UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf
Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Uso de MQTT para el control de dispositivos de IoT

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Lliso Cosin, Alejandro

Tutor: Manzoni, Pietro

Curso: 2020 - 2021

Resumen

A lo largo de este proyecto se estudiará la comunicación de distintos dispositivos mediante Wifi utilizando el protocolo MQTT para poder diferenciar entre distintos dispositivos. Para poder facilitar la visualización de los datos obtenidos también se va a desarrollar una pequeña aplicación web que será capaz de recibir los datos de los dispositivos externos y representarlos de forma intuitiva. Como cliente MQTT se utilizará una placa Lopy, la cual tiene sensores variados para obtener información del ambiente. En la parte del servidor se utilizarán aplicaciones como Home Assistant o Grafana para la representación de datos. Cabe destacar que todo el desarrollo se realiza teniendo en cuenta la privacidad del usuario y la mantenibilidad del código desarrollado.

Palabras clave: IoT, MQTT, apps, Lopy, Home Assistant

Abstract

Throughout this project we will study the communication of different devices through Wifi using the MQTT protocol in order to differentiate between each device. In order to allow the visualization of the obtained data, a small web application will be developed to receive the data from the external devices and represent them in an intuitive way. As MQTT client a Lopy board will be used, which has various sensors to obtain information from the environment. On the server side, applications such as Home Assistant or Grafana will be used for data representation. It should be noted that all the development is done considering the user's privacy and the maintainability of the developed code.

Keywords : IoT, MQTT, apps, Lopy, Home Assistant

Tabla de contenido

Tabla de ilustraciones

Tabla de tablas

1. Introducción

Vivimos en una época en la que cada vez estamos más conectados, y cada vez los dispositivos son más costosos y existe la posibilidad de que en países con menos recursos estos dispositivos no sean tan accesibles y no puedan facilitarles también la vida.

Además de lo anterior también hay zonas del planeta en la que la conectividad, el acceso a electricidad y otros recursos es realmente complicado.

Por otro lado, existen países en que desechamos muchos dispositivos por no ser la última tecnología, siendo dispositivos completamente válidos y que podrían ser aprovechados en proyectos que requieran poca capacidad de cómputo en lugares del planeta con menos recursos.

Es indiscutible que la falta de recursos en algunos lugares del planeta es un gran problema y que tiene muchas posibles soluciones. Aunque no se pueda eliminar la desigualdad por completo, podemos reducir parte de la brecha tecnológica reutilizando los dispositivos desechados en otras partes del planeta.

1.1. Motivación

Por todo lo anterior la principal motivación de este proyecto es poder conectar dispositivos a la red con los mínimos recursos posibles, esto ayudaría a ciertas zonas a mejorar su calidad de vida, como por ejemplo automatizando cultivos. El poder reutilizar los dispositivos también nos ayudaría a reducir la contaminación por desechos electrónicos en un gran porcentaje.

Otra gran motivación es la de cada vez poder conectar dispositivos más pequeños ya que al poderse conectar con menos potencia se puede reducir el tamaño de los componentes necesarios para conectarlo.

Otra motivación es la necesidad de tener un sistema IOT seguro el cual no se conecte necesariamente a internet para enviar y recibir cualquier tipo de mensaje, pudiendo ser este un gran problema de privacidad.

1.2. Objetivos

Con este proyecto intentamos cumplir cuatro objetivos principales:

- Poder recibir información a un servidor central desde cualquier tipo de dispositivo que tenga la posibilidad de conectarse.
- Poder enviar información desde un servidor central a cualquier dispositivo conectado a este.
- Poder conectar los dispositivos sin necesitar la conexión a internet de todos los dispositivos de la malla.

- Como último objetivo tenemos que poder monitorizar y analizar esta información de forma práctica e intuitiva para facilitar el uso por parte del usuario.

2. Estado de la cuestión

En esta sección hablaremos en primer lugar de una visión general del software utilizado en IoT, y después de haber analizado los distintos tipos de ecosistemas software pasaremos a analizar los distintos tipos de hardware que pueden ser utilizados para montar un sistema IoT utilizando un protocolo publicador-suscriptor.

