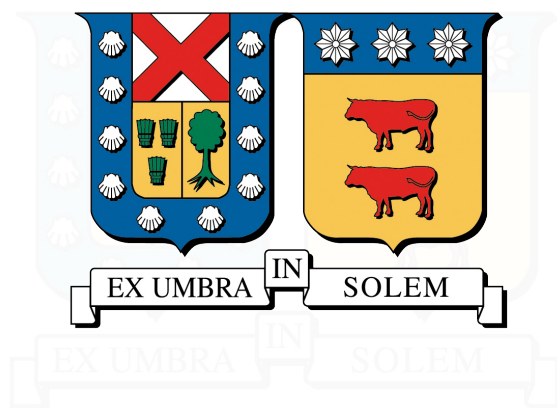


UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE ELECTRÓNICA
VALPARAISO - CHILE

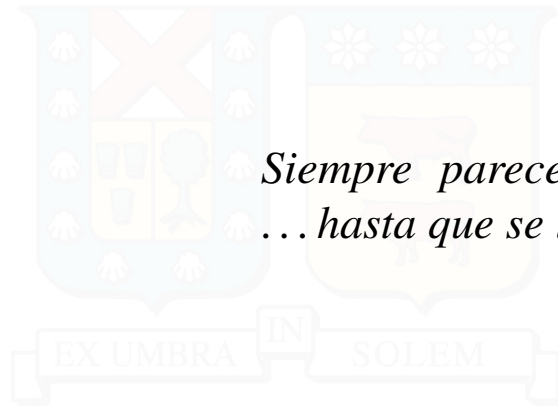


**DISEÑO E IMPLEMENTACIÓN DE UNA PLATAFORMA IOT PARA
MONITOREO DE CARGA EN HOGARES INTELIGENTES**

NICOLÁS CALDERÓN

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL TELEMÁTICO

PROFESOR GUÍA : MOHAMED ABDELHAMID
PROFESOR CORREFERENTE : JOSÉ MANUEL MARTINEZ



*Siempre parece imposible
...hasta que se logra.*

AGRADECIMIENTOS

Agradezco a quienes fueron parte de este camino. Seres queridos, familia y amigos que me acompañaron durante cada paso de esta etapa. También a mi profesor guía, Mohamed, por su paciencia, motivación y apoyo entregado durante todo el período de la memoria, y a Sebastián, compañero de memoria, por su compañerismo, complicidad, dedicación y gran trabajo que hicimos en conjunto. Su ayuda fue lo que me permitió estar donde hoy estoy. Además me gustaría agradecer a la Agencia Nacional de Investigación y Desarrollo (ANID), Proyecto Fondecyt de Iniciación en Investigación 2020, Project ID 11200178, por el apoyo a esta investigación.

Agradezco a mi madre, que nunca dudo de mí y siempre confió en que yo podría lograrlo. Y por último, agradezco especialmente a mi polola *Ela*, por apoyarme en los momentos difíciles, comprenderme y ser mi contención en todo momento. Sin ella nada de esto sería posible. No importa lo fuerte que seamos, todos necesitamos un abrazo.

RESUMEN EJECUTIVO

El consumo de energía eléctrica está aumentando de manera significativa a lo largo de todo el mundo, llegando a estimarse que en un par de años las distribuidoras de este preciado recurso, no sean capaces de dar abasto para solventar la necesidad de todos los usuarios.

Con el fin de poder concientizar a los usuarios de su consumo energético es que nace el desafío de desarrollar e implementar una plataforma IoT para el monitoreo de los electrodomésticos del hogar, la que permite a los usuarios de ella observar su consumo en tiempo real. Este trabajo de título se ve alimentado por datos provenientes de electrodomésticos reales, con ayuda de un intermediario entre la toma de corriente y el dispositivo objetivo. Estos intermediario son los Smart Plugs, los que envían continuamente información hacia un servidor en la nube para ser accedida por la plataforma y ser desplegada al usuario en distintos intervalos de tiempo, los que pueden ser seleccionados por él.

La idea de la plataforma propuesta es que los usuarios puedan tener su propia cuenta dentro de ella, donde pueden ver el consumo de sus dispositivos organizados por habitaciones. Además, cada usuario tiene la posibilidad de adquirir el número necesario de Smart Plugs y agregarlos o desagregarlos dentro de la plataforma por medio de un código asociado al dispositivo físico.

En este documento se explora el desarrollo y diseño de las dos primeras capas arquitectura IoT del desafío, las que se encargan la toma de datos y la comunicación de ellos. Finalmente, se muestra el resultado obtenido y su comparación con el resultado esperado junto con una serie de recomendaciones futuras para mejorar el desafío.

keywords: HEMS (Home Energy Management System), ILM (Intrusive Load Monitoring), IoT platform, SLN (Smart Load Node)

ABSTRACT

The consumption of electricity is increasing significantly throughout the world, and it is estimated that in a couple of years the distribution power company will not be able to meet the demand of all users.

In order to make users aware of their energy consumption, the challenge of developing and implementing an IoT platform for monitoring household appliances, which allows users to observe their consumption in real time, was born. This thesis work is fed by data coming from real household appliances, with the help of an intermediary between the power outlet and the target device. These intermediaries are the smart plugs, which continuously send information to a server in the cloud to be accessed by the platform and displayed to the user at different time intervals, which can be selected by the user.

The idea of the proposed platform is that users can have their own account within it, where they can see the consumption of their devices inside the home. In addition, each user has the possibility to acquire the required number of smart plugs and add or remove them within the platform by means of a code associated to the physical device.

This work explores the development and design of the first two IoT architecture layers of the challenge, which are responsible for data collection and communication. Finally, the result obtained and its comparison with the expected result is shown together with a series of future recommendations to improve the challenge.

keywords: HEMS (Home Energy Management System), ILM (Intrusive Load Monitoring), IoT platform, SLN (Smart Load Node)

Glosario

- **Access point:** Aparato que interconecta dispositivos de comunicación inalámbrica para formar una red inalámbrica.
- **API:** Conjunto de funciones y procedimientos que permiten acceder a las características o datos de otro sistema informático.
- **Backend:** Parte de un sistema informático a la que no accede directamente el usuario, normalmente responsable de almacenar y manipular datos.
- **Broker MQTT:** Servidor que acepta mensajes publicados por clientes y los difunde entre los clientes suscritos en un modelo de comunicación Pub-Sub
- **Chipset:** Conjunto de componentes electrónicos que están integrados en el procesador de un dispositivo electrónico
- **Domoticz:** Sistema de automatización del hogar que le permite controlar y configurar varios dispositivos como: luces, interruptores y sensores. Además, puede enviar notificaciones/alertas a cualquier dispositivo móvil
- **Endpoint:** URL a través del que se accede a la API.
- **ESP8266:** Microchip Wi-Fi de bajo coste, con software de red TCP/IP incorporado, y capacidad de microcontrolador.
- **Firmware:** Clase específica de software informático que proporciona el control de bajo nivel para el hardware específico de un dispositivo.
- **Frontend:** Parte de un sistema informático con la que el usuario interactúa directamente.
- **Gateway:** Dispositivo utilizado para conectar dos redes diferentes, especialmente una conexión a Internet.
- **HEMS:** Plataforma tecnológica compuesta por hardware y software que permite al usuario supervisar el uso y la producción de energía y controlar y/o automatizar manualmente el uso de la energía en un hogar.

-
- **Home Assistant:** Software gratuito y de código abierto para la automatización del hogar que está diseñado para ser el sistema de control central para dispositivos domésticos inteligentes con enfoque en el control local y la privacidad.
 - **IoT:** Interconexión a través de Internet de dispositivos informáticos integrados en objetos cotidianos, lo que les permite enviar y recibir datos.
 - **JSON:** Estándar de formato de archivos e intercambio de información
 - **Machine Learning:** Uso y desarrollo de sistemas informáticos capaces de aprender y adaptarse sin seguir instrucciones explícitas, mediante el uso de algoritmos y modelos estadísticos para analizar y sacar conclusiones de los patrones de los datos.
 - **Microgrids:** Pequeña red de usuarios de electricidad con una fuente de suministro local que suele estar conectada a una red nacional centralizada pero que es capaz de funcionar de forma independiente.
 - **Node.js:** Entorno de ejecución de JavaScript de código abierto y multiplataforma.
 - **Open Source:** Código abierto creado con una licencia para el uso público.
 - **Raspberry Pi:** Ordenador de bajo costo de tamaño compacto con la habilidad de interactuar con el mundo exterior por medio de módulos.
 - **SLN:** Nodo que permite el monitoreo del consumo de electrodomésticos además de agregar una capa de inteligencia a los mismos sin la necesidad de modificarlos ni al tomacorriente,
 - **Smart Grids:** Red de suministro de electricidad que utiliza la tecnología de las comunicaciones digitales para detectar y reaccionar a los cambios locales en el uso.
 - **Smart Homes:** Casa equipada con iluminación, calefacción y dispositivos electrónicos que pueden controlarse a distancia mediante un smartphone o un ordenador.
 - **Smart Plug:** Enchufe inteligente que va colocado entre los cables de alimentación de un dispositivo y las tomas de corriente.
 - **ThingSpeak:** Servicio de plataforma de análisis de IoT que permite agregar, visualizar y analizar flujos de datos en directo en la nube. Además permite enviar datos a

ThingSpeak desde sus dispositivos, crear una visualización instantánea de los datos en directo y enviar alertas.

- **Uptime:** Tiempo durante el cual una máquina, especialmente un ordenador, está en funcionamiento.
- **Telemetría:** Proceso de registro y transmisión de las lecturas de un instrumento, en este caso de los smart plugs.
- **TIC:** Tecnologías de la Información y la Comunicación son los recursos y herramientas que se utilizan para el proceso, administración y distribución de la información a través de elementos tecnológicos.
- **WAN:** Red de área amplia, o WAN, es una red de computadoras que une varias redes locales, aunque sus miembros no estén todos en una misma ubicación física.

Índice de Contenidos

1. Introducción	1
1.1. ¿Cómo son los HEMS actualmente?	1
1.2. Equipo de desarrollo	2
1.3. ¿Cuál es nuestro desafío?	2
1.4. Acercamiento a la solución	2
1.5. Objetivo General	4
1.6. Objetivos Específicos	4
1.7. Estructura del documento	5
2. Marco Teórico	6
2.1. ¿Qué son las Smart Grids?	6
2.2. Internet de las cosas (IoT)	7
2.3. Internet de la Energía (IoE)	8
2.4. Monitoreo de carga	10
2.5. Arquitectura de Smart Home	11
2.6. Funciones de Smart Home	12
2.6.1. Monitoreo	13
2.6.2. Registro	13
2.6.3. Control	13
2.6.4. Gestión	13
2.6.5. Alarma	14
2.7. Diseño de Hogar	14
3. Estado del arte	16
4. Arquitectura IoT de la Plataforma	29
4.1. Capa Física	29
4.2. Capa de Comunicación	30
4.3. Capa de Middleware	31
4.4. Capa de Aplicación	31
4.5. Requisitos del Sistema	32
4.5.1. Requisitos Funcionales	32
4.5.2. Requisitos no Funcionales	33
4.5.3. Requisitos de Interfaces	34
4.6. Propuesta de Solución	36

4.6.1.	¿Como afecta lo estudiado en el estado del arte en nuestra solución?	37
4.6.2.	Arquitectura del Sistema	38
5.	Hardware	40
5.1.	Capa Física	40
5.1.1.	Smart Plug	40
5.1.1.1.	Shelly Plug	41
5.1.1.2.	Shelly Plug S	42
5.1.1.3.	TP-Link HS110	42
5.1.1.4.	Develco Smart Plug Mini	44
5.1.1.5.	Sonoff Pow R2	45
5.1.2.	Firmware	46
5.1.2.1.	Tasmota	47
5.1.2.2.	ESPurna	48
5.1.2.3.	ESPEasy	48
5.1.2.4.	ESPHome	48
5.1.3.	Gateway	49
5.1.3.1.	Jetson Nano	50
5.1.3.2.	Raspberry Pi	51
5.1.3.3.	NodeMCU	51
5.2.	Capa de Comunicación	53
5.2.1.	Tecnologías	53
5.2.1.1.	Wi-Fi	54
5.2.1.2.	Zigbee	54
5.2.1.3.	Z-Wave	55
5.2.1.4.	BLE	55
5.2.1.5.	LoRaWAN	55
5.2.2.	Protocolos	56
5.2.2.1.	HTTP - Hyper Text Transfer Protocol	56
5.2.2.2.	CoAP - Constrained Application Protocol	57
5.2.2.3.	MQTT	57
5.2.3.	Modelo de Comunicación	59
5.2.3.1.	Request-Response	60
5.2.3.2.	Publisher-Subscriber	60
6.	Implementación	63
6.1.	Arquitectura del Sistema	63
6.2.	Configuración de Hardware	66
6.2.1.	Smart Plugs	66
6.2.2.	Gateway	71
7.	Resultados	74
7.1.	Firmware	74
7.2.	Comunicación	75
7.3.	Bases de datos	75
7.4.	Verificación y Validación	76
7.5.	Plataforma Web	78

8. Conclusión	81
Bibliografía	83
ANEXO	87



Índice de Tablas

3.1. Métricas recolectadas en cada configuración	24
3.2. Resumen de estado del arte	27
4.1. Eventos externos.	34
4.2. Respuestas del sistema.	36
5.1. Características Shelly Plug.	41
5.2. Características Shelly Plug.	42
5.3. Características TP-Link HS110.	43
5.4. Características Develco Smart Plug Mini.	44
5.5. Características Sonoff Pow R2.	45
5.6. Estudio de mercado Smart Plugs.	46
5.7. Comparación de Firmwares.	49
5.8. Características Jetson Nano.	50
5.9. Características Raspberry Pi.	51
5.10. Características NodeMCU.	52
5.11. Comparación de Gateways.	52
5.12. Características de tecnologías inalámbricas usados en HAN.	54
5.13. Comparación entre HTTP, CoAP y MQTT.	59
6.1. Descripción de Módulos.	66
6.2. Información de electrodomésticos objetivo.	69

Índice de Figuras

2.1. Smart Home	11
2.2. Funciones principales de los Smart HEMS	12
2.3. Diseño modular de recintos habitables.	14
2.4. Diseño modular de recintos no habitables.	15
3.1. Diagrama esquemático CEMS	16
3.2. Diagrama de integración de los VE con los ME	17
3.3. Diagrama de concepto V2H-HCPV	18
3.4. Vista previa del sistema WHEM	20
3.5. Funciones de aplicación móvil	22
3.6. Framework de Software HEMS	25
3.7. Diagrama del prototipo de sistema de gamificación	26
3.8. Aplicación Web de Gamificación	27
3.9. App Toshiba Feminity Club	27
4.1. Diagrama de Capas IoT	30
4.2. Arquitectura de componentes propuesta	39
5.1. Shelly Plug	41
5.2. Shelly Plug S	42
5.3. TP-Link HS110	43
5.4. Develco Smart Plug Mini.	44
5.5. SonOff Pow R2	45
5.6. Firmware Tasmota.	47
5.7. Firmware ESPEasy.	48
5.8. Jetson Nano	50
5.9. Raspberry Pi 3B+	51
5.10. NodeMCU V2	52
5.11. Analogía del camión Request-Response	60
5.12. Analogía del camión Publisher-Subscriber	61
5.13. Modelo de comunicación Request-response.	61
5.14. Modelo de comunicación Publisher-subscriber.	62
6.1. Arquitectura de componentes propuesta.	64
6.2. Diagrama de Flujo Módulos.	65
6.3. Conexión Sonoff a adaptador TTL	67
6.4. Sonoff POW R2 conectado a adaptador TTL.	68

6.5. GPIO de Sonoff POW R2.	68
6.6. Diagrama de casa iHEMS.	70
6.7. Smart Plug conectado a Calefactor	71
6.8. Configuración mosquitto.conf	73
6.9. Modelo-Navegación	73
7.1. Consumo de un dispositivo en ESPurna	74
7.2. Configuración MQTT en ESPurna	75
7.3. Topico sonoff en Mosquitto dentro de Raspberry Pi	76
7.4. Topico sonoff en Mosquitto dentro de servicio en la nube (DigitalOcean)	76
7.5. Base de datos InfluxDB	76
7.6. Resultado login	78
7.7. Resultado Refrigerador	78
7.8. Resultado TV Dormitorio	79
7.9. Resultado Calefactor	79
7.10. Resultado Lavadora	80
7.11. Resultado Living	80
1. Menú ESPurna	87

1 | Introducción

En la actualidad, el consumo de energía de la red residencial -incluidos los hogares y los edificios- aumenta continuamente.

Estamos en una era en que se necesita un cambio, porque como usuarios nos estamos volviendo más conscientes del uso eficiente de la energía, queriendo incluso llegar a producirla mediante paneles fotovoltaicos o turbinas eólicas, y ese cambio viene de la mano de las Smart Grids, el Internet of Things (IoT), el Internet of Energy (IoE) y los Home Energy Management System (HEMS).

Se ha prestado gran atención a los sistemas de gestión de la energía en el hogar (HEMS por su sigla en inglés). Sin embargo, los electrodomésticos que ya se utilizan son cargas no inteligentes que no admiten ninguna instalación de comunicación y procesamiento de la información.

1.1. ¿Cómo son los HEMS actualmente?

Se pueden considerar dos enfoques principales para la comunicación entre los electrodomésticos y los HEMS:

- Hacer que los electrodomésticos sean inteligentes incorporando la comunicación y el procesamiento de la información.
- Integrar inteligencia en las tomas de corriente.

Para la primera solución, hay muchas cargas no inteligentes y resulta más costoso incrustar la inteligencia específica en los electrodomésticos para convertirlos en nodos de carga inteligentes (SLN por su sigla en inglés). En el caso de la segunda solución, para

implantar la inteligencia (comunicación y procesamiento de la información) en las tomas de corriente, es necesario realizar importantes modificaciones en la infraestructura del cableado eléctrico [1].

1.2. Equipo de desarrollo

Dentro del contexto de memorias multidisciplinarias se formó el equipo iHEMS, el cual esta compuesto por:

- Nicolás Calderón: Estudiante de Ingeniería Civil Telemática, encargado del desarrollo de la capa física y de comunicación del desafío.
- Sebastián Yuste: Estudiante de Ingeniería Civil Telemática, encargado del desarrollo de la capa middleware y aplicación del desafío.

1.3. ¿Cuál es nuestro desafío?

Como equipo iHEMS se enfrento el desafío de desarrollar una plataforma para: recoger, almacenar y mostrar el consumo energético de diferentes electrodomésticos. Para ello se utilizó un modelo de monitoreo intrusivo de cargas, que implica la instalación de smart plugs, los que se ubican entre los electrodomésticos y la toma de corriente, con el fin de monitorear continuamente el consumo de energía de cada uno de los aparatos del hogar.

1.4. Acercamiento a la solución

Este desafío nace con el propósito de extender el trabajo realizado previamente por el equipo de EViG, en el que hacen una prueba de conceptos acerca de la toma y almacenamiento de mediciones de propiedades físicas de los dispositivos eléctricos del hogar (potencia, corriente, voltaje, entre otros). Su objetivo principal es desarrollar una plataforma de gestión de energía para la red integrada de vehículos eléctricos (EViG) en el sistema de distribución de energía. Se pretende alcanzar el siguiente objetivos de investigación:

- Desarrollo de una plataforma de gestión energética inteligente basada en IoT para una casa inteligente con un vehículo eléctrico. La plataforma propuesta debe ser capaz de gestionar la carga del vehículo eléctrico mediante el control de los electrodomésticos inteligentes de la casa y los recursos de energía renovable para cumplir con las necesidades del consumidor sin sobrecargar el sistema de distribución de energía.

En este documento se exploró un nuevo enfoque, que tiene como objetivo presentar un funcionamiento eficiente para los electrodomésticos no inteligentes utilizando SLN en una red de área doméstica (HAN por su sigla en inglés). La solución SLN es una solución de bajo costo que no necesita ningún cambio en la infraestructura del cableado en el hogar -ya que va entre la toma de corriente y la carga- ni ningún cambio en los electrodomésticos en la fase de fabricación. Los nodos de carga inteligentes se distribuyen dentro del hogar para formar una HAN.

Se pueden utilizar diferentes tecnologías de comunicación inalámbrica como medio de comunicación.

Las principales aportaciones de la solución propuesta son:

- No es necesario modificar las infraestructuras eléctricas del hogar.
- No es necesario modificar los aparatos eléctricos del hogar.
- La solución puede soportar todo tipo de cargas no inteligentes.
- La solución proporciona diferentes medidas como la tensión, la corriente, la potencia y el factor de potencia.

El desafío comienza desde la toma de datos hasta su visualización y no se enfoca solamente en el desarrollo de una plataforma, debido a que el equipo espera que en el desarrollo de este se explore y compruebe que las tecnologías utilizadas hasta el momento son las mejores o si existen soluciones alternativas que permitan desarrollar un prototipo más sólido que logre mejores resultados para la solución general. **Se destaca que en este documento se abarca solo contenidos referentes a la toma de datos y comunicación de ellos.**

Una vez estos datos se recopilen y almacenen serán utilizados por el equipo para realizar detección de actividades del diario vivir analizando el uso de los dispositivos

electrónicos mediante machine learning. Con esto se podrá analizar las conductas de personas que viven solas y generar algún tipo de alerta en caso de no recibir información de sus electrodomésticos.

1.5. Objetivo General

Diseño e implementación de una plataforma IoT para recoger, almacenar y mostrar el consumo de energía de diferentes electrodomésticos en una casa inteligente con el fin de concientizar sobre el consumo de energía a los usuarios.

1.6. Objetivos Específicos

1. **Definición de requisitos y servicios:** Permite conocer las problemáticas del cliente y definir los objetivos a lograr durante el desarrollo total del proyecto planteado por la contraparte. Además, da paso a la discusión con la contraparte para que las funcionalidades propuestas y desarrolladas como solución sean acorde a lo que ellos requieren.
2. **Definir una arquitectura de red para los sensores y medidores inteligentes:** Estructurar una arquitectura de red que conecte nodos, gateway y servidores de forma eficiente, con el fin de poder realizar mediciones de consumo eléctrico y transmitir la información obtenida para luego ser utilizada.
3. **Levantar un servidor de almacenamiento en la nube:** Permite almacenar y acceder a la información almacenada desde cualquier lugar, mediante el uso de una API.
4. **Desarrollar plataforma para la visualización:** Dicha plataforma cuenta con la visualización del consumo exacto de electricidad a la fecha en que se hace la revisión en la plataforma. Esta información se puede desglosar en el consumo individual de cada dispositivo y el tiempo de actividad del mismo.

Si bien la definición de los objetivos específicos abarca la completitud del desarrollo del desafío, en este documento solo se expone el objetivo específico 1 y 2.

1.7. Estructura del documento

Este documento se estructura de la siguiente forma: en el Capítulo 2 se estudió los conceptos referentes a Smart Grid, IoT, IoE y la estructura y funcionamiento de los Smart Homes. En el Capítulo 3 se hizo una revisión de los trabajos previos referentes al uso de hardware, protocolos y aplicaciones que utilizaron en HEMS los autores de los diferentes documentos mencionados en él. En el Capítulo 4 se estructura el modelo de capas del desafío, y se explica los componentes pertenecientes a cada capa y la función que cumplen dentro del desarrollo. Además, se señala la arquitectura de componentes propuesta de la solución basados en lo estudiado en el Capítulo 3. En el Capítulo 5 se comparan y seleccionan las diferentes tecnologías que se utilizan en cada capa, ahondando específicamente en los smart plugs y el firmware para la capa física y los protocolos y modelos de comunicación para la capa de comunicación. En el Capítulo 6 se presenta las configuraciones y desarrollo por capas de las tecnologías elegidas en el capítulo anterior. En Capítulo 7 se muestra lo obtenido durante el desarrollo completo del desafío, además de analizar el cumplimiento de los requisitos establecidos por el cliente, la verificación de su funcionamiento y el plan de testing realizado. Por último, en el Capítulo 8 se genera un análisis del trabajo realizado junto con un plan de trabajos futuros para mejorar el funcionamiento de la plataforma.

2 | Marco Teórico

En el capítulo anterior se plantaron los conceptos de Smart Grids, el Internet of Things (IoT) y el Internet of Energy (IoE). Ahora es cuando se ahonda un poco más en ellos y se explica su relevancia en el desarrollo de un HEMS.

2.1. ¿Qué son las Smart Grids?

Como se explica en [2], las Smart Grids son aquellas redes eléctricas que permiten la integración inteligente de las diferentes acciones de los usuarios, con el fin de entregar energías sostenibles, económicas y seguras. Además, una Smart Grid busca entregar tecnologías innovadoras, junto con el monitoreo, control y comunicación inteligente, con el objetivo de:

- Reducir el impacto ambiental que produce el consumo eléctrico.
- Promover e informar a los consumidores de diferentes fuentes de suministro eléctrico.
- Entregar un servicio seguro y de confianza a los consumidores finales.
- Integrar de manera sencilla a la red eléctrica generadores de distintos tipos y baterías de diferentes tamaños.

¿Por qué Smart Grids?

Los Smart Grids permiten innovar en 4 aspectos significativos [2]:

1. Energías más limpias: plantas más eficientes, energías renovables no convencionales más accesibles.
2. Mejoras significativas en las tecnologías de almacenamiento.

3. Sistemas de comunicaciones inteligentes y autónomos.
4. Generación distribuida.

Esto lleva a tener hogares que se abastezcan de manera autónoma de electricidad, provenientes de fuentes de energía renovables con diferentes tipos de generadores -solares, eólicos, etc-, pudiendo incluso abastecer a otros consumidores o entregar la energía a las distribuidoras para disminuir los costos de consumo. También se podrá pensar en casas que cuenten con estaciones eléctricas que abastezcan a los autos eléctricos o incluso tener sistemas V2H (Vehicle to Home) en que los vehículos entreguen parte de la energía de sus baterías para alimentar a los dispositivos del hogar. Incluso se puede pensar en servicios que permitan al consumidor decidir cuándo, cuánto y a qué costo consumen la energía, logrando un mercado eléctrico elástico y que favorezca a todos sus integrantes.

2.2. Internet de las cosas (IoT)

Otro de los conceptos relevantes para la comprensión de la solución propuestas es el del “Internet de las cosas”, el que se explica en [3]. Con él se busca dar acceso a internet a los diferentes dispositivos que utilizamos en nuestro día a día como autos, electrodomésticos, sensores, entre otros. El fin de esto es lograr que los objetos intercambien datos por medio de internet para lograr automatizar algunas funciones de los dispositivos, y así liberar el requerimiento de una supervisión de personas.

El IoT es utilizado en diversos campos de trabajo debido a su versatilidad y fácil implementación dentro de la industria, permitiendo monitoreo y control de regadíos inteligentes, conocer la localización del ganado, detectar movimiento de personas dentro de un área, entre muchas otras cosas. Todo esto ha sido posible gracias al respaldo tecnológico que ha tenido el IoT. Algunas de las tecnologías relevantes son:

- **Acceso a tecnología de sensores de bajo costo y bajo consumo:** La industria de sensores cada vez hace sensores más asequibles y fiables, lo que favorece al desarrollo del IoT.
- **Conectividad:** Se ha creado una gran cantidad de protocolos para transferencia de datos entre cosas.

- **Plataformas de Cloud Computing:** La gran cantidad de proveedores de servicios en la nube permite escalabilidad en los proyectos de la industria, sin necesidad de controlarlo todo.
- **Machine Learning y Análisis:** El creciente campo del aprendizaje automático permite manipular la información recolectada por los objetos y realizar análisis sobre ella, ampliando los límites del IoT.
- **Inteligencia artificial conversacional:** Los avances en redes neuronales han llevado al procesamiento de lenguaje natural, generando asistentes personales digitales como Alexa, Cortana o Siri y se han vuelto atractivos, asequibles y fáciles de usar de manera doméstica.

Dentro de los campos a nivel industrial se encuentran:

- Fabricación Inteligente.
- Ciudades Inteligentes.
- Redes eléctricas inteligentes.
- Entre otros.

En el desafío se utilizó el IoT dentro del campo de las redes eléctricas inteligentes (Smart Grid) para conocer los valores de consumo de distintos electrodomésticos detectado por smart plugs, los que son enviados por telemetría hacia un nodo central para luego ser procesados y almacenados.

2.3. Internet de la Energía (IoE)

También es necesario conocer el concepto del internet de la energía (IoE), explicado en mayor profundidad en [4]. El internet de la energía busca que los consumidores sean conscientes del ciclo energético por medio de las TIC y el IoT, formando una cadena digital de energía basado en el cuidado de la misma.

Dentro del ciclo de la electricidad influyen diferentes actores y en diferentes procesos que se detallaran a continuación.

Los actores involucrados son: Consumidores/Productores, red eléctrica principal y las fuentes de energía renovable.

Respecto a las fuentes de energía renovables se busca incrementar el acceso a ellas. Con esta acción se podría:

- Generar soluciones de generación energética innovadoras y vanguardistas.
- Acelerar los procesos de generación de energía de manera local.
- Equilibrar la oferta y demanda de energía.
- Generar la integración de energías heterogéneas.

Todo esto debido a que se espera que la energía pueda fluir en múltiples direcciones: de la distribuidora al cliente y del cliente a la distribuidora o incluso a otro cliente.

Por otro lado, la distribuidora eléctrica procura mejorar la estabilidad de la red, lo que repercute en:

- Evitar cortes del suministro eléctrico.
- Disminuir tiempos de inactividad .
- Tener un mayor control de la calidad de la energía entregada por la red.

Esto ayuda a obtener un servicio continuo y con un flujo constante y sin fluctuaciones de energía hacia los distintos hogares.

Por último, la IoE enfocada en los productores y consumidores (desde ahora prosumers) se encarga de optimizar el uso y la gestión de los recursos distribuidos, los cuales representan todo tipo de energía generada a baja escala y localizada cercana a la fuente de consumo. Algunos ejemplos de fuentes de energías distribuidas son los paneles fotovoltaicos y las turbinas eólicas que se instalan en los hogares.

Dentro de la optimización y gestión de los recursos distribuidos se espera:

- Generar transacciones de energía peer-to-peer (P2P).
- Elaborar un sistema de comercio autónomo basado en microgrids.
- Controlar los recursos distribuidos de manera automática.

Con estos 3 actores se espera lograr un ciclo energético que contribuya a disminuir la carga que existe a la red eléctrica, y para eso las microgrids son la piedra angular del IoE que permitirán abastecer gran la demanda energética que los consumidores generan.

2.4. Monitoreo de carga

Al centrarse en el desarrollo de una plataforma de monitoreo de cargas es inevitable que surjan las preguntas: ¿Qué es el monitoreo de carga? y ¿Como se realiza?.

El monitoreo de cargas se encarga de detectar el consumo de los electrodomésticos en hogares inteligentes permitiendo que los usuarios puedan gestionar y manejar sus dispositivos para que funcionen de forma autónoma y óptima, siendo el primer paso para lograr la eficiencia energética y la concientización de los usuarios, como se explica en [5]. En este artículo también se responde la segunda pregunta: ¿Como se realiza?. Para ello se puede realizar monitoreo de carga mediante 2 métodos: Monitoreo de carga no intrusiva -NILM por su sigla en inglés- y monitoreo de carga intrusivo -ILM por su sigla en inglés-.

En NILM se utiliza un enfoque de monitoreo basado en software en donde se utiliza un único punto de detección de consumo energético. Esta solución ha sido ampliamente estudiada durante el último período de tiempo por ser una manera más atractiva de monitorear, debido a su bajo costo de implementación -ya que solo se monitorea en 1 punto-, pero a pesar de esto, el método NILM muestra resultados con menor precisión y una mayor dificultad de implementación en escenarios reales.

Por otro lado, ILM es un método basado en hardware donde se distribuyen los puntos de medición con el fin de obtener mayor fiabilidad de los datos y eficiencia. Esta alternativa es relativamente más cara que su contra parte, debido a que se necesita un mayor número de dispositivos para su implementación. Estos dispositivos son submedidores o nodos sensores que se instalan cerca de los electrodomésticos objetivos -generalmente entre el dispositivo y la toma de corriente-. Sin embargo, el ILM está tomando fuerza, debido a que permite el control del funcionamiento de dispositivos y electrodomésticos, además de medir su consumo, sin mencionar que permite obtener el perfil de consumo de los electrodomésticos

mediante el etiquetado de los datos.

Dado lo anterior, el equipo de trabajo ha decidido utilizar el método ILM para la toma de datos del consumo, y es el método que se utilizó a lo largo de todo el diseño y desarrollo de la solución propuesta.

2.5. Arquitectura de Smart Home

En general, la configuración del sistema del hogar inteligente/HEMS se muestra en la Figura 2.1. La configuración del sistema puede clasificarse en diferentes categorías como: hogar autónomo, hogar conectado a la red, hogar conectado a la red con sistema híbrido de energía renovable (HRES por su sigla en inglés), hogar conectado a la red con HRES y sistema de almacenamiento de energía (ESS por su sigla en inglés).

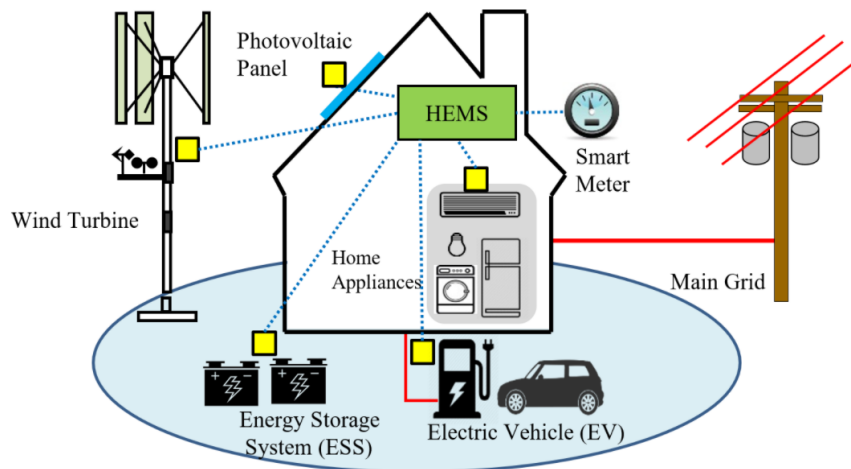


Figura 2.1: Diagrama esquemático de HEMS en el Smart House

Tipo 1: Hogar autónomo/HEMS. Esta configuración se suele denominar fuera de la red. La mayoría de los hogares/HEMS autónomos incluyen fuentes de energía renovable como la fotovoltaica y/o los aerogeneradores.

Tipo 2: Hogar/HEMS conectado a la red. En esta configuración, el hogar/HEMS está conectado a la red sin RES o ESS. El HEMS pretende optimizar el consumo de energía en función de diferentes criterios como el tiempo de uso (TOU por su sigla en inglés), el precio en tiempo real y el confort del usuario.

Tipo 3: Hogar/HEMS conectado a la red con HRES. En esta configuración, el hogar/HEMS está conectado a la red con RES como la fotovoltaica y/o los aerogeneradores.

Tipo 4: Hogar conectado a la red con HRES y ESS. Esta configuración admite la conexión a la red y la integración de las RES con los ESS.

El desarrollo del desafío se centró en el **Tipo 2: Hogar/HEMS conectado a la red.**

2.6. Funciones de Smart Home

En la Figura 2.2 se muestran las principales funcionalidades de los Smart HEMS, que incluyen el monitoreo, el registro, el control, la gestión y la alarma [6]. Estas se resumen en Figura 2.2.

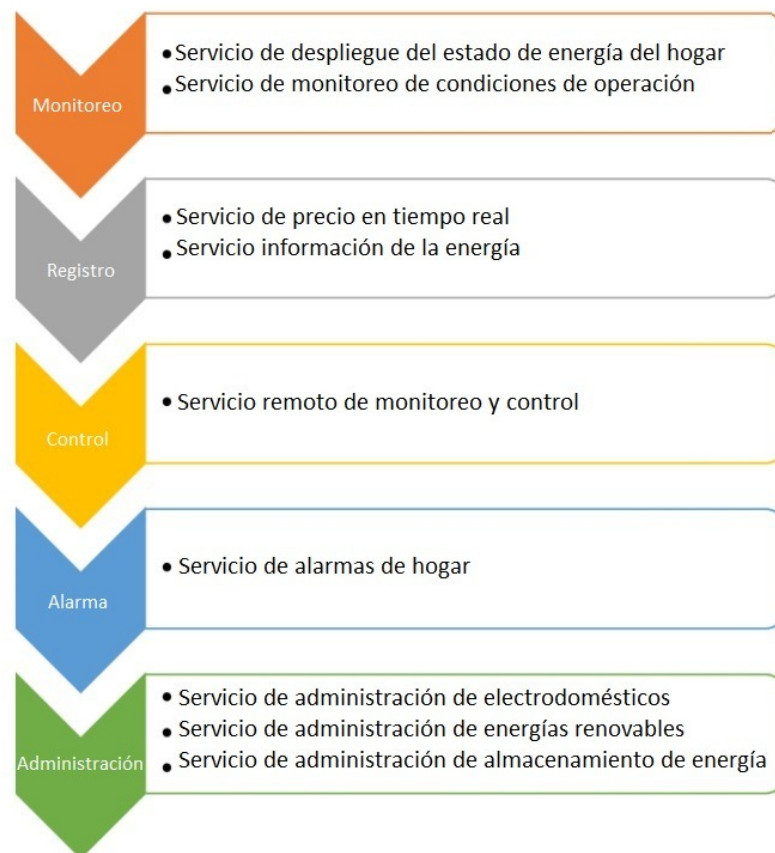


Figura 2.2: Funciones principales de los Smart HEMS

Entrando en más detalle sobre cada una de las funciones se tiene:

2.6.1. Monitoreo

El monitoreo ofrece un fácil acceso a la información en tiempo real sobre el consumo de energía y permite a los usuarios centrarse en el ahorro de electricidad. También puede proporcionar servicios de visualización de los modos de funcionamiento y el estado energético de cada electrodoméstico.

2.6.2. Registro

El registro consiste en recopilar y guardar la información de los datos sobre el uso de la electricidad de los electrodomésticos, las generaciones de los DER y el estado del almacenamiento de energía. Este servicio también contiene un análisis de la respuesta a la demanda para conocer los precios en tiempo real de la red eléctrica.

2.6.3. Control

Hay dos tipos de control: el control directo y el control remoto. El control directo se implementa tanto en el equipo como en el sistema de control; mientras que el control remoto significa que los clientes pueden acceder en línea para supervisar y controlar los patrones de uso de los dispositivos en el hogar a través de un ordenador personal o un teléfono inteligente desde el exterior.

2.6.4. Gestión

La gestión es la función más importante de HEMS para mejorar la optimización y la eficiencia del uso de la energía eléctrica en la casa inteligente. Abarca una gama de servicios que incluyen el servicio de gestión de sistemas de energía renovable, el servicio de gestión de almacenamiento de energía, el servicio de gestión de electrodomésticos y servicio de gestión de vehículos eléctricos y baterías.

2.6.5. Alarma

Genera una alarma y la envía al centro HEMS con información sobre la localización de fallos, por ejemplo, si se detecta alguna anomalía.

2.7. Diseño de Hogar

Para generar una estructura del hogar se diseñó un prototipo modular, en donde se establece la cantidad de smart plugs por habitación. Estos módulos se dividieron entre recintos habitables y no habitables como se muestra en la Figura 2.3 y Figura 2.4

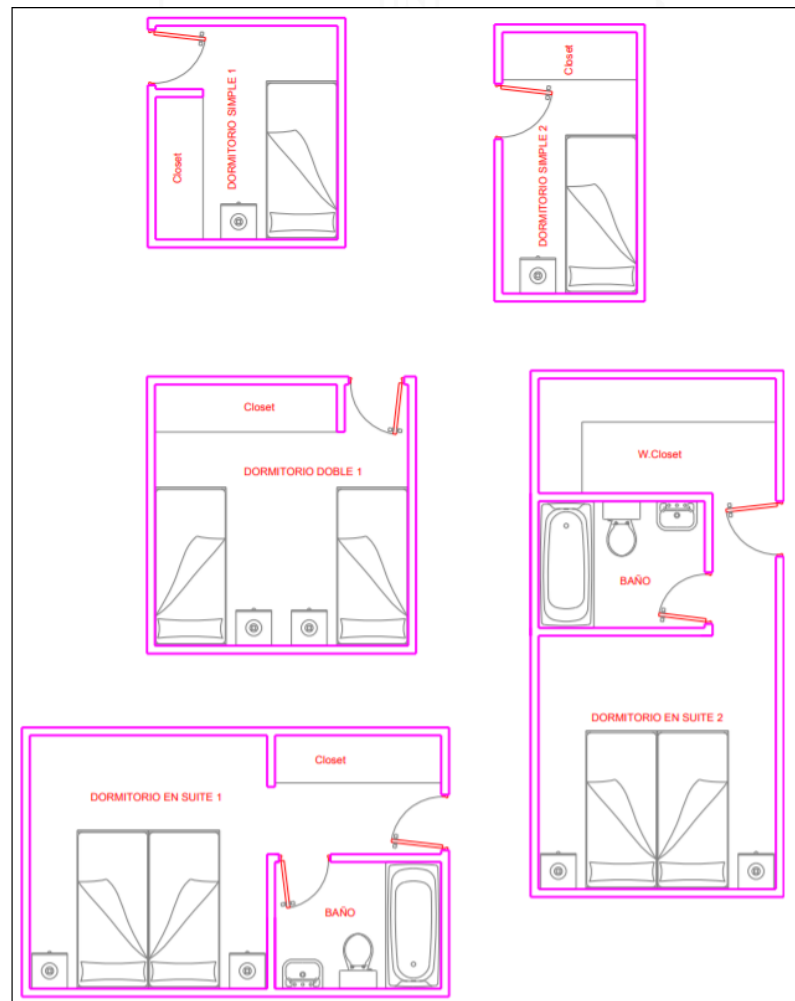


Figura 2.3: Diseño modular de recintos habitables.

Con esto en mente, el prototipo de hogar a desarrollar es una conjunción de habita-

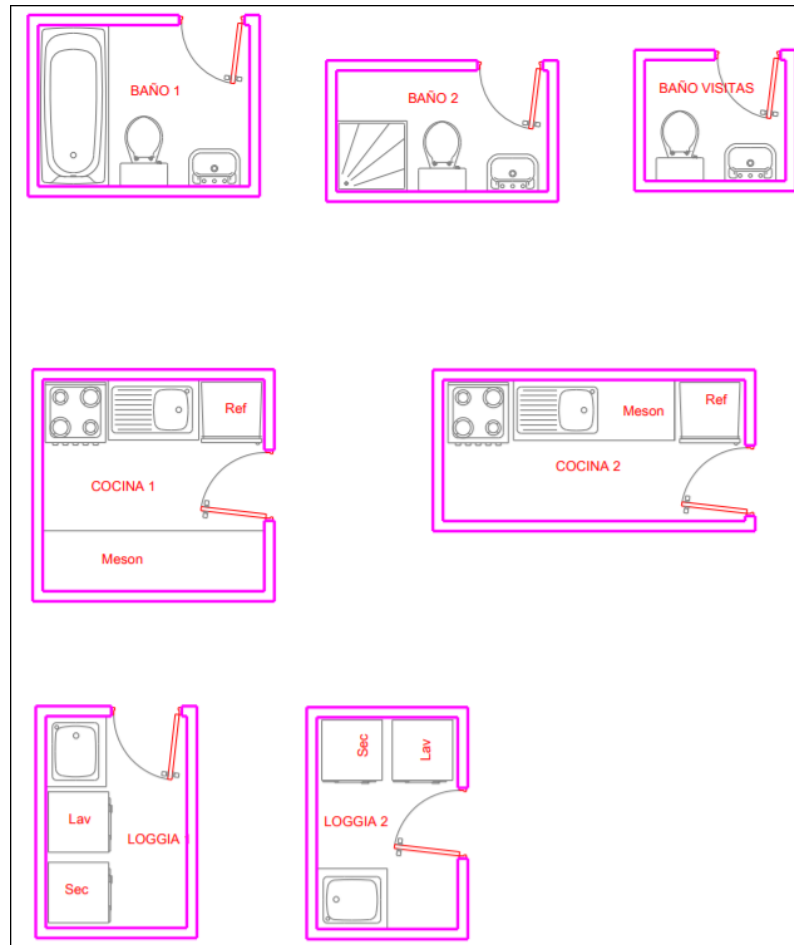


Figura 2.4: Diseño modular de recintos no habitables.

ciones las cuales tienen definida la cantidad de smart plugs a utilizar por recinto.

El diseño de estos recintos fueron hechos en base a las leyes chilenas de habitabilidad, las cuales están definidas en [7]. Sin embargo, la casa de testing cuenta con 5 smart plugs distribuidos en diferentes habitaciones, monitoreando los siguientes electrodomésticos objetivos: Refrigerador, TV Dormitorio, Calefactor, Lavadora, TV Living.

3 | Estado del arte

Una vez comprendidos los principales conceptos asociados al monitoreo energético de los electrodomésticos, se procede a mostrar el proceso investigativo realizado, con el fin de obtener referencias y extraer información realizada por diferentes autores para poder generar una solución que permita el monitoreo de energía en un hogar.

En [8; 9], los autores clasificaron el sistema de gestión de la energía (EMS) en cuatro grupos: sistema de gestión de la energía del hogar (HEMS), sistema de gestión de la energía del edificio (BEMS), sistema de gestión de la energía de la fábrica (FEMS), sistema de gestión de la energía de la comunidad (CEMS), como se muestra en la Figura 3.1.

El CEMS desempeña un rol importante en toda la comunidad mediante la comunicación de otros sistemas dentro y fuera de ella. Además, este calcula el balance energético óptimo y -en caso de emergencia- controla el flujo de energía hacia las entidades inferiores para cubrir la demanda energética.

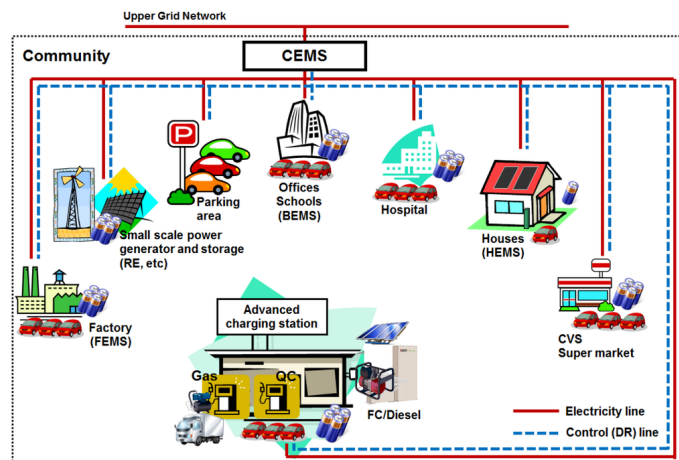


Figura 3.1: Diagrama esquemático CEMS [8]

La Figura 3.2 muestra la integración de los vehículos eléctricos como apoyo al CEMS a pequeña escala. El EMS controla la demanda y el suministro de electricidad a través de la información procedente de diferentes fuentes como: la carga, la información meteorológica, la empresa de servicios públicos y el sistema de información de vehículos (VIS).

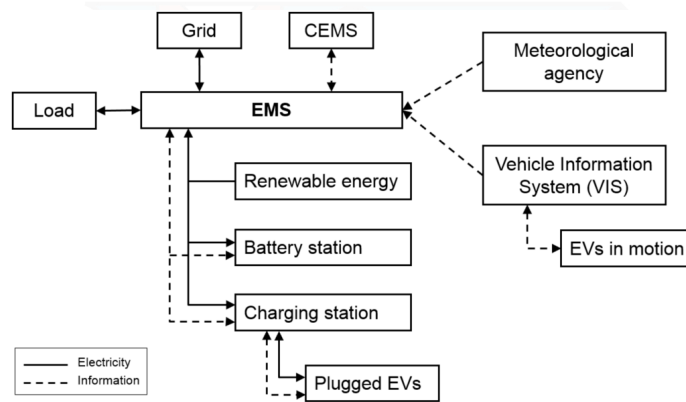


Figura 3.2: Diagrama de integración de los VE con los ME [8]

Como el vehículo eléctrico tiene una gran cantidad de almacenamiento de energía, puede hacer que el EMS sea más eficaz.

En [9], se propuso un control predictivo del modelo de la batería de almacenamiento del vehículo eléctrico, basado en un modelo autorregresivo y en la optimización del enjambre de partículas en un sistema de gestión de la energía doméstica. El objetivo principal era controlar el estado del almacenamiento del VE para minimizar el coste de la electricidad del HEMS y evaluar el beneficio de la red eléctrica. Se consideraron tres casos “sin PV y sin red eléctrica”, “con PV y sin red eléctrica” y “con PV y con red eléctrica”. El modelo a utilizar es el llamado “AR model” [10], en donde se controla el almacenamiento de energía de un VE, considerando viajes, ya que la función principal de un automóvil es transportar a los usuarios. En él se consideraron dos perfiles con diferentes acciones, incluyendo “perfil de desplazamiento: ir a la oficina, trabajar y volver a casa” y “perfil de ama de casa: llevar a un niño a la guardería, hacer la compra y buscar a un niño en la guardería”.

La manera en que solucionan el problema de optimización es mediante el uso de PSO -Particle Swarm Optimization-, considerando 3 casos:

- VE sin panel fotovoltaico (PV) y sin red eléctrica

- VE con PV y sin red eléctrica
- VE con PV y con red eléctrica

El termino “con/sin red eléctrica” hace referencia a cuando el VE se puede cargar/-descargar en la oficina del usuario, y con esto obtener ganancias de la venta de energía a la empresa para la cual trabaja el dueño del VE.

Como resultado del problema de optimización se determinó que el mejor de los casos para reducir el costo asociado al consumo eléctrico del hogar es "VE con PV y empresa".

En [11], al igual que el artículo anterior, propone un sistema combinado de gestión de la energía doméstica que incluye la tecnología Vehicle-To-Home (V2H) o Home Centralized Photovoltaic (HCPV). El enfoque propuesto busca controlar la demanda de energía en el hogar mediante la programación óptima de los aparatos de automatización, contemplando las diferentes condiciones climáticas que afectan la producción de energía de los PV.

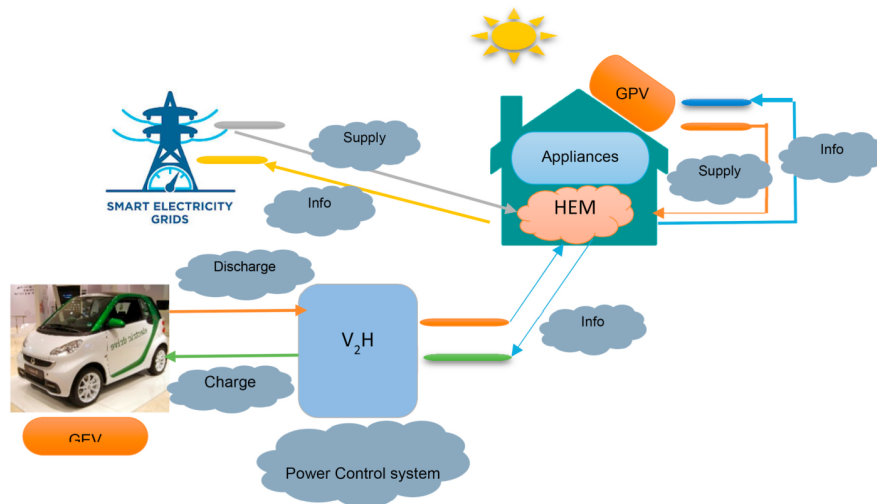


Figura 3.3: Diagrama de concepto V2H-HCPV [11]

Sin energía adicional de la red, el diseño propuesto de HCPV cubre la demanda de electricidad del hogar en tiempo soleado y nublado. Por otro lado, la combinación de PV y electricidad transmitida por V2H es suficiente para cubrir la demanda de carga doméstica con tiempo nublado. En este sentido, se diseña y discute un Sistema de Gestión de Energía Doméstica programado con varias restricciones para realizar operaciones correctas del sistema y satisfacer la demanda de carga.

El algoritmo propuesto intenta caracterizar y percibir las circunstancias de la fuente utilizando la demanda de energía.

Los resultados obtenidos demuestran que la tecnología de funcionamiento de H2V puede reducir eficazmente la demanda de energía bajo la volatilidad de la carga solar.

En [12] se investigó un algoritmo de control para el sistema de gestión de la energía en el hogar (HEMS) para supervisar y programar los aparatos eléctricos que se aplican en cualquier casa convencional.

El objetivo es reducir el consumo de electricidad y el costo de la misma en lugares con un modelo de precios basado en el tiempo de uso (Time Of Use). Para ello se realizó la implementación de hardware en una casa de prueba utilizando un sistema de control propio, smart plugs -diseñados y fabricados por los autores- y un paneles fotovoltaicos como fuente suplementaria de energía.

Los smart plugs utilizados son de 3 tipos, capaces de medir el consumo de electrodomésticos, la temperatura de aparatos térmicos, o el estado de carga y potencia de salida de las fuentes de energía.

En la arquitectura del sistema, el consumo global de los aparatos eléctricos que funcionan en la red está limitado por un algoritmo, y se mantiene por debajo de un determinado valor que se fija en función del consumo total del edificio y de la capacidad de la batería.

Este algoritmo propuesto se centra en la optimización del gasto monetario asociado al consumo eléctrico, dado que la electricidad en horas punta tiene un costo más elevado, por lo que se intenta gestionar el funcionamiento de los aparatos eléctricos para que funcionen con batería o con electricidad de la red dependiendo de la hora.

Las baterías se recargan a través del controlador de suministro de los paneles solares durante el día y de la red durante la noche, cuando el precio de la electricidad es más barato. El algoritmo puede priorizar los electrodomésticos en función de su prioridad de uso -que se define de antemano- para optimizar el consumo de electricidad basado en el límite predefinido de la demanda.

La premisa incluía 14 electrodomésticos que funcionaron durante tres meses para

su verificación. Los resultados muestran que el sistema implementado con el algoritmo propuesto es capaz de reducir el precio de la electricidad significativamente, hasta un 15 % por día en comparación con el consumo habitual.

En [13] , los autores desarrollaron un prototipo para un sistema de comunicación inalámbrica de HEMS en una casa/edificio inteligente mediante la implementación de una interfaz web entre un gestor de energía y las cargas pesadas. El prototipo del sistema ha sido diseñado, construido y probado en el laboratorio. Además, se ha analizado el canal entre el consumidor y el gestor de energía mediante simulaciones con respecto al rendimiento, el retardo de extremo a extremo y el retardo de ida y vuelta. En la Figura 3.4 se muestra el diagrama esquemático del sistema de gestión de energía doméstica inalámbrica (WHEM).

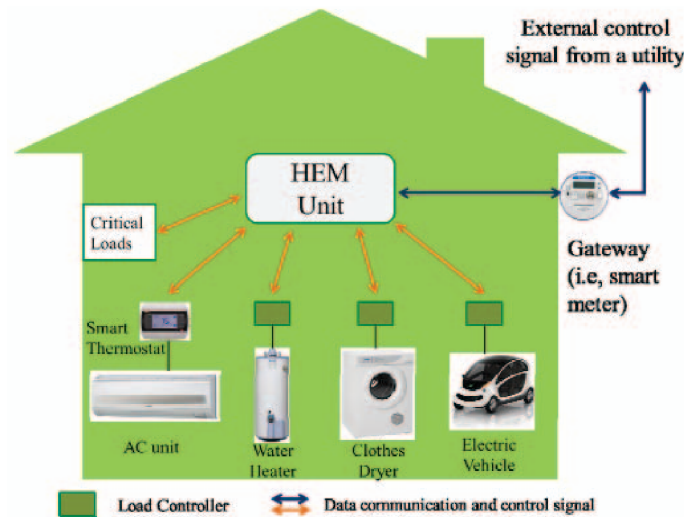


Figura 3.4: Vista previa del sistema WHEM [13]

Los autores dividieron su solución en 3 bloques: el módulo de control de la carga, que se encarga de gestionar y realizar las mediciones de consumo de los dispositivos; la unidad de comunicación, que facilita el intercambio de información; y el gateway, que es el enlace de comunicación entre la unidad de gestión doméstica y el gestor de energía en el lado de la red inteligente.

El sistema WHEM se compone de los 3 módulos explicados con anterioridad en donde cada uno sirve para lo siguiente:

- **Control de carga:** Proporciona monitorización, control y comunicación.

- **Comunicación:** Establece la comunicación entre los controladores de carga y la unidad HEM a través de diferentes tecnologías de comunicación como PLC, WiFi, Bluetooth y ZigBee
- **Gateway:** Permite la comunicación bidireccional entre el lado de la empresa de servicios públicos y el sistema HEM.

Para la implementación de WHEM, el módulo de control de carga consiste en:

- Un microcontrolador (ATmega329)
- Un módulo de RF (CC1101) -que opera a una frecuencia de 433 MHz-.
- Un relé DPDT de 10A

La unidad HEM se ha implementado con una Raspberry Pi, que se comunica con los sensores a través de un enlace de radio formando una topología en estrella, que se ha configurado como un servidor web para incluir los datos de la carga, así como enviar las señales de control ON/OFF para las cargas conectadas.

Para medir el rendimiento y la fiabilidad de la red, se simuló la red de comunicación utilizando el simulador OPNET. El modelo a simular consta de 5 nodos, un concentrador y un servidor y se consideraron diferentes métricas, como el rendimiento de la red y el retardo de extremo a extremo.

En [14] se propone un método eficiente de almacenamiento de datos en bases de datos NoSQL, con el fin de supervisar el estado de los dispositivos de un hogar en tiempo real. Además se desarrollo una aplicación para la gestión de energía doméstica (HEM), la cual se utilizó para evaluar el método propuesto.

Los autores consideran el uso de una base de datos NoSQL distribuida, destacando 3 características importantes, la consistencia de los datos, la disponibilidad, y la tolerancia a particiones. La idea de utilizar una base de datos distribuida es disminuir los tiempos de solicitud de grandes volúmenes de datos, recurriendo a varios nodos de la base al mismo tiempo. La aplicación para teléfonos inteligentes implementada incluye las siguientes tres características: recordatorio, visor de costes y analizador de costes, como se muestra en la Figura 3.5.

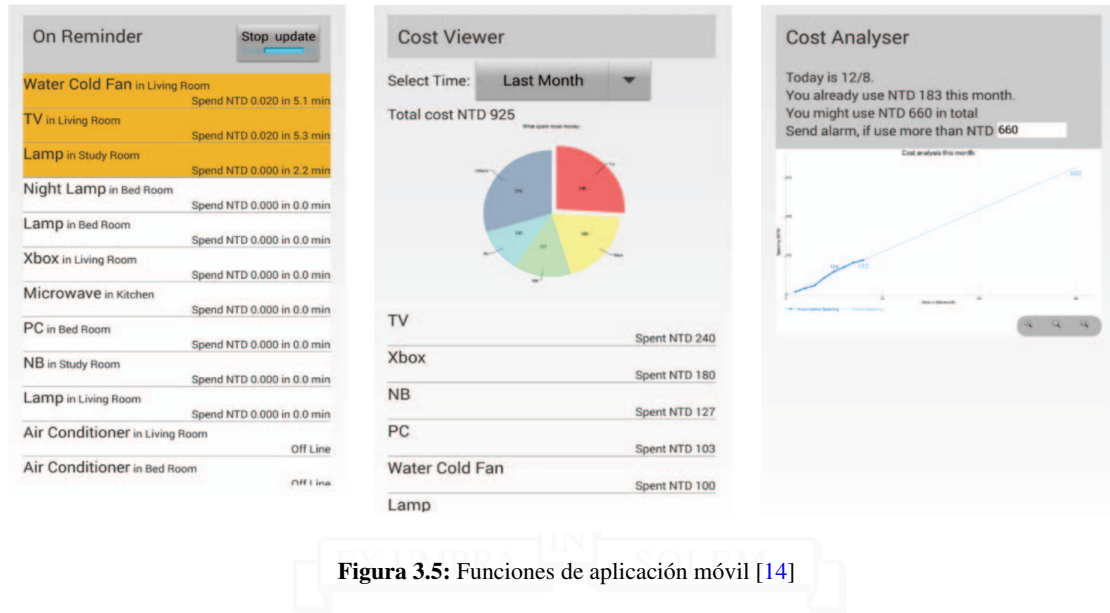


Figura 3.5: Funciones de aplicación móvil [14]

El problema que enfrentan es que al almacenar datos de series de tiempo -como las que se obtienen al realizar mediciones constantemente-, la información cercana en el tiempo suele almacenarse dentro de los mismos nodos, desperdiciando el provecho que se puede obtener de la arquitectura distribuida.

La solución implementada fue aplicar una función de hash a la llave de los registros para desagregarlos, recuperando las bondades de una base de datos distribuida.

La evaluación se hizo sobre un clúster HBase que construyeron los autores como dataset de pruebas. El método propuesto mostró resultados prometedores, entregando la información solicitada a más del 95 % de los usuarios en 0.5 s mientras otros cientos o miles de aplicaciones solicitaban información al mismo tiempo, lo que permite satisfacer al menos cientos de aplicaciones móviles simultáneamente.

En [15] se analiza un sistema de monitoreo de energía en tiempo real, basado en IoT para el control y la supervisión dentro de una industria, donde se ha decidido establecer un sistema eficiente de monitoreo de la energía para analizar su utilización diaria, como paso preliminar de las actividades de conservación de esta. En este sentido, se consideran los sistemas de monitorización de energía basados en IoT. Muchos tipos de dispositivos IoT como Arduino, Raspberry Pi, Intel Edison, Mediatek linkit one, NVIDIA Jetson Nano, etc. están disponibles para realizar esta tarea.

En comparación con otros métodos de adquisición de datos, los sistemas basados en Raspberry Pi resultan ser la mejor opción para realizar la tarea en base a sus criterios técnicos, por su bajo coste y por ser una tecnología altamente fiable para la monitorización del consumo energético de la industria. Por lo tanto, se establece un sistema de monitorización de energía basado en Raspberry Pi con los contadores de energía existentes en la industria de los equipos de conmutación. Esta se utiliza con el entorno de desarrollo Node.js para recoger los datos (varios parámetros eléctricos) de los contadores de energía existentes en la industria y almacenarlos localmente en una base de datos InfluxDB, para acceder a ellos a través del ordenador portátil o del teléfono móvil utilizando Grafana.

El sistema de monitorización es útil para la industria para entender el patrón de energía del día a día, que es esencial para facilitar las medidas de conservación de energía para minimizar su consumo.

En [16] se estudia el impacto que tiene Telegraf - un software de recopilación de métricas-. Para ello realizaron varios experimentos para estudiar cómo afecta Telegraf al sistema base en un servidor de centro de datos y un nodo IoT.

Dentro de las pruebas realizadas se evaluaron distintas configuraciones de Telegraf evaluando diferentes escenarios, en donde se varió el orden de magnitud del intervalo medido -10ms, 100ms, 1s, 10s y 100s- y el número de métricas recogidas -1, 5 y 10 métricas-. Estas últimas están correlacionadas en la Tabla 3.1. Además, se utilizó InfluxDB 1.7.6 como el servicio de almacenamiento que recogió los datos de Telegraf, y reside en otra máquina en una red diferente del servidor o nodo IoT que tiene Telegraf instalado.

Para el escenario del servidor del centro de datos, utilizaron:

- Una instancia de Google Cloud Compute con CentOS7 (centos-7- v20190213). Esta tiene:
 - 1 vCPU (Intel Skylake o superior).
 - 3,75 GB de memoria.
 - 30 GB de disco de almacenamiento.

Para el escenario del nodo IoT, utilizaron:

Tabla 3.1: Métricas recolectadas en cada configuración [16]

Telegraf Inputs	Number of Metrics		
	<i>1</i>	<i>5</i>	<i>10</i>
inputs.cpu	X	X	X
inputs.disk		X	X
inputs.diskio		X	X
inputs.mem		X	X
inputs.net		X	
inputs.processes			X
inputs.swap			X
inputs.system			X
inputs.interrupts			X
inputs.kernel			X
inputs.kernel_vmstat			X

- Una Raspberry Pi 3 Modelo B con Ubuntu 19.10. Esta tiene:
 - 1 CPU de cuatro núcleos a 1,2GHz (ARM Cortex A53).
 - 1GB de memoria RAM.
 - 1 tarjeta de memoria MicroSD de 32GB de disco de almacenamiento.
 - 1 sensor de temperatura DS18B20 conectado a la Raspberry Pi.

Para cada uno de los casos se comparó el uso de Telegraf en el sistema, frente a un sistema de las mismas características sin Telegraf instalado.

Los resultados mostrados en la tabla II y III de [16] le permiten a los autores determinar que Telegraf es ligero y adecuado para servir como agente de monitorización en tiempo real en ambos escenarios. Sin embargo, para situaciones que requieren una monitorización de muy alta frecuencia, es decir, una recogida de métricas más rápida que 1s, este sistema puede ser demasiado intensivo en recursos para el despliegue de nodos IoT.

En [17], los autores propusieron un marco HEMS de software que es diferente de las soluciones existentes y no requiere ningún hardware adicional, lo que disminuye

significativamente el costo que implica la instalación de medidores en el hogar.

La solución propuesta ha sido diseñada e implementada en el sistema operativo T-Kernel, utilizado en las aplicaciones embebidas de los clientes. El consumo de energía se estima a partir de la CPU que se correlaciona con el porcentaje de inactividad.

En la Figura 3.6 se muestra una visión general de la arquitectura HEMS. En ella se puede observar que la capa “Software HEMS Aggregator” tiene un color diferente. Esto es debido a que los autores explican que es esta capa la más importante del sistema, ya que se encuentra continuamente recogiendo y agregando el consumo de los periféricos de los dispositivos. Por otro lado, la capa “CoAP” es la encargada de enviar la data a un servidor HEMS central. Este último se encarga de sumar la información de los dispositivos para luego desplegarla al usuario.

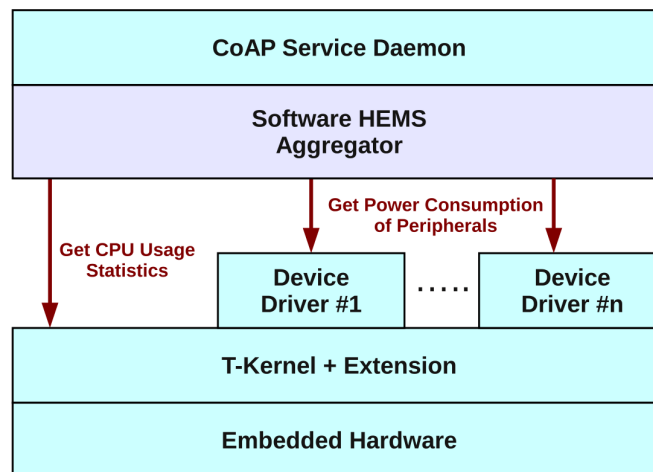


Figura 3.6: Framework de Software HEMS [17]

En [18], los autores desarrollaron y evaluaron un sistema de “experiencia de usuario como servicio” en su sistema de gestión de la energía doméstica.

Los estudios convencionales consideran la visualización del consumo de energía eléctrica mediante: tablas, gráficos y números. En ellos no se considera realizar acciones en beneficio de reducir el consumo de energía eléctrica de los consumidores. Tomando lo anterior en cuenta, el sistema desarrollado consiste en la gamificación y el ciclo Planificar-Hacer-Verificar-Actuar (PDCA, por su sigla en inglés), donde se utiliza el "perfil" del registro de vida en un juego, mientras que el ciclo PDCA permite una utilización continua

del juego por parte del usuario.

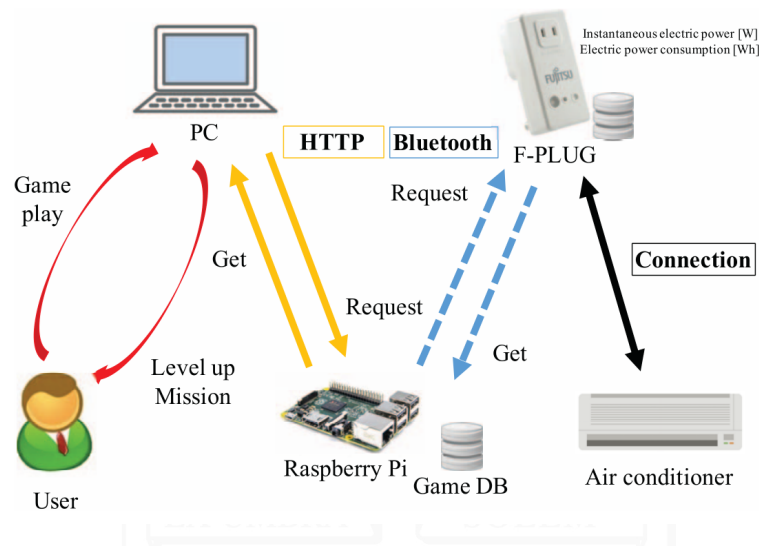


Figura 3.7: Diagrama del prototipo de sistema de gamificación [18]

En el sistema prototipo, el usuario se conecta a un servidor de juegos alojado en una Raspberry Pi a través de un PC utilizando HTTP. Se ha considerado un acondicionador de aire debido al alto consumo de energía, como se muestra en la Figura 3.7, donde se ha utilizado un smart plug para la adquisición del consumo de energía eléctrica. Por otro lado, La comunicación entre el smart plug y el servidor de juegos se realiza vía Bluetooth.

La Figura 3.8 muestra la aplicación web en la que el usuario es un personaje que se enfrenta a sus electrodomésticos. Al derrotar a los electrodomésticos, los usuarios ganan experiencia e ítems del juego al reducir el consumo de energía durante ciertas franjas horarias. Esta solución es bastante innovadora y permite a los usuarios conocer mejor el consumo asociado a sus electrodomésticos, volviéndolos conscientes de la demanda energética generada periódicamente, y así generar un plan de acción para reducir este consumo en caso de estar superando las cifras de su consumo medio mensual.

Dentro de las referencias del documento se analizó una aplicación creada por “Toshiba Feminity Club” [Figura 3.9] en la cual se observó un diseño basado en la personificación de un usuario en un personaje de la plataforma y la administración de los dispositivos del hogar por medio de una plataforma atractiva y personalizable, pudiendo modificar los ambientes del hogar y el estado de los electrodomésticos, los cuales esta separados por habitaciones.

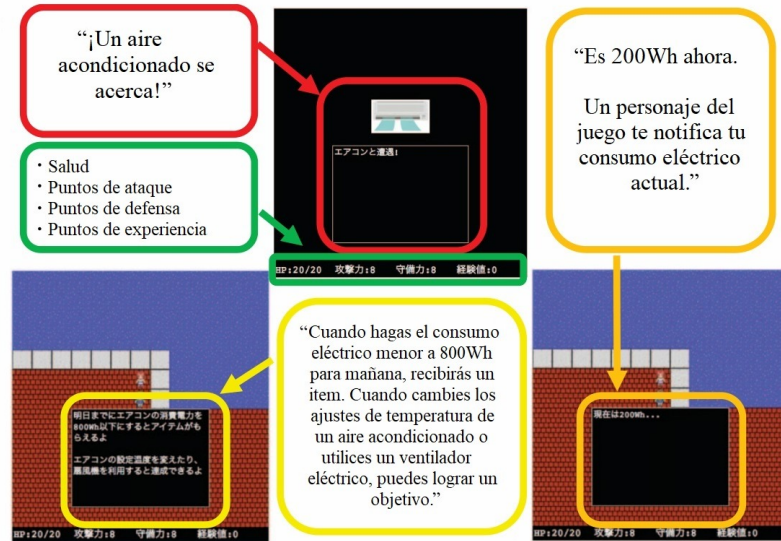


Figura 3.8: Aplicación Web de Gamificación [18]

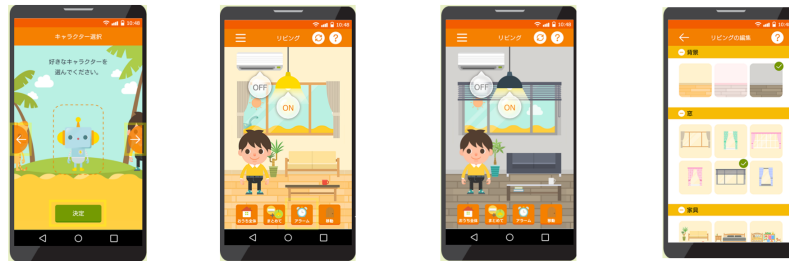


Figura 3.9: Aplicación móvil Toshiba Feminity Club [19]

Resumen

Con toda la información recopilada se logra confeccionar la tabla de resumen mostrada en Tabla 3.2 destacando las capas trabajadas por cada uno de los autores. Además, se observa cuales son las capas que se abarca en la solución presentada en el documento.

Tabla 3.2: Resumen de estado del arte

Referencia	Capa Física	Capa de Comunicación	Capa de Middleware	Capa de Aplicación
[8]	✓	✗	✗	✓
[9]	✓	✗	✗	✓
[10]	✓	✗	✗	✓
[11]	✓	✗	✗	✓
[12]	✓	✗	✗	✓
[13]	✓	✓	✗	✓

[14]	x	x	✓	x
[15]	✓	x	✓	✓
[16]	x	x	✓	x
[17]	x	x	x	✓
[18]	x	x	x	✓
iHEMS	✓	✓	✓	✓



4 | Arquitectura IoT de la Plataforma

Las mediciones del consumo es la capa más baja del sistema y la base de toda la arquitectura del sistema. Pero, si es la base de todo, ¿Cuales son el resto de capas?.

Para responder a esa pregunta es necesario visualizar nuevamente el artículo [5, Figura 2], en donde se visualiza una arquitectura para el monitoreo intrusivo de cargas. A partir de este diagrama se generó una arquitectura compacta a utilizar en este documento, con el fin de simplificar el modelo y lograr una mayor comprensión del mismo. [Figura 4.1]

4.1. Capa Física

En esta capa se produce la recolección de los datos del consumo de los electrodomésticos que se enviarán con posterioridad a las capas superiores. Contiene tanto los dispositivos físicos (electrodomésticos, vehículos eléctricos y calefacción), como los dispositivos de percepción, encargados de la adquisición de los datos.

Dentro de los dispositivos de percepción de la arquitectura se encuentran los smart plugs y la smart lighting. Los smart plugs son la base de la propuesta de solución, ya que estos dispositivos se encargan de detectar y empaquetar el consumo de los electrodomésticos objetivos. Con ellos es posible medir características dependientes del tiempo y/o basadas en la frecuencia. Las características típicas dependientes del tiempo son la potencia activa (P), la potencia reactiva (Q), la tensión (V), la corriente (I) y las trayectorias V-I. A partir de estas características, se pueden obtener otras, como la potencia compleja y la potencia aparente. Sin embargo, estos dispositivos tienen un factor crítico que viene dado por las limitaciones tecnológicas, y es el tiempo de muestreo [5]. Este tiempo es variable y dependiente del



Figura 4.1: Diagrama de arquitectura para visualización de consumo

modelo específico de smart plug a utilizar.

4.2. Capa de Comunicación

En esta capa se produce la integración y comunicación entre dispositivos. Aquí es donde se definen las tecnologías de comunicación y los protocolos a utilizar. La elección de ellos depende netamente del objetivo al cual se apunta.

Como se observa en [20] y [5], los servidores donde se procesan y almacenan los datos se encuentran fuera de las Home Area Network (HAN) y en el extremo de la red, por lo que las tecnologías de mayor utilización en este tipo de arquitecturas son las tecnologías inalámbricas de corto alcance como Bluetooth, Wi-Fi o Zigbee, además de integrar un nodo central o gateway que es el intermediario entre la HAN y la WAN -lugar donde se

encuentran los servidores-.

4.3. Capa de Middleware

Esta capa es la encargada de recopilar los datos provenientes de capas anteriores con el fin interactuar entre los dispositivos IoT y la aplicación. Los requisitos de esta capa son muy elevados, contemplando: la abstracción de la arquitectura, la gestión de las funciones, el diseño de la programación y la implementación. La capa de middleware no sólo oculta las complejidades de las capas inferiores, sino que también proporciona servicios de datos para que las capas superiores cumplan fácilmente sus objetivos funcionales. Por lo tanto, esta capa es la más difícil de diseñar, ya que es independiente de la aplicación y se centra más en los datos. Además, debe tener en cuenta los requisitos funcionales, como la gestión de datos, el almacenamiento de datos, el procesamiento de datos (análisis de big data, análisis de datos en tiempo real, análisis de datos profundos con IA), y los requisitos no funcionales, como la escalabilidad, la seguridad, la disponibilidad, la fiabilidad, la entrega de datos en tiempo real, la privacidad, entre otros [21].

Algunos de los servicios estudiados para el desarrollo de esta capa en la solución son: DigitalOcean, AWS, Google Cloud. Estas plataformas nos permiten almacenar los datos provenientes de los smart plugs y le proveen al equipo la posibilidad de escalar los servicios en un futuro.

4.4. Capa de Aplicación

La capa de aplicación es aquella que se enfoca en los servicios ofrecidos a los usuarios. Es por esto que dentro de esta capa se pueden agregar diferentes tipos de soluciones relacionadas con el monitoreo de carga intrusivo (ILM).

Dentro de las aplicaciones más utilizadas para ILM esta la comprensión del consumo energético a nivel local y global, la evaluación y simulación de entornos NILM, el reconocimiento de la actividad humana y la localización de aparatos [5]. Los detalles relativos a cada una de estas aplicaciones se pueden encontrar en [22]. Este proyecto se centra en la comprensión del consumo energético a nivel local, buscando generar conciencia del uso de

la energía en los usuarios, y a la vez generar un conjunto de datos que puedan alimentar un algoritmo de machine learning para la detección de electrodomésticos, y con ello reconocer las actividades del día a día de las personas. Cabe mencionar que los datos a desplegar por esta capa son los obtenidos por los submedidores asociados a sus electrodomésticos objetivos, por lo que los datos obtenidos deben ser etiquetados de alguna manera.

4.5. Requisitos del Sistema

Antes de comenzar a describir las tecnologías que se utilizaron para la resolución del problema, es necesario conocer los requisitos descritos por el cliente para obtener una plataforma con las características necesarias para el funcionamiento de su prototipo, además del perfil de usuarios esperados que utilizarán el sistema.

4.5.1. Requisitos Funcionales

Los requisitos funcionales definen el comportamiento del sistema, es decir, describen lo que este debe hacer. Esto desde el punto de vista de interpretar lo que espera el usuario objetivo de la plataforma. Con esto en mente los requisitos funcionales definidos son los siguientes:

RF1 El smart plug debe ser capaz de medir la energía que consumen los electrodomésticos.

RF2 El smart plug debe enviar la información de cada medición al gateway.

RF3 El gateway debe proveer conectividad a los dispositivos.

RF4 El gateway proporciona conectividad segura por medio de autenticación y posibilidad de configuración remota.

RF5 El gateway envía los datos a un servicio en la nube con los metadatos necesarios para otorgar temporalidad a la información (fecha y hora de medición).

RF6 Los paquetes de información son recibidos, procesados, almacenados y estandarizados en un servicio en la nube.

RF7 Se disponibiliza endpoints de una API para acceder a los datos desde la nube.

- RF8** La plataforma permite el ingreso de usuarios mediante autenticación.
- RF9** La plataforma permite la creación de cuentas para usuarios nuevos.
- RF10** La plataforma accede a los datos en la nube para su posterior visualización.
- RF11** La plataforma permite agregar o desagregar dispositivos a la red para su monitoreo.
- RF12** La plataforma permite ordenar los dispositivos a través de habitaciones.
- RF13** La plataforma permite navegar por los dispositivos agregados en ella para visualizar sus mediciones.
- RF14** La plataforma permite modificar y eliminar dispositivos y habitaciones.
- RF15** La plataforma notifica a los usuarios cuando se agrega, desagrega o se desvincula un dispositivo electrónico.
- RF16** La plataforma permite extracción de datos en formato csv para un usuario administrador.

Estos requisitos fueron extraídos por el equipo durante las reuniones con el cliente, pero para el desarrollo de este documento se trabajó los requisitos **RF1** a **RF5**

4.5.2. Requisitos no Funcionales

Los requisitos no funcionales del sistema definen atributos de calidad por los cuales el usuario juzga la operación del sistema. Para esto se definieron los siguientes:

- RNF1** Los dispositivos electrónicos deben estar conectados constantemente a los smart plugs.
- RNF2** La disponibilidad de la red entregada por el gateway debe tener un uptime del 100 %.
- RNF3** El gateway debe tener conectividad a internet todo el tiempo.
- RNF4** El gateway debe tener comunicación con la nube todo el tiempo.
- RNF5** La plataforma debe ser amigable, responsiva, auto informativa e intuitiva.
- RNF5** La plataforma presenta los datos de forma clara e intuitiva.

4.5.3. Requisitos de Interfaces

Una vez se conocen los requisitos funcionales y no funcionales, se procede a conocer los requisitos de interfaces, en los cuales se explica como espera el usuario que responda el sistema frente a diferentes eventos.

La Tabla 4.1 muestra la lista de eventos externos a los que el sistema responde. La primera columna indica el nombre del evento; la segunda es la descripción del mismo. El “iniciador” es el componente externo al sistema que inicia el evento. Los parámetros son los datos asociados al evento. La respuesta es el nombre de una respuesta, cuya descripción está en la Tabla 4.2.

Tabla 4.1: Eventos externos.

Evento	Descripción	Iniciador	Parámetros	Respuesta
Medición de parámetros	El smart plug detecta que el dispositivo ha comenzado a ser utilizado y comienza la medición de sus parámetros.	Detección del uso de un dispositivo	Dispositivo electrónico encendido	Paquete con propiedades físicas del dispositivo
Envío de información	Una vez se captura los datos proveniente del dispositivo, el smart plug envía el paquete al gateway	Medición de parámetros completada	Paquete con propiedades físicas del dispositivo	Transmisión de paquete a gateway
Recepción de paquete	Al recibir el paquete del smart plug, el gateway almacena los datos que luego serán enviada a un servidor en la nube	Recepción de paquete proveniente de smart plug	Paquete con propiedades físicas del dispositivo recepcionado	Almacenamiento del paquete.
Almacenamiento en la nube	Una vez el paquete es almacenado de manera local, este se procesa y estandariza para ser redireccionado y almacenado en un servicio en la nube	Estandarización y procesamiento del paquete almacenado localmente	Paquete estandarizado y procesado	Redirección y almacenamiento en la nube.
Acceso a datos en la nube	Se disponibiliza endpoints a los cuales se puede acceder para extraer los datos que luego será utilizada por la plataforma	Acceso a endpoint de la API	Consulta de información	Paquete con información solicitada

Iniciar Sesión	Cuando se accede a la plataforma con un usuario previamente registrado se muestra una vista con sus habitaciones y dispositivos registrados	Iniciar sesión en plataforma	Usuario y contraseña	Visualización de dashboard principal
Registrarse	Cuando un usuario desea utilizar la plataforma, pero no tiene una cuenta, este debe registrarse para obtener sus credenciales que le permitan ingresar a la plataforma.	Usuario registrando su cuenta en la plataforma	Datos de usuario requeridos	Visualización de dashboard principal
Visualización de la información	Cuando se recibe el paquete con la información solicitada, este es desplegado en una plataforma de visualización -ordenado por habitaciones y dispositivos- para entregarle significado a la información.	Recepción de paquete con información solicitada	Paquete con información solicitada	Visualización de la información
Modificación de parámetros	Si el usuario desea cambiar algún parámetro de los dispositivos puede hacerlo mediante la plataforma	Selección de cambio de parámetros en la plataforma	Valor de los parámetros nuevos	Notificación de cambio realizado
Alerta de nuevo dispositivo	Se alerta al usuario en caso de que haya agregado un nuevo dispositivo a la plataforma.	Conexión de un dispositivo	Paquete con información de un nuevo dispositivo	Notificación de nuevo dispositivo
Alerta de desvinculación	Cuando el usuario desvincula un dispositivo se envía una alerta informativa al usuario	Desvinculación de un dispositivo	Paquete con información de desvinculación	Notificación de desvinculación.

La Tabla 4.2 muestra de manera detallada las respuestas del sistema frente a eventos externos.

Tabla 4.2: Respuestas del sistema.

Respuesta	Descripción	Parámetros
Paquete con propiedades físicas del dispositivo	Contiene la información relevante extraída por los smart plugs	JSON con información de potencia, corriente, voltaje entre otros parámetros físicos de los dispositivos.
Transmisión de paquete a gateway	Envía el paquete desde el smart plug al gateway para su posterior almacenamiento	JSON con información de potencia, corriente, voltaje entre otros parámetros físicos de los dispositivos.
Almacenamiento del paquete.	Guarda localmente la información obtenida del paquete JSON	JSON con información de potencia, corriente, voltaje entre otros parámetros físicos de los dispositivos.
Redirección y almacenamiento en la nube.	Se estandariza y procesa el paquete JSON para ser almacenado en la nube	JSON estandarizado y parametrizado
Paquete con información solicitada	Paquete formato JSON entregado por la API	JSON con información solicitada
Visualización del dashbord principal	Vista que despliega la información del usuario que inició sesión	JSON con información solicitada
Visualización de la data	Formato gráfico para la data obtenida	JSON con información solicitada
Notificación de cambio realizado	Alerta de modificación de parámetros	Alerta emergente con mensaje de notificación
Notificación de nuevo dispositivo	Alerta de dispositivo conectado	Alerta emergente con mensaje de notificación
Notificación de desvinculación	Alerta de desvinculación de dispositivos	Alerta emergente con mensaje de notificación

4.6. Propuesta de Solución

En esta sección se toma la información descrita anteriormente para seleccionar las diferentes tecnologías que darán vida a la solución desarrollada. Para esto se debe partir analizando el estado del arte y sus aportes al proyecto.

4.6.1. ¿Como afecta lo estudiado en el estado del arte en nuestra solución?

De los documentos estudiados se pudo extraer información y puntos de vista de otras soluciones similares que llevaron a la exploración de diferentes alternativas de resolución de problemas.

Del artículo [18], se observa que la experiencia del usuario y la manera de desplegar la información puede favorecer al momento de utilizar la plataforma y lograr que esta tenga coherencia con el mundo real. Es por esto que se decide dividir el despliegue de la información referente a los dispositivos del hogar en habitaciones, las cuales son creadas y configuradas por el usuario para simular los espacios de su casa.

En [13] se destaca la arquitectura utilizada y el uso de la Raspberry Pi como dispositivo central de administración.

En [14] se da una mirada a las distintas opciones disponibles de bases de datos para aplicaciones de IoT.

De [17] y [9] se rescataron funciones que competen al proyecto, pero que por tiempo quedaron fuera del desarrollo. Estas funciones son la predicción del consumo energético para establecer un modelo de comercio en donde el usuario pueda saber cuando le conviene vender el excedente de energía generado en su hogar, utilizando incluso el almacenamiento energético de su VE y sin perder la posibilidad de realizar sus labores diarias por la venta de sus recursos. Además, en [11] se estudia la diversidad de cambios que pueden afectar a la producción de energía frente a diferentes situaciones climáticas.

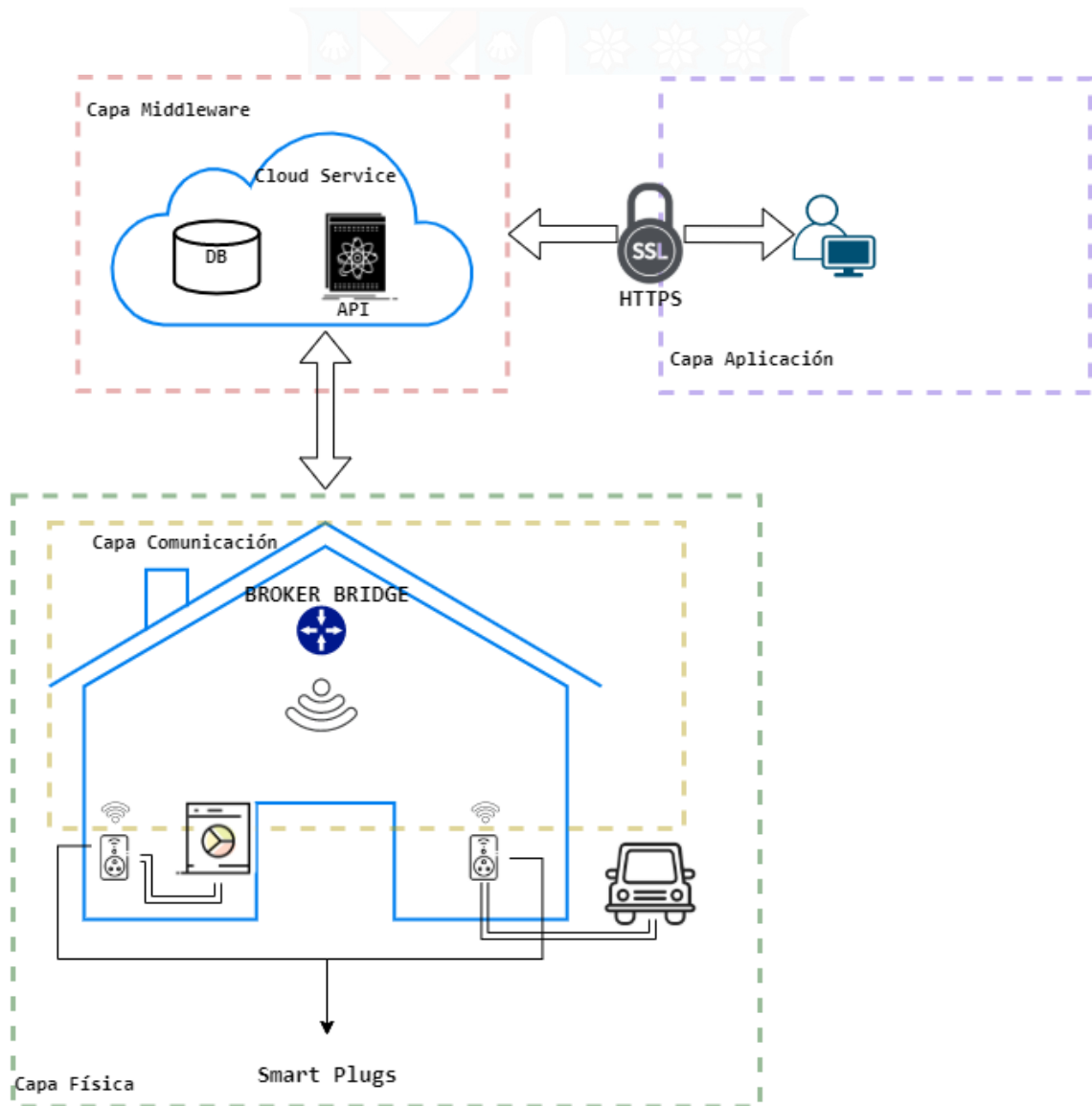
Similarmente a los casos anteriores, la integración de un panel fotovoltaico al sistema, y la implementación de un algoritmo para la utilización de la fuente de energía óptima, propuesta en [12], quedó fuera del alcance de este proyecto.

Por último, el stack de tecnologías usadas por [15] para la implementación de su sistema resulta un punto base de comparación contra las elecciones para este proyecto.

4.6.2. Arquitectura del Sistema

Con todo lo mencionado anteriormente y sumado al Capítulo 4 se realizó en la Figura 4.2 la arquitectura propuesta para comprender de manera gráfica los componentes de la solución propuesta. Es necesario mencionar que el diagrama presentado esta dividido por las capas descritas con anterioridad.

El sistema se compone de 3 entidades principales: el usuario, que espera visualizar gráficos de consumo; el servicio en la nube, que recibe los datos telemétricos de los dispositivos del hogar, los almacena en una base de datos y disponibiliza mediante una API; y los electrodomésticos, los que están conectados directamente a un smart plug, enviando la información a un nodo central llamado gateway. Este último tiene la función de reenviar la información recibida al servicio en la nube. Estas entidades se representan en la Figura 4.2 como el usuario, la nube y la casa respectivamente. En esta última se encuentran smart plugs y el gateway.

**Figura 4.2:** Arquitectura de componentes propuesta

5 | Hardware

En esta sección se evaluó las tecnologías que permiten el desarrollo de la capa física y de comunicación. Luego, se divide en diferentes apartados en donde se analizó la elección de tecnologías y se mostró la implementación de ellas. En específico, se da a conocer los dispositivos físicos necesarios para la toma de datos y la transmisión de ellos por la red del hogar.

5.1. Capa Física

Es necesario destacar que la funcionalidad de este proyecto requiere de dos piezas de hardware importantes. La primera es el uso de smart plugs, los cuales conectan los dispositivos electrónicos a la toma de corriente y son capaces de monitorear periódicamente las cantidades de energía que estos consumen. La segunda pieza es un gateway, que sirve de intermediario entre los sensores y el servidor principal en la nube.

5.1.1. Smart Plug

Con enfoque en los smart plugs, se enfrentó la problemática de un error de conceptos, ya que “smart plug” son aquellos nodos, ubicados entre la toma de corriente y un electrodoméstico, que permiten medir el consumo energético, pero muchos proveedores confunden este concepto con el de Smart Switch, que es un enchufe que permite el control de estados -ON y OFF- de los dispositivos conectados a él. Es por esto que se elaboró una comparativa de diferentes smart plugs que existen en el mercado y permiten el monitoreo del consumo energético, junto con una tabla de sus especificaciones.

5.1.1.1. Shelly Plug

Este smart plug permite a los usuarios controlar los dispositivos eléctricos conectados a él, además de monitorear su consumo y permite programar horarios de encendido y apagado de ellos. Este funciona con la tecnología inalámbrica Wi-Fi conectado a la HAN o puede funcionar como Access Point.



Figura 5.1: Shelly Plug

Tabla 5.1: Características Shelly Plug.

Característica	Valor
Alimentación AC	110-230 [V]
Protección de Sobrecarga	Sí
Medición de Potencia	Sí
Carga máxima	16 [A]
Consumo de potencia del dispositivo	<1 [W]
Encendido Inteligente	Sí
Control Local y Remoto	Sí
Conectividad Wireless	Sí
Protocolo Wi-Fi	802.11 b/g/n
Frecuencia Wi-Fi	2.4 [GHz]
Dimensiones	85mm x 57mm x 98mm

5.1.1.2. Shelly Plug S

El Shelly Plug S es la versión reducida del smart plug mencionado anteriormente. Este realiza las mismas funciones que su versión anterior y funciona con Wi-Fi conectado a la HAN o puede funcionar como Access Point.



Figura 5.2: Shelly Plug S

Tabla 5.2: Características Shelly Plug.

Característica	Valor
Alimentación AC	110-230 [V]
Protección de Sobrecarga	Sí
Medición de Potencia	Sí
Carga máxima	12 [A]
Consumo de potencia del dispositivo	<1 [W]
Encendido Inteligente	Sí
Control Local y Remoto	Sí
Conectividad Wireless	Sí
Protocolo Wi-Fi	802.11 b/g/n
Frecuencia Wi-Fi	2.4 [GHz]
Dimensiones	46mm x 69mm

5.1.1.3. TP-Link HS110

El TP-Link HS110 es un smart plug Wi-Fi que se caracteriza por:

- **Acceso Remoto:** Permite controlar los dispositivos conectados a él desde cualquier parte del mundo por medio de internet a través de una aplicación móvil gratuita.
- **Programación:** Se programa el smart plug para que automáticamente encienda o apague la alimentación de los electrodomésticos según se necesite.
- **Monitorización de Energía:** Analiza el consumo de potencia a tiempo real.
- **Modo Fuera de Casa:** Enciende y apaga tus dispositivos en diferentes horarios para dar la apariencia de que alguien está en casa cuando sea necesario.
- **Control de Voz por Google Home:** Permite la integración con asistentes de Google Home para el manejo de electrodomésticos por voz.



Figura 5.3: TP-Link HS110

Tabla 5.3: Características TP-Link HS110.

Característica	Valor
Alimentación AC	100-240 [V]
Protección de Sobrecarga	No especifica
Medición de Potencia	Sí
Carga máxima	13 [A]
Carga máxima de Potencia	3.68 [KW]
Encendido Inteligente	Sí
Control Local y Remoto	Sí
Conectividad Wireless	Sí
Protocolo Wi-Fi	802.11 b/g/n
Frecuencia Wi-Fi	2.4 [GHz]

Dimensiones	100.3mm x 66.3mm x 77mm
-------------	-------------------------

5.1.1.4. Develco Smart Plug Mini

Es el smart plug más pequeño del mercado que mide el consumo de los dispositivos con alta precisión y se enfoca principalmente en medidas de seguridad para el usuario, controlando la temperatura y la corriente con el fin de generar cortes automáticos en caso de sobre consumo o alerta. Además, permite el encendido y apagado a distancia, y funciona con el protocolo Zigbee.

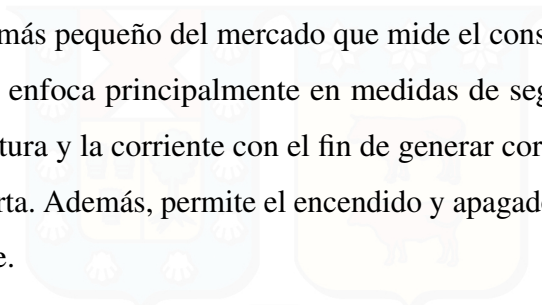


Figura 5.4: Develco Smart Plug Mini.

Tabla 5.4: Características Develco Smart Plug Mini.

Característica	Valor
Alimentación AC	230 [V]
Protección de Sobrecarga y sobrecalentamiento	Sí
Medición de Potencia	Sí
Carga máxima	16 [A]
Encendido Inteligente	Sí
Control Local y Remoto	Sí
Conectividad Wireless	Sí
Protocolo	Zigbee
Dimensiones	41mm x 45mm

5.1.1.5. Sonoff Pow R2

Sonoff Pow R2 es un interruptor inteligente WiFi, que permite administrar y controlar los dispositivos de forma remota y controlar el consumo de energía de su hogar. Este funciona como un monitor de alimentación, que realiza el seguimiento del 99 % de la corriente, el voltaje y la potencia en tiempo real desde una aplicación móvil. Además, puede proteger los electrodomésticos contra sobrecargas estableciendo previamente el valores de umbral para la corriente, el voltaje y la potencia. Por último, funciona como una calculadora de [KWh] que puede medir el consumo de electricidad durante un período específico y un día.



Figura 5.5: SonOff Pow R2

Tabla 5.5: Características Sonoff Pow R2.

Característica	Valor
Alimentación AC	90-250 [V]
Protección de Sobrecarga y sobrecalentamiento	Sí
Medición de Potencia	Sí
Carga máxima	16 [A]
Carga máxima de Potencia	3.5 [KW]
Encendido Inteligente	Sí
Control Local y Remoto	Sí
Conectividad Wireless	Sí
Protocolo Wi-Fi	802.11 b/g/n
Frecuencia Wi-Fi	2.4 [GHz]
Dimensiones	114mm x 52mm x 32mm

En la Tabla 5.6 se hace la recopilación de los smart plugs estudiados anteriormente, comparando los apartados más importantes para el desarrollo de la solución, los que son: monitoreo energético, tecnología de comunicación y precio.

Tabla 5.6: Estudio de mercado Smart Plugs.

Marca/ Modelo	Monitoreo Energético	Tecnología de Comunicación	Precio
Shelly Plug	✓	Wi-Fi	25.370 CLP
Shelly Plug S	✓	Wi-Fi	15.576 CLP
TP-Link HS110	✓	Wi-Fi	14.734 CLP
Develco Smart Plug Mini	✓	Zigbee	42.275 CLP
Sonoff Pow R2	✓	Wi-Fi	14.990 CLP

Elección de Smart Plug

Analizando las diferentes opciones se eligió el smart plug Sonoff Pow R2 -hardware basado en ESP8266- para el prototipo frente a las demás opciones debido a que:

- El acceso a la mayoría de las opciones depende de un largo periodo de espera, ya que es necesario comprarlos al extranjero.
- Los modelos que asemejan enchufes no utilizan el tipo de enchufe de Chile, por lo que se necesitarían adaptadores.
- Sonoff Pow R2 se encuentra en el país y su precio es uno de los más bajos.

Sin embargo, es necesario abrir un apartado extra dentro del análisis, ya que, aunque es posible utilizar los smart plug Sonoff Pow R2 con su firmware de fábrica, instalar un firmware open source otorga mayor control sobre el hardware y permite obtener beneficios como conectividad con asistentes virtuales, broker MQTT e incluso manejar el tiempo de las mediciones.

5.1.2. Firmware

Como se mencionó anteriormente fue necesario evaluar diferentes firmwares para hacer una elección apropiada. Es necesario destacar que el smart plug Sonoff Pow R2 esta

basado en el chipset ESP8266, por lo que se buscó firmwares con eso en mente.

Se compararon los siguientes firmware:

- Tasmota
- ESPurna
- ESPEasy
- ESPHome

5.1.2.1. Tasmota

Tasmota es un firmware open source diseñado para los chipsets: ESP8255, ESP32, ESP32-S y ESP32-C3 que proporciona un control local de los dispositivos inteligentes a través de MQTT, HTTP, Serial, o WEB UI. Este firmware es una gran herramienta que permite el control casi absoluto de los dispositivos basados en los chipsets mencionados anteriormente.

Como se muestra en la Figura 5.6, mediante la interfaz gráfica de Tasmota es posible configurar: conexión Wi-Fi, conexión con broker MQTT, Domoticz, cambiar la contraseña, entre otras cosas.

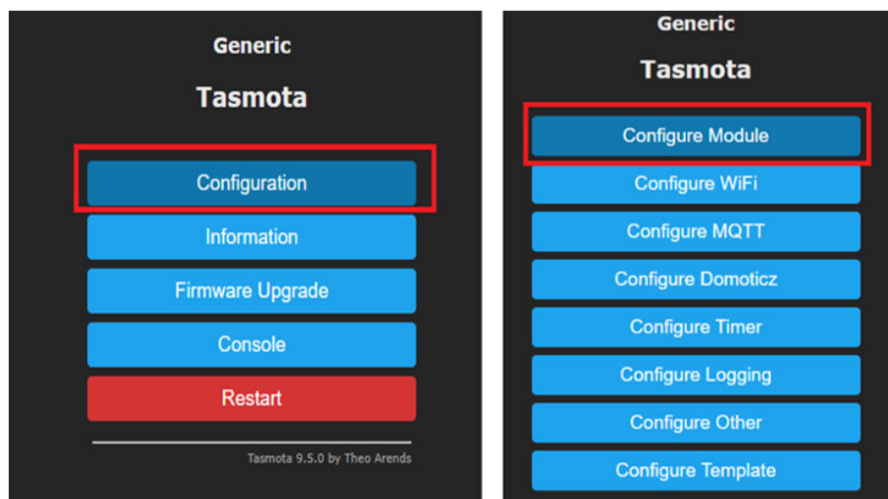


Figura 5.6: Firmware Tasmota.

5.1.2.2. ESPurna

En segundo lugar se encuentra ESPurna, un firmware open source diseñado para los chipset ESP8266 y ESP32. Este se encarga de proveer mayor control sobre los dispositivos como se muestra en [Anexo Figura 1]. En ella se puede observar las diferentes configuraciones posibles, dentro de las que se encuentran: configuración de domoticz, Home Assistant, broker MQTT, tiempo de muestreo de consumo, configuración de Thingspeak, configuración Wi-Fi, cambiar contraseña e incluso trae una consola de debug. Esto la vuelve una herramienta sumamente completa y versátil, otorgando a los dispositivos un control total de sus configuraciones y sumándole funcionalidades para robustecer el funcionamiento de los smart plugs.

5.1.2.3. ESPEasy

ESPEasy es un firmware open source diseñado para los chipset: ESP8266, ESP8285 y ESP32. Este se basa en una configuración completa mediante un servidor web que se levanta al flashear el dispositivo, la que permite configuraciones básicas como Wi-Fi y contraseña, además de inclusión de Domoticz y añadir un broker MQTT. Su interfaz se muestra en Figura 5.7.

ESP Easy Mega: Sonoff POW R1								
Main Config Controllers Hardware Devices Rules Tools								
	Task	Enabled	Device	Name	Port	Ctr (IDX)	GPIO	Values
Edit	1	✓	Switch input - Switch	Button1			GPIO-0 (D3)	State: 0
Edit	2	✓	Switch input - Switch	Relay1		1 (475)	GPIO-12 (D6)	State: 0
Edit	3	✓	Energy (AC) - HLW8012/BL0937 [TESTING]	HLW8012		1 (248)	GPIO-5 (D1) GPIO-13 (D7) GPIO-14 (D5)	Voltage: 230.00 Current: 0.00 Power: 0.00 PowerFactor: 0.00
Edit	4	✓	Output - Domoticz MQTT Helper	domoticz			GPIO-12 (D6)	Output: 0

Figura 5.7: Firmware ESPEasy.

5.1.2.4. ESPHome

Por último ESPHome esta diseñado para los chipset ESP8266 y ESP32. Este utiliza una interfaz web para su configuración y permite configuraciones básicas de los dispositivos

para integración en un hogar inteligente por medio de un Home Assistant.

A continuación se muestra la Tabla 5.7 donde se analizó la plataforma que soporta su instalación, los protocolos de comunicación a los que está enfocado, la calificación de su plataforma web y la cantidad de descargas -que nos permiten saber que tan grande es su comunidad-.

Tabla 5.7: Comparación de Firmwares.

	Plataforma soportada	Protocolo Principal	Web UI Rate	Descargas
Tasmota	ESP8255, ESP32, ESP32-S y ESP32-C3	MQTT, Telnet, HTTP/S, KNX	1/3	5.7 M [23]
ESPurna	ESP8266 y ESP 32	MQTT, Rest over HTTP/S, Telnet	3/3	150K [24]
ESPeasy	ESP8266, ESP8285 y ESP32	MQTT, HTTP/S, LoRA/TTN	1/3	677K [25]
ESPHome	ESP8266 y ESP 32	MQTT, Rest over HTTP/S, HomeAssistant Native API	2/3	No aplica

Esta información fue extraída de [26].

Elección de Firmware

Comparando cada una de las alternativas se apreció que los diferentes firmwares presentaban características similares, sobresaliente Tasmota y ESPurna. Es por esto que la elección se hizo en base a la documentación y resolución de errores. Con esto se eligió ESPurna, porque a pesar de no tener una gran documentación, es un firmware de fácil instalación, plataforma intuitiva y por último, la mayor parte de los errores comunes están resueltos por la comunidad, la cual está continuamente resolviendo nuevos problemas y errores. Sin mencionar la gran cantidad de características de configuración que le añade al smart plug.

5.1.3. Gateway

Para la elección del gateway se hizo una comparación de 3 tecnologías que se utilizan para cumplir esta función -Jetson Nano, Raspberry Pi y NodeMCU- y sus características

-componentes, precio, etc- las cuales definen la elección.

5.1.3.1. Jetson Nano

Es un pequeño y poderoso computador que permite ejecutar sistemas operativos basados en Linux -como Ubuntu-. Esto lo transforma de manera automática en una tecnología versátil y adaptable a las necesidades del usuario. Además, tiene soporte para sensores de alta resolución, pudiendo controlar gran cantidad de sensores en paralelo. Cabe mencionar que está pensado en soporte de inteligencia artificial, de ahí sus enormes capacidades [27]. El modelo que se comparó es el modelo B01. [Figura 5.8]



Figura 5.8: Jetson Nano

Tabla 5.8: Características Jetson Nano.

Característica	Jetson Nano
CPU	ARM® A57 de cuatro núcleos
GPU	NVIDIA Maxwell de 128 núcleos
RAM	4 GB de LPDDR4 de 64 bits
Almacenamiento	MicroSD no incluida
GPIO	40 pines
Conectividad	Gigabit Ethernet 10/100/1000
USB	4x USB 3.0
Alimentación	MicroUSB 5V 2A
Precio	\$157.490

5.1.3.2. Raspberry Pi

Esta empresa se enfocó en la creación de micro-computadores que permiten desarrollar proyectos de diferente índole.

Al igual que la tecnología anterior, las Raspberry Pi soportan sistemas operativos basados en Linux, siendo Raspian una distribución diseñada específicamente para ellas.

De entre todos los modelos de la marca, se consideró el modelo 3B+. [Figura 5.9]



Figura 5.9: Raspberry Pi 3B+

Tabla 5.9: Características Raspberry Pi.

Característica	Raspberry Pi 3B+
CPU	Broadcom BCM2837B0
GPU	Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
RAM	1GB LPDDR2 SDRAM
Almacenamiento	MicroSD no incluida
GPIO	40 pines
Conectividad	Wi-Fi + Bluetooth: 2.4GHz y 5GHz IEEE 802.11.b/g/n/ac
USB	4x USB 2.0
Alimentación	MicroUSB 5V 2A
Precio	\$ 45.990

5.1.3.3. NodeMCU

Es una plataforma IoT que incluye un firmware diseñado para el microcontrolador con modulo Wi-Fi ESP8266. El modelo analizado es el NodeMCU V2, porque es fácil de

programar y ensamblar en conjunto con sensores, protoboards y realizar prototipos IoT de diferente índole. [Figura 5.10]

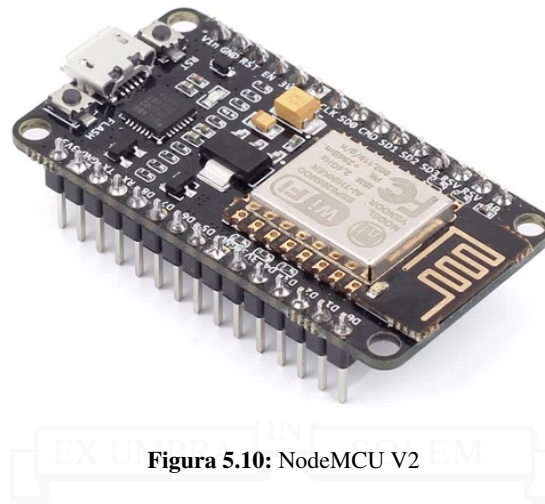


Figura 5.10: NodeMCU V2

Tabla 5.10: Características NodeMCU.

Característica	NodeMCU V2
CPU	x
GPU	x
RAM	96 KB
Almacenamiento	4MB
Conectividad	Wi-fi: 802.11 b/g/n
GPIO	18 pines
USB	x
Alimentación	MicroUSB 5V 2A
Precio	\$ 4.990

En la Tabla 5.11 se generó una tabla recopilatoria de las características con las que se comparan las diferentes opciones de gateways.

Tabla 5.11: Comparación de Gateways.

Característica	Jetson Nano	Raspberry Pi 3B+	NodeMCU V2
CPU	ARM® A57 de cuatro núcleos	Broadcom BCM2837B0	x
GPU	NVIDIA Maxwell de 128 núcleos	Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz	x
RAM	4 GB de LPDDR4 de 64 bits	1GB LPDDR2 SDRAM	96 KB

Almacenamiento	MicroSD no incluida	MicroSD no incluida	4MB
GPIO	40 pines	40 pines	18 pines
Conectividad	Gigabit Ethernet 10/100/1000	Wi-Fi + Bluetooth: 2.4GHz y 5GHz IEEE 802.11.b/g/n/ac, Bluetooth 4.2, BLE. Ethernet: Giga- bit Ethernet sobre USB 2.0 (300 Mbps)	Wifi: 802.11 b/g/n
USB	4x USB 3.0	4x USB 2.0	x
Alimentación	MicroUSB 5V 2A	MicroUSB 5V 2A	MicroUSB 5V
Precio	\$157.490 *	\$ 45.990 *	\$4.990 *

Elección de Gateway

Como gateway se eligió la Raspberry Pi modelo 3B+. La elección se hizo basado en su versatilidad, sus características, su precio y porque se adaptó de mejor manera a la solución planteada. Cabe mencionar que la Jetson Nano es un mini computador demasiado potente -y de alto costo- que no se adapta al proyecto por sobredimensionar las características necesarias y por otro lado, el NodeMCU V2 esta infradimensionado para cumplir las necesidades requeridas.

5.2. Capa de Comunicación

Para la capa de comunicación fue necesario comparar las diferentes tecnologías inalámbricas que se pueden usar en el hogar, y cuales se adaptan a las necesidades del desafío.

5.2.1. Tecnologías

En [28; 29] se describen las tecnologías usadas en HAN -Home Area Network-, y se puede observar en la Tabla 5.12 las características de ellas. Basados en estos artículos se realizó una comparativa de las principales tecnologías utilizadas para IoT en el hogar.

Tabla 5.12: Características de tecnologías inalámbricas usados en HAN.

Característica	Wi-Fi	Zigbee	Z-Wave	BLE	LoRaWAN
IEEE Standard	802.11	802.15.4	N/A	802.15.1	N/A
Banda de Frecuencia	2.4 GHz, 5 GHz	2.4 GHz	sub-GHz	2.4 GHz	sub-GHz
Rango Nominal	LAN <300 m	LAN <100m	LAN <100m	LAN <100m	WAN 5 km
Throughput	1 Gbps	250 Mbps	40 Kbps	100 Mbps	50 Kbps
Topología de Red	Star, Mesh	Star, Tree, Mesh	Mesh	Star, Mesh	Celular
Número de Nodos Máximos	250/access point	64.000	232	32.000	104 / BS
Encriptación	WPA and WPA2	AES-CCM	AES-CCM	AES-CCM	AES

5.2.1.1. Wi-Fi

El Wi-Fi es ahora una de las principales formas de comunicación para muchos dispositivos y, con el aumento de la automatización del hogar, son aún más los dispositivos que lo utilizan. El Wi-Fi doméstico es un gran ámbito de uso de la tecnología, ya que la mayoría de los hogares que utilizan conexiones de banda ancha a Internet emplean el acceso Wi-Fi como medio clave de comunicación [30]. Además, es una tecnología versátil del cual desprenden diferentes tecnologías como 802.11b, 802.11g, 802.11ac, entre muchos más. Cada uno de ellos esta pensado en situaciones diferentes, volviéndolo adaptable a las necesidades.

5.2.1.2. Zigbee

ZigBee es una tecnología mesh inalámbrica de baja potencia. Utiliza frecuencias basadas en el estándar IEEE 802.15.4 para HAN centradas en aplicaciones de supervisión, control y sensores. Funciona principalmente en la banda de 2,4 GHz y está impulsando el estándar para dispositivos domésticos, desde sistemas de temperatura e iluminación hasta monitores de seguridad y detectores de humo.

5.2.1.3. Z-Wave

Es una tecnología de comunicación inalámbrica de baja potencia para la automatización del hogar, específicamente para aplicaciones de control remoto en entornos residenciales y comerciales ligeros. Se utiliza en una amplia gama de productos de consumo en todo el mundo. Entre ellos se encuentran en home theater, la automatización de ventanas, el control de piscinas y spa y la lectura automática de medidores.

5.2.1.4. BLE

Bluetooth Low Energy es un estándar que pretende ser una forma segura y barata de conectar y transferir datos entre dispositivos compatibles. Este reduce significativamente el consumo de energía de los dispositivos Bluetooth y permite un funcionamiento a largo plazo utilizando baterías. BLE ofrece una infraestructura de conexión directa desde smartphones y tabletas, lo que permite a los usuarios controlar los electrodomésticos desde sus dispositivos móviles.

5.2.1.5. LoRaWAN

LoRaWAN es una tecnología de bajo consumo para WAN que busca comunicar dispositivos a larga distancia. Este protocolo ha tomado fuerza durante el último período de tiempo, llegando a utilizarse incluso dentro de los hogares, comunicando los dispositivos del hogar por medio de él.

Al ser un protocolo de bajo consumo es principalmente utilizado en IoT para sensores, ya que prolonga el tiempo de vida de las baterías, siendo utilizado por dispositivos de medición de: temperatura, humedad, humo, etc.

Elección de tecnología de comunicación

A partir de lo anterior, se eligió Wi-Fi como tecnología de comunicación para la transmisión de información, por: sus rangos nominales de funcionamiento y la cantidad de nodos que permiten en la red, además de ser la más utilizadas actualmente, la más versátil, adaptable a las necesidades y las que tienen mayor soporte por la comunidad -tutoriales, configuraciones, etc-.

5.2.2. Protocolos

Si se observa con detención la conectividad entre los smart plugs y el gateway entra en la categoría de comunicación Machine-to-Machine (M2M). Esta se puede categorizar en 2 patrones: basada en eventos y basada en sondeo. El primero sigue un modelo de comunicación request-response. Por otro lado el segundo patron se basa en el modelo publisher-subscriber.

Este último no requiere la interacción sin estado entre cliente y servidor, por lo que utilizar una arquitectura REST. Además, este utiliza operaciones CRUD (Create, Read, Update, Delete) para manipular los recursos. Estas operaciones pueden manipular cualquier recurso y, por tanto, una misma arquitectura puede ser utilizada por varias aplicaciones.

Con todo lo anterior se vuelve necesario comparar los diferentes protocolos utilizados para este tipo de comunicación. Dentro de los más utilizados se encuentran CoAP, HTTP y MQTT, pero también existen otros como AMQP y XMPP. A continuación se hace una comparativa de los protocolos más utilizados basada en [31].

5.2.2.1. HTTP - Hyper Text Transfer Protocol

Este protocolo está diseñado para dispositivos de internet sin limitaciones de recursos.

Es uno de los primeros protocolos de transmisión creados y su modelo de comunicación es request-response.

Utiliza el protocolo de transporte TCP, garantizando la entrega y recepción de mensajes. Por lo que puede enviar grandes cantidades de datos asegurando su llegada. En IoT es especialmente usado si se desea enviar videos, imágenes o archivos.

Al trabajar con el modelo cliente-servidor, este último necesita de una petición por parte del cliente para entregar la información requerida.

5.2.2.2. CoAP - Constrained Application Protocol

CoAP es un protocolo ligero que se ajusta al paradigma REST, y está diseñado para su uso en redes y nodos con recursos restringidos dentro de aplicaciones M2M.

Al igual que en HTTP, los clientes realizan operaciones sobre recursos almacenados en un servidor mediante intercambios de peticiones y respuestas.

Este protocolo soporta las transacciones asíncronas, debido a que está separado en dos subcapas: una capa de mensajería que proporciona servicios de mensajería asíncrona sobre el transporte de datagramas, y una capa de petición-respuesta que proporciona el manejo y seguimiento de las peticiones y respuestas intercambiadas entre un cliente y servidor.

La capa de mensajería utiliza el modelo publisher-subscriber, volviendolo un protocolo con la posibilidad de usar ambos modelos de comunicación.

Al ser un protocolo diseñado para dispositivos con recursos limitados, el protocolo de transporte que utiliza es UDP (User Datagram Protocol), debido a que tiene menor sobrecarga de protocolo que TCP y puede reducir su cabecera a 4 bytes.

CoAP proporciona dos tipos de entregas de mensajes en modo pub-sub. El primero es un mensaje “Confirmable (CON)”, donde se reenvía el mensaje en caso de no recibir el acuso de recibo (ACK). El otro es un mensaje “Non-Confirmable (NON)”, que no necesita acuso de recibo (ACK). Además, hay otros dos tipos de mensajes adicionales: ACK y RST.

El mensaje ACK se utiliza para acusar recibo de los mensajes CON, y el mensaje RST notifica al otro punto final que se ha recibido un mensaje CON pero que falta algún contexto, o se utiliza para cancelar las suscripciones.

5.2.2.3. MQTT

Es un protocolo ligero de mensajería pub-sub basado en un broker y diseñado para ser abierto, simple, ligero y fácil de implementar.

Las características de MQTT lo hacen ideal para su uso en entornos restringidos,

donde, por ejemplo: la red es cara, tiene poco ancho de banda o es poco fiable, o cuando se ejecuta en dispositivos que tienen capacidades de procesamiento o memoria limitadas.

Este es un protocolo asíncrono y posee 14 tipos de mensajes diferentes definidos en MQTT, incluyendo CONNECT, PUBLISH, SUBSCRIBE, UNSUBSCRIBE, DISCONNECT, y PINGREQ y PINGRESP. Varios de estos mensajes tienen mensajes de acuse de recibo dedicados. Por ejemplo, si un suscriptor MQTT quiere suscribirse a un tema, envía un mensaje SUBSCRIBE y espera la correspondiente respuesta SUBACK del broker (servidor).

Aunque el uso de MQTT implica el uso de un protocolo de transporte orientado a la conexión fiable, como TCP, MQTT soporta tres tipos de fiabilidad de la aplicación para la entrega de mensajes de publicación:

- **Nivel 2:** Entrega el mensaje exactamente una vez, lo que garantiza que los mensajes llegan exactamente una vez.
- **Nivel 1:** Entrega el mensaje al menos una vez, lo que garantiza que los mensajes llegan, pero pueden producirse duplicados.
- **Nivel 0:** Entrega el mensaje como máximo una vez, en el que los mensajes llegan según los mejores esfuerzos de la red TCP/IP subyacente, lo que significa que pueden producirse pérdidas o duplicados de mensajes, introducidos por el software u otras causas.

Además MQTT tiene una extensión para redes de sensores: MQTT-S.

MQTT-S está optimizado para la implementación en dispositivos de bajo costo, que funcionan con baterías y con recursos de procesamiento y almacenamiento aún más limitados, como los dispositivos de sensores inalámbricos. Mientras que MQTT se basa en la pila TCP/IP, MQTT-S funciona con cualquier tecnología de red que proporcione un servicio de datagramas, como UDP. El objetivo de MQTT-S es minimizar el ancho de banda de la red y los requisitos de recursos de los dispositivos, al tiempo que se busca la fiabilidad. Además, se puede integrar una pasarela MQTT-S en el broker para traducir entre MQTT-S y MQTT.

Para concluir, en la Tabla 5.13 se hizo una comparativa entre las principales caracte-

rísticas de los protocolos comparados para posteriormente realizar su elección.

Tabla 5.13: Comparación entre HTTP, CoAP y MQTT.

Característica	HTTP	CoAP	MQTT
Modelo de comunicación	req-res	req-res o pub-sub	pub-sub
RESTful	Sí	Sí	No
Protocolo de la capa de transporte	TCP	Preferiblemente UDP; puede usar TCP	Preferiblemente TCP; puede usar UDP (MQTT-S)
Header size	8 Kbytes	4 Bytes	2 Bytes
Mensajería	Síncrona	Síncrona y Asíncrona	Asíncrona
Aplicación	Diseñado para dispositivos con recursos no limitados	Diseñado para dispositivos de red con recursos limitados como WSN / IoT / M2M	Diseñado para dispositivos con capacidad de procesamiento o memoria limitada

Elección de Protocolo

Basado en la comparativa anterior, se eligió MQTT como protocolo de transporte. Esta elección se hizo en base a su tamaño, ya que es ligero y fácil de implementar, en comparación con CoAP que tiene un mayor nivel de complejidad al implementar. Además los niveles de confiabilidad al momento de la entrega de mensajes soportada por MQTT entrega una mayor versatilidad del protocolo.

Por último, se eligió Mosquitto como servidor broker para prueba de concepto en comparación con otros broker como RabbitMQ o EMQ, que son más robustos, pero pensados en un nivel de producción.

5.2.3. Modelo de Comunicación

Por último, es necesario conocer el tipo de modelo de comunicación que se utilizó. Para eso se comparó el modelo request-response vs publisher-subscriber enfocado en IoT basados en [32].

5.2.3.1. Request-Response

Este es el modelo clásico de comunicación entre computadores de una red. En él, un computador o software solicita datos o servicios y un computador o software es el que responde a la solicitud entregando los datos [Figura 5.13].

Se puede pensar en este modelo como que un cliente envía un camión vacío para que se llene de datos. Por otro lado el servidor envía los datos en el camión hacia el lado del cliente como se ve en la Figura 5.11.

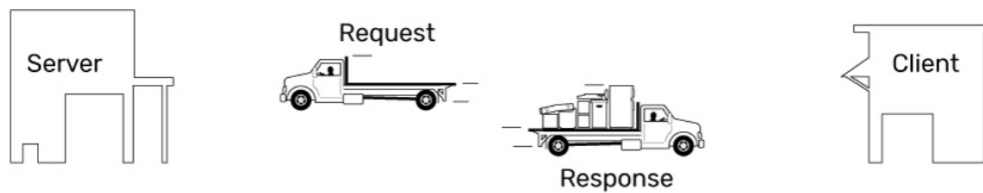


Figura 5.11: Analogía del camión Request-Response

5.2.3.2. Publisher-Subscriber

Por otro lado se encuentra el modelo pub-sub, el cual cuenta con una fuente central llamada “Broker”. Este se encarga de recibir y distribuir la información que llega a él. Los clientes en este modelo pueden enviar datos al broker o suscribirse a un tópico para obtenerlos, o ambas.

Los clientes que publican datos los envían solo cuando estos cambian. En cambio los clientes suscritos reciben automáticamente del broker los datos que recibe este último [Figura 5.14].

En la analogía del camión no hay camiones vacíos. Cuando un cliente publica datos, envía un camión lleno hacia el broker. Este ve llegar el camión, pero no lo descarga, sino que lo dirige hacia un suscriptor -clonando el camión en caso de existir más de un suscriptor-. Además se puede observar en la Figura 5.12 que el cliente de la parte inferior derecha también publica datos para otros suscriptores.

Elección de modelo de comunicación

El primer modelo funciona con múltiples servidores y múltiples clientes, lo que hace

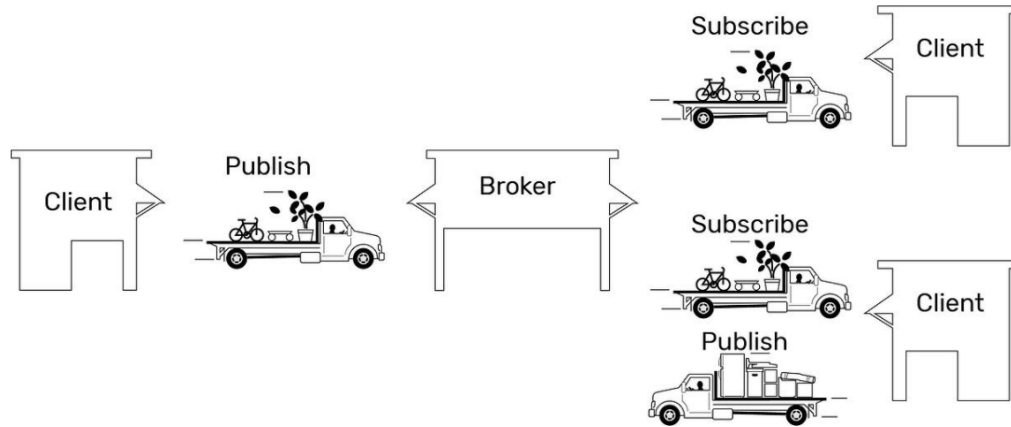


Figura 5.12: Analogía del camión Publisher-Subscriber

que el tráfico de la red sea más pesado, esto provoca que el ancho de banda demandado sea más grande.

En comparación, el segundo modelo funciona con múltiples clientes pero un solo servidor broker, el cual replica la información recibida en un tópico específico -por ejemplo: temperatura o humedad- a todos aquellos clientes que estén suscritos al tópico. Este último tiene un tráfico liviano y un ancho de banda bajo, lo que lo vuelve ideal para proyectos de IoT donde los mensajes transmitidos son pequeños y constantes. Además, mencionan a MQTT como el protocolo de transporte de información más usado en los modelos pub-sub, ya que esta pensado principalmente en la telemetría -Message Queing Telemetry Transport-. Es por esto que se utilizó el modelo pub-sub.

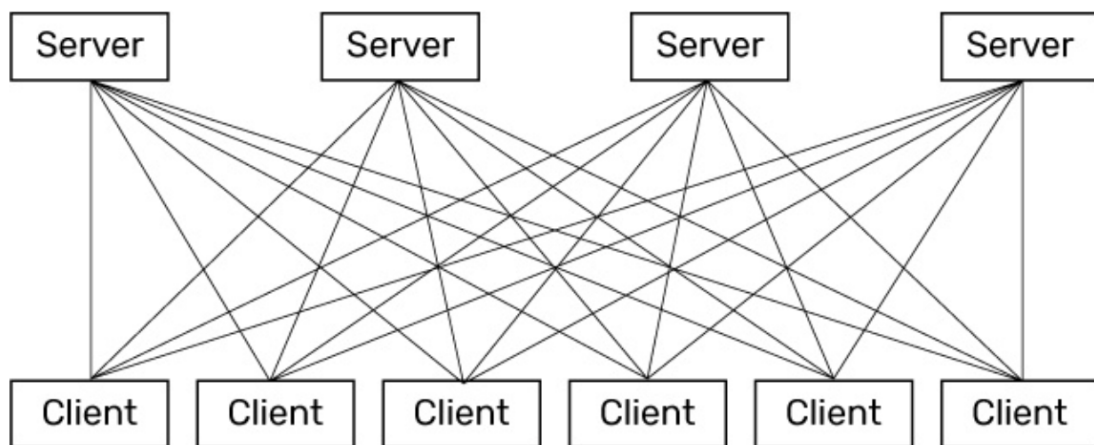


Figura 5.13: Modelo de comunicación Request-response.

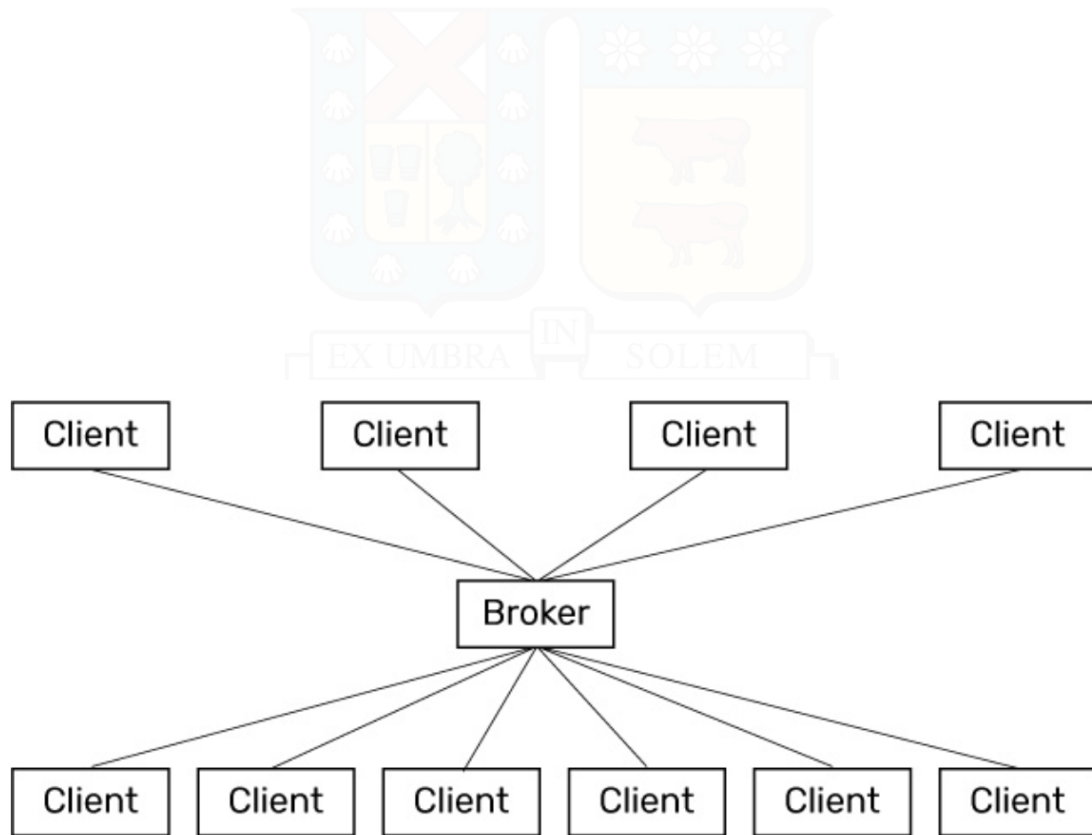


Figura 5.14: Modelo de comunicación Publisher-subscriber.

6 | Implementación

6.1. Arquitectura del Sistema

Con todo lo mencionado en Capítulo 5 se actualizó el diagrama de arquitectura para incluir las tecnologías seleccionadas para la implementación de las capas referentes a este documento. Cabe mencionar que la elección de tecnologías de backend y frontend fue realizada luego de un estudio comparativo al respecto, en donde otro miembro del equipo comparó diferentes alternativas y eligió las que se ven en la Figura 6.1.

En la siguiente lista se muestra una recopilación en detalle de todas las tecnologías utilizadas en el prototipo de solución.

- **Smart Plug:** Sonoff Pow R2
- **Gateway:** Raspberry Pi 3B+
- **Front-end:** React + Giraffe
- **Back-end:**
 - **API REST:** NodeJS + Express
 - **Base de datos telemétricos:** InfluxDB + Telegraf
 - **Base de datos usuarios:** MongoDB
- **Cifrado:** SSL

Con el objetivo de comprender la funcionalidad de cada componente del sistema se dividieron las tecnologías en módulos como se observa en Figura 6.2, los que se detallan en la Tabla 6.1

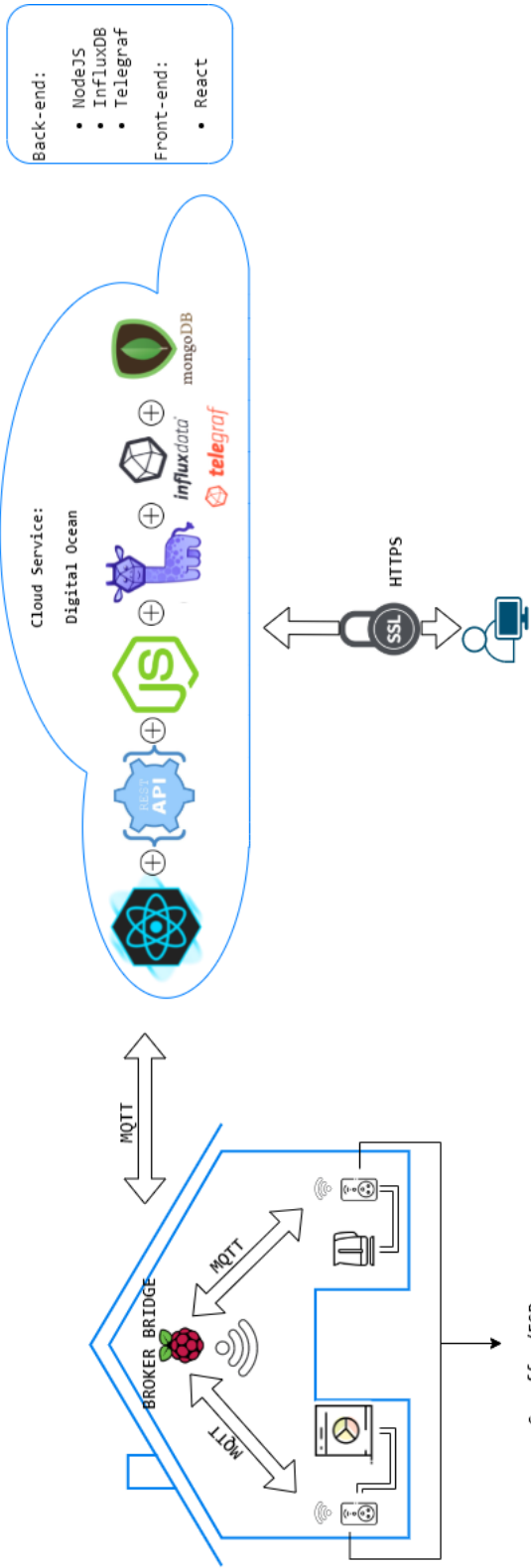


Figura 6.1: Arquitectura de componentes propuesta.

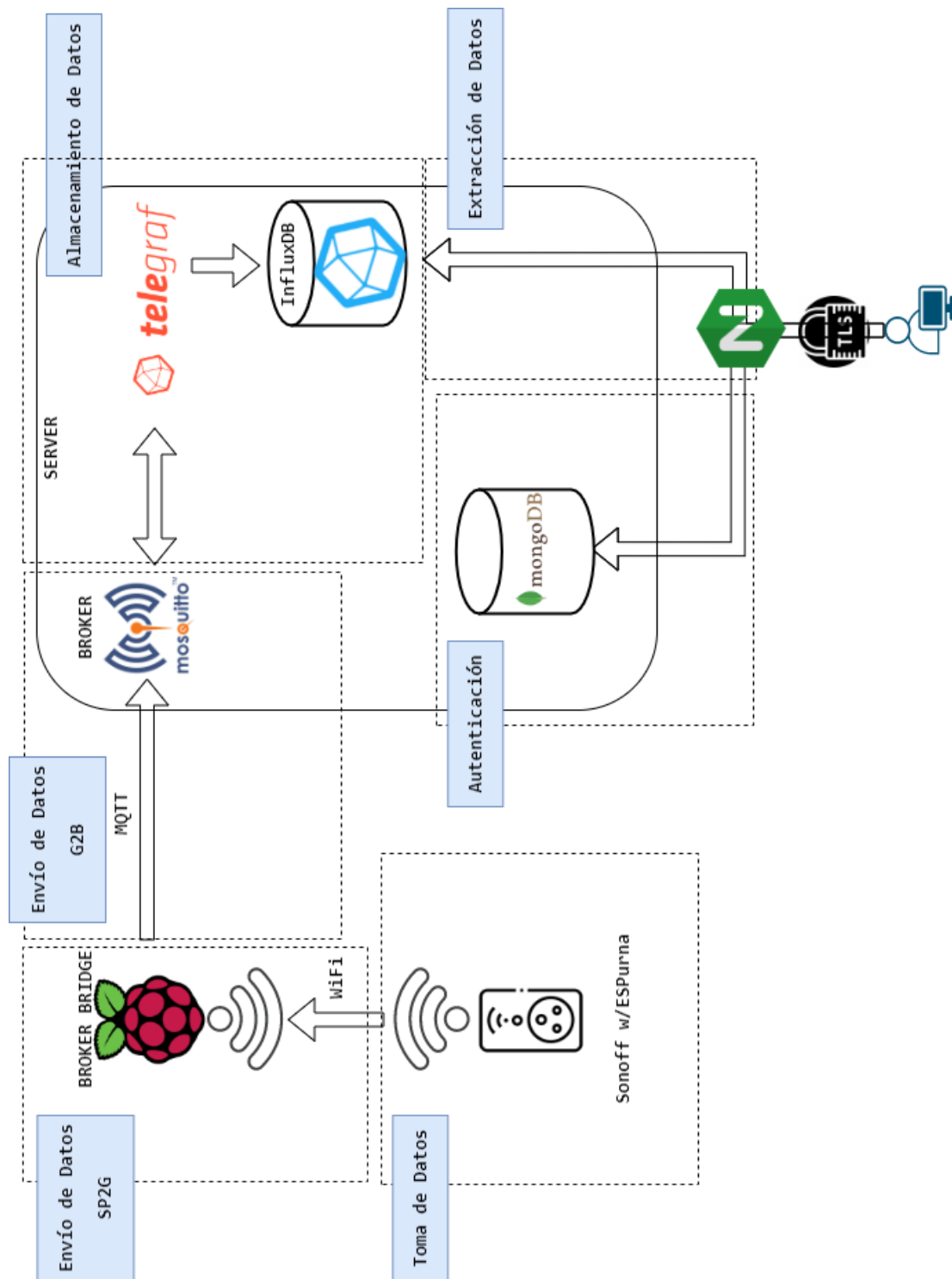


Figura 6.2: Diagrama de Flujo Módulos.

Tabla 6.1: Descripción de Módulos.

Módulo	Función
1.-Toma de datos	En este módulo se realiza la toma de datos de los electrodomésticos y se envían al gateway.
2.-Envío de datos Smart Plug a Gateway (SP2G)	Una vez los datos son recopilados, se encapsulan en un paquete para ser enviados al gateway. Este último es el encargado de otorgar conectividad segura para su configuración.
3.-Envío de datos Gateway a Broker (G2B)	Cuando el gateway recibe datos de los smart plugs se reenvían automáticamente a un servidor alojado en la nube.
4.-Almacenamiento de datos	El servidor almacena los datos en una base de datos cuando son recibidos.
5.-Autenticación	El módulo de autenticación está encargado de almacenar las credenciales de los usuarios de la plataforma y, además, de entregarle un token a los usuarios que sirve para acceder a los datos. Para esto se disponibiliza una API que permite realizar el proceso de autenticación
6.-Extracción de datos	La misma API descrita anteriormente tiene la funcionalidad de extracción de la información desde la base de datos. Para ello se utiliza el token obtenido en el módulo de autenticación, con el fin de verificar que cada usuario acceda solo a los datos que le corresponden, para el despliegue visual de ellos en la web.

6.2. Configuración de Hardware

Esta sección se divide en 2 partes. La primera se centra en el cambio de firmware de los smart plugs y la segunda en la instalación y configuración del gateway en la Raspberry Pi.

6.2.1. Smart Plugs

Con el fin de tener un mayor control sobre los smart plugs, se procede a realizar un cambio de firmware o “flashear” en cada dispositivo. El flasheo es, en palabras simples, cambiar el sistema operativo con el cual funcionan los smart plugs.

Para este caso el firmware con el cual se flashean los sonoff pow r2 es ESPurna. Se eligió este, debido a que permite obtener un gran control sobre los dispositivos, permitiendo modificar el tiempo de muestreo, su ip, la red wifi de preferencia, además de permitir conexión con asistentes del hogar y el uso de MQTT. Este firmware también tiene soporte

para una gran cantidad de dispositivos existentes en el mercado, lo que permitiría en un futuro hacer integración de dispositivos de otras marcas a la solución.

Para cambiar el firmware primero es necesario obtener la versión más actualizada de la página oficial de ESPurna:

<https://github.com/xoseperez/espurna/releases>

Una vez descargada la versión que se desea instalar, se debe descargar el programa esptool, el que permite enviar el firmware a los smart plugs mediante conexión TTL.

Esptool se puede descargar de:

<https://github.com/igrr/esptool-ck/releases>

Una vez descargado, se debe situar en un directorio que contenga tanto el programa esptool como la versión de ESPurna que desea instalar.

Paralelamente se debe conectar el cable TTL al Sonoff en el siguiente orden:

- GND-GND
- VCC-VDD
- TXD-ERX
- RXD-ETX

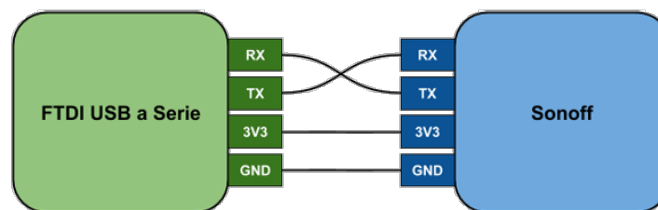


Figura 6.3: Conexión Sonoff a adaptador TTL

El resultado de la conexión se muestra en la Figura 6.4.

Y antes de conectar el cable TTL al USB del computador, se debe mantener pulsado el botón del sonoff por 2-3 segundos, para que al conectarse entre en modo de flasheo.

Luego es necesario abrir una consola de comandos y ejecutar la siguiente serie de comandos:

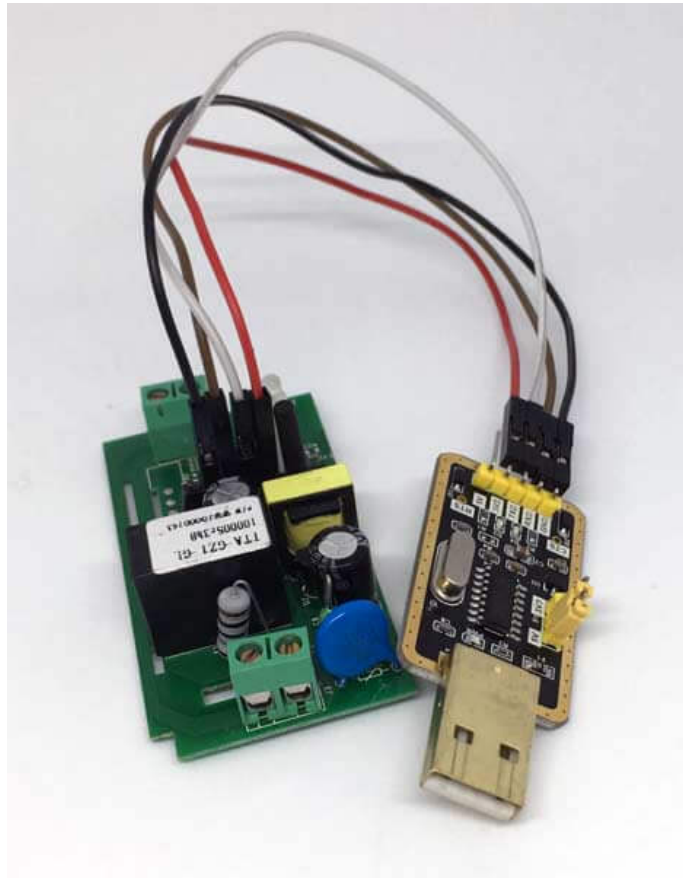


Figura 6.4: Sonoff POW R2 conectado a adaptador TTL.

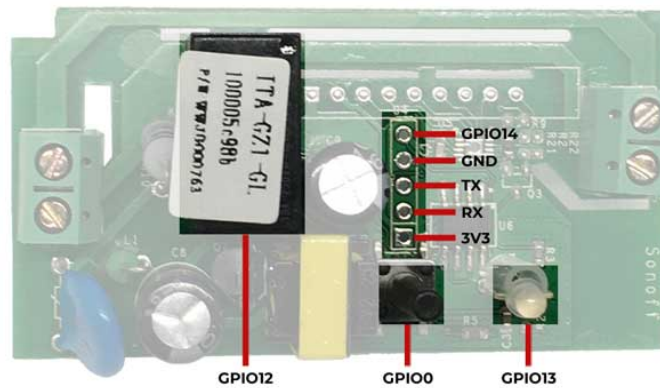


Figura 6.5: GPIO de Sonoff POW R2.

1. `esptool --port COMx --baud 115200 read_flash 0x000000 0x100000 sonoff-backup.bin`
2. `esptool --port COMx --baud 115200 erase_flash`
3. `esptool --port COMx --baud 115200 write_flash -fs 4MB -fm dout 0x000000
espurna-1.14.1-itead-sonoff-pow-r2.bin`

Cabe destacar que se debe esperar que cada comando se ejecute el tiempo necesario para ejecutar el siguiente comando.

El primer comando permite realizar un firmware de backup en caso de que algo salga mal o luego querer volver al estado de fábrica los dispositivos.

El segundo borra el firmware que posea en ese momento el smart plug, que por defecto será el de fábrica.

Y por último se procede a la escritura del firmware de ESPurna en la versión que se haya elegido, en este caso la 1.14.1.

Una vez se termina la configuración, se procede a realizar las conexiones de los smartplugs para generar el intermediario entre el tomacorriente y el electrodoméstico objetivo como se muestra en Figura 6.7 para el caso del calefactor. Esto se replicó en todos los electrodomésticos de la tabla Tabla 6.2, en una casa con 4 habitaciones: baño, cocina, dormitorio y living como se muestra en la Figura 6.6. En esta figura se representan los electrodomésticos objetivos encerrados en un círculo rojo punteado.

Tabla 6.2: Información de electrodomésticos objetivo.

Electrodoméstico	Marca	Modelo	Habitación
Lavadora	Mademsa	Evoluzione 8.5 kg lvme	Baño
Refrigerador	Mademsa	Nordik MR 415	Cocina
TV	AOC	19 pulgadas le19a1321	Dormitorio
Calefactor	BLITZ	EH30708G	Dormitorio
TV	SAMSUNG	7 series tu7090 43	Living

En este punto es necesario configurar de manera gráfica los smart plugs a través de ESPurna. Para eso es necesario conectarse a la red Wi-Fi que emiten los smart plugs con SSID similar a `ESPURNA-E2E9DE` donde `E2E9DE` son los 3 últimos octetos de la MAC que identifican al smart plug.

Una vez conectado acceder a `http://espurna-e2e9de.local/`, donde se puede acceder a la interfaz gráfica del smart plug y configurar.

Lo primero que se tuvo que hacer fue cambiar la clave por defecto, la que es `fibonacci` y el usuario es `admin`. Una vez dentro de la interfaz, se procede a configurar en

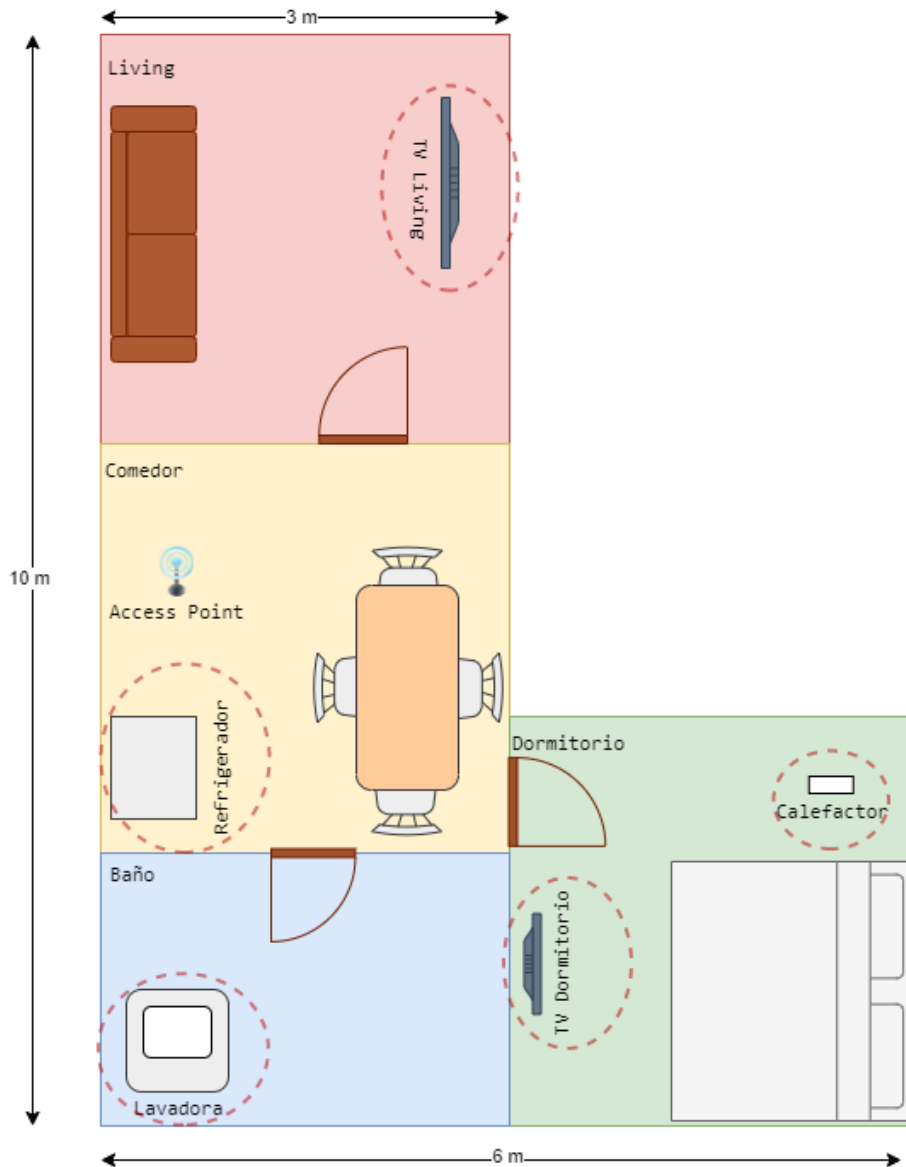


Figura 6.6: Diagrama de casa iHEMS.

la sección Wi-Fi, permitiendo al smart plug tener acceso a internet por medio del proveedor del hogar. Luego se procede a configurar la sección MQTT con: la IP del Gateway, el usuario MQTT configurado y su contraseña. Es necesario luego de cada configuración darle al botón `save`. Cabe mencionar que la configuración de MQTT se muestra en la Subsección 6.2.2

Con esto, los smart plug están configurados para enviar la información descrita en Listing 6.1 al gateway y este lo reenvía al servidor para ser almacenado.

Listing 6.1: JSON MQTT



Figura 6.7: Smart Plug conectado a Calefactor

```
{  
  "current":float  
  "voltage":int  
  "power":int  
  "reactive":int  
  "apparent":int  
  "factor":int  
  "energy":float  
  "time":string  
  "mac":string  
  "host":string  
  "ip":string  
  "id":int  
}
```

6.2.2. Gateway

Para esta sección se instaló Raspbian Lite OS en una MicroSD de 32GB. En específico se utilizó una versión sin interfaz gráfica, para disminuir el consumo de recursos de la Raspberry Pi 3B+.

Una vez insertada la microSD en la Raspberry se comienza con la configuración. Lo primero que se realiza es ejecutar en la consola los siguientes comandos:

1. `sudo raspi-config`
2. Seleccionar `Network Options`
3. Seleccionar `Wi-Fi`
4. Ingresar manualmente el `SSID` de Wi-Fi
5. Ingresar manualmente el `PASSWORD` de Wi-Fi
6. Seleccionar el botón `FINISH`

Esta serie de comandos permite configurar la interfaz Wi-fi de la Raspberry Pi para tener acceso a internet y poder descargar los programas necesarios.

El siguiente paso es instalar MQTT, pero antes de eso es necesario actualizar los paquetes de la Raspberry con los siguientes comandos:

1. `sudo apt update`
2. `sudo apt upgrade`

Y luego:

1. `sudo apt-get install mosquitto mosquitto-clients`

Una vez instalado se procede a configurar el archivo `mosquitto.conf` ubicado en `/etc/mosquitto` con el editor de texto habitual -en este caso se usó nano-. La configuración final es la que se muestra en Figura 6.8. En ella se observa que, en la línea `allow_anonymous false` que es necesario tener un usuario registrado para enviar información y en `password_file etc/mosquitto/pwfile` se entrega la ruta al archivo que almacena las contraseñas encriptadas de los usuarios.

Otra de las observaciones es que en la línea `connection bridge-02` se configura el broker MQTT del gateway como Broker Bridge, reenviando toda la información al puerto `1883` del servidor `104.131.0.238`, en el que se instaló un broker MQTT con la configuración por defecto.

```

pi@raspberrypi:~ $ cat /etc/mosquitto/mosquitto.conf
# Place your local configuration in /etc/mosquitto/conf.d/
#
# A full description of the configuration file is at
# /usr/share/doc/mosquitto/examples/mosquitto.conf.example

pid_file /var/run/mosquitto.pid

persistence true
persistence_location /var/lib/mosquitto/

log_dest file /var/log/mosquitto/mosquitto.log

include_dir /etc/mosquitto/conf.d

allow_anonymous false
password_file /etc/mosquitto/pwfile
#listener 1883

#Creacion de bridge

connection bridge-02
address 104.131.0.238:1883
topic /sonoff/data out 1
pi@raspberrypi:~ $

```

Figura 6.8: Configuración mosquitto.conf

Para finalizar se creó un usuario MQTT en la Raspberry Pi, el que se utilizó previamente para autenticar los smart plugs. Para ello fue necesario ingresar las siguientes líneas de comando:

```
mosquitto_passwd -c pwfile user
```

Donde **user** es el nombre del usuario que se esta agregando. Una vez se ingresa el comando, se solicita la clave del usuario y su confirmación.

Al finalizar se puede observar que el usuario se ha creado en el archivo `pwfile` alojado en `/etc/mosquitto/pwfile` y su contraseña esta encriptada como muestra en 6.9:

```

pi@raspberrypi:~ $ cat /etc/mosquitto/pwfile
sonoff:$6$JdjR3y8PbCIPRsyi$iwJd+oziC//xb6zJr3xz+5MdDrnaPyrt+IwNYIj4XivvM//PwGj4
e7Ki+aIsKOJ7XLEYQO/h52cMn2BU7u5Dg==

```

Figura 6.9: Archivo pwfile

Una vez realizado todo el procedimiento del gateway se da por finaliza la implementación y configuración de la capa física y de comunicación.

7 | Resultados

En esta sección se muestran todos los resultados obtenidos del desarrollo del desafío. Estos se observan siguiendo el flujo de los datos -desde los smart plugs hasta su visualización-.

7.1. Firmware

Con la instalación del firmware en los smart plugs se pudo visualizar el consumo detectado de los electrodomésticos objetivos por medio de un servidor web alojado en los mismos Figura 7.1.

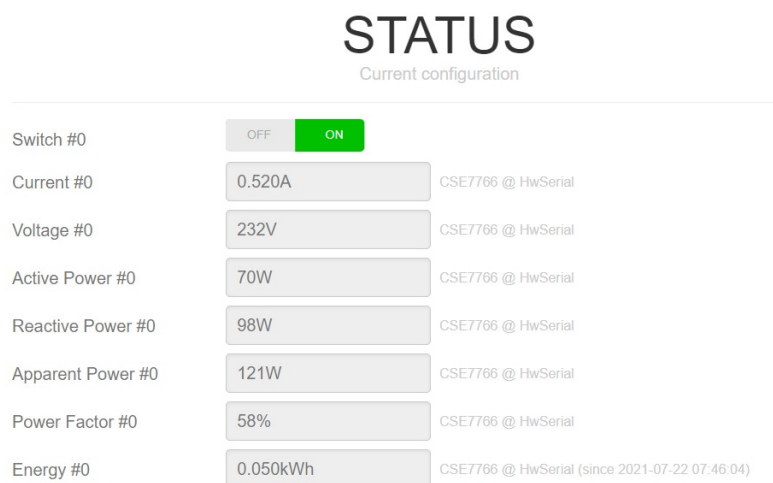


Figura 7.1: Consumo de un dispositivo en ESPurna

7.2. Comunicación

Se logró configurar la red Wi-Fi y el broker MQTT en los Smart Plugs de manera gráfica dentro de ESPurna para obtener la conectividad con el gateway, como se muestra en la Figura 7.2 .

Cuando la conectividad de los smart plugs con el gateway se estableció, se pudo observar los mensajes enviados por el protocolo de comunicación MQTT hacia el gateway, tal como se muestra en la Figura 7.3. A su vez, el gateway reenvía la información directamente al servicio en la nube como se muestra en la Figura 7.4

MQTT

Configure an MQTT broker in your network and you will be able to change the switch status via an MQTT message.

Enable MQTT ☐ NO ☒ YES

MQTT Broker

MQTT Port

MQTT User

You can use the following placeholders: {hostname}, {mac}

MQTT Password

MQTT Client ID

If left empty the firmware will generate a client ID based on the serial number of the chip. You can use the following placeholders: {hostname}, {mac}

MQTT QoS

MQTT Retain ☐ NO ☒ YES

MQTT Keep Alive

MQTT Root Topic

This is the root topic for this device. The {hostname} and {mac} placeholders will be replaced by the device hostname and MAC address.

- <root>/relay/#set Send a 0 or a 1 as a payload to this topic to switch it on or off. You can also send a 2

Figura 7.2: Configuración MQTT en ESPurna

7.3. Bases de datos

El envío de información al servidor por medio de MQTT se vio reflejado en el almacenamiento de la información en la base de datos InfluxDB como se muestra en la Figura 7.5.

Por otro lado la información de registro y login, junto con los dispositivos asociados

```

pi@raspberrypi:~$ mosquitto sub -h localhost -t /sonoff/data -u sonoff -P ihems2021
{"current":0,"voltage":231,"power":0,"reactive":0,"apparent":0,"factor":100,"energy":0.183,"time":"2021-11-22 05:37:11",
"mac":"98:F4:AB:E2:E8:B8","host":"ESPURNA-E2E8B8","ip":"192.168.0.104","id":187}
{"current":0,"voltage":231,"power":0,"reactive":0,"apparent":0,"factor":100,"energy":0.178,"time":"2021-11-22 05:37:12",
"mac":"98:F4:AB:E2:E9:DE","host":"ESPURNA-E2E9DE","ip":"192.168.0.103","id":200}
{"current":0,"voltage":232,"power":0,"reactive":0,"apparent":0,"factor":100,"energy":0.012,"time":"2021-11-22 05:37:13",
"mac":"98:F4:AB:E3:0A:DE","host":"ESPURNA-E30ADF","ip":"192.168.0.106","id":186}
{"current":0,"voltage":233,"power":0,"reactive":0,"apparent":0,"factor":100,"energy":0.001,"mac":"98:F4:AB:E2:DC:3D",
"host":"ESPURNA-E2DC3D","ip":"192.168.0.101","id":192}
{"current":0,"voltage":232,"power":0,"reactive":0,"apparent":0,"factor":100,"energy":0.012,"time":"2021-11-22 05:37:12",
"mac":"98:F4:AB:E2:D7:24","host":"ESPURNA-E2D724","ip":"192.168.0.105","id":185}
{"app":"ESPURNA","version":"1.14.1","board":"ITEAD SONOFF POW R2","host":"ESPURNA-E2DC3D","ssid":"Tenda_5B07A0","ip":
192.168.0.101,"mac":"98:F4:AB:E2:DC:3D","rssi":-53,"uptime":5700,"freeheap":18784,"relay/0":1,"vcc":3186,"loadavg":1,
"id":193}
{"current":0,"voltage":232,"power":0,"reactive":0,"apparent":0,"factor":100,"energy":0.012,"time":"2021-11-22 05:37:18",
"mac":"98:F4:AB:E2:D7:24","host":"ESPURNA-E2D724","ip":"192.168.0.105","id":186}
{"current":0,"voltage":231,"power":0,"reactive":0,"apparent":0,"factor":100,"energy":0.178,"time":"2021-11-22 05:37:18",
"mac":"98:F4:AB:E2:E9:DE","host":"ESPURNA-E2E9DE","ip":"192.168.0.103","id":201}
{"current":0,"voltage":232,"power":0,"reactive":0,"apparent":0,"factor":100,"energy":0.012,"time":"2021-11-22 05:37:19",
"mac":"98:F4:AB:E3:0A:DE","host":"ESPURNA-E30ADF","ip":"192.168.0.106","id":187}

```

Figura 7.3: Topico sonoff en Mosquitto dentro de Raspberry Pi

```

nkaldebuntu-s-lvcpu-igb-nyc3-01:~$ mosquitto sub -h localhost -t /sonoff/data
{"current":0.138,"voltage":226,"power":19,"reactive":25,"apparent":31,"factor":60,"energy":0.887,"time":"2021-11-06 19:39:10",
"mac":"98:F4:AB:E2:D7:24","host":"ESPURNA-E2D724","ip":"192.168.0.104","id":67428}
{"current":0,"voltage":225,"power":0,"reactive":0,"apparent":0,"factor":100,"energy":4.318,"time":"2021-11-06 19:39:11",
"mac":"98:F4:AB:E2:E9:DE","host":"ESPURNA-E2E9DE","ip":"192.168.0.106","id":112150}
{"current":0,"voltage":225,"power":0,"reactive":0,"apparent":0,"factor":100,"energy":0.599,"time":"2021-11-06 19:39:11",
"mac":"98:F4:AB:E2:E8:B8","host":"ESPURNA-E2E8B8","ip":"192.168.0.108","id":67423}
{"current":0,"voltage":225,"power":0,"reactive":0,"apparent":0,"factor":100,"energy":3.422,"time":"2021-11-06 19:39:11",
"mac":"98:F4:AB:E3:0A:DE","host":"ESPURNA-E30ADF","ip":"192.168.0.107","id":74850}
{"current":0,"voltage":228,"power":0,"reactive":0,"apparent":0,"factor":100,"energy":4.739,"time":"2021-11-06 19:39:12",
"mac":"98:F4:AB:E2:DC:3D","host":"ESPURNA-E2DC3D","ip":"192.168.0.105","id":74883}
{"current":0.138,"voltage":226,"power":19,"reactive":25,"apparent":31,"factor":60,"energy":0.887,"time":"2021-11-06 19:39:16",
"mac":"98:F4:AB:E2:D7:24","host":"ESPURNA-E2D724","ip":"192.168.0.104","id":67420}
{"current":0.528,"voltage":225,"power":69,"reactive":96,"apparent":119,"factor":58,"energy":4.318,"time":"2021-11-06 19:39:17",
"mac":"98:F4:AB:E2:E9:DE","host":"ESPURNA-E2E9DE","ip":"192.168.0.106","id":112151}
{"current":0,"voltage":224,"power":0,"reactive":0,"apparent":0,"factor":100,"energy":0.599,"time":"2021-11-06 19:39:17",
"mac":"98:F4:AB:E2:E8:B8","host":"ESPURNA-E2E8B8","ip":"192.168.0.108","id":67424}
{"current":0,"voltage":225,"power":0,"reactive":0,"apparent":0,"factor":100,"energy":3.422,"time":"2021-11-06 19:39:17",
"mac":"98:F4:AB:E3:0A:DE","host":"ESPURNA-E30ADF","ip":"192.168.0.107","id":74860}
{"current":0,"voltage":227,"power":0,"reactive":0,"apparent":0,"factor":100,"energy":4.739,"time":"2021-11-06 19:39:18",
"mac":"98:F4:AB:E2:DC:3D","host":"ESPURNA-E2DC3D","ip":"192.168.0.105","id":74884}

```

Figura 7.4: Topico sonoff en Mosquitto dentro de servicio en la nube (DigitalOcean)

```

> select * from mqtt_consumer_json_test limit 10
name: mqtt_consumer_json_test
time      apparent button/0 current energy factor freeheap host      id ip      loadavg mac      power reactive relay/0 rssi topic      uptime vcc  voltage
-----
1627346012000000000 0      0      0.204 100      ESPURNA-E2E9DE 518 192.168.0.113      98:F4:AB:E2:E9:DE 0      0      /sonoff/data 231
1627346022000000000 0      0      0.204 100      ESPURNA-E2E9DE 519 192.168.0.113      98:F4:AB:E2:E9:DE 0      0      /sonoff/data 231
1627346032000000000 0      0      0.204 100      ESPURNA-E2E9DE 520 192.168.0.113      98:F4:AB:E2:E9:DE 0      0      /sonoff/data 231
1627346042000000000 0      0      0.204 100      ESPURNA-E2E9DE 521 192.168.0.113      98:F4:AB:E2:E9:DE 0      0      /sonoff/data 231
1627346052000000000 0      0      0.204 100      ESPURNA-E2E9DE 522 192.168.0.113      98:F4:AB:E2:E9:DE 0      0      /sonoff/data 231
1627346062000000000 0      0      0.204 100      ESPURNA-E2E9DE 523 192.168.0.113      98:F4:AB:E2:E9:DE 0      0      /sonoff/data 231
1627346072000000000 0      0      0.204 100      ESPURNA-E2E9DE 524 192.168.0.113      98:F4:AB:E2:E9:DE 0      0      /sonoff/data 231
1627346075000000000 0      0      0.204 100      ESPURNA-E2E9DE 525 192.168.0.113 1      98:F4:AB:E2:E9:DE 0      0      /sonoff/data 5249 3116
1627346082000000000 0      0      0.204 100      ESPURNA-E2E9DE 526 192.168.0.113      98:F4:AB:E2:E9:DE 0      0      /sonoff/data 231
1627346092000000000 0      0      0.204 100      ESPURNA-E2E9DE 527 192.168.0.113      98:F4:AB:E2:E9:DE 0      0      /sonoff/data 231

```

Figura 7.5: Base de datos InfluxDB

a cada usuario se almaceno en una base de datos MongoDB.

7.4. Verificación y Validación

Los requerimientos definidos en el Capítulo 4 y los constantes cambios que estos han sufrido, han sido motivados y validados gracias a reuniones cada semana con el cliente, quien ha dejado claro las necesidades del proyecto y sus prioridades. El avance en el cumplimiento de estos requerimientos puede verse en la siguiente lista asociada a su respectiva validación:

- RF1: Listo

- La capacidad de medir el consumo eléctrico se observa en la Figura 7.1, donde se muestra propiedades físicas tales como: potencia, corriente, voltaje, entre otros.
- RF2: Listo
- RF3: Listo
 - Como se observa en la Figura 7.3, el smart plug tiene conectividad al gateway y a su vez envía la información a él. Esto valida el requisito funcional 2 y 3.
- RF4: Listo
 - En la Figura 7.2 se muestra que el smart plug se conecta al gateway mediante autenticación de usuario y contraseña.
- RF5: Listo
- RF6: Listo
 - Estos requisitos 5 y 6 se comprueban con la Figura 7.5, ya que se observa dentro de la base de datos los datos almacenados directamente por Telegraf en InfluxDB
- RF7: Posibilidad de mejoras
- RF8: Listo
- RF9: Listo
- RF10: Listo
- RF11: Listo
- RF12: Listo
- RF13: Listo
- RF14: Listo
- RF15: Listo
- RF16: En desarrollo
 - Los requisitos funcionales del 7 al 16 se verifican en otro documento por parte de Sebastián, integrante del grupo.

7.5. Plataforma Web

El desarrollo de la capa física y de comunicación fue utilizado por Sebastián para generar la plataforma de monitoreo energético. A continuación se muestra una serie de figuras que representan el resultado obtenido. Se destaca la Figura 7.7, Figura 7.9, Figura 7.8, Figura 7.11 y la Figura 7.10, en las que muestra en tiempo real los datos obtenidos de cada uno de los dispositivos conectados a los smart plugs asociados a: potencia, corriente, voltaje, energía acumulada, potencia aparente y potencia reactiva.

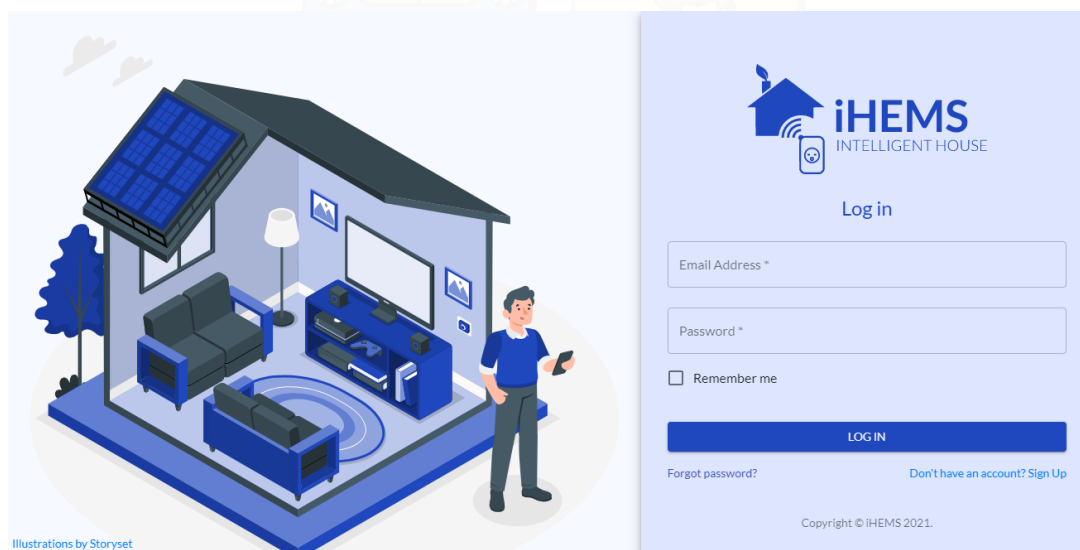


Figura 7.6: Vista login



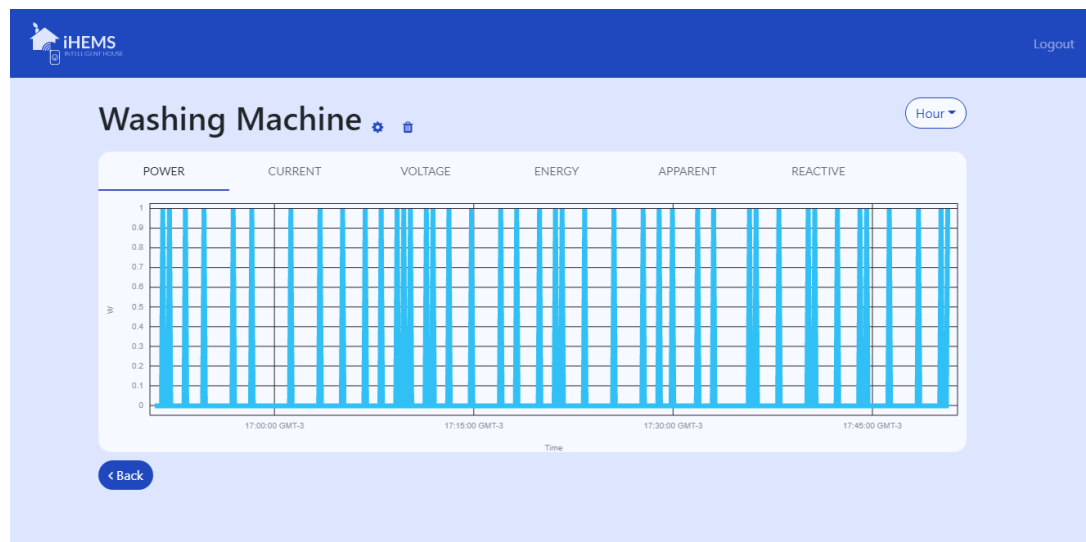
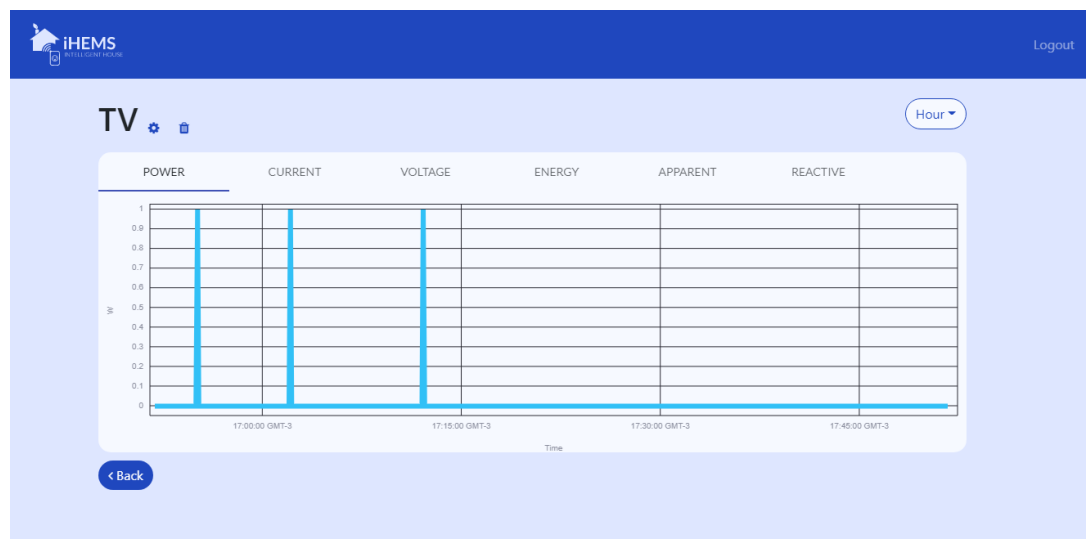
Figura 7.7: Vista Refrigerador



Figura 7.8: Vista TV Dormitorio



Figura 7.9: Vista Calefactor

**Figura 7.10:** Vista Lavadora**Figura 7.11:** Vista TV Living

8 | Conclusión

En este documento se presentó el desarrollo de una plataforma de monitoreo energético mediante un modelo de monitoreo de cargas intrusivo.

Se generó solución a los objetivos para este desafío, tanto general como específicos obteniendo información relevante para el desarrollo del proyecto mediante reuniones con el cliente, generando arquitectura de red por medio del estudio de smart plugs, tecnologías inalámbricas y protocolos y analizando las diferentes opciones para dar solución a la problemática planteada.

También, como se observó en el Capítulo 7, se dio cumplimiento a los requerimientos, generando un prototipo de red y aplicación funcional, logrando recoger, transmitir, almacenar y desplegar datos provenientes de electrodomésticos utilizando los smart plug Sonoff Pow R2.

Trabajos Futuros

Si bien el trabajo realizado cumple con las expectativas, se propone realizar las siguientes mejoras:

- Instalación de OpenWRT a gateway, con el objetivo de facilitar al usuario la tarea de dar conectividad al gateway.
- Configurar el encendido y apagado de los electrodomésticos a través de la plataforma.
- Asociación de smart plugs con código QR, para facilitar la conectividad entre los dispositivos y la plataforma.

Estas mejoras quedan como trabajos futuros a aquellas personas que continúen con el trabajo realizado en este documento y tienen el objetivo de mejorar la calidad de la solución

entregada.



Bibliografía

- [1] Shashank Singh, Amit Roy, and M.P. Selvan. Smart load node for nonsmart load under smart grid paradigm: A new home energy management system. *IEEE Consumer Electronics Magazine*, 8(2):22–27, 2019. 1.1
- [2] Cristóbal Escobar y Juan Francisco Garcés. El mundo y los smart grids: ¿que tan cerca estamos? *Pontificia Universidad Catolica de Chile*, pages 1–31, 2009. 2.1
- [3] What is the internet of things (iot)? <https://www.oracle.com/internet-of-things/what-is-iot/>. 2.2
- [4] Ying Wu, Yanpeng Wu, Josep M Guerrero, Juan C Vasquez, Emilio J Palacios-Garcia, and Jiao Li. Convergence and interoperability for the energy internet: From ubiquitous connection to distributed automation. *IEEE Industrial Electronics Magazine*, 14(4):91–105, 2020. 2.3
- [5] Patricia Franco, José Manuel Martínez, Young-Chon Kim, and Mohamed A Ahmed. Iot based approach for load monitoring and activity recognition in smart homes. *IEEE Access*, 9:45325–45339, 2021. 2.4, 4, 4.1, 4.2, 4.4
- [6] Bin Zhou, Wentao Li, Ka Wing Chan, Yijia Cao, Yonghong Kuang, Xi Liu, and Xiong Wang. Smart home energy management systems: Concept, configurations, and scheduling strategies. *Renewable and Sustainable Energy Reviews*, 61:30–40, 2016. 2.6
- [7] Cuadro normativo y tabla de espacios y usos mínimos para el mobiliario., 2017. 2.7
- [8] Muhammad Aziz, Takuya Oda, Takashi Mitani, Yoko Watanabe, and Takao Kashiwagi.

- Utilization of electric vehicles and their used batteries for peak-load shifting. *Energies*, 8(5):3720–3738, 2015. 3, 3.1, 3.2, 3.2
- [9] Yuto Yoshimura, Tomoaki Kondo, Michihiro Kawanishi, Tatsuo Narikiyo, and Akinori Sato. Model predictive control of ev storage battery with hems based on particle swarm optimization. In *2015 IEEE Innovative Smart Grid Technologies-Asia (ISGT ASIA)*, pages 1–5. IEEE, 2015. 3, 3, 3.2, 4.6.1
- [10] T Yamaguchi, M Sumiya, S Inagaki, T Suzuki, A Ito, M Fujita, and J Kanamori. Model predictive control of car storage battery in hems considered car traveling. In *The SICE Annual Conference 2013*, pages 1352–1358. IEEE, 2013. 3, 3.2
- [11] Sami Abdullah Ben Slama. Design and implementation of home energy management system using vehicle to home (h2v) approach. *Journal of Cleaner Production*, page 127792, 2021. 3, 3.3, 3.2, 4.6.1
- [12] Mohammad Shakeri, Mohsen Shayestegan, SM Salim Reza, Iskandar Yahya, Badariah Bais, Md Akhtaruzzaman, Kamaruzzaman Sopian, and Nowshad Amin. Implementation of a novel home energy management system (hems) architecture with solar photovoltaic system as supplementary source. *Renewable energy*, 125:108–120, 2018. 3, 3.2, 4.6.1
- [13] MA Abouelela and MM Abouelela. Wireless communication role in home energy management system (hems). In *2015 23rd Telecommunications Forum Telfor (TELFOR)*, pages 204–207. IEEE, 2015. 3, 3.4, 3.2, 4.6.1
- [14] Wei-Chen Chen, Ya-Hung Chen, Chao-Lin Wu, and Li-Chen Fu. An efficient data storage method of nosql database for hem mobile applications in iot. In *2014 IEEE International Conference on Internet of Things (iThings), and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCoM)*, pages 336–339. IEEE, 2014. 3, 3.5, 3.2, 4.6.1
- [15] Mani Dheeraj Mudaliar and N Sivakumar. Iot based real time energy monitoring system using raspberry pi. *Internet of Things*, 12:100292, 2020. 3, 3.2, 4.6.1
- [16] Prapaporn Rattanathamrong, Yoottana Boonpalit, Siwakorn Suwanjinda, Ayuth Mangmeesap, Ken Subraties, Vahid Daneshmand, Shava Smallen, and Jason Haga.

- Overhead study of telegraf as a real-time monitoring agent. In *2020 17th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pages 42–46. IEEE, 2020. 3, 3.1, 3, 3.2
- [17] Takeshi Yashiro, Shinsuke Kobayashi, Noboru Koshizuka, and Ken Sakamura. A software hems framework for consumer electronics. In *2013 IEEE 2nd Global Conference on Consumer Electronics (GCCE)*, pages 229–230. IEEE, 2013. 3, 3.6, 3.2, 4.6.1
- [18] Kazuki Arima, Masayuki Kaneko, Takashi Murakami, Masao Isshiki, and Hiroshi Sugimura. User experience for hems based on gamification and life log. In *2016 IEEE 5th Global Conference on Consumer Electronics*, pages 1–2. IEEE, 2016. 3, 3.7, 3.8, 3.2, 4.6.1
- [19] Toshiba feminity club. <https://www.tlt.co.jp/tlt/products/hems/about/uchiconne.htm>. 3.9
- [20] G Dileep. A survey on smart grid technologies and applications. *Renewable Energy*, 146:2589–2625, 2020. 4.2
- [21] Ying Wu, Yanpeng Wu, Josep M Guerrero, and Juan C Vasquez. Digitalization and decentralization driving transactive energy internet: Key technologies and infrastructures. *International Journal of Electrical Power & Energy Systems*, 126:106593, 2021. 4.3
- [22] Antonio Ridi, Christophe Gisler, and Jean Hennebert. A survey on intrusive load monitoring for appliance recognition. In *2014 22nd International Conference on Pattern Recognition*, pages 3702–3707, 2014. 4.4
- [23] Tasmota. <https://tasmota.github.io/docs/>. 5.7
- [24] Espurna. <https://github.com/xoseperez/espurna>. 5.7
- [25] Espeasy. <https://github.com/letscontrolit/ESPEasy>. 5.7
- [26] Firmware comparison. <https://lobradov.github.io/FOSS-firmware-comparison-revisited/>. 5.7

- [27] Jetson nano. <https://www.nvidia.com/es-la/autonomous-machines/embedded-systems/jetson-nano/>. 5.1.3.1
- [28] S. Sujin Issac Samuel. A review of connectivity challenges in iot-smart home. In *2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC)*, pages 1–4, 2016. 5.2.1
- [29] Jonathan Tournier, François Lesueur, Frédéric Le Mouël, Laurent Guyon, and Hicham Ben-Hassine. A survey of iot protocols and their security issues through the lens of a generic iot stack. *Internet of Things*, page 100264, 2020. 5.2.1
- [30] Wifi ieee 802.11, url=<https://www.electronics-notes.com/articles/connectivity/wifi-ieee-802-11/what-is-wifi.php>, howpublished = <https://www.electronics-notes.com/articles/connectivity/wifi-ieee-802-11/what-is-wifi.php>. 5.2.1.1
- [31] Carlos Pereira and Ana Aguiar. Towards efficient mobile m2m communications: Survey and open challenges. *Sensors (Basel, Switzerland)*, 14:19582–19608, 10 2014. 5.2.2
- [32] Request-response vs pub-sub. <https://blog.opto22.com/optoblog/request-response-vs-pub-sub-part-2>. 5.2.3

ANEXO

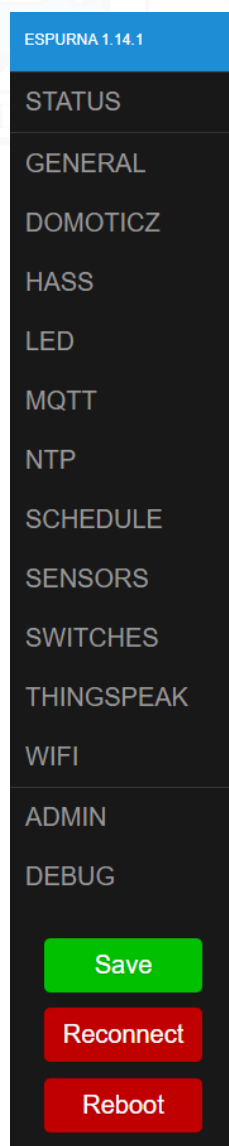


Figura 1: Menú ESPurna