando como referencia el final de la transmisión. Un dispositivo terminal no transmitirá otro mensaje de enlace ascendente antes de haber recibido un mensaje de enlace descendente en la primera o segunda ventana de recepción de la transmisión anterior, o de que haya expirado la segunda ventana de recepción de la transmisión anterior. (Koon, 2019)

2.8.5. Tipos de mensajes

En la presente se desarrolla los tipos de mensajes LoRaWAN MAC. Los mensajes de solicitud de ingreso y aceptación son utilizados por el proceso de activación por aire (OTAA). Los mensajes de datos pueden ser comandos MAC o mensajes de datos. Los mensajes no confirmados no requieren cualquier mecanismo de acuse de recibo (ACK)



para verificar la entrega del mensaje (Kurniawan, Building Wireless Sensor Networks with ESP32 LoRa, 2019).

Tabla 6: *Tipos de mensajes LoRaWAN MAC*

Código	Descripción
000	“Petición de unión”
001	“Aceptación de unión”
010	“Datos no confirmados Up”
011	“Datos no confirmados Down”
100	“Datos confirmados Up”
101	“Datos confirmados Down”
110	RFU
111	Propietario

Fuente: (Kurniawan, Building Wireless Sensor Networks with ESP32 LoRa, 2019)

2.8.6. Comandos MAC

Para el protocolo LoRaWAN ejecuta un mecanismo de control simple para coordinar los dispositivos medios y finales a través de los comandos presentados. Los comandos MAC se identifican mediante un identificador de octeto llamado CID (identificador de comando). Cada comando tiene una longitud predefinida y debe identificarse implícitamente por la implementación (Seneviratne, 2019).

Tabla 7: Comandos MAC de LoRaWAN.

CID	Command	Transmitted By	Description
0x02	LinkCheckReq	Node	Connection validation.
0x03	LinkCheckAns	GW	Reply to LinkCheckReq.
0x03	LinkADRReq	GW	Adjust node data rate, tx power, rx rate or channel
0x04	LinkADRAns	Node	Response to LinkADRReq.
0x04	DutyCycleReq	GW	Set tx duty cycle.
0x04	DutyCycleAns	Node	Response DutyCycleReq.
0x05	RXParamSetupReq	GW	Set rx slot parameters.
0x05	RXParamSetupAns	Node	Response RXParamSetupReq.
0x06	DevStatusReq	GW	Request status of node.
0x06	DevStatusAns	Node	Response to DevStatusReq with battery level and demodulation margin.
0x07	NewChannelReq	GW	Setup/modify new radio channel.
0x07	NewChannelAns	Node	Response to NewChannelReq.
0x08	RXTimingSetupReq	GW	Set reception slot timing.
0x08	RXTimingSetupAns	Node	Response to RXTimingSetupReq.
0x08-0xFF	Proprietary	Node/GW	Reserved further extensions.

Fuente: (Seneviratne, 2019).

El dispositivo final utiliza el comando LinkCheckReq para validar la conectividad de la red. LinkCheckAns es respondido por el GW, incluido GwCnt, que indica la cantidad de GW que recibieron la solicitud con éxito. LinkADRReq se envía desde GW a través de los nodos para adoptar una tasa de datos. El mensaje ADR (tasa de datos adaptable) contiene la tasa de datos y la potencia de salida Tx TXPower con valores específicos de la región. Además, los canales de enlace ascendente utilizables se codifican utilizando el campo ChMask en el mensaje. Los dispositivos finales envían LinkADRAns para indicar que la tasa de datos solicitada por el clima, la potencia de transmisión y los estados de canal solicitados se descartan o implementado con éxito (Systems, 2022).

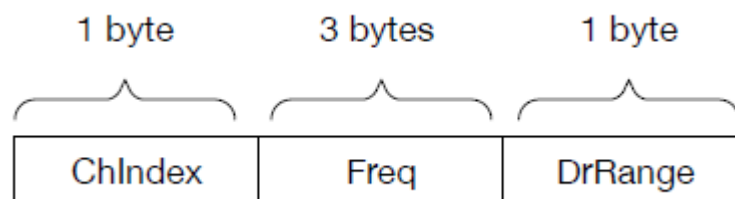
El ciclo de trabajo agregado máximo se controla a través de la solicitud DutyCycleReq y el mensaje contiene el campo MaxDCycle en el rango [0:15] que se implementará para todas las sub bandas como. 0 indica que no hay limitación del ciclo de trabajo y 255 es el interruptor de apagado para apagar el dispositivo remoto inmediatamente (M. A. Ertürk et al., 2019).

$$\text{Ciclo de trabajo agregado} = \frac{1}{2^{\text{MaxDCycle}}} \quad (3)$$

DutyCycleAns es una respuesta a DutyCycleReq sin ninguna carga útil. RXParamSetupReq controla la tasa de datos de la segunda ventana de recepción (RX2) y la frecuencia para cada enlace ascendente RXParamSetupAns contiene bits de estado que indican que la configuración es correcta o no es válida. DevStatusReq es un mensaje vacío que se utiliza para verificar el estado de los dispositivos finales. Respuestas del dispositivo final el mensaje con nivel de batería, margen de modulación de la relación SNR con DevStatusAns. El servidor de red puede modificar un canal existente o agregar uno nuevo con NewChannelReq solicitud. El mensaje contiene la frecuencia central y el rango de tasas de datos utilizables como se muestra. De acuerdo con el protocolo, los dispositivos finales deben manejar 16 canales diferentes indexados de 0 a N – 1 donde N es el número de canales predeterminados. ChIndex es el índice del canal que se está configurando. El campo Freq es un entero sin signo de 24 bits para referirse a la frecuencia real (100 Freq). Por debajo de 100 MHz reservado para casos de uso futuros, la frecuencia se puede configurar entre 100 y 1670 MHz con pasos de 100 Hz si frecuencia valor establecido en 0, el canal está deshabilitado. El subcampo DrRange en la Figura 15 en NewChannelReq especifica el rango de datos permitido para el canal dado con índices de 4 bits. El campo DrRange tiene min y max velocidades de datos denominadas MinDR,

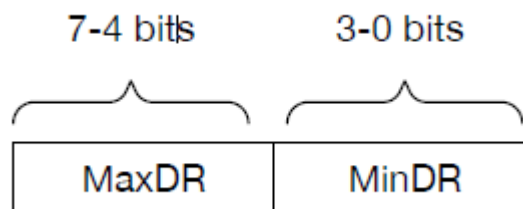
MaxDR. DrRange = 0 50 se refiere a DR0/125 kHz–DR5/125 kHz están disponibles. El dispositivo final envía un marco ACK (Figura 16) que indica el estado de la solicitud NewChannelReq ya sea con éxito o no (Gomez, 2017). El retraso entre el enlace ascendente Tx y la primera ventana de recepción se configura con RXTimingSetupReq mensaje. La segunda ventana de recepción se abre después de un segundo de la primera ventana.(M. Ertürk et al., 2019).

Figura 13:Formato de trama de mensaje LoRaWAN NewChannelReq.



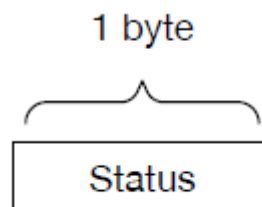
Fuente: (M. Ertürk et al., 2019).

Figura 14:Formato de marco de rango de datos LoRaWAN NewChannelReq.



Fuente: (M. Ertürk et al., 2019).

Figura 15:Formato de trama de mensaje LoRaWAN NewChannelAns.



Fuente: (M. Ertürk et al., 2019).

2.8.7. Clase B ventana de recepción sincronizada

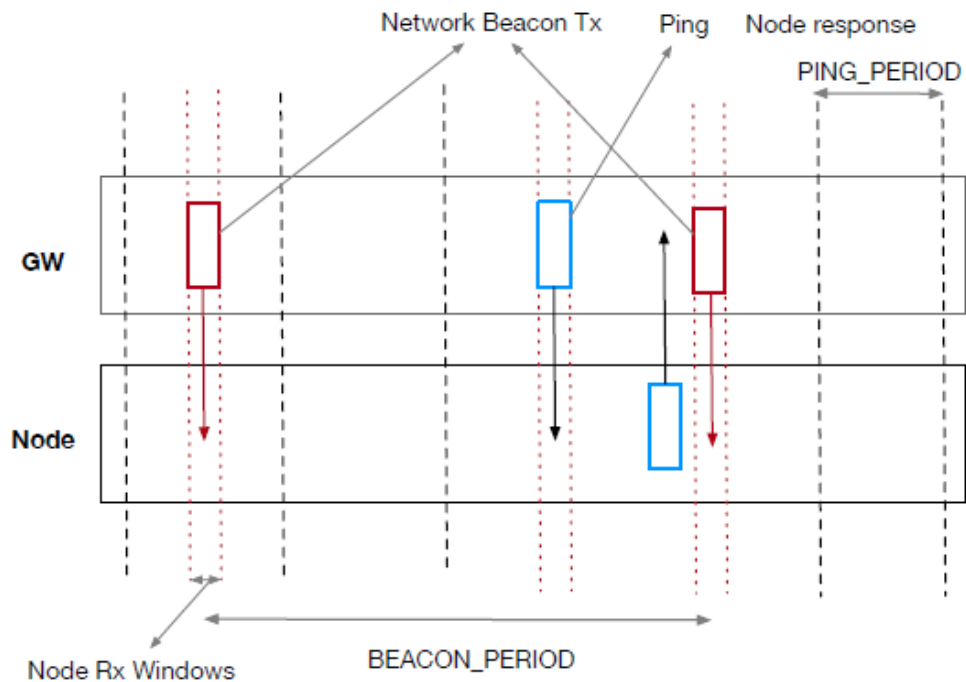
La especificación del dispositivo de clase B tiene como objetivo resolver el mensaje DL limitado en la clase A mediante el uso de un dispositivo sincronizado, la



ranura de recepción. En la Clase B, los GW transmiten un mensaje de difusión a los nodos con una referencia de tiempo. Los nodos finales usan esta referencia para abrir la ventana de recepción que se llama "ranuras de ping" para iniciar DL proceso de recepción. Todos los nodos de la red deben implementar la Clase A e iniciar la red con la misma unión proceso. Si un dispositivo final decide operar como dispositivos de Clase B, debe seguir estos pasos: (Singh, 2019).

- El nodo hace una solicitud para operar como modo Clase B y busca una baliza. La baliza puede ser encontrado BEACON_LOCKED o no BEACON_NOT_FOUND con la ayuda de BeaconTimingReq mensaje.
- El nodo selecciona una velocidad de datos de ranura de ping apropiada y un período de ranura dependiendo de la señal, fuerza y nivel de batería.
- El modo Clase B se controla sobre el campo FCTRL de cada trama UL transmitida por el dispositivo final.
- El dispositivo final informa periódicamente su ubicación y ruta DL al servidor de red.
- Si no se recibe ninguna trama de baliza durante un tiempo determinado, los dispositivos vuelven al dispositivo de Clase A.

Figura 16: Ventana de recepción de clase B y temporización de balizas

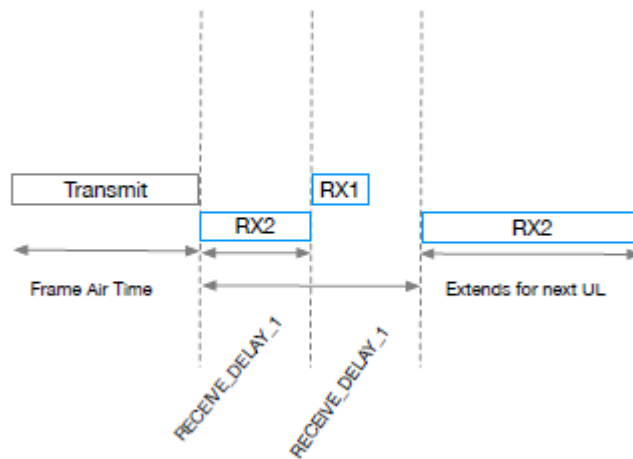


Fuente: (Singh, 2019).

2.8.8. Clase C Nodos de escucha continua

Los dispositivos de Clase C implementan un mecanismo de ventana de recepción de Clase A obligatorio y estos dispositivos mantener las ventanas de recepción abiertas después de segundos de retraso RECIEVE_DELAY2. Este proceso se muestra en la figura 17 (Singh, 2019).

Figura 17: Ventanilla de recepción clase C.

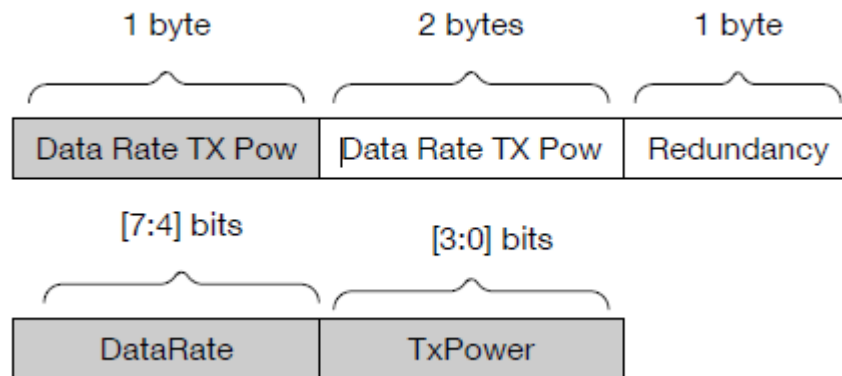


Fuente: (Singh, 2019).

2.8.9. Velocidad de datos adaptativa (ADR)

El protocolo admite un mecanismo de control de velocidad de datos incorporado que permite que los nodos finales sean configurados con diferentes tasas de datos dependiendo de la condición de la red. Sin embargo, el protocolo no define un algoritmo para controlar las tasas de transmisión de los nodos, el servidor de red tiene que implementar dependiendo de las condiciones. Objetivo de ADR para reducir el tiempo de transmisión de tramas ,ya que los nodos más cercanos a GW pueden usar valores de SF más bajos y tener tasas de transmisión más altas que reducirán el uso del canal y el consumo de energía. El proceso ADR se inicia con el comando LinkADRReq dados los parámetros en la figura18. (Blokdyk, LPWAN A Complete Guide, 2019).

Figura 18: *Enlace ADRReq.*



Fuente: (Singh, 2019).

The Things Network ilustra un mecanismo ADR. Calcula la tasa de datos óptima a través de 20 UL más recientes mediciones. Incluye un contador de tramas, SNR y número de GW que recibió el enlace UP. Cuando Se determina el ADR, el sistema inicia un nuevo proceso de medición para determinar el siguiente valor de ADR. ADR es una característica importante en las redes LoRaWAN para controlar la cobertura, la interferencia y la energía consumo. Sin embargo, los investigadores aún no han investigado completamente la ADR (M. Ertürk et al., 2019).

2.9. SEGURIDAD EN LPWAN

LoRaWAN tiene seguridad incorporada para proteger el protocolo de red y los datos del usuario. Datos a través de nodo a aplicación y nota a servidor de red están protegidos por encriptación AES 128 por cuatro componentes(Gomez, 2017):

- DevAddr: un identificador de dispositivo de 32 bits.
- AppEUI: un idenntificador único de aplicación que es un espacio de direcciones IEEE UI64.
- NwksKey: una clave de sesión de red utilizada para cifrar la comunicación entre el dispositivo final y el servidor de red.



- AppSKey: una clave de sesión de aplicación (clave AES-128) utilizada para proteger datos específicos de la aplicación.

LoRaWAN admite dos métodos principales de activación de dispositivos finales: activación por aire (OTAA) y activación por personalización (ABP) (M. Ertürk et al., 2019).

2.9.1. Adquisición y seguimiento de balizas

Según Singh (2019) los dispositivos finales deben unirse a la red antes de la etapa de transmisión de datos específica de la aplicación. Los nodos finales inician el proceso de activación con una solicitud de unión sin cifrar, con AppEUI (8 octetos), DevEUI (8 octetos) y DevNonce (2 octetos). DevNonce es un generador aleatorio y GW realiza un seguimiento de estos valores para cada nodo final. Si el servidor de red responde a la solicitud de unión con una aceptación de unión, el nodo final puede unirse a la red. (Singh, 2019) Los mensajes sin respuesta indican que el nodo final no puede participar en la red. Unirse Aceptar mensaje contiene; un AppNonce aleatorio (3 octetos), Identificador de red—NetID (3 octetos), una dirección de nodo final: DevAddr (4 octetos), un retraso RxDelay entre TX y RX (1 octeto), Parámetros de configuración de DL DLSettings (1 octeto) y lista de frecuencias de canal opcional CFList. El mensaje de solicitud de unión inicial enviado desde un nodo no está cifrado, sin embargo, el mensaje de respuesta del servidor de red (mensaje de aceptación de unión) está encriptado con AppKey. NwkSKey y AppSKey son calculados de la siguiente manera, vea la figura 19.

Figura 19: *Calculo NwkSKey y AppSKey*

$$NwkSKey = ases128_encrypt(AppKey, 0x01|AppNonce|NetID|DevNonce|pad_{16})$$

$$AppSKey = ases128_encrypt(AppKey, 0x02|AppNonce|NetID|DevNonce|pad_{16})$$

Fuente: (Singh, 2019).