2.1. Software

En el primero de estos apartados vamos a discutir sobre el software actual relacionado con el IoT. En primer lugar, de una visión general del IoT en nuestras vidas, en segundo lugar, describiremos los principales ecosistemas comerciales de IoT, después pasaremos a ver los protocolos IoT más utilizados para conectar dispositivos y por último veremos distintas aplicaciones para la representación de datos.

2.1.1. Visión general

Hoy en día las tecnologías IoT están muy extendidas incluso en dispositivos que no imaginamos. Esto es debido a la aparición de tecnologías inalámbricas con buena relación de velocidad y consumo, también teniendo en cuenta el tamaño de los chips que son capaces de incorporar estas tecnologías.

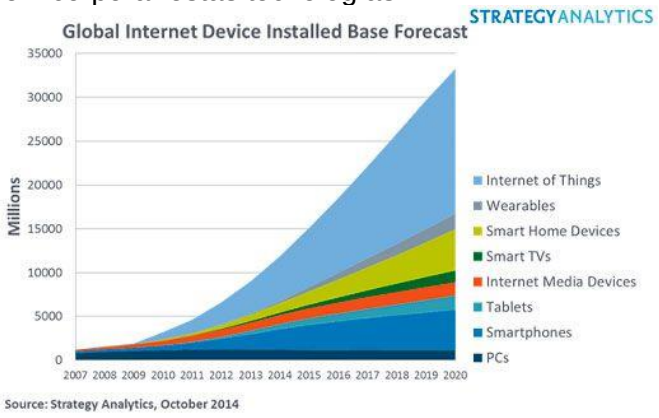

Ilustración 1. Evolución dispositivos IoT

En la gráfica anterior podemos ver distintos tipos de dispositivos IoT y que tienen todo tipo de usos, en ella no podemos ver el uso de estos dispositivos en la industria pero es una muestra bastante visual de que el uso de este tipo de dispositivos está extendido en todo tipo de ámbitos.

2.1.2. Ecosistemas IOT

Existe un amplio conjunto de soluciones de IOT en el ámbito doméstico, casi tantas como marcas distintas, que normalmente no suelen ser compatibles entre ellas. Vamos a proceder a mencionar las más destacadas:

- **Amazon Alexa:** El ecosistema de Amazon destaca principalmente por sus altavoces inteligentes, los cuales son capaces de enviar y recibir órdenes de los dispositivos compatibles con Alexa. Posiblemente el punto más destacado de este asistente es su amplia gama de aplicaciones que son conocidas como Skills y que incluyen funcionalidades como poder leer titulares de periódicos o

seleccionar canciones de Spotify con nuestra voz. Por último, ya solo hay que mencionar que los únicos dispositivos compatibles con este ecosistema son los que incluyen la etiqueta Works with Amazon Alexa y estos suelen funcionar normalmente mediante WIFI.

- **Google Home:** Este ecosistema, tiene su gran ventaja en que todos los dispositivos Android vienen con el asistente de Google preinstalado y actualmente es el que más dispositivos compatibles tiene en su arsenal, incluyendo dispositivos de grandes marcas como Ikea, Philips o Xiaomi. Posiblemente su gran desventaja es la falta de personalización en comparación a su gran contrincante Amazon Alexa. Por último, solo me queda mencionar que la mayoría de los dispositivos disponibles en este ecosistema también utilizan conexión mediante WIFI.

- **Apple HomeKit:** Esta es la propuesta por el otro gigante tecnológico, en este caso parece que es el menos competitivo, ya que cuenta con las mismas ventajas que los anteriores, pero tiene muchos menos dispositivos compatibles y en muchas ocasiones ofrece problemas a la hora de emparejar dispositivos. Otro de los motivos por los cuales está perdiendo esta batalla es el alto precio de los dispositivos en comparación a sus competidores. También cabe mencionar que el gran conjunto de estos dispositivos se conecta mediante WIFI.

- **Xiaomi:** Este ya no es tan conocido en el sector de la domótica ya que esta empresa China aún se está iniciando en este, pero, posiblemente es la firma con más dispositivos distintos y que en un futuro pueda tener la mayor cantidad de dispositivos inteligentes. Otra ventaja de este ecosistema es la posibilidad de utilizar dispositivos Xiaomi con los ecosistemas de Amazon o Google, haciendo posible el tener dispositivos de esta firma en otros ecosistemas, incluso utilizarlos como puentes entre distintos ecosistemas. Una novedad frente a las propuestas ya mencionadas es que esta compañía ya incluye otro tipo de conexión, que en este caso se trata de ZigBee.

- **Samsung SmartThings:** Esta compañía ya nos trae una novedad respecto al resto, necesita un hub para conectar los distintos dispositivos. Esto trae una serie de ventajas como que incluye conexiones de tres tipos como WIFI, ZigBee y Z-Wave. Otra gran ventaja es que le quitamos saturación a nuestro propio rúter. Por otra parte, trae alguna desventaja como la necesidad de tener amplificadores de este hub si el rango de este no fuera suficiente y también añadimos un cuello de botella en este elemento.

2.1.3. Protocolos de comunicación usados en IOT

En el ámbito de los protocolos de comunicación solo tenemos dos grandes competidores que cumplan con unos requisitos mínimos que sean ligeros y nos generen suficiente confianza para el uso en dispositivos inteligentes, que son los siguientes:

- **MQTT:** El protocolo MQTT es un protocolo de publicación-suscripción que nos va a facilitar poder conectar una gran cantidad de dispositivos. Este protocolo también tiene cierta facilidad para comprobar que el mensaje llega a su objetivo y nos proporciona seguridad mediante certificados electrónicos.
- **COAP:** El segundo es COAP, un protocolo cliente-servidor que usualmente utiliza el sistema de comunicación publicador-suscriptor, pero también es capaz de enviar mensajes de un cliente a otro sin pasar por el servidor.

2.1.4. Representación de datos

En esta sección vamos a ver distintas herramientas para el análisis y monitorización de datos:

- Grafana[1]: Es un software libre que nos ofrece la posibilidad de crear un cuadro de instrumentos y gráficos a partir de datos métricos, nos ofrece un sistema de visualización completamente personalizado.
- Kibana[2]: Kibana es un software gratuito, que obtiene más características si tenemos la licencia de Elastic. Kibana está escrito en JavaScript, por lo que puede ser utilizado en cualquier tipo de plataforma. Kibana también ofrece la posibilidad de examinar textos y posiciones geográficas.

2.2. Hardware

En este apartado analizaremos los distintos tipos de hardware que podrían ser útiles para montar cada una de las partes necesarias para hacer funcionar el protocolo MQTT, tanto el broker como los suscriptores.

2.2.1. Bróker

Debido al poco consumo del protocolo MQTT, no necesitaremos nada muy especial para hacer de bróker de este protocolo. Necesitaremos cualquier tipo de ordenador con acceso a la misma red que tienen los suscriptores por lo que, aunque existen varias posibilidades solo vamos a mencionar la RaspBerry Pi[3]. De esta solo vamos a mencionar sus principales características y su precio:

- **Microcontrolador:** Broadcom BCM2711
- **CPU:** Procesador de cuatro núcleos a 1,5 GHz con brazo Cortex-A72
- **GPU:** VideoCore VI

---

[1] *Grafana: The open observability platform*. (2014). Grafana Labs. https://grafana.com/

[2] *Kibana: Explora, visualiza y descubre datos*. (2019). Elastic. https://www.elastic.co/es/kibana/

[3] Pi, R. (2021, 3 September). *Teach, Learn, and Make with*. Raspberry Pi. https://www.raspberrypi.org/

- **Memoria:** 4GB LPDDR4 RAM
- **Conectividad:** 802.11ac Wi-Fi / Bluetooth 5.0, Gigabit Ethernet
- **Alimentación:** 5V/3A vía USB-C, 5V vía cabezal GPIO
- **Precio:** Alrededor de los 60€

2.2.2. Suscriptores

Pese a que los suscriptores pueden ser cualquier dispositivo que soporte el protocolo MQTT, solo vamos a hablar de hardware que sea completamente configurable y que disponga de características como el bajo consumo, conectividad WIFI y algún tipo de sensor para controlar parámetros como la temperatura o la humedad. Vamos a hablar sobre dos grandes marcas de dispositivos que cumplirían con estas características.

La primera de las empresas es Arduino [4], y de esta vamos a mencionar las características de su producto insