



UNIVERSIDAD TECNOLÓGICA ISRAEL
ESCUELA DE POSGRADOS "ESPOG"

MAESTRÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN
Resolución: RPC-SO-09-No.265-2021

PROYECTO DE TITULACIÓN EN OPCIÓN AL GRADO DE MAGISTER

Título del proyecto:
Aplicación IoT para el monitoreo de consumo eléctrico residencial utilizando software libre.
Línea de Investigación:
Ciencias de la Ingeniería aplicadas a la producción, sociedad y desarrollo sustentable.
Campo amplio de conocimiento:
Ingeniería, Industria y construcción.
Autor/a:
Ángel Rogelio Minta Toapanta
Tutor/a:
MSc. René Ernesto Cortijo Leyva

Quito – Ecuador

2022

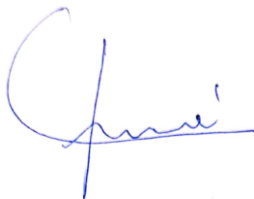
APROBACIÓN DEL TUTOR



Yo, Mg. **René Ernesto Cortijo Leyva** con C.I: **1719010108**, en mi calidad de Tutor del proyecto de investigación titulado: **“Aplicación IoT para el monitoreo de consumo eléctrico residencial utilizando software libre”**.

Elaborado por: **Ángel Rogelio Minta Toapanta**, de C.I: **0604620088**, estudiante de la Maestría: **Electrónica y Automatización**, de la **UNIVERSIDAD TECNOLÓGICA ISRAEL (UISRAEL)**, como parte de los requisitos sustanciales con fines de obtener el Título de Magister, me permito declarar que luego de haber orientado, analizado y revisado el trabajo de titulación, lo apruebo en todas sus partes.

Quito 8 de septiembre del 2022



Firma

DECLARACIÓN DE AUTORIZACIÓN POR PARTE DEL ESTUDIANTE



Yo, Ángel Rogelio Minta Toapanta con C.I: 0604620088, autor/a del proyecto de titulación denominado: **Aplicación IoT para el monitoreo de consumo eléctrico residencial utilizando software libre**. Previo a la obtención del título de Magister en Electrónica y Automatización.

1. Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar el respectivo trabajo de titulación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.
2. Manifiesto mi voluntad de ceder a la Universidad Tecnológica Israel los derechos patrimoniales consagrados en la Ley de Propiedad Intelectual del Ecuador, artículos 4, 5 y 6, en calidad de autor@ del trabajo de titulación, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente. En concordancia suscribo este documento en el momento que hago entrega del trabajo final en formato impreso y digital como parte del acervo bibliográfico de la Universidad Tecnológica Israel.
3. Autorizo a la SENESCYT a tener una copia del referido trabajo de titulación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de prosperidad intelectual vigentes.

Quito D.M., 8 de septiembre de 2022.

Firma

Tabla de contenidos

APROBACIÓN DEL TUTOR	2
DECLARACIÓN DE AUTORIZACIÓN POR PARTE DEL ESTUDIANTE	3
1 INFORMACIÓN GENERAL	1
Contextualización del tema	1
Problema de investigación	2
Objetivo general	3
Objetivos específicos	3
Vinculación con la sociedad y beneficiarios directos:	3
2 CAPÍTULO I: DESCRIPCIÓN DEL PROYECTO	5
1.1. Contextualización general del estado del arte	5
1.2. Proceso investigativo metodológico	9
3 CAPÍTULO II: PROPUESTA	11
2.1 Fundamentos teóricos aplicados	11
Introducción	11
Arquitectura IoT de cuatro niveles	11
Nivel de percepción	12
PZEM-004T	12
<i>Raspberry Pi 4 Model B</i>	12
<i>NodeMCU ESP8266</i>	13
Nivel de conectividad	14
Protocolo MQTT	14
Nivel de servicio	15
Mosquitto	15
<i>InfluxDB</i>	15
Nivel de aplicación	15
<i>Node-RED</i>	15
<i>Grafana</i>	15
2.2 Descripción de la propuesta	16
2.3 Validación de la propuesta	29
2.4 Matriz de articulación de la propuesta	31
2.5 Análisis de resultados. Presentación y discusión.	33
4 CONCLUSIONES	39
5 RECOMENDACIONES	40

6	BIBLIOGRAFÍA	41
7	ANEXOS	44

Índice de tablas

Tabla 1	Comandos de instalación de software	19
Tabla 2	Campos de información para la creación de la base de datos Smart_Meter	24
Tabla 3	Campos de configuración para agregar una base de datos desde InfluxDB	25
Tabla 4	Información del perfil de especialistas	30
Tabla 5	Evaluación de criterios de los tres especialistas seleccionados	30
Tabla 6	Matriz de articulación	31
Tabla 7	Error porcentual de los valores de voltaje de Smart_Energy vs Fluke 323	35
Tabla 8	Valores de energía obtenidos de Smart_Meter	36
Tabla 9	Lectura inicial y final obtenidas del medidor analógico	37
Tabla 10	Error porcentual de valores de energía de Smart_Energy vs medidor analógico. .	37

Índice de figuras

Figura 1	Estructura IoT de alto nivel	11
Figura 2	Diagrama de conexión de PZEM-004T-100 A.....	12
Figura 3	Raspberry Pi 4 Modelo B.....	13
Figura 4	NodeMCU ESP8266	14
Figura 5	Diagrama de conexión MQTT.....	14
Figura 6	Arquitectura de cuatro niveles planteada	16
Figura 7	Diagrama de bloques para el registro de variables eléctricas	17
Figura 8	Diagrama de flujo de aplicación IoT Planteada.....	17
Figura 9	Diagrama de conexión de NodeMCU y PZEM 004T	18
Figura 10	Declaración de variables en IDE Arduino	19
Figura 11	Eclipse Mosquitto instalado correctamente.....	20
Figura 12	Node-RED instalado correctamente	20
Figura 13	Base de Datos InfluxDB instalado correctamente.	20
Figura 14	Servidor Grafana instalado correctamente	21
Figura 15	Programación de MQTT	21
Figura 16	Publicación de topics MQTT.....	22
Figura 17	Programación para la comunicación MQTT.....	22
Figura 18	Configuración del nodo mqtt in parte uno	23
Figura 19	Configuración del nodo mqtt in parte dos.....	23
Figura 20	Configuración del nodo influxdb out parte uno.....	24
Figura 21	Configuración del nodo influxdb out parte dos	25
Figura 22	Interfaz InfluxDB Smart_Meter	26
Figura 23	Código Flux para visualizar el panel de corriente	27
Figura 24	Dashboard general de la aplicación IoT planteada.....	27
Figura 25	Conexión física Raspberry Pi	28
Figura 26	Conexión física NodeMCU y PZEM 004T.....	28
Figura 27	Scrum de aplicación IoT	29
Figura 28	Comprobación de comunicación del dispositivo de borde con Mosquitto.	33
Figura 29	Dashboard de voltaje, corriente, frecuencia, potencia y factor de potencia	34
Figura 30	Dashboard de energía, consumo y costo.....	34
Figura 31	Voltaje obtenido en Smart_Energy y en FLUKE 323	35
Figura 32	Gráfico de energía en el intervalo de tiempo de 03:00-17:00.....	37
Figura 33	Resultados de prueba	38
Figura 34	Resultados de validación de especialistas	38

INFORMACIÓN GENERAL

Contextualización del tema

Es ineludible mencionar que la Industria 4.0 comprende la integración de tecnologías de innovación e información, direccionadas a la comunicación dentro de la industria. El desarrollo de una red inteligente de productos y servicios dentro de la cadena de valor es su enfoque principal (Viáfara et al., 2021, p. 98), esto impulsado por desarrollo de nuevos modelos organización y control. El Internet de las cosas o por sus siglas IoT es una tecnología de las tantas de la Industria 4.0, Lozada (2022), destaca que este se orienta a servicios para consumidores (p. 16). Día a día se visto el gran avance en temáticas del IoT, *cloud computing*, el desarrollo de componentes o elementos versátiles tales como los sistemas embebidos, *gateway IoT*, entre otros, han permitido crear soluciones que pueden mejorar notablemente la ejecución de distintas actividades (Uslenghi, 2019, p. 9). Esto para los consumidores que se desenvuelven en diferentes entornos tal es el caso del hogar.

Para el desenvolvimiento de las diferentes actividades económicas de un país, se requiere de formas de energía, como la eléctrica. Una gestión eficiente en redes de transmisión y distribución eléctrica es necesaria, misma que converja recursos humanos, planes óptimos de desarrollo, recursos económicos y tecnologías de punta para las etapas de administración y control (ARCERNNR, 2022). El sector eléctrico comprende un campo de gran aplicación del IoT, esto direccionado a una adecuada gestión energética, debido a diversos problemas que se han producido en este campo, a esto se añade que en Ecuador la categoría residencial es la que presenta un alto consumo eléctrico.

Frecuentemente durante la distribución de la energía eléctrica pueden presentarse pérdidas relacionadas con el voltaje de línea, cargas, gestión de las empresas de energía etc., en este contexto empresas prestadoras de energía eléctrica han tenido que enfrentarse a inconvenientes como: errores en medición, mantenimiento deficiente de los registros de clientes, medidores en estado de fallo o mal funcionamiento entre otras, donde usuarios residenciales, comerciales e industriales se han visto afectados. Trabajos de investigación promueven el desarrollo de prototipos basados en IoT, software libres, servicios en la nube etc., que permiten tener un control y monitoreo de la red eléctrica, como es el caso de la toma de datos del consumo residencial de manera instantánea, que permitan focalizar acciones pertinentes como medidas de ahorro energético para los usuarios, y para empresas prestadoras les permita optimizar procesos de toma de lecturas y disponer datos confiables.

En una residencia de una familia promedio ciertos meses del año en curso canceló cantidades elevadas por consumo energético, motivo que causó preocupación para la familia. En la residencia habitan cuatro integrantes, donde su caja de distribución consta de un circuito de iluminación, uno de fuerza y el de cargas especiales (ducha, lavadora). El saber de manera detallada como es el consumo de energía que se efectúa en esta, es importante para la toma de decisiones necesarias en cuanto al ahorro energético. Para el presente trabajo esta residencia será el objeto de estudio, es decir mediante la implementación de la aplicación IoT propuesta se pretenderá monitorear el consumo diario que la familia tiene.

Problema de investigación

Ecuador se encuentra en una posición que le permite indagar en tecnologías relacionadas con la Industria 4.0 para alcanzar una digitalización que le permita crecer y evolucionar, creando sistemas innovadores de manufactura, gestión y negocio (Paredes, 2020, p. 2). Sin embargo, ante lo expuesto en la industria ecuatoriana, empresas de servicio etc., se sigue manteniendo métodos de negocio y manufactura tradicionales, control o comunicación inadecuada de procesos, provocando un bajo nivel de competitividad de las organizaciones y de manera específica imposibilitando un monitoreo en tiempo real. En su TFM Paredes, (2020) diseñó y construyó un sistema de control y monitoreo con sistemas embebidos para entornos industriales, que puede ser adaptable a cualquier sistema, como el sector eléctrico que requiere de medidas de índole eficiente en cuanto al uso del recurso eléctrico. Por otro lado, Echeverri & Patiño, (2018) menciona que la implementación de infraestructuras de red ha sido un poco lenta debido a los altos costos. Por ello es necesario desarrollar aplicaciones particulares eficientes enfocadas en resolver problemáticas en el sector eléctrico, específicamente consumo residencial.

Empresas prestadoras de servicio eléctrico requieren de la asignación de un determinado porcentaje del presupuesto, demandando grandes cantidades de recursos a actividades como el registro del consumo de energía de cada mes, por lo que usuarios residenciales, comerciales e industriales deben pagar mensualmente el servicio eléctrico prestado, de esto se obtiene que la forma de recaudo mediante factura no brinda una información detallada del uso de la cantidad de energía cobrada. Por ello ante la creciente necesidad de técnicas de control y monitoreo sofisticados a nivel de distribución eléctrica, enfatizando el consumo residencial, se plantea la incorporación de un sistema inteligente empleando tecnologías de la industria 4.0 como lo es el IoT. Además, se pretende concientizar a usuarios residenciales sobre el ahorro

energético ante altos niveles de consumo eléctrico, pues mediante el monitoreo de parámetros eléctricos el cliente podrá visualizar su consumo y tomar las medidas pertinentes.

Al ser el sector residencial el que más consumo presenta y tomando en cuenta que el hogar es uno de los entornos donde más se desenvuelve el ser humano, conviene realizar un estudio del consumo energético residencial, específicamente esto se llevará a cabo en una residencia de una familia promedio que por años han cancelado sus tarifas mensuales. Algunos meses el valor a pagar de su planilla se elevó inexplicablemente, por esto el usuario requiere saber de manera detallada sobre su consumo energético, para la correspondiente toma de decisiones.

Objetivo general

Desarrollar una aplicación IoT para el monitoreo de consumo eléctrico residencial utilizando software libre.

Objetivos específicos

Definir los software con los que se puede desarrollar el proyecto de monitoreo de parámetros eléctricos.

Diseñar la arquitectura de alto nivel IoT para monitorear parámetros eléctricos utilizando principios de integración e interoperabilidad.

Implementar la aplicación IoT para visualizar los parámetros eléctricos del consumo de energía en tiempo real en un dashboard, mediante la arquitectura definida anteriormente.

Validar si la aplicación IoT de monitoreo de consumo eléctrico residencial cumple con las funciones para las cuales fue desarrollado mediante pruebas experimentales.

Vinculación con la sociedad y beneficiarios directos:

Ecuador cuenta con un Plan Nacional de Eficiencia Energética, en uno de sus artículos se menciona lo siguiente:

Artículo 17.-Ahorro y uso eficiente de energía.- A nivel nacional, todo consumidor de energía debe velar permanente porque sus consumos entes enmarcados en el uso racional de la energía, y adaptar sus comportamientos de consumo, orientándose al ahorro energético sin que signifique disminuir sus condiciones de confort y producción (Gobierno del Ecuador, 2019, p. 7).

El desarrollo de prototipos destinados al monitoreo de parámetros eléctricos es posible gracias a las tecnologías de la industria 4.0 y a plataformas de bajo costo. Una aplicación IoT con

una arquitectura asequible permitirá su implementación en nodos de medición en el sector eléctrico, tal es el caso de medidores de energía eléctrica residencial, pues un usuario al poder visualizar su consumo energético detallado en tiempo real, a través de una interfaz gráfica permitirá concientizar sobre el adecuado uso de la energía eléctrica, y a la vez tomar medidas de ahorro energético de acuerdo con el monitoreo. Por otro lado, empresas prestadoras de servicio eléctrico deben dar a conocer los consumos reales mediante dispositivos tecnológicos pertinentes y acorde a su capacidad técnica y financiera, de ahí la importancia del desarrollo de dispositivos o aplicaciones de monitoreo eficientes.

De forma directa se ofrecerá una aplicación IoT innovadora para usuarios residenciales y para empresas prestadoras de servicio eléctrico del país, y a la vez se pretende fortalecer el pilar de la investigación y aportar en el avance tecnológico en materia de eficiencia energética. Se añade que el presente trabajo servirá de base o consulta para futuros trabajos relacionados con las tecnologías de Industria 4.0 y plataformas de bajo costo.

La Revista Panorama Eléctrico edición 10, compila datos sobre indicadores representativos del área eléctrica ecuatoriana. Además, ofrece una reseña sobre la Agencia de Regulación y Control de Energía y Recursos Naturales no Renovables y del Sector Eléctrico (ARCERNNR). En esta edición con fecha a febrero de 2022, en la sección de distribución se detalla sobre los principales elementos de infraestructura de las empresas de distribución eléctrica del país, Ecuador a nivel general dispone de 5'470.596 medidores eléctricos, los cuales se hallan ligados para cada una de las empresas de distribución que son 20, de esto se deduce que el campo de aplicación del sistema IoT a desarrollar puede gozar de gran apertura (ARCERNNR, 2022, p. 6).

CAPÍTULO I: DESCRIPCIÓN DEL PROYECTO

1.1. Contextualización general del estado del arte

El IoT es una tecnología que actualmente se encuentra implementando sistemas enriquecidos de dispositivos, controladores, subsistemas interconectados permitiendo nuevos modelos de innovación digital etc., su aplicación se enfoca en los relacionados con la domótica, impulsando el desarrollo de diversos estudios y trabajos. Debido al despliegue del IoT, se han desarrollado aplicaciones y servicios innovadores, ofreciendo modelos integradores de medición, monitoreo, control, transmisión de datos etc. Recientemente la aplicación de esta tecnología a sistemas de eficiencia energética ha motivado interés, tal es el caso del monitoreo de variables eléctricas en tiempo real, pues la sociedad actual ha empezado a implementar tecnología en sus actividades cotidianas y en entornos donde frecuentemente se desenvuelve. El hogar es uno de los ambientes donde ya se evidencia la adaptación de sistemas de monitoreo y control, ya sea con fines de confort o solución de problemas, contrariamente a esto empresas prestadoras del servicio eléctrico emplea recursos humanos y económicos elevados para llevar a cabo ciertas actividades como la recolección de lecturas de medidores residenciales, lo cual puede ser mejorado mediante sistemas IoT escalables.

Antes de abordar los trabajos relacionados con el desarrollo de aplicaciones IoT para monitoreo, en primera instancia se recalca la tesis doctoral de Yacchirema, (2019) titulada “Arquitectura de interoperabilidad de dispositivos físicos para el Internet de las Cosas (IoT)” expone que el IoT, dentro del Internet constituye un gran avance, lo conceptualiza como “una red de objetos físicos (cosas) que incorpora sensores, software y otras tecnologías con el fin de conectar e intercambiar datos con otros dispositivos y sistemas a través del Internet”. La autora hace hincapié que la interoperabilidad representa una necesidad de suma importancia para el avance del IoT, por lo cual acentúa el gran desafío que significa alcanzar una comunicación entre tipos de dispositivos o sistemas que posean diferentes características técnicas. En un ecosistema IoT, la posibilidad que dos o más entidades heterogéneas se comuniquen, intercambien datos y/o utilicen información entre sí, se conoce como interoperabilidad (pp. 13-36). Ante esta extensa investigación, Yacchirema, además de recopilar características esenciales del IoT y respectivos casos de estudio de esta tecnología, detalló sobre representativas arquitecturas guía para el desarrollo de proyectos o prototipos IoT en un contexto real. Gracias a este estudio se seguirá una arquitectura IoT de alto nivel de cuatro etapas o capas que son: percepción, conectividad, servicio y aplicación, la cual resultó idónea para el desenvolvimiento de este trabajo.

A continuación, se detallará trabajos de diseño e implementación de aplicaciones IoT para la monitorización de variables, teniendo en cuenta la estructura IoT de cuatro niveles.

En 2022, Cutus y Recalde desarrollaron un sistema domótico con fines de seguridad, utilizaron dispositivos multiplataforma, los autores para la construcción de este sistema definieron tres elementos fundamentales: nodo de sensores, controlador y actuadores e interfaces. Donde cada nodo sensor se constituyó por el módulo ESP8266, microcontrolador de monitoreo de los sensores, obtuvieron así datos para su protocolo seleccionado que fue MQTT el cual los enviará al controlador del sistema; configuraron en *Raspberry Pi 4* (controlador) un bróker de mensajería MQTT mediante el módulo ESP8266 y enlazaron los datos censados, instalaron el software *Node-RED* para notificación a actuadores ante eventos sospechosos; por último a través de una aplicación móvil el usuario podría recibir notificaciones sobre la seguridad de su vivienda, donde los Smartphones fueron sus actuadores. Los autores concluyeron que su sistema domótico contó con una precisión del 100% tanto para el control, transmisión y recepción datos cumpliendo los aspectos de seguridad planteados y sobre todo mencionaron que los dispositivos de su sistema consumieron un mínimo de energía (Cutus & Recalde, 2022, p. 75). Los autores emplearon *Firebase* como base de datos de su proyecto domótico, además resaltan el uso de *Raspberry Pi 4*, la cual sin inconvenientes ejecutó el servicio de comunicación mediante MQTT, protocolo que recomiendan para futuros proyectos domóticos complejos. De esto se evidencia el uso de *Raspberry Pi 4* y del protocolo MQTT, los cuales se emplearán en el presente proyecto.

Los autores Angelino y otros, (2022) diseñaron un prototipo IoT que lo implementaron a un sistema de generación fotovoltaica (FV) obteniendo una aplicación de costo bajo, código abierto y hardware programable con enfoque educativo. El desarrollo de este sistema, lo hicieron mediante una secuencia de cinco partes: sistema embebido con conexión wifi, almacenamiento de datos en la nube, detección, data entregada por inversor y por último programación del sistema embebido. Utilizaron un microcontrolador ESP32 por su capacidad de procesamiento y almacenamiento, un sensor de corriente ACS712-20A y resistores de voltaje como principales dispositivos, donde los datos de voltaje y corriente obtenidos por ESP32 fueron enviados a una base de datos en la nube permitiendo el acceso desde cualquier parte del mundo. Con la implementación del sistema didáctico y mediante su operación obtuvieron datos que fueron comparados con la data suministrada por el inversor, logrando así la validación del sistema didáctico. Esta arrojó un error menor a 1% concluyendo que el prototipo de bajo costo puede ser empleado para medir plantas FV semejantes a la del proyecto (pp. 44-50). Esta investigación se orientó al diseño, desarrollo y validación de un prototipo didáctico de monitoreo de voltaje y

corriente de un sistema FV, mediante este tipo de trabajos se promueve la enseñanza de conceptos de IoT y de generación FV, energía renovable que merece ser impulsada. Además, se resalta la realización de validaciones a los sistemas de monitoreo, puesto que esto representa una gran contribución para esta área. Los autores recomendaron la aplicación de inteligencia artificial aplicada a la data obtenida para detección de fallos.

Para lograr el almacenamiento y gestión de datos en aplicaciones IoT se requiere de software que permitan el intercambio de mensajes entre los diferentes dispositivos de una arquitectura IoT, por ello en su trabajo de maestría “Sistema de captura y monitorización de variables fisiológicas basado en tecnologías IoT y plataformas de bajo coste” Pons, (2021). El autor desarrolló un prototipo orientado al área de salud para almacenar y monitorear señales fisiológicas. Para esto el autor inició con la selección del microcontrolador para la captura de las señales a monitorear mediante un estudio riguroso de las distintas alternativas, eligió Arduino Nano 33 IoT. Posteriormente analizó los distintos protocolos de comunicación y servidores existentes, seleccionando MQTT y Elipse Mosquito respectivamente. Además, seleccionó como base de datos a *InfluxDB*, a *Node-RED* para la programación entre el *broker* e *InfluxDB*, seguidamente usó Grafana para el monitoreo de la data recopilada. El autor a través de pruebas para la gestión de alertas con *Node-RED*, y con la implementación del módulo ESP32 que no fue contemplado inicialmente, obtuvo como resultado un prototipo IoT de bajo costo para el monitoreo y registro de señales biológicas en tiempo real (Pons, 2021, pp. 66-75). Utilizando plataformas de bajo coste y software de código abierto el autor desarrolló su prototipo de monitoreo, aspecto relevante para el desarrollo del presente trabajo; otro aspecto que se resalta es *Node-RED* como base de datos, pues posee una amplia librería incluso para generar alertas y notificaciones. El autor mediante su trabajo recomendó el uso de nuevos protocolos y dispositivos de bajo costo, sobre todo orientó el desarrollo de una aplicación que a más del monitoreo también efectuó el control.

Hoy en día existen otras formas de comunicación que representan un futuro en las redes, lo que permitirá aprovechar de mejor manera las tecnologías actuales entre ellas el IoT. Para responder a inconvenientes de baja escalabilidad que dispone la infraestructura de red convencional Cobos, (2021) en su trabajo denominado “Diseño e implementación de una red virtual basada en *Docker* en un ambiente de redes definidas por software (SDN) utilizando *ZeroTier* y *Raspberry Pi*”, con el diseño desarrollado se demostró una solución basada en redes definidas por software, destacando que hay otros caminos de comunicación con la tecnología SDN que actualmente se está abriendo paso, donde cualquier dispositivo mediante una dirección de red de *ZeroTier* puede ponerse en contacto con cualquier dispositivo en cualquier

momento (p. 218). Este es un trabajo que abre puertas para la realización de futuros despliegues en cuanto a IoT, pues brinda una solución novedosa de aplicación a través de *ZeroTier*.

En el TFM “Sistema de control y monitoreo con sistemas embebidos para entornos industriales” su autor Paredes, (2020) ante el bajo nivel de monitorización y control de procesos industriales en Ecuador, desarrolló un sistema de control y monitoreo con sistemas embebidos. Para esto utilizó principalmente la *Raspberry Pi 4* y el *NodeMCU ESP8266* la cual fue programada usando el IDE de Arduino y el software *Node-RED* para la programación por su capacidad de brindar conexión a la nube y a dispositivos hardware. Usó el protocolo MQTT que permitió el intercambio de información, donde *Raspberry Pi* actuó como un *broker* y como la plataforma de programación de *Node-RED*. Finalmente, el autor realizó pruebas de su prototipo indicando que en la parte monitoreo obtuvo una variación del 1% y en control del 100%, esto lo comparó con valores patrones de la Norma INEN, obteniendo un grado de aceptabilidad. Por otro lado, el prototipo fue viablemente económico respecto a otros (pp. 85-91). El autor alcanzó una solución práctica, novedosa y de bajo costo, pues los hardware principales son de código abierto y sobre todo están al alcance para cualquier persona y empresa. De esto se resalta la forma de validación del prototipo puesto que el autor se apoya de una norma.

En fin, ante el gran avance de la tecnología, empresas desarrolladoras de software y/o hardware, cada año lanzan al mercado nuevos productos o servicios dejando obsoletos a sus antecesores, por ejemplo, Arduino ha desarrollado una larga lista de microcontroladores con diversas funcionalidades, a esto se añade que esta empresa posee su propia plataforma de desarrollo. Sin embargo, *Raspberry Pi Foundation* se caracteriza por disponer de dispositivos de bajo costo y con sus propias funcionalidades, tal es el caso de su último lanzamiento *Raspberry Pi 4 Model B*. Esto hace referencia a la gran variedad de dispositivos y plataformas que se ofertan actualmente, cada uno con características intrínsecas como: tamaño reducido, bajo costo y consumo de potencia, código abierto entre otras, incrementando el la investigación y desarrollo de aplicaciones IoT para el monitoreo y control de variables como los eléctricos. Dicho de otra forma, los estudios e investigaciones mencionadas anteriormente plantean la integración de tecnologías *cloud* y dispositivos IoT, mediante la propuesta de entornos que permitan la interacción entre sus componentes, plataformas de procesamiento y visualización de información desplegados en la nube.

Mediante la revisión bibliográfica realizada, se debe tomar muy en cuenta la importancia de la interoperabilidad, característica que debe poseer toda aplicación IoT independiente de su área de aplicación, motivo por el cual esta deberá ser ofrecida por los proveedores, pues ante

los distintos protocolos de comunicación, dispositivos IoT con diferentes limitaciones a nivel de procesamiento de recursos computacionales, se requiere de un protocolo de comunicación fiable, escalable, ligero, seguro e interoperable. El IoT como una tecnología en auge debe ser explorada de manera profunda a través de estudios e investigaciones con enfoque general y específico, por ello el principal objetivo del presente trabajo radica en el diseño e implementación de una aplicación IoT para el monitoreo de parámetros eléctricos utilizando hardware y plataformas asequibles económicamente. En base a todo lo expuesto la *NodeMCU ESP8266* y *Raspberry Pi 4* se convertirán en el microcontrolador y microprocesador respectivamente para aplicación IoT propuesta. *Raspberry Pi 4* almacenará a MQTT como protocolo de comunicación principal y a su servidor (Mosquitto), además se le instalará *Node-RED*, *InfluxDB*, y *Grafana* permitiendo la posibilidad de visualizar los parámetros eléctricos medidos con PZEM-004T.

1.2. Proceso investigativo metodológico

El tipo de investigación a utilizar en este trabajo es mixto entre lo cualitativo y cuantitativo. Se utilizará la investigación cualitativa debido a la extensa revisión bibliográfica de tesis electrónicas, libros, artículos de revista etc., todos relacionados a la temática del empleo de tecnología IoT para el diseño y desarrollo de aplicaciones o sistemas de monitoreo. Con esto se alcanzará a recopilar, seleccionar y clasificar material teórico, conceptual y metodológico para este proyecto; consecuentemente se detallará aspectos claves, ventajas y características extraídos de las fuentes primarias y secundarias revisadas, permitiendo discernir información base y de apoyo para el desarrollo de la aplicación planteada. De manera puntual la investigación toma un enfoque cuantitativo puesto que se obtendrán volúmenes de datos de consumo y otros parámetros eléctricos, los cuales se representarán en un panel de visualización accesible para el usuario final. Y por otro lado se medirán valores de voltaje y corriente. En fin, el desarrollo de la aplicación IoT propuesta será secuencial y experimental.

El método analítico se aplicará para establecer características de equipos, dispositivos, software adicionales, determinación de una arquitectura, entre otros. A raíz de un análisis y síntesis de trabajos, se utilizará el método de inducción-deducción. Estos métodos permitirán inferir y ampliar los diferentes criterios, teorías, aplicaciones y enfoques de distintos autores, con el apoyo de fuentes bibliográficas primarias y secundarias orientadas a dispositivos IoT. En el presente trabajo se plantea el desarrollo de una aplicación IoT, por esto el método experimental, que en este caso se llevará a cabo en dos etapas: la primera se enfocará en pruebas preliminares de los dispositivos IoT, sensores que integrarán la aplicación y pruebas de

software; la segunda se desarrollará específicamente en la residencia a monitorear para obtener valores de parámetros eléctricos, principalmente consumo que alimentarán a la aplicación web, y a su vez datos que servirán de soporte para el desarrollo del presente proyecto. Es importante realizar una caracterización de los datos obtenidos por ello se empleará la técnica de muestreo para la presentación de resultados.

La observación cumplirá un rol fundamental durante el desarrollo del presente proyecto dado que es una herramienta empírica de reconocimiento de problemas y de recolección de datos e información. Se va a realizar diagramas de flujo con el fin de organizar y presentar la secuencia del desarrollo de la aplicación IoT, así como de gráficos para consolidar y explicar la información obtenida. Para la validación de la aplicación IoT se realizará una encuesta a especialistas sobre la temática con el fin de obtener sus criterios sobre el prototipo, el número de participantes serán tres.

Scrum, es una de las metodologías ágiles más utilizadas para desarrollo de prototipos, aplicaciones, software etc., además resulta ser óptima para la planificación y seguimiento de proyectos de manera general. Por lo expuesto se va a emplear y adaptar esta metodología para el desarrollo de la aplicación IoT propuesta.

CAPÍTULO II: PROPUESTA

2.1 Fundamentos teóricos aplicados

Introducción

Actualmente no se dispone de una definición universal del Internet de las cosas, esto ante el desarrollo de nuevas tecnologías y a la mejora de algunas existentes. Una de las definiciones más relevantes es la dada por el *Institute of electrical and Electronics Engineers (IEEE)*, plantean que, ante los escenarios de aplicación, el IoT es “una red que conecta cosas identificables de forma única a Internet, las cosas tienen capacidades de detección/actuación y programación (Yacchirema, 2019, p. 18). El IoT se orienta concretamente al ser humano, por ello se propone el desarrollo de una aplicación IoT para el monitoreo del consumo energético residencial.

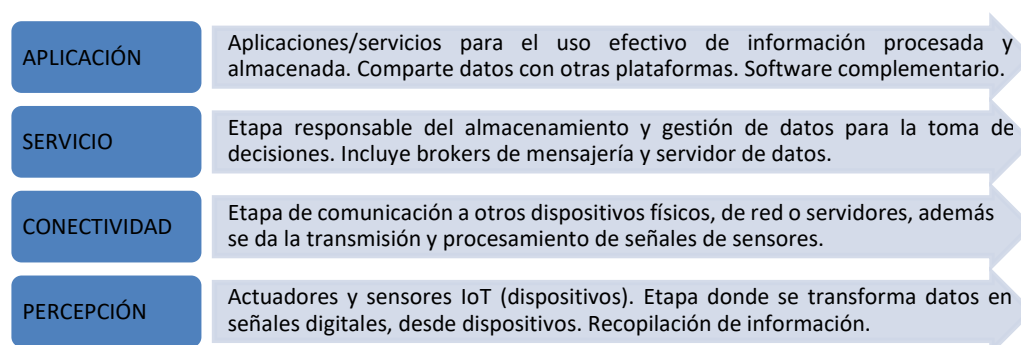
En el presente apartado se abordarán los fundamentos teóricos requeridos para el desarrollo de la aplicación IoT planteada, partiendo de una arquitectura de alto nivel que permita la interacción de todos los dispositivos, plataformas de procesamiento y visualización de información que constituirán el prototipo del presente trabajo.

Arquitectura IoT de cuatro niveles

El aspecto inicial para el correcto diseño e implementación de prototipos IoT, depende en su mayoría de una arquitectura guía. Esto debido a los usos de los dispositivos a emplear, pues se necesita disponer de funcionalidades específicas. Por lo mencionado se seguirá una arquitectura de cuatro niveles como se puede visualizar en la Figura 1.

Figura 1

Estructura IoT de alto nivel



Nota. La figura representa a la arquitectura IoT de cuatro niveles. Adaptado de *Arquitecturas de IoT* (p. 25), por Yacchirema, 2019.

Nivel de percepción

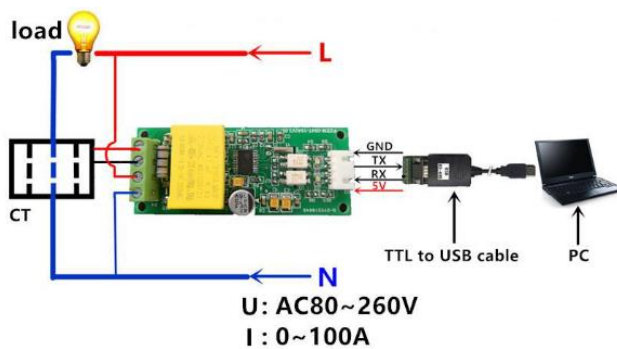
La primera etapa de la arquitectura definida representa principalmente a los dispositivos IoT (sensores y actuadores).

PZEM-004T

Es una tarjeta de comunicación y seguimiento eléctrico, obtiene medidas de: voltaje, corriente, potencia, frecuencia, factor de potencia, energía. Se cataloga como un medidor multifunción permitiendo restituir un valor de energía acumulado y almacenar energía medida antes de apagarse sin reponer datos al encender otra vez el sistema. Mediante un puerto de comunicación TTL (ver Figura 2) es capaz de conectar una LCD/LED, permitiendo ver datos y la comunicación con otros dispositivos (Romero, 2017, p. 21).

Figura 2

Diagrama de conexión de PZEM-004T-100 A



Nota. Reproducido del Diagrama de cableado PZEM-004T-100 A, por Manuales +, 2022.

Raspberry Pi 4 Model B

Raspberry Pi fue desarrollada con fines investigativos informáticos, integrando en sus capacidades el uso de dispositivos electrónicos con lenguajes de programación como Scratch, Java, JavaScript o Python. Raspberry Pi 4 Model B es un microprocesador potente que contiene puertos de entrada y salida como cualquier computador, además de un conjunto de pines GPIO (General Purpose Input/Output), brindando la capacidad de conectar actuadores y sensores (Cutus & Recalde, 2022, p. 30). Raspberry Pi 4 requiere de un teclado para ingreso de comandos, un monitor y una fuente de poder, sus principales ventajas son (Paredes, 2020, p. 16):

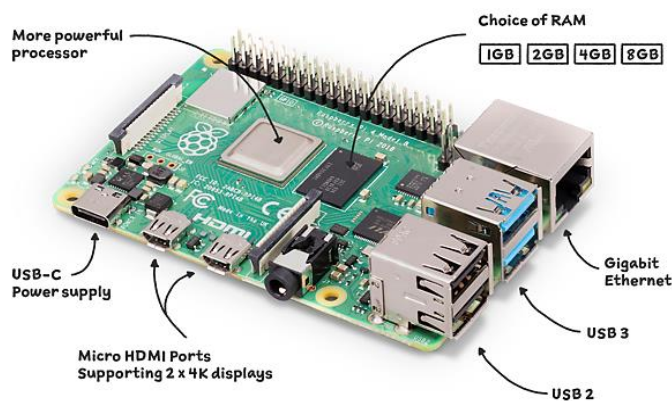
- Posee memoria RAM suficiente y conectividad Wifi y Bluetooth.
- Puede expandir la memoria para almacenamiento de datos (hasta 64Gb), además de sus funcionalidades usando placas shields.

- Posibilita el uso de componentes electrónicos mediante entradas y salidas digitales con protocolos I2C o SPI, también lenguajes de programación como *C*, *Python*, *C++* y *Java*.
- Sirve de soporte de algunos sistemas operativos como: GNU/Linux ARM (*Debian*, *Fedora*, *Arch Linux*), RISC OS2, *Windows 10 IoT CORE*, etc. Se destaca la opción de Linux denominada *Raspbian*, la cual es de código abierto.

En la Figura 3 se muestran todos los elementos que componen a *Raspberry Pi*.

Figura 3

Raspberry Pi 4 Modelo B



Nota. Reproducida de *Raspberry Pi 4*, por *Raspberry Pi Foundation*, 2019.

Las características de *Raspberry Pi*, en especial Pi 4 modelo B se convierte en el microprocesador idóneo para desarrollar el prototipo IoT propuesto. Sobre esta se configurará el servidor MQTT y se instalará el software *Node-RED*.

NodeMCU ESP8266

Es un microcontrolador de desarrollo con firmware de código abierto, encargado de procesar datos y ejecutar software, por esto es utilizada en la construcción de aplicaciones IoT. Como tal es un kit/placa complejo similar a Arduino, pero con la posibilidad de conectarse a Internet con la integración de ESP8266. Posee múltiples pines GPIO permitiendo la conexión con otros periféricos generando PWM y comunicaciones I2C, SPI, UART y serial (Paredes, 2020, p. 17). Se destaca su utilidad en el desarrollo de proyectos IoT, puesto que se caracteriza por su software y hardware libre. La NodeMCU ESP8266 se visualiza en la Figura 4, dispositivo que será empleado para el presente proyecto debido a las características mencionadas.

Figura 4

NodeMCU ESP8266



Nota. Reproducida de *NodeMCU ESP8266* (p.17.), por Paredes, 2020.

Nivel de conectividad

Nivel encargado de conectarse a otros dispositivos físicos, servidores y dispositivos de red. En este se abordará sobre el protocolo MQTT, el cual se utilizará para la aplicación planteada.

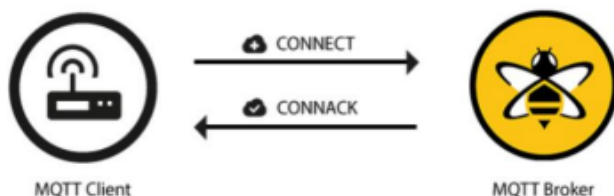
Message Queuing Telemetry Transport

Popularmente conocido como MQTT, se conceptualiza como un protocolo de transporte de mensajería de publicación y suscripción del servidor cliente, se caracteriza por ser ligero, de código abierto y de sencilla implementación. Es ideal cuando se presentan situaciones de comunicación limitada o donde se necesite de un pequeño código y/o el ancho de banda es indispensable. Para comunicaciones de máquina a máquina o M2M e IoT, MQTT resulta idóneo (HiveMQ, 2020, p. 5). Los elementos principales en la comunicación MQTT son: (Cobos, 2021, p. 11)

- Cliente: dispositivos que hablan MQTT sobre una pila TCP/IP.
- *Broker*: determina quién está suscrito a cada mensaje y lo envía.
- Conexión: Se ejecuta en medio del cliente y *broker* (Ver Figura 5).

Figura 5

Diagrama de conexión MQTT



Nota. En la figura se presenta el envío de un mensaje connect por parte de un cliente al intermediario, y el bróker responde con un connack. Reproducida de *MQTT Connection* (p.12), por HiveMQ, 2020.

Nivel de servicio

Dentro de este se encuentra el eclipse *Mosquito* y *Node-RED*, encargados de la gestión de datos, almacenamiento y procesamiento de información.

Mosquitto

Es un servidor que implementa las versiones 5.0, 3.1.1 y 3.1 del protocolo MQTT. Sirve de intermediario para mensajería de código abierto. Es ligero y pertinente para emplearlo en computadoras de placa única de bajo consumo, inclusive servidores completos (Mosquitto, 2021). Mediante la publicación o suscripción de sus clientes tomando como base el *topic* se puede gestionar información de forma eficiente y segura.

InfluxDB

Es una base de datos código abierto, utiliza protocolos HTTP, TCP, MQTT para almacenar información, se caracteriza por permitir ingresar gran cantidad de escrituras por segundo, por cubrir requerimientos de monitoreo IoT y por recopilar información en tiempo real. Su lenguaje es *InfluxQL* (Coloma & Rodríguez, 2020, p. 25). Este en su versión gratuita permite un almacenamiento de sus datos hasta 30 días.

Nivel de aplicación

Este nivel hace referencia a las aplicaciones software y a los servicios de usuarios, donde se hace uso efectivo de la información procesada.

Node-RED

Potente herramienta de programación gráfica y de código abierto (*Open Source*), utilizada especialmente para aplicaciones controladas por eventos IoT (Yacchirema, 2019, p. 176). *Node-RED* comunica distintos dispositivos de hardware y servicios, se creó a partir de *NodeJs* y la librería *Java D3.js*; es sencilla e intuitiva de aprender, y además es muy compatible con Raspberry Pi (Cutus & Recalde, 2022, pp. 13-14). La programación propuesta por este es amigable, pues se la realiza mediante la conexión de flujos a través de nodos gráficos.

Grafana

Software de código abierto, caracterizado por visualizar, analizar, monitorear datos de series temporales o logs. La información es presentada en forma gráfica o dashboard, la cual es

obtenida de base de datos conectadas a ella por medio de otras herramientas (Coloma & Rodríguez, 2020, p. 25).

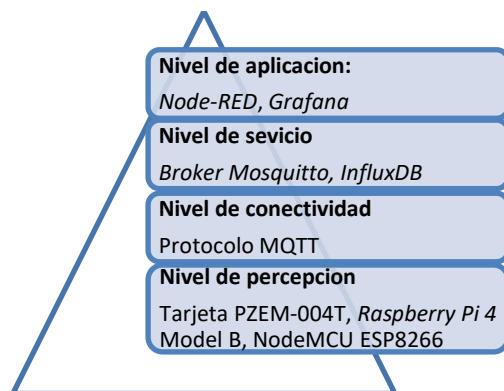
2.2 Descripción de la propuesta

a. Estructura general

La propuesta de este trabajo se basa en el diseño e implementación de una aplicación IoT para el monitoreo de consumo de energía eléctrica residencial. La implementación exitosa de trabajos IoT se relaciona directamente con la determinación de una arquitectura de referencia, como se mencionó en apartados anteriores, por esto se eligió una de cuatro niveles como se sintetiza en la Figura 6, donde además se observa la relación de dispositivos, software y servidores a utilizar en el presente proyecto.

Figura 6

Arquitectura de cuatro niveles planteada



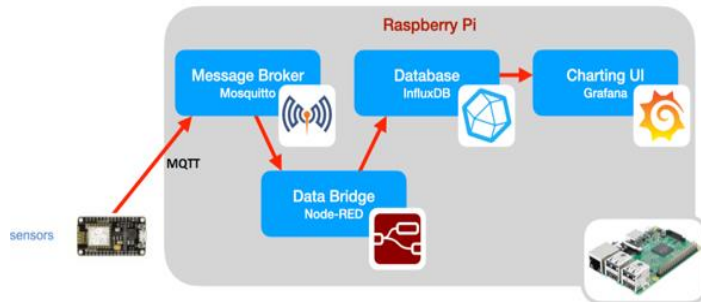
Ante la identificación de los distintos componentes básicos a emplear en la aplicación IoT propuesta, se realizó un diagrama de bloques para llevar a cabo el registro de datos como se puede visualizar en la Figura 7. Se observa una combinación particular de software alojados en una Raspberry Pi, que resulta ideal para la automatización en residencias.

Siguiendo la Figura 7, la aplicación IoT propuesta se va a implementar con un hardware de código abierto *NodeMCU ESP8266*, mediante el cual se obtendrá los datos proporcionados por el módulo PZEM 004T, posterior a esto se enviará la información obtenida a través de un protocolo eficiente y escalable en este caso MQTT, hacia un *Gateway* implementado en el *broker Mosquitto* que se instalará en el microprocesador (*Raspberry Pi 4*). Esta manejará una capa de abstracción (*Node-RED* e *InfluxDB*) que permitirá la comunicación bidireccional entre aplicaciones en la nube. Por último, cualquier dispositivo que sea considerado cliente puede visualizar la información de los principales parámetros eléctricos medidos en un dashboard

desarrollado en la plataforma de código libre *Grafana*, permitiendo un análisis de las métricas presentadas.

Figura 7

Diagrama de bloques para el registro de variables eléctricas

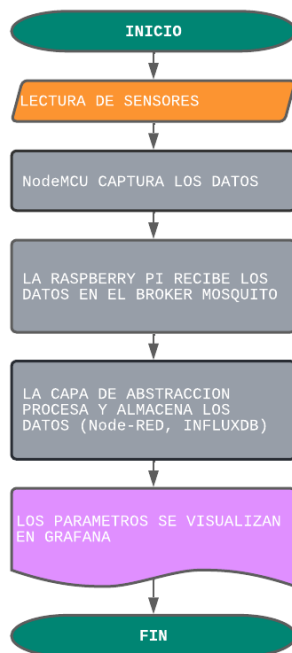


Nota. Adaptado de Data logging building blocks, por OXER, 2020

También se elaboró un diagrama de flujo del proceso para desarrollar la aplicación IoT (ver Figura 8). Se notó que la última acción es la visualización de las variables eléctricas en *Grafana*, software que se instalará en la *Raspberry Pi 4*. En el dashboard se mostrará los dispositivos alojados en la red local, sin embargo, para extender la aplicación del presente trabajo se recomendará a *ZeroTier* como servicio de virtualización de red.

Figura 8

Diagrama de flujo de aplicación IoT Planteada



b. Explicación del aporte

Para desarrollo de la aplicación IoT que permita el monitoreo del consumo eléctrico residencial se inició con los siguientes pasos:

Lectura de sensores

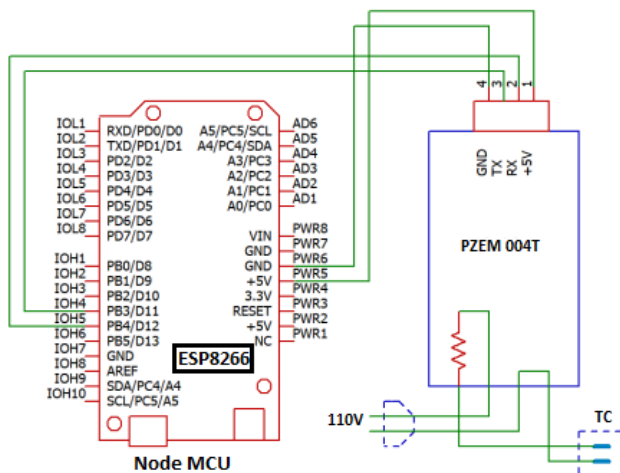
Uno de los dispositivos importantes de la aplicación IoT, es la tarjeta PZEM 004T 100A encargada de medir las distintas variables eléctricas a monitorear. Esta consta de una interfaz de comunicación TTL ideal para comunicarse con el dispositivo superior *NodeMCU*. En la Figura 9, se observa a la tarjeta PZEM 004T 100A conectada a 110V para medir voltaje, y a un TC (transductor de corriente) para la toma de mediciones de corriente.

Captura de datos y programación de *NodeMCU* ESP8266

El microcontrolador *NodeMCU* consta de un módulo wifi ESP8266 ideal para la comunicación requerida. En la Figura 9 se observa la PZEM 004T conectada a la *NodeMCU*, donde es importante la correcta alimentación de esta (5VDC) e identificar los pines a usar para la comunicación TTL (Rx ,Tx).

Figura 9

Diagrama de conexión de *NodeMCU* y PZEM 004T



Nota. El diagrama de esta figura fue empleado para la conexión física del dispositivo IoT propuesto. Adaptado de <https://n9.cl/do9r9>.

En cuanto a la programación de la tarjeta *NodeMCU* ESP8266, es muy importante la instalación de la librería *PZEM004Tv30.h* la cual permitirá la comunicación mediante la interfaz

serie (TTL UART), para esto se declaró los pines a utilizar (D1, D2) e igualmente las variables para los distintos parámetros eléctricos a ser medidos (ver Figura 10).

Figura 10

Declaración de variables en IDE Arduino

```
//Configuración Smart-Meter
PZEM004Tv30 pzem(4, 5); //Pines TX y RX conectados a la NodeMCU D1,D2

char svoltaje[20];
char scorriente[20];
char spotencia[20];
char senergia[20];
char sfrecuencia[20];
char sfactorpotencia[20];
```

Para capturar los datos de las variables eléctricas medidas se empleó el código que se encuentra en el Anexo 1.

Instalación de software necesarios para la aplicación IoT

Antes de la instalación de los servidores es importante especificar el uso de un sistema operativo que en este caso fue *Raspbian Bullseye*, versión más actual. Este fue descargado y almacenado en una memoria SD y luego colocada en la Raspberry Pi 4. Los software instalados fueron Eclipse *Mosquitto*, *Node-RED*, *InfluxDB* y *Grafana*, para esto se abrió la interfaz de línea de comandos (CLI) y se digitó los códigos, que se detallan en la Tabla 1.

Tabla 1

Comandos de instalación de software

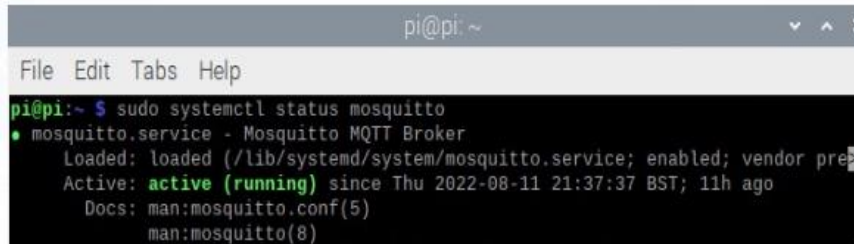
Broker Mosquitto	
<code>sudo apt-get update</code>	Actualiza el sistema operativo y librerías.
<code>sudo apt-get install mosquitto</code>	Instala el servidor Mosquitto.
<code>sudo apt-get install mosquitto-clients</code>	Instala los clientes Mosquitto.
Servicio Node-RED	
<code>sudo apt-get install nodered</code>	Instala el servidor <i>Node-RED</i> .
<code>sudo apt-get install npm</code>	Instala el sistema de paquetes <i>Node.js</i> .
Base de Datos InfluxDB	
<code>sudo apt install influxdb</code>	Instala el servicio <i>InfluxDB</i> .
Servidor Grafana	
<code>sudo apt install grafana</code>	Instala el servidor <i>Grafana</i> .

Nota. En esta tabla se enumera cada comando utilizado para la instalación de los software y servidores empleados, cada uno presenta su descripción.

En la figura 11 se muestra que el *broker Eclipse Mosquito* está instalado correctamente, se recalca que este se ejecutará en segundo plano al momento del arranque de *Raspberry Pi*. Se observa el mensaje *active (running)* indicando que todo está bien.

Figura 11

Eclipse Mosquitto instalado correctamente

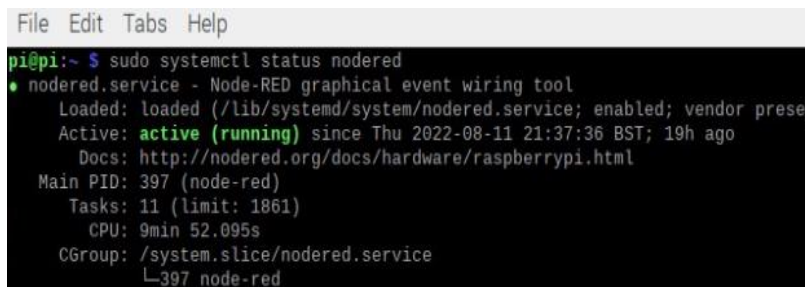


```
pi@pi: ~  
File Edit Tabs Help  
pi@pi:~$ sudo systemctl status mosquitto  
● mosquitto.service - Mosquitto MQTT Broker  
   Loaded: loaded (/lib/systemd/system/mosquitto.service; enabled; vendor prese  
   Active: active (running) since Thu 2022-08-11 21:37:37 BST; 11h ago  
     Docs: man:mosquitto.conf(5)  
           man:mosquitto(8)
```

Mediante el mensaje *active (running)* que se observa en la Figura 12, se demuestra que *Node-RED* ha sido instalado correctamente. Al igual que *Eclipse Mosquitto*, *Node-RED* se ejecuta en segundo plano al arranque de *Raspberry*.

Figura 12

Node-RED instalado correctamente

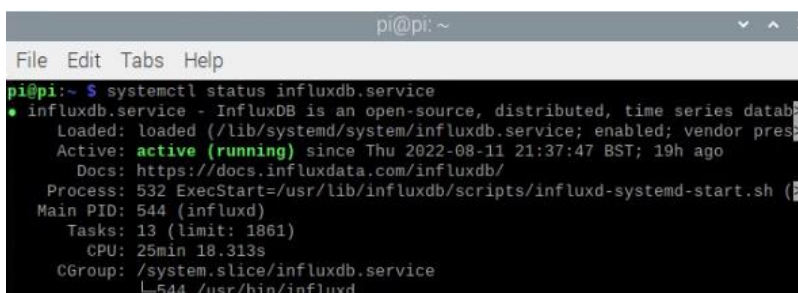


```
File Edit Tabs Help  
pi@pi:~$ sudo systemctl status nodered  
● nodered.service - Node-RED graphical event wiring tool  
   Loaded: loaded (/lib/systemd/system/nodered.service; enabled; vendor prese  
   Active: active (running) since Thu 2022-08-11 21:37:36 BST; 19h ago  
     Docs: http://nodered.org/docs/hardware/raspberrypi.html  
   Main PID: 397 (node-red)  
     Tasks: 11 (limit: 1861)  
    CPU: 9min 52.095s  
   CGroup: /system.slice/nodered.service  
           └─397 node-red
```

En la Figura 13 se observa que la base de datos *InfluxDB*, ha sido instalada correctamente, la cual también se ejecutará en segundo plano. Así mismo es el mensaje *active (running)* indica que la base de datos está corriendo con normalidad. *InfluxDB* también se ejecuta en segundo plano.

Figura 13

Base de Datos InfluxDB instalado correctamente.



```
pi@pi: ~  
File Edit Tabs Help  
pi@pi:~$ systemctl status influxdb.service  
● influxdb.service - InfluxDB is an open-source, distributed, time series datab  
   Loaded: loaded (/lib/systemd/system/influxdb.service; enabled; vendor prese  
   Active: active (running) since Thu 2022-08-11 21:37:47 BST; 19h ago  
     Docs: https://docs.influxdata.com/influxdb/  
   Process: 532 ExecStart=/usr/lib/influxdb/scripts/influxd-systemd-start.sh (p  
   Main PID: 544 (influxd)  
     Tasks: 13 (limit: 1861)  
    CPU: 25min 18.313s  
   CGroup: /system.slice/influxdb.service  
           └─544 /usr/bin/influxd
```

Por último, en la Figura 14 se visualiza que el servidor *Grafana* se está ejecutando sin inconvenientes desde la fecha de instalación. Con la instalación de Grafana se culminó con los software necesarios para la aplicación IoT, en este se mostrarán los parámetros eléctricos monitoreados. Se destaca que todos los software y servidores instalados se ejecutan en segundo plano.

Figura 14

Servidor Grafana instalado correctamente

```

pi@pi: ~
File Edit Tabs Help
pi@pi:~$ sudo systemctl status grafana-server
● grafana-server.service - Grafana instance
   Loaded: loaded (/lib/systemd/system/grafana-server.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2022-08-11 21:37:36 BST; 19h ago
     Docs: http://docs.grafana.org
   Main PID: 530 (grafana-server)
    Tasks: 16 (limit: 1861)
      CPU: 4min 37.589s
   CGroup: /system.slice/grafana-server.service
           └─530 /usr/sbin/grafana-server --config=/etc/grafana/grafana.ini
  
```

Programación de *Raspberry Pi* y *NodeMCU* ESP8266 para la comunicación.

Nuevamente para la comunicación se programó la *NodeMCU* mediante el IDE *Arduino*, donde es necesaria la instalación de la librería *PubSubClient.h*, la cual permite enviar y recibir mensajes MQTT. En esta sección es importante la configuración de la IP, donde se instaló el *broker Mosquitto* que a su vez corresponde a la IP 192.168.100.198 de la *Raspberry*, notándose que la *NodeMCU* actúa como cliente. Esto se observa en la Figura 15.

Figura 15

Programación de MQTT

```

//Tesis Maestria Angel Minta
//Energy Smart-Meter

#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <PZEM004Tv30.h>
#include <ESP8266WiFiMulti.h>
#include <SoftwareSerial.h>

ESP8266WiFiMulti WiFiMulti;

//Configuración de los parámetros de red

const char* ssid="Scaum-FastSystems"; //Nombre de la red WiFi
const char* password="9733sandra"; //Constraseña de la red WiFi
const char* mqtt_server = "192.168.100.198"; //Dirección IP del bróker MQTT (Raspberry Pi 4)//146
  
```

Para publicar los datos en el *broker* se empleó el puerto 1883, donde se destaca los temas con los cuales el cliente envía información para el *topic* “*medidor/voltaje*”, como se observa en la Figura 16. En el Anexo 1 se detalla todo el código empleado.

Figura 16

Publicación de topics MQTT

```
//Publicar temas MQTT del medidor

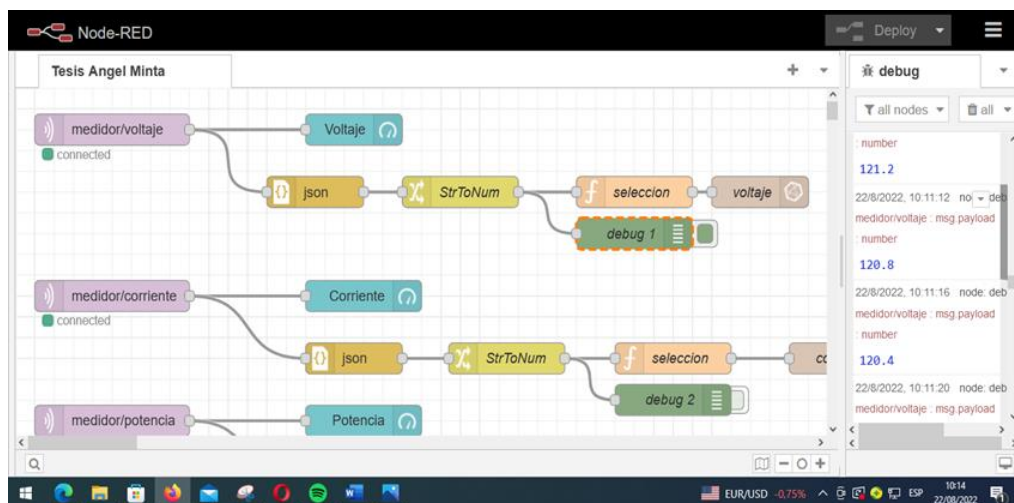
client.publish("medidor/voltaje", "Datos de voltaje: ");
client.publish("medidor/corriente", "Datos de corriente: ");
client.publish("medidor/potencia", "Datos de potencia: ");
client.publish("medidor/energia", "Datos de energia: ");
client.publish("medidor/frecuencia", "Datos de frecuencia: ");
client.publish("medidor/factorpotencia", "Datos de factor de potencia: ");
```

Con lo anteriormente mencionado la Raspberry Pi actúa como Gateway gestionando los datos de suscripción y publicación a través del servidor Mosquitto previamente instalado.

Para la recopilación de los datos se usó *Node-RED*, al cual se accedió mediante la dirección IP de la *Raspberry Pi 4*, y el puerto 1880 de la siguiente forma: 192.168.100.198:1880 como se ilustra en la Figura 17. Toda la programación realizada en *Node-RED* se podrá visualizar en el Anexo 2

Figura 17

Programación para la comunicación MQTT



De la Figura 17, se observa todos los nodos utilizados para la programación, donde se inicia con el nodo *mqtt in* que permite suscribirse a los mensajes de un tema específico, en este caso *medidor/voltaje*. Lo señalado se puede observar en la Figura 18, donde resaltan los parámetros importantes para configuración del nodo *mqtt in* que son *Server* y *Topic*.

Figura 18

Configuración del nodo mqtt in parte uno

The screenshot shows the 'Edit mqtt in node' interface. At the top, there are buttons for 'Delete', 'Cancel', and 'Done'. Below is a 'Properties' section with a gear icon and a document icon. The 'Server' dropdown menu is set to 'Mosquitto' and is highlighted with a red box. The 'Action' dropdown is set to 'Subscribe to single topic'. The 'Topic' text field contains 'medidor/voltaje' and is also highlighted with a red box. The 'QoS' dropdown is set to '0'.

Seguidamente con la configuración del nodo mencionado es necesario establecer la dirección del servidor (*Server*) y el protocolo de comunicación (*Protocol*) como se observa en las partes resaltadas de la Figura 19.

Figura 19

Configuración del nodo mqtt in parte dos

The screenshot shows the 'Edit mqtt in node > Edit mqtt-broker node' interface. At the top, there are buttons for 'Delete', 'Cancel', and 'Update'. Below is a 'Properties' section with a gear icon and a document icon. The 'Name' text field contains 'Mosquitto'. There are three tabs: 'Connection', 'Security', and 'Messages'. The 'Connection' tab is active. The 'Server' text field contains '192.168.100.198' and is highlighted with a red box. The 'Port' text field contains '1883' and is also highlighted with a red box. Below the 'Server' field, there is a checked checkbox for 'Connect automatically' and an unchecked checkbox for 'Use TLS'. The 'Protocol' dropdown menu is set to 'MQTT V3.1.1' and is highlighted with a red box.

Base de datos en *InfluxDB*

Para crear la base de datos se accede a la dirección 192.168.100.198:8086, inmediatamente esto da paso en la interfaz de *InfluxDB*, se procedió a llenar los campos de información que

detalla en la Tabla 2. Se observa que la base de datos obtenida se nombró como *Smart_Meter* para este trabajo.

Tabla 2

Campos de información para la creación de la base de datos Smart_Meter

Usuario	Angel
Contraseña	*****
Organización	Uisrael
Bucket	Smart_Meter

Una vez creada la base de datos *Smart_Meter*, se procedió con la configuración del nodo *influxdb out*, previamente se convirtió los *topics* creados a un tipo de dato numérico, mediante los nodos *Json* y *StrToNum*. Dando doble clic en el nodo a configurar, se seleccionó el servidor y se llenaron los campos de organización, *bucket* y *medición*. Los parámetros mencionados deben coincidir con la información que se usó para crear la base de datos. Los campos mencionados se resaltan en la Figura 20.

Figura 20

Configuración del nodo influxdb out parte uno

The image shows a configuration window titled "Edit influxdb out node". At the top, there are three buttons: "Delete", "Cancel", and "Done". Below this is a "Properties" section with a gear icon and three sub-panels. The main area contains five configuration fields, each with an icon and a text input or dropdown:

- Name**: voltaje
- Servidor**: [v2.0] BaseDatos (dropdown menu)
- Organización**: Uisrael
- Bucket**: Smart_Meter
- Medición**: voltaje

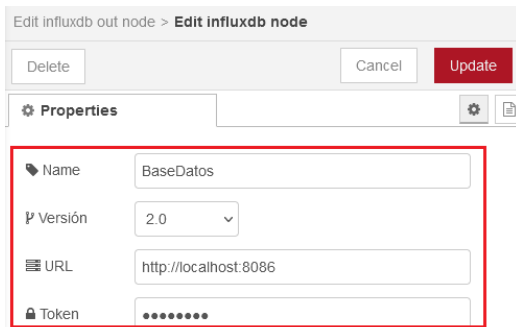
A red rectangular box highlights the "Organización", "Bucket", and "Medición" fields.

Posteriormente el valor numérico obtenido por *Json* y *StrToNum*, se enlazó el nodo con la base de datos *Smart_Meter* creada anteriormente con los parámetros versión 2.0, URL dirección localhost:8086 y el *token* (obtenido al momento del registro), estos campos de información se observan resaltados en la Figura 21.

De la misma forma se procedió para la configuración de los demás nodos que se visualizan en la Figura 17 como *medidor/corriente*.

Figura 21

Configuración del nodo influxdb out parte dos



De acuerdo con las configuraciones realizadas anteriormente, se obtuvo una interfaz en *InfluxDB*, obteniendo la base de datos nombrada como *Smart_Meter*, el parámetro *measurement* y la opción *field=voltaje_measurement=voltaje*, los cuales se hallan remarcados en la Figura 22. Los dos últimos permiten filtrar los datos que se requieran observar: corriente, energía, frecuencia etc. También se observa los datos de voltaje censados con su respectiva marca de tiempo.

Presentación de parámetros en Grafana

En apartados anteriores se instaló el servidor *Grafana* y para acceder a este se debe utilizar la dirección IP 192.168.100.198:3000. Para monitorear los parámetros eléctricos medidos es importante agregar la base de datos creada en *InfluxDB*, esto se realizó en *Grafana* en la opción de configuración donde se añadió una fuente de datos. Se configuró con los parámetros que se hallan detallados en la Tabla 3.

Tabla 3

Campos de configuración para agregar una base de datos desde InfluxDB

Nombre	InfluxDB Smart_Meter
Lenguaje	Flux
URL	http://localhost:8086
Usuario	Angel
Contraseña	*****
Organización	Uisrael
Tocken	*****
Base de datos	Smart_Meter

Figura 22

Interfaz InfluxDB Smart_Meter

The screenshot displays the InfluxDB Data Explorer interface. The main table shows the following data:

_start	_stop	_time	_value	_field	_measurement
2022-08-19 13:05:30 GMT-5	2022-08-19 14:05:30 GMT-5	2022-08-19 13:05:30 GMT-5	117.60	voltaje	voltaje
2022-08-19 13:05:30 GMT-5	2022-08-19 14:05:30 GMT-5	2022-08-19 13:05:50 GMT-5	121.25	voltaje	voltaje
2022-08-19 13:05:30 GMT-5	2022-08-19 14:05:30 GMT-5	2022-08-19 13:06:00 GMT-5	121.83	voltaje	voltaje
2022-08-19 13:05:30 GMT-5	2022-08-19 14:05:30 GMT-5	2022-08-19 13:06:10 GMT-5	121.80	voltaje	voltaje
2022-08-19 13:05:30 GMT-5	2022-08-19 14:05:30 GMT-5	2022-08-19 13:06:20 GMT-5	121.90	voltaje	voltaje
2022-08-19 13:05:30 GMT-5	2022-08-19 14:05:30 GMT-5	2022-08-19 13:06:30 GMT-5	116.85	voltaje	voltaje
2022-08-19 13:05:30 GMT-5	2022-08-19 14:05:30 GMT-5	2022-08-19 13:06:40 GMT-5	119.87	voltaje	voltaje

The interface also shows a sidebar with filters and settings. The 'FROM' section is set to 'Smart_Meter'. The 'Filter' section shows a dropdown for '_measurement' set to '5' and a list of selected fields: voltaje, energia, frecuencia, corriente, and factor_de_potencia. The 'WINDOW PERIOD' section is set to 'CUSTOM' and 'AUTO'. The 'AGGREGATE FUNCTION' section is set to 'CUSTOM' and 'mean'.

Una vez agregada la base de datos, se seleccionó una opción de panel para su posterior programación, esto en base al requerimiento del usuario final. Se programó el panel de la corriente para esto se empleó el código Flux que se ilustra en la Figura 23. Como se visualiza, para el código empleado se debe consultar el nombre de la base de datos, con una marca de tiempo específica de inicio y fin. Mediante filtros de medida y campo se accede a un parámetro determinado para posteriormente mostrarlo. De la misma manera se procedió con las demás variables a ser monitoreadas.

Figura 23

Código Flux para visualizar el panel de corriente

```
(InfluxDB_Smart_Meter)
from(bucket: "Smart_Meter")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "corriente")
  |> filter(fn: (r) => r["_field"] == "corriente")
  |> aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false)
  |> yield(name: "last")
```

Con el tipo de código Flux empleado se construyó un dashboard general, el cual se visualiza en la Figura 24. Los demás códigos flux empleados pueden ser revisados en el Anexo 3.

Figura 24

Dashboard general de la aplicación IoT planteada



Conexiones físicas de la aplicación IoT

La primera conexión que se realizó fue en la Raspberry Pi 4 relacionada a la forma de como alimentarla. El voltaje que se usa para este microprocesador es de 5VDC, que puede ser obtenido de adaptadores de corriente de baterías de celulares. De acuerdo con el entorno de implementación presentado, de un regulador de voltaje de un computador de escritorio se alimentó a la Raspberry Pi 4 usando el adaptador de corriente USB tipo C de 5VDC y 3A, como se observa en la Figura 25.

Figura 25

Conexión física Raspberry Pi



La segunda conexión realizada fue en la NodeMCU ESP8266 la cual fue insertada en un protoboard, para conectarla principalmente con la PZEM 004T. Se alimentó a la NodeMCU utilizando un cargador de 5VDC y de 3A, a través del puerto micro USB. Además, se usó el puerto serial de la PZEM 004T para la transmisión de los datos. La NodeMCU con la PZEM 004T se implementó en la caja de distribución de la residencia de estudio, esto puede ser visualizado detenidamente en la Figura 26.

Figura 26

Conexión física NodeMCU y PZEM 004T



c. Estrategias y/o técnicas

Para la programación e implementación del dispositivo IoT de monitoreo de consumo eléctrico, se eligió a SCRUM como metodología de trabajo, misma que permite desarrollar proyectos tecnológicos. Para esto se determinó tiempos cortos y de duración fija facilitando el cumplimiento de los objetivos planteados de forma parcial y frecuente. Este tipo de metodología ofrece una retroalimentación y pronta solución a los problemas presentados al instante, lo que permitió un control más riguroso de cada uno de los sprints planteados.

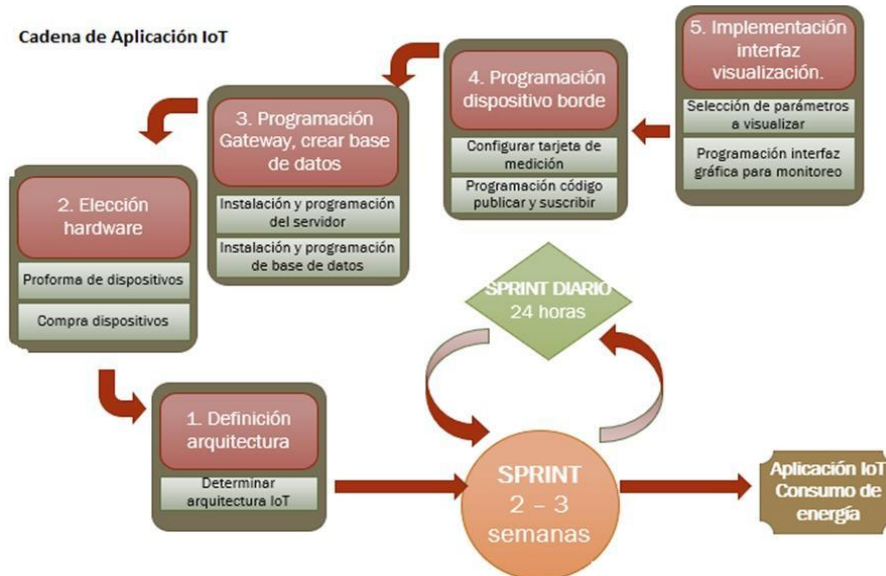
Ante lo mencionado, el proyecto se dividió en cinco sprints principales (ver Figura 27) que se ordenaron por su prioridad, estos se listan a continuación.

- Definición de la arquitectura a emplear.

- Elección del hardware, tomando en cuenta el criterio de bajo costo.
- Programación del *Gateway* y creación de base de datos.
- Programación del dispositivo de borde.
- Implementación de la interfaz de visualización.

Figura 27

Scrum de aplicación IoT



Como se puede observar en la Figura 27, cada etapa consta de varios subprocesos, además se especifica que los *sprints* tendrán una duración de 2 a 3 semanas y otros de 24 horas. El trabajo diario es importante, puesto que si surgen problemas deben corregirse al momento o como máximo al día siguiente, antes de la retroalimentación de refuerzo o compensación según sea el caso.

2.3 Validación de la propuesta

En esta sección se mostrará la validación de especialistas del prototipo IoT propuesto. Para esto el proyecto de investigación fue expuesto brevemente a cada especialista, lo que les permitió valorarla de acuerdo con los criterios planteados. Cabe mencionar que las reuniones con los especialistas se llevaron a cabo mediante la plataforma Zoom.

Es importante destacar que los especialistas elegidos cumplieron con la formación académica y experiencia laboral orientada al IoT. En la Tabla 4 se detalla el perfil de los tres especialistas que fueron elegidos.

Tabla 4*Información del perfil de especialistas*

N°	Nombres y Apellidos	Años de experiencia	Titulación Académica	Cargo
1	Germán Arévalo Bermeo	18	Doctor en Ingeniería	Profesor Universitario, Investigador, Consultor
2	Edgar Emanuel González Malla	9	Máster Universitario en Tecnologías, Sistemas y Redes de Comunicación	Docente/Investigador
3	Jorge Luis Paredes Carrillo	3	Máster Universitario en Industria 4.0 Ingeniero Electrónico en Control y Redes Industriales	Técnico Electrónico en UCEM S.A. Aspirante aprobado en programa de doctorado en Ingeniería Eléctrica de la EPN.

La valoración de criterios de los especialistas se detalla en la Tabla 5 y las valoraciones completas se hallan en el Anexo 4.

Tabla 5*Evaluación de criterios de los tres especialistas seleccionados*

EVALUACIÓN SEGÚN IMPORTANCIA Y REPRESENTATIVIDAD					
Especialista: Germán Arévalo Bermeo					
CRITERIOS	En Total desacuerdo	En Desacuerdo	Ni de Acuerdo Ni en desacuerdo	De Acuerdo	Totalmente de acuerdo
Impacto					X
Aplicabilidad					X
Conceptualización					X
Actualidad					X
Calidad Técnica					X
Factibilidad					X
Pertinencia					X
EVALUACIÓN SEGÚN IMPORTANCIA Y REPRESENTATIVIDAD					
Especialista: Edgar Emanuel González Malla					
CRITERIOS	En Total desacuerdo	En Desacuerdo	Ni de Acuerdo Ni en desacuerdo	De Acuerdo	Totalmente de acuerdo
Impacto					X
Aplicabilidad					X
Conceptualización					X
Actualidad					X
Calidad Técnica					X
Factibilidad					X
Pertinencia					X
EVALUACIÓN SEGÚN IMPORTANCIA Y REPRESENTATIVIDAD					
Especialista: Jorge Luis Paredes Carrillo					
CRITERIOS	En Total desacuerdo	En Desacuerdo	Ni de Acuerdo Ni en desacuerdo	De Acuerdo	Totalmente de acuerdo
Impacto					X
Aplicabilidad					X
Conceptualización					X
Actualidad					X
Calidad Técnica					X
Factibilidad					X
Pertinencia					X

2.4 Matriz de articulación de la propuesta

En la Tabla 6, se detalla el desarrollo de una matriz que sintetiza la articulación del producto realizado con los respectivos sustentos teóricos, metodológicos, estratégicos-técnicos y tecnológicos utilizados.

Tabla 6

Matriz de articulación

Ejes o partes principales del proyecto		Breve descripción de los resultados de cada parte	Sustento teórico que se aplicó en la construcción del proyecto	Metodologías, herramientas técnicas y tecnológicas que se emplearon
1	Definición de arquitectura IoT de cuatro niveles.	1.1. Determinación de software y de hardware en base criterios de interoperabilidad y de costo. 1.2. Indagación de proformas de dispositivos. 1.3. Compra de dispositivos.	Industria 4.0 Internet de las Cosas. Comunicaciones Inalámbricas. Software de código abierto. Sistemas embebidos.	-Revisión bibliográfica. -Sitios web de empresas desarrolladoras de hardware libre. -Sitios web oficiales de software libre.
2	Diseño y programación de arquitectura IoT	2.1. Diagrama de bloques para el registro de variables eléctricas. 2.2. Configuración de PZEM 004T 100A. 2.3 Instalación de Eclipse <i>Mosquitto</i> , <i>Node-RED</i> , <i>InfluxDB</i> y <i>Grafana</i> . 2.4. Programación de <i>Raspberry Pi</i> y <i>NodeMCU ESP8266</i> . 2.5. Creación de base de datos en <i>InfluxDB</i> . 2.6. Programación de panel de monitoreo en <i>Grafana</i> .	Programación de <i>NodeMCU</i> . Librerías de Arduino. Protocolo de comunicación MQTT. Programación en <i>Node-RED</i> . Código Flux. SDN (<i>Software Defined Networks</i>)	-Sistema operativo <i>Raspbian Bullseye</i> . -Librería <i>PZEM004Tv30.h</i> disponible en sitio web de Arduino. -Servidor Eclipse <i>Mosquitto</i> . -Servidor <i>Node-RED</i> . -Base de datos <i>InfluxDB</i> . -Servidor <i>Grafana</i> . - Aplicación <i>ZeroTier</i> .

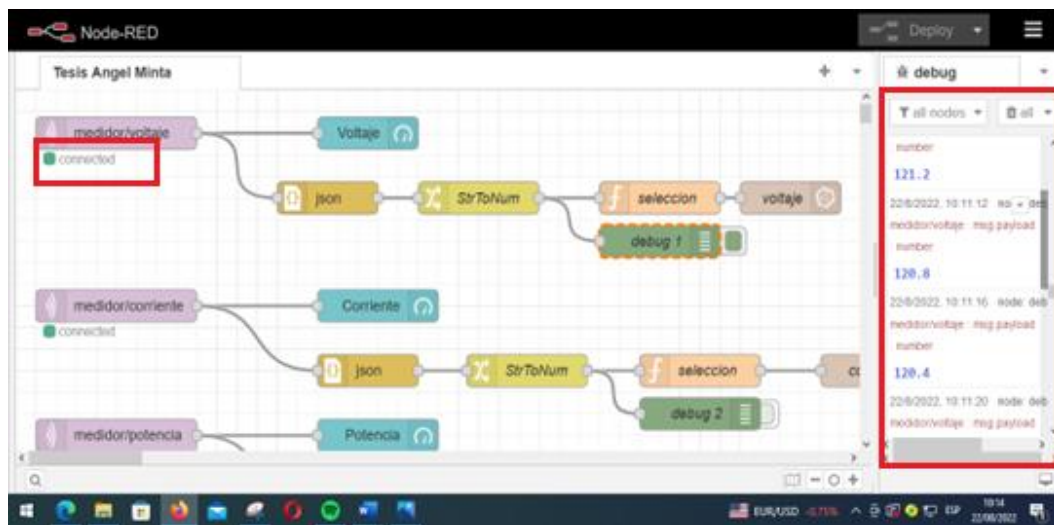
3	Implementación de aplicación IoT.	<p>3.1. Conexiones físicas de dispositivos eléctricos y electrónicos.</p> <p>3.2. Integración de software y hardware de la aplicación.</p> <p>3.3. Panel de monitoreo de corriente voltaje, frecuencia, potencia, factor de potencia, consumo, energía y costo.</p> <p>3.4 Base de datos variables eléctricas monitoreadas.</p>	<p>Electrónica de potencia.</p> <p>Electrónica digital.</p> <p>Instalación eléctrica residencial.</p> <p>Protocolo wifi.</p> <p>Dirección del Protocolo de Internet (IP).</p>	<p>-Metodología ágil scrum.</p> <p>-Manuales técnicos de los dispositivos.</p> <p>-Servicio de internet.</p> <p>-Programación de software empleados en el apartado 2.</p> <p>- Aplicación <i>ZeroTier</i>.</p> <p>-Excel, Drive.</p>
---	-----------------------------------	---	---	--

2.5 Análisis de resultados. Presentación y discusión.

La aplicación IoT asignada con el nombre de *Smart_Energy* fue implementada en una residencia de una familia promedio, en el sector Rumicruz, cantón Chambo. El prototipo desarrollado se instaló en la caja de control eléctrico de la residencia y mediante *Node-RED* se comprobó el establecimiento de la comunicación con el *topic medidor/voltaje* en el *payload* (ver Figura 28)

Figura 28

Comprobación de comunicación del dispositivo de borde con Mosquitto.



Seguidamente se efectuó el monitoreo en el dashboard de *Grafana* de los siguientes parámetros eléctricos: corriente, voltaje frecuencia, factor de potencia, potencia, energía y sobre todo consumo. Adicionalmente se creó el panel de costo estimado, se tomó como referencia el costo del kWh establecido por EERSA.

En la Figura 29 se observa el dashboard del voltaje con un valor de 118,9 V y de la Frecuencia con 59,90 Hz. Por otro lado, se visualizan gráficas estadísticas de la corriente, potencia y factor de potencia.

El dashboard de energía, consumo y costo se muestra en la Figura 30, donde la primera se halla representada mediante un gráfico. El segundo parámetro marca un total de 18,04 kWh que es consumo de energía, que es un valor acumulativo. Adicionalmente se creó el panel de costo estimado, se tomó como referencia el precio del kWh establecido por la EERSA.

Figura 29

Dashboard de voltaje, corriente, frecuencia, potencia y factor de potencia

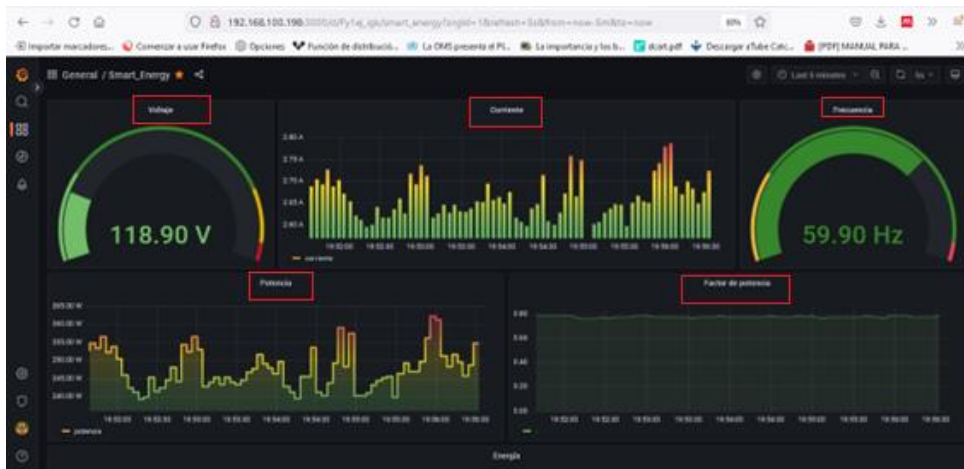


Figura 30

Dashboard de energía, consumo y costo



Con el *Smart_Meter* en funcionamiento se obtiene una amplia base de datos, pues las variables son monitoreadas diariamente cada cuatro o cinco segundos. A partir de la instalación del prototipo se obtuvo un gran volumen de datos.

Prueba 1: Voltaje

Para verificar que los valores obtenidos por el prototipo propuesto sean confiables, se realizó una comparativa con los registrados en el multímetro *FLUKE 323*. Para esto se eligió una muestra de 15 datos del monitoreo del día 18 de agosto del presente año. Estos se importaron de *Grafana* y se seleccionaron los obtenidos cada 30 segundos, escala de tiempo que sirvió para la medición con el multímetro (ver Figura 31).

Figura 31

Voltaje obtenido en *Smart_Energy* y en *FLUKE 323*



La Tabla 7 muestra el detalle de los datos de voltaje obtenidos de *Smart_Energy* y de *Fluke 323* en las marcas de tiempo determinadas. También se puede observar el porcentaje de error del conjunto de datos, en este caso del 0,27%, valor que es relativamente bajo demostrándose la confiabilidad de los datos obtenidos.

Tabla 7

Error porcentual de los valores de voltaje de *Smart_Energy* vs *Fluke 323*.

#	Fecha/Hora	Smart_Meter	FLUKE 323	Error Absoluto	Error Porcentual
		(V)	(V)		
1	2022-08-18 10:59:00	119,45	119,6	0,15	0,126%
2	2022-08-18 10:59:30	119,90	199,7	0,20	0,167%
3	2022-08-18 11:00:00	119,85	199,7	0,15	0,125%
4	2022-08-18 11:00:30	120,10	199,7	0,60	0,500%
5	2022-08-18 11:01:00	119,97	119,6	0,37	0,308%
6	2022-08-18 11:01:30	119,85	119,6	0,25	0,209%
7	2022-08-18 11:02:00	119,90	119,6	0,30	0,250%
8	2022-08-18 11:02:30	119,85	119,4	0,45	0,375%
9	2022-08-18 11:03:00	120,05	119,5	0,55	0,458%
10	2022-08-18 11:03:30	120,07	119,9	0,17	0,142%
11	2022-08-18 11:04:00	120,05	119,8	0,25	0,208%
12	2022-08-18 11:04:30	119,97	119,5	0,47	0,392%
13	2022-08-18 11:05:00	120,10	119,7	0,40	0,333%
14	2022-08-18 11:05:30	120,15	119,8	0,35	0,291%
15	2022-08-18 11:06:00	120,27	120,1	0,17	0,141%
Promedio				0,32	0,27%

A parte de *Grafana*, también se puede obtener datos de *InfluxDB*, si se precisará realizar análisis de datos de las variables monitoreadas.

Prueba 2: Consumo eléctrico

Se realizó una segunda prueba con valores de energía. Para esto los valores de consumo del medidor de energía fue comparado con los del prototipo *Smart_Meter*, el cual entró en funcionamiento el 11 de agosto a las 17:30 pm, con una lectura inicial de 0 kWh. Para esta prueba se tomó los datos de *InfluxDB* en formato CSV (*comma-separated values*) hasta el 18 de agosto, 17:00 pm. Los datos mencionados se detallan en la Tabla 8 con una marca de tiempo de una hora.

Del medidor de energía analógico, el 11 de agosto y el 18 de agosto a la hora establecida se registró una lectura inicial y final respectivamente, estos datos permitieron el cálculo del consumo total de energía para la prueba, de los cuales se halló su diferencia (Ver Tabla 9). En el Anexo 5, se muestran fotografías de los datos empleados.

Tabla 8

Valores de energía obtenidos de Smart_Meter

#	Fecha/Hora	Smart_Meter
		(kWh)
1	8/17/2022 3:00:03	18,874
2	8/17/2022 4:00:01	18,914
3	8/18/2022 5:00:02	18,968
4	8/18/2022 6:00:01	18,994
5	8/18/2022 7:00:02	19,103
6	8/18/2022 8:00:02	19,190
7	8/18/2022 9:00:01	19,279
8	8/18/2022 10:00:02	20,103
9	8/18/2022 11:00:02	20,342
10	8/18/2022 12:00:02	20,612
11	8/18/2022 13:00:02	20,622
12	8/18/2022 14:00:02	20,756
13	8/18/2022 15:00:02	20,902
14	8/18/2022 16:00:01	21,035
15	8/18/2022 17:00:02	21,172

El dato 15 detallado en la Tabla 8, se puede visualizar en la Figura 32, la cual corresponde a una gráfica que genera *Grafana* mostrando la precisión en el monitoreo de datos.

Figura 32

Gráfico de energía en el intervalo de tiempo de 03:00-17:00.



Tabla 9

Lectura inicial y final obtenidas del medidor analógico

Datos para prueba de energía	
Consumo inicial (kWh)	9364
Consumo final (kWh)	9385
Consumo durante prueba (kWh)	21

Finalmente, en la Tabla 10 se detalla la comparación de las lecturas obtenidas entre el dispositivo *Smart_Energy* y el medidor analógico de energía, en las fechas de prueba. Además, se observa un valor bajo de error porcentual (0,82%), demostrándose la confiabilidad en los datos.

Tabla 10

Error porcentual de valores de energía de Smart_Energy vs medidor analógico.

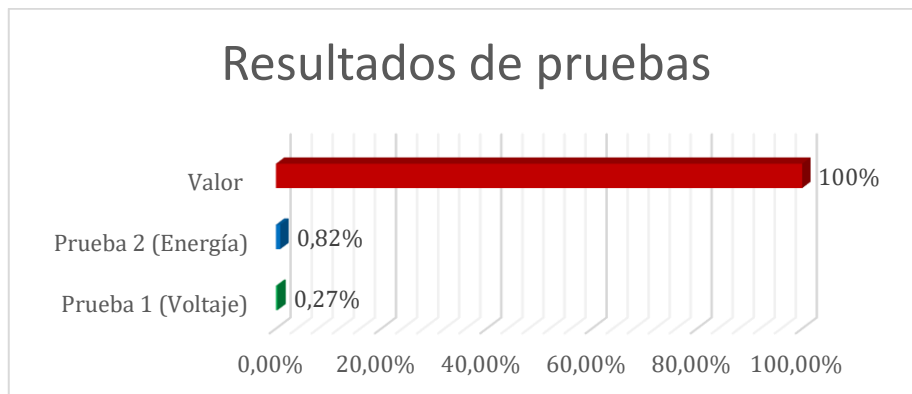
Medidor Eléctrico (kWh)	Smart_Energy (kWh)	Error Absoluto	Error Relativo
			(%)
21	21,172	0,172	0,82

Resultado de pruebas

En la Figura 33, se muestra los resultados de los errores porcentuales obtenidos en la prueba uno y dos, se observa que los errores no alcanzan ni el 1% por lo que son relativamente bajos. Esto significa que los valores obtenidos de *Smart_Energy* son confiables y por ende el funcionamiento del prototipo.

Figura 33

Resultados de prueba

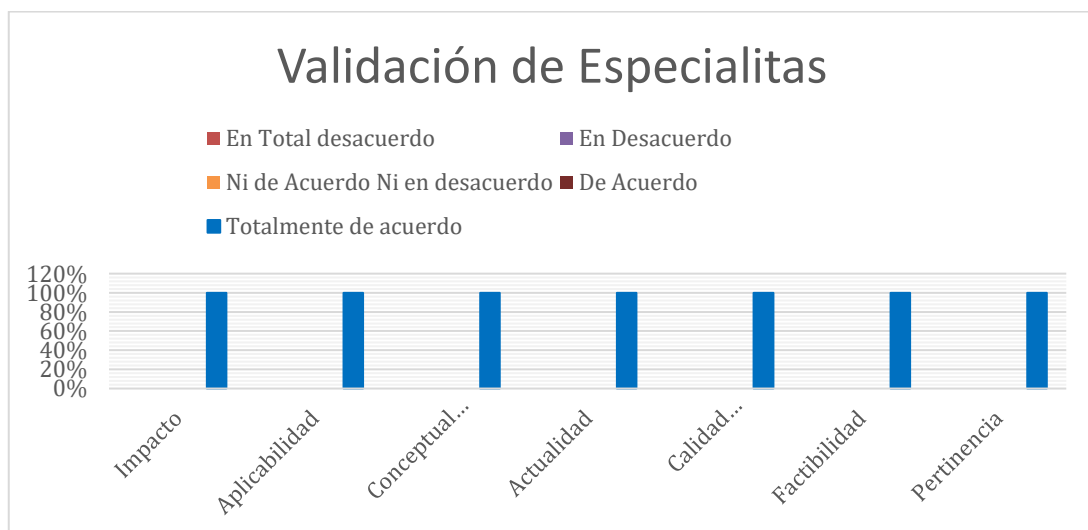


Resultados de validación

En la Figura 34, se presentan los resultados de los tres especialistas de la validación, se observa que todos los criterios fueron evaluados con el rango de totalmente de acuerdo. Esto significa que la aplicación IoT propuesta cumple con los objetivos propuestos.

Figura 34

Resultados de validación de especialistas



CONCLUSIONES

La aplicación IoT nombrada en este proyecto como Smart_Energy, se desarrolló utilizando la tecnología del Internet de las cosas, sistemas embebidos (principalmente Raspberry Pi), el uso de un protocolo flexible, liviano y escalable (MQTT), y sobre todo de software no licenciado y sin pago. El prototipo obtenido permitió realizar el monitoreo del corriente, voltaje, frecuencia, potencia, factor de potencia, energía, consumo eléctrico y costo de una residencia.

Los software utilizados en el prototipo Smart_Energy fueron definidos mediante una profunda indagación de aquellos que eran no licenciados y sin pago, los elegidos se instalaron en la Raspberry Pi 4 siendo estos Node-RED, InfluxDB y Grafana. Este último mediante un dashboard programado usando código flux, permitió visualizar los parámetros eléctricos a monitorear.

Se diseñó una arquitectura IoT constituida por cuatro niveles: percepción, conectividad, servicio y aplicación permitiendo relacionar los sistemas, servidor y software empleados en Smart_Energy con cada nivel o etapa. Percepción (Tarjeta PZEM-004T, Raspberry Pi 4 Model B, NodeMCU ESP8266), conectividad (Protocolo MQTT), servicio (Broker Mosquitto, InfluxDB) y de aplicación (Node-RED, Grafana); obteniendo una aplicación escalable e interoperable.

El prototipo Smart_Energy fue implementado en una residencia localizada en el cantón Chambo, se instaló librerías necesarias en Raspberry Pi para enviar datos a MQTT desde PZEM 004T que se conectó a NodeMCU ESP8266. Un Gateway implementado en el broker Mosquitto manejó una capa de abstracción (Node-RED e InfluxDB) que permitió una comunicación bidireccional entre aplicaciones en la nube. Se visualizó los datos de los principales parámetros eléctricos medidos en un dashboard desarrollado en Grafana. Además, se tomó en cuenta aspectos técnicos necesarios: conexiones físicas de la aplicación, caja de control de la vivienda, red de Wifi, entre otras.

Se monitoreó los valores del consumo energético de la residencia en tiempo real, y para verificar la información obtenida se realizó pruebas de errores porcentuales. En la prueba uno se comparó los datos de voltaje obtenidos de Grafana con los medidos con un multímetro, obteniéndose un error del 0,27%. En la segunda prueba los datos de consumo de Grafana se contrastaron con los marcados en el medidor analógico de la vivienda se obtuvo un error del 0,82%. Los porcentajes fueron relativamente bajos, demostrándose la confiabilidad de los datos de Smart_Energy.

RECOMENDACIONES

Con el rápido desarrollo de IoT y la computación en la nube, se recomienda temas de investigación abiertos tales como: ciberseguridad, control de acceso, confidencialidad de datos, ubicación privacidad, minería de datos para preservar la privacidad, el enorme volumen de complejidad computacional que todavía impide significativamente su amplia aplicación en usuarios con recursos limitados.

El uso de software libre en su versión no licenciado y sin pago representa un gran aliado para el desarrollo de aplicaciones IoT a nivel de prototipos. Para responder a necesidades en redes de producción a gran escala se recomienda emplear versiones de pago y evaluar su empleabilidad, misma que podría justificarse con las mejoras en la producción.

Se recomienda aplicar la arquitectura IoT desarrollada a procesos industriales, integrando la parte de control, y a su vez de la data obtenida realizar análisis de datos mediante técnicas de inteligencia artificial, permitiendo el nacimiento de los sistemas ciber-físicos.

Se debe tomar en cuenta la utilización de respaldos de energía UPS para protección de dispositivos electrónicos cuando un prototipo como el desarrollado en el presente trabajo, haya sido validado completamente. Para pruebas iniciales considerar el estado de instalaciones físicas existentes donde se implementará un prototipo.

Se recomienda el uso de Redes Definidas por Software, como ZeroTier puesto que combina la capacidad de una VPN y SD-WAN para interconectar a los usuarios de un determinado sistema desde cualquier parte del mundo y en cualquier dispositivo.

BIBLIOGRAFÍA

- Angelino, T., Cervalho, D., Gomes, F., y Fernández, L. (2022). Diseño IoT y validación de sistema de medida para generación fotovoltaica. *INGENIUS*, 9, 44-50.
<https://doi.org/10.17163/ings.n28.2022.04>
- ARCERNNR. (2022). *Panorama Eléctrico*, 10,1-6.
- Arduino. (28 de octubre 2019). *Arduino IoT Cloud: Support for ESP8266 and other third party boards*. Arduino.cc. Arduino.cc: <https://www.arduino.cc/>
- Asamblea Nacional República del Ecuador. (2019). *Ley Orgánica de eficiencia energética*. Editora Nacional.wwe.registroficial.gob.ec.
- Canales, M., Paucar, W., y Juipa, N. (2017). Método de investigación para ingenierías basado en la metodología de la investigación científica. *Revista Científica*, 7(4), 5-9.
<https://doi.org/https://revistas.unas.edu.pe/index.php/revia/article/view/172/0>
- Cobos, Y. (2021). *Diseño e implementación de una red virtual basada en Docker en un ambiente de redes definidas por software (SDN) utilizando Zerotier y Raspberry Pi*. [Tesis de Titulación, Universidad Politécnica Salesiana]. Repositorio Institucional- Universidad Politécnica Salesiana. <http://dspace.ups.edu.ec/handle/123456789/21982>
- Coloma, D., y Rodríguez, C. (2020). *Diseño e implementación de prototipo electrónico inalámbrico para el monitoreo de consumo de agua usando tecnología de bajo costo en los hogares con acceso a internet de la parroquia el Laurel*. [Tesis de Titulación, Universidad de Guayaquil]. Repositorio Institucional- Universidad Guayaquil. <http://repositorio.ug.edu.ec/handle/redug/49481>
- Cutus, B., y Recalde, M. (2022). *Monitorización y control de seguridad en hogares vulnerables en el sector de Chillogallo desde dispositivos móviles multiplataforma, haciendo uso de Node-red y Raspberry Pi4*. [Tesis de Titulación, Universidad Politécnica Salesiana]. Repositorio Institucional- Universidad Politécnica Salesiana. <http://dspace.ups.edu.ec/handle/123456789/22220>
- Echeverri, J., y Patiño, L. (2018). *Sistema Inteligente de Monitoreo de Consumo Eléctrico*. [Tesis de Titulación, Universidad Tecnológica de Pereira]. Repositorio Institucional- Universidad Tecnológica de Pereira. <https://hdl.handle.net/11059/9426>
- Fernández, C. (2021). *Aula virtual como refuerzo académico para la enseñanza de Química en primero de bachillerato*. [Tesis de Maestría, Universidad Tecnológica Israel]. Repositorio Institucional- Universidad Tecnológica Israel. <http://repositorio.uisrael.edu.ec/handle/47000/2723>
- Gil Inchaurrea, G. (2018). *Comparativa teórica y práctica de middlewares MQTT*. [Tesis de Maestría, Universidad del País Vasco]. Archivo digital docencia investigación-Universidad del País Vasco. <http://hdl.handle.net/10810/29792>
- Gobierno del Ecuador. (2019). *Ley orgánica de eficiencia energética*. Quito: Registro Oficial
- HiveMQ. (2020). *MQTT & MQTT 5 Essentials: A comprehensive overview of MQTT facts and features*. Landshut: HiveMQ GmbH. <https://www.hivemq.com/blog/announcing-mqtt-ebook/>
- Jimenez, P., y Cabrera, J. (2020). Sistema de monitoreo remoto del consumo energético para hogares en la ciudad de Cuenca, basado en principios de IoT y servicios en la nube. *Polo del conocimiento: Revista científico-profesional*, 5, 443-458. 10.23857/pc.v5i1.1949.

- Lozada, C. (2022). *Desarrollo de un sistema basado en Internet Industrial de las Cosas para el monitoreo y control de un banco de pruebas de intercambiador de calor*. [Tesis de Maestría, Universidad Politécnica Salesiana]. Repositorio Institucional- Universidad Politécnica Salesiana. <https://dspace.ups.edu.ec/bitstream/123456789/21950/1/UPS-GT003631.pdf>
- Manuales +. (2022). *Módulo de comunicación de CA PZEM-004T V3.0 Manual de usuario*. [Archivo PDF] <https://manuals.plus/es/InnovatorsGuru/m%C3%B3dulo-de-comunicaci%C3%B3n-de-ca-pzem-004t-v3-0-manual#axzz7bqXyX7AH>
- Mosquitto. (31 de agosto de 2021). *Eclipse Mosquitto Un corredor MQTT de código abierto*. https://mosquitto-org.translate.google/?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es-419&_x_tr_pto=sc
- Ordoñez, L. (2019). *Desarrollo de un sistema web de gestión de fichas médicas para la organización "Salud Integral para Todos", bajo la metodología SCRUM*. [Tesis de Maestría, Universidad Tecnológica Israel]. Repositorio Institucional- Universidad Tecnológica Israel. <http://repositorio.uisrael.edu.ec/handle/47000/2045>
- Oxer, J. (10 de diciembre de 2020). *Datalogging with MQTT, Node-RED, Influxdb, and Grafana*. <https://www.superhouse.tv/41-datalogging-with-mqtt-node-red-influxdb-and-grafana/>
- Paredes, J. (2020). *Sistema de control y monitoreo con sistemas embebidos para entornos industriales*. [Tesis de Maestría no publicada]. Universidad Internacional de la Rioja.
- Pérez, K. (2019). *Estudio y análisis del protocolo de mensajería avanzado en el internet de las cosas para aplicación en el campo de la domótica*. [Tesis de Titulación, Universidad Católica de Santiago de Guayaquil]. Repositorio Institucional- Universidad Católica de Santiago de Guayaquil. <http://repositorio.ucsg.edu.ec/handle/3317/13368>
- Pons, S. (2021). *Sistema de captura y monitorización de variables fisiológicas basado en tecnologías IoT y plataformas de bajo coste*. [Tesis de Maestría, Universidad del País Vasco]. Archivo digital docencia investigación-Universidad del País Vasco. <http://hdl.handle.net/10810/54030>
- Raspberry Pi Foundation. (2019). *Raspberry Pi 4*. <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/>
- Rodríguez, D., Rodríguez, J., Zúñiga, M., y Solano, L. (2017). Un enfoque para la integración de dispositivos IoT en el desarrollo de SIG en la nube. *Maskana*, 8(1), 57–65. <https://publicaciones.ucuenca.edu.ec/ojs/index.php/maskana/article/view/1966>
- Romero, E. (2017). *Implementación de un prototipo de medidor de energía residencial considerando las pérdidas no técnicas por hurto*. [Tesis de Titulación, Escuela Superior Politécnica de Chimborazo]. Repositorio Institucional- Escuela Superior Politécnica de Chimborazo. <http://dspace.esoch.edu.ec/handle/123456789/8956>
- Torres, Á., Pisco, J., Pérez, R., y Vera, I. (2020). Monitoreo en tiempo real del consumo de energía eléctrica residencial que permita su apropiada gestión. *Revista Universidad y Sociedad*, 12(2), 218-222. https://doi.org/http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2218-36202020000200218&lng=es&tlng=es.
- Uslenghi, J. (2019). *Diseño de aplicación IoT para la supervisión energética de una instalación frigorífica comercial*. [Tesis de Maestría, Universidad Politécnica de Valencia]. Repositorio Universidad Politécnica de Valencia. <http://hdl.handle.net/10251/124858>

- Viáfara, L., Quintero, J., Montero, V., Garzón, J., & López Andrés. (2021). *Industria 4.0 Gestión del conocimiento*. Editorial Universidad Icesi.
<http://dspace.esPOCH.edu.ec/handle/123456789/8956>
- Yacchirema, D. (2019). *Arquitectura de interoperabilidad de dispositivos físicos para el internet de las cosas (IOT)*. [Tesis de Doctorado, Universidad Politécnica de Valencia]. Repositorio Universidad Politécnica de Valencia. <http://hdl.handle.net/10251/129858>
- Yerovi, F. (2022). *Desarrollo de un Entorno Vital de Aprendizaje para fortalecer la enseñanza-aprendizaje del inglés para cuarto grado EGB*. [Tesis de Maestría, Universidad Tecnológica Israel]. Repositorio Institucional- Universidad Tecnológica Israel.
<http://repositorio.uisrael.edu.ec/handle/47000/2998>

ANEXOS

ANEXO 1

CÓDIGO NODEMCU ESP8266

```
//Tesis Maestria Angel Minta
//Energy Smart-Meter

#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <PZEM004Tv30.h>
#include <ESP8266WiFiMulti.h>
#include <SoftwareSerial.h>

ESP8266WiFiMulti WiFiMulti;

//Configuración de los parámetros de red

const char* ssid="Scaum-FastSystems";           //Nombre de la red WiFi
const char* password="9733sandra";           //Constraseña de la red WiFi
const char* mqtt_server = "192.168.100.198"; //Dirección IP del bróker MQTT (Raspberry Pi 4)//146

//Configuración Smart-Meter

PZEM004Tv30 pzem(4, 5); //Pines TX y RX conectados a la NodeMCU D1,D2

char svoltaje[20];
char scorriente[20];
char spotencia[20];
char senergia[20];
char sfrecuencia[20];
char sfactorpotencia[20];

//Configuración Cliente MQTT

unsigned long lastMsg = 0;

WiFiClient espClient;
PubSubClient client(espClient);

//Función para la conexión a la red

void setup_wifi() {
  delay(10);

  Serial.println();
  Serial.print("Conectado dispositivo a la red WiFi: ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);
```

```

while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}

Serial.println("");
Serial.println("Dispositivo conectado");
Serial.println("Dirección IP del dispositivo");
Serial.println(WiFi.localIP());
Serial.println("");
}

//Función para reconectar en caso de pérdida de señal

void reconnect() {
while(!client.connected()) {
    Serial.println("Intentando conexión MQTT");
    if (client.connect("ESP8266Client")) {
        Serial.println("Dispositivo conectado");

        //Publicar temas MQTT del medidor

        client.publish("medidor/voltaje", "Datos de voltaje: ");
        client.publish("medidor/corriente", "Datos de corriente: ");
        client.publish("medidor/potencia", "Datos de potencia: ");
        client.publish("medidor/energia", "Datos de energia: ");
        client.publish("medidor/frecuencia", "Datos de frecuencia: ");
        client.publish("medidor/factorpotencia", "Datos de factor de potencia: ");

        //client.subscribe("ucem/led");
    }
    else {
        Serial.print("Conexión fallida, código: ");
        Serial.print(client.state());
        Serial.println(" Intentando conectar en 5 segundos");
        delay(5000);
    }
}
}

void medidor(){
float voltage = pzem.voltage();
float current = pzem.current();
float power = pzem.power();
float energy = pzem.energy();
float frequency = pzem.frequency();
float fp = pzem.pf();

Serial.println("voltaje");
Serial.println(voltage);
dtostrf(voltage, 10, 4, svoltaje);

```

```

client.publish("medidor/voltaje", svoltaje);

Serial.println("corriente");
Serial.println(current);
dtostrf(current, 10, 4, scorriente);
client.publish("medidor/corriente", scorriente);

Serial.println("potencia");
Serial.println(power);
dtostrf(power, 10, 4, spotencia);
client.publish("medidor/potencia", spotencia);

Serial.println("energia");
Serial.println(energy);
dtostrf(energy, 10, 4, senergia);
client.publish("medidor/energia", senergia);

Serial.println("frecuencia");
Serial.println(frequency);
dtostrf(frequency, 10, 4, sfrecuencia);
client.publish("medidor/frecuencia", sfrecuencia);

Serial.println("factor de potencia");
Serial.println(fp);
dtostrf(fp, 10, 4, sfactorpotencia);
client.publish("medidor/factorpotencia", sfactorpotencia);

delay(4000);
}

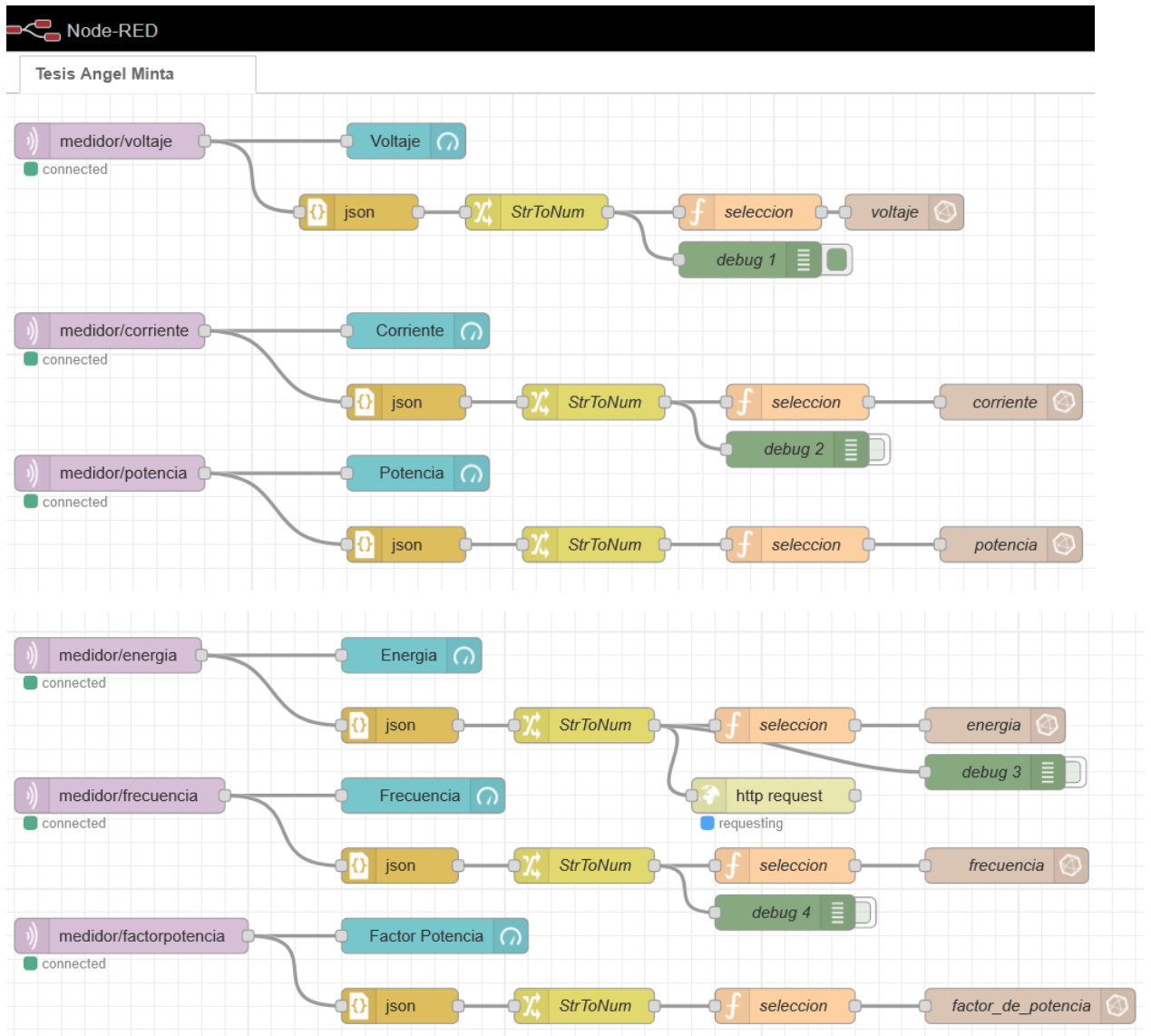
void setup() {
  Serial.begin(115200);          //velocidad del puerto serial
  setup_wifi();                //iniciar funcion para conectar red wifi
  client.setServer(mqtt_server, 1883); //puerto 8883 MQTT para propocionar conexión cifrada
}

void loop() {
  medidor();
  if(!client.connected()) {
    reconnect();
  }
  client.loop();
}

```

ANEXO 2

PROGRAMACION COMPLETA EN NODE-RED



ANEXO 3

CÓDIGO FLUX DE PARAMETROS ELÉCTRICOS

Voltaje	<pre> from(bucket: "Smart_Meter") > range(start: v.timeRangeStart, stop: v.timeRangeStop) > filter(fn: (r) => r["_measurement"] == "voltaje") > filter(fn: (r) => r["_field"] == "voltaje") > aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false) > yield(name: "last") </pre>
Corriente	<pre> from(bucket: "Smart_Meter") > range(start: v.timeRangeStart, stop: v.timeRangeStop) > filter(fn: (r) => r["_measurement"] == "corriente") > filter(fn: (r) => r["_field"] == "corriente") > aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false) > yield(name: "last") </pre>
Frecuencia	<pre> from(bucket: "Smart_Meter") > range(start: v.timeRangeStart, stop: v.timeRangeStop) > filter(fn: (r) => r["_measurement"] == "frecuencia") > filter(fn: (r) => r["_field"] == "frecuencia") > aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false) > yield(name: "last") </pre>
Potencia	<pre> from(bucket: "Smart_Meter") > range(start: v.timeRangeStart, stop: v.timeRangeStop) > filter(fn: (r) => r["_measurement"] == "potencia") > filter(fn: (r) => r["_field"] == "potencia") > aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false) > yield(name: "last") </pre>
Factor de potencia	<pre> from(bucket: "Smart_Meter") > range(start: v.timeRangeStart, stop: v.timeRangeStop) > filter(fn: (r) => r["_measurement"] == "factor_potencia") > filter(fn: (r) => r["_field"] == "factor_potencia") > aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false) > yield(name: "last") </pre>
Energía consumida	<pre> from(bucket: "Smart_Meter") > range(start: v.timeRangeStart, stop: v.timeRangeStop) > filter(fn: (r) => r["_measurement"] == "energia") > filter(fn: (r) => r["_field"] == "energia") > aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false) > yield(name: "last") </pre>

ANEXO 4

INSTRUMENTOS DE VALIDACIÓN A ESPECIALISTAS



UNIVERSIDAD TECNOLÓGICA DE ISRAEL

ESCUELA DE POSGRADOS "ESPOG"

MAESTRÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN

Resolución: RPC-SO-09-No.265-2021

Instrumento para la validación de la propuesta mediante criterio de especialista.

Objetivo general de la investigación: Desarrollar una aplicación IoT para el monitoreo de consumo eléctrico residencial utilizando software libre.

Datos del Especialista

Nombre y Apellidos	Años de experiencia	Titulación Académica	Cargo
Edgar Emanuel González Malla	Nueve	Master Universitario en Tecnologías, Sistemas y Redes de Comunicación	Docente/Investigador

1. Valore cada criterio propuesto colocando una X según corresponda.

CRITERIOS	EVALUACION SEGUN IMPORTANCIA Y REPRESENTATIVIDAD				
	En Total desacuerdo	En Desacuerdo	Ni de Acuerdo Ni en desacuerdo	De Acuerdo	Totalmente de acuerdo
Impacto					X
Aplicabilidad					X
Conceptualización					X
Actualidad					X
Calidad Técnica					X
Factibilidad					X
Pertinencia					X

2. Desde su punto de vista, me interesaría conocer sus observaciones y/o recomendaciones.

El sistema de monitoreo propuesto en este trabajo se puede aplicar a distintos escenarios. Es decir, a pesar de que está orientado a medición de consumo eléctrico, se aplica a diferentes servicios del IoT. Ya que la idea es comercializar el prototipo, recomiendo que se defina un costo adecuado para usuarios finales.

FIRMA



**Universidad
Israel**

UNIVERSIDAD TECNOLÓGICA DE ISRAEL

ESCUELA DE POSGRADOS “ESPOG”

MAESTRÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN

Resolución: RPC-SO-09-No.265-2021

Instrumento para la validación de la propuesta mediante criterio de especialista.

Objetivo general de la investigación: Desarrollar una aplicación IoT para el monitoreo de consumo eléctrico residencial utilizando software libre.

Datos del Especialista

Nombres y Apellidos	Años de experiencia	Titulación Académica	Cargo
Germán Arévalo Bermeo	18	Doctor en Ingeniería	Profesor Universitario, Investigador, Consultor

1. Valore cada criterio propuesto colocando una X según corresponda.

CRITERIOS	EVALUACION SEGUN IMPORTANCIA Y REPRESENTATIVIDAD				
	En Total desacuerdo	En Desacuerdo	Ni de Acuerdo Ni en desacuerdo	De Acuerdo	Totalmente de acuerdo
Impacto					X
Aplicabilidad					X
Conceptualización					X
Actualidad					X
Calidad Técnica					X
Factibilidad					X
Pertinencia					X

2. Desde su punto de vista, me interesaría conocer sus observaciones y/o recomendaciones.

Se trata de un proyecto muy interesante y de aplicación actual. Si bien como proto tipo es suficiente el uso de software no licenciado y sin pago, solamente para contextualizar las ventajas y necesidades reales en redes de producción, sería interesante mencionar que soluciones se pueden usar para implementaciones a gran escala.


FIRMA



UNIVERSIDAD TECNOLÓGICA DE ISRAEL

ESCUELA DE POSGRADOS “ESPOG”

MAESTRÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN

Resolución: RPC-SO-09-No.265-2021

Instrumento para la validación de la propuesta mediante criterio de especialista.

Objetivo general de la investigación: Desarrollar una aplicación IoT para el monitoreo de consumo eléctrico residencial utilizando software libre.

Datos del Especialista

Nombres y Apellidos	Años de experiencia	Titulación Académica	Cargo
Jorge Luis Paredes Carrillo	3 años	Máster universitario en Industria 4.0 Ingeniero electrónico en control y redes industriales	Técnico Electrónico en UCEM S.A. Aspirante aprobado en el programa de doctorado en ingeniería eléctrica de la EPN.

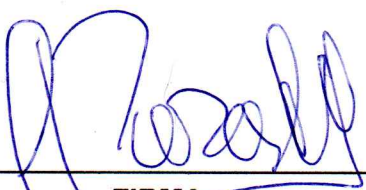
1. Valore cada criterio propuesto colocando una X según corresponda.

CRITERIOS	EVALUACION SEGUN IMPORTANCIA Y REPRESENTATIVIDAD				
	En Total desacuerdo	En Desacuerdo	Ni de Acuerdo Ni en desacuerdo	De Acuerdo	Totalmente de acuerdo
Impacto					X
Aplicabilidad					X
Conceptualización					X
Actualidad					X
Calidad Técnica					X
Factibilidad					X
Pertinencia					X

2. Desde su punto de vista, me interesaría conocer sus observaciones y/o recomendaciones.

Actualmente en Ecuador términos utilizados como Industria 4.0, Internet de las Cosas o Software Libre son pocos conocidos en varios sectores productivos. Desarrollar una investigación como la presentada constituye un gran aporte a diferentes campos como pueden ser el académico, el industrial o el social. Considero que la aplicación desarrollada constituye una gran herramienta en el campo educativo ya que su impacto, aplicabilidad y aspectos técnicos hacen comprender de una manera muy sencilla los términos antes

mencionados, además, de que constituye en un ejemplo concreto de la aplicación de conceptos inmersos en la Industria 4.0. En el campo industrial y social constituye un gran avance ya que automatiza varios procesos que en la actualidad son completamente manuales, por lo que la mano de obra puede utilizarse en el desarrollo de nuevos proyectos. Recomiendo difundir esta investigación hacia universidades y empresas que estén interesadas en el desarrollo de este tipo de aplicaciones. También, es importante seguir mejorando a futuro la aplicación utilizando otros habilitadores de la Industria 4.0 como por ejemplo la inteligencia artificial.



FIRMA

ANEXO 5

DATOS EMPLEADOS PARA LA PRUEBA DE CONSUMO ELECTRICO