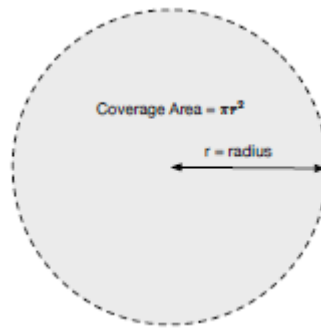
2.9.2. Activación por Personalización (ABP)

En modo ABP, los dispositivos finales almacenan DevAddr, NwkSKey y AppSKey para eliminar Join Etapas de solicitud/unión de respuesta. Estos dispositivos pueden comunicarse directamente con el servidor de red con llaves dadas. Almacenar y asegurar claves críticas es una tarea vital, cualquier ataque de hardware puede comprometer estas claves y pueden ocurrir accesos no deseados a la red. Además, ABP une estrictamente los nodos finales al Servidor de red específico donde estos tipos de dispositivos no pueden beneficiarse de las funciones de roaming disponibles (Singh, 2019).

2.9.3. Simulación LoRaWAN

LoRaWAN diseñado para simplificar el control de acceso medio (MAC) para reducir la complejidad en la red y los nodos finales. Sin embargo, este enfoque puede conducir a problemas de rendimiento cuando un alto número de nodos activos participan en la red. Por estas razones, hemos implementado un MAC simulación de capas del protocolo usando Matlab para probar la eficiencia (Singh, 2019).

Figura 20:Área de cobertura ilustrada



Fuente: (Singh, 2019).

Tabla 8:Frecuencias de los canales.

Canal NB del “beacon”	Frecuencia (Mhz)
0	923.3
1	923.9
2	924.5
3	925.1
4	925.7
5	926.3
6	926.9
7	927.5

Fuente: (Chou, 2016).



2.10. SERVIDOR DE RED LORAWAN

El servidor de red es el corazón del protocolo LoRaWAN, donde se controla toda la funcionalidad MAC por el servidor de red. El servidor de red es el punto de terminación de la capa MAC de LoRaWAN para los nodos finales y responsable de:

- Administrar toda la funcionalidad de la capa MAC.
- Gestión de solicitudes de unión, respuestas de unión y roaming.
- Control de dirección de nodo final.
- Contador de fotogramas y controles de autenticación.
- Mecanismo de reconocimiento.
- Adopción de velocidad de datos.
- Entrega de datos a servidores de aplicaciones.
- Entrega de datos a dispositivos finales desde servidores de aplicaciones.

Al abarcar los servidores de red LoRaWAN y herramientas disponibles en la actualidad. El crecimiento en el ecosistema de IoT lideró el aumento en las comunidades donde voluntarios individuales compartir sus conocimientos, intereses, infraestructura (conexión a internet) y otros recursos. Con uso público, The Things Network (TTN) es una de las conocidas comunidades de IoT que tiene como objetivo construir una comunidad LoRa global abierta para proporcionar una amplia cobertura. TTN tiene soluciones comerciales y gratuitas, sin embargo, proporcionar un servidor de red gratuito tomó la atención de aficionados, investigadores y creadores para sumergirse rápidamente en LoRaWAN. La comunidad tiene documentos de ayuda y apoyo, foros y voluntarios que se ayudan mutuamente. Además, servidor de red TTN está disponible como código abierto para iniciar redes LoRaWAN privadas. Presentan un protocolo abierto alternativo



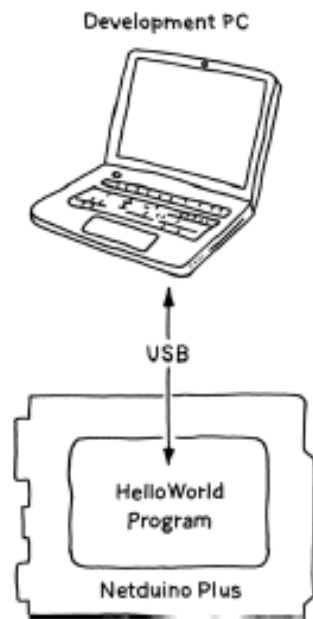
openChirp, que tiene un propósito para proporcionar servicios de red LoRaWAN simplificados como; serialización de datos, metadatos y tiempo, almacenamiento de datos en serie (M. Ertürk et al., 2019).

2.11. ENTORNO DE PROGRAMACIÓN

Debido a un hardware tan económico y un desarrollo fácil de usar plataformas, ahora es posible que los aficionados creen sistemas que interactúen con el mundo físico de todas las formas imaginables. Por ejemplo, un sensor puede medir la humedad en una maceta y una válvula controlada por computadora (actuador) deja pasar agua a la olla cuando la humedad baja demasiado. Además, dado que el hardware permite el uso de protocolos estándar de Internet. El seguimiento y control se puede realizar a través de Internet. Los servicios se pueden utilizar para almacenar datos, visualizarlos, compartirlos con otras personas, etc. Por ejemplo, para aprender sobre los efectos estacionales en la humedad, puede almacenar mediciones de la humedad de su maceta en el transcurso de un año. Si bien estas posibilidades son fascinantes y prometedoras, también hay algo espeluznante sobre el potencial de los dispositivos para espiar cada uno de nosotros. Esto proporciona otra razón por la que deberíamos tratar de aprender cómo tales funcionan los sistemas. Esta comprensión es, o al menos debería ser, la base por pensar en políticas de privacidad que serán necesarias tarde o temprano luego (Pfister, 2011).

Para familiarizarlo con el entorno de desarrollo, su primer programa debería ser un simple HelloWorld. Porque la placa Netduino Plus no tiene una pantalla, use la conexión USB entre la placa y la PC de desarrollo para escribir la cadena Hello World en la salida del entorno de desarrollo ventana que se ejecuta en la PC, como se ilustra en la Figura 21. La conexión USB es se usa para implementar y depurar sus programas, y en el ejemplo de HelloWorld, le permite enviar la cadena Hello World a su PC de desarrollo (Pfister, 2011).

Figura 21: *Arquitectura de conexión para el entorno de desarrollo.*

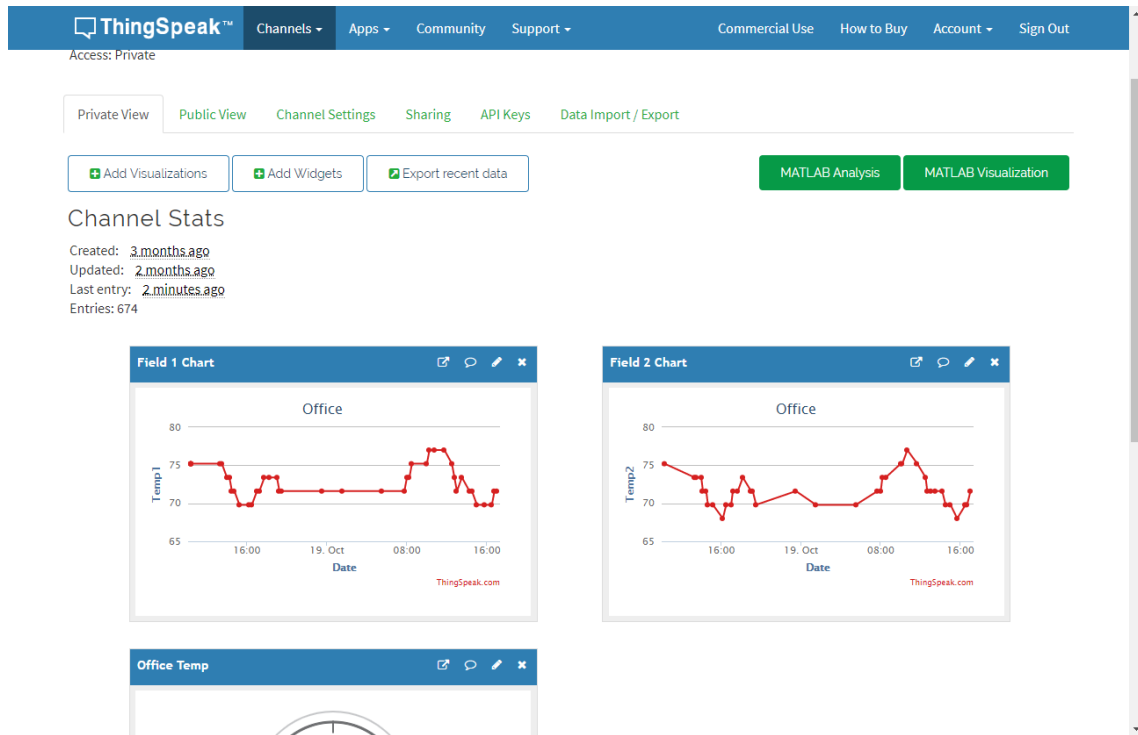


Fuente: (Pfister, 2011).

2.12. DATOS EN LA RED

Imagina que los dispositivos utilizan un sensor para tomar medidas periódicamente. Después de cada medición, un sistema de medición como en el caso propuesto del trabajo inmediatamente envía la muestra a un servidor para su almacenamiento y posterior recuperación. Este servidor proporciona efectivamente un recurso de fuente en el que publica sus datos muestras. Es posible que ya conozca el concepto de feeds de RSS feed lectores, una entrada de noticias puede ser cualquier cosa de interés, desde noticias políticas hasta Entradas de blog a medidas, como es el caso de los dispositivos que recolectan datos a través de sensores. De alguna manera, ello que contiene mediciones que puede considerarse como una noticia, en este caso sería la fuente sobre el mundo físico. Para tal ejemplo, necesita un servicio web adecuado al que su dispositivo puede enviar sus medidas. Convenientemente, hay un servicio gratuito, thingspeak, que hace exactamente esto, proporciona interfaces basadas en web para almacenar y acceder a los feeds (Pfister, 2011).

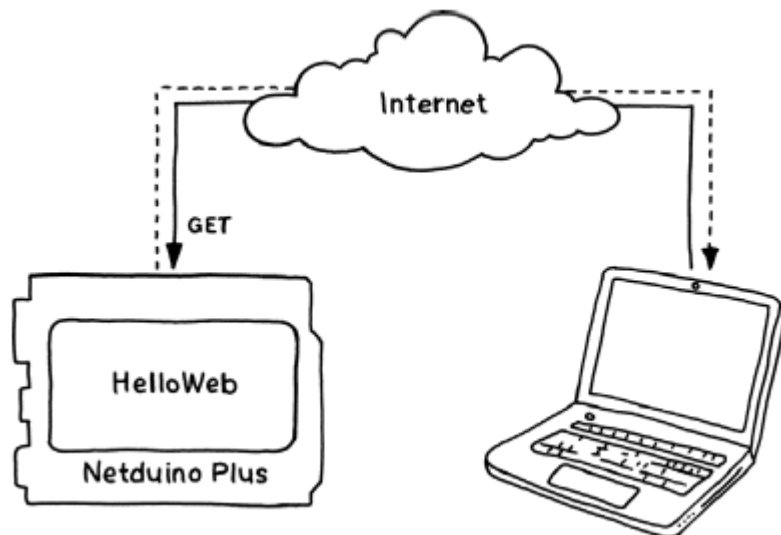
Figura 22:Entorno de supervisión.



Fuente: (Pfister, 2011).

En la Web, el equivalente es un programa de servidor que maneja OBTENER solicitudes de navegadores web y les devuelve un mensaje.

Figura 23:Modelo de comunicación con la nube.

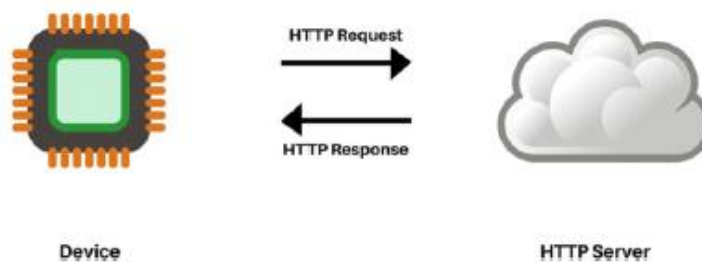


Fuente: (Pfister, 2011)

2.13. PROTOCOLOS DE COMUNICACIÓN

La web utiliza el Protocolo de transferencia de hipertexto (HTTP) como protocolo subyacente. HTTP admite múltiples métodos de transmisión de datos, pero en este proyecto va a escribir código para los dos métodos más populares, GET y POST. Los métodos GET y POST hacen el mismo trabajo y su código es muy similar, pero hay una ligera variación en su solicitud de formatos. GET tiene un límite en la cantidad de datos que puede transmitir en comparación con POST, que no tiene tales limitaciones. POST también se considera más seguro en comparación con GET. Basado en tus requisitos, puede decidir cuál funciona mejor para usted. La figura 24 muestra una interacción de alto nivel entre un dispositivo y un servidor HTTP (Javed, 2016).

Figura 24: Protocolo de transferencia de hipertexto (HTTP).



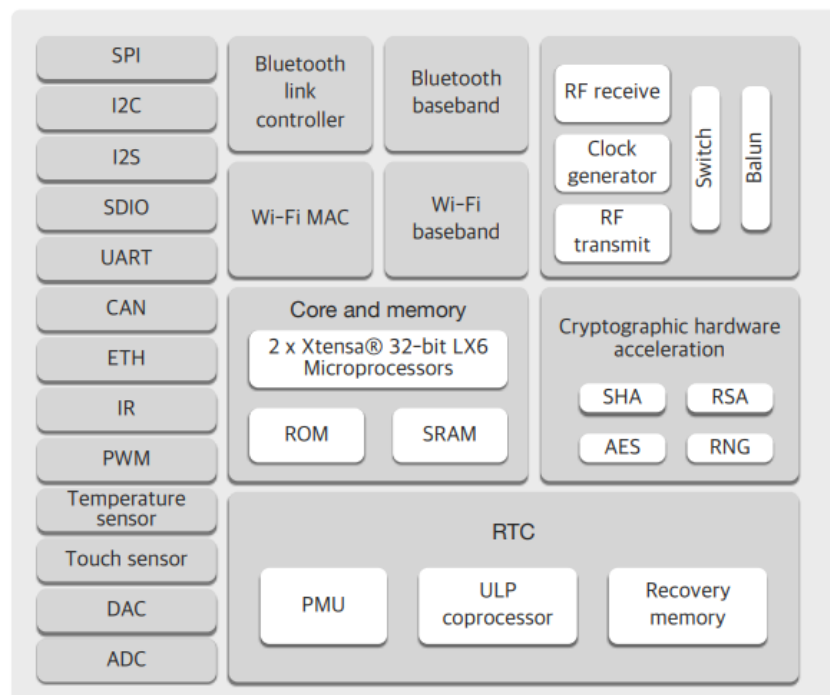
Fuente: (Javed, 2016).

2.14. ESP32

ESP32 es un único chip combinado de wifi y bluetooth de 2,4 GHz diseñado con el TSMC de 40 nm de potencia ultra baja tecnología. Está diseñado para lograr el mejor rendimiento de potencia y RF, mostrando robustez, versatilidad y confiabilidad en una amplia variedad de aplicaciones y escenarios de energía. ESP32 está diseñado para aplicaciones móviles, dispositivos electrónicos portátiles e Internet de las cosas (IoT). Cuenta con todas las características de vanguardia de los chips de bajo consumo, que incluyen activación de reloj de grano fino, múltiples modos de alimentación, y escalado dinámico de potencia. Por ejemplo, en un escenario de aplicación de concentrador de

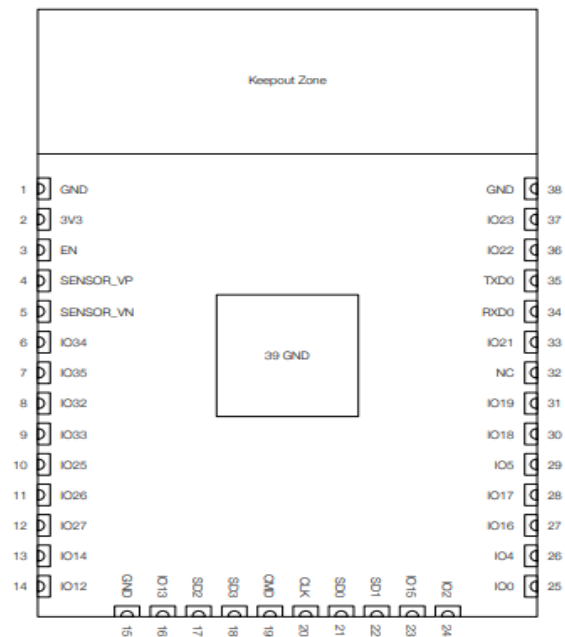
sensores IoT de baja potencia, ESP32 se activa periódicamente solo cuando se detecta una condición específica. El ciclo de trabajo bajo se utiliza para minimizar la cantidad de energía que gasta el chip. La salida del amplificador de potencia también es ajustable, lo que contribuye a una compensación óptima entre rango de comunicación, velocidad de datos y consumo de energía (Systems, 2022).

Figura 25: *Functional Block Diagram.*



Fuente: (https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf).

Figura 26: Pines de conexión esp32.



Fuente: (https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf).

2.14.1. Especificaciones técnicas

Las principales especificaciones técnicas de ESP32 en relación con conectividad, seguridad, CPU y memoria y otras funcionalidades son las siguientes:

WIFI

- 802.11 b/g/n/e/i
- 802.11 n (2.4 GHz), up to 150 Mbps
- 802.11 e: QoS for wireless multimedia technology
- WMM-PS, UAPSD
- A-MPDU and A-MSDU aggregation
- Block ACK
- Fragmentation and defragmentation
- Automatic Beacon monitoring/scanning



- 802.11 i security features: pre-authentication and TSN
- Wi-Fi Protected Access (WPA)/WPA2/WPA2-Enterprise/Wi-Fi Protected Setup (WPS)
- Infrastructure BSS Station mode/SoftAP mode
- Wi-Fi Direct (P2P), P2P Discovery, P2P Group Owner mode and P2P Power Management
- UMA compliant and certified
- Antenna diversity and selection

BLUETOOTH

- Compliant with Bluetooth v4.2 BR/EDR and BLE specification
- Class-1, class-2 and class-3 transmitter without external power amplifier
- Enhanced power control
- +10 dBm transmitting power
- NZIF receiver with -98 dBm sensitivity
- Adaptive Frequency Hopping (AFH)
- Standard HCI based on SDIO/SPI/UART
- High speed UART HCI, up to 4 Mbps
- BT 4.2 controller and host stack
- Service Discover Protocol (SDP)
- General Access Profile (GAP)
- Security Manage Protocol (SMP)
- Bluetooth Low Energy (BLE)
- ATT/GATT
- HID



- All GATT-based profile supported
- SPP-Like GATT-based profile
- BLE Beacon
- A2DP/AVRCP/SPP, HSP/HFP, RFCOMM
- CVSD and SBC for audio codec
- Bluetooth Piconet and Scatternet

CPU Y MEMORIA

- Xtensa® Dual-Core 32-bit LX6 microprocessors, up to 600 DMIPS
- 448 KByte ROM
- 520 KByte SRAM
- 16 KByte SRAM in RTC
- QSPI Flash/SRAM, up to 4 x 16 MBytes
- Power supply: 2.2 V to 3.6 V

CLOCKS AND TIMMERS

- Internal 8 MHz oscillator with calibration
- Internal RC oscillator with calibration
- External 2 MHz to 40 MHz crystal oscillator
- External 32 kHz crystal oscillator for RTC with calibration
- Two timer groups, including 2 x 64-bit timers and 1 x main watchdog in each group
- RTC timer with sub-second accuracy
- RTC watchdog



SEGURIDAD

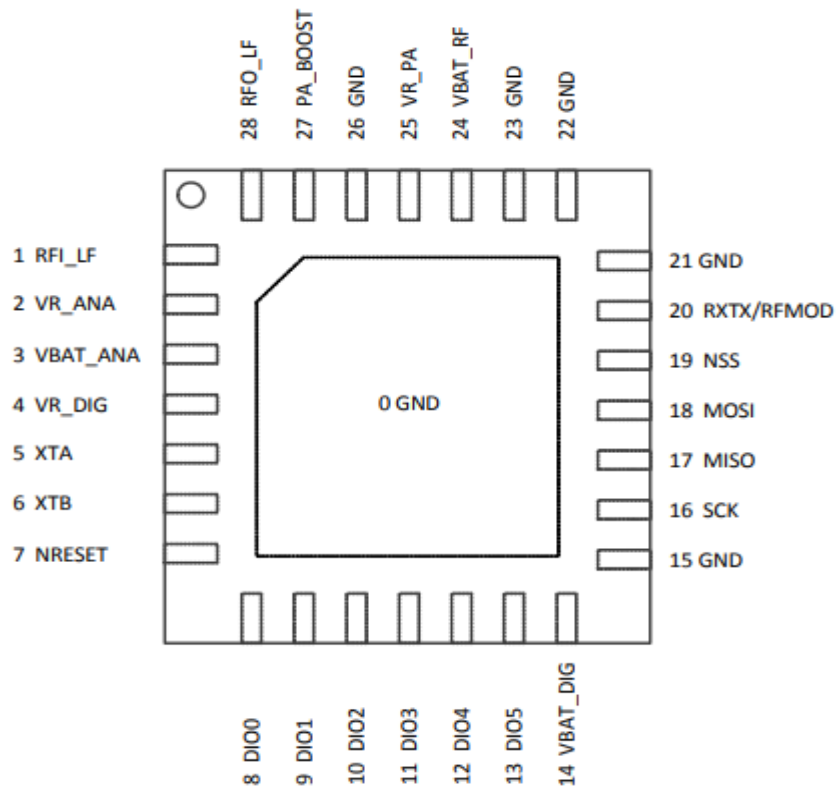
- IEEE 802.11 standard security features all supported, including WFA, WPA/WPA2 and WAPI
- Secure boot
- Flash encryption

2.15. CHIP LORA

2.15.1. LoRa SX1276

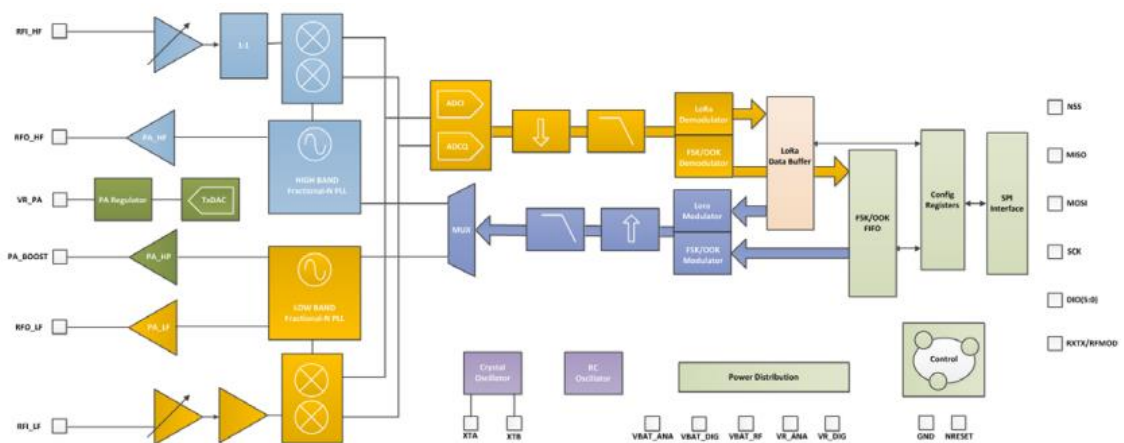
Los chips LoRa vienen diversificados en diferentes versiones, estableciendo así sus versiones de los sx1276, sx1277 sx1278 sx1279, la característica fundamental de los chips es la comunicación a largo alcance y la alta inmunidad a interferencias que se puedan presentar a los momentos de una comunicación, para lograr la comunicación utiliza la técnica de modulación LORA esta fue patentada por semtech con una sensibilidad de más de 148 dBm. Esta sensibilidad combinada con la potencia de +20 dBm permiten que el amplificador integrado trabaje en sus máximas características. Los dispositivos admiten FSK con un gran alto rendimiento que incluyen IEEE802.15.4g. Los chips de la serie SX ofrecen un ruido de fase excepcional, selectividad, linealidad del receptor y IIP3 para significativamente menor consumo actual que otros dispositivos (Semtech, 2016).

Figura 27: Pines de chip LoRa.



Fuente(https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf).

Figura 28: Diagrama de bloque chip LoRa.



Fuente: (<https://www.semtech.com/legal>).

2.15.2. Sensor de Corriente

Los sensores de corriente permiten recolectar y/o controlar el consumo y pérdidas en el suministro de energía eléctrica. Teóricamente, son dispositivos que miden el flujo

de corriente de un determinado circuito, también se denomina transductores de corriente (logicbus, s. f.).

Figura 29: *Sensor de corriente.*

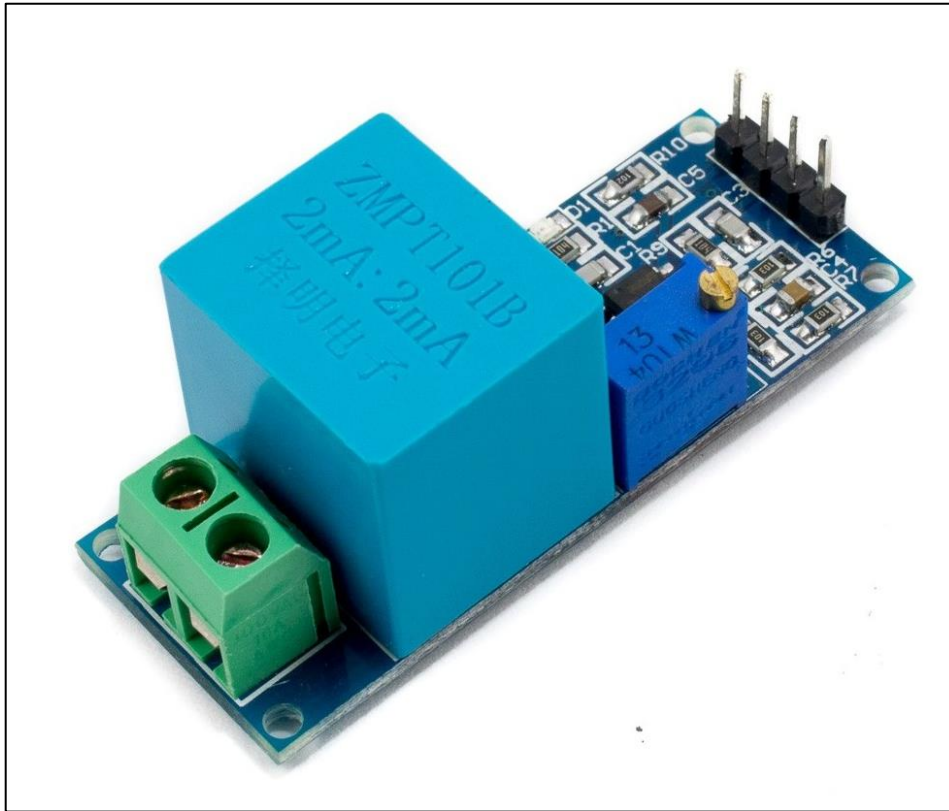


Fuente: (logicbus, s. f.).

2.15.3. Sensor de Voltaje

El sensor de voltaje son dispositivos que permiten realizar el seguimiento de los niveles de voltaje de alimentación. Estos sensores básicamente detectan e identifican señales eléctricas. La implementación de este sensor es aplicable para el proyecto de sistema de supervisión de medidores eléctricos (unitelectronics, s. f.).

Figura 30: Sensor de voltaje.



Nota. Sensor de voltaje alterna ZMPT101B.



CAPITULO III

MATERIALES Y MÉTODOS

3.1. MATERIALES

3.1.1. Hardware

Tabla 9: *Lista de materiales hardware*

DESCRIPCIÓN	CANTIDAD	PRECIO S/.
Ordenador	1 unid	2000
Modulo TTGO LoRa Esp32 v2.0 + Arduino	2 unid	100
Cable UTP Cat5	2 metros	16
Antena RF 3dbi	2 unid	30
Sensores	3 unid	50
Batería para módulos TTGO LoRa	2 unid	30
Cables para conexión	50 unid	10

Fuente: Elaboración grupal.

3.1.2. Software

Tabla 10: *Lista de materiales software.*

DESCRIPCIÓN	CANTIDAD	PRECIO S/.
Sistema Operativo Windows 10	1 unid	0
Software Arduino IDE	1 unid	0
Plataformas tecnológicas	1 unid	0

Fuente: Elaboración grupal.

3.2. DISEÑO, NIVEL Y TIPO DE LA INVESTIGACIÓN

3.2.1. Diseño de la investigación

Para la presente investigación se hace uso de la investigación experimental. Pues la investigación requiere de pruebas fuera de laboratorios, ya que haciendo uso de los prototipos se obtiene los datos e información que se requerirá para demostrar las hipótesis planteadas, cabe recalcar que los estudios de experimentales abordan los problemas de investigación. La gran ventaja de una investigación en un campo abierto permite la experimentación (Hernandez et al., 2014).

3.2.2. Nivel de la investigación

Según Hernández (2014) al referirnos al nivel de investigación se refiere al grado de conocimiento que se desea lograr con relación al planteamiento del problema. Para el presente estudio, se utiliza el nivel de investigación exploratoria, este es un tipo de investigación que se utiliza para estudiar un problema que no está claramente definida o no se tiene mucha información, debido a ello se lleva a cabo para comprenderlo mejor, y para el caso de la investigación el nivel exploratorio busca abrir y centrar nuevos conocimientos sobre la supervisión de medidores eléctricos utilizando IoT y tecnología LoRa.



3.3. POBLACIÓN Y MUESTRA

La población y muestra está representada por el prototipo implementado utilizando IoT y tecnología LoRa para la supervisión de muchos medidores eléctricos, ya que es el objeto de estudio de la presente investigación.

3.4. UBICACIÓN Y DESCRIPCIÓN DE LA INVESTIGACIÓN

3.4.1. Ubicación

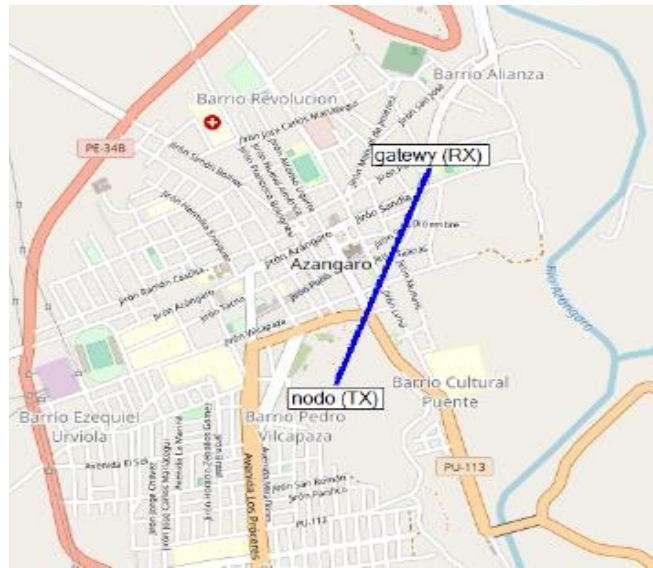
La investigación se llevó a cabo en el departamento de Puno, Provincia de Azángaro y distrito de Azángaro.

Tabla 11: *ubicación y elevación de los nodos Rx y Tx*

Nodo Tx	Nodo Rx
Latitud: -14.914221°	Latitud: -14.905819°
Longitud: -70.195472°	Longitud: -70.192137°
Elevación de terreno:3962.5m	Elevación de terreno:4793.9m

Fuente: Elaboración grupal.

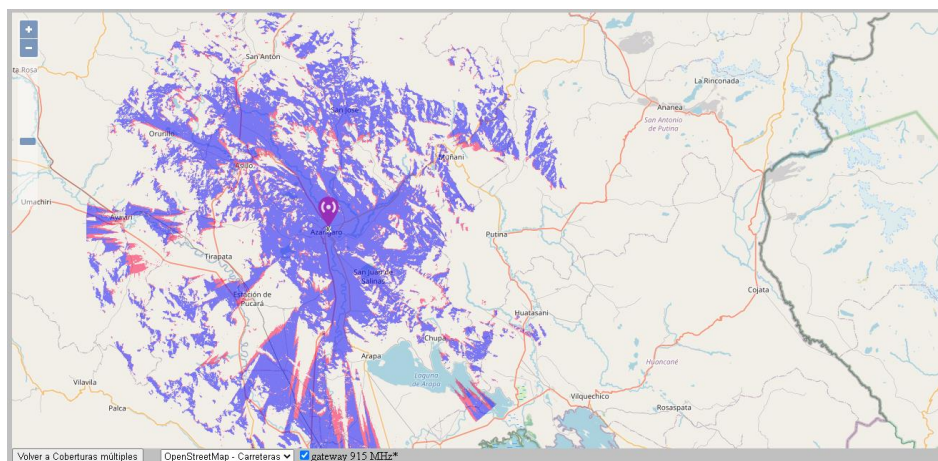
Figura 31: *Ubicación del nodo Tx y del servidor y nodo Rx*



Fuente: Elaboración grupal.

Nota. Para la ubicación del nodo receptor se tomó en cuenta el lugar más alto de la ciudad Azángaro con el fin de que tenga mayor cobertura y abarque mayor enlace con los nodos Tx.

Figura 32: *Rango de cobertura teórico de LoRa del nodo*



Fuente: Elaboración grupal.

Figura 33: *Ubicación del nodo Rx y Tx vista satelital.*



Fuente: Elaboración grupal.

3.5. TÉCNICAS E INSTRUMENTOS DE RECOLECCIÓN DE DATOS

Según Hernández-Sampieri et al (2018) recolección de datos utilizados para la investigación son las herramientas para obtener la información que será útil para cumplir nuestros objetivos, la cual lo presentamos en la tabla 12.

Tabla 12: *Técnicas e instrumentos de recolección de datos.*

Técnicas	Instrumentos
Consulta de bibliográficas	Tesis, Papers, Libros, Revistas, blogs.
Prueba de funcionamiento	Módulo TTGO LoRa32 TX RX, Arduino
Recolector de datos	Sensores de voltaje y corriente
Observación	Software de Arduino, capturas de pantalla, fotos.

Fuente: Elaboración grupal.

3.6. DISEÑO DEL PROTOTIPO

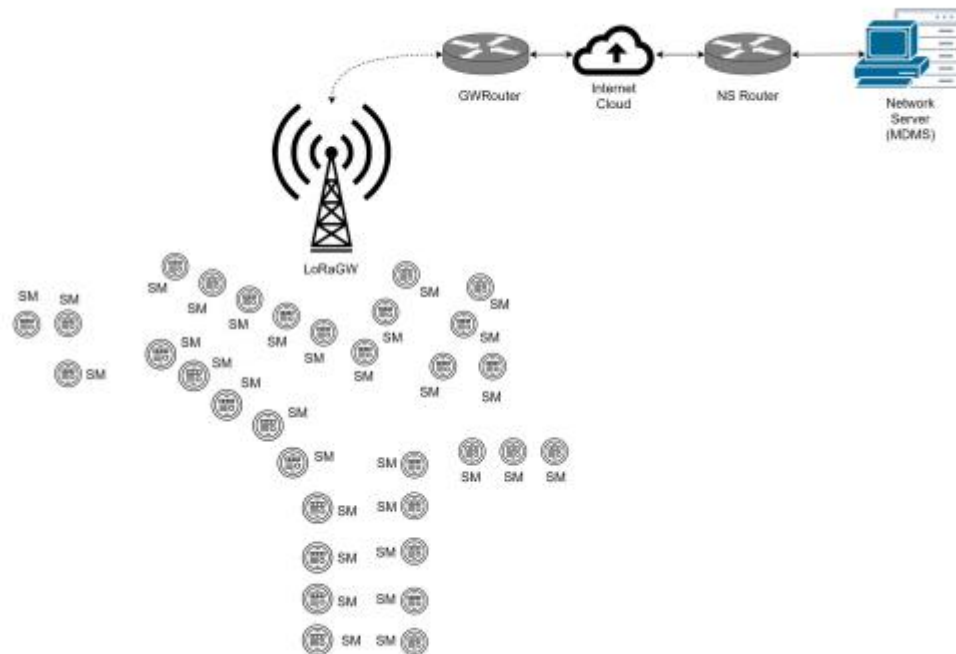
Para el diseño se optó por utilizar microcontroladores como el ESP32 y el Arduino uno, por su fácil aplicación, siguiendo un modelo de conexión y programación el cual nos permite gestionar y controlar las fases del proyecto. La metodología de diseño está orientada al desarrollo o procesamiento del software y hardware, cabe mencionar que el desarrollo del software no solo es para el presente trabajo sino también para otros proyectos.

Esta metodología de implementación de prototipo nos permite evaluar validar hipótesis y hacer pruebas para que futuros usuarios para puedan acceder o utilizarlo. En el proceso de diseño se tuvo en consideración lo siguiente:

- Selección de los microcontroladores tecnológicos idóneos para el diseño e implementación de un prototipo utilizando IoT para supervisión de medidores eléctricos con tecnología LoRa.
- Selección de los sensores recolectores del voltaje y corriente, los cuales permiten recolectar la información para el procesamiento de datos.

- El diseño de Red LoRa que soporta la comunicación punto a punto. Mediante el cual se transmite la información del medidor.
- Diseñar la plataforma del sistema de supervisión de los medidores eléctricos utilizando tecnología LoRa.

Figura 34: *diagrama de red y topología de implementación LoRa*

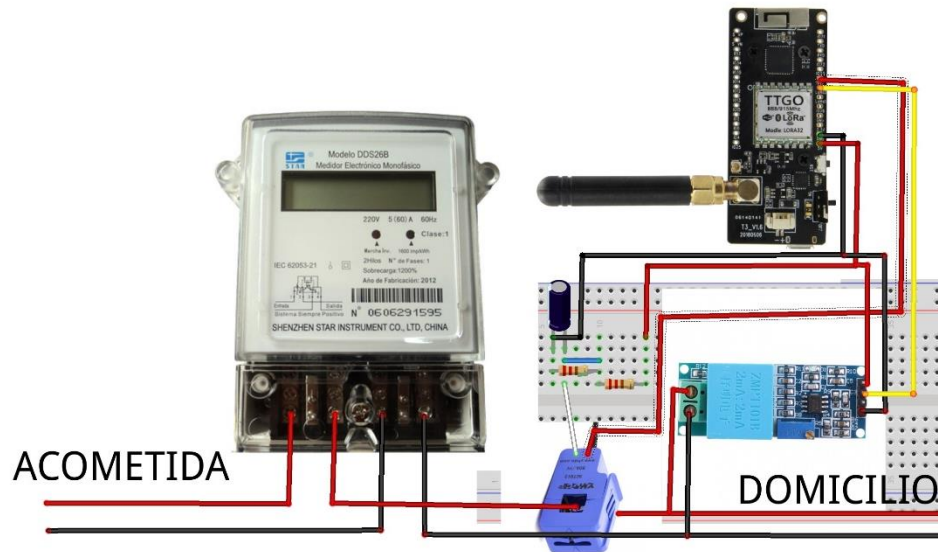


Fuente: Elaboración grupal.

3.6.1. Desarrollo del hardware

Para el desarrollo del hardware del proyecto de supervisión de medidores eléctricos utilizando la tecnología LoRa, se será necesario la implementación de los sensores conjuntamente con los módulos TTGO LoRa ESP32. Para calcular la potencia consumida de un determinado circuito se deberá utilizar sensores de medición de corriente y voltaje.

Figura 35: Diagrama de conexión de sensores a medidor eléctrico



Fuente: Elaboración grupal.

La conexión se muestra en la figura 35 en donde la conexión de los sensores son la siguientes el sensor de corriente se conecta mediante dos resistencias de 10kohm, y un condensador de 0.1uF, y conectado al pin IO.23. el sensor de voltaje es conectado en paralelo con el circuito que sale del medidor, y es conectado al pin IO.12 del módulo, con respectiva alimentación.

En la figura 36 y 37 se muestra la declaración en el IDE Arduino de los sensores de corriente y voltaje por pines análogos en el microcontrolador, los sensores medirán supervisarán la corriente y voltaje así nos permitirá supervisar ambas variables y en consecuencia la potencia.

Figura 36: Programación para sensor de corriente.

```
void setup() {  
  Serial.begin(9600);  
  analogReference(INTERNAL);  
  
void loop() {  
  float Irms=get_corriente(); //Corriente eficaz (A)  
  float P=Irms*220.0; // P=IV (Watts)  
  Serial.print("Irms: ");  
  Serial.print(Irms,3);  
  Serial.print("A, Potencia: ");  
  Serial.print(P,3);  
  Serial.println("W");  
}  
float get_corriente()  
{  
  float voltajeSensor;  
  float corriente=0;  
  float Sumatoria=0;  
  long tiempo=millis();  
  int N=0;  
  while(millis()-tiempo<500)//Duración 0.5 segundos(Aprox. 30 ciclos de 60Hz)  
  {  
    voltajeSensor = analogRead(A0) * (1.1 / 1023.0);////voltaje del sensor  
    corriente=voltajeSensor*30.0; //corriente=VoltajeSensor*(30A/1V)  
    Sumatoria=Sumatoria+sq(corriente);//Sumatoria de Cuadrados  
    N=N+1;  
    delay(300);  
  }  
  Sumatoria=Sumatoria*2;//Para compensar los cuadrados de los semiciclos negativos.  
  corriente=sqrt((Sumatoria)/N); //ecuación del RMS  
  return(corriente);  
}
```

Fuente: Elaboración grupal.

Figura 37: Programación para sensor de voltaje.

```
int decimalPrecision = 2;
int VoltageAnalogInputPin = A2;
float voltageSampleRead = 0;
float voltageLastSample = 0;
float voltageSampleSum = 0;
float voltageSampleCount = 0;
float voltageMean ;
float RMSVoltageMean ;
float adjustRMSVoltageMean;
float FinalRMSVoltage;
    float voltageOffset1 =0.00 ;
    float voltageOffset2 = 0.00;
void setup() {
    Serial.begin(9600);}
void loop()
{
    if(micros() >= voltageLastSample + 1000 )
    {
        voltageSampleRead = (analogRead(VoltageAnalogInputPin)- 512)+ voltageOffset1;
        voltageSampleSum = voltageSampleSum + sq(voltageSampleRead) ;
        voltageSampleCount = voltageSampleCount + 1;
        voltageLastSample = micros() ;    }
    if(voltageSampleCount == 1000)
    { voltageMean = voltageSampleSum/voltageSampleCount;
      RMSVoltageMean = (sqrt(voltageMean))*1.5;
      adjustRMSVoltageMean = RMSVoltageMean + voltageOffset2;
      FinalRMSVoltage = RMSVoltageMean + voltageOffset2;
      if(FinalRMSVoltage <= 15)
      {FinalRMSVoltage = 0;}
      Serial.print(" Volt: ");
      Serial.print(FinalRMSVoltage,decimalPrecision);
      Serial.println(" V ");
      voltageSampleSum =0;
      voltageSampleCount=0; }
```

Fuente: Elaboración grupal.

3.6.2. Desarrollo del software

Con el desarrollo del software realizamos los cálculos necesarios para el prototipo una vez de obtenidas las señales de voltaje y corriente se procede a calcular la potencia demandada que pasa por circuito.

Se calculo con la siguiente formula:

$$P = V * I \quad (4)$$

Donde:

P: potencia demandada

V: voltaje de alimentación

I: corriente

Una vez definida la fórmula para el cálculo de la potencia procedemos a desarrollar el código de programación para el voltaje y la corriente y por consecuencia la potencia en el IDE Arduino.

El proceso de programación se inicia en el aplicativo IDE Arduino, con la declaración de la variable voltaje, en la figura 38 se especifica el pin analógico utilizado.

Figura 38: *Declaracion de Variables*

```
int decimalPrecision = 2;

int voltageOffset1 = 0;
int voltageOffset2 = 0;
int VoltageAnalogInputPin = A0;
float voltageSampleRead = 0;
float voltageLastSample = 0;
float voltageSampleSum = 0;
float voltageSampleCount = 0;
float voltageMean ;
float RMSVoltageMean ;
```

Fuente: Elaboración grupal.

El código se elabora para obtener la medición del voltaje, en esta etapa se elabora fórmula para el muestreo. En el a figura 39 se muestra el desarrollo para censar través del sensor de voltaje, el cual se definirá como una variable constante. También se puede apreciar la programación para que se pueda visualizar en el serial print del programa IDE Arduino.

Figura 39: *Bucle de adquisicion de datos voltaje.*

```
if(micros() >= voltageLastSample + 1000 )
{
    voltageSampleRead = (analogRead(VoltageAnalogInputPin)- 512) + voltageOffset1;
    voltageSampleSum = voltageSampleSum + sq(voltageSampleRead) ;
    voltageSampleCount = voltageSampleCount + 1;
    voltageLastSample = micros() ;
}

if(voltageSampleCount == 1000)
{
    voltageMean = voltageSampleSum/voltageSampleCount;
    RMSVoltageMean = sqrt(voltageMean)+ voltageOffset2;
    if(RMSVoltageMean <=2)
    {RMSVoltageMean = 0;}
    Serial.print("Volt: ");
    Serial.print(RMSVoltageMean, decimalPrecision);
    Serial.println("V ");
    voltageSampleSum =0;
    voltageSampleCount=0;
}
```

Fuente: Elaboración grupal.

A continuación, se implementa y se programa el sensor de corriente, se conecta al pin análogo para que se logre obtener los datos. En la figura 40 se describe la programación, también en se elabora el código para que el sensor logre descifrar los datos de corriente.

Figura 40: *Bucle de adquisicion de datos corriente.*

```
if(millis() >= currentLastSample + 1)
{
    currentSampleRead = analogRead(CurrentAnalogInputPin)-512 + currentOffset1;
    currentSampleSum = currentSampleSum + sq(currentSampleRead) ;
    currentSampleCount = currentSampleCount + 1;
    currentLastSample = millis();
}
```

Fuente: Elaboración grupal.

En la figura 41 se observa la programación para que la corriente censada se pueda visualizar o imprimir en el serial print del programa de IDE Arduino.

Figura 41: Impresión de datos en el IDE Arduino.

```
Serial.print("Current RMS: ");  
Serial.print(FinalRMSCurrent, decimalPrecision);  
Serial.println(" A ");  
currentSampleSum =0;  
currentSampleCount=0;
```

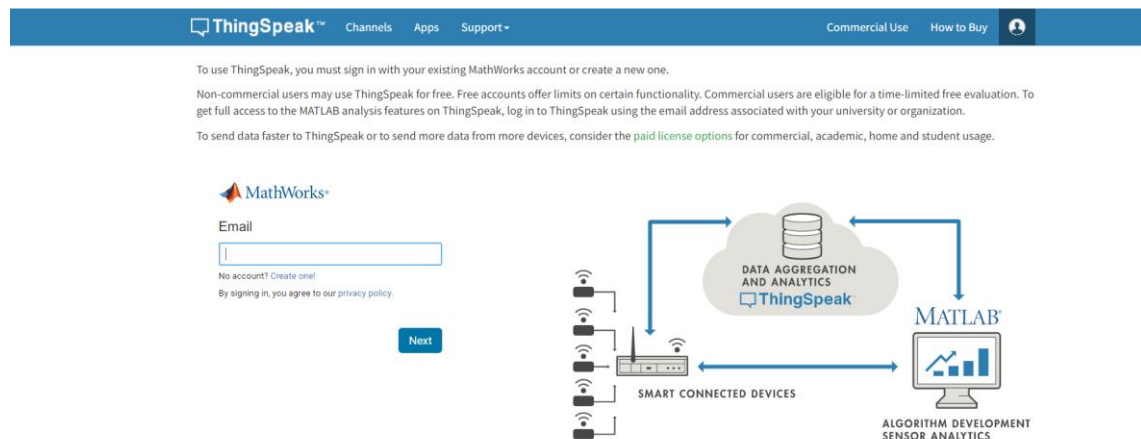
Fuente: Elaboración grupal.

3.6.3. Desarrollo de la plataforma de supervisión

Para el sistema de supervisión planteado se realiza mediante la implementación en una plataforma IoT thingspeak conjuntamente con el Arduino y un módulo ethernet shield. El cual gestionara los datos obtenidos en la adquisición de los datos.

En la figura 42 se puede apreciar la configuración establecida para acceder a la plataforma IoT. Iniciamos creando el usuario para tener acceso al sistema de supervisión de medidores eléctricos.

Figura 42: Crear cuenta de acceso a thingspeak.



Fuente: Elaboración grupal.

En la figura 43 se muestra la configuración del email para el acceso también se indica la localización del proyecto, asignamos el first Name y el last Name para que reconozca como usuario.

Figura 43.

Asignamos la localización del proyecto.

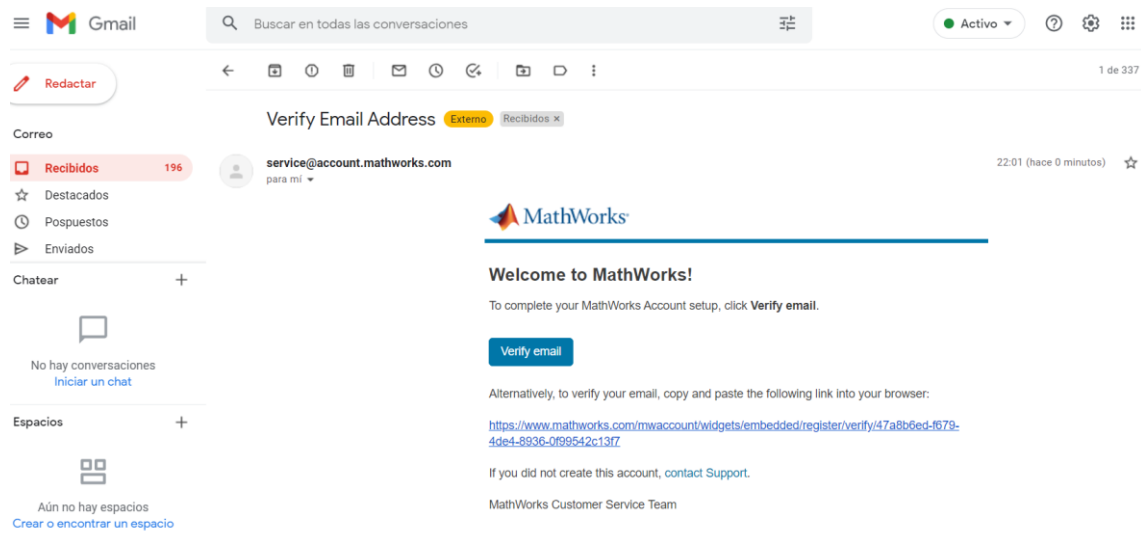
The image shows a screenshot of the ThingSpeak website's account creation process. The form is titled "Create MathWorks Account" and includes the following fields: "Email Address" (with the value "proy.lora@gmail.com" and a red error message: "A MathWorks Account already exists with that email address. Please choose a different email address or log in."), "Location" (a dropdown menu set to "Peru"), "First Name" (with the value "proy" and a green checkmark), and "Last Name" (with the value "lora lo" and a green checkmark). There are "Continue" and "Cancel" buttons at the bottom of the form. Below the form, a small text reads: "This site is protected by reCAPTCHA Enterprise and the Google Privacy Policy and Terms of Service apply."

To the right of the form is a diagram illustrating the IoT data flow. It shows "SMART CONNECTED DEVICES" (represented by several wireless signal icons) sending data to a central cloud labeled "DATA AGGREGATION AND ANALYTICS ThingSpeak". From this cloud, data is sent to a computer monitor labeled "MATLAB" with the sub-label "ALGORITHM DEVELOPMENT SENSOR ANALYTICS".

Fuente: Elaboración grupal.

Para continuar con la configuración de la plataforma de supervisión IoT thingspeak, se debe confirmar en el link de la descripción del correo ingresado, se hace clic para que el programa redirija la página, y verifique el correo anexo.

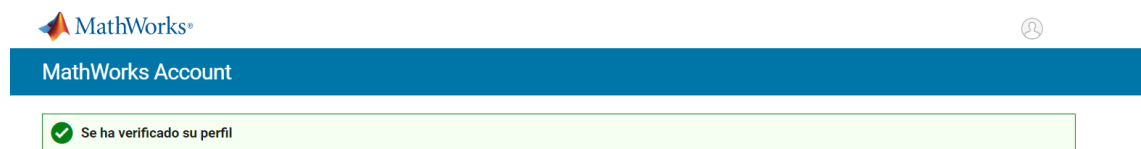
Figura 44: Verificación de correo.



Fuente: Elaboración grupal.

En la figura 45 se aprecia que el perfil o correo a sido verificado esto nos especifica que el correo ha sido aceptado por la plataforma IoT. Con la verificación se logra configurar el acceso asignando password y user.

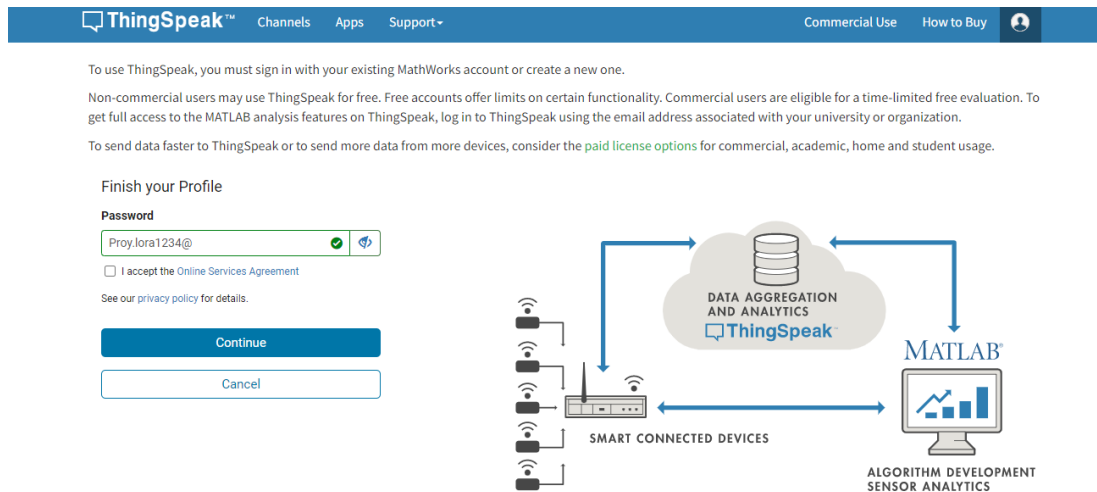
Figura 45; Correo verificado.



Fuente: Elaboración grupal.

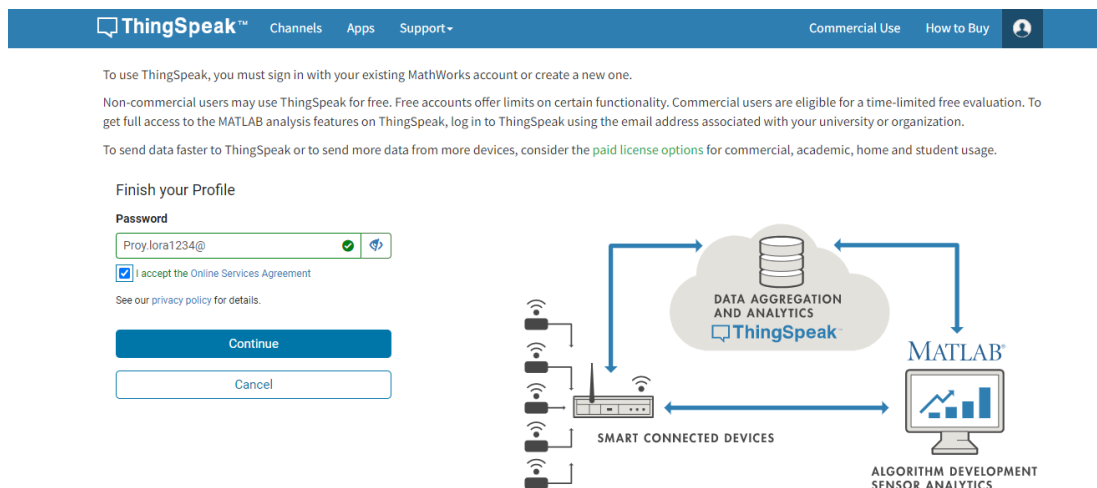
Para acceder a la plataforma IoT debemos asignar una contraseña que debe contener entre números una letra en mayúscula y símbolos, para tener la seguridad correspondiente, realizamos un check “yo acepto los servicios en línea y las políticas de privacidad” ver figura 46.

Figura 46:Asignación de una contraseña



Fuente: Elaboración grupal.

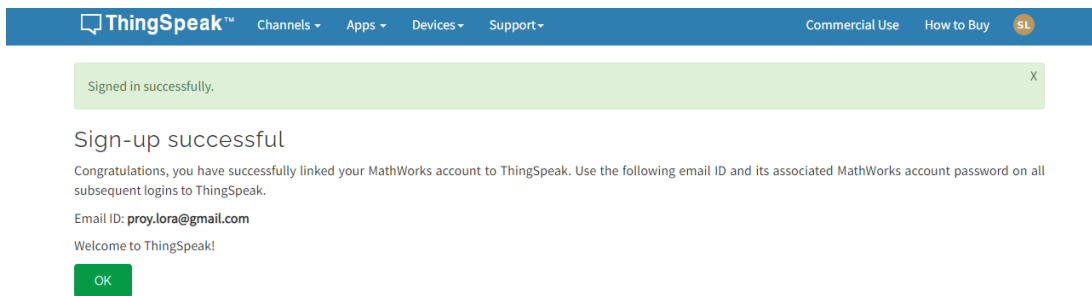
Figura 47:Políticas y privacidad de la plataforma.



Fuente: Elaboración grupal.

Después de crear la contraseña, se vincula correctamente la cuenta MathWorks a ThingSpeak. Para el acceso se utiliza el siguiente ID de correo electrónico y la contraseña de cuenta de MathWorks asociada en todos los inicios de sesión posteriores en ThingSpeak.

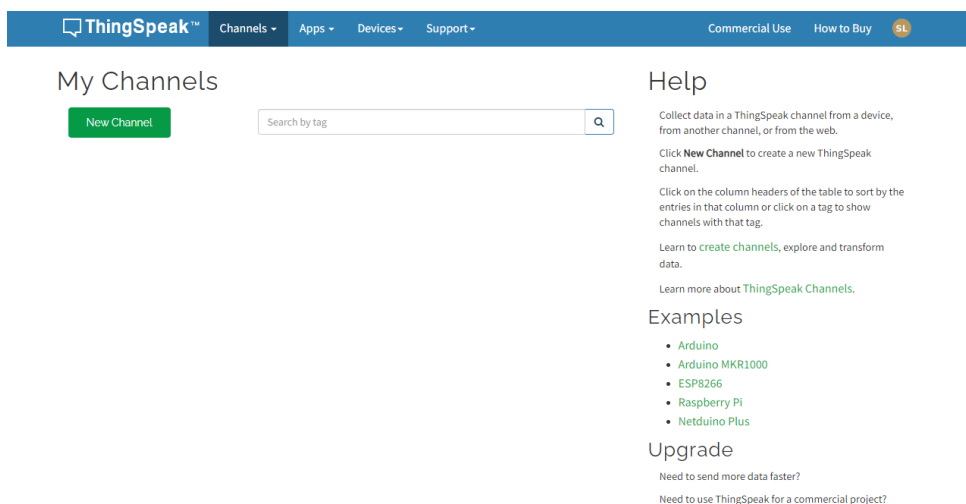
Figura 48: Mensaje de vinculacion de cuenta en ThingSpeak.



Fuente: Elaboración grupal.

En la figura 49 se muestra la ventana donde se recopila datos en un canal de ThingSpeak desde un dispositivo, desde otro canal o desde la web. En el icono New Channel se puede crear un nuevo canal de ThingSpeak. Para configurar las columnas hacer clic en los encabezados de la tabla para ordenar por las entradas en esa columna o hacer clic en una etiqueta para mostrar los canales con esa etiqueta.

Figura 49: Crear nuevo canal en ThingSpeak.



Fuente: Elaboración grupal.

En la figura 50 se puede apreciar los canales que almacenan todos los datos que recopila una aplicación ThingSpeak. Cada canal incluye ocho campos que pueden contener cualquier tipo de datos, más tres campos para datos de ubicación y uno para datos de estado. Una vez que recopila datos en un canal, puede usar las aplicaciones de ThingSpeak para analizarlos y visualizarlos. En esta etapa también se asigna los nombres de las variables voltaje, corriente y potencia, los cuales se visualizarán en el canal principal del ThingSpeak.

Figura 50: *Asignación de las variables corriente, voltaje.*

The screenshot shows the 'New Channel' page in the ThingSpeak interface. The form includes the following fields:

- Name:** SUPERVISION DE MEDIDORES ELECTRICOS
- Description:** INTERNET DE LAS COSAS
- Field 1:** Field Label 1 (checked)
- Field 2:** (empty)
- Field 3:** (empty)
- Field 4:** (empty)
- Field 5:** (empty)
- Field 6:** (empty)
- Field 7:** (empty)
- Field 8:** (empty)

The 'Channel Settings' section on the right provides the following instructions:

- Percentage complete:** Calculated based on data entered into the various fields of a channel. Enter the name, description, location, URL, video, and tags to complete your channel.
- Channel Name:** Enter a unique name for the ThingSpeak channel.
- Description:** Enter a description of the ThingSpeak channel.
- Field:** Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- Metadata:** Enter information about channel data, including JSON, XML, or CSV data.
- Tags:** Enter keywords that identify the channel. Separate tags with commas.
- Link to External Site:** If you have a website that contains information about your ThingSpeak channel, specify the URL.
- Show Channel Location:**
 - Latitude:** Specify the latitude position in decimal degrees. For example, the latitude of the city of London is 51.5072.

Fuente: Elaboración grupal.

En la figura 51 se aprecia el porcentaje completado, calculado en base a los datos ingresados en los distintos campos de un canal. Se ingrese el nombre, la descripción, la ubicación, la URL y las etiquetas del proyecto para completar la configuración del canal. Se ingresa el nombre del canal como supervisión de medidores eléctricos como nombre único para el canal de ThingSpeak. En la descripción se ingresa una descripción como internet de las cosas ThingSpeak. El field especifica el campo que medirá cada variable se marca tres casillas para habilitar el campo. Cada canal de ThingSpeak puede tener hasta 8 campos. Los metadatos ingresan información sobre los datos del canal, incluidos los datos JSON, XML o CSV. Las etiquetas introducen palabras clave que identifican el

canal. Se enlace el sitio externo con el prototipo de supervisión de medidores eléctricos el cual contiene información de los datos sobre su canal de ThingSpeak.

Figura 51:Asignación de datos en el canal Thingspeak.

New Channel

Name: SUPERVISION DE MEDIDORES ELECTRICOS

Description: INTERNET DE LAS COSAS

Field 1: VOLTAJE

Field 2: CORRIENTE

Field 3: POTENCIA

Field 4:

Field 5:

Field 6:

Field 7:

Field 8:

Help

Channels store all the data that a ThingSpeak application collects. Each channel includes eight fields that can hold any type of data, plus three fields for location data and one for status data. Once you collect data in a channel, you can use ThingSpeak apps to analyze and visualize it.

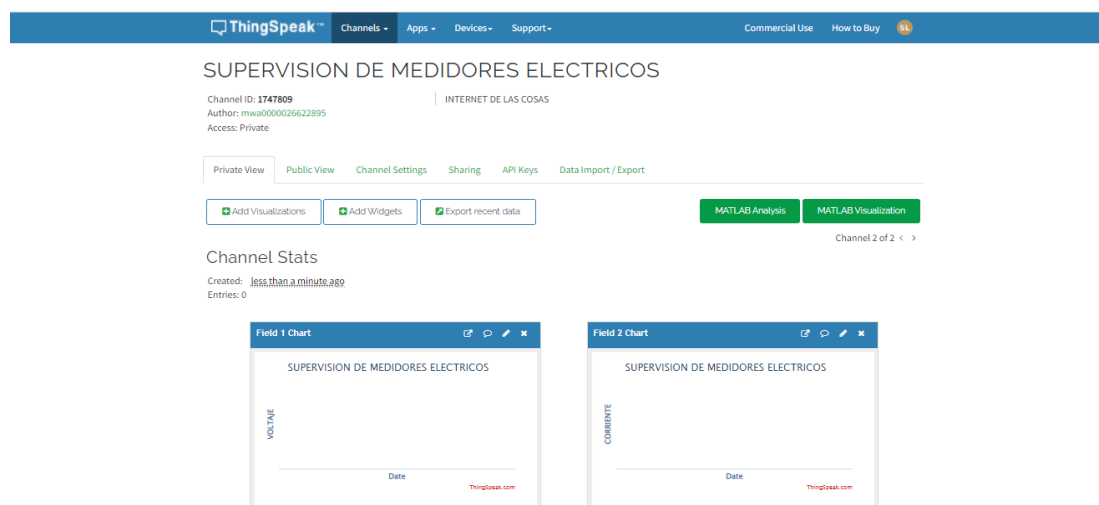
Channel Settings

- Percentage complete:** Calculated based on data entered into the various fields of a channel. Enter the name, description, location, URL, video, and tags to complete your channel.
- Channel Name:** Enter a unique name for the ThingSpeak channel.
- Description:** Enter a description of the ThingSpeak channel.
- Field#:** Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- Metadata:** Enter information about channel data, including JSON, XML, or CSV data.
- Tags:** Enter keywords that identify the channel. Separate tags with commas.
- Link to External Site:** If you have a website that contains information about your ThingSpeak channel, specify the URL.
- Show Channel Location:**
 - Latitude:** Specify the latitude position in decimal degrees. For example, the latitude of the city of London is 51.5072.

Fuente: Elaboración grupal.

Puede obtener los datos en el nuevo canal desde un dispositivo prototipo, se procede a conectar para que se logre monitorear supervisar la potencia. A continuación, puede visualizar los datos en tablas estadísticas del proyecto supervisión de medidores eléctricos en la plataforma IoT ThingSpeak ver figura 52.

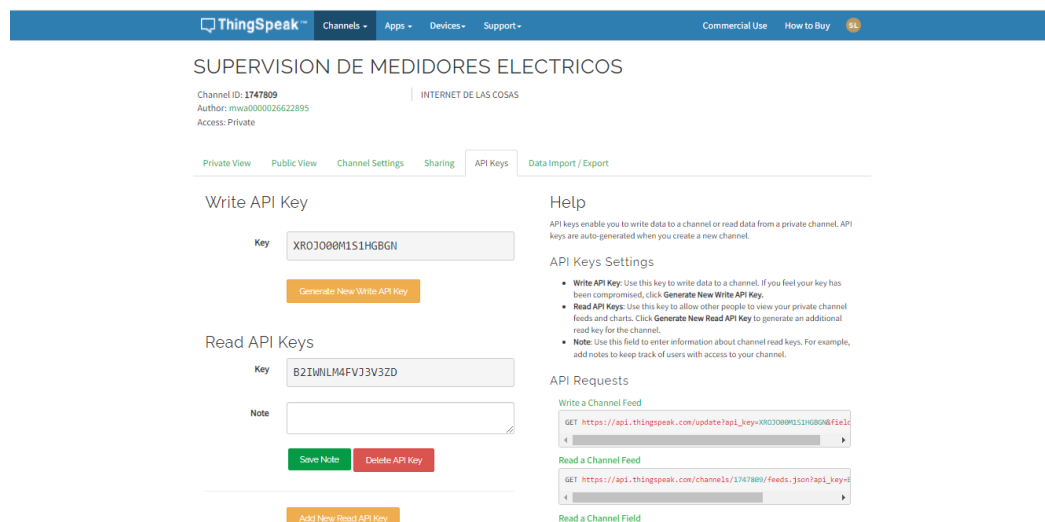
Figura 52:Datos por visualizar en plataforma IoT Thingspeak.



Fuente: Elaboración grupal.

En la figura 53 se aprecia los apis keys que nos ayudan a conectar el prototipo a la plataforma de supervisión el Write API Key o Escribir clave de API se utiliza esta clave para escribir datos en un canal. Esta clave se puede generar como nueva, para ello haga clic en generar nueva clave API de escritura. Por otro lado, se tiene Read API Keys o leer claves API esta clave se usa para permitir que otras personas vean los gráficos y las fuentes de su canal privado. Al igual que el Write API Key se puede generar una clave de lectura adicional para el canal.

Figura 53: Apikey para la conexión con el prototipo y la plataforma de supervisión.



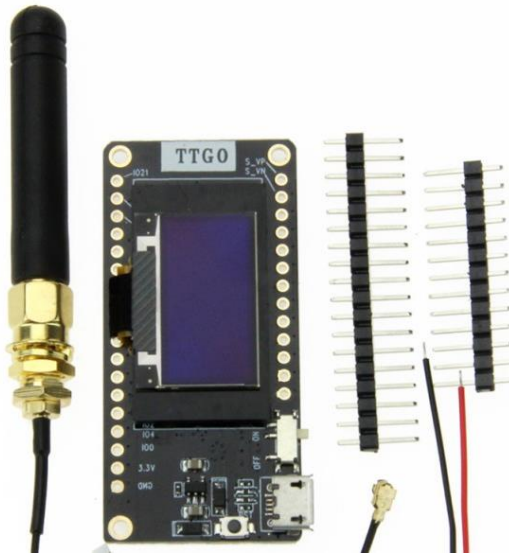
Fuente: Elaboración grupal.

3.6.4. DESARROLLO DEL SISTEMA DE COMUNICACIÓN

Para el desarrollo de sistema de comunicación se resalta dos etapas, la primera sería la comunicación LoRa que se encuentra en configuración punto a punto el cual utiliza el medio de comunicación de radiofrecuencia ISM 910 MHZ. El uso de esta tecnología nos permite ignorar las interferencias por su gran ventaja y versatilidad que lo caracteriza. Cabe resaltar que el tipo de modulación que utiliza es altamente libre de las interferencias. Los módulos que se utilizan son el TTGO LoRa 32 V2, este dispositivo lleva integrado un chip SX1276 que utiliza modulación spread spectrum bajo la

tecnología LoRa también tiene integrada una antena con ganancia de 18 dbi y el microcontrolador ESP32 la cual hace posible que podamos configurarla manipular y configurar a través del programa IDE Arduino.

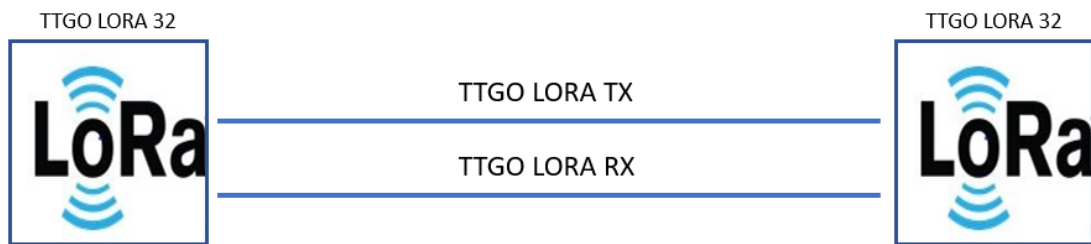
Figura 54:Dispositivos TTGO LoRa32.



Fuente: Elaboración grupal.

En la figura 55 se muestra el esquemático de la comunicación LoRa, también se puede apreciar el tipo de topología que utiliza los dispositivos LoRa, para lograr la comunicación ambos dispositivos se deberán configurar en modo receptor y transmisor todo ello mediante el programa IDE de Arduino.

Figura 55: Esquemático de comunicación LoRa



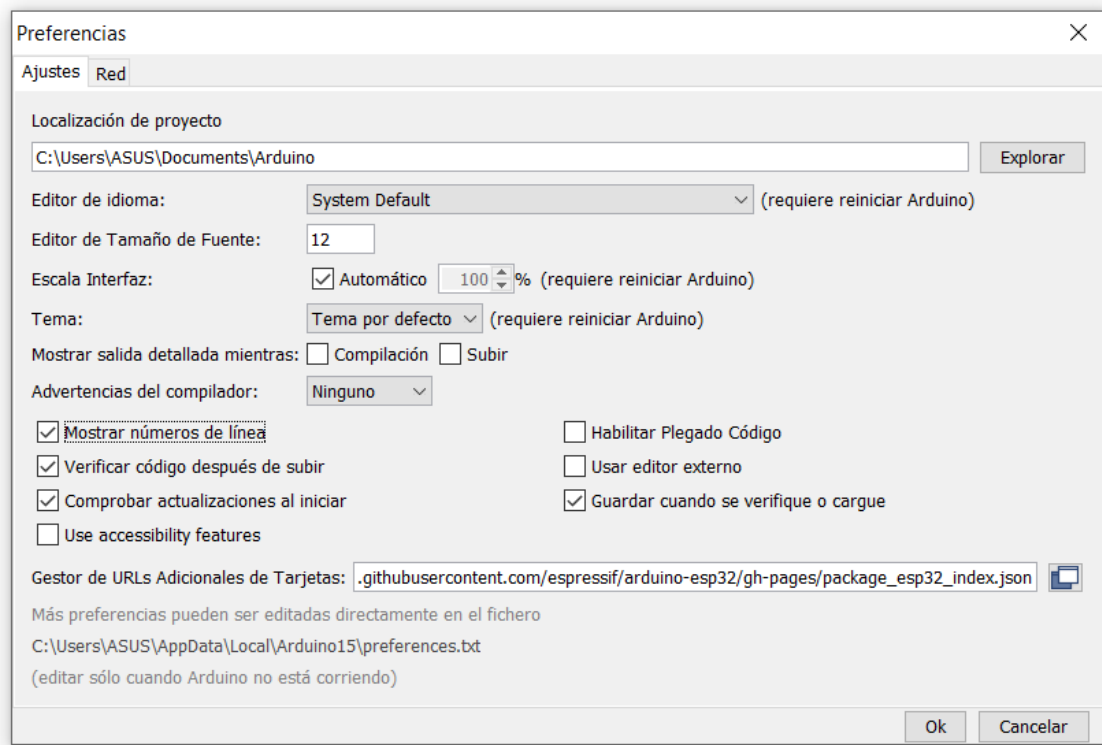
Fuente: Elaboración grupal.

Para poder programar y escribir el código para el dispositivo TTGO LoRa, tenemos que conectar directamente a la computadora o laptop, mediante el puerto USB que contiene la placa, el dispositivo es programable con el IDE Arduino software, para lograr ello se necesita instalar las librerías y complementos del ESP32 y bibliotecas compatibles con LoRa.

Lo primero que debemos incluir en el IDE Arduino es la biblioteca ESP32 ya que el chip sx1276 LoRa está integrado con el ESP32. Antes de ello verificar se esté utilizando la última versión del IDE Arduino.

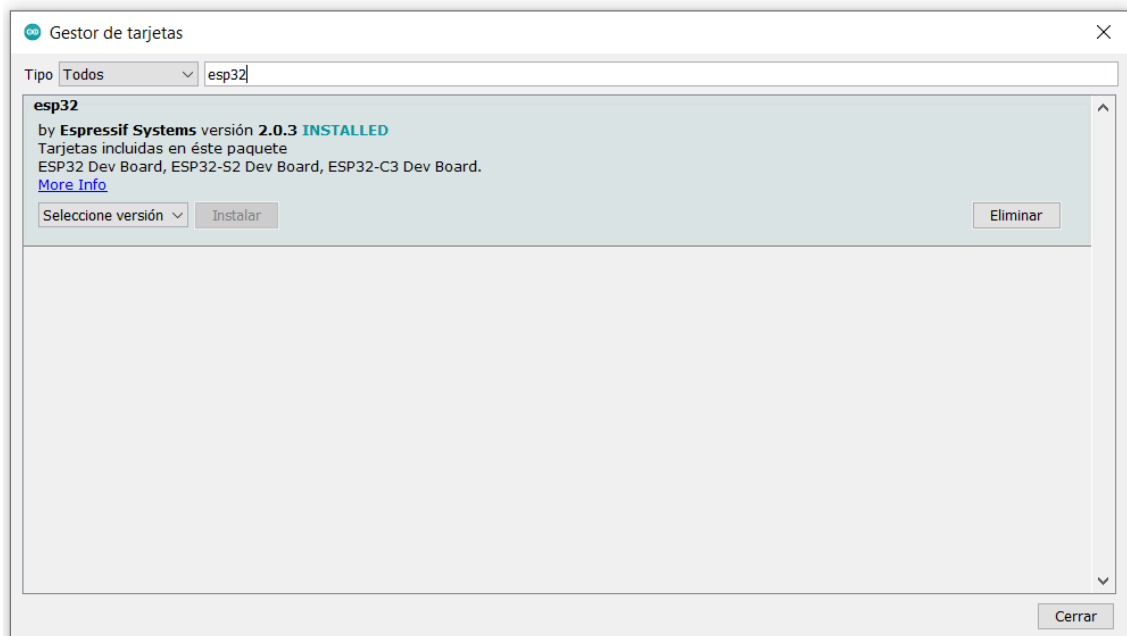
Haga clic en Archivo luego Preferencias y en gestor de URLs adicionales de tarjetas Pegue el enlace [HTTPS://dl.espressif.com/dl/package_esp32_index.json](https://dl.espressif.com/dl/package_esp32_index.json) luego aceptar ver figura 56. El siguiente paso es instalar la placa. Para este propósito, vaya a herramientas, placa, Arduino y administrador de placa y buscar la placa ESP32 en el panel de búsqueda e instarlo ver figura 57.

Figura 56: *Preferencias del IDE Arduino*



Fuente: Elaboración grupal.

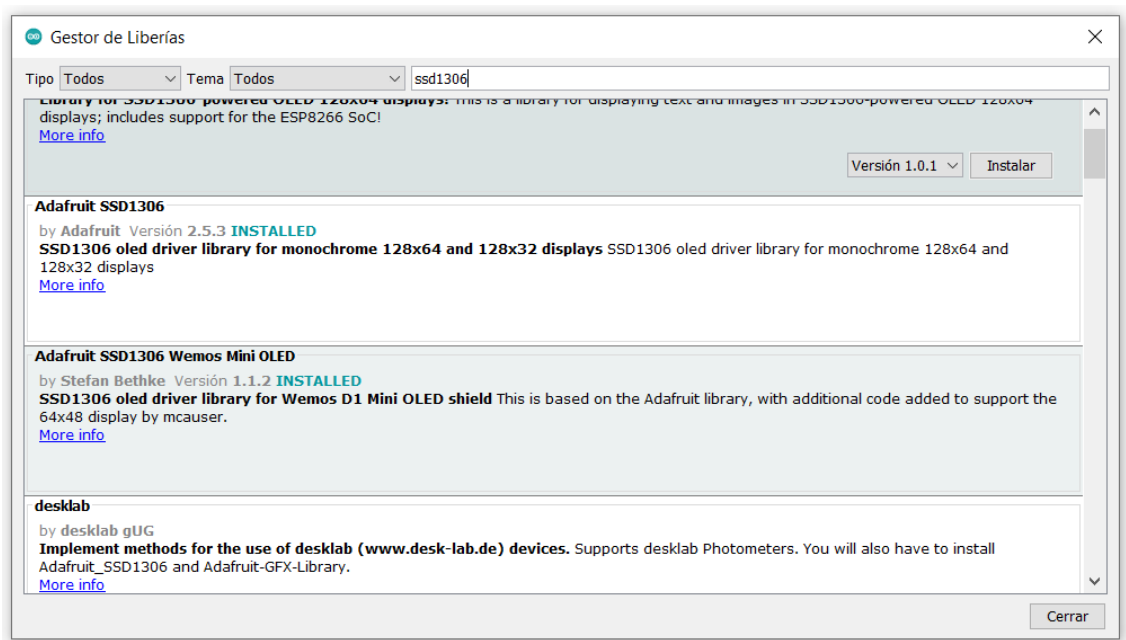
Figura 57: *Instalar la placa ESP32.*



Fuente: Elaboración grupal.

Para lograr visualizar y enviar los datos en la pantalla del dispositivo se incluye las librerías OLED y LORA. Primero para la pantalla se incluye dos librerías Adafruit_SSD1306 y Adafruit_GFX. Estas bibliotecas escritas específicamente son útiles en la interfaz de pantallas OLED. Después de abrir el IDE de Arduino, damos un clic programa, incluir librerías y administrar bibliotecas, buscamos SSD1306 e instalamos ver figura 58.

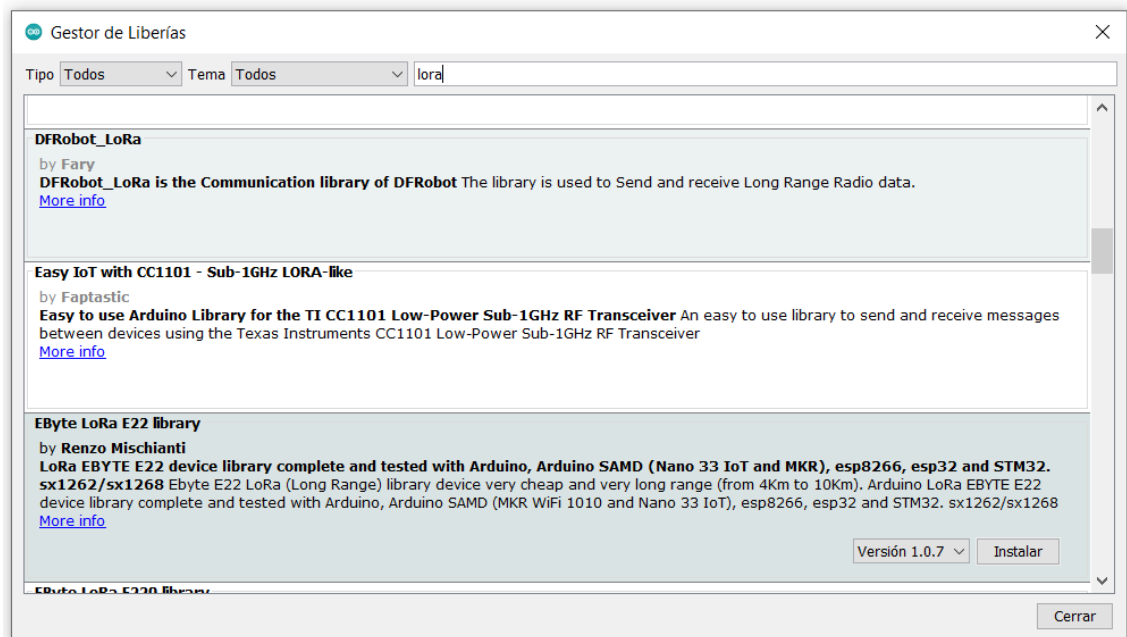
Figura 58: *Instalar las librerías para activar la pantalla OLED.*



Fuente: Elaboración grupal.

Para lograr transmitir datos por los dispositivos TTGO LORA ESP32 se incluye la librería LoRa ver figura 59, después de a ver incluido todas las librerías y tarjetas reiniciamos el software IDE Arduino y procedemos con la segunda etapa que corresponde a la programación.

Figura 59: Instalar las librerías LoRa para transmitir datos



Fuente: Elaboración grupal.

La segunda etapa consiste en desarrollar la comunicación entre el prototipo con la red de internet y así publicar los datos de los sensores. Para ello utilizamos el ethernet shield como medio de comunicación.

En la figura 60 se aprecia la configuración que se le asigna para lograr la comunicación a la red internet, se establece una MAC, la IP, el Gateway y la máscara de Subred, de acuerdo al dispositivo router al que se está conectando

Figura 60: Configuración de red.

```
byte mac[]      = { 0x60, 0x32, 0xB1, 0x51, 0x76, 0x901 };
byte ip[]       = { 192,168,0,150 };
byte gateway[]  = { 192,168,0,1};
byte subnet[]   = { 255, 255, 255, 0 };

const char* server = "184.106.153.149";
```

Fuente: Elaboración grupal.

Para que los datos que se están censando se puedan visualizar y tengan conexión con la plataforma de supervisión se inserta en la programación el apikey que se obtiene al crear el canal del ThingSpeak. En la figura 61 se aprecia el Write API Key que se utiliza para escribir los datos de los sensores en el canal.

Figura 61: *Apikey de plataforma thingspeak.*

```
String apiKey ="J3KNU1CRUQ2TDC4A";  
EthernetClient client;
```

Fuente: Elaboración grupal.

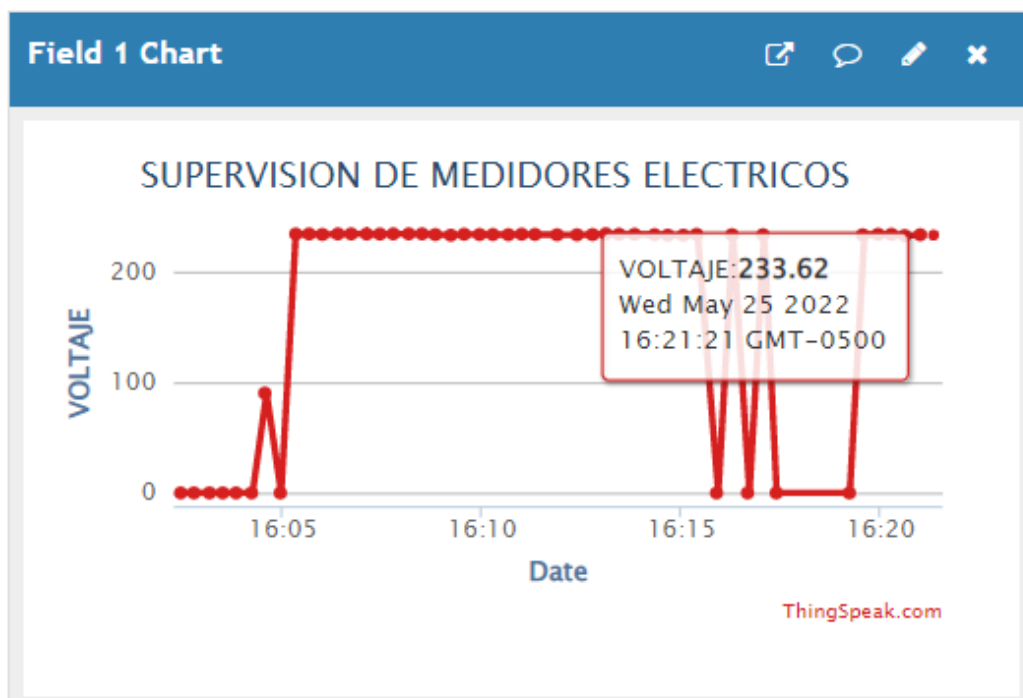
CAPITULO IV

RESULTADOS Y DISCUSIÓN

4.1. RESULTADOS DE LA PLATAFORMA DE SUPERVISIÓN

En la figura 62 se aprecia la supervisión del voltaje de la resistencia donde el prototipo desarrollado monitorea en tiempo real la variable voltaje. También se observa el valor que se está midiendo la fecha y hora específicamente. Estos datos se obtienen directamente del sensor ZMPT101B que mide en paralelo el voltaje y lo envía por la primera etapa de comunicación LoRa, para así después enviar a la plataforma de supervisión mediante prototipo con ethernet shield.

Figura 62:Supervisión del voltaje

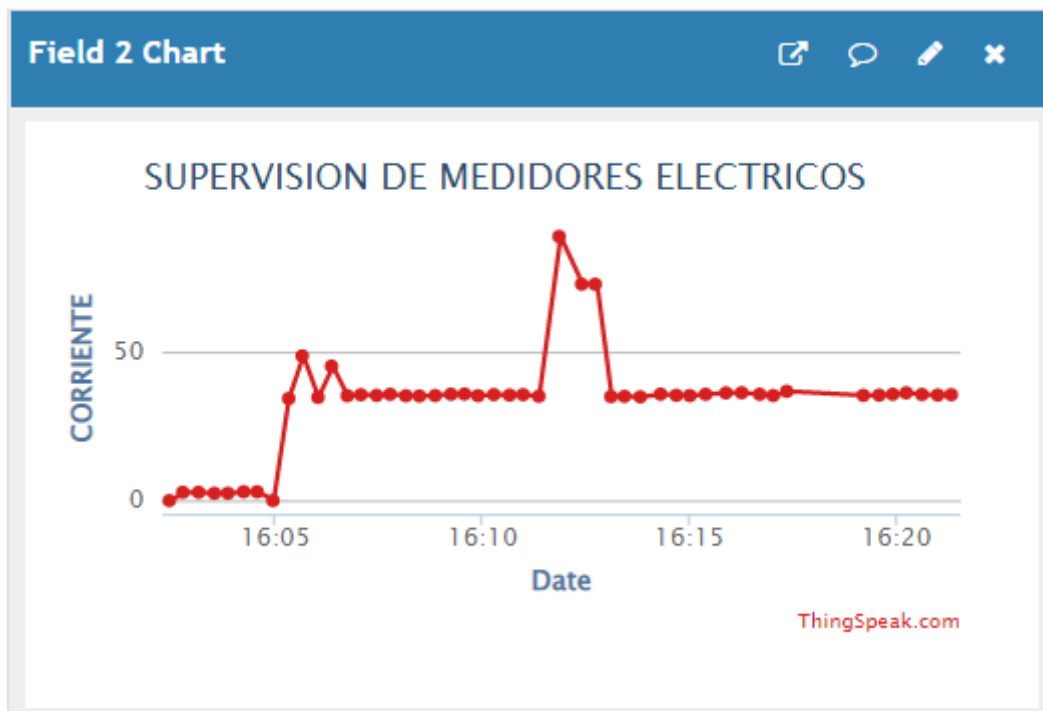


Fuente: Elaboración grupal.

La supervisión de la corriente eléctrica se lleva a cabo por medio del sensor SCT-013-030. Los datos se supervisan en tiempo real a través la plataforma IoT thingspeak.

En la figura 62 aprecia los máximos y mínimos valores recolectados por el sensor antes descrito, la figura 63 también muestra la fecha y hora específicamente en que se obtuvo los datos. El método de medición se lleva a cabo insertando un cable por el sensor, esta a su vez emite un campo electromagnético.

Figura 63: Supervisión de la corriente.



Fuente: Elaboración grupal.

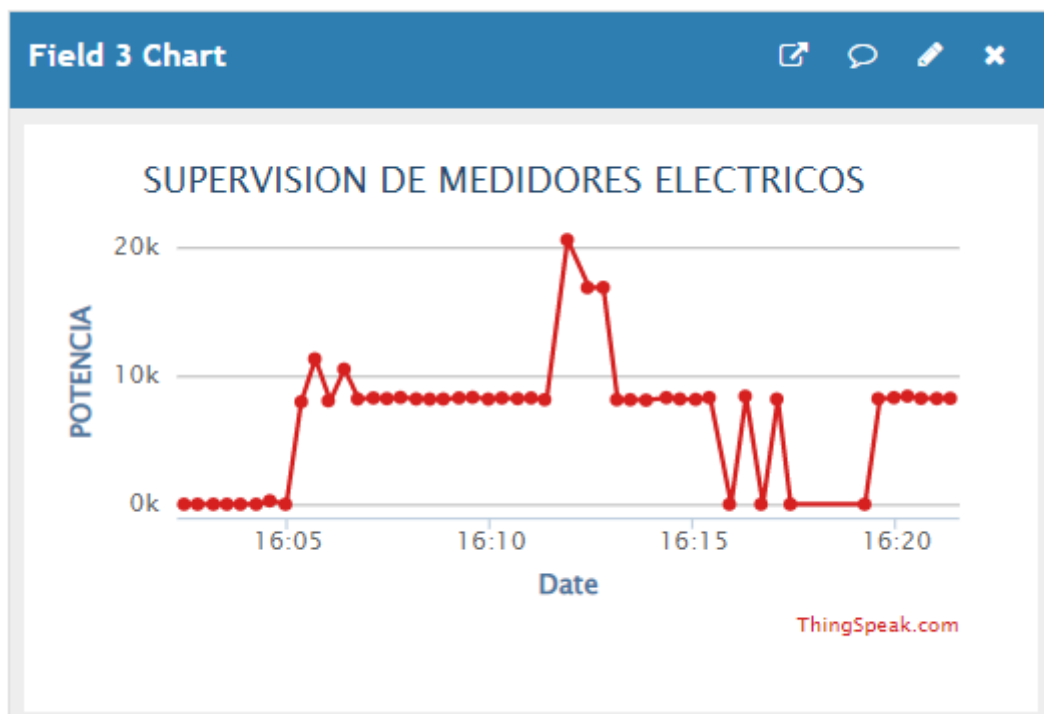
En la figura 64 se muestra la supervisión de la potencia en tiempo real, cabe recalcar que también especifica la hora y fecha en que se leen el valor de dicha variable potencia. Este valor se obtiene mediante los sensores de corriente SCT-013-030 y voltaje ZMPT101b. En la figura 64 podemos apreciar la programación del IDE Arduino se especifica con el tipo de operación producto de ambos sensores.

Figura 64: Configuración para obtener la potencia.

```
if(millis() >= powerLastSample + 1)
{
    sampleCurrent1 = analogRead(CurrentAnalogInputPin)-512+ currentOffset1;
    sampleCurrent2 = (sampleCurrent1/1024)*5000;
    sampleCurrent3 = sampleCurrent2/mVperAmpValue;
    voltageSampleRead = 2*(analogRead(VoltageAnalogInputPin)- 512)+ voltageOffset1 ;
    powerSampleRead = voltageSampleRead * sampleCurrent3 ;
    powerSampleSum = powerSampleSum + powerSampleRead ;
    powerSampleCount = powerSampleCount + 1;
    powerLastSample = millis();
}
```

Fuente: Elaboración grupal.

Figura 65: Supervisión de la corriente.



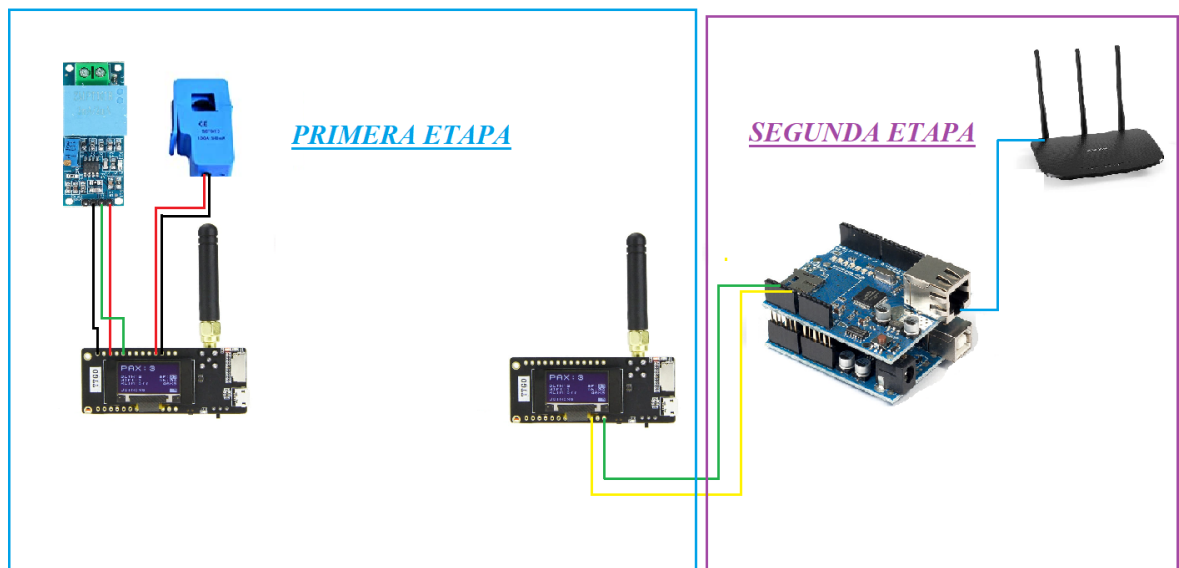
Fuente: elaboración grupal.

4.2. RESULTADOS DEL PROTOTIPO

En la figura 66 se muestra el esquemático de conexión, para la comunicación LoRa se muestra la primera etapa que consiste en conectar los sensores a los pines correspondientes del TTGO LoRa ESP 32, para transmitir los datos se realiza la

programación en IDE Arduino, se utilizó la modulación LoRa “Long Range Modulation” de espectro expandido. Con la modulación se logró transferir datos manteniendo un consumo bajo de energía y una amplia cobertura, se utiliza un factor de ensanchamiento ortogonal SF manteniendo un ancho de banda BW constante y una frecuencia de transmisión de 915 MHz.

Figura 66: Esquemático de conexión



Fuente: Elaboración grupal.

Iniciamos con la descripción del módulo LoRa Sensor, el cual se encarga de censar y enviar los datos al siguiente módulo, este proceso se realiza con el siguiente código mostrado en la imagen 63, en donde se declara las librerías necesarias para LoRa, los pines que corresponden para el módulo LoRa, se define la frecuencia en que se va transmitir la información, en este caso es en la frecuencia de 915MHz, y al final un contador de paquetes para ordenar la transmisión.

Figura 67: Declaración de librerías y pines de uso.

```
//Librerías para LoRa
#include <SPI.h>
#include <LoRa.h>
//Librerías para OLED
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
//Defina los pines utilizados por el módulo LoRa
#define SCK 5
#define MISO 19
#define MOSI 27
#define SS 18
#define RST 14
#define DIO0 26
//Se define la frecuencia LoRa
#define BAND 915E6
//OLED pins
#define OLED_SDA 4
#define OLED_SCL 15
#define OLED_RST 16
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
//Contador de paquetes
int counter = 0;
```

Fuente: Elaboración grupal.

Posteriormente, se define los pines SPI usados por el chip LoRa y se configura el módulo del transceptor del chip LoRa, para luego con una condicional iniciar la comunicación en la banda declarada.

Figura 68: Inicialización de LoRa.

```
//Defina los pines SPI utilizados por el chip LoRa
SPI.begin(SCK, MISO, MOSI, SS);
//Configure el módulo transceptor LoRa.
LoRa.setPins(SS, RST, DIO0);
//Inicializa LoRa
if (!LoRa.begin(BAND)) {
  Serial.println("Starting LoRa failed!");
  while (1);
}
Serial.println("LoRa Initializing OK!");
display.setCursor(0,10);
display.print("LoRa Initializing OK!");
display.display();
delay(2000);
```

Fuente: Elaboración grupal.

Finalmente, en caso del módulo sensor, se envía los datos censados y ya calculados, hacia el módulo Transfer, estos datos son enviados en formato de paquetes, el cual inicia con el comando de *beginPacket()* y termina con el paquete *endPacket()*.

Figura 69: Programación para el envío de datos.

```
//Envío de datos por LoRa
LoRa.beginPacket();
LoRa.print(FinalRMSVoltaje);
LoRa.print("V \n");
LoRa.print(FinalRMSCurrent);
LoRa.print("A \n");
LoRa.endPacket();
```

Fuente: Elaboración grupal.

Para el módulo de transferencia las líneas vienen siendo similares, las diferencias son las que se utilizan para la recepción, en caso de la declaración de variables, librerías

y pines, la diferencia se encuentra en la declaración de una variable string dedicada para los datos recibidos.

Figura 70: Declaración de variable del módulo LoRa transfer.

```
//Librerías para LoRa
#include <SPI.h>
#include <LoRa.h>
//Librerías para OLED
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
//Defina los pines utilizados por el módulo LoRa
#define SCK 5
#define MISO 19
#define MOSI 27
#define SS 18
#define RST 14
#define DI00 26
//Se define la frecuencia LoRa
#define BAND 915E6
//OLED pins
#define OLED_SDA 4
#define OLED_SCL 15
#define OLED_RST 16
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
//Declaración de la variable para los datos recibidos
String LoRaData;
```

Fuente: Elaboración grupal.

La inicialización del chip LoRa, es similar al módulo sensor, en cambio en recepción de paquetes es diferente, inicia comprobando si hay nuevos paquetes para luego mostrarlos, también es posible obtener RSSI, para determinar si la señal está siendo débil, de esta manera determinar lugares más óptimos.

Figura 71:Recepción de paquetes LoRa

```
//comprobamos si hay nuevos paquetes para recibir
int packetSize = LoRa.parsePacket();
if (packetSize) {
  //received a packet
  Serial.print("Received packet ");
  //leer el paquete
  while (LoRa.available()) {
    LoRaData = LoRa.readString();
    Serial.print(LoRaData);
  }

  //También obtenemos el RSSI
  int rssi = LoRa.packetRssi();
  Serial.print(" with RSSI ");
  Serial.println(rssi);
}
```

Fuente: Elaboración grupal.

La comunicación entre los Arduino es mediante la comunicación I2C, y los datos son enviados en un formato *string*. Primero se declara sus librerías y la inicialización de la comunicación.

Figura 72:Inicialización de la comunicación I2C.

```
#include <Wire.h>

void setup() {

  Wire.begin(8);           // join i2c bus with address #8
  Wire.onRequest(requestEvent); // register event
  Serial.begin(9600);

} //end setuo()
```

Fuente: Elaboración grupal.

Luego continuamos con el envío de datos por el esclavo, en este caso por el módulo LoRa, para lo que transforma la declaración de los datos.

Figura 73:Envío de datos por I2C

```
void requestEvent() {  
  
    String voltaje = String(FinalRMSVoltaje);  
    String corriente = String(FinalRMSCurrent);  
    String Data = ""+voltaje+", "+corriente;  
    char dataArray[Data.length()+1];  
    Data.toCharArray(dataArray, data.length()+1);  
    Wire.write(dataArray);  
  
} //end requestEvent()
```

Fuente: Elaboración grupal.

Para en caso del maestro la programación es identifica al esclavo, en cuanto a la declaración de variables, pero se muestra la captura del código.

Figura 74:Inicialización la comunicación I2C (maestro).

```
include <Wire.h>  
  
String received;  
  
void setup() {  
    Wire.begin();           // join i2c bus (address optional for master)  
    Serial.begin(9600);    // start serial for output  
  
}
```

Fuente: Elaboración grupal.

La lectura de datos es lo distinto en este Arduino. En el que se debe separar los datos enviados por el módulo LoRa. En la figura 75 siguiente se muestra la partición de

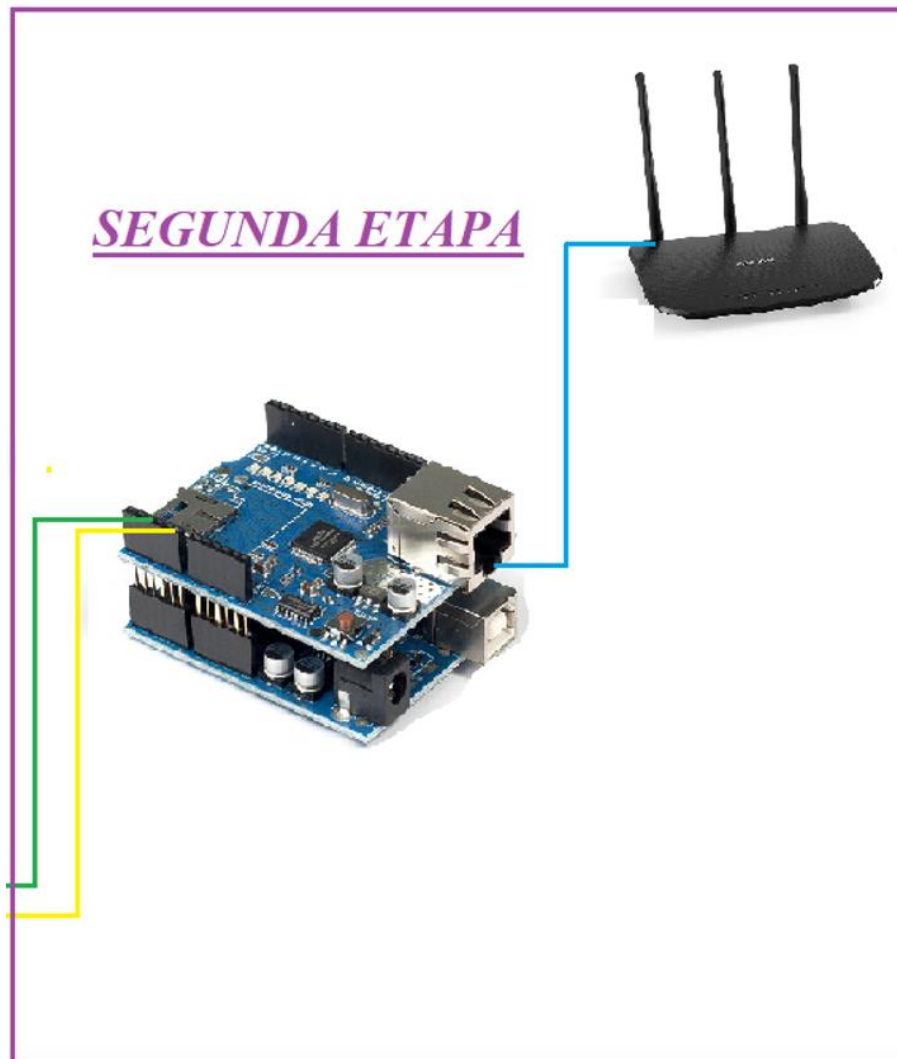
la línea de datos en el cual fue unido por comas, estas nos ayudan como indicadores, al final se tiene los valores de voltaje y corriente ya censados.

Figura 75:Recepción de datos por I2C.

```
void loop() {  
  
    Wire.requestFrom(8, 11);           // request 11 bytes from slave  
    while (Wire.available()) {        // slave may send less than requested  
        char c = Wire.read();         // receive a byte as character  
        received = received + c;     //Add the character to the received string  
    }  
}  
  
//Split the recieved string  
//String comes in the format of "xxx,xxx,xxx" which is a RGB color  
int firstComma = received.indexOf(',');  
int secondComma = received.indexOf(',', firstComma + 1);  
  
String vString = received.substring(0, firstComma);  
String aString = received.substring(firstComma + 1, secondComma);  
  
//Convert the strings to ints  
int volt = vString.toInt();  
int corrien = aString.toInt();  
}
```

Fuente: Elaboración grupal.

Figura 76: Esquemático conexión del servidor



Fuente: Elaboración grupal.

En la figura 72 se muestra la segunda etapa que es parte del servidor para ello se utilizó el microcontrolador Arduino y el ethernet shield, la versatilidad del chip wiznet w5100 con un buffer interno de 16k la cual se encarga de dar una dirección IP al Arduino nos permite la configuración, para así después conectar por el puerto RJ45 con una velocidad de 10/100 Mb. Para poder realizar la programación llamamos a la librería ethernet.h, cabe precisar que se logró la comunicación entre el Arduino y el ethernet shield utilizando los pines 10,11,12,13 también se incluye la librería SPI.h.



V. CONCLUSIONES

Se logró desarrollar un prototipo utilizando internet de las cosas IoT para supervisión de medidores eléctricos con tecnología LoRa en la ciudad de Azángaro, así mismo se logró el monitoreo en tiempo de real de las variables Voltaje corriente y por consecuencia la potencia. Se logro diseñar e implementar el prototipo para la supervisión de medidores eléctricos en la ciudad de Azángaro utilizando microcontrolador, dispositivos LoRa y los sensores de corriente SCT013 y voltaje ZMPT101B. La implementación constó de dos etapas la primera básicamente es la comunicación LoRa punto a punto con una frecuencia de transmisión de 915 MHZ sobre la modulación spread spectrum en esta etapa se logró enviar los datos de los sensores de corriente y voltaje y por consecuencia la potencia, en la segunda etapa se logró implementar el sistema de internet de las cosas que funciona en forma de hardware y software los cuales fueron capaces de supervisar de manera remota los datos de los sensores de corriente y voltaje almacenándose así en un servidor central que funciona sobre una arquitectura de internet de las cosas Thingspeak.



VI. RECOMENDACIONES

Para desarrollar muchos prototipos utilizando IoT para la supervisión de medidores eléctricos con tecnología LoRa en la ciudad de Azángaro se debe destinar una inversión económica por parte de la distribuidora, así se pueda adquirir e instalar un módulo TTGO LoRa ESP32 y sensores que recolecten los datos de corriente y voltaje, para cada medidor.

Para diseño del sistema de supervisión con tecnología LoRa se recomienda que para las pruebas de comunicación LoRa de punto a punto transmitir con las antenas instaladas en cada dispositivo transmisor y receptor respectivamente, así mismo la frecuencia de transmisión deberá ser la misma en ambos dispositivos, también se recomienda utilizar la pantalla OLED de los dispositivos para verificar si la información obtenida por los sensores en esta se visualiza los datos enviados o recibidos según sea el caso.

En el sistema de supervisión remota se puede optar por utilizar almacenamiento de datos avanzados como el MYSQL o el Oracle para el tratamiento de los datos, y así poder tener un registro de la supervisión de los medidores eléctricos en muchas viviendas de la ciudad de Azángaro, también nos facilitaría el poder tener acceso a la información del registro de datos por fechas.

Se recomienda para el diseño de la red LoRa, un análisis de desempeño en la cobertura tanto en lugares libres y con obstáculos por medio de experimentación con módulos transceptores de la serie SX 276/278/279.

Para el diseño e implementación del prototipo de supervisión de medidores eléctricos se recomienda instalar todas las librerías en el IDE Arduino para que tenga compatibilidad con la tecnología LoRa.



VII. REFERENCIAS BIBLIOGRÁFICAS

- Aguilar Zavaleta, S. (2019). Diseño de un sistema basado en IOT para la supervisión y control de estaciones remotas de la dirección de hidrografía y navegación de la Marina de Guerra del Perú. Lima, Perú: Pontificia Universidad Católica del Perú..
- Burbano Ordoñez, C. Y. (2017). Implementación De Una Red De Sensores Inalámbricos Lpwan Mediante Módulos Lora Para El Monitoreo De La Calidad Del Agua En 2 Ríos. Bogotá, Colombia: Universidad Distrital Francisco José De Caldas.
- Burbano, C. (2019). Implementación de una red de sensores inalámbricos LPWAN mediante módulos LoRa para el monitoreo de la calidad del agua en 2 ríos.
- Chou, T. (2016). Precision: Principles, Practices and Solutions for the Internet of Things. Estados Unidos: CrowdBook.
- Córdoba Peñalver, E. J. (2017). Análisis y diseño de una red de sensores en un parque natural. Catalunya, España: Universitat Oberta de Catalunya.
- Dávila, J., Aguilar, R., Ruiz, J., Parra-Michel, J. R., & Martínez Peláez, R. (2018). Gaszen: plataforma para monitorizar el nivel de gas L.P. RIIIT. Revista internacional de investigación e innovación tecnológica.
- Campo, G., Gomez, I., Cañada, G., & Santamaria, A. (2019). Hybrid LPWAN Communication Architecture for Real-Time Monitoring in Power Distribution Grids. IEEE Xplore.
- Díaz, J., Fava, L., Vilches Antao, D., & Candia, A. (2019). Soluciones IoT con tecnología LoRaWAN. XXI Workshop de Investigadores en Ciencias de la Computación (WICC 2019, Universidad Nacional de San Juan), 1-5.
- Ertürk, M. A., Aydın, M. A., Büyükakkaşlar, M. T., & Evirgen, H. (2019). A Survey on LoRaWAN Architecture, Protocol and Technologies. Future Internet, 11(10), 216. <https://doi.org/10.3390/fi11100216>



- Ertürk, M., Aydın, M., Büyükakkaslar, M., & Hayrettin, E. (2019). A Survey on LoRaWAN Architecture, Protocol and Technologies. *Future Internet*, 11, 1-34. <https://doi.org/10.3390>
- Flor Salazar, P. (2018). Diseño de un sistema de alumbrado público inteligente basado en internet de las cosas para el centro histórico de la ciudad de Popayán. <http://repository.unad.edu.co/handle/10596/21237>
- Ferrer Sanabria, R. D. (2017). Desarrollo de una estación meteorológica de bajo costo. Bucaramanga, Colombia: Universidad Santo Tomás Bucaramanga.
- Freire Tejada, M. F. (2019). Diseño y desarrollo de un prototipo para transmitir parámetros de perforación, en tiempo real utilizando LPWAN en un pozo petrolero. Quito, Ecuador: Universidad de las Américas.
- Haxhibeqiri, J., De Poorter, E., Moerman, I., & Hoebeke, J. (2018a). A Survey of LoRaWAN for IoT: From Technology to Application. *University—, Department of Information Technology at Ghent*, 18, 1-38. <https://doi.org/10.3390>
- Haxhibeqiri, J., De Poorter, E., Moerman, I., & Hoebeke, J. (2018b). A Survey of LoRaWAN for IoT: From Technology to Application. *Sensors (Basel, Switzerland)*, 18(11), 3995. <https://doi.org/10.3390/s18113995>
- Hernández Sampieri, R., Fernández Collado, C., & Baptista Lucio, M. (2014). *Metodología de la Investigación*. Mexico D.F.: mcgraw-hill / interamericana editores, s.a. de c.v.
- Javed, A. (2016). *Building Arduino Projects for the Internet of Things*. <https://doi.org/10.1007>
- Jinchuña Illa, Juan Carlos. (s. f.). Análisis técnico-económico de sistemas de telegestión para la iluminación eficiente con luminarias de tecnología LED en los principales parques céntricos de la ciudad de Puno—2019. Recuperado 13 de



diciembre de 2022, de

<https://repositorioslatinoamericanos.uchile.cl/handle/2250/3280643#.Y5jLyZfq>

Xq4.mendeley

Kosari, A., & Wentzloff, D. (2019). MURS Band for LPWAN Applications. IEEE Xplore, 6–9.

Kurniawan, A. (2019). Building Wireless Sensor Networks with ESP32 LoRa. Berlin: Agus Kurniawan.

Girón, N. A. G., & LópezHurtado, M. A. (2020). Selección de tecnologías LPWAN para la implementación de un sistema IoT aplicado a la lombricultura. Memorias. <https://doi.org/10.22490/25904779.4170>

Guaca, N., & Lopez, M. (2020). Selección de tecnologías LPWAN para la implementación de un sistema IOT aplicado a la lombricultura. Universidad Nacional, Abierta y a Distancia, Colombia.

Mikhaylov, K., Stusek, M., Masek, P., Petrov, V., Petajarvi, J., Andreev, S., . . . Koucheryavy, Y. (2018). Multi-RAT LPWAN in Smart Cities: Trial of LoRaWAN and NB-IoT Integration. IEEE Xplore.

Petäjajarvi, J., Mikhaylov, K., Yasmin, R., Hämäläinen, M., & Iinatti, J. (2017). Evaluation of LoRa LPWAN Technology for Indoor Remote Health and Wellbeing Monitoring. International Journal of Wireless Information Networks, 24(2), 153-165. <https://doi.org/10.1007/s10776-017-0341-8>

Pfister, C. (2011). Getting Started whit the internet of things. Published by O'Reilly Media, Inc.

Rauh, S., Robert, J., Schadhauer, M., & Heuberger, A. (2018). LPWAN occupancy model parameter IDentification for license exempt sub-GHz frequency bands. IEEE Xplore.



- Semtech. (5 de agosto de 2016). Semtech Corporation. Obtenido de <https://www.mouser.com/datasheet/2/761/sx1276-1278113.pdf>.
- Semtech. (2016). Semtech Corporation.
<https://www.mouser.com/datasheet/2/761/sx1276-1278113.pdf>
<https://www.mouser.com/datasheet/2/761/sx1276-1278113.pdf>
- Systems, E. (2022). ESP32 Series. <https://www.espressif.com/>.
https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf
- Singh, A. (2019). 5G Simply In Depth. India: Ajitvoice.
- Szalai, A., Szabo, T., Horvath, P., Timar, A., & Poppe, A. (2016). Smart SSL: Aplicación de plataformas de diseño IOT / CPS en luminarias de alumbrado público basadas en LED. IEEE Lighting Conference of the Visegrad Countries,1-6.
- Valarezo Troya, W. F., & Criollo Jaramillo, A. F. (2017). Diseño y estudio del uso de una red LPWAN (Low Power Wide Área Network) para la optimización de la medición, comunicación y corrección de errores, en el consumo de agua potable. Guayaquil, Ecuador: Escuela Superior Politécnica del Litoral.
- unit electronics. (s. f.). ZMPT101B Sensor De Voltaje AC 2mA. UNIT Electronics.
Recuperado 13 de diciembre de 2022, de <https://uelectronics.com/producto/zmpt101b-sensor-de-voltaje-ac-2ma/>
- logicbus. (s. f.). Sensores, Transductores y Acondicionadores. Recuperado 13 de diciembre de 2022, de <https://www.logicbus.com.mx/transductores-sensores.php>



ANEXOS

ANEXO A. Código de programación

```
#include <SPI.h>
#include <Ethernet.h>

int decimalPrecision = 2;
int voltageOffset1 = 0;
int voltageOffset2 = 0;
int voltageAnalogInputPin = A0;
float voltageSampleRead = 0;
float voltageLastSample = 0;
float voltageSampleSum = 0;
float voltageSampleCount = 0;
float voltageMean ;
float RMSVoltageMean ;

/* 2- AC Corriente */

int currentOffset1 = 0;
int currentOffset2 = 0;
int CurrentAnalogInputPin = A1;
float mVperAmpValue = 31.25;

float currentSampleRead = 0;
float currentLastSample = 0;
float currentSampleSum = 0;
float currentSampleCount = 0;
float currentMean ;
float RMSCurrentMean =0 ;
float FinalRMSCurrent ;

/* 3- AC Potencia*/

int powerOffset =0;
float sampleCurrent1 ;
float sampleCurrent2 ;
```



```
float sampleCurrent3 ;
float apparentPower;
float realPower = 0;
float powerSampleRead = 0;
float powerLastSample = 0;
float powerSampleCount = 0;
float powerSampleSum = 0;
float powerFactor = 0;

byte mac[] = { 0x60, 0x32, 0xB1, 0x51, 0x76, 0x901 };
byte ip[] = { 192,168,0,150 };
byte gateway[] = { 192,168,0,1};
byte subnet[] = { 255, 255, 255, 0 };

const char* server = "184.106.153.149";

String apiKey ="J3KNU1CRUQ2TDC4A";

EthernetClient client;
void setup()

{
    /* 0- General */
    Serial.begin(9600);
    delay(3000);
}
void loop()
{
    /* 1- AC voltage Measurement */

    if(micros() >= voltageLastSample + 1000 )
    {
        voltageSampleRead = (analogRead(voltageAnalogInputPin)-
512) + voltageOffset1;
        voltageSampleSum = voltageSampleSum +
sq(voltageSampleRead) ;
        voltageSampleCount = voltageSampleCount + 1;
        voltageLastSample = micros() ;
    }
}
```



```
if(voltageSampleCount == 1000)
{
    voltageMean = voltageSampleSum/voltageSampleCount;
    RMSVoltageMean = sqrt(voltageMean)+ voltageOffset2;
    if(RMSVoltageMean <=2)
    {RMSVoltageMean = 0;}
    Serial.print("Volt: ");
    Serial.print(RMSVoltageMean,decimalPrecision);
    Serial.println("V ");
    voltageSampleSum =0;
    voltageSampleCount=0;
}

/* 2- AC Current Measurement */

if(millis() >= currentLastSample + 1)
{
    currentSampleRead = analogRead(CurrentAnalogInputPin)-512
+ currentOffset1;
    currentSampleSum = currentSampleSum +
sq(currentSampleRead) ;
    currentSampleCount = currentSampleCount + 1;
    currentLastSample = millis();
}

if(currentSampleCount == 1000)
{
    currentMean = currentSampleSum/currentSampleCount;
    RMSCurrentMean = sqrt(currentMean)+currentOffset2 ;
    FinalRMSCurrent = (((RMSCurrentMean /1024) *5000)
/mVperAmpValue);
    Serial.print("Current RMS: ");
    Serial.print(FinalRMSCurrent,decimalPrecision);
    Serial.println(" A ");
    currentSampleSum =0;
    currentSampleCount=0;
}

/* 3- AC Power with Direction */
```



```
        if(millis() >= powerLastSample + 1)
        {
            sampleCurrent1 = analogRead(CurrentAnalogInputPin)-512+
currentOffset1;
            sampleCurrent2 = (sampleCurrent1/1024)*5000;
            sampleCurrent3 = sampleCurrent2/mVperAmpValue;
            voltageSampleRead = 2*(analogRead(voltageAnalogInputPin)-
512)+ voltageOffset1 ;
            powerSampleRead = voltageSampleRead * sampleCurrent3 ;
            powerSampleSum = powerSampleSum + powerSampleRead ;
            powerSampleCount = powerSampleCount + 1;
            powerLastSample = millis();
        }

        if(powerSampleCount == 1000)
        {
            realPower = ((powerSampleSum/powerSampleCount)+
powerOffset) ;
            Serial.print("Real Power (W): ");
            Serial.print(realPower);
            Serial.println(" W ");
            apparentPower= FinalRMSCurrent*RMSVoltageMean;
            Serial.print("Apparent Power (VA): ");
            Serial.print(apparentPower,decimalPrecision);
            Serial.println(" VA ");
            powerFactor = realPower/apparentPower;
            if(powerFactor >1 || powerFactor<0)
            {
                powerFactor = 0;
            }
            Serial.print("Power Factor: ");
            Serial.println(powerFactor,decimalPrecision);
            Serial.println(" ");
            powerSampleSum =0;
            powerSampleCount=0;

            Serial.println("enviado dato");
            Ethernet.begin(mac, ip);

            //delay(1000);
```



```
Serial.println("conectando...");

if (client.connect(server, 80))
{
Serial.println(" Client connected ");

String postStr = apiKey;
postStr += "&field1=";
postStr += String(RMSVoltageMean,decimalPrecision);
postStr += "&field2=";
postStr += String(FinalRMSCurrent,decimalPrecision);
postStr += "&field3=";
postStr += String(apparentPower);
postStr += "\r\n\r\n";

client.print("POST /update HTTP/1.1\n");
client.print("Host: api.thingspeak.com\n");
client.print("Connection: close\n");
client.print("X-THINGSPEAKAPIKEY: " + apiKey + "\n");
client.print("Content-Type: application/x-www-form-urlencoded\n");
client.print("Content-Length: ");
client.print(postStr.length());
client.print("\n\n");
client.print(postStr);
delay(20000);

}
client.stop();
Serial.println("uso funcion");
    }
}
```


ANEXO B. Supervisión del voltaje



ANEXO C. Prueba del circuito en el medidor electrico

