UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Aeroespacial y Diseño Industrial

Estudio e implementación de sistemas inalámbricos sin conexión a Internet en edificios inteligentes

Trabajo Fin de Grado

Grado en Ingeniería Electrónica Industrial y Automática

AUTOR/A: Amorós Conejero, Francisco

Tutor/a: Perles Ivars, Ángel Francisco

Cotutor/a externo: LABORDA MACARIO, JAIME

CURSO ACADÉMICO: 2023/2024

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

ESTUDIO E IMPLEMENTACIÓN DE SISTEMAS INALÁMBRICOS SIN CONEXIÓN A INTERNET EN EDIFICIOS INTELIGENTES

1. MEMORIA

Autor:

D. Francisco Amorós Conejero

Tutor:

D. Ángel Francisco Perles Ivars

Valencia, abril de 2024

ÍNDICE DEL PROYECTO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

GLOSARIO DE SIGLAS Y ABREVIATURAS

AWS	Amazon Web Services
BLE	Bluetooth Low Energy
CG-NAT	Carrier Grade Network Address Translation
CoAP	Constrained Application Protocol
DBPSK	modulación por desplazamiento de fase
DDNS	Dynamic Doman Name System
DDoS	Distributed Denia lof Service
DNS	Domain Name System
ETCS	European Credit Transfer and Accumulation System
HA	Home Assistant
HDD	Hard Disk Drive
HDMI	High Definition Multimedai Interface
HTTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Secure
IDE	entorno de desarrollo integrado
IoT	Internet de las cosas
IP	Internet Protocol
ISM	Industrial, Scientific and Medical
LAN	Local Area Network
LoRaWAN	LoRa Wide Area Networt
LPWAN	Low Power Wide Area Networks
M2M	Machine to Machine
MAC	Media Access Control
MQTT	Message Queuing Telemetry Transport
NAS	Network Attached Storage
NAT	Network Address Translator
NB-IoT	NarrowBand – Internet of Things
OPC UA	Open Productivity Collaboration Unified Architecture
P2P	Peer to Peer

PC	Personal Computer
PWA	Progressive Web Applications
QoS	Quality of Services
RAM	Random Access Memory
RISC	Reduced instruction set computing
SDN	Software Defined Networking
SDRAM	Synchronous Dynamic Random Access Memory
SSD	Solid State Drive
SSL	Secure Sockets Layer
TCP	Trasmition Control Protocol
UDP	User Datagram Protocol
UI	User Interface
USB	Universal Serial Bus
VPN	Virtual Private Network
VX-LAN	Virtual Extensible Local Area Network
WiFi	Wireless Fidelity
YAML	YAML Ain’t Markup Language

1. OBJETO DEL PROYECTO

El objeto del presente proyecto es estudiar opciones y desarrollar un demostrador para un sistema de bajo consumo para gestionar una red de dispositivos inteligentes independiente de la conexión a Internet. Para ello se hará uso de diferentes técnicas de Internet of Things.

El sistema será capaz de mantener la funcionabilidad de los diferentes dispositivos inteligentes de la red, permitiendo la comunicación entre estos, en caso de corte a la conexión de Internet. Además, se evaluará un sistema de alimentación ininterrumpida para mantener el funcionamiento del sistema en caso de un corte de suministro.

2. ANTECEDENTES

Con el abaratamiento de la electrónica es cada vez más frecuente que en fábricas, comercios o viviendas encontrar dispositivos inteligentes en la iluminación, termostatos y calefacción, sistemas de vigilancia… todo ello con la característica de tener una fuerte dependencia de la conexión a Internet y, si esta falla, la gran mayoría de estos dispositivos inteligentes dejaran de funcionar correctamente. Debido a esto en los últimos años, se ha producido un creciente interés en el desarrollo de tecnologías y estándares para garantizar la robustez y la seguridad de los edificios inteligentes.

Los edificios inteligentes son aquellos que utilizan tecnologías avanzadas para mejorar la eficiencia energética, la seguridad, la comodidad y la gestión de los recursos. Estos edificios están equipados con sensores, actuadores y otros dispositivos IoT que permiten controlar y monitorear sistemas y equipos de manera remota.

Existen varios sistemas y servidores para el control de dispositivos IoT. En los sistemas de automatización del hogar destacan Amazon Echo, Google Nest, Apple Homekit entre otros. Mientras en la industria se ofrece productos más especializados para las necesidades de cada industria como los sistemas de Siemens Desigo, Hohnson Controls Metasys, Schneider Electric StruxuseWare, Rockwell Automation, Schneider Electric entre otros.

Los dispositivos IoT se están convirtiendo en una parte integral de la industria y de nuestras vidas. La seguridad, la interoperabilidad y la escalabilidad son solo algunos de los desafíos que deben abordarse para mejorar la tecnología de edificios inteligentes.

Es por eso por lo que este proyecto es tan importante. Al implementar un servidor IoT eficaz y eficiente, podemos garantizar el correcto funcionamiento de los dispositivos IoT en un edificio inteligente, lo que a su vez puede mejorar la seguridad, la eficiencia energética y la comodidad de los usuarios. Este proyecto tiene como objetivo abordar los desafíos actuales en la implementación de servidores IoT en edificios inteligentes y contribuir a la evolución de la tecnología de edificios inteligentes en general.

3. ESTUDIO DE NECESIDADES, FACTORES A CONSIDERAR: LIMITACIONES Y CONDICIONANTES

El primer paso en la realización de cualquier proyecto consiste en hacer un análisis exhaustivo de las necesidades a satisfacer, así como de las diversas áreas que se desean abordar, con el propósito de establecer especificaciones precisas.

A continuación, se describen con detalle los factores y necesidades a tener en cuenta en la implementación de un sistema inalámbrico sin conexión a Internet en edificios inteligentes.

3.1. Tecnologías inalámbricas

En el estudio de los requisitos para el desarrollo de edificios inteligentes, se han identificado varios desafíos asociados con las soluciones actualmente disponibles en el mercado.

Históricamente, estas soluciones se basaban en la transmisión de datos mediante cables, utilizando protocolos propietarios de cada compañía o en estándares costosos combo KNX [1]. Esta aproximación a menudo dificultaba la integración con otras tecnologías y requería la contratación de servicios de soporte específicos de la empresa proveedora. Sin embargo, con el avance de las tecnologías, han surgido métodos inalámbricos y nuevos estándares que facilitan la integración de sensores y actuadores en edificios inteligentes.

Las soluciones inalámbricas tradicionales y comúnmente conocidas, como las redes WiFi, Bluetooth o la radiofrecuencia de 433MHz, no son adecuadas para nuestro propósito debido al uso de protocolos de comunicación propietarios y a su alcance limitado (aproximadamente 100 metros). Estas soluciones son válidas en entornos pequeños como edificios residenciales, pero no son adecuadas para supervisar múltiples edificios, grandes almacenes o ubicaciones al aire libre.

Por lo tanto, es necesario buscar una solución versátil, que sea válida para cualquier situación que pueda surgir en la implementación de un edificio inteligente. Esta solución implica la búsqueda de un protocolo de comunicación inalámbrica de largo alcance que minimice el impacto de obstáculos, y que permita mejorar tanto el consumo energético utilizado para la transmisión como el alcance y cobertura de la señal. Con este propósito se llevará a cabo un estudio de varias tecnologías inalámbricas para, finalmente, integrar una de ellas en la implementación de un edificio inteligente

3.2. Consumo energético

El estudio de necesidades del consumo energético para la implementación de un edificio inteligente se centra en encontrar soluciones eficientes que permitan un funcionamiento continuo y de bajo consumo para el dispositivo que actúa como servidor y controlador de la instalación. Dado que este dispositivo estará en funcionamiento las 24 horas del día, es fundamental minimizar su consumo energético y garantizar en operaciones en caso de fallo de suministro eléctrico.

Para lograr esto, se explorarán diferentes tecnologías de hardware y software que utilicen componentes de bajo consumo y algoritmos eficientes para minimizar el uso de recursos energéticos.

Además, se considerará la implementación de baterías de respaldo como una medida de contingencia en caso de cortes en el suministro energético. Estas baterías permitirán mantener el funcionamiento del dispositivo durante interrupciones temporales del suministro, garantizando la continuidad de las operaciones y la preservación de los datos almacenados. Es importante destacar que, al evaluar las opciones de baterías, se debe también considerar la sostenibilidad y el impacto medioambiental y socioeconómico de los materiales utilizados en dichas baterías

Por otro lado, en cuanto a los dispositivos periféricos, como sensores y actuadores, se debe prestar atención a la eficiencia energética en diferentes aspectos. Por ejemplo, se pueden explorar soluciones de iluminación eficiente que utilicen tecnología LED y sensores de luz natural o de movimiento, lo cual reducirá el consumo energético relacionado con la iluminación en todo el edificio. Asimismo, se puede implementar una gestión inteligente de la aclimatación, utilizando sistemas de control que optimicen el uso de calefacción, refrigeración o ventilación para minimizar el consumo energético.

Además, es importante considerar la monitorización y control de energía en tiempo real, lo que permitirá identificar patrones de consumo y optimizar el uso de energía en diferentes áreas del edificio. Esto incluye la implementación de sistemas de monitorización para dispositivos conectados. También se puede abordar la eficiencia en el uso de otros recursos, como el agua, mediante la implementación de sistemas de control y monitorización de consumo.

En resumen, el estudio de necesidades del consumo energético en la implementación de un edificio inteligente se enfocará en encontrar soluciones de bajo consumo para el dispositivo central, así como en valorar la incorporación de baterías de respaldo para garantizar la continuidad operativa y la eficiencia energética del sistema. Además, se deben tener en cuenta aspectos de eficiencia energética en los dispositivos periféricos, tales como la iluminación, la gestión inteligente de la aclimatación y la monetización y control de energía y otros recursos.

3.3. Seguridad y privacidad

El estudio de necesidades en cuanto a seguridad y privacidad en la instalación de un edificio inteligente se centra en garantizar la protección y la confidencialidad de los datos, así como en prevenir intrusiones y ataques cibernéticos. Para lograr esto, se deben considerar diversos factores fundamentales.

En primer lugar, es necesario implementar medidas sólidas de protección contra intrusiones y ataques cibernéticos. Esto implica el uso de firewalls, sistemas de detección y prevención de intrusiones, y estar al tanto de las últimas vulnerabilidades y parches de seguridad para asegurar la integridad del sistema.

La gestión de acceso y control de permisos también es esencial. Se debe establecer un sistema que garantice que solo los usuarios autorizados tengan acceso a los datos y funcionalidades específicas del sistema. Esto implica asignar roles y privilegios adecuados, así como revocar el acceso cuando sea necesario.

El cifrado de datos y la autenticación de usuarios y dispositivos son aspectos clave. El uso de algoritmos de cifrado robustos y la implementación de autenticación sólida garantizan que solo los usuarios legítimos puedan acceder a los datos. Es importante recordar la utilización de contraseñas fuertes.

Además, se debe garantizar la privacidad de los usuarios. Esto implica cumplir con las regulaciones y normativas de privacidad aplicables, como la GDPR [2], y adoptar medidas técnicas y organizativas para proteger los datos personales recopilados y procesados en la instalación.

La reducción de la dependencia de infraestructura externa también es un aspecto a considerar. Al minimizar la dependencia de recursos y servicios externos, se fortalece la seguridad y se evita la exposición a posibles brechas de seguridad relacionadas con terceros.

El respaldo y recuperación de datos es crucial para garantizar la disponibilidad y la integridad de la información en caso de fallos o incidentes. Implementar sistemas de respaldo periódicos y contar con planes de contingencia para restaurar los datos en caso de pérdida o corrupción es fundamental.

Por último, la supervisión de actividades sospechosas y la detección de comportamientos anómalos son esenciales para identificar posibles amenazas. La implementación de sistemas de supervisión y análisis de registros de eventos permiten detectar actividades sospechosas y tomar medidas de manera oportuna.

En resumen, el estudio de necesidades en seguridad y privacidad en la instalación de un edificio inteligente abarca la protección contra intrusiones y ataques cibernéticos, la gestión de acceso y control de permisos, el cifrado de datos, la autenticación de usuarios y dispositivos, la garantía de privacidad, la reducción de la dependencia de infraestructura externa, el respaldo y recuperación de datos, y la supervisión de actividades sospechosas.

3.4. Versatilidad

La versatilidad es un factor clave en el estudio de necesidades para la instalación de un edificio inteligente. El sistema debe ser capaz de adaptarse a diversas situaciones y requerimientos sin depender exclusivamente de la conectividad a Internet. A continuación, se destacan algunos aspectos esenciales.

En primer lugar, la capacidad de operar de forma autónoma sin depender exclusivamente de la conectividad a Internet. Esto implica que pueda llevar a cabo sus funciones y tomar decisiones basadas en la información disponible localmente, garantizando un funcionamiento continuo y sin interrupciones.

Es fundamental minimizar la latencia o el retraso en las operaciones del sistema. Esto es especialmente importante en entornos donde la toma de decisiones y la respuesta deben ser rápidas. Al reducir la latencia, se garantiza una mayor eficiencia y efectividad en el control y gestión del edificio inteligente.

El sistema debe ser compatible con una amplia gama de dispositivos IoT, como sensores y actuadores. Esto permite una integración de diversos dispositivos y sistemas en la

infraestructura del edificio inteligente, brindando flexibilidad y capacidad para adaptarse a diferentes necesidades y requisitos.

Es importante que el sistema sea escalable y flexible, lo que significa que pueda adaptarse a cambios en los requisitos y necesidades del edificio con facilidad. Esto incluye la capacidad de agregar, reemplazar o actualizar dispositivos y componentes del sistema sin afectar negativamente al funcionamiento general.

El sistema debe ofrecer capacidades de gestión y configuración local, permitiendo a los usuarios realizar ajustes y modificaciones sin depender de servicios en la nube o de una conexión a Internet. Esto proporciona mayor autonomía y control directo sobre el funcionamiento del sistema en el entorno del edificio.

Finalmente, optar por soluciones de código abierto se evita la dependencia de proveedores específicos. Esto significa que el proyecto puede evolucionar y adaptarse con el tiempo sin estar restringido por las limitaciones de un proveedor específico. Además, las soluciones de código abierto que están respaldadas por comunidades de desarrolladores activas aseguran no solo el soporte continuo y la resolución de problemas, sino también proporciona una amplia gama de recursos y conocimientos compartidos.

En resumen, el estudio de necesidades de la instalación de un edificio inteligente destaca la importancia de la versatilidad. El sistema debe ser capaz de operar de forma autónoma, reducir la latencia, ser compatible con diversos dispositivos IoT, ser escalable y flexible, y permitir la gestión y configuración local. Estos factores garantizan un funcionamiento eficiente y adaptable del edificio inteligente, independientemente de la conectividad a Internet.

4. PLANTEAMIENTO DE SOLUCIONES ALTERNATIVAS Y JUSTIFICACIÓN DE LA SOLUCIÓN ADOPTADA

En todo proyecto, y en especial en un proyecto de ingeniería, se requiere de un estudio de investigación con el objetivo de conocer el estado del arte sobre las tecnologías que se van a tratar y tomas la decisión idónea de cada tipo para el proyecto. Para el proyecto que nos atañe, se han seleccionado una serie de elementos crítico que conviene comparar entre distintas propuestas para elegir la más idónea.

El proyecto implicará un estudio de tecnologías inalámbricas, evaluación de diferentes hardware para el sistema central, selección de una plataforma de software, elección de un protocolo de mensajería eficiente, identificación de elementos para acceso remoto seguro y elección de una plataforma de desarrollo. Cada decisión se tomará considerando factores como rendimiento, costo, facilidad de uso, seguridad y soporte de la comunidad.

4.1. Elección de la tecnología inalámbrica

La elección de la tecnología a usar para la transmisión de datos entre los diferentes dispositivos IoT de la instalación y la aplicación es una decisión crítica en el desarrollo de este proyecto, ya que ella dependerá de numerosos factores a tener en consideración como la cobertura o la transmisión de datos entre otros.

Las opciones alámbricas se han descartado por considerarse que no cumplen con la filosofía de versatilidad que se exige que tenga. A menudo resulta inabordable cablear todo el edificio, incluso provocando en algunos casos un impacto estético al conllevar la instalación de canaletas o similares. Sin embargo, la tecnología alámbrica ofrece una serie de ventajas que no proporciona la inalámbrica, como puede ser la confianza o confiabilidad de la conexión, la velocidad, la robustez o la seguridad.

Teniendo en cuenta todo ello, se ha seleccionado un abanico de tecnologías inalámbricas actuales que se evaluarán a continuación, con la intención de elegir la que mejor se ajuste al proyecto.

4.1.1. WiFi

La tecnología inalámbrica WiFi (Wireless Fidelity) permite la interconexión de dispositivos electrónicos habilitados para ello, tales como ordenadores, videoconsolas, teléfonos inteligentes, televisores entre el cada vez más creciente abanico de dispositivos. Esta tecnología define un estándar que permite la conexión de estos dispositivos entre sí y/o la conexión de estos a la red de redes de Internet.

Figura 1. Logo de certificación WiFi.

Esta tecnología surge en 1999 por la necesidad de establecer un mecanismo de conexión inalámbrica compatible entre distintos y diversos dispositivos, ya fueran ordenadores personales, asistentes digitales, impresoras u otros. Para esta labor se juntaron varias empresas entre las que destacan Nokia y 3Com que se unieron para crear lo que actualmente se conoce como WiFi Alliance [3], un colectivo de empresas que estandarizan y regulan el uso de esta tecnología inalámbrica, asegurando que todos los equipos con el sello WiFi (Figura 1) puedan comunicarse entre ellos sin problemas, independientemente del tipo de dispositivo o del fabricante.

La tecnología WiFi se rige por los estándares establecidos por el Institute of Electrical and Electronics Engineers (IEEE) en la familia de estándares 802.11 (IEEE 802.11). Este estándar define y regula tanto la capa física como la capa MAC (*Medium Access Control*) del modelo OSI para la tecnología WiFi. El modelo OSI (*Open System Interconnection)* [4] es como se transmiten los paquetes de datos y su codificación, el resto es totalmente idéntico, lo que garantiza la compatibilidad de todos los servicios de redes locales entre sí.

Existen diversos tipos de WiFi basados en el mismo estándar, el clásico que trabaja en una frecuencia de 2,4 GHz y permite transmitir datos a una velocidad de hasta 300Mbit/s y el más actual conocido como WiFi 6, que puede trabajar en las frecuencias 2,4GHz, 5GHz y 6GHz permitiendo una mayor tasa de transmisión de entre 600 y 9.608 Mbits/s. Además, este nuevo estándar opera canales relativamente limpios y sin interferencias al no existir otras tecnologías como si ocurre con la banda de 2,4GHz o donde están presentes Bluetooth, Zigbee, WUSB…. Aunque presenta un menor alcance respecto al 2,4GHz al ser de mayor frecuencia, y por ende, menor longitud de onda.

WiFi permite un ancho de banda muy grande lo que permite un envío rápido de gran cantidad de información, como videos, a costa de un gran gasto de energía para la transmisión y la modulación y demodulación de los datos Esto lo hace poco práctico para dispositivos alimentados a baterías, ya que requeriría de grandes baterías y/o su autonomía no sobrepasaría el día de uso. Además, al operar en la banda de 2,4 GHz su alcance no es superior a los 100-200 metros en el mejor de los casos debido a las propiedades físicas de estas longitudes de onda, teniendo a menudo la necesidad de instalar repetidores de señal.

4.1.2. Bluetooth Low Energy

Como alternativa al alto consumo del WiFi se considera Bluetooth Low Energy (BLE) o también conocido como *Bluetooth Smart.* BLE es una versión ligera del Bluetooth clásico y fue introducido como parte del Bluetooth 4.0. Aunque comparte muchas cosas con el Bluetooth tradicional, BLE tiene una arquitectura completamente diferente.

Figura 2. Logo Bluetooth Smart.

En 1994 Ericsson inició un proyecto de investigación para encontrar una forma de conectar dispositivos móviles y periféricos de manera inalámbrica. En 1998, Ericsson se unió con otras compañías como Nokia, IBM, Intel y Toshiba para formar el Special Interest Group (SIG) de Bluetooth [5], con el objetivo de establecer un estándar común para la tecnología Bluetooth. Surgiendo así la especificación Bluetooth 1.0 que fue lanzada para 1999, y desde entonces ha habido varias versiones y mejoras en la tecnología.

BLE fue desarrollado por SIG en respuesta a la creciente demanda de dispositivos electrónicos que pudieran funcionar con batería pequeñas durante largos períodos de tiempo. Por ende, se buscaba una solución inalámbrica que permitiera una conectividad eficiente y de bajo consumo para este tipo de dispositivos. Finalmente, el lanzamiento oficial de BLE se produjo con la especificación Bluetooth 4.0 en 2010. Desde entonces, ha habido varias actualizaciones y mejoras de la tecnología con las especificaciones Bluetooth 4.1, 4.2, 5.0, etc.

Este nuevo estándar Bluetooth es usado en los conocidos como *wearebles* o llevables en español, ya que permite la conexión entre un dispositivo central (comúnmente un teléfono inteligente) con un dispositivo periférico como auriculares inalámbricos, altavoces, teclados entre otros, con un consumo muy bajo de energía que permite reducir el tamaño de estos dispositivos al no ser necesarias grandes baterías, y sin perjudicar la autonomía.

No es el objeto de este proyecto entrar en profundidad sobre cómo funciona este estándar de transmisión, por lo que nos quedaremos con su topología y con las razones por las cuales ha sido descardado su uso en el proyecto.

Este protocolo de transmisión inalámbrica está pensado para dispositivos sencillos no inteligentes (sensores que tan solo transmiten datos y no realizan un gran procesado de estos) que han de estar siempre conectados a un dispositivo máster o cliente que es el que hace la recolección y procesado de los datos. Este dispositivo máster también será el que tenga más

posibilidades de conectividad para subir los datos a la red de Internet si fuera necesario. Nos quedaremos por tanto con la idea de que para hacer uso de BLE es necesario disponer de un dispositivo auxiliar en el radio de cobertura de los sensores, que ronda los 100 metros en campo abierto en el mejor de los casos. Por la naturaleza del proyecto, no podemos asegurar que los sensores vayan a estar separados entre sí menos de 100 metros, ya que, de ser así, esto implicaría, por ejemplo, el instalar un dispositivo master en cada sección del edificio para monitorizar todos los sensores y actuadores instalados [6].

4.1.3. ZigBee

El nombre "Zigbee" se inspiró en el comportamiento de las abejas, que se comunican entre sí a través de una serie de movimientos en forma de zigzag. Esto representa la idea de una red de dispositivos que se comunican entre sí de manera similar, formando una red en malla.

Zigbee es un estándar de comunicación inalámbrica diseñado para aplicaciones de bajo consumo de energía y baja tasa de transmisión de datos. Fue desarrollado por la Zigbee Alliance, actualmente Connectivity Standards Alliance [7], una organización sin fines de lucro compuesta por varias compañías de tecnología.

Zigbee fue introducido por primera vez en 2003 con el objetivo de proporcionar una solución de conectividad inalámbrica confiable y de bajo consumo de energía para aplicaciones de automatización del hogar, control industrial, monitorización ambiental y otros dispositivos IoT.

Figura 3. Logo ZigBee.

El estándar Zigbee opera en las bandas de frecuencia de radio sin licencia, conocidas como bandas libres. Estas son, entre otras, las bandas de ISM (Industrial, Scientific and Medical) de 2.4 GHz (igual que WiFi y Bluetooth), pero también puede operar las de 868 Mhz en Europa y 915 Mhz en EEUU. Ofrece características como bajo consumo de energía, seguridad mediante señales cifradas con encriptación AES-128 [8], escalabilidad y confiabilidad en entornos con múltiples dispositivos. Aunque, presenta una velocidad de transmisión baja de hasta 250 kbps con un rango de hasta 75 metros para la banda de 2,4 GHz. Está basada en el estándar IEEE 802.15.4 de redes inalámbricas de área personal (Wireless Personal Area Networt, WPAN) [9].

Las soluciones comerciales que hacen uso de la red ZigBee suelen ser productos para el hogar como bombillas LED inteligentes con temperatura de color ajustable, mandos remotos para controlarlas, control de brillo para paredes, sensores de gas, de temperatura, de movimiento, de apertura de ventanas y puertas, entre otros.

ZigBee funciona bajo una tipología de malla, aunque se pueden utilizar otras topologías como estrella u árbol (Figura 4). ZigBee define 3 tipos de dispositivos distintos:

- **Coordinador**: Toda red ZigBee precisa de un coordinador o master, que se encargará de manejar, procesar y controlar las conexiones de los demás dispositivos. Al coordinador solamente irá conectado uno o más routers y solo puede haber una figura de coordinador por red.
- **Router ZigBee**: Dispositivo donde van conectados los end-devices, es el encargado de enrutar las conexiones y datos al dispositivo final que se conecta. Un router puede ser conectado a otro router y/o end-devices (hasta 255 dispositivos por router)
- **Dispositivo final ZigBee (end device)**: Es aquel que actúa como dispositivo de aplicación final e implementa los sensores, actuadores u etiquetas inteligentes. Dicho dispositivo está dormido la mayor parte del tiempo con un consumo de energía cercano a cero, y se despierta para transmitir datos o escuchar periódicamente. No realiza tareas de enrutamiento.

De esta manera, ZigBee permite crear redes de malla muy versátiles y escalables para cualquier situación. El ejemplo clásico de ZigBee en domótica de hogar pasa por tener un dispositivo coordinador central y un enrutador en cada habitación de la casa con varios sensores u actuadores conectados al mismo.

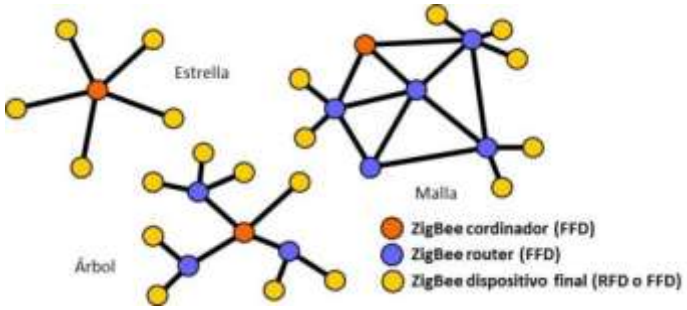


Figura 4. Ejemplos de topología en Zigbee.

No obstante, se trata de un protocolo de red con cierta complejidad, que implica tener una red muy definida y controlada. Esto lo hace una tecnología muy versátil, pero por otro lado complica sustancialmente la creación y mantenimiento de la red. Además del corto alcance que ofrece, al operar normalmente en la red de 2,4 GHz. Pensando en la aplicación que nos concierne, la parte de un consumo eficiente y reducido de energía optimizado para dispositivos IoT la tendríamos resuelta, pero no la parte de amplio alcance. Es por ello por lo que tampoco se considerará la tecnología ZigBee.

4.1.4. Thread

Figura 5. Logo Thread.

Thread es una tecnología de red inalámbrica de baja potencia. Fue desarrollado por The Thread Group, compuesto por marcas como Yale Security, Silicon Labs, Samsung Electronics, Nest Labs, Freescale Semiconductor y ARM. Con el objetivo de proporcionar una comunicación fluida entre los productos, permitiendo que los dispositivos conectados en el Internet de las Cosas se descubran y comuniquen entre sí. Finalmente lanzado al mercado en 2015. Desde 2019 es respaldado por la Connectivity Standards Alliance [7].

Está basado en la tecnología Zigbee. Por lo que tienen muchas características en comunes, como el uso de la frecuencia 2,4 GHz y el tipo de topología malla. Sin embargo, implementa el protocolo IPv6 para tener una conexión más natural con otras redes. Además, no requiere tener un coordinador de la red malla como en Zigbee, si no que se necesitará un Thread Borde Router, que será un dispositivo que tendrá la función de unir la red Thread con las demás redes. Puede haber más de un Thread Borde Router en una misma red.

Fue diseñado con el propósito de ser un protocolo de radio de muy bajo consumo. Sin embargo, al tener el protocolo IPv6 y otras funciones requieren más recursos en los dispositivos como por ejemplo, tener más memoria, en comparación con otras tecnologías como Zigbee.

Al ser una tecnología inalámbrica relativamente nueva, aún a día de realizar este proyecto no hay mucha variedad de dispositivos, por lo que se dificulta crear una amplia red de dispositivos inteligentes. Aunque poco a poco cada vez más empresas adoptan Thread en sus nuevos productos.

4.1.5. Matter

Figura 6. Logo y certificado Matter.

Matter no es una nueva tecnología inalámbricas, sino un coordinador entre varias tecnologías inalámbricas. Su objetivo es permitir la interoperabilidad entre dispositivos de diferentes fabricantes y de diferentes tecnologías inalámbricas facilitando la vida del usuario.

Matter es un estándar de código abierto desarrollado y promovido por la Conectivity Standard Alliance [7]. Fue anunciado oficialmente en la primera mitad de 2022. Fue creado con el objetivo de unificar la industria y permitir una comunicación fluida entre dispositivos IoT de todas las plataformas y fabricantes.

Así este protocolo se presenta como una capa adicional a los estándares existentes tales como Ethernet, WiFi, Bluetooth o Zigbee, entre otros. Funcionamiento en local, por lo que no hará falta que ciertos dispositivos inteligentes tengan que conectarse a un servido en Internet. Además, permite una compatibilidad con varios hubs, es decir, cada dispositivo debe poder manejarse, por ejemplo, desde Google Home, Alexa y Home Assistant al mismo tiempo.

Aunque Matter es una tecnología prometedora que tiene el potencial de simplificar significativamente la domótica. actualmente aún es una tecnología relativamente nueva y cuesta hacer que esa interoperabilidad funcione de forma fluida y estable.

4.1.6. Z-Wave

Z-Wave es un protocolo de comunicaciones inalámbricas utilizado principalmente para la automatización del hogar y la oficina [10]. Fue diseñado por la compañía danesa Zen-Systems en 1999 con el objetivo de proporcionar una solución para la creciente demanda de una tecnología inalámbrica eficiente para dispositivos IoT.

Figura 7. Logo Z-Wave.

Con el objetivo de promover la tecnología en 2005 cinco empresas formaron la Z-Wave Alliance [11]. Más tarde en 2008, Zen-Systems fue comprada por la empresa Sigma Designs. Desde entonces, Z-Wave Alliance es un consorcio global de empresas que utilizan Z-Wave en sus productos.

Utiliza una topología tipo malla, semejante a la de Zigbee, donde un dispositivo puede actuar como repetidor para transmitir las señales de otros dispositivos. Hace uso de las bandas de frecuencia 868 MHz en Europa y 900 MHz en EEUU. Esta frecuencia es menos propensa a la interferencia que las bandas de 2,4 GHz utilizadas por Zigbee y WiFi. Con la frecuencia de 969 MHz proporciona un amplio alcance, incluso a través de las paredes. Lo que, en combinación con una red de malla, da como resultado una conectividad fiable en un amplio rango.

Se considera una tecnología segura que utiliza señales cifradas mediante la encriptación AES-128 [8]. Y tiene un bajo consumo de energía, lo que lo hace ideal para dispositivos que funcionan con baterías.

4.1.7. Low-Power Wide-Area Networks

En la última década, el desarrollo del Internet de las Cosas (IoT) ha impulsado el surgimiento de nuevas tecnologías de comunicación inalámbricas como las redes *Low Power Wide Area Networks* (LPWAN). Dichas tecnologías han surgido como una necesidad de cubrir las carencias que presentan las tecnologías tradicionales como el WiFi o las redes móviles.

LPWAN es un tipo de red inalámbrica diseñada para proporcionar conectividad de larga distancia y bajo consumo de energía para dispositivos IoT que requieren comunicación de datos de baja tasa de transferencia. Surgiendo por la necesidad de conectar dispositivos IoT de manera eficiente y rentable en áreas geográficas amplias.

Siendo así el objetivo principal de LPWAN de habilitar la conectividad de dispositivos IoT que envían pequeñas cantidades de datos de manera periódica, como sensor de temperatura, medidores inteligentes, sistemas de riego agrícola, monitoreo ambiental, seguimiento de activos, entro otros. Estos dispositivos pueden ser ubicados en áreas remotas o de difícil acceso, ofreciendo LPWAN una solución confiable y de bajo costo.

Generalmente pueden enviar y recibir paquetes de datos de 10 a 1000 bytes a unas velocidades de entre 100 bps y 200 kbps y unos rangos van de 2 kilómetros hasta 1000 kilómetros, según diversos factores ambientales y técnicos. La mayoría de las redes de este tipo tiene una topología de estrella, similar al WiFi, donde cada dispositivo, comúnmente denominados *endpoint*, se conecta directamente a un punto central de acceso denominado *gateway,* o pasarela en español. Las redes LPWAN no ofrecen una conectividad IP punto a punto, por lo que, aunque el dispositivo final de los paquetes de datos sea una red basada en el protocolo TCP/IP, como lo es la red de Internet, el *endpoint* que envía los paquetes no hará uso de una conexión directa TCP/IP. Por lo tanto, el paquete tendrá que ser decodificado y convertido a través del *gateway.*

La principal característica de esta tecnología, su gran rango de cobertura respetando un bajo consumo, solo puede conseguirse a un precio, y es el de disminuir el ancho de banda disponible y por lo tanto la cantidad de datos que puede ser transmitida. Esto implica que no podremos trasmitir contenido multimedia en alta calidad como si podemos hacer usando WiFi o redes móviles modernas 4G/LTE, ni audio de alta fidelidad como si nos permite Bluetooth. Pese a esto, esta tecnología es ideal si tan solo necesitamos transmitir unos pocos bytes, como por ejemplo lo que necesita un sensor de temperatura o un interruptor inteligente, y esto es precisamente lo que nos interesa para la aplicación de nos atañe.

LPWAN está compuesta por un conjunto de diferentes tecnologías y protocolos de comunicación específicos siguiendo la misma filosofía, como LoRaWAN, NB-IoT y Sigfox. Estas tecnologías operan en bandas de frecuencia licenciadas o no licenciadas.

LoRaWAN

LoRaWAN fue desarrollado por la empresa Cycleo [12], con sede en Grenoble, Francia [12]. Posteriormente Semtech adquirió Cycleo y todas sus propiedades, empresas y operaciones en 2012 [13]. LoRaWAN ue diseñado para proporcionar una solución de conectividad inalámbrica eficiente y rentable para dispositivos IoT distribuidos en áreas amplias y de difícil acceso. Tiene como objetivo ofrecer una conectividad confiable y escalable para aplicaciones de IoT en diversos sectores, como agricultura, ciudades inteligentes, monitorización ambiental, gestión de activos, entre otros.

Figura 8. Logo Lora.

LoRaWAN viene de LoRa Wide Area Networt. Donde Lora hace referencia a la modulación inalámbrica apta para aplicaciones long-range low power low datarate desarrollada por Semtech. Siendo una especificación para redes de baja potencia y área amplia, LPWAN.

En Enero de 2015 se fundó LoRa Alliance [14] como resultado de la creciente popularidad y demanda de la tecnología LoRaWAN. Siendo una organización sin ánimo de lucro para promover y estandarizar la tecnología LoRaWAN y su adopción en aplicaciones en el Internet de las Cosas. Los objetivos claves de la organización son:

- Fomentar la colaboración entre diversas organizaciones y empresas a lo largo del mundo que estén interesadas en el desarrollo y despliegue de la tecnología LoRaWAN, generando un marco para el intercambio de conocimientos entre los miembros.
- Establecer estándares a nivel global para la tecnología LoRaWAN, asegurando la interoperabilidad y la compatibilidad entre diferentes soluciones y proveedores. Siendo esto fundamental para el crecimiento y adopción de la tecnología en el mercado.
- Promover la adopción de la tecnología, educar sobre sus beneficios y usos en diferentes industrias. Fomentando el despliegue de soluciones basadas en LoRaWAN.

Lora Alliance recoge las especificaciones del protocolo red de LoRaWAN en el documento *LoRaWAN Specification* [15] que define los detalles y limitaciones del protocolo, así como la manera de implementarlo y trabajar con él. Se define las capas 1, 2 y 3 del modelo OSI. Siendo la primera capa correspondiente a la implementación física LoRa, con sus parámetros físicos

tales como la frecuencia de transmisión, tipo de modulación, etc. Mientras que las capas 2 y 3, correspondientes a las capas MAC y NET respectivamente, definen el direccionamiento físico y la distribución ordenada de las tramas y flujo de red, y el enrutamiento de los paquetes dentro de la red.

La tecnología LoRaWAN se basa en un protocolo de red libre y altamente flexible, lo que permite a los usuarios implementar sus propios protocolos de red personalizados aprovechando la tecnología de modulación LoRa. Esto facilita la creación de redes privadas de manera sencilla y adaptada a las necesidades específicas de cada usuario.

La arquitectura de LoRaWAN se basa en una topología de red en estrella (Figura 9) en la cual los nodos de aplicación o sensores se conectan a una o varias pasarelas o gateways utilizando la modulación de radio frecuencia LoRa. Estas pasarelas están a su vez conectadas al servidor de red LoRaWAN mediante el protocolo TCP/IP convencional. En esta configuración, el network server es responsable de seleccionar qué paquete recibir de las diferentes pasarelas y cuáles descartar, asegurando así una óptima transmisión de datos.

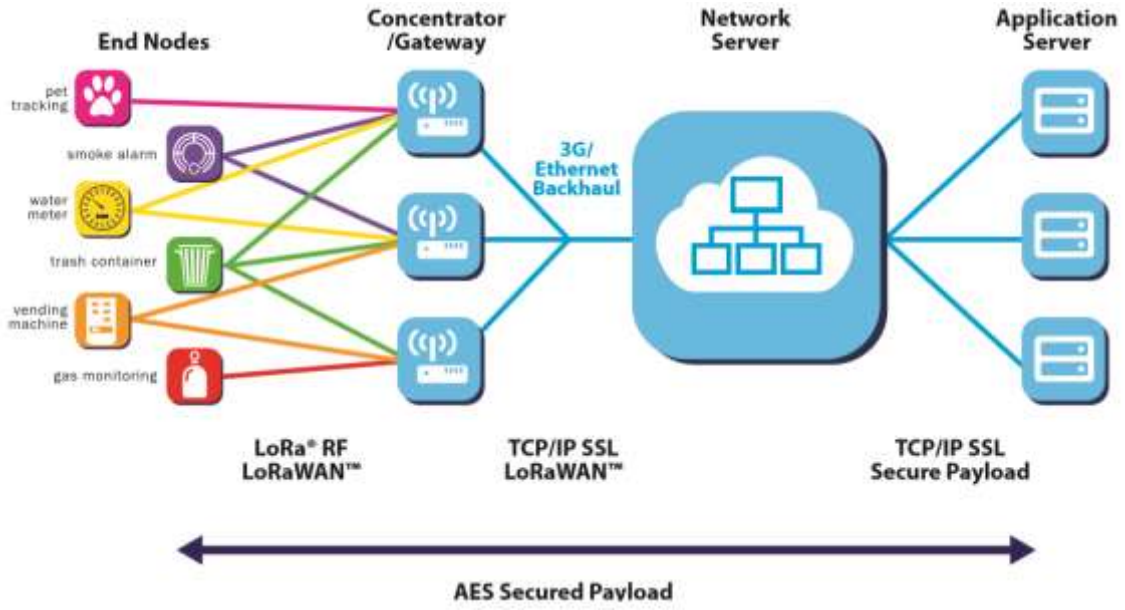


Figura 9. Topología LoRaWAN.

Esta arquitectura en estrella simplifica la conectividad de los dispositivos IoT, ya que los nodos solo necesitan comunicarse con los *gateway* más cercanos, y no entre sí directamente. Los *gateway* actúan como puntos de acceso que recopilan los datos de los nodos y los envían al servidor de red, *Network Server*, donde se procesan y gestionan. Esta estructura centralizada permite una gestión eficiente de la red y facilita la escalabilidad, ya que se pueden agregar o eliminar nodos y pasarelas según sea necesario.

Sigfox

Sigfox es una empresa francesa fundada en 2009 con el objetivo de crear una red global dedicada al Internet de las Cosas que pudiera proporcionar una cobertura amplia y eficiente para una variedad de dispositivos conectados. Surgiendo así la tecnología Sigfox basada en la comunicación de dispositivos a través de pequeños mensajes de datos transmitidos a través de ondas de radio, siguiendo la misma filosofía de las redes LPWAN.

Figura 10. Logo Sigfox.

Sigfox está basada en la modulación DBPSK (modulación por desplazamiento de fase) para la subida de paquetes hacia el servidor (uplinks), y en una modulación GFSK (modulación por desplazamiento de frecuencia) para los downlinks o envíos desde la aplicación hacia los dispositivos. Con esto, Sigfox funciona muy bien transmitiendo datos desde sensores hacia la plataforma, pero sin ser eficaz al revés. Se trata por tanto de una tecnología claramente enfocada a sensores que solo requieran de envíos de datos hacia la nube y no requieran de gran ancho de banda para recepción. No obstante, esto no quiere decir que no lo permita.

Trabaja transmitiendo sobre las bandas sub-Ghz de licencia libre, 868 MHz para Europa y 900 Mhz en EEUU, al igual que LoRaWAN, y hace uso de la comunicación *Ultra Narrow Band.* Sigfox limita tanto el tamaño de los paquetes de información que puedes transmitir, como el número de paquetes que te permite enviar por día por normativa, ya que, al trabajar en un espectro de frecuencia sin licenciar, la ley establece un máximo de uso de red. En cuanto al tamaño del paquete, Sigfox lo limita a 12 bytes de información, que junto con 12 bytes más correspondientes a un header e identificador de dispositivo y otros metadatos más, componen un total de 25 bytes por mensaje u envío. El límite de envíos diarios está fijado en 140 mensajes tardando una media de 2 segundos en estar disponibles en la plataforma, es decir, que la velocidad de envío está fijada en unos 100 bps para envíos, y bastante más lentas para la subida de información.

El alcance teórico varía entre los 30 y 50 km en entornos rurales y hasta 10 km en entornos urbanos.

La infraestructura para utilizar la red la provee la empresa propietaria, Sigfox. Esto engloba tanto las infraestructuras físicas como antenas, *gateway,* routers y repetidores, como los servidores para la propia red y la plataforma de gestión en la nube y, por lo tanto, dispone de una tarifa de suscripción que en el momento de la redacción de este proyecto ronda los 3€ anuales [16], aunque esto varía según la cantidad de dispositivos que los clientes deseen conectar, pudiendo llegar a los 9€ al año. Con esto se puede apreciar semejanzas al modelo usado por las compañías de telefonía móvil, aplicando una tarifa mensual a sus clientes por conectarse y usar su red.

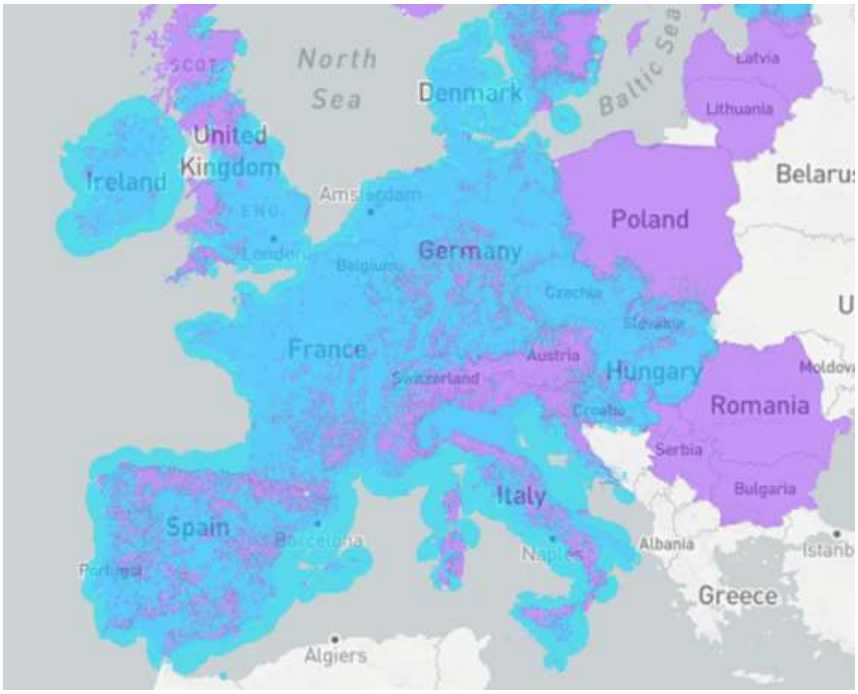


Figura 11. Mapa de cobertura de Sigfox en Europa.

La compañía ha desplegado su red en varios países y continúa expandiéndose a nivel global (Figura 11). Sigfox ha sido utilizada en diversas aplicaciones, como seguimiento de activos, gestión de residuos, agricultura inteligente y monitorización de infraestructuras, entre otras.

NarrowBand-IoT

NarrowBand – Internet of Things (NB-IoT) fue desarrollado por 3GPP para habilitar servicios en un amplio abanico de dispositivos. Diseñada siguiendo la filosofía LPWAN para una comunicación eficiente entre dispositivos distribuidos, con una gran duración de baterías y enfocado tanto para aplicaciones urbanas como agrícolas. Esta tecnología fue estandarizada en junio de 2016.

Figura 12. Logo NB-IoT.

NB-IoT opera en un espectro de frecuencias licenciadas para garantizar la operabilidad del servicio y está basada en la red celular LTE, ofreciendo una cobertura adicional de 20 dB sobre el GSM. Se puede entender como la respuesta de los operadores móviles al futuro de las comunicaciones IoT. Trabaja en las mismas frecuencias e infraestructuras existentes para la tecnología LTE.

Siguiendo una filosofía similar a Sigfox y las compañías de telefonía móvil de suscripción de pago para hacer uso de la infraestructura de red, pero operando bajo el estándar de LTE. Los diversos dispositivos o sensores precisan de una tarjeta de identificación SIM, como la que utilizan los teléfonos móviles, que les brinda una conexión bidireccional cifrada y pretenden convivir hasta 100.00 dispositivos en un único repetidos.

La ventaja de esta tecnología es precisamente el uso de la red GSM o LTE ya existente y la operatividad en una frecuencia licenciada evita las colisiones de envio que se pueden producir en otras tecnologías LPWAN de banda en espectros de libre licencia.

Actualmente existen diversos proyectos enfocados en las *Smart Cities,* o ciudades inteligentes en español, haciendo uso de esta tecnología. Como el proyecto valenciano *TSwasTe* [17], un sensor de llenado de contenedores de bajo coste diseñado para facilitar la gestión de residuos urbanos mediante la monitorización de diferentes parámetros del contenedor.

4.2. Elección del sistema central

A continuación, se explorará diferentes opciones que ofrece el mercado para encontrar el más idóneo que cumpla la función el dispositivo de control central de un edificio inteligente. La elección del hardware es un aspecto crítico para garantizar el correcto funcionamiento de todo el sistema. Es por ello por lo que se evaluará diferentes aspectos clave como las prestaciones, el rendimiento, la eficiencia energética o el coste.

4.2.1. Microcontrolador Arduino

Arduino no es un microcontrolador en sí, si no una plataforma de desarrollo basada inicialmente en un microcontrolador de la familia AVR de Atmel (ahora Microchip). Arduino es un proyecto, compañía y comunidad con la filosofía *open source* y *open hardware* para acercar y facilitar el uso de la electrónica y la programación de sistemas embebidos [18].

Figura 13. Logo Arduino.

Arduino fue concebido en 2005 como un programa para estudiantes en el Interaction Design Intitute en Ivrea, Italia, con el objetico de proporcionar una forma fácil y económica a estudiantes y profesionales para crear dispositivos que pudieran interactuar con su entorno mediante sensores y actuadores. Arduino surge a partir del proyecto *Wiring* y el proyecto original se compone de una placa electrónica programable basada en un microcontrolador con la circuitería necesaria para ser programado fácilmente a través de un ordenador y un entorno

de desarrollo integrado (IDE) con una biblioteca de funciones sencillas. Además de facilitar la conexión de sensores, periféricos y actuadores gracias a sus conectores universales.

La versatilidad del proyecto reside en su comunidad, y es que, gracias a ser de código libre, actualmente existen cientos de placas compatibles y programables mediante Arduino siguiendo esa misma filosofía de acercar la electrónica a las personas no familiarizadas con la ingeniería. Siendo *Arduino Uno* (Figura 14) la placa de desarrollo más conocida, usada y replicada.

Figura 14. Placa Arduino Uno.

Está basada en el chip *ATmega238P*, un microcontrolador CMOS de 8 bits de la familia AVR de Atmel (ahora Microchip), basado en la arquitectura RISC (*Reduced instruction set computing*). No es el más potente de la familia Arduino, pero sí el más equilibrado en cuanto a relación pecio/potencia. Dispone de 131 instrucciones, 32Kbytes de memoria flash para almacenamiento de programas, con capacidad de programación de un bootloader, y 2Kbytes de memoria RAM. Además de disponer una cantidad muy interesante de periféricos como 3 timers (dos de 8bits y otro de 16), un reloj de tiempo integrado, un convertidor analógico a digital de 8 canales con 10 bits de resolución capaz de hacer quince mil conversiones por segundo, buses SPI e I2C y una UART. También dispone de 23 pines de propósito general que pueden ser configurados como entrada o salida, dos de ellos con capacidades de generar interrupciones externas por hardware y otras muchas más configuraciones y capacidades que pueden ser encontradas en la hoja de datos correspondiente. La placa Arduino Uno oficial puede ser comprada a un precio muy asequible de 20€.

Arduino sigue creciendo y actualmente dispone de muchísima variedad de placas de desarrollo oficiales y no oficiales compatibles con su IDE y bibliotecas. Recientemente con el auge del Internet de las Cosas el desarrollo de Arduino no se queda atrás y dispone de unas decenas de placas y módulos con conectividades inalámbricas como WiFi (MKR 1010), Sigfox (MKR 1200), LoRa (MKR 1300) o Narrow Band IoT (MKR 1500), entre otras. Estas placas de desarrollo están basadas en procesadores ARM de la familia SAM, permitiendo un bajo consumo energético con modos *sleep* muy eficientes y transceptores de radio frecuencia inalámbrica que permiten disponer de un sensor comunicándose a la red de manera fácil, rápida y barata.

Sin embargo, para el proyecto que nos concierne, esta placa de desarrollo se queda corta en prestaciones para cumplir con el cometido de ser el controlador central de un edificio inteligente.

4.2.2. Computador personal

Los ordenadores personales llevan con nosotros varias décadas habiendo generaciones que han crecido desde pequeños con ordenadores en su entorno. El uso de los computadores personales, PC por sus siglas en ingles de *Personal Computer*, se popularizó en la década de 1980. Periodo donde las PC se volvieron más accesibles y comenzaron a utilizarse en hogares y oficinas en todo el mundo. Varios factores contribuyeron a su popularidad, como la reducción de costos, el desarrollo de interfaces de usuario más amigables y entendibles por las personas poco familiarizadas con la informática y la disponibilidad de software diverso y variado útiles para un sinfín de campos y sectores.

En esta popularización de las PC ciertas compañías jugaron un papel clave. IBM, con su primer modelo IBM Personal Computer lanzado en 1981, fue de las primeras PC ampliamente adoptadas y estableció el estándar de muchas características que se utilizan hasta el día de hoy. Apple también tuvo un impacto significativo con el lanzamiento del Apple II en 1977, y más tarde, con la introducción de la serie Macintosh en 1984. Estas computadoras ofrecieron interfaces gráficas de usuario intuitivas convirtiéndose en un referente en términos de diseño y facilidad de uso. Y Microsoft con el desarrollo del software MS-DOS en la década de 1980 se convirtió en el sistema operativo estándar. Y con el lanzamiento de Windows en 1985 se convirtió en el sistema operativo dominante en las PC y contribuyó a su crecimiento. Estas son solo algunas de las compañías que desempeñaron un papel clave en la popularización de las PC en la década de 1980. Sin embargo, a lo largo de los años, muchas otras empresas han contribuido al desarrollo y la evolución de las computadoras personales.

En la actualidad existe un gran abanico de opciones de diferentes compañías y fabricantes. A grosso modo se divide en tres grandes grupos según su precio, calidad y prestaciones: gama baja, gama media y gama alta. Para nuestro cometido nos centraremos en que características podemos encontrar en el mercado de la gama baja.

En general, un PC de gama baja en calidad-precio es concebido para realizar tareas básicas como navegación web, correo electrónico, procesamiento y edición de texto y reproducción de multimedia. Sin embargo, presenta limitaciones en rendimiento y capacidad a la hora de ejecutar aplicaciones más exigentes, como diferentes softwares del ámbito de la ingeniería, edición de videos o de populares videojuegos. Siendo las principales características comunes a un PC de gama baja:

- Procesador: probablemente encontremos modelos como un Intel Celeron, un Intel Atom, un AMD Athlon o AMD Series. Con una velocidad de reloj en torno al rango de 1,6 GHz y 3,0 GHz y entre 2 o 4 núcleos.
- Memoria RAM: generalmente se encuentra entre 4GB o 8 GB.
- Almacenamiento: Es común encontrar un disco duro tradicional (HDD) en lugar de una unidad de estado sólido (SSD).
- Conectividad y puertos: comúnmente encontramos los puertos esenciales, como USB, HDMI, Ethernet. Y de conectividad inalámbrica básica como Bluetooth y WiFi.

Para nuestro acometido en este proyecto, un PC de gama baja nos cubriría nuestras necesidades de prestaciones. Sin embargo, en relación con el consumo y eficiencia energética no es la mejor opción como veremos más adelante. Pues un PC de gama baja presenta un consumo de energía entorno a los 200 o 400 vatios.

4.2.3. Mini PC

Un mini PC es una computadora de tamaño reducido que integra todos los componentes esenciales en un espacio compacto. Estos dispositivos son diseñados para ocupar menos espacio y ofrecer una solución de computación versátil y portátil.

Los mini PC surgieron en la década de 2000 como una alternativa a las computadoras de escritorio tradicionales. A medida que la tecnología avanzaba y los componentes se volvían más pequeños y eficientes, los fabricantes comenzaron a desarrollar computadoras de menor tamaño sin sacrificar demasiado rendimiento.

Figura 15. Mini PC de la marca Intel.

Los mini PC se han popularizado en aplicaciones específicas como reproducción de multimedia, servidores de archivos, estaciones de trabajo compactas, etc. Las principales características que han permitido esta popularización son:

- Ahorro de espacio y portabilidad: Su tamaño compacto permite colocarlos fácilmente en escritorios pequeños, estanterías o incluso montarlos detrás de un monitor. Además, por ese mismo motivo son fáciles de transportar.
- Conectividad: a pesar de su reducido tamaño suelen ofrecer una amplia variedad de puertos y conectividad esenciales como USB, HDMI, Ethernet, WiFi y Bluetooth.
- Eficiencia energética: Utilizan componentes de bajo consumo energético, lo que le permite tener un rango de consumo de entre 50 y 100 vatios, habiendo algunos modelos diseñados específicamente para ser más eficientes. En lo que se traduce en una factura de electricidad más baja y una menor huella de carbono.

En relación con el proyecto que nos atañe, un mini PC sería una buena opción, cubre las necesidades en prestaciones teniendo un consumo energético razonablemente bajo. Sin embargo, aún hay otras opciones que nos cubriría de igual forma estas necesidades a un precio más reducido. Pues en el mercado encontramos que un mini PC está entre 400 € a 1000€.

4.2.4. Raspberry Pi

La Raspberry Pi es una plataforma de computación de placa única (Single-Board Computer, SBC) que, si bien tiene características y capacidades comunes a una computadora, generalmente no se considera un mini PC en el sentido tradicional. Puesto que su enfoque está más destinado a proyectos de electrónica y aprendizaje y no tanto en puestos de escritorio.

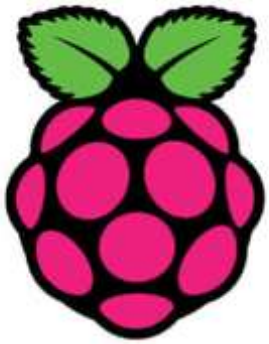


Figura 16. Logo de la Fundación Raspberry [19].

La primera Raspberry Pi, conocida como Raspberry Pi Model B, fue lanzada en febrero de 2012 por la Fundación Raspberry Pi [20]. Sin embargo, el proyecto comenzó en 2006 por el profesor Eben Upton y sus colegas en la universidad de Cambridge con el objetivo de fomentar la enseñanza de la informática en las escuelas y brindar una plataforma asequible para el aprendizaje y la experimentación.

Figura 17. Rasberry Pi 4.

Por esta razón, Raspberry Pi ofrece una opción de reducido coste. En torno a 100€ a día de realización de este proyecto para el modelo Raspberry Pi 4 (Figura 17) de 4 GB de RAM de tecnología SDRAM DDR3, un procesador Cortex de 1,5 GHz y 4 núcleos y con los puertos esenciales USB 2.0 y 3.1, Ethernet y conectividad WiFi y Bluetooth. Con unas dimensiones de 10 x 7 x 3 cm y 50 gramos, haciendo de un producto bastante versátil. Y un consumo de 2,7 vatios cuando está inactiva o con una carga de trabajo baja, pudiendo ascender hasta 5 vatios o más según la intensidad de tareas o de los componentes periféricos conectados.

Este se presenta como una opción idónea para el acometido del proyecto cubriendo las necesidades en prestaciones, eficiencia energética con un consumo bastante bajo, lo que repercute en la reducción de la factura de electricidad y en la huella de carbono, y además a unos costes muy razonables.

4.3. Elección de plataforma de software.

Cuando nos referimos a una plataforma de software, hablamos de la plataforma domótica como un sistema integrado que permite la gestión y control automatizado de diversos dispositivos y sistemas del hogar. El término “domótica” proviene de la combinación de las palabras “domus” (casa en latín) y “automática”.

Podemos encontrarnos una diversidad de opciones. Las más conocidas son las plataformas domóticas de Google Home, Amazon Alexa y la HomeKit de Apple. Sin embargo, estas tres quedan totalmente descartadas al requerir necesariamente de la conexión a internet, lo que no cumpliría los objetivos de este proyecto.

4.3.1. Domoticz

Domoticz es un software de código libre publicado por primera vez en 2012 [21]. Fue desarrollado en el lenguaje C++. Está disponible para las plataformas Windows y Linux. Es uno de los primeros softwares de este tipo por lo que tiene un amplio recorrido.

Figura 18. Logo Domoticz.

Domoticz se caracteriza por consumir muy pocos recursos del sistema. Ofrece soporte a una variedad de protocolos domóticos como Z-Wave, EnOcean o X10. Así como una buena integración con diferentes dispositivos inalámbricos como miniestaciones meteorológicas o cámaras IP.

Cuenta además con una interfaz web y aplicaciones móviles que convierten su consumo multiplataforma en una tarea sencilla.

Aunque la interfaz está algo desactualizada respecto a las tendencias actuales y, lo que es más importante, los diferentes módulos que lo componen están poco desacoplados del “core” del sistema, lo que complica el desarrollo de nuevos módulos y funcionalidades.

Además, la documentación es bastante escasa en comparación con otras alternativas, por lo que puede dificultar su uso a nuevos usuarios con poca experiencia.

4.3.2. OpenHab

OpenHab fue fundada por Kai Kreuzer, quien es el presidente de la fundación y también el fundador de OpenHAB Fundation [22]. Su desarrollo comenzó en 2010 y finalmente se lanzó en 2016. La versión más actualizada es OpenHab 4.0 lanzada el 23 de Julio de 2023.

Figura 19. Logo OpenHab.

Es un software de código abierto disponible en gran variedad de plataformas como Windows, Linux, sistemas embebidos como NAS, etc. Está escrito en Java [23].

Su diseño se basa en una arquitectura totalmente modular donde tenemos el *core* del sistema y sobre él se construye alrededor los diferentes módulos que le añaden una diversidad de funcionalidades. Se puede integrar fácilmente con KNX, ZWave, X10, el termostato Nest, bombillas HUE, bases de datos, servicios Web como IFTTT o una API que nos sirva el tiempo y la previsión.

Tiene un buen diseño visual y consta de bastante documentación, tanto del Core como de los módulos accesorios. Ofrece una variedad de opciones para crear reglas de automatización. Te permite personalizar la interfaz para adaptarla a las necesidades del usuario. Sin embargo, se requiere de conocimientos de programación. Es importante mencionar que aunque OpenHAB tiene una interfaz de usuario avanzada y potente, también puede ser más compleja de usar nuevos usuarios. Cuenta con aplicaciones oficiales para iOS y Android.

Precisamente para su configuración cuenta con un entorno preparado que deriva de Eclipse (totalmente multiplataforma) y que busca facilitar la vida al instalador /configurador del sistema.

Cuenta con una comunidad que se reúnen alrededor del foro oficial y que en muchos casos es la referencia oficial para solucionar los errores con los que nos encontremos y que resaltan las carencias de la documentación en algunos apartados.

4.3.3. HomeBridge

HomeBridge es una plataforma que permite integrar dispositivos de hogar inteligente que no son compatibles de forma nativa con HomeKit. Puedes controlar tus accesorios HomeBridge usando tu iPhone, iPad, Apple Watch, Mac, voz o centro de hogar, y usar más de 2,000 complementos para conectar con varios accesorios inteligentes [24].

Interfaz de usuario simple y fácil de usar para administrar complementos, configuraciones y accesorios.

Figura 20. Logo Homebridge.

Proporciona una imagen prehecha para Raspberry Pi que permite tener HomeBridge de forma local en la red privada. Esto significa que los datos de los dispoticos se comunican directamente con HomeBridge en la red local. Sin embargo, para acceder a los dispositivos desde fuera de la red local es necesario el uso de HomeKit a través de un dispositivo Apple como un iPhone o un iPad configurado como centro de hogar o HomePod.

Es un software destinado al ecosistema de dispositivos de Apple, por lo que queda descartado.

4.3.4. Gladys Assistant

Gladys Assistant es un software de código abierto desarrollado por Mathieu Andrade [25]. Su versión 4 se publicó en Noviembre de 2020 siendo el resultado de casi 2 años de trabajo por parte de toda su comunidad.

Su desarrollo tenía por objetivo proporcionar una solución moderna, centrada en la privacidad de la automatización del hogar. Pretende ofrecer una experiencia del hogar inteligente intuitiva y personalizable, manteniendo al mismo tiempo la seguridad y la privacidad de los datos del usuario [26].

Figura 21. Logo Gladys Assistant.

Dispone de una aplicación web progresiva (PWA) que puede instalarse en cualquier dispositivo: Android, iOS, MacOS, Windows, Linux. Sin embargo, no se encuentra en las tiendas de aplicaciones de Android o iOS. En su lugar hay que seguir las instrucciones de plus.gladysassistant.com para agregarla a la página de inició del navegador.

Te permite crear automatizaciones y escenas a través de una interfaz de usuario limpia e intuitiva. Además, Gladys es un asistente inteligente con el que se puede hablar a través de la interfaz o de la integración de Telegram.

Sin embargo, a diferencia de otras soluciones. Las integraciones disponibles se limitan a las que Gladys Assistant incorpora de serie, que son: Zigbee2Mqtt, Philips Hue, Sonoff, Camera, Xiaomi y TP-Link [26]. Dado que Gladys Assistant es un proyecto de código abierto, existe la posibilidad que se puedan agregar más integraciones en el futuro.

Estas limitaciones dificultan una implementación con LoRaWAN.

4.3.5. Home Assistant

Home Assistant es una plataforma de software de código abierto para la automatización del hogar. Su principal objetivo es permitir a los usuarios controlar y monitorear una variedad de dispositivos y servicios domésticos inteligentes de forma local, es decir, sin recurrir a infraestructuras o servicios de terceros. [27]

Figura 22. Logo Home Assistant

Paulus Schoutsen [28] inició el proyecto como una aplicación para Python en septiembre de 2013 y publicado por primera vez en Github [29] en noviembre de 2013. Desde entonces, la plataforma ha experimentado un crecimiento significativo y ha ganado una comunidad activa de usuarios y desarrolladores.

Home assistant se puede ejecutar en diversas plataformas desde ordenadores personales hasta en contenedores Docker, incluyendo en Raspberry Pi, y más. Permite a los usuarios crear automatizaciones, scripts, escenarios e integrar y controlar una infinitud de dispositivos inteligentes, como luces, termostatos, cámaras, altavoces, sensores… También es compatible con una amplia gama de protocolos de comunicación como MQTT, Zigbee, Z-Wave y servicios en la nube.

En tema de seguridad, de forma predeterminada no presta acceso remoto. Los datos se almacenan únicamente en el propio dispositivo lo que garantiza la privacidad del usuario. Y los usuarios se pueden proteger mediante autenticación de dos factores evitando el acceso de un atacante incluso si conoce la contraseña del usuario.

A la hora de instalar Home Assistant nos podemos encontrar con que hay 3 versiones:

- **Home Assistant Core**. Es el software que contiene todas las funciones básicas de Home Assistant. Puede ser ejecutado en Docker o en emuladores de Python.
- **Home Assistant Supervisor**. Añade más funciones y da soporte a un ecosistema de complementos, también conocidos como Add-os.
  No está soportado por Docker.
- **Home Assistant SO**. Es un sistema operativo integrado y minimalista diseñado específicamente para ejecutar el ecosistema de Home Assistant. Puede funcionar

tanto en miniordenadores de forma nativa, como de forma virtualizada usando VMware, Virtual Box, Proxmox o cualquier sistema de virtualización.

Cabe señalar algunos inconvenientes de aportar esta solución. El primer lugar es que el usuario tiene que asumir la responsabilidad de mantener y actualizar su sistema. En segundo lugar, el usuario tiene que hacer frente a posibles problemas como fallos de hardware, problemas de red, problemas de incompatibilidades de versiones o de software, cortes de electricidad o ataques mal intencionados. Por lo que puede requerir conocimientos técnicos y tiempo.

Debido a su gran versatilidad lo convierte en una solución idónea para el proyecto que nos concierne.

4.4. Elección de Protocolo de Comunicación IoT

En este apartado, exploraremos varios protocolos de mensajería de datos para dispositivos IoT. A través de un análisis detallado de cada uno, buscaremos determinar cuál de estos protocolos sería el más adecuado para nuestro proyecto de edificios inteligentes.

4.4.1. Apache Kafka

Apache Kafka es una plataforma distribuida para la transmisión de datos que permite publicar, almacenar y procesar flujos de eventos de forma inmediata, y administrar los flujos de datos de varias fuentes y enviarlos a distintos usuarios [30].

Figura 23. Logo de Kafka.

Kafka comenzó como un sistema interno que LinkedIn desarrolló en 2011 para gestionar 1,4 billones de mensajes por día. Ahora, es una solución open source de transmisión de datos que permite satisfacer diversas necesidades empresariales. Siendo ideal para abordar los desafíos relacionados con el big data, por su capacidad de gestionar millones de datos por segundo [31].

Kafka almacena mensajes de clave-valor que provienen de un número arbitrario de procesos llamados "producer". Los datos pueden dividirse en diferentes particiones, “partition” dentro de diferentes temas, “topics”. Otros procesos denominados “consumer” pueden leer los mensajes de las particiones (Figura 24).

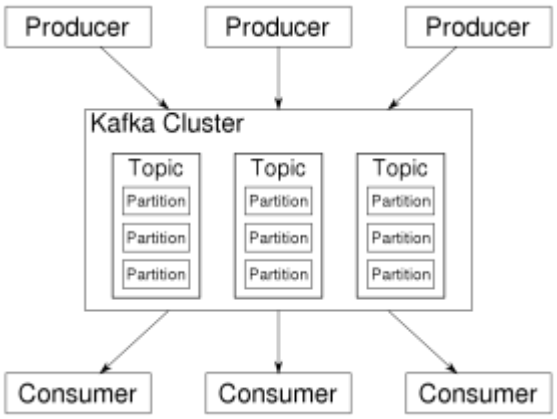


Figura 24. Visión general de Kafka.

Algunas características por resaltar son que Kafka es resistente a fallos de nodos, debido a que los datos se replican en el clúter. Alta escalabilidad al permitir que las particiones de distribuyan en diferentes servidores. Y alta velocidad al poder manejar millones de datos por segundo.

Aunque Apache Kafka es una plataforma de transmisión de datos en tiempo real poderosa, con capacidad de manejar grandes volúmenes de datos. Para un proyecto como el que nos concierne es probable que no se genere suficientes datos como para justificar el uso de una herramienta tan robusta y compleja como Kafka. Además, la configuración y el mantenimiento de Kafka pueden requerir un conocimiento técnico avanzado.

4.4.2. Constrained Application Protocol

Constrained Application Protocol o Protocolo de Aplicación Restringida (CoAP) es un protocolo de la capa de aplicación de internet diseñado para dispositivos con recursos limitados, como en el caso de los dispositivos IoT. Permite que estos dispositivos se comuniquen con cualquier nodo de Internet.

Fue desarrollado por la IETF (Internet Engineering Task Force) para abordar la necesidad de un protocolo ligero que pudiera ser utilizado por dispositivos con capacidades de procesamiento y almacenamiento limitadas, y que a menudo operasen en redes con restricciones de ancho de banda. Finalmente fue publicada en junio de 2014 [32].

El núcleo del protocolo está basado en la especificación internet RFC 7252 [32]. Diseñado para ser fácilmente traducible a HTTP para una integración sencilla con la web. Utiliza un modelo cliente-servidor y el protocolo de transporte UDP. Los errores y reintentos de mensajes se gestionan desde la capa de aplicación. Opcionalmente emplea la seguridad DTLS, que es la implementación del protocolo SSH en el protocolo UDP.

CoAP es un protocolo bastante complejo diseñado para la comunicación a través de Internet, por lo que para este proyecto sería añadir una complejidad innecesaria para una herramienta que no usará su principal funcionalidad. Además, es un protocolo que no está ampliamente estandarizado.

4.4.3. Open Productivity Collaboration Unified Architecture

Open Productivity Collaboration Unified Architecture (OPC UA) es un estándar internacional que permite la comunicación de datos entre dispositivos, sistemas y aplicaciones para la industria 4.0 de forma segura, eficiente e interoperable. Desarrollado por la OPC Foundation en 2008 para cubrir las nuevas necesidades y retos en seguridad y modelado de datos. [33]

Está disponible de forma gratuita diseñado específicamente para la automatización industrial. Permite el intercambio de información y datos en dispositivos dentro de máquinas, entre máquinas y desde maquinas a sistemas. Siendo así el intermediario entre la tecnología de la información y la tecnología operativa.

OPC UA reemplaza el protocolo COM y DCOM, específicos de Windows, por protocolos abiertos e independientes que, además, funcionan bajo otros sistemas operativos. Incorpora un modelo de información orientado a objetos. Permite una alta velocidad de transmisión de datos y encripta los mensajes e incluye sistemas de autenticación y permisos lo que lo hace fiable y seguro.

OPC UA es un protocolo bastante complejo diseñado específicamente para la automatización en la industria 4.0. Con respecto al proyecto que nos involucra, es posible que no necesite todas las características que ofrece OPC UA, lo que podría añadir una complejidad innecesaria. Además, este protocolo puede requerir cierta capacidad en los dispositivos.

4.4.4. MQTT

Message Queuing Telemetry Transport (MQTT) es un protocolo de mensajería ligero y estandarizado que se utiliza para la comunicación máquina a máquina (M2M) [34]. Fue desarrollado por Dr. Andy Stanford-Clark de IBM y Arlen Nipper de Eurotech en 1999 por la necesidad de un protocolo de comunicación eficiente para la industria del petróleo.

Figura 25. Logo MQTT

MQTT se diseñó para ser ligero y eficiente, lo que lo hace ideal para su uso en dispositivos de Internet de las Cosas (IoT) que menudo tienen limitaciones de potencia, consumo y ando de banda. Además es un protocolo altamente escalable al admitir la comunicación con una gran cantidad de dispositivos IoT. Las características de QoS (calidad de servicio) y mensajería cifrada que ofrece hace de MQTT un protocolo muy fiable y seguro.

MQTT es un protocolo de comunicación que se ejecuta sobre TCP/IP. Y se basa en el modelo de publicación/suscripción (Figura 26). Donde el bróker es el servidor que recibe, almacena y distribuye la información. El “Publisher” es un cliente que publica, es decir, envía información al bróker en “topics” o temas específicos. Los topics son cadenas de texto UTF-8. Están formados por uno o más “niveles” separados entre sí por barras inclinadas ‘/’. Esto permite

clasificar y discriminar unos mensajes de otros. Y finalmente el “Subscriber” es un cliente que se suscribe, ósea, lee uno o varios topics.

Figura 26. Funcionamiento de bróker-cliente.

Por las características presentadas de MQTT, en comparación con otros protocolos se ha decidico usar MosquitoMQTT como software que actuará como MQTT Broker local.

Mosquitto es un servidor MQTT de código abierto que actúa como un intermediario de mensajes del protocolo MQTT. Fue desarrollado por Eclipse Foundation, que fue creada por IBM en Noviembre de 2001 [35]. Más tarde, en Enero de 2004 Eclipse Foundation se convirtió en una organización independiente sin ánimo de lucro.

Figura 27. Mosquito logo.

Mosquitto está distribuido bajo la licencia EPL/EDL (Eclipse Public License/Eclipse Distribution License) [36]. Es un software muy liviano, adecuado para un amplio abanico de dispositivos, desde un PC hasta placas empotradas de bajo consumo. Con esto es altamente versátil pudiendo ser instalado en diferentes plataformas como en Windows, en Docker, en una instancia de Home Assistant y sistemas NAS, entre otros.

4.5. Elección de elementos para el acceso remoto

En este apartado se analizarán los diferentes elementos que se pueden utilizar para facilitar el acceso remoto al servicio de un dispositivo que se encuentra dentro de una red privada. Estos elementos son: los servicios de DNS dinámico, los gestores de proxy inverso y las redes privadas virtuales (VPN). Con el objetivo de elegir la mejor combinación de elementos para lograr un acceso remoto seguro, eficiente y sencillo.

4.5.1. DuckDNS

DuckDNS es un servicio gratuito de servidor DNS dinámico alojado en AWS [37]. Se encarga de dirigir direcciones IP a subdominios del tipo duckdns.org para disponer de nombres fácilmente recordables y tener un servicio DDNS con conexión externa.

El cliente de DuckDNS permite enviar actualizaciones de la dirección IP al servicio que aloja el dominio cada vez que esta cambie. Se puede instalar en una gran variedad de plataformas desde ordenadores con Windows o Mac, algunos modelos de routers, servidores NAS y en Docker.

4.5.2. No-IP

No-IP es un servicio de servidor DNS administrado por la compañía estadounidense Vitalwrks Internet Solutions, LLC fundada por Dan Durren en 1999 [38]. Ofrece servicios DNS, DDNS, correo electrónico, monitoreo de red y certificados SSL. Aunque la mayoría de sus servicios son de pago, ofrece limitados servicios gratuitos de dominios del tipo no-ip.org o ddns.net, entre otros.

El cliente de Noip permite enviar actualizaciones de la dirección IP al servicio que aloja el dominio cada vez que esta cambie. Se puede instalar en varias plataformas como Windows, Mac o en Docker.

4.5.3. NGINX

NGINX es un software de código abierto que ofrece servidores web de alto rendimiento, proxy inverso, servidor proxy de correo y proxy genérico TCP/UDP [39].

Figura 28. Logo de NGINX

Fue creado por Igor Sysoey en 2002 como intento de solucionar el problema C10k, reto de gestionar diez mil conexiones al mismo tiempo [40]. Lanzado el Octubre de 2004. En origen funcionaba con HTTP, ahora sirve como proxy inverso, balanceador de carga HTTP y proxy de correo electrónico para IMAP, POP3 y SMTP.

Está diseñado con una arquitectura modular, donde cada módulo es responsable de una funcionalidad específica dentro del servidor. El módulo principal es responsable de manejar la conexión y, además, hay una serie de módulos para diferentes tipos de procesamiento. Esta arquitectura modular hace que NGINX sea muy flexible y personalizable. Además, funciona mediante una arquitectura asíncrona, controlada por eventos. Esto significa que en que en lugar de crear un nuevo proceso o hilo para cada petición web, NGINX utiliza un enfoque basado en eventos donde las solicitudes se manejan en un solo hilo. Debido a estas características lo hace muy confiable, veloz y escalable.

Aunque su uso destinatario es hacia grandes servidores webs, con productos altamente costosos para un pequeño proyecto. También ofrece la posibilidad de instalar gratuitamente un modulo de NGINX en un contenedor Docker o como complemento de Home Assistant que nos permitirá configurar un proxy inverso.

En cuanto a seguridad. NGINX presenta una autenticación robusta. Certificados SSL. Optimización SSL/TLS.Desactivación de protocolos SSL/TLS débiles: SSL 3, TLS 1.0 y TLS 1.1 son vulnerables, y sólo permitiremos un protocolo TLS 1.2 fuerte.

4.5.4. Cloudflare

Cloudflare es una plataforma de conectividad en la nube que ayuda a conectar, proteger y optimizar a las personas, aplicaciones y datos en entornos locales, nube pública, SaaS e Internet. Ofrecen CDN, DNS, protección frente a DDoS y seguridad.

Figura 29. Logo Cloudflare.

Fue fundado por Matthew Prince, Lee Holloway y Michelle Zatlyn en 2009. El proyecto inició con Project Honey Pot en 2004 como un rastreador de spammers [41]. Así, Cloudflare nació de la idea de llevar a Project Honey Pot al siguiente nivel: no solo rastrearía las amenazas de Internet, sino que también las detendría.

Cloudflare funciona como un intermediario que provee réplicas exactas de tu sitio web sin afectar el desempeño [Figura 30]. Utiliza unos sistemas llamados proxies reversos para crear copiar espejo y cachés de sitios web. Pudiendo actuar como un filtro entre el sitio web y las solicitudes de los usuarios [42].

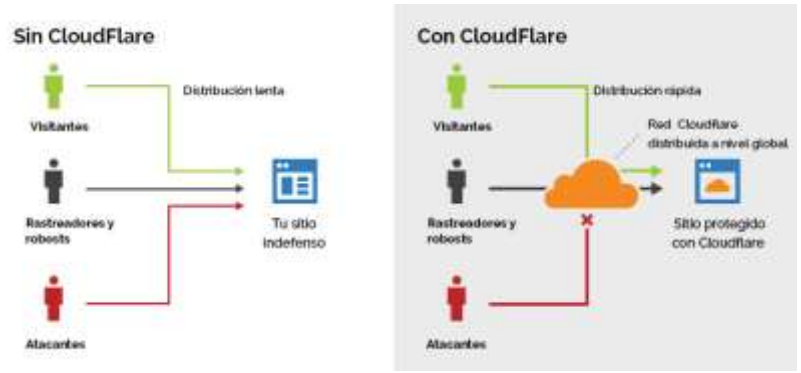


Figura 30. Como funciona cloudflare.

Aunque los principales servicios de Cloudflare son de pago. También ofrece unos servicios básicos gratuitos que incluyen mitigación de ataques DDoS y certificado SSL. Proporcionadon imágenes que pueden ser instaladas en un contenedor Docker o como complemento de Home Assistant.

4.5.5. Tailscacle

Tailscale Inc. es una compañía de software que permite interconectar equipos a través de internet mediante una red privada virtual (VPN) de forma rápida, fácil y segura. [43]

Figura 31. Logo de Tailscale.

Fue fundada en 2019 por Avery Pennarun, David Carney y Brad Fitzpatrick [44]. Su objetivo fue proporcionar una forma sencilla y segura de interconectar diferentes equipos a través de Internet. Para ello Tailscale hace uso de las redes definidas por software (SDN) para intercomunicar los diferentes nodos a la red privada virtual (VPN) e interconectarlos entre sí.

En su diseño se basaron en el protocolo de WireGuard para la red privada virtual, por su rendimiento y seguridad. Mientras, Tailscale se encarga de manejar la distribución de claves y de todas las configuraciones, facilitando el uso a usuarios no técnicos.

Este servicio proporciona la posibilidad de interconectar diferentes equipos a través de Internet en una red privada virtual, instalando un pequeño programa en cada equipo e iniciando sesión con las credenciales de registro en el servicio. No se necesita abrir ningún tipo de puerto, ni montar un servidor VPN en cada equipo, ni realizar complicadas configuraciones. El objetivo de Tailscale es permitir acceder a todos los equipos de forma remota, independientemente de si están detrás de un CG-NAT o un NAT, de forma rápida y muy fácil.

El programa cliente de Tailscale se puede instalar tanto en Windows, Mac, Android, iOS como en Docker y como complemento de Home Assistant. Ofreciendo una encriptación de

tráfico peer-to-peer (punto a punto) entre los dispositivos. Sin embargo, al no requerir montar un servidor VPN propio, el tráfico entre diferentes dispositivos será manejado por los servidores de Tailscale. Introduciendo la dependencia en la seguridad de Tailscale. Es importante tener en cuenta que la clave privada de un dispositivo nunca sale del dispositivo y, por lo tanto, Tailscale no puede descifrar el tráfico.

4.5.6. Zerotier

Zerotier es una compañía de software que permite interconectar equipos a través de una red privada virtual (VPN) de forma rápida y segura.

Figura 32. Logo de Zerotier.

Fue fundada por Adam Leymenko en 2015 [45]. La primera versión de Zerotier Oen fue publicado en 2014. Su visión era simple, conectar los dispositivos directamente entre sí y permitir una nueva era de comunicación descentralizada [46].

Para ello se diseñó su propio protocolo VPN llamado Zerotier One. Con el objetivo de que todo el tráfico vaya cifrado de punto a punto (peer-to-peer). Su plataforma consta de dos capas de virtualización: la “Capa Virtual 1” (VL1) es la columna vertebral de la red peer-to-peer que cifra las comunicaciones, garantiza la autenticación de los puntos finales y verifica las credenciales utilizando claves asimétricas. La “Capa Virtual 2” (VL2) se construye sobre VL1 y utiliza principios de redes definidas por software para funcionar como una red de área local extensible virtual (VX-LAN) [47].

ZeroTier ofrece una forma fácil de comunicar equipos en Internet, sin necesidad de crear nuestros propios túneles VPN. Esta herramienta sirve tanto en el ámbito doméstico como también en el profesional, ya que tenemos la posibilidad de interconectar hasta 50 dispositivos en una misma red de manera gratis. Gracias a su ZeroTier One, la empresa pone a nuestra disposición un programa cliente que permite a los PC, servidores, smartphones, servidores NAS, Docker, Home Assistant y otros dispositivos comunicarse entre ellos a través de una red privada virtual (VPN), y todo ello de manera completamente gratuita y segura.

4.5.7. OpenVPN

OpenVPN es un software de código abierto que ofrece diferentes soluciones para establecer, mantener y personalizar redes privadas virtuales de forma segura.

Figura 33. Logo de OpenVPN.

Fue creado por James Yonan y publicado por primera vez el 13 de Mayo de 2001. Su creación fue motivada por su necesidad de conectarse de forma segura a su oficina mientras viajaba por Asia Central [48].

Fue diseñado para ser fácilmente implementado y para ofrecer una combinación de seguridad, facilidad de uso y riqueza de características. Desde su creación, OpenVPN ha simplificado la configuración de VPN's frente a otras más antiguas y difíciles de configurar como IPsec, haciéndola más accesible para gente inexperta en este tipo de tecnología.

OpenVPN permite formar túneles de comunicación cifrada por medio de protocolos SSL (Secure Socket Layer), que son los estándares para HTTPS. OpenVPN ofrece conectividad punto-a-punto con validación jerárquica de usuarios y host conectados remotamente. Resulta una muy buena opción en tecnologías Wi-Fi (redes inalámbricas IEEE 802.11) y soporta una amplia configuración, entre ellas balanceo de cargas.

4.5.8. Wireguard

WireGuard es un protocolo y software de código abierto que pretende proporcionar una alternativa más moderna a las soluciones VPN establecidas como OpenVPN o IPsen. Siendo más rápido, seguro y ligero. [49]

Figura 34. Logo de Wireguard.

Fue creado por Jason A. Donenfeld por la necesidad de conectarse de forma segura a sus oficinas mientras viajaba. Se publicó por primera vez en Marzo de 2015 [49].

Fue diseñado para ser fácilmente implementado en muy pocas líneas de código, haciéndolo muy versátil. En su totalidad el código del kernel no ocupa más de 4000 líneas. En comparación con OpenVPN o IPsec suele tener entre 100 000 y 600 000 líneas. Un código base más breve es también más seguro, ya que en él los bugs se detectan más fácilmente y el campo de ataque se reduce. Está escrito en los lenguajes C y GO. Creado como una VPN de propósito general.

Además de la mayor seguridad que ofrece, la sencillez del software también favorece un mayor rendimiento. En análisis comparativos, WireGuard demuestra alcanzar una mayor velocidad de transmisión y una latencia más baja que sus competidores.

Por su estructura, WireGuard es un protocolo VPN descentralizado *peer-to-peer*. En lugar de requerir un servidor, WireGuard puede abrir directamente un túnel entre dos dispositivos. Lo que podríamos considerar un servidor WireGuard es simplemente un dispositivo en el que se han realizado configuraciones de conexión para varios dispositivos.

En el establecimiento de la conexión los usuarios generan claves públicas con WireGuard y las intercambian entre ellos. Gracias a ellas pueden identificarse los unos a los otros y encriptar los paquetes de datos para su destinatario correspondiente.

Cuando el usuario no está enviando datos a través del túnel, WireGuard se mantiene en stand-by. De esta forma se ahorra energía y se alarga la autonomía de la batería. Siendo está eficiencia energética muy importante en dispositivos móviles o dispositivos IoT.

El software de Wireguard está sujeto a la licencia GPLv2 [50] como software libre y es multiplataforma. Compatible con una gran variedad de plataformas desde Windows, macOS, iOS y Android hasta en contenedores Docker, Home Assistant, servidores NAS.

4.6. Elección de herramientas de software.

Partiendo de que la plataforma software principal a emplear será Home Assistant. A continuación evaluaremos varias herramientas de software, AppDaemon, PyScript, YAML y Node-RED. El objetivo es proporcionar una visión detallada de cada herramienta para seleccionar la más adecuada que nos permitan abordar algunos desafíos del presente proyecto.

4.6.1. AppDaemon

AppDaemon es un entorno de ejecución de scripts de Python diseñado para escribir aplicaciones de automatización para Home Assistant. Es un sistema multihilo y aislado que permite la ejecución de scripts de Python de manera segura [51].

AppDaemon permite adjuntar devoluciones de llamada a cada evento de cambio de estado, lo que facilita la implementación de desencadenadores que involucran múltiples variables. AppDaemon requiere funciones auxiliares para usar las variables de estado de Home Assistant.

Cada paso en AppDaemon normalmente necesitará una devolución de llamada para esperar un tiempo u otro evento, como un cambio de estado. AppDaemon admite subprocesos y tareas asincrónicas, aunque en su documentación señala que el usuario necesita control manual y experiencia para usar Async correctamente.

Aunque AppDaemon utiliza código nativo de Python y se ejecuta más rápido, se ejecuta dentro de Home Assistant, lo que puede resultar en algunas sobrecargas de IPC y cambio de contexto [52].

AppDaemon tiene muchas otras características interesantes, incluido un panel de administración que muestra detalles acceso directo a MQTT y plugins.

4.6.2. PyScript Home Assistant

PyScript como integración de Home Assistant permite desarrollar funciones o scripts en Python que pueden implementar una amplia gama de automatizaciones [53].

Las variables de estado están enlazadas a variables de Python y los servicios se pueden llamar como funciones de Python, por lo que es fácil y conciso la implementación. Las funciones que se escriba pueden se llamadas como un servicio en Home Assistant.

Con PyScript, los desencadenadores de estado pueden ser expresiones completas de Python que involucran múltiples variables de estado, lo que proporciona una gran flexibilidad. Además, una automatización con varios pasos puede implementarse como una sola función en PyScript, eliminando la necesidad de dividir las cosas usando *callbacks*.

PyScript está totalmente basado en async y se ejecuta dentro de Home Assistant, lo que significa que los usuarios no necesitan tener un conocimiento profundo de la programación asíncrona. PyScript interpreta el código Python, lo que puede ralentizar la ejecución de fragmentos largos de código de usuario. Por último, PyScript es compatible con las interfaces de Jupyter, lo que facilita enormemente el desarrollo y la depuración de código, funciones y disparadores [52].

4.6.3. YAML

YAML es un lenguaje de serialización de datos diseñado para ser leído y escrito por humanos [54]. Se encarga de almacenar archivos de configuración y se puede usar junto con todos los lenguajes de programación. También permite serializar objetos, es decir, escribir la estructura de un objeto en modo cadena de texto para posteriormente poderlo recuperar. Este lenguaje resulta altamente legible para las personas, siendo más legible que JSON y que XML. Es sensible a mayúsculas y minúsculas y comúnmente se identifica mediante las extensiones .yml o .yaml.

El termino *YAML* que significa "YAML Ain't Markup Language" (en castellano, 'YAML no es un lenguaje de marcado'). En sus orígenes de desarrollo YAML significaba "Yet Another Markup Language" ('otro lenguaje de marcado más').

Existen unas reglas generales que deben cumplirse en un documento YAML:

- Los datos de un documento YAML deben ser legibles, imprimibles y utilizando caracteres Unicode, UTF-8 o UTF-16.
- Los comentarios se realizan utilizando el carácter '#' dentro de la línea que contiene el comentario.
- Los caracteres ',' y ';' deben ir seguidos de un espacio en blanco. De esta forma, se podrán representar valores que queramos que tengan esos caracteres.
- Los espacios en blanco están permitidos, pero no los tabuladores.
- Las listas comienzan por el carácter '-' con un valor por cada línea, aunque también se pueden utilizar corchetes [] poniendo los valores dentro de ellos separados por comas ',' junto con un espacio en blanco.
- Un vector estará formado por el par clave/valor, estando separados ambos por ':' poniendo uno por línea, aunque también podemos utilizar {} poniendo cada uno de ellos dentro separados por comas ',' junto con un espacio en blanco.
- Se pueden utilizar caracteres de escape '\' para representar caracteres especiales.
- Para incluir múltiples documentos, los separaremos por tres guiones seguidos '—' e indicando el fin de un documento con tres puntos seguidos …
- La estructura del documento se denota indentando con espacios en blanco; sin embargo, no se permite el uso de caracteres de tabulación para indentar.

4.6.4. Node Red

Node-RED es una herramienta de programación visual de código abierto que proporciona una solución lista para usar. Desarrollada en 2013 por Nick o' Leary y Dave Conway del grupo de Servicios de Tecnologías Emergentes de IBM [55]. Se basa en Node.js, que es un motor de ejecución asíncrono de JavaScript para servidores.

Figura 35. Logo Node-RED.

Node-RED trabaja mostrando de manera visual las relaciones y funciones de programación a través de un panel de flujo. Este sistema de representación facilita visualizar

gráficamente el flujo de la información. Permitiendo que se pueda programar sin escribir y que se pueda usar tecnologías complejas de manera sencilla e intuitiva.

Esta herramienta de programación ofrece total flexibilidad para conectar dispositivos, bases de datos, aplicaciones, protocolos de red, etc. Por ejemplo, Node-RED proporciona una forma sencilla de interactuar con un brókerMQTT mediante nodos de entrada y salida MQTT. Dispone de una extensa variedad de librerías, además, de una comunidad muy activa.

Node-RED puede correr en una diversidad de plataformas desde sistemas operativos como Windows o MAC, en servidores NAS, en contenedores Docker y como complemento de Home Assistant.

4.7. Sumario de las elecciones realizadas

En este proyecto se ha optado por utilizar las siguientes tecnologías para el desarrollo de una red de dispositivos inteligentes que se comunican de forma inalámbrica, segura y eficiente:

- LoRaWAN: una red de baja potencia y área amplia que permite la transmisión de datos a larga distancia y con bajo consumo de energía entre los dispositivos.

- Como plataforma hardware: Raspberry Pi 4, una placa de computación de bajo costo y consumo.

- Como plataforma software Home Assistant: una plataforma de software libre que permite el control y la automatización de los dispositivos mediante una interfaz web.

- MQTT: un protocolo de mensajería ligero y basado en el modelo publicador-suscriptor que facilita la comunicación entre los dispositivos y las aplicaciones.

- WireGuard: un protocolo de red privada virtual que ofrece una conexión segura y cifrada entre los dispositivos y el sistema, permitiendo el acceso remoto al mismo.

- YAML: herramienta que facilita la integración de los dispositivos LoRaWAN en Home Assistant, mediante la definición de entidades.

- Node-RED: herramienta que facilita la comunicación entre los dispositivos LoRaWAN y Home Assistant, usando flujos de datos.

5. DESCRIPCIÓN DETALLADA DE LA SOLUCIÓN ADOPTADA

Este documento describe el despliegue de dispositivos IoT y servicios de software, que se desarrollará en varias etapas (Figura 36). Los 'End Nodes' están compuestos por los dispositivos IoT que utilizan la tecnología LoRaWAN. El 'Gateway' actúa como 'Network Server'. El software Node-RED aloja programas que traducen estos paquetes de datos para que sean manejables por la plataforma domótica Home Assistant.

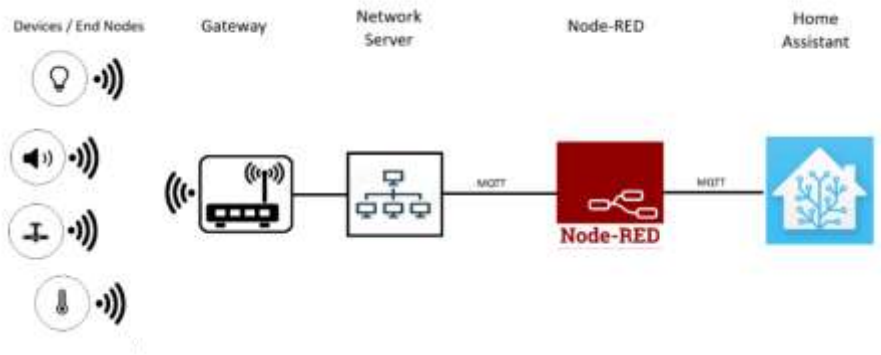


Figura 36. Diagrama de Bloques del sistema.

5.1. Red LoRaWAN

En este apartado se explica cómo configurar la red LoRaWAN, que permite la comunicación inalámbrica entre los dispositivos inteligentes. Para ello, se detallan los pasos para configurar el Gateway, que es el dispositivo que recibe y envía información a los nodos de la red, y la puesta a punto de los diferentes dispositivos LoRaWAN, que son los que envían y reciben información al gateway.

5.1.1. Configuración pararela RAK7268

El Gateway RAK7268 (Figura 37) se encargará de recibir y enviar paquetes de datos de los sensores o dispositivos LoRaWAN. Estará conectado mediante cable Ethernet al Router, aunque ofrece la posibilidad de conectividad vía WiFi. Mediante el protocolo MQTT redirigirá los mensajes de los dispositivos a la Rasberry Pi, y viceversa, de forma local. Para ello tendremos que configurarlo para que adopte la función de Network Server accediendo a la interfaz de configuración y en el apartado *LoRa Network* y subapartado *Network Settings* configuraremos el Modo como Network Server (véase Figura 38).

Figura 37. LoRaWAN Gateway RAK7268.

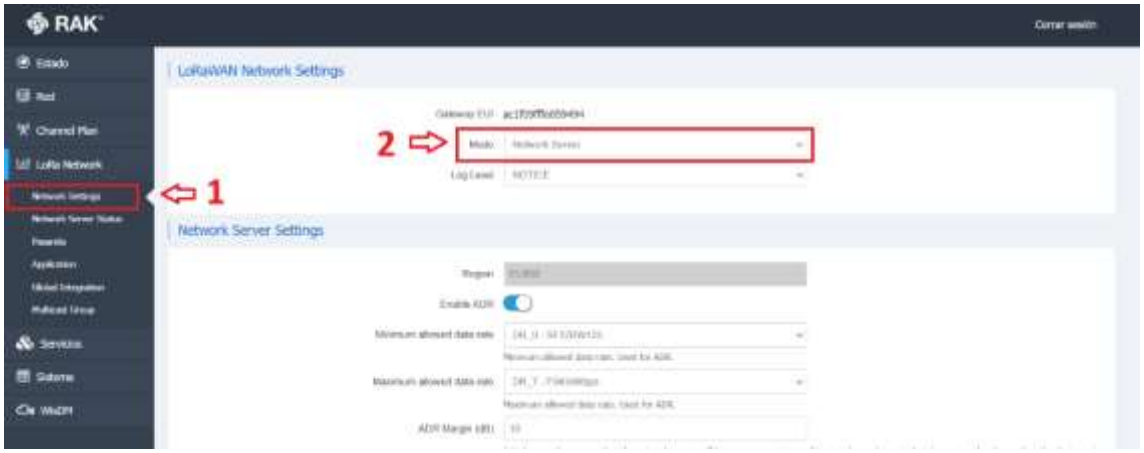


Figura 38. Configuración RAK7268 como Network Server.

A continuación, configuraremos el protocolo MQTT. Para ello en la misma pestaña de *LoRa Network* iremos al subapartado Global Integration. Seleccionaremos el *Integration Mode* como *Generic MQTT*. En MQTT Broker Addres pondremos la IP y el puerto asignado al bróker MQTT que configuraremos más adelante. Usaremos versión estándar, para fecha de este proyecto es V3.1.1.

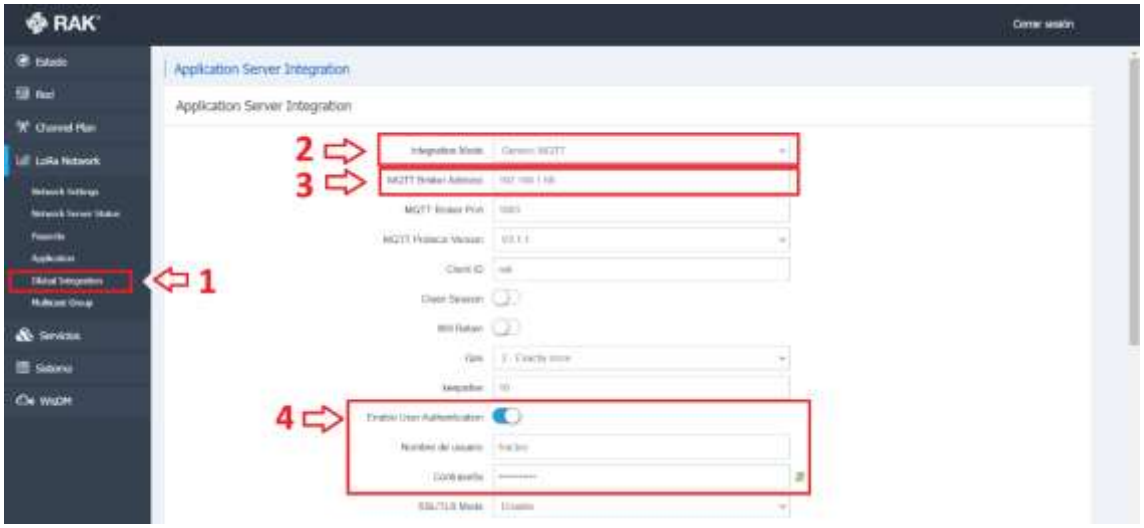


Figura 39. Gateway configuración MQTT

El siguiente paso sería añadir los diferentes dispositivos al Network Server. En el mismo apartado de LoRa Network, en el subapartado de *Aplication* añadiremos un nuevo *Aplication Key* de tipo 2: "Separate Aplication Key", que nos permitiría desarrollar nuestro proyecto sin interferir o eliminar otras aplicaciones que estén en marcha, si el GateWay no se usara para otros fines serviría con el tipo 1. Clicando en Editar nos saldrá una pestaña donde podremos ir agregando los *device EUI* de los diferentes dispositivos (Figura 40). A continuación, seleccionando cada dispositivo nos mostrará una pestaña "Resumen" de la actividad del dispositivo. Accediendo a la pestaña "Configuration" terminaremos de configurar cada dispositivo introduciendo la *Aplication Key* y seleccionando la *Class* de funcionamiento de cada dispositivo (Figura 41). También podremos agregar un nombre para identificar más fácilmente el dispositivo.

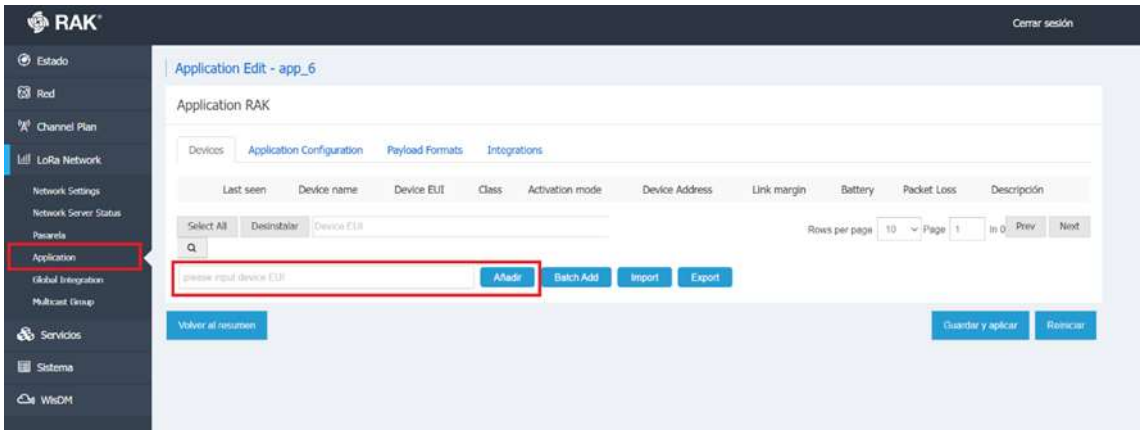


Figura 40. Creación de la aplicación donde agregar los dispositivos.

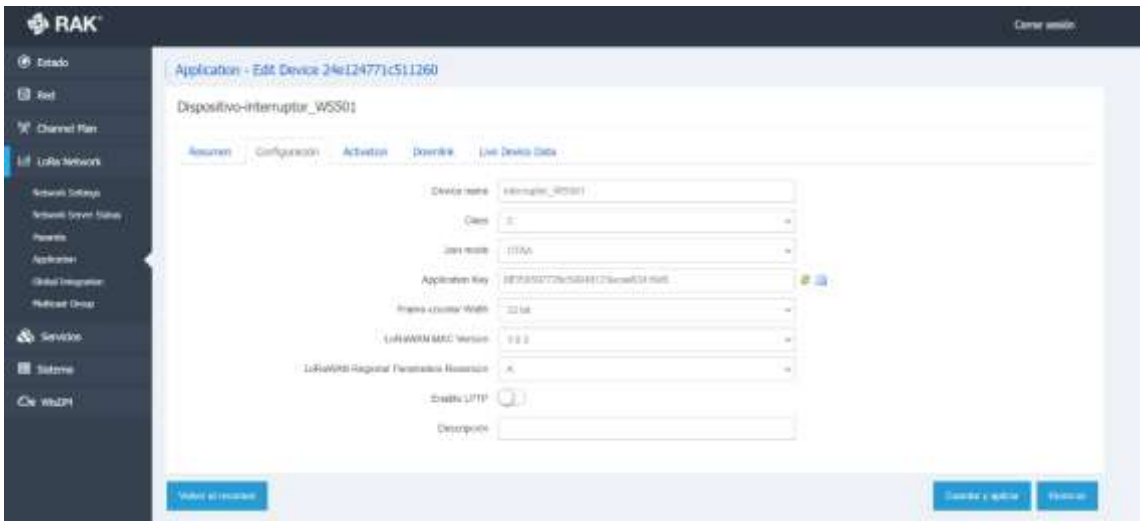


Figura 41. Interfaz de configuración de dispositivo.

A continuación se detallará la puesta a punto de cada dispositivo:

5.1.2. Puesta a punto sensor de temperatura Dragino LHT52

Como sensor de temperatura y humedad se usará el LHT52 de la marca Dragino (Figura 42). Admite además una sonda externa, que para la prueba de concepto no utilizaremos. El sensor de temperatura incorporado presta un rango de funcionamiento entre -20℃ y 50℃ con una resolución de 0,01℃. Usa Protocolo LoRaWAN v1.0.3 Clase A. El sensor de humedad incorporado presta un rango de 0 ~ 99,0 % RH (sin rocío), con una resolución de 0,1 % RH. Necesita como fuente de alimentación 2 pilas AAA LR03.

Figura 42. Sensor de temperatura y humedad de Dragino. A la derecha se muestra la parte frontal. En el centro se muestra la parte de abajo, donde estás la información del dispositivo. Y a la izquierda el interior del dispositivo.

En el partado de la *Aplication* (Figura 40) creada introduciremos el device EUI del sensor Dragino que se encuentra en la parte trasera de la carcasa, y le damos a 'añadir'. En el apartado de 'Configuración' terminamos de configurar el dispositivo(Figura 43). Usaremos Join mode OTAA, y como se ha mencionado antes, LoRaWAN versión 1.0.3 y Class A.

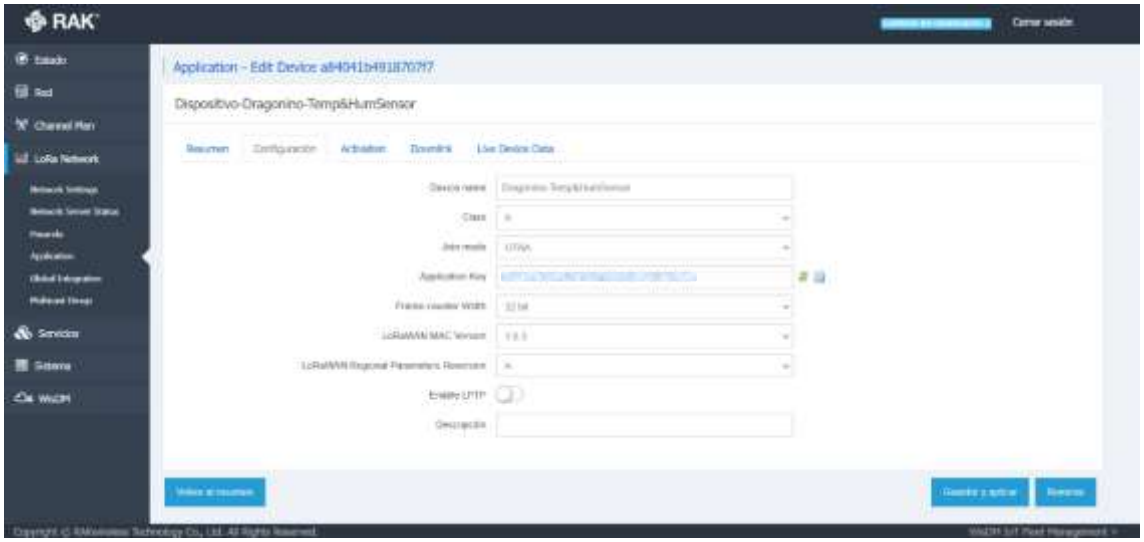


Figura 43. Configuración Dragino.

Para comprobar el correcto funcionamiento, en la pestaña Resumen, en la sección de Traffic History se debe ver el mensaje que envía el Dragino cada 20 min (Figura 44).

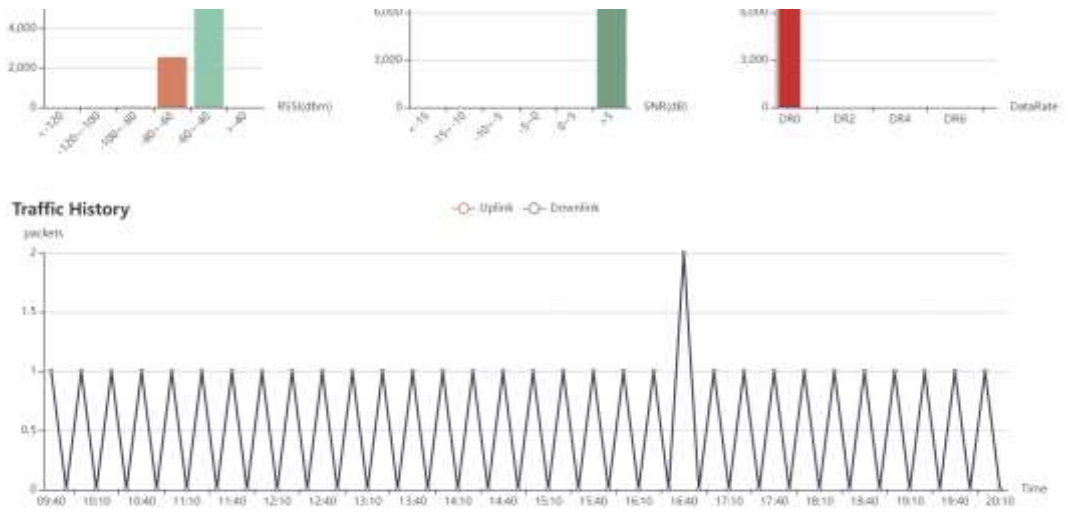


Figura 44. Comprobación de funcionamiento de Dragino.

Sin embargo, en la realización del proyecto esto no funcionó como se esperaba. Parece ser que el dispositivo necesitaba que el Rx 1 Delay estuviera a 5 segundos (Figura 45). El inconveniente de ajustar este parámetro es que puede interferir en el funcionamiento de otros dispositivos.

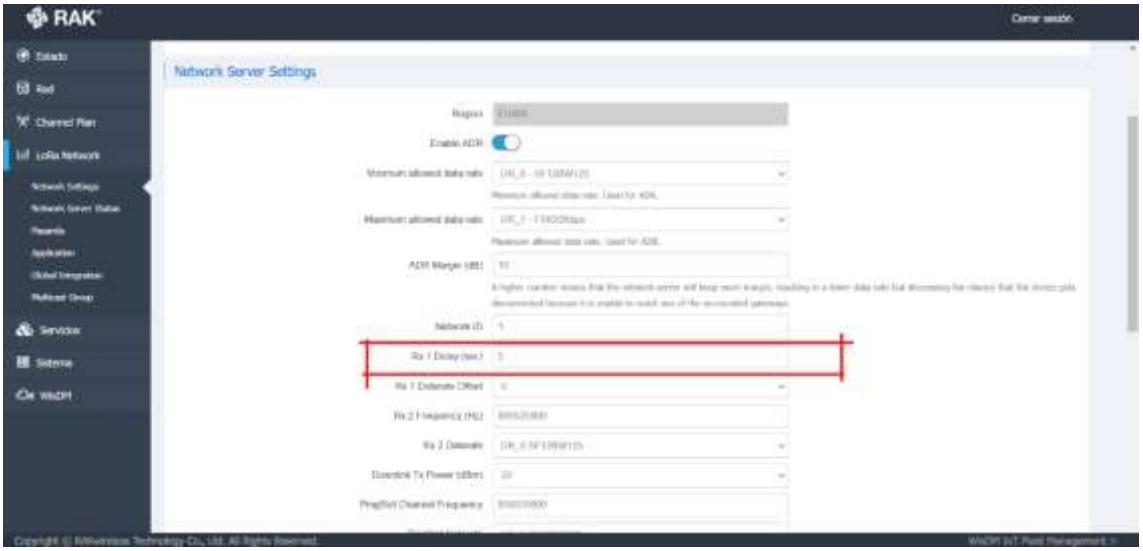


Figura 45. Configurar el Rx 1 Delay para el Dragino.

5.1.3. Puesta a punto sensor de movimiento Milesight WS202

Como sensor de movimiento se usará el WS202 de Milesight (Figura 46). Utiliza un sensor de detección PIR y un medidor de brillo. Utiliza una batería Li-SOCL 2 ER14335 de 1650 mAh.

Figura 46. Sensor de movimiento WS202 Mileght.

Se configura mediante tecnología NFC desde un móvil o PC con la aplicación de Milesight ToolBox (Figura 47Figura 47). Con NFC habilitado se usa NFC Read para establecer comunicación con el dispositivo. Y una vez establecida, en el apartado 'Setting' y sección 'LoRaWAN Setting' se muestra los datos del dispositivo para agregar en la configuración de nuevo dispositivo del Network Server de la pasarela RAK7268 (Figura 47).

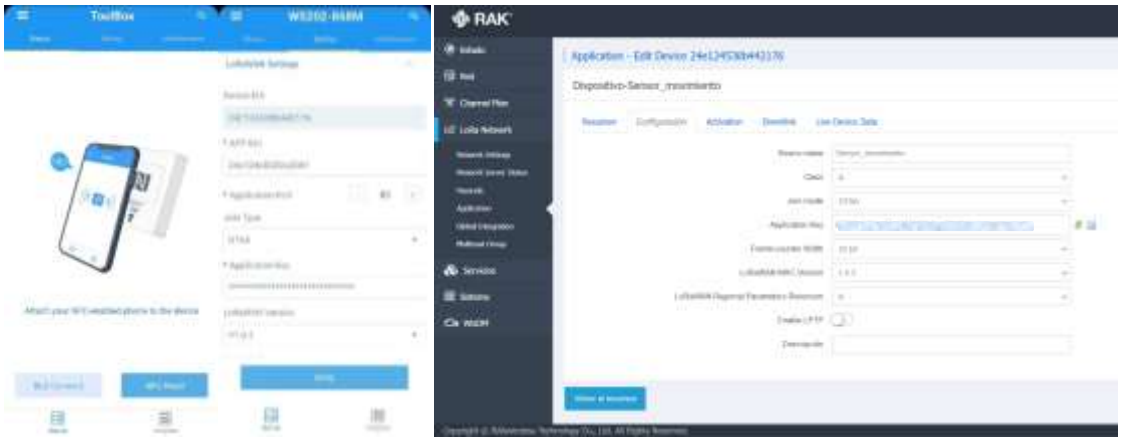


Figura 47. A la izquierda la Herramienta ToolBox para teléfono móvil. A la derecha configuración del sensor de movimiento en RAK7268.

Finalmente comprobamos el correcto funcionamiento mediante la pestaña resumen del Network Server.

5.1.4. Puesta a punto interruptor inteligente Milesight WS501

Como dispositivo de control de la iluminación usaremos un interruptor inteligente WS501 de Milesight. Interruptor de un botón con cable neutro. Además proporciona medición del consumo eléctrico, del voltaje y el amperaje.

El primer paso es conectarlo a la red eléctrica del edificio siguiendo el esquema que proporciona la documentación (Figura 48). Esto es, de izquierda a derecha, en el primer borne se conecta la línea neutra, en el segundo la línea de fase y en el tercero la línea que alimenta la bombilla.

El siguiente paso es agregarlo a la red LoRaWAN. Que se configura mediante la tecnología NFC al igual que el sensor de movimiento visto previamente. Por lo que se seguirá los mismos pasos.

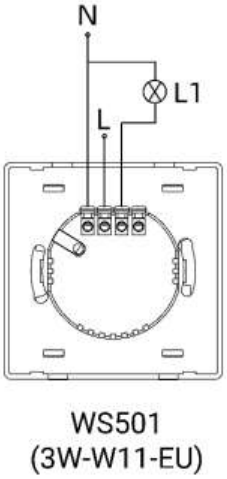


Figura 48. Esquema de conexiones del interruptor [56].

5.1.5. Puesta a punto interruptor inteligente Milesight WS502

WS502 es un interruptor inteligente de Milesight de dos pulsadores sin neutro. Al no tener neutro tiene el inconveniente de que con cargas inferiores a 38W genera parpadeos por la descargar de la parte electrónica [56].

Se conecta a la red eléctrica del edificio siguiendo el esquema que proporciona la documentación (Figura 49). Esto es, de izquierda a derecha, en el primer borne se conecta la línea de fase, en el segundo la línea accionada por el primer interruptor, y en el tercero la línea accionada por el según interruptor.

Por último, al igual que el interruptor y el sensor de movimiento se configura mediante la herramienta ToolBox. Por lo que se seguirá los mismos pasos.

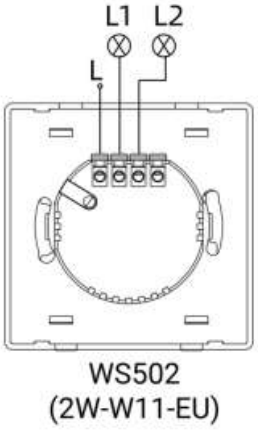


Figura 49. Esquema de conexiones del interruptor doble [56].

5.1.6. Puesta a punto enchufe portable inteligente Milesight WS523

WS523 de Milesight como medidor de consumo de enchufe. Proporciona medición de voltaje, amperaje, factor de potencia, potencia y energía consumida.

Figura 50. Medidor de consimo WS523 Milesight.

Como los dispositivos anteriores de Milesight, se configura mediante la herramienta ToolBox, siguiendo los mismos pasos.

5.1.7. Puesta a punto sensor de temperatura Milesight EM320-TH

Por último un sensor de temperatura y humedad EM320-TH de Milesight. Con rango de -30℃ a 60℃ y una resolución de 0,1℃ y 0,5%RH.

Como los dispositivos anteriores de Milesight, este también se configura mediante la herramienta ToolBox, siguiendo los mismos pasos.

5.2. Controlador central sobre Raspberry Pi

5.2.1. Configuración Hardware

Los componentes del proyecto son:

- Raspberry Pi modelo 4 con 4GB RAM.
- Cable USB C macho a USB macho.
- Fuente de alimentación de 5V con conexión USB hembra.
- Tarjeta SD High Endurance de 16 Gb.
- Cable Ethernet de 1Gbps.

La Raspberry Pi se alimentará mediante una tensión de 5V, suministrada a través del cable USB 3.0. Para la conectividad en red, se utilizará el cable Ethernet para conectar la Raspberry Pi con el router. Como unidad de memoria, se usará una tarjeta SD por defecto en la Raspberry Pi, la cual se insertará en la ranura correspondiente (véase Figura 51).

Figura 51. Montaje de la Raspberry Pi.

5.2.2. Despliegue servicios software

El sistema operativo que se utilizará será Raspberry Pi OS (64 bit) Lite, el cual está basado en Debian Bullseye. Este sistema operativo proporciona actualizaciones de seguridad y no incluye un entorno de escritorio, lo que lo hace ideal para aplicaciones que requieren un rendimiento óptimo y un uso mínimo de recursos.

Mediante la herramienta Raspberry Pi Image (Figura 52), se selecciona la imagen SO y el puerto donde esté conectado la tarjeta SD. En configuración avanzada se activa SSH que permitirá acceder a la Raspberry Pi mediante el protocolo SSH desde otro ordenador. Y configuraremos el usuario y contraseña de autenticación. Por último escribir la imagen en la tarjeta SD.

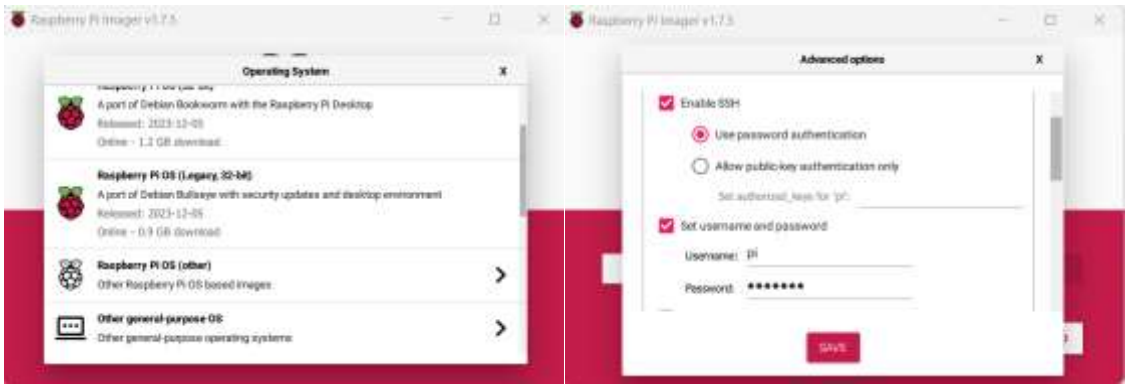


Figura 52. Herramienta Raspberry Pi Imager. A la izquierda insterfaz de selección se Sistema Operativo. A la derecha interfaz de opciones avanzadas.

Una vez que se ha insertado la tarjeta SD en la ranura correspondiente de la Raspberry Pi y se ha encendido el dispositivo, se puede acceder a él a través de SSH. Para ello, se introduce el comando correspondiente en la terminal o en PowerShell del dispositivo que se conectará desde SSH a la Raspberry Pi:

```
ssh <username>@<dirección_IP>
```

sustituyendo username con el introducido en las opciones avanzadas al instalar el iSO y la dirección_IP asignada a la Raspberry Pi. Esta se puede saber consultando los dispositivos conectador en el router

En el contexto del presente proyecto, se ha optado por utilizar Docker Community Edition (Docker-CE), una plataforma de contenedores que simplifica el despliegue de aplicaciones dentro de contenedores, parecidos a maquinas virtuales livianas. Esto permite empaquetar una aplicación, con todas sus dependencias, en una unidad estandarizada de software que permite su ejecución en cualquier infraestructura.

Su instalación se realiza mediante los siguientes comandos en terminal desde ssh:

```
curl -fsSL https://get.docker.com -o get-docker.sh
```

Este comando descarga el script de instalación de Docker del sitio web https://get.docker.com y lo guarda en un archivo llamado `get-docker.sh` en actual directorio.

```
sh get-docker.sh
```

Este comando ejecuta el script de instalación.

```
sudo usermod -aG docker $USER
```

Este comando agrega el usuario al grupo Docker, lo que permite ejecutar comandos Docker sin necesidad de usar constantemente sudo.

El siguiente comando instala Docker Compose que es una herramienta para definir y ejecutar aplicaciones Docker de varios contenedores.

```
apt install docker-compose
```

Por último, se opta por instalar Portainer-CE, una interfaz visual de código abierto que facilita la creación, administración y mantenimiento de entornos de contenedores. Se instala como un contenedor Docker mediante los siguientes comandos docker.

```
docker volume create portainer_data

docker run -d -p 8000:8000 -p 9443:9443 --name portainer \
    --restart=always \
    -v /var/run/docker.sock:/var/run/docker.sock \
    -v portainer_data:/data \
    portainer/portainer-ce:latest
```

Primero se crea el volumen para el contenedor. Luego se ejecuta el contenedor docker con docker run en modo segundo planto (“detached”) con `-d` y se mapea los puertos de escucha del host con los del contenedor con `-p`. Se configura el contenedor para que se reinicie automáticamente si se detiene con `--restart=always`. La opción `-v` monta los volúmenes. Monta el socket de Docker del host en el contenedor. Esto permite que Portainer interactúe con el motor Docker del host. Y monta el volumen creado “portainer_data” en el directorio /data del contenedor. Y `portainer/portainer-ce:latest` es la imagen que se va a ejecutar.

Previamente a lanzar los diferentes servicios software es necesario generar una estructura de directorios donde almacenarán los datos y archivos permanentes correspondientes a dichos servicios. Para ello, se ejecuta el siguiente comando:

```
mkdir -p Docker/{home-assistant,mosquitto /{config, data, log}, node-red, wireguard/{config,db}}
```

En el directorio denominado 'Docker', se procede a la creación del archivo `docker-compose.yml` (véase Anexo I). Para ello, se ejecutan los siguientes comandos:

```
cd Docker
nano docker-compose.yml
```

Estos comandos abren la interfaz de edición de Nano, donde se introduce el contenido especificado en el Anexo I. Para guardar los cambios realizados, se presiona `Ctrl+X`, seguido de `Y` para confirmar la intención de guardar los cambios, y finalmente `Enter` para confirmar el nombre del archivo.

Una vez completada la configuración del archivo `docker-compose.yml`, se inician los servicios especificados en el mismo mediante la ejecución del siguiente comando:

```
docker-compose up -d
```

Este comando lanza los servicios en segundo plano (debido a la opción `-d`), permitiendo que continúen ejecutándose incluso después de cerrar la terminal."

Los servicios que se usan en este proyecto son:

- Home Assistant
- Mosquitto
- Node-RED
- Hass-configuration
- WireGuard

A continuación se detallará las peculiaridades de cada servicio software.

- Home Assistant

En el servicio software de Home Assistant se lanzará la imagen `ghcr.io/home-assistant/home-assistant:stable` en el contenedor. El puerto de escucha por defecto de Home Assistant es 8123. Los archivos permanentes se encuentran en el directorio `/config` del

contendedor. Por lo que se mapea el volumen al directorio creado anteriormente en el host `Docker/homeassistant`

- Mosquitto

En el servicio software del bróker MQTT Mosquitto MQTT se lanzará la imagen `eclipse-mosquitto:latest` en el contenedor. El puerto de escucha por defecto de mosquittoMQTT son 1883 y 9001, que corresponden con los de MQTT. Los volúmenes necesarios de mapear son los directorios `/mosquitto/config/ `, `/mosquitto/data` y `/mosquitto/log ` del contenedor en los respectivos directorios creados anteriormente.

El archivo mosquitto.conf es el archivo de configuración principal para el bróker MQTT Mosquitto cuando se ejecuta en un contededor Docker, se encuentra en el directorio `/mosquitto/config`.

Para iniciar el servicio basta con introducir en mosquitto.con una configuración básica como la que se muestra:

```
persistence true
persistence_location /mosquitto/data/
log_dest file /mosquitto/log/mosquitto.log
listener 1883
allow_anonymous true
```

Con persistence true habilita la persistencia de los mensajes que se guardarán en `/mosquitto/data/`. Esto permite mantener la información de los mensajes en caso de que se reinicie el servicio. Con log_dest file indica donde se almacenará el archivo de registro. listener establece el puerto de escucha del bróker. Y por último, allow_anonymous true permite conexiones de clientes anónimos, lo cual puede ser una brecha de seguridad. El siguiente comando en terminal del contenedor creará un archivo de contraseñas.

```
sudo mosquitto_passwd -c /etc/mosquitto/passwd nombredeusuario
```

Sustituyendo `nombredeusuario` por el que se deseé. Este comando encripta las contraseñas y las almacena de forma segura. Y por último agregando en mosquitto.conf:

```
password_file /etc/mosquitto/passwd
allow_anonymous false
```

- Node-RED

En el servicio software de Node-RED se lanzará la imagen ` nodered/node-red:latest` en el contenedor. El puerto de escucha por defecto de node-RED es 1880. El volumen necesario de mapear es el directorio `/data` del contenedor en el respectivo directorio creado anteriormente.

- Hass-configuration

En el servicio software Hass-configuration se lanzará la imagen `causticlab/hass-configurator-docker:latest` en el contenedor. El puerto de escucha por defecto de node-RED es 3218. El volumen necesario de mapear es el directorio `/config` del contenedor con el directorio Docker/home-assistant para permitir la edición de los archivos configuration.yaml y automation.yaml desde este servicio.

- WireGuard

Este servicio costa de dos partes. El servidor VPN Wireguard y Wireguard UI que es una interfaz que facilita la configuración y gestión del servidor VPN. Aunque cabe señalar que Wireguard es desarrollada independientemente por Ngô Duy Khánh [57], es igualmente bastante popular y activamente mantenida.

Para el servidor VPN se lanza la imagen `linuxserver/wireguard:v1.0.20210914-ls7`. La última versión no es estable con Wireguard UI a fecha de realización de este proyecto. El puerto por defecto es 51820/udp. El puerto 5000 permitirá acceder a la interfaz de Wireguard UI.

Para el servicio Wireguard UI se lanza la imagen ` ngoduykhanh/wireguard-ui:latest`.

5.3. Plataforma domótica Home Assistant

Este apartado se centra en aspectos fundamentales para la implementación de la plataforma de control local, es decir, sin necesidad de conexión a Internet, en edificios inteligentes. Se detalla la configuración inicial de Home Assistant, seguida de la integración de los dispositivos LoRaWAN. Posteriormente, se examinan las reglas de automatización de Home Assistant. Finalmente, se abordan temas cruciales como el acceso remoto y la realización de copias de seguridad, componentes esenciales para garantizar la accesibilidad y seguridad del sistema.

5.3.1. Configuración

Para iniciar la configuración de Home Assistant, es necesario acceder al servicio a través de un navegador web, introduciendo la IP asignada a la Raspbery PI con el puerto 8123, el predeterminado de HA. Una vez establecido la conexión, el sistema solicitará la realizacion de una serie de pasos de preconfiguración donde se pide al usuario que información como el idioma, la zona horaria, la ubicación geográfica. Además, se requerirán las credenciales del usuario propietario y administrador de Home Assistant. Estos datos se podrán cambiar en cualquier momento pulsando en el botón del usuario del panel lateral de la interfaz una vez iniciado sesión.

Para establecer la comunicación entre el servicio del broker MQTT y Home Assistant se requiere instalar la integración MQTT en HA. Este proceso se realiza accediendo a los ajustes desde el panel lateral, luego seleccionando las opciones de 'Dispositivos y Servicios' y pulsar el botón '+ Añadir Integración'. A continuación, se busca el servicio MQTT y se selecciona. Se configura la IP y el puerto del bróker MQTT, que por defecto es 1880, y las credenciales de autenticación que se establecieron al desplegar el servicio mosquittoMQTT.

5.3.2. Integración dispositivos

Cada dispositivo está representado en Home Assisntant como una o varias entidades. Cada entidad almacena y proporciona los datos al sistema sobre un sensor, un parámetro o un actuador como un interruptor. Como usuario, nuestra preocupación no debe ser como funciona internamente.

Home Assistant permite crear entidades a partir de una serie de clases predefinidas de entidades que implementan las propiedades y métodos necesarios para la integración de dispositivos en su plataforma.

La creación de estas entidades se realiza editando el archivo 'configuration.yaml' que se encuentra en el directorio /Docker/home-assistant/config/. El contenido del archivo 'configuration.yaml' está adjuntado en el ANEXO II.

En la creación de las entidades del proyecto se utiliza la plataforma 'mqtt' que definirá en las entidades las propiedades necesarias para realizar esa conexión. Para las diferentes entidades se usarán las entidades predefinidas 'sensor', para sensores de valores numéricos, 'binary_sensor', para sensores booleanos, y 'switch' y 'light' para los interruptores.

El dispositivo sensor de temperatura Dragino LHT52 se compone por las entidades sensor de temperatura, sensor de humedad, sensor de voltaje de la batería en milivoltios y un sensor de carga de batería.

El dispositivo sensor de movimiento Milesight WS202 se compone por las entidades binary sensor de movimiento, binary sensor de detección de luz y sensor de la carga de batería.

El interruptor Milesight WS501 se compone por la entidad light, para el actuador y entidades sensor de voltaje, corriente, energía y potencia consumida.

El interruptor Milesight WS502 se compone de dos entidades 'switch' para cada actuados.

El dispositivo enchufe portable inteligente Milesight WS522 se compone por una entidad swicht para el accionador y entidades sensor para voltaje, corriente, energía y potencia consumida, factor de potencia y un indicador de sobretensión booleano.

El dispositivo sensor de temperatura Milesight EM320-TH se compone por entidades sensor de temperatura, humedad y carga de la batería.

Para evitar extender innecesariamente el documento se detallará los aspectos clave utilizando de ejemplo la creación de dos entidades del interruptor Milesight WS501.

```
16   mqtt:
17     sensor:

200      - name: "Office light voltage"
201        unique_id: "light_voltage"
202        state_topic: "edificio_historico/sala1/actuador/luz/data"
203        value_template: "{{ value_json.voltage }}"
204        unit_of_measurement: "V"
205        device_class: voltage
206
207        device:
208          identifiers: "light"
209
```

En el ejemplo se define la entidad correspondiente al sensor de voltaje del interruptor Milesight WS501. Se declara con tipo sensor debido a que el valor es un número. Con 'name' se indica el nombre de la entidad, con 'unique_id' se define el valor de identificación único de la entidad, esto es por que puede darse el caso de que varias entidades tengan el mismo nombre. Con 'state_topic' se indica a qué topic del Brocker MQTT está suscripto la entidad. Es de donde consultará el valor. Con 'value_template' indica que solo lea el valor del campo deseado, en este caso el campo de 'voltage'. 'unit_of_measure' indica la unidad de medida. Con 'device_class' indicamos a Home Assistant que esta entidad es un sensor de voltaje lo que permite una mejor integración en la plataforma. Y por último con 'device' indicamos que esta entidad pertenece al dispositivo con el identificador "light".

```
16   mqtt:

255  #Interruptor de iluminación
256    light:
257      - name: "Office light"
258        unique_id: "light"
259        state_topic: "edificio_historico/sala1/actuador/luz/status"
260        command_topic:"edificio_historico/sala1/actuador/luz/switch"
261        qos: 0
262        payload_on: "ON"
263        payload_off: "OFF"
264
265        device:
266            name: "Interruptor Iluminación"
267            identifiers: "light"
268            manufacturer: "Milesight"
269            model: "WS501-868M"
270            sw_version: "1.1"
271            hw_version: "1.0"
272            configuration_url: "https://www.milesight-
     iot.com/lorawan/switch/ws50x/"
```

En este ejemplo se define la entidad correspondiente al actuador del interruptor Milesight WS501. Se define con la clase 'light' que es la que Home Assistant maneja como interruptor de iluminación. Los campos 'name' y 'unique_id' igual que antes sirven para identificar a la entidad. Sin embargo, obligatoriamente hay dos topics a que suscribirse. 'state_topic' es el topic en el que la entidad consultará el estado del interruptor, es decir, si se encuentra encendido o apagado. Mientras que 'command_topic' es donde la entidad pública las órdenes de encender o apagar el interruptor. Con 'payload_on' indicamos el valor que espera la entidad para el estado encendido, en este caso, si lee "ON" interpretará que el interruptor se encuentra encendido. Ojo, no es lo mismo "on" que "On" que "ON". Y 'payload_off' indicamos el caso contrario, para el valor del estado apagado. Y por último con 'device' indicamos que pertenece al dispositivo con el indicador "light" y a la vez describimos las demás características del dispositivo como el nombre, el fabricante, el modelo, la versión de software y hardware y una URL que en este caso redirige a la página web del fabricante donde se encuentra la descripción y los documentos relacionados con el dispositivo.

El gateway publica en el Broker MQTT la información de los diferentes dispositivos en formato JSON, que contiene diferentes campos de datos como devEUI, deviceName, fPort, data y data_encode, entre otros. Nos centraremos en el campo 'data', que contiene la información trasmitida por el dispositivo, y el campo 'data_encode' que indica la codificación del campo 'data'. Con la pasarela usada en este proyecto la codificación por defecto es Base64. Mientras que con los dispositivos LoRaWAN manejan *payloads* en formato hexadecimal como se indica en sus respectivos documentos.

El primer paso para poder interpretar la información recibida es decodificar el valor de 'data'. Esto nos permitirá trabajar con el valor hexadecimal. Como cada fabricante en cada dispositivo ha codificado la información de los diferentes sensores en la forma que optaron conveniente, es necesario abordar cada dispositivo individualmente consultando la respectiva documentación.

Para este objetivo se hará uso de la herramienta Node-RED. Tiene tres elementos básicos:

- **Nodos** representados como broques, son los diferentes servicios o funciones.
- **Flows** o flujos, son las conexiones entre los nodos. Representa la secuencia de eventos o la lógica del programa.
- **Mensages (msg)** son objetos JavaScript que transporta la información entre nodos. Esto es, cada nodo recibe un mensaje de entrada, realiza una operación y envía un mensaje de salida.

A continuación se detallarán las partes más destacables de los programas desarrollados a través del servicio docker Node-RED.

- Función decodificar Base64

Como cada dispositivo necesita decodificar el campo 'data' de 'base64' a hexadecimal es conveniente crear un nodo para esta función.

Para ello se arrastra el tipo de nodo 'funtion' del desplegable de la izquierda a la pantalla de trabajo. Al pulsar sobre el nodo se abre la interfaz de edición del nodo (Figura 53). En el campo 'Name' se indica nombre del nodo deseado. Y en la sección 'On Message' se indica el comportamiento del nodo mediante código en JavaScript.

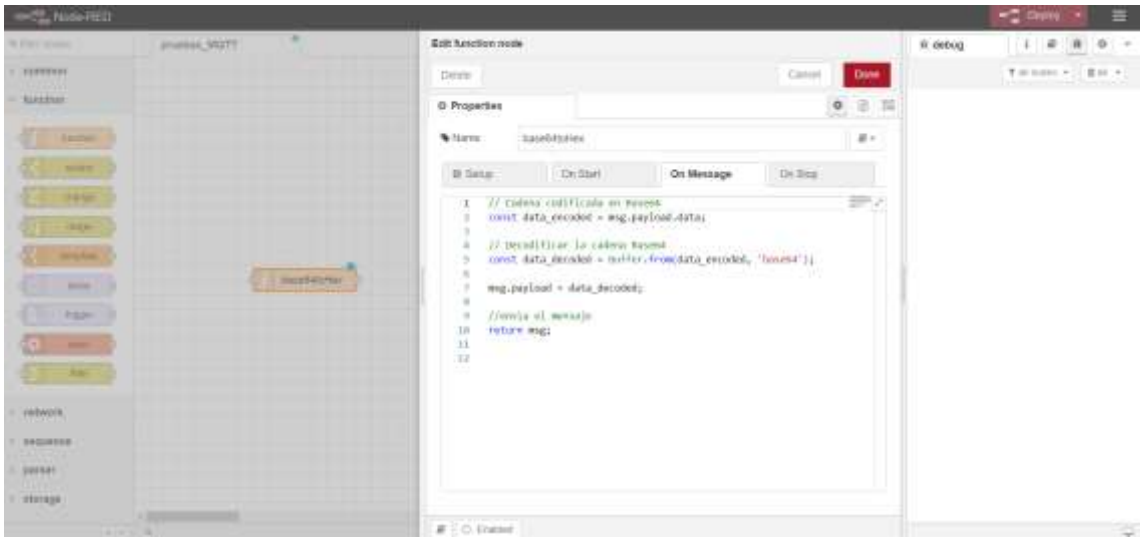


Figura 53. Nodo Base64toHex.

En código extrae el valor del campo 'data' del mensaje para luego mediante la función 'Buffer.from' decodificarla. El resultado lo guarda en el mensaje de envío al siguiente nodo.

- Sensor de Temperatura Dragino LHT52

El sensor de Dragino por defecto envía cada veinte minutos un payload con la información codificada con la estructura mostrada en la Tabla 1. Payload del dispositivo Dragino LHT52 en fPort 2.

Tabla 1. Payload del dispositivo Dragino LHT52 en fPort 2.

Size(bytes)	2	2	2	1	4
Value	Temperature	Humidity	External Temperature	Ext#	Unix TimeStamp

Teniendo en cuenta que el rango de temperatura del sensor es de -20°C a 50 °C con una resolución de 0,01°C. Y una resolución de 0,1 %RH para el sensor de humedad.

Por ejemplo para un payload (fPort = 2) de 08 CD 02 20 7F FF 01 61 CD 4E DD

La temperatura sería igual a 0x08CD/100 = 22,53°C

Y la humedad sería igual a 0x0220/10 = 54,4%

Y por fPort 5 se envía cada doce horas información sobre el estado del dispositivo codificado con la estructura mostrada en la Tabla 1Tabla 2.

Tabla 2. Payload del dispositivo Dragino LHT52 en fPort 5

Size(bytes)	1	2	1	1	2
Value	Sensor Model	Firmeware Version	Frequency Band	Sub-band	Batery (mV)

El programa intérprete (Figura 54) recibe el payload mediante el nodo MQTT suscrito al topic 'application/5/device/{device_ID}/rx' que es donde el Gateway publica los mensajes de este dispositivo. El nodo switch nombrado 'fPort' separa el flujo de mensaje según el valor de fPort. Después el mensaje pasa por el nodo 'base64toHex' descrito anteriormente y por el nodo que se encarga de interpretar el payload hexadecimal para extraer los valores de temperatura, humedad, batería, etc. El código de cada nodo 'interpretación de datos' se encuentra adjunto en ANEXO III. Y por último se publica a través del nodo MQTT el mensaje en formato JSON con los diferentes campos para cada valor (temperatura, humedad…) en el topic 'edificio_historico/sala1/sensores/temperatura/state' que corresponde con el que las entidades de Home Assistant leerán.

Figura 54. Diagrama de flujo del programa para interpretar la información del sensor de temperatura Dragino en Node-RED.

- Sensor de Movimiento Milesight WS202

De forma predeterminada el sensor WS202 transmite los datos del sensor y el nivel de batería cada treinta minutos o cuando cambie el estado de detección de movimiento o luminosidad. El contenido del mensaje se estructura de la Tabla 3:

Tabla 3. Payload del sensor Milesight WS202.

Canal	Tipo	Descripción

01	75 (nivel de batería)	uint8, Unidad: %
03	00 (PIR Status)	01: Occupied 00: Vacant
04	00 (Light Status)	01: Bright 00: Dark

El programa para este dispositivo recoge el mensaje a través del nodo de suscripción MQTT al topic 'application/5/device/{device_ID}/rx'. El nodo de interpretación de datos está basado en el código facilitado por Milesight en github [58], su contenido se encuentra adjunto en ANEXO III. Y por último se publica los datos finales en el topic "edificio_historico/sala1/sensores/movimiento/state" (Figura 55).

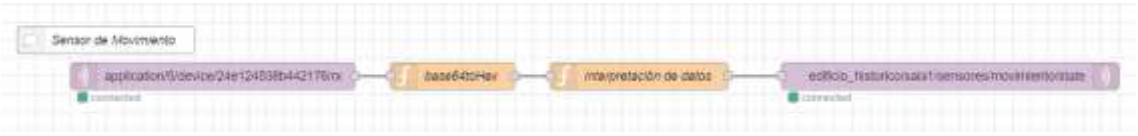


Figura 55. Diagrama de flujo del programa para interpretar la información del sensor de movimiento Milesgiht en Node-RED.

- Interruptor Inteligente Milesight WS501

El programa para el interruptor costa de dos partes. La parte de recibir el mensaje y la parte de enviar los comandos para encender o apagar.

El dispositivo Milesight WS501 envía de forma predeterminada los datos de los sensores cada veinte minutos. Además, cuando el estado del interruptor cambia envía la actualización del mismo.

El programa recibe el mensaje a través del nodo de suscripción MQTT al topic 'application/5/device/{device_ID}/rx'. Tras pasarlo de base64 a hexadecimal se extrae la información por la función "interpretación de datos" cuyo código adjuntado en ANEXO III está basado en código facilitado por Milesight en Github [58]. Los datos sobre el voltaje, amperaje, potencia y energía se publican al topic 'edificio_historico/sala1/actuador/luz/data'. Mientras que mediante el nodo switch se valora el estado del interruptor para publicar en el topic "edificio_historico/sala1/actuador/luz/status" el mensaje "ON" o "OFF" según el caso.

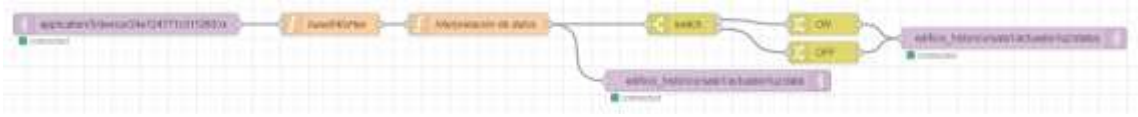


Figura 56. Diagrama de del programa para interpretar la información recibida del interruptor WS501 de Milesgiht en Node-RED.

El interruptor WS501 puede recibir una variedad de comandos desde el simple abrir o cerrar hasta comandos de abrir y después de cierto tiempo indicado cerrar, o viceversa. Sin embargo, para el presente proyecto solo se necesita enviar comandos de abrir o cerrar el interruptor, pues las demás funciones pueden delegarse en la plataforma domótica Home Assistant.

El interruptor espera un mensaje en el canal 08 con el contenido del byte 1 indicando que interruptores se accionan con los bits de 0 al 3 y a que estado cambian con los bits del 4 al

7. Mientras que el byte 2 está reservado, por lo que espera el valor hexadecimal 'FF'. Por lo que para encender se envía el comando en hexadecimal '08 11 FF', que traducido a base64 es 'CBD/', mientas que para apagar se envía el comando '08 10 FF', que traducido a base64 es 'CBH/'.

La parte del programa para enviar las instrucciones recibe las mismas a través del nodo de suscripción MQTT del topic 'edificio_historico/sala1/actuador/luz/swicht'. Acto seguido se actualiza el nuevo estado con el nodo de publicación MQTT al topic 'oficina/actuador/status'. El nodo 'switch' evalúa la instrucción, si recibe "ON" dirige el mensaje al nodo 'Encender' que cargará el mensaje en base64 "CBD/", si recibe "OFF" dirige el mensaje al nodo 'Apagar' que cargará el mensaje en base64 "CBH/". Por último, el nodo de publicación MQTT publica el mensaje al topic 'application/{id_app}/device/{device_ID}/tx' donde la aplicación del dispositivo en el Gateway está suscrita (Figura 57).

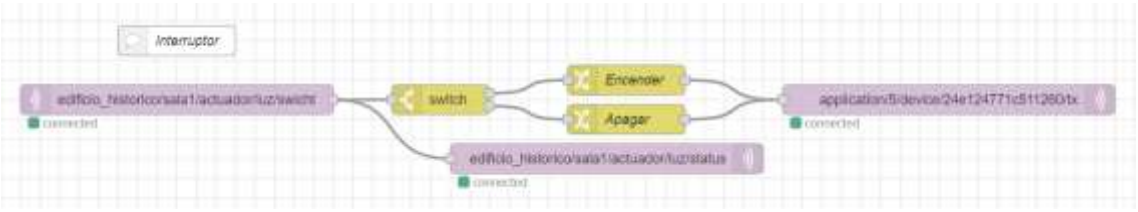


Figura 57. Diagrama de del programa para enviar instrucciones del interruptor WS501 de Milesgiht en Node-RED.

Además de las instrucciones hay que indicar al Gateway a qué dispositivo van dirigidas. Para ello es necesario añadir a los nodos 'Enceder' y 'Apagar' los siguientes campos:

```
{
    "applicationID": "5",
    "applicationName": "RAK",
    "devEUI": "24e124771c511260",
    "deviceName": "interruptor_WS501",
    "fPort": 85,
    "data": <instrucciones>, //Apagar: "CBH/", Encender: "CBD/"
    "data_encode": "base64"
}
```

Cambiando <instrucciones> por el valor que corresponda.

- Interruptor Inteligente Milesight WS502

Este programa es similar en funcionamiento al anterior, salvo por la diferencia de que este interruptor costa de dos accionadores.

La diferencia en la parte de recepción de los datos se observa en la salida del nodo 'Interpretación de datos' el mensaje se divide en dos caminos. Uno donde se evalúa el cambio de estado del 'switch 1' y se publica en el topic 'edificio_historico/sala1/actuador/ventilacion'. Y el otro de igual manera evalúa el cambio de estado del 'switch 2' y se publica en el topic 'edificio_historico/sala1/actuador/calefaccion'. (Figura 58)

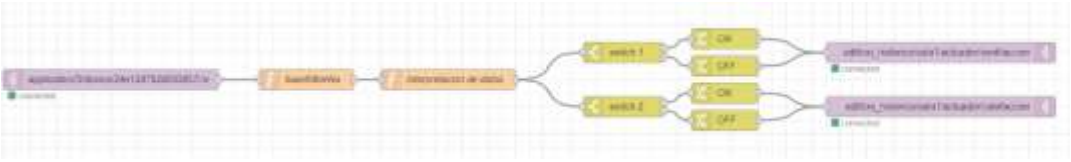


Figura 58. Diagrama de del programa para interpretar la información recibida del interruptor WS502 de Milesgiht en Node-RED.

El interruptor WS502 espera un mensaje con la misma estructura de contenido que el interruptor WS501 visto anteriormente. Para el desarrollo de la parte del programa de enviar las instrucciones no se ha considerado la situación de accionar los dos contactos del interruptor de forma simultánea.

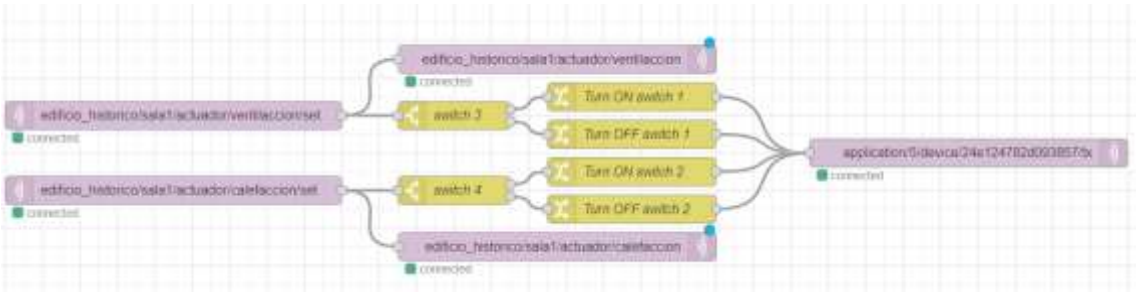


Figura 59. Diagrama de del programa para enviar instrucciones del interruptor WS502 de Milesgiht en Node-RED.

Home Assistant publica las instrucciones de cambio de estado del 'switch 1' en el topic 'edificio_historico/sala1/actuador/ventilacion/set' mientras que las del 'switch 2' en el topic 'edificio_historico/sala1/actuador/calefaccion/set'. Los nodos de publicación MQTT de los topics 'edificio_historico/sala1/actuador/ventilacion' y 'edificio_historico/sala1/actuador/calefaccion' actualizan los cambios de estado. El nodo 'switch 1' evalúa el cambio de estado de este, si recibe "ON" dirige el mensaje al nodo 'Turn ON switch 1' que cargará el mensaje en base64 "CBD/", si recibe "OFF" dirige el mensaje al nodo 'Turn OFF switch 1' que cargará el mensaje en base64 "CBH/". Mientras que el nodo 'switch 2' evalúa el cambio de estado de este, si recibe "ON" dirige el mensaje al nodo 'Turn ON switch 2' que cargará el mensaje en base64 "CCL/", equivalente a hexadecimal a '08 22 FF', si recibe "OFF" dirige el mensaje al nodo 'Turn OFF switch 2' que cargará el mensaje en base64 "CCD/", equivalente en hexadecimal a '08 20 FF'. Por último el mensaje se publica con el nodo de publicación MQTT al topic 'application/5/device/{device_ID}/tx'. Al igual que en el dispositivo anterior se necesita indicar al Gateway a qué dispositivo va dirigido el mensaje, por lo que hay que agregar los siguientes campos en los nodos 'Turn ON/OFF switch 1/2':

```
{
    "applicationID": "5",
    "applicationName": "RAK",
    "devEUI": "24e124782d093857",

    "deviceName": "interruptor_WS502",
    "fPort": 85,
    "data": <instrucciones>,
    "data_encode": "base64"
```

}

Cambiando <instrucciones> por el valor que corresponda.

- Enchufe portable inteligente Milesight WS523

El programa para el enchufe inteligente costa de dos partes. La parte de recibir el mensaje con la información de los sensores y la parte de enviar los comandos para abrir o cerrar el enchufe.

El dispositivo Milesight WS523 envía de forma predeterminada los datos de los sensores cada veinte minutos. Además, cuando el estado del interruptor cambia envía la actualización del mismo.

El programa lee el mensaje a través del nodo de suscripción MQTT al topic 'application/5/device/{device_ID}/rx'. Tras pasarlo de base64 a hexadecimal se extrae la información por la función "interpretación de datos" cuyo código adjuntado en ANEXO III está basado en código facilitado por Milesight en Github [58]. Los datos sobre el voltaje, amperaje, potencia y energía se publican al topic 'edificio_historico/sala1/sensores/power/state'. Mientras que mediante el nodo switch se valora el estado del interruptor para publicar en el topic 'edificio_historico/sala1/sensores/power/switch' el mensaje "ON" o "OFF" según el caso. (Figura 60)

Figura 60. Diagrama de del programa de interpretar datos del enchufe inteligente WS523 de Milesgiht en Node-RED

La parte del programa para enviar las instrucciones recibe las mismas a través del nodo de suscripción MQTT del topic 'edificio_historico/sala1/sensores/power/switch'. Acto seguido se actualiza el nuevo estado con el nodo de publicación MQTT al topic 'edificio_historico/sala1/sensores/power/state'. El nodo 'switch' evalúa la instrucción, si recibe "ON" dirige el mensaje al nodo '*turn open*' que cargará el mensaje en base64 "CAEA/w==", si recibe "OFF" dirige el mensaje al nodo '*turn close*' que cargará el mensaje en base64 "CAAA/w==". Por último, el nodo de publicación MQTT publica el mensaje al topic 'application/5/device/{device_ID}/tx' donde la aplicación del dispositivo en el Gateway está suscrita. (Figura 61).

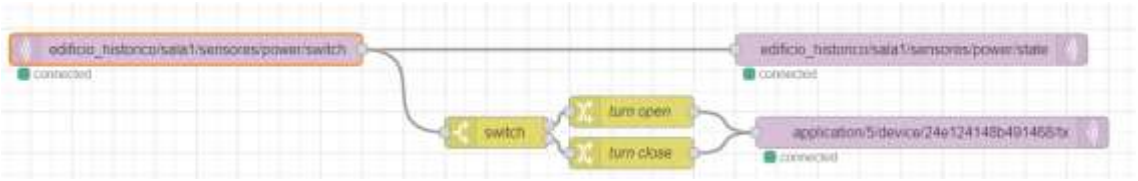


Figura 61. Diagrama de del programa para enviar instrucciones al enchufe inteligente WS523 de Milesgiht en Node-RED

- Sensor de temperatura Milesgiht ME320-HM

De forma similar al sensor de movimiento Milesight WS202, el sensor de temperatura Milesight ME320-HM enviar la información de sus sensores cada diez minutos de forma predeterminada. El contenido del mensaje se desglosa en la siguiente tabla.

Tabla 4. Payload sensor Milesight ME320-HM.

Sensor	Canal	Tipo	Descriptión
Nivel de Batería	01	75	UINT8, unidad: %
Temperatura	03	67	INT16, Unidad: ºC, Resolución: 0,1 ºC
Humedad	04	68	UINT8, Unidad: %RH, Resolución: 0,5 %RH

El programa para este dispositivo recoge el mensaje a través del nodo de suscripción MQTT al topic 'application/5/device/{device_ID}/rx'. El nodo de interpretación de datos está basado en el código facilitado por Milesight en githab [58], su contenido se encuentra adjunto en ANEXO III. Y por último se publica los datos finales en el topic 'edificio_historico/sala2/sensores/temperatura/state' (Figura 62).

Figura 62. Diagrama de del programa para interpretar datos del sensor de temperatura ME320-HM de Milesgiht en Node-RED

5.3.3. Reglas de automatizaciones

Las reglas de automatización se componen de tres partes: disparador o *trigger*, condición y la acción.

Para el estudio de las automatizaciones atendamos al siguiente ejemplo, "Al atardecer, los días laborales, enciende la luz del salón".

Trigger → Al atardecer

Los disparadores *trigger* son los que inician el procesamiento de una regla de automatización. Es posible especificar varios *triggers* para la misma regla y cuando cualquiera de los *trigger* se convierta en verdadero, se iniciará la automatización. Una vez que salta un *trigger* Home Assistante valuará las condiciones, si las hubiera, y activará la acción. En este caso, cuando amanezca.

Condición → Los días laborales

Las condiciones son pruebas opcionales que pueden delimitar una regla de automatización para que solo funcione en casos de uso específicos. Una condición prueba un estado actual del sistema, como en este ejemplo, si estamos en un día laboral.

Acción → Enciende las luces del salón

La acción se realiza cuando se activa un disparador y se cumplen todas las condiciones. En este ejemplo, se encienden las luces.

Las reglas de automatización interactúan directamente con el estado interno de Home Assistant, es decir, con los estados, los cuales se muestran en la sección herramientas del desarrollador de la interfaz de Home Assistant, disponible en la parte inferior de la barra lateral. En icono '<>' mostrará todos los estados disponibles actualmente. Una entidad corresponde a cualquier sensor inteligente o variable, una bombilla, un interruptor, una persona y calendarios.

Entre los parámetros que definen una entidad, la *ID_entity,* el estado y los atributos son los necesarios para crear una regla de automatización.

- ***ID_entidad***: identifica de forma única a la entidad. Ejemplo: light.salon.
- **Estado**: es el estado actual del dispositivo o sensor. Ejemplo: encendido.
- **Atributos**: son datos extra relacionados con el estado. Ejemplo: brillo.

Los cambios de estado se pueden utilizar como *trigger* y el estado actual se puede usar como condición.

Las acciones llaman a los servicios, y para ver los servicios disponibles accedemos a la herramienta de desarrollo de servicios. Cada servicio tiene un dominio y un nombre. Por ejemplo, el servicio *light.turn_on* se encarga de encender cualquier luz del sistema, es decir, puede encender todas las luces de una habitación en concreto o de un grupo de luces. Los servicios pueden pasar parámetros para, por ejemplo, indicar qué dispositivo activar o qué color usar al encender una luz.

En las versiones más recientes de Home Assistant, hay dos formas de crear automatizaciones. La forma *legacy* a través de la edición del archivo *automatization.yaml*, encontrado en el directorio /config, junto al fichero *configuration.yaml*. O a través de la interfaz de usuario con un editor visual UI.

Mediante la edición del archivo *automatization.yaml*, el ejemplo anterior de "Al atardecer, los días laborales, enciende la luz del salón" quedaría de la siguiente forma:

```
automation:
# Ejemplo automatizacion
 - alias: 'Enciende las luces'
    trigger:
      - platform: sun
        event: sunset
        offset: 0
    condition:
      - condition: state
        entity_id: binary_sensor.workday_sensor
        state: "on"
    action:
      - service: light.turn_on
        data: {}
        target:
          area_id: salon
```

La automatización recibe el nombre de 'Enciende las luces'. Tiene un disparador que es la caída del sol detectada por Home Assistant con el componente que le viene integrado por defecto. Tiene una condición del sensor *workday*, que es un boolean, es decir, devuelve 'on' cuando HA detecta que estamos en un día laboral y devuelve 'off' en caso contrario. En este caso la condición es que el estado sea 'on'. Entonces, cumpliendo con estas condiciones se encienden todas las luces del salón haciendo uso del servicio *light.turn_on*.

Para crear una regla de automatización en Home Assistant a través de la interfaz UI, se navega hasta la página de automatizaciones haciendo clic en `Ajustes` de la barra lateral y luego en `Automatizaciones`. En esta página, se crea una nueva automatización haciendo clic en el botón `+ Crear automatización`. A continuación, se especifican los detalles de la automatización, esto es el *trigger* o desencadenante, las condiciones y la acción. Para finalizar, se guarda la automatización haciendo clic en el botón `Guardar`, introduciendo el nombre de la automatización y, si se desea, una breve descripción.

5.3.4. Acceso remoto

Home Assistant por defecto solo permite el acceso desde redes LAN. Aunque se puede habilitar la opción para permitir el acceso desde Internet, esta opción no es la más idónea debido a que cualquiera desde internet tendrá la capacidad de acceder al sistema, resultando en una vulnerabilidad de seguridad. En su lugar se ha decidido utilizar VPN con el software de código libre WireGuard.

Para la configuración del sistema, se debe accede desde un navegador web introduciendo la dirección IP correspondiente a la Raspberry Pi junto con el puerto 5000, tal como se especificó durante la creación del contenedor Docker. Solicitará autenticarnos, tal como se estableció durante la creación del contenedor Docker, el usuario y contraseña por defecto es 'admin' y 'password' respectivamente. Se recomienda encarecidamente cambiar la contraseña. Para hacerlo, se debe seleccionar el botón de usuario del panel lateral, introducir la nueva contraseña en el campo correspondiente y hacer clic en el botón 'Update' para guardar el cambio (véase Figura 63).

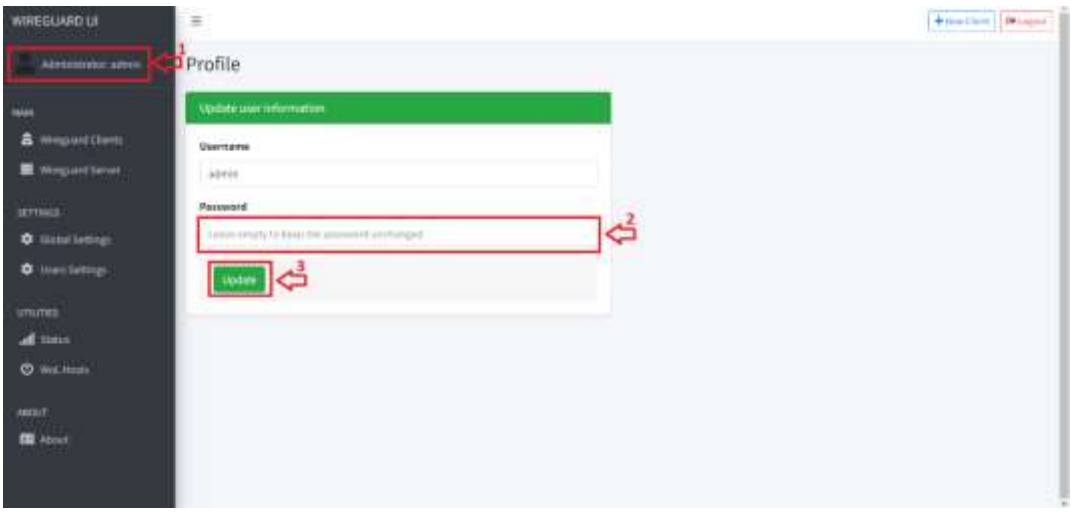


Figura 63. Pasos para cambiar la contraseña en WireGuard UI.

Para la puesta en marcha del servidor VPN, es imprescindible la configuración de los scripts PostUp y PostDown. Estos scripts contienen comandos que se ejecutan tras la activación (PostUp) y previo a la desactivación (PostDown) de la interfaz de Wireguard. En este contexto, el término 'interfaz' se refiere a una conexión de red virtual creada específicamente para gestionar el tráfico de la VPN. Para realizar esta configuración, se debe acceder a la sección 'Wireguard Server' ubicada en el panel lateral de la interfaz (Figura 64).

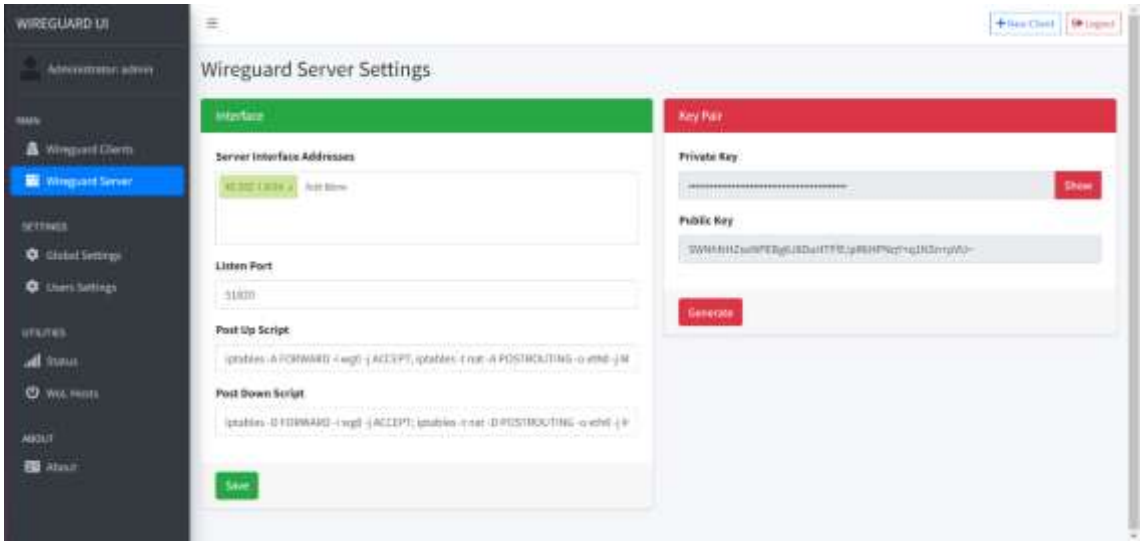


Figura 64. Configuración del servidor Wireguard desde Wireguard UI.

En el Post Up script, es necesario introducir el siguiente comando como una configuración básica:

iptables -A FORWARD -i wg0 -j ACCEPT; iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE

Este comando realiza dos tareas. En primer lugar, permite que todos los paquetes que llegan a la interfaz 'wg0' de Wireguard sean reenviados a otras interfaces. Sin esta regla, los paquetes que llegan a la interfaz 'wg0' no se reenviarían, y los clientes de la VPN no podrían acceder a la red a la que el servidor VPN está conectado. Y en segundo lugar, realiza una operación de NAT en los paquetes que salen de la interfaz 'eth0'. Esto significa que los paquetes que salen de la interfaz 'eth0' aparecerán como si se originaran en el propio servidor, lo que permite poder acceder a Home Assistant como si el cliente estuviera en una red LAN.

En Post Down Script es necesario introducir el siguiente comando como una configuración básica:

iptables -D FORWARD -i wg0 -j ACCEPT; iptables -t nat -D POSTROUTING -o eth0 -j MASQUERADE

Este comando elimina las reglas que se agregaron en el Post Up script, deteniendo el reenvío de paquetes y la operación de NAT cuando la interfaz de Wireguard se desactiva.

Finalmente, para vincular distintos dispositivos al servidor VPN, es necesario que estos tengan instalada la aplicación cliente de Wireguard. Para añadir un cliente al servidor, se debe hacer clic en el botón '+ New client'. Se abrirá una ventana para introducir diversos parámetros (véase Figura 64). Estos incluyen el nombre, el correo electrónico (opcional), la dirección IP

asignada en la red virtual, y las direcciones IP permitidas. Por defecto, se establece 0.0.0.0/0, lo que indica que este nuevo cliente tendrá acceso a toda la red. Para guardar la configuración, se debe hacer clic en el botón 'Submit'. Esto generará las claves de cifrado y otros archivos necesarios para este cliente.

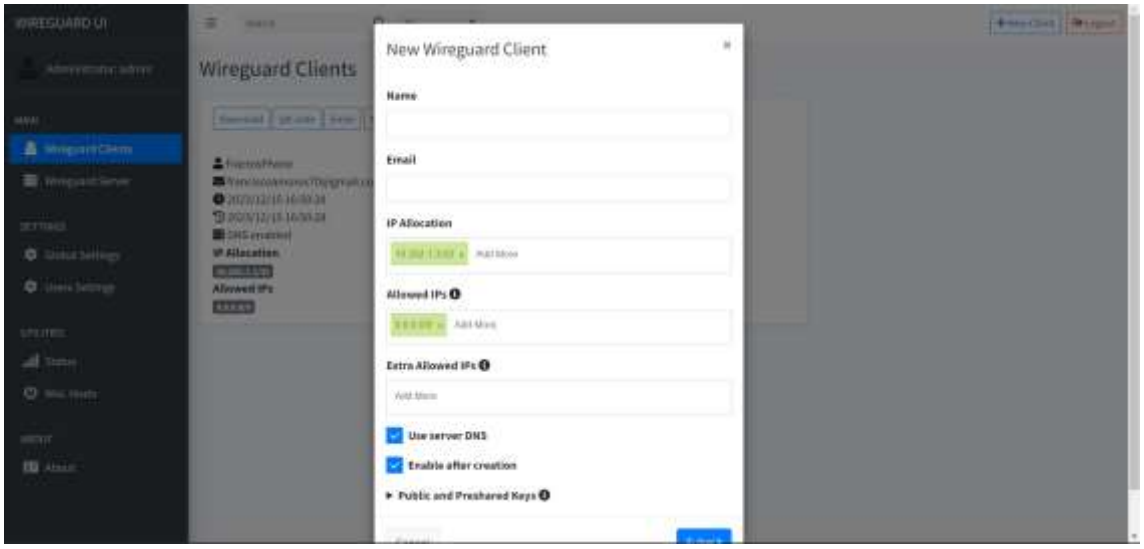


Figura 65. Agregar clientes Wireguard desde Wireguard UI.

Para vincular el cliente con el dispositivo, es necesario transferir estas claves al dispositivo correspondiente. En el caso de un teléfono móvil, el proceso es tan sencillo como escanear el código QR generado en el servidor utilizando el escáner QR de la aplicación cliente de Wireguard. Este código QR se encuentra haciendo clic en el botón 'QR code' de ese cliente en la sección 'Wireguard Clients' del panel lateral de la interfaz.

Para un ordenador, es necesario transferir el archivo al dispositivo correspondiente, ya sea vía correo electrónico o descargando el archivo utilizando los botones correspondientes. Una vez en el dispositivo, con la aplicación cliente de Wireguard, se debe seleccionar ese archivo.

5.3.5. Copia de seguridad

Es posible realizar copias de seguridad de Home Assistant a través de las opciones de sistema disponibles en los ajustes. En Node-Red, se tiene la opción de exportar e importar todos los flujos de trabajo creados, ya sea en conjunto o individualmente. No obstante, en el escenario de un fallo total del sistema, estas medidas no proporcionan una solución eficiente para restaurar todo el sistema.

Por lo tanto, previamente hemos mapeado los archivos persistentes durante el despliegue de servicios en contenedores Docker al directorio /Docker de la Raspberry Pi. La solución propuesta consiste en generar un archivo comprimido del directorio /Docker utilizando el comando `tar` y almacenar este archivo en una plataforma segura, como un disco externo o un servicio de almacenamiento en la nube.

```
tar -czvf backup.tar.gz /Docker
```

Para restaurar el proyecto, sería suficiente descomprimir el archivo backup.tar.gz en el directorio /Docker y ejecutar el archivo docker-compose.yml, que se encuentra en el Anexo I y que también se ha guardado en la copia de seguridad. Al hacer esto, se desplegarán los servicios Docker con toda su información y configuración restaurada. El comando utilizado para descomprimir es: `tar -xzvf backup.tar.gz`

5.4. Prueba de concepto

En la realización de la prueba de concepto se ha planteado la situación de un edificio patrimonial donde se desea monitorizar la temperatura y humedad y automatizar un deshumidificador, la calefacción, la ventilación y el encendido y apagado de la iluminación.

5.4.1. Visualización y monitorización

Para la visualización de los datos se ha decidido crear tres dashboards. El primero a modo de resumen del estado de los diferentes sensores y control de los interruptores y las automatizaciones nombrado “Resumen”. El segundo como monitorización de la temperatura y humedad con el icono de temperatura. Y el tercero como monitorización del consumo energético nombrado “Energía”.

La edición de los dashboard se realiza agregando tarjetas pulsando el botón de tres puntos de arriba a la derecha de la interfaz del usuario administrador. Para el dashboard “Resumen” está compuesto por tarjetas tipo lista de entidades para mostrar los diferentes sensores de cada dispositivo de forma agrupada, tarjeta tipo botón para accionar los interruptores. Además, se ha añadido tres entidades virtuales para el control de las automatizaciones, un selector con las opciones manual y automático, y dos entradas de valores numéricos (Figura 66).

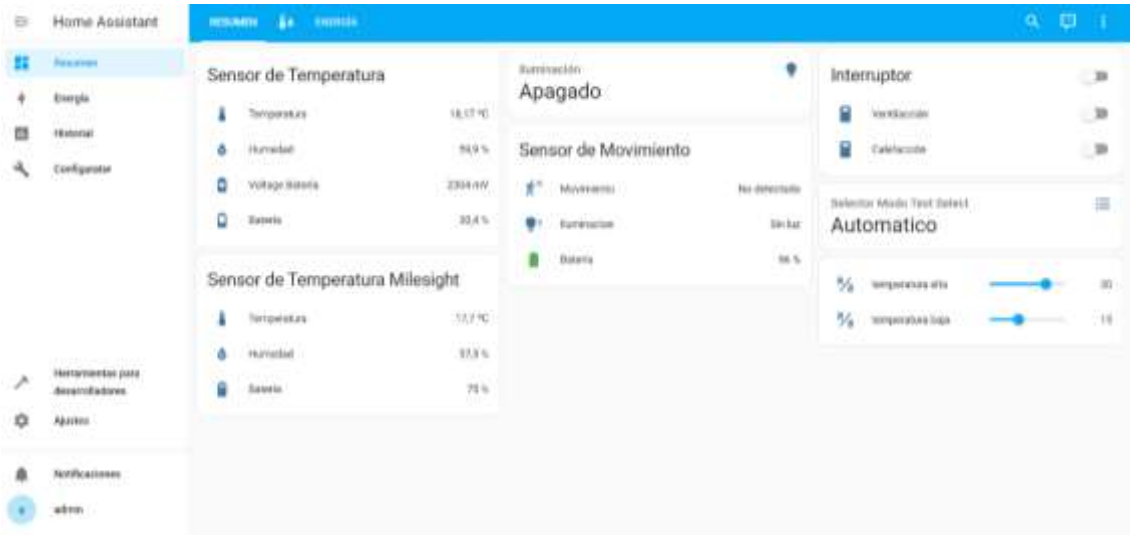


Figura 66. Dashboard Resumen, con los sensores de temperatura y movimiento y el control de los interruptores y opciones de automatización.

El dashboard de monitorización de temperatura y humedad está compuesto por dos tarjetas tipo grafico para la evolución de la temperatura y humedad y tarjetas tipo estadísticas para mostrar los valores medios, máximos y mínimos de la temperatura y de la humedad (Figura 67).

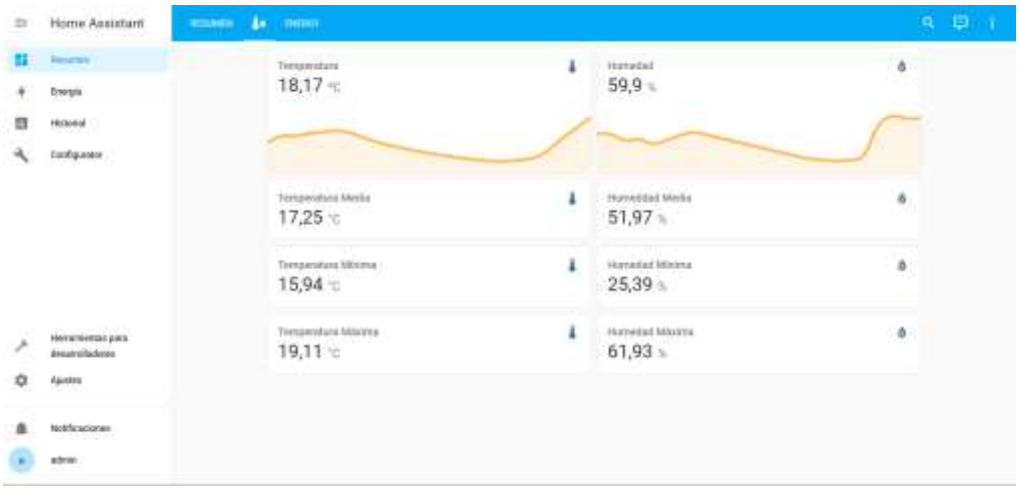


Figura 67. Dashboard de monitorización de temperatura y humedad.

El dashboard "Energía" está compuesto por dos tarjetas de lista de entidades. La primera tarjeta muestra los sensores de potencia, corriente, voltaje y consumo del interruptor de iluminación. Y la segunda con las entidades que compone el dispositivo del enchufe inteligente de Milesight. Además, de la entidad con los precios del kilovatio hora del PVPC que permitirá calcular en gasto energético en las opciones de la pestaña "Energía" del panel lateral de la interfaz (Figura 69).

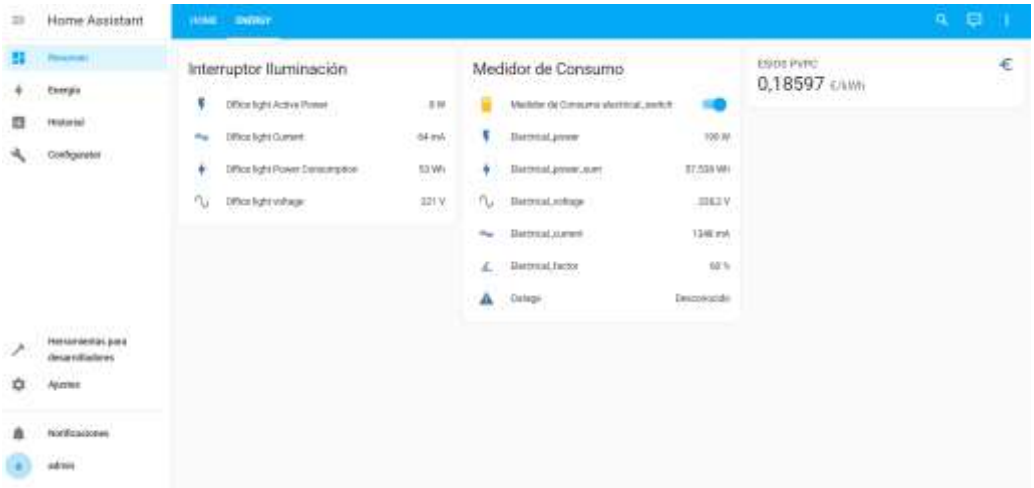


Figura 68. Dashboard de monitorización de consumo eléctrico.

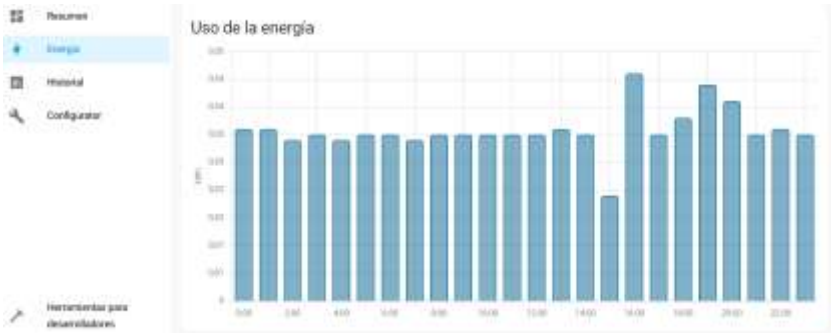


Figura 69. Dashboard de visualización de consumo eléctrico mediante las opciones gráficas de Home Assistant.

5.4.2. Automatizaciones

Se ha decidido crear tres entidades virtuales. Una entidad de tipo selector, donde se desplegará las opciones 'Automático' y 'Manual' para activar o desactivar todas las automatizaciones desde el dashboard de Home Assistant. Las otras dos entidades son de tipo *input_number* que servirán para asignar los valores de temperatura donde se desea encender o apagar la calefacción o la ventilación respectivamente.

Automatización de la iluminación

Control automático de la iluminación a partir de la presencia de visitantes. Se disminuye el problema de la degradación por luz y se mejora el consumo eléctrico. Para esta automatización se utiliza el interruptor Milesight WS501 y el sensor de movimiento Milesight WS202.

Esta automatización encenderá la iluminación cuando el sensor detecte movimiento y la estancia esté oscura, evitando el encendido de la luminaria cuando la luz solar entre por las ventanas. También se encenderá al anochecer si la estancia está ocupada. Y la iluminación se apagará cuando la estancia esté vacía.

La lógica de la automatización está compuesta por dos automatizaciones simples, una que se encarga de encender la iluminación y la otra de apagarla.

La automatización del encendido utiliza dos disparadores, cuando el sensor de movimiento empezó a detectar movimiento y cuando el sensor de movimiento dejó de detectar luz. Tiene tres condiciones: confirmar que el selector de modo está en Automático, Confirmar que el sensor de movimiento no detecta luz y confirmar que el sensor de movimiento ha detectado movimiento. Y por último, llama al servicio 'Luz: Encender'.

La automatización del apagado utiliza como disparador el momento que el sensor de movimiento deja de detectar movimiento durante más de un minuto. Comprueba que el selector de modo está en Automático como condición. Y por último, llama al servicio 'Luz:Apagar'.

Automatización de un deshumidificador

Control automático de un deshumidificador con el fin de mantener controlados los niveles de humedad importante para la preservación de los materiales históricos del edificio. Además, al ajustar la humedad solo cuando es necesario se contribuye a la eficiencia energética.

Esta automatización utiliza el enchufe inteligente Milesight WS523 para activar y desactivar el deshumidificador y el sensor de temperatura y humedad de Dragino LHT52.

La lógica de la automatización está compuesta por dos automatizaciones simples, una que se encarga de activar el deshumidificador y la otra de desactivarlo.

La automatización de activado tiene como disparador cuando el sensor de humedad supere 65%RM. Comprueba que el selector de modo esté en automático. Y, por último, activa el interruptor del enchufe inteligente.
La automatización de desactivado tiene como disparador cuando el sensor de humedad baje de 35%RM. Comprueba que le selector de modo esté en automático. Y, por último, desactiva el interruptor del enchufe inteligente.

Automatización de la calefacción y la ventilación

Control de la calefacción y la ventilación en edificios patrimoniales permite mantener un ambiente estable, crucial para la preservación de los materiales históricos. Además, al ajustar la temperatura solo cuando es necesario, se minimiza el uso de energía, contribuyendo a la eficiencia energética. Esta automatización utiliza para accionar la calefacción y la ventilación el interruptor de dos contactos de Milesight WS502 y el sensor de temperatura y humedad de Dragino LHT52. Además, de las entidades vitales de tipo *input_number* nombradas 'temperatura alta' y 'temperatura baja'

La lógica de la automatización se compone por los disparadores que se activan cuando el sensor de temperatura supera el valor asignado al 'temperatura_alta', cuando el sensor de temperatura baja del valor asignado al 'temperatura_baja' y cuando los valores de 'temperatura_alta' o 'temperatura_baja' cambia. Comprueba que el selector de modo está en automático, como condición. Una vez activada la automatización y cumplida la condición, se ejecutan dos acciones condicionales. La primera acción enciende la ventilación si la temperatura está por encima de 'temperatura_alta' y la apaga en caso contrario. La segunda acción enciende la calefacción si la temperatura está por debajo de 'temperatura_baja' y la apaga en caso contrario.

5.4.3. Prueba de Funcionamiento

Durante un periodo continuo de 180 días, el sistema ha demostrado un rendimiento óptimo y estable, sin presentar incidencias ni interrupciones. Se ha verificado la eficacia y precisión de las reglas de automatización en una variedad de escenarios, confirmando su correcta implementación y funcionamiento. Además, se ha logrado un control remoto efectivo de los dispositivos IoT a través de la VPN.

6. CONCLUSIONES

El proyecto ha estado enfocado como un prototipo a modo de prueba de concepto con la intención de explorar la viabilidad de un sistema inalámbrico sin conexión a Internet en edificios inteligentes. Para ello se ha realizado una gran labor del estudio del arte en tecnologías del internet de las cosas.

Se eligió para el proyecto la plataforma domótica Home Assistant. Como red inalámbrica, tras el estudio de diferentes tecnologías y el planteamiento de la hipotética situación de instalar el sistema en un edificio patrimonial, se eligió la tecnología LoRaWAN.

Se ha logrado cumplir los objetivos. El sistema desarrollado a partir de software de código abierto opera de manera autónoma de forma local, es decir, sin depender de ningún servicio en la nube. Se ha logrado un bajo consumo energético mediante la elección del hardware y de la tecnología inalámbrica. Además, se ha implementado un control remoto para supervisar y gestionar el sistema desde ubicaciones externas de forma segura.

Personalmente, este proyecto ha sido una experiencia enriquecedora. Durante el desarrollo de este proyecto, he tenido la oportunidad de conocer tecnologías relacionadas con el Internet de las Cosas, plataformas domóticas y redes que en un principio no sabía que existían. Aunque no pude profundizar en cada tecnología o aspecto tanto como hubiera deseado debido a la amplitud del campo, me siento satisfecho con el resultado. Este trabajo me ha permitido adentrarme en el vasto y creciente mundo del IoT, y considero que ha sido un logro significativo en mi formación académica.

El proyecto ha integrado con éxito dispositivos IoT basados en la tecnología LoRaWAN de manera manual e individual, lo que ha proporcionado un control detallado y una comprensión profunda de cada dispositivo. Sin embargo, al contemplar un despliegue a gran escala con numerosos dispositivos, se abre una emocionante oportunidad para optimizar y acelerar este proceso. Durante el desarrollo del proyecto, noté que empresas innovadoras como Milesight publican en Github el código para decodificar los payloads de sus productos. Esto inspira la visión de una herramienta futura que pueda identificar automáticamente los dispositivos IoT basados en la tecnología LoRaWAN y agilizar su integración en el sistema.

El proyecto ha demostrado la capacidad de recolectar y generar datos de manera efectiva. Sin embargo, para maximizar el valor de estos datos, se presenta la oportunidad de explorar enfoques para su recolección y análisis externo. Esto podría implicar la conexión con bases de datos externas e, incluso, la integración con software de análisis avanzado que podrían facilitar la toma de decisiones más informadas basadas en los datos recolectados por el sistema.

7. BIBLIOGRAFÍA

[1] Asociación KNX, «¿Qué es KNX?,» Mayo 2021. [En línea]. Available: https://www.knx.es/es/documentacion/que-es-knx.html. [Último acceso: 2023].

[2] Intersoft Consulting Services AG, «Regulación General de Protección de Datos,» 29 Abril 2016. [En línea]. Available: https://gdpr-info.eu/ . [Último acceso: 10 Julio 2023].

[3] WiFi Alliance, «WiFi Alliance,» [En línea]. Available: https://www.wi-fi.org/. [Último acceso: 2023].

[4] «Open Systems Interconnection,» 15 Noviembre 1994. [En línea]. Available: https://www.iso.org/standard/20269.html. [Último acceso: 2023].

[5] Bluetooth SIG, «Bluetooth® Technology Website,» 3 Septiembre 1998. [En línea]. Available: https://www.bluetooth.com/about-us/. [Último acceso: Junio 2023].

[6] Bluetooth SIG, «Bluetooth Low Energy,» 15 Diciembre 2020. [En línea]. Available: https://www.bluetooth.com/specifications/specs/enhanced-chp-for-proximity-bonding-and-reconnection-with-ble-nwp/. [Último acceso: Julio 2023].

[7] Connectivity Standards Alliance, «About Connectivity Standards Alliance,» 14 Agosto 2019. [En línea]. Available: https://csa-iot.org/about/. [Último acceso: Julio 2023].

[8] J. Serra, «Cifrado AES: ¿Qué es y cómo funciona?,» 12 Octubre 2022. [En línea]. Available: https://ciberseguridad.com/guias/prevencion-proteccion/criptografia/cifrado-aes/. [Último acceso: Noviembre 2023].

[9] Connectivity Standards Alliance, «Zigbee Solutions,» 19 Agosto 2019. [En línea]. Available: https://csa-iot.org/all-solutions/zigbee/. [Último acceso: Julio 2023].

[10] Z-Wave, [En línea]. Available: https://www.z-wave.com/. [Último acceso: 09 Diciembre 2023].

[11] Z-Wave Alliance, «About Z-Wave Alliance,» [En línea]. Available: https://z-wavealliance.org/. [Último acceso: 09 Diciembre 2023].

[12] O. B. A. S. y. N. Sornin, «Patente LoRa (EP2763321 y US7791415),» 2013. [En línea]. Available: https://patents.google.com/patent/EP2763321A1/en. [Último acceso: 5 Diciembre 2023].

[13] Business Wire, «Semtch Acquires Wireless Long Range IP Provider Cycleo,» [En línea]. Available: https://www.businesswire.com/news/home/20120307006338/en/Semtech-Acquires-Wireless-Long-Range-IP-Provider-Cycleo. [Último acceso: 2023].

[14] Lora Alliance, «lora-alliance.org,» 29 Mayo 1995. [En línea]. Available: https://lora-alliance.org/. [Último acceso: Diciembre 2022].

[15] Lora Alliance, «lora-alliance.org,» 15 Diciembre 2020. [En línea]. Available: https://lora-alliance.org/resource_hub/lorawan-specification-v1-1/. [Último acceso: Diciembre 2022].

[16] Sigfox España, «Sigfox.es,» 27 Julio 2020. [En línea]. Available: https://www.sigfox.es/blogs/post/sigfox-suscripci%C3%B3n-cobertura-competidores. [Último acceso: 15 Mayo 2023].

[17] TST, «tst-sistemas.com,» 15 Octubre 2019. [En línea]. Available: https://tst-sistemas.com/soluciones/. [Último acceso: 7 Julio 2023].

[18] Arduino CL, 29 Mayo 1995. [En línea]. Available: https://arduino.cl/que-es-arduino/#:~:text=Arduino%20Naci%C3%B3%20en%20el%20a%C3%B1o,que%20fuera%20de%20bajo%20coste.. [Último acceso: 7 Julio 2023].

[19] Rasberry Pi Foundation, «rasberrypi.org,» [En línea]. Available: https://www.raspberrypi.org/about/ . [Último acceso: Julio 2023].

[20] Raspberry Pi User ®, «Raspberrypiusers.com,» [En línea]. Available: https://www.raspberrypiusers.com/modelos/#google_vignette. [Último acceso: Julio 2023].

[21] Domoticz, «About Domoticz,» 2012, [En línea]. Available: https://www.domoticz.com/wiki/About_Domoticz. [Último acceso: 2023].

[22] OpenHab Foundation, «About OpenHab Foundation,» [En línea]. Available: https://www.openhabfoundation.org/about/ . [Último acceso: 2023].

[23] OpenHab Foundation, «OpenHab,» [En línea]. Available: https://www.openhab.org/ . [Último acceso: 2023].

[24] «HomeBridge,» [En línea]. Available: https://homebridge.io/ . [Último acceso: 2023].

[25] M. andrade, «Members GladysAssistant,» [En línea]. Available: https://github.com/orgs/GladysAssistant/people. [Último acceso: 2023].

[26] GladyAssistant, «About GladyAssistant,» [En línea]. Available: https://gladysassistant.com/ . [Último acceso: 2023].

[27] «Home assistant,» Noviembre 2013. [En línea]. Available: https://www.home-assistant.io/. [Último acceso: Mayo 2023].

[28] Nabu Casa, «Acerca de nosotros: Nabu Casa,» [En línea]. Available: https://www.nabucasa.com/about/. [Último acceso: 2023].

[29] Nabu Casa Inc., «Github,» Noviembre 2013. [En línea]. Available: github.com/home-assistant. [Último acceso: Mayo 2023].

[30] Kafka, «Apache Kafka Oficial Web,» [En línea]. Available: https://kafka.apache.org/. [Último acceso: 13 Diciembre 2023].

[31] J. Rao, «Open Source LinkeInd Kafka,» 11 Enero 2011. [En línea]. Available: https://www.linkedin.com/blog/member/archive/open-source-linkedin-kafka. [Último acceso: 13 Diciembre 2023].

[32] The Internet Engineering Task Force, «The Constrained Application Protocol (CoAP),» [En línea]. Available: https://datatracker.ietf.org/doc/html/rfc7252. [Último acceso: 13 Diciembre 2023].

[33] OPC Foundation, «OPC Arquitectura Unificada,» [En línea]. Available: https://opcfoundation.org/about/opc-technologies/opc-ua/. [Último acceso: 13 Diciembre 2023].

[34] «MQTT web site,» [En línea]. Available: https://mqtt.org/faq/. [Último acceso: 2023].

[35] Eclipse Foundation, Inc., «Eclipse Foundation,» [En línea]. Available: https://www.eclipse.org/org/. [Último acceso: 2023].

[36] «Mosquitto web site,» [En línea]. Available: https://mosquitto.org/. [Último acceso: 2023].

[37] «About DuckDNS,» [En línea]. Available: https://www.duckdns.org/about.jsp?ref=upstract.com. [Último acceso: 2023].

[38] Vitalwrks Internet Solutions, LLC, «About Noip,» [En línea]. Available: https://www.noip.com/es-MX/about. [Último acceso: 2023].

[39] NGINX, «About NGINX,» [En línea]. Available: https://www.nginx.com/. [Último acceso: 2023].

[40] «The C10K Problem,» [En línea]. Available: http://www.kegel.com/c10k.html. [Último acceso: 2023].

[41] «About Project Honey Pot,» [En línea]. Available: https://www.projecthoneypot.org/about_us.php. [Último acceso: 2023].

[42] Cloudflare, «Cloudflare,» [En línea]. Available: https://www.cloudflare.com/es-es/learning/what-is-cloudflare/. [Último acceso: 2023].

[43] Tailscale Inc., «About Tailscale,» [En línea]. Available: https://tailscale.com/ . [Último acceso: 2023].

[44] A. Pennarun, D. Carney y B. Fitzpatrick, «Tailscale foundetion,» [En línea]. Available: https://archive.ph/20230223174332/https://techcrunch.com/2022/05/04/tailscale-

lands-100-million-to-transform-enterprise-vpns-with-mesh-technology/. [Último acceso: 2023].

[45] A. Leymenko, «github zerotier,» 2014. [En línea]. Available: https://github.com/zerotier/ZeroTierOne/releases/tag/0.9.2. [Último acceso: 2023].

[46] Zerotier, «About Zerotier,» 2015. [En línea]. Available: https://www.zerotier.com/about/. [Último acceso: 2023].

[47] Talescale Inc., «ZeroTier vs Tailscale,» [En línea]. Available: https://tailscale.com/compare/zerotier/. [Último acceso: 7 Diciembre 2023].

[48] OpenVPN, «About OpenVPN,» [En línea]. Available: https://openvpn.net/blog/openvpn-community-edition-vs-access-server/. [Último acceso: 7 Diciembre 2023].

[49] WireGuard, «WireGuard: fast, modern, secure VPN tunnel,» [En línea]. Available: https://www.wireguard.com/ . [Último acceso: 2023].

[50] Free Software Foundation, «Licencias - Proyecto GNU,» [En línea]. Available: https://www.gnu.org/licenses/. [Último acceso: 2023].

[51] «AppDaemon,» Junio 2020. [En línea]. Available: https://community.home-assistant.io/t/home-assistant-community-add-on-appdaemon-4/163259. [Último acceso: Diciembre 2023].

[52] «Comparación de Pyscript con AppDaemon,» [En línea]. Available: https://github.com/custom-components/pyscript/wiki/Comparing-Pyscript-to-AppDaemon. [Último acceso: Diciembre 2023].

[53] «Python Script integration Home Assistant,» [En línea]. Available: https://www.home-assistant.io/integrations/python_script/. [Último acceso: 2023].

[54] «The Oficial YAML web site,» [En línea]. Available: https://yaml.org/. [Último acceso: 2023].

[55] Node-RED, «About Node-RED,» [En línea]. Available: https://nodered.org/about/. [Último acceso: 2023].

[56] Milesight, «WS50x user guide,» 26 October 2022. [En línea]. Available: https://resource.milesight.com/milesight/iot/document/ws50x-user-guide-en.pdf. [Último acceso: Julio 2023].

[57] Geeksforgeeks, «Geekforgeeks.org,» 3 Julio 2023. [En línea]. Available: https://www.geeksforgeeks.org/who-invented-bluetooth/. [Último acceso: 10 Julio 2023].

[58] Circuit Digest Pvt. Ltd., «Electronicshub.org,» 15 Octubre 2019. [En línea]. Available: https://www.electronicshub.org/bluetooth-low-energy-ble-introduction-and-ble-vs-bluetooth-classic/. [Último acceso: Julio 2023].

[59] Leverege, «iotfotall.org,» 26 Junio 2020. [En línea]. Available: https://www.iotforall.com/what-is-lpwan/. [Último acceso: 15 Junio 2023].

[60] Gurtam, «flespi.com,» 26 Julio 2017. [En línea]. Available: https://flespi.com/blog/top-7-technologies-for-iot-connectivity-2017. [Último acceso: Julio 2023].

[61] semtech, «semtech.com,» [En línea]. Available: https://www.semtech.com/lora. [Último acceso: Enero 2023].

[62] Sigfox, «Sigfox.com,» 29 Mayo 1995. [En línea]. Available: https://www.sigfox.com/. [Último acceso: 15 Mayo 2023].

[63] 3GPP, «3gpp.org,» 29 Mayo 1995. [En línea]. Available: https://www.3gpp.org/ . [Último acceso: 7 Julio 2023].

[64] The Thread Group, «The Thread Group Site,» [En línea]. Available: https://www.threadgroup.org/. [Último acceso: Diciembre 2023].

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

ESTUDIO E IMPLEMENTACIÓN DE SISTEMAS INALÁMBRICOS SIN CONEXIÓN A INTERNET EN EDIFICIOS INTELIGENTES

2. PLANOS

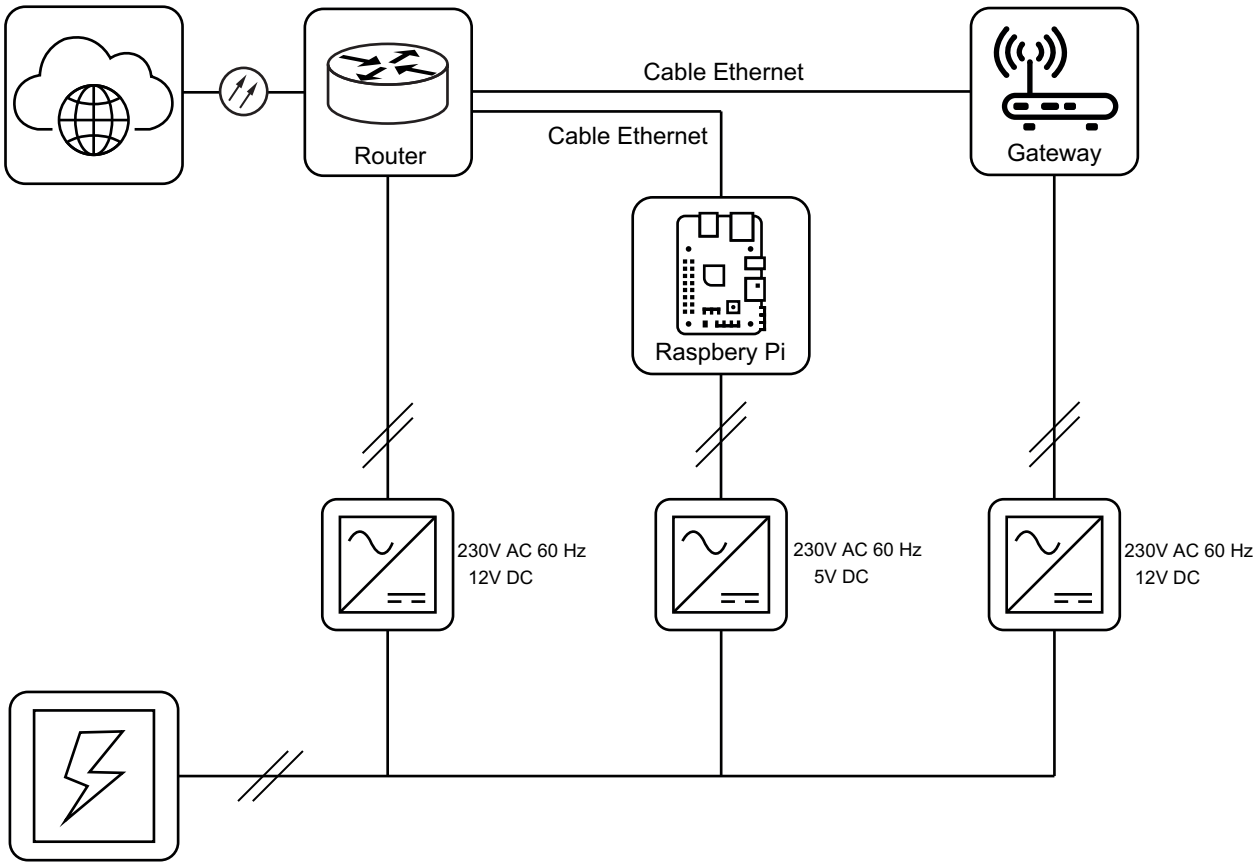
Autor:

D. Francisco Amorós Conejero

Tutor:

D. Ángel Francisco Perles Ivars

Valencia, abril de 2024



PROYECTO: Estudio e implementación de sistemas inalámbricos sin conexión a Internet en edificios inteligentes.

Fecha: 03/04/2024

Escala

No procede

Autor: Francisco Amorós Conejero

Revisor: Ángel Perles Ivars

Plano:

Diagrama de conexiones

Plano Nº

01

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

ESTUDIO E IMPLEMENTACIÓN DE SISTEMAS INALÁMBRICOS SIN CONEXIÓN A INTERNET EN EDIFICIOS INTELIGENTES

3. PLIEGO DE CONDICIONES

Autor:

D. Francisco Amorós Conejero

Tutor:

D. Ángel Francisco Perles Ivars

Valencia, abril de 2024

ÍNDICE DEL PROYECTO

1. DEFINICIÓN Y ALCANCE DEL PLIEGO

El presente pliego de condiciones especifica los requisitos generales y técnicos necesarios para la ejecución del proyecto. Asimismo, define las directrices que deben seguirse para la creación, replicación y utilización del sistema desarrollado, tal como se describe en el documento de la memoria.

Esta especificación se refiere a la implementación de un sistema inalámbrico sin conexión a Internet en edificios inteligentes. Se trata de un sistema que funciona en local, sin depender de servicios en la nube. Está compuesto por diversos dispositivos individuales enfocados en un bajo consumo y un computador central.

El sistema no requiere de un proceso de diseño o fabricación de hardware. Los componentes físicos y materiales que lo integran no son propiedad del autor. El carácter del proyecto se manifiesta en la configuración de estos componentes y el desarrollo software. Por lo tanto, las directrices que se presentan a continuación se centran en la instalación, uso y mantenimiento del mencionado conjunto.

2. CONDICIONES Y NORMAL DE CARÁCTER GENERAL

2.1. Instalación

El despliegue deberá realizarse por personal cualificada y familiarizada con los equipos. Esto se debe principalmente a que durante la instalación es necesario configurar los distintos dispositivos para que puedan funcionar correctamente en la red tal como se describe en el documento de la memoria. A continuación se detallan las condiciones específicas que deberán cumplirse con el fin de que el sistema funcione de forma adecuada.

La instalación del sistema central se realizará en un ambiente protegido y de fácil acceso con el fin de evitar el deterioro de los componentes electrónicos y favorecer la inspección visual de los mismos. Así mismo, para el Gateway LoRaWAN debe evitarse su colocación cerca de elementos que puedan generar una interferencia electromagnética en la banda de 868MHz. Además de comprobarse que no interfiera con otros dispositivos de radio. Una vez situados los elementos del sistema central se conectan los cables ethernet y los cables a la alimentación.

Los dispositivos (*end nodes*) se ubicarán en el rango de cobertura de la antena Gateway. Se realizarán las comprobaciones pertinentes que comprueben el estado de deterioro de la señal con el fin de garantizar la comunicación entre todos los elementos del sistema. Además, en el entorno de edificios patrimoniales se debe valorar el impacto visual de los dispositivos así como consultar con un experto en conservación.

Los dispositivos con sensores de temperatura y humedad no deben ubicarse en puntos próximos a fuentes de calor o frío, así como a fuentes de humedad que puedan alterar las mediciones o deteriorar los componentes electrónicos.

Los dispositivos de sensores de movimientos deben ubicarse en la posición que garantice la detección teniendo en cuenta el volumen de detección descrito por el fabricante.

en ubicaciones donde se desea detectar la presencia de personas. Tener en cuenta que en sensores DIR, aunque la sala este llena de gente, si no se mueven, el sensor detectará como si estuviera vacía. (no detecta movimiento)

Los interruptores inteligentes deben instalarse tal como se indica en el manual del fabricante. En caso de que se sustituya un interruptor normal por uno inteligente con neutro se deberá adaptar la instalación eléctrica mediante personal cualificado.

No se garantiza el correcto funcionamiento de este sistema empleando software, hardware o cualquier configuración diferente a los indicados en la memoria del proyecto. En caso de modificación, la responsabilidad del correcto funcionamiento del sistema recaerá sobre la persona u organismo que realice la modificación.

2.2. Utilización

Una vez desplegado el sistema y configurado los diferentes dispositivos y servicios software siguiendo las directrices detalladas en el documento de la memoria.

El usuario tendrá que crear unas credenciales de autenticación que le permitirán acceder a través de un navegador web al panel de control del servicio de Home Assistant. Este podrá ser editado según las necesidades del usuario final. El panel de control permite monitorizar y controlar los diferentes dispositivos, servicios y automatizaciones.

Para acceder de forma remota a los servicios del sistema, se utiliza la VPN de WireGuard. El usuario puede iniciar sesión en la interfaz web de WireGuard UI para añadir un nuevo dispositivo. Para vincular el dispositivo, se generará un código QR que puede ser escaneado utilizando la aplicación WireGuard Client en dispositivos móviles. Alternativamente, se puede descargar un archivo de configuración que puede ser importado al cliente WireGuard en un ordenador personal.

Para más detalles sobre el funcionamiento y utilización del equipo, póngase en contacto con el autor del proyecto.

2.3. Mantenimiento

El mantenimiento requerido por el sistema incluye el recambio o recarga de las baterías, la inspección visual periódica y la comprobación del correcto funcionamiento de los servicios software.

La inspección visual consiste en comprobar el correcto funcionamiento de los dispositivos, comprobar el correcto conexionado de los cables Ethernet y cables de alimentación y realizar la limpieza de polvo.

Las baterías de los dispositivos IoT requieren de un recambio o recarga periódica. El periodo varía en función del dispositivo, y depende de la frecuencia de mediciones que se puede determinar según las necesidades del usuario. Al final de su vida útil, las baterías no recargables deben ser desechadas de acuerdo con la normativa 2006/66/CE, que regula la gestión ambiental de pilas y acumuladores y sus residuos.

Por último, aunque es conveniente mantener el software actualizado para mejorar la seguridad y evitar posibles vulnerabilidades, al tratarse de un proyecto en fase de pruebas no se puede garantizar el correcto funcionamiento con los cambios de versiones. Esto se debe a que las nuevas versiones de software pueden interrumpir funcionalidades del sistema o la interoperabilidad entre los diferentes servicios de software. Por lo tanto, es esencial tener en cuenta este factor al gestionar las actualizaciones del sistema. Realizar copias de seguridad periódicas correspondientes para evitar la pérdida de información en caso de fallo del sistema.

3. CONDICIONES TÉCNICAS PARTICULARES

3.1. Especificaciones de los dispositivos IoT

Los dispositivos IoT utilizados en el sistema están basados en la tecnología inalámbrica LoRaWAN haciendo uso de la frecuencia de radio 868MHz. A continuación se detallán las especificaciones de cada dispositivo:

- Sensor de temperatura y humedad Dragino LHT52.
  - Resolución del sensor de temperatura: 0,01 °C, rango de operación: -20 ~ 50°C.
  - Resolución del sensor de humedad: 0.1 %RH, rango de operación: 0 ~ 99,0 %RH (sin condensación).
  - Soporta 2 pilas AAA
- Sensor de movimiento PIR y medición de brillo, Milesight WS202.
  - Resolución: 1 lux, rango: 1 a 60000 lux.
  - Alcance de detección ultrawide: 120° horizontal, 100° vertical, hasta 8 m de distancia.
  - Batería: 1 × 1650 mAh ER14335 Li-SOCL2
- Interruptor de pared inteligente. Milesight WS501, Milesight WS502 y Milesght WS523
  - configuración a través de NFC
  - Voltaje de entrada: 100~250 VAC, 50~60 Hz.
  - Corriente máxima: inductiva: 5 A por banda, total 10 A; resistiva: 10 A por banda, total 10 A.
  - Carga nominal máxima: 1000 W por banda, total 2000 W.
- Sensor de temperatura y humedad Milesight EM320-TH
  - Rango de medición: -30℃ a 60℃; 0% a 100% HR (Humedad Relativa).
  - Alta resolución: 0.1℃; 0.5% HR.
  - Precisión optimizada: ±0.2℃, ±2% HR.
  - Alimentación: 2 × 2700 mAh ER14505 Li-SOCL2 Baterías

3.2. Especificaciones de la plataforma domótica

3.2.1. Especificaciones de hardware

El sistema central está equipado con un hardware que consiste en una Raspberry Pi 4 y un Gateway LoRaWAN RAK7289 WisGate Edge Pro de RAK Wireless. La Raspberry Pi cuenta con una memoria RAM de 4GB basada en la tecnología SDRAM DDR3 y un procesador ARM Cortex que opera a una frecuencia de 1,5 GHz con 4 núcleos. Además, incluye puertos esenciales como USB 2.0 y 3.1, Ethernet, y ofrece conectividad WiFi y Bluetooth. La Raspberry Pi 4 tiene unas dimensiones de 10 x 7 x 3 cm y un peso de 50 gramos. Su consumo medio de energía es de 2,7 vatios.

En cuanto al almacenamiento, se utiliza una tarjeta SD High Endurance de 16 GB. La fuente de alimentación requerida es de 5V. Para la conexión entre la Raspberry Pi y el router, así como entre el router y el gateway se utiliza un cable Ethernet de 1 Gbps.

3.2.2. Especificaciones de software

El sistema implementado hace uso de una serie de servicios de software. A continuación, se proporciona una lista de estos servicios junto con las versiones específicas que se han empleado en el desarrollo del presente proyecto:

- Plataforma domótica: Home Assistant, versión del Core 2024.1.6
- Broker MQTT: MosquittoMQTT, versión 2.0.18
- Servidor VPN: WireGuard, versión 1.0.20210914
- Herramienta de programación visual: Node-RED, versión 3.1.1
- Herramienta de edición: Hass-Configuration, versión 0.5.2

4. PRUEBAS DE FUNCIONAMIENTO Y VALIDACIONES

Durante el proceso de implementación y configuración del sistema se debe llevar a cabo una serie de pruebas para garantizar el correcto funcionamiento.

Se comprobará el correcto funcionamiento de la comunicación entre los diferentes dispositivos y sistemas.

Se verifica la correcta comunicación entre los dispositivos (*End Nodes*) y el Gateway. Utilizando la interfaz del Gateway, se comprueba que el Gateway recibe y trasmite datos a los *End Nodes*. Para el Gateway RAK7289, se deben seguir los pasos detallados en el documento de la memoria para asegurar una configuración adecuada.

Se realiza una validación para asegurar que los payloads enviados por los End Nodes lleguen correctamente al bróker MQTT. Herramientas como MQTT Explorer u otras pueden utilizarse para monitorear y verificar esta comunicación.

Se verifica que los datos enviados desde los End Nodes lleguen sin problemas a Node-RED. Además, se confirma que Node-RED transmita correctamente los datos a Home Assistant. Herramientas como MQTT Explorer, la consola de registro del servicio de MosquittoMQTT y las herramientas de registro de Node-RED y Home Assistant se puede utilizarse para monitorear y verificar esta comunicación.

Se comprobará el correcto funcionamiento de los dispositivos según las especificaciones del fabricante.

Se evaluará los sensores de temperatura y humedades miden correctamente las magnitudes en su posición de trabajo dentro del rango de precisión proporcionado por el fabricante.

Se evaluará los sensores de movimiento detecte los movimientos en su posición de trabajo como se espera según las especificaciones del fabricante.

Se verificará la capacidad para registrar el consumo eléctrico de manera fiable de los sensores de medición de consumo eléctrico mediante una carga de potencia y consumo conocidas.

Se evaluará el comportamiento de las automatizaciones implementadas. Esto implica simular y probar diferentes situaciones para asegurar que las acciones automáticas se ejecuten según lo esperado. Las evaluaciones de las automatizaciones del sistema de prueba incluyen:

- **Encendido y Apagado Automático de la Iluminación:**

Se evalúa las situaciones: en la que el sensor de movimiento pasa de no detectado a detectado mientras el sensor de iluminación se mantiene en no iluminado: se enciende la iluminación. En la que el sensor de movimiento pasa de detectado a no detectado: después de 5 minutos se apaga la iluminación. En la que el sensor de movimiento se mantiene en detectado y el sensor de iluminación pasa de iluminado a no iluminado: se enciende la iluminación.

- **Encendido y Apagado Automático de un Deshumidificador**:

Se evalúa las siguientes situaciones: en la que el sensor de humedad supera el 55 %RH se activa el deshumificador. Y cuando el sensor de humedad baja al 35 %RH se desactiva el deshumificador.

- **Encendido y Apagado Automático de un Radiador**:

Se evalúa las siguientes situaciones: en la que el sensor de temperatura baja del valor indicado, se activa el radiador, y cuando supera dicho valor se desactiva. Además, también se comprobará cuando manteniendo el valor de temperatura del sensor constante, se cambia el valor indicado.

- **Encendido y Apagado Automático de la Ventilación**:

Se evalúa las siguientes situaciones: en la que el sensor de temperatura supera del valor indicado, se activa la ventilación, y cuando baja de dicho valor se desactiva. Además, también se comprobará cuando manteniendo el valor de temperatura del sensor constante, se cambia el valor indicado.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

ESTUDIO E IMPLEMENTACIÓN DE SISTEMAS INALÁMBRICOS SIN CONEXIÓN A INTERNET EN EDIFICIOS INTELIGENTES

4. PRESUPUESTO

Autor:

D. Francisco Amorós Conejero

Tutor:

D. Ángel Francisco Perles Ivars

Valencia, abril de 2024

ÍNDICE DEL PRESUPUESTO

ÍNDICE DE TABLAS

1. SOBRE EL PRESUPUESTO

El presupuesto contempla la mano de obra y la compra de todo el material necesario para la elaboración del proyecto. Para el cálculo del material, se han considerado los precios disponibles en las tiendas online de distribuidores autorizados y las cotizaciones solicitadas a los fabricantes de los productos originales. De esta forma, se ha determinado el precio de adquisición para el consumidor final de todos los equipos hasta marzo de 2024. Debido al cambiante ambiente económico, este presupuesto tiene una validez y garantía de un año.

Se ha incorporado un 13% adicional al presupuesto para cubrir cualquier gasto general que pueda surgir durante el desarrollo del proyecto. Además, se ha considerado un 6% como beneficio industrial. Ambos porcentajes contribuyen a aumentar el presupuesto de ejecución material del proyecto

El presupuesto debe presentarse junto con el documento de la memoria del proyecto y solo cubre lo que se detalla en dicho documento. Cualquier requisito adicional que no esté incluido en el documento puede conllevar costos adicionales que no están reflejados en este presupuesto.

2. MATERIALES Y COMPONENTES

Uds.	DENOMINACIÓN	CANTIDAD	PRECIO(€)	TOTAL(€)
ud	Raspbery Pi 4*	1	76,90	76,90
ud	Gateway RAK7268*	1	187,90	187,90
ud	Router Livebox*	1	17,00	17,00
ud	Cable Ethernet 1Gbps 1 metro	2	3,99	7,98
ud	Tarjeta SD High Endurance 16Gb	1	7,11	7,11
ud	Dragino LTH52	1	25,99	25,99
ud	Milesight WS202	1	57,03	57,03
ud	Milesight WS501	1	53,26	53,26
ud	Milesight WS502	1	88,81	88,81
ud	Milesight WS523	1	74,60	74,60
ud	Milesight EM320-TH	1	63,94	64,94
ud	Pila AAA	4	0,50	2,00
ud	Pila ER14335 (AA 2/3)	1	1,75	1,75
	SUBTOTAL MATERIALES Y COMPONENTES			665,27

Tabla 1. Costes de materiales y componentes.

El coste total de los materiales asciende a 665,27€.

*Cable y fuente de alimentación incluido en el precio.

3. MANO DE OBRA

El coste de la mano de obra se ha calculado teniendo en cuenta la dedicación estimada obtenida conforme a la carga de 12 ECTS (European Credit Transfer and Accumulation System). Dado que cada crédito corresponde a 25h, la dedicación total estimada es de 300 horas de trabajo. Se ha estimado un salario de 25 €/h.

Uds.	DENOMINACIÓN	CANTIDAD	PRECIO(€)	TOTAL(€)
h	Investigación, diseño y desarrollo	300	25,00	7.500,00
	GRADUADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA			
	SUBTOTAL MANO DE OBRA			7.500,00

Tabla 2. Coste mano de obra.

El precio en concepto de mano de obra asciende a 7.500,00€.

4. COSTE DEL PROYECTO

COSTE DEL PROYECTO

Materiales y componentes	665,27 €
Mano de obra	7.500,00 €
TOTAL	8.165,27 €
13% de gastos generales	1.061,49 €
6% de beneficio industrial	489,92 €
SUMA	9.716,68 €
21% IVA	2.040,50 €
COSTE TOTAL DEL PROYECTO	11.757,18 €

Tabla 3. Coste total del proyecto.

La elaboración del proyecto tiene un coste total asociado de 11.757,18€.- ONCE MIL SETECIENTOS CINCUENTA Y SIETE EUROS CON DIECIOCHO CÉNTIMOS.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

ESTUDIO E IMPLEMENTACIÓN DE SISTEMAS INALÁMBRICOS SIN CONEXIÓN A INTERNET EN EDIFICIOS INTELIGENTES

5. ANEXOS

Autor:

D. Francisco Amorós Conejero

Tutor:

D. Ángel Francisco Perles Ivars

Valencia, abril de 2024

ÍNDICE DE CONTENIDO

ANEXO I. Archivo docker-compuse.yml

```
version: '3'
services:
  homeassistant:
    image: homeassistant/home-assistant:stable
    volumes:
      - /Docker/home-assistant:/config
    ports:
      - "8123:8123"
    restart: always
  mosquitto:
    image: eclipse-mosquitto:latest
    volumes:
      -
/Docker/mosquitto/config/mosquitto.conf:/mosquitto/config/mosquitto.conf
      - /Docker/mosquitto/data:/mosquitto/data
      - /Docker/mosquitto/log:/mosquitto/log
    ports:
      - "1883:1883"
      - "9001:9001"
    restart: always
  node-red:
    image: nodered/node-red:latest
    volumes:
      - /Docker/node-red:/data
    ports:
      - "1880:1880"
    restart: always
hass-configurator:
    image: causticlab/hass-configurator-docker:latest
    volumes:
      - /Docker/home-assistant:/config
    ports:
      - "3218:3218"
    restart: always
wireguard:
    image:  linuxserver/wireguard:v1.0.20210914-ls7
    container_name: wireguard
    cap_add:
      - NET_ADMIN
    volumes:
      - /Docker/wireguard/config:/config

    ports:
      - "5000:5000"
      - "51820:51820/udp"

  wireguard-ui:
    image: ngoduykhanh/wireguard-ui:latest
    container_name: wireguard-ui
    depends_on:
      - wireguard
```

```
51      cap_add:
52        - NET_ADMIN
53      network_mode: service:wireguard
54      environment:
55        - SENDGRID_API_KEY
56        - EMAIL_FROM_ADDRESS
57        - EMAIL_FROM_NAME
58        - SESSION_SECRET
59        - WGUI_USERNAME=admin
60        - WGUI_PASSWORD=password
61        - WG_CONF_TEMPLATE
62        - WGUI_MANAGE_START=true
63        - WGUI_MANAGE_RESTART=true
64      logging:
65        driver: json-file
66        options:
67          max-size: 50m
68      volumes:
69        - /Docker/wireguard/db:/app/db
70        - /Docker/wireguard/config:/etc/wireguard
```

ANEXO II. Archivo config/configuration.yaml:

```yaml
# Loads default set of integrations. Do not remove.
default_config:

# Load frontend themes from the themes folder
frontend:
  themes: !include_dir_merge_named themes

# Text-to-speech
tts:
  - platform: google_translate

automation: !include automations.yaml
script: !include scripts.yaml
scene: !include scenes.yaml

mqtt:
  sensor:
  #Sensor de Temperatura y Humedad
    - name: "temperature"
      unique_id: "temp"
      state_topic: "edificio_historico/sala1/sensores/temperatura/state"
      value_template: "{{ value_json.temperature }}"
      device_class: temperature

      device:
          identifiers: "temp"

    - name: "humidity"
      unique_id: "humd"
      state_topic: "edificio_historico/sala1/sensores/temperatura/state"
      value_template: "{{ value_json.humidity }}"
      device_class: humidity

      device:
          identifiers: "temp"

    - name: "temp&Hum_batteryVoltage"
      unique_id: "temp_batV"
      state_topic: "edificio_historico/sala1/sensores/temperatura/state"
      unit_of_measurement: "mV"
      value_template: "{{ value_json.batteryVoltage }}"
      device_class: voltage

      device:
          identifiers: "temp"

    - name: "temp&Hum_battery"
      unique_id: "temp_bat"
      state_topic: "edificio_historico/sala1/sensores/temperatura/state"
      unit_of_measurement: "%"
      value_template: "{{ value_json.batteryPercentage }}"
```

```
52        device_class: battery
53
54        device:
55            name: "Sensor de Temperatura"
56            identifiers: "temp"
57            manufacturer: "Dragino"
58            model: "LHT52"
59            sw_version: "1.1.0"
60            configuration_url: "http://wiki.dragino.com/xwiki/bin/view/Main/User%2
61  %20LoRaWAN%20Temperature%20%26%20Humidity%20Sensor%20User%20Manual/#H2.2A0HowtoA
62
63
64    #Sensor de Movimiento: valor batería
65
66      - name: "movimiento_battery"
67        unique_id: "move_bat"
68        state_topic: "edificio_historico/sala1/sensores/movimiento/state"
69        unit_of_measurement: "%"
70        value_template: "{{ value_json.battery }}"
71        device_class: battery
72
73    #Definición de dispositivo: Sensor de Movimiento
74        device:
75            name: "Sensor de Movimiento"
76            identifiers: "move"
77            manufacturer: "Milesight"
78            model: "WS202-868M"
79            sw_version: "1.3"
80            hw_version: "1.1"
81            configuration_url: "https://www.milesight-
82  iot.com/lorawan/sensor/ws202/?utm_term=ws202%20milesight&utm_campaign=WS+Series+
83  d-2065465341046&hsa_kw=ws202%20milesight&hsa_mt=b&hsa_net=adwords&hsa_ver=3&gad=
84
85    #Medidor de consumo
86
87      - name: "electrical_voltage"
88        unique_id: "elec_voltage"
89        state_topic: "edificio_historico/sala1/sensores/power/state"
90        unit_of_measurement: "V"
91        value_template: "{{ value_json.voltage }}"
92        device_class: voltage
93
94        device:
95            identifiers: "power_sum"
96
97      - name: "electrical_current"
98        unique_id: "elec_current"
99        state_topic: "edificio_historico/sala1/sensores/power/state"
100       unit_of_measurement: "mA"
101       value_template: "{{ value_json.current }}"
102       device_class: current
103
104       device:
105           identifiers: "power_sum"
106
107     - name: "electrical_power"
```

```
108       unique_id: "elec_power"
109       state_topic: "edificio_historico/sala1/sensores/power/state"
110       unit_of_measurement: "W"
111       value_template: "{{ value_json.power }}"
112       device_class: power
113
114       device:
115           identifiers: "power_sum"
116
117     - name: "electrical_power_sum"
118       unique_id: "elec_power_sum"
119       state_topic: "edificio_historico/sala1/sensores/power/state"
120       unit_of_measurement: "Wh"
121       value_template: "{{ value_json.power_sum }}"
122       state_class: total
123
124       device:
125           identifiers: "power_sum"
126
127     - name: "electrical_factor"
128       unique_id: "elec_factor"
129       state_topic: "edificio_historico/sala1/sensores/power/state"
130       unit_of_measurement: "%"
131       value_template: "{{ value_json.factor }}"
132       device_class: power_factor
133
134       device:
135           identifiers: "power_sum"
136
137     #igual es mejor que sea un binary sensor;
138     - name: "electrical_outage"
139       unique_id: "elec_outage"
140       state_topic: "edificio_historico/sala1/sensores/power/state"
141       value_template: "{{ value_json.outage }}"
142
143       device:
144           name: "Medidor de Consumo"
145           identifiers: "power_sum"
146           manufacturer: "Milesight"
147           model: "WS522-868M"
148           sw_version: "1.1"
149           hw_version: "1.0"
150           configuration_url: "https://www.milesight-iot.com/lorawan/socket/ws52x
151
152
153   #Sensor de Temperatura Milesight
154     - name: "temperature_milesight"
155       unique_id: "temp_milesight"
156       state_topic: "edificio_historico/sala2/sensores/temperatura/state"
157       value_template: "{{ value_json.temperature }}"
158       device_class: temperature
159
160       device:
```

```
161             identifiers: "temp_milesight"
162
163       - name: "humidity_milesight"
164         unique_id: "humd_milesight"
165         state_topic: "edificio_historico/sala2/sensores/temperatura/state"
166         unit_of_measurement: "%"
167         value_template: "{{ value_json.humidity }}"
168         device_class: humidity
169
170         device:
171             identifiers: "temp_milesight"
172
173       - name: "temp&Hum_battery_milesight"
174         unique_id: "bat_milesight"
175         state_topic: "edificio_historico/sala2/sensores/temperatura/state"
176         unit_of_measurement: "%"
177         value_template: "{{ value_json.battery }}"
178         device_class: battery
179
180         device:
181             name: "Sensor de Temperatura Milesight"
182             identifiers: "temp_milesight"
183             manufacturer: "Milesight"
184             model: "EM320-TH-868M"
185             sw_version: "1.3"
186             hw_version: "1.1"
187             configuration_url: "https://www.milesight-iot.com/lorawan/sensor/em320
188
189   #Interruptor Luz
190       - name: "Office light Current"
191         unique_id: "light_current"
192         state_topic: "edificio_historico/sala1/actuador/luz/data"
193         unit_of_measurement: "mA"
194         value_template: "{{ value_json.current }}"
195         device_class: current
196
197         device:
198           identifiers: "light"
199
200       - name: "Office light voltage"
201         unique_id: "light_voltage"
202         state_topic: "edificio_historico/sala1/actuador/luz/data"
203         unit_of_measurement: "V"
204         value_template: "{{ value_json.voltage }}"
205         device_class: voltage
206
207         device:
208           identifiers: "light"
209
210       - name: "Office light Power Consumption"
211         unique_id: "light_power_sum"
212         state_topic: "edificio_historico/sala1/actuador/luz/data"
213         unit_of_measurement: "Wh"
214         value_template: "{{ value_json.power_sum }}"
215         state_class: total
216
```

```
217       device:
218         identifiers: "light"
219
220     - name: "Office light Active Power"
221       unique_id: "light_power"
222       state_topic: "edificio_historico/sala1/actuador/luz/data"
223       unit_of_measurement: "W"
224       value_template: "{{ value_json.power }}"
225       device_class: power
226
227       device:
228         identifiers: "light"
229
230 #Sensor de movimiento e iluminacion
231   binary_sensor:
232     - name: "movimiento"
233       unique_id: "move"
234       state_topic: "edificio_historico/sala1/sensores/movimiento/state"
235       value_template: "{{ value_json.move }}"
236       device_class: occupancy
237       payload_off: "Vacant"
238       payload_on: "Occupied"
239
240       device:
241           identifiers: "move"
242
243     - name: "iluminacion"
244       unique_id: "bright"
245       state_topic: "edificio_historico/sala1/sensores/movimiento/state"
246       value_template: "{{ value_json.iluminacion }}"
247       device_class: light
248       payload_off: "Dark"
249       payload_on: "Bright"
250
251       device:
252           identifiers: "move"
253
254
255   #Interruptor de iluminación
256   light:
257     - name: "Office light"
258       unique_id: "light"
259       state_topic: "edificio_historico/sala1/actuador/luz/status"
260       command_topic: "edificio_historico/sala1/actuador/luz/switch"
261       qos: 0
262       payload_on: "ON"
263       payload_off: "OFF"
264
265       device:
266           name: "Interruptor Iluminación"
267           identifiers: "light"
268           manufacturer: "Milesight"
269           model: "WS501-868M"
```

```
270           sw_version: "1.1"
271           hw_version: "1.0"
272           configuration_url: "https://www.milesight-iot.com/lorawan/switch/ws50x
273
274
275     #Interruptor doble ¿ventilación?
276   switch:
277     - unique_id: switch_1
278       name: "Switch_1"
279       state_topic: "edificio_historico/sala1/actuador/ventilacion"
280       command_topic: "edificio_historico/sala1/actuador/ventilacion/set"
281       payload_on: "ON"
282       payload_off: "OFF"
283
284       device:
285        identifiers: "switch"
286
287     - unique_id: switch_2
288       name: "Switch_2"
289       state_topic: "edificio_historico/sala1/actuador/calefaccion"
290       command_topic: "edificio_historico/sala1/actuador/calefaccion/set"
291       payload_on: "ON"
292       payload_off: "OFF"
293
294       device:
295         name: "Interruptor doble"
296         identifiers: "switch"
297         manufacturer: "Milesight"
298         model: "WS501-868M"
299         sw_version: "1.1"
300         hw_version: "1.1"
301         configuration_url: "https://www.milesight-iot.com/lorawan/switch/ws50x/"
302
303   #Medidor de Consumo
304     - unique_id: "electrical_switch"
305       name: "electrical_switch"
306       state_topic: "edificio_historico/sala1/sensores/power/state"
307       command_topic: "edificio_historico/sala1/sensores/power/switch"
308       payload_on: "open"
309       payload_off: "close"
310
311       device:
312           identifiers: "power_sum"
313
314   #entidades adicionales:
315   select:
316       - command_topic: "edificio_historico/automatizacion/modo"
317         name: "Test Select"
318         unique_id: "modo"
319         options:
320           - "Automatico"
321           - "Manual"
322
323         device:
324             name: "Selector Modo"
325             identifiers: "modo"
```

ANEXO III. Código usado en Node-RED.

Bloque Base64Hex

```
// Cadena codificada en Base64
const data_encoded = msg.payload.data;

// Decodificar la cadena Base64
const data_decoded = Buffer.from(data_encoded, 'base64');

msg.payload = data_decoded;

//envia el mensaje
return msg;
```

Interpletación de datos LTH52 Dragino. Para fPort = 2:

```
const data_decoded = msg.payload;

// Interpretar los bytes para obtener los valores de temperatura y humedad
const temperature = (data_decoded[0] << 24 >> 16 | data_decoded[1]) / 100.0;
const humidity = (data_decoded[2] << 8 | data_decoded[3]) / 10.0;
const temperature_ext = (data_decoded[4] << 24 >> 16 | data_decoded[5]) / 100.0;

const timeStamp = (data_decoded[7] << 24 | data_decoded[8] << 16 | data_decoded[9] << 8 | data_decoded[10]);

//Guarda la información en el mensaje
msg.payload = {
    "temperature":temperature,
    "humidity":humidity,
    "External temperature":temperature_ext,
    "Extension":data_decoded[6],
    "TimeStamp":timeStamp
};


//envia el mensaje
return msg;
```

Interpletación de datos LTH52 Dragino. Para fPort = 5:

```
const data_decoded = msg.payload;

// Obtener los bytes correspondientes a cada campo
const sensorModel = data_decoded[0];
const firmwareBytes = data_decoded.slice(1, 3);
const frequencyBand = data_decoded[3];
const subBand = data_decoded[4];
const batteryVoltage = data_decoded.slice(5, 6);

// Interpretar los bytes para obtener los valores específicos
const firmwareVersion = `v${firmwareBytes[0]}.${firmwareBytes[1] >> 4}.${firmwareBytes[1] & 0x0F}`;
const batteryVoltageValue = (batteryVoltage[0] << 8 | batteryVoltage[1]);
const batteryPercentage = (batteryVoltageValue / 3072) * 100; // Suponiendo que la batería es de 3072 mV

// Guardar la información en el mensaje
msg.payload = {
    "sensorModel": sensorModel,
    "firmwareVersion": firmwareVersion,
    "frequencyBand": frequencyBand,
    "subBand": subBand,
    "batteryVoltage": batteryVoltageValue,
    "batteryPercentage": batteryPercentage
};

// Enviar el mensaje
return msg;
```

Interpretación de datos sensor de movimiento WS202 Milesight

```
//variables
var move;
var bright;
var battery;

if (msg.payload[5] == 0x0) {
    move = "Vacant";
} else {
    move = "Occupied";
}

if (msg.payload[8] == 0x0) {
    bright = "Dark";
} else {
    bright = "Bright";
}

battery = msg.payload[2];

msg.payload = {
    "move":move,
    "iluminacion":bright,
    "battery":battery
}

return msg;
```

ANEXO IV: RELACIÓN DEL TRABAJO CON LOS OBJETIVOS DE DESARROLLO SOSTENIBLE DE LA AGENDA 2030

Objetivos de Desarrollo Sostenibles		Alto	Medio	Bajo	No procede
ODS 1.	**Fin de la pobreza**				X
ODS 2.	**Hambre cero**				X
ODS 3.	**Salud y bienestar**		X		
ODS 4.	**Educación de calidad**			X	
ODS 5.	**Igualdad de género**				X
ODS 6.	**Agua limpia y saneamiento**	X			
ODS 7.	**Energía asequible y no contaminante**	X			
ODS 8.	**Trabajo decente y crecimiento económico**			X	
ODS 9.	**Industria, innovación e infraestructuras**		X		
ODS 10.	**Reducción de las desigualdades**				X
ODS 11.	**Ciudades y comunidades sostenibles**	X			
ODS 12.	**Producción y consumo responsables**	X			
ODS 13.	**Acción por el clima**		X		
ODS 14.	**Vida submarina**				X
ODS 15.	**Vida de ecosistemas terrestres**				X
ODS 16.	**Paz, justicia e instituciones sólidas**			X	
ODS 17.	**Alianzas para lograr objetivos**				X

Tabla 1. Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Descripción de la alineación del TFG con los ODS con un grado de relación más alto.

Al ser un trabajo que consiste en un desarrollo de software para la implementación de sistemas inalámbricos en edificios inteligentes, este proyecto contribuye de manera significativa a la sostenibilidad y eficiencia. Debido a su potencial para mejorar la eficiencia energética y fomentar un consumo eficiente de recursos como el agua. Además, su aplicación puede generar impactos indirectos en diferentes áreas como la salud y el bienestar, la educación, la privacidad y la independencia de servicios de terceros, entre otros.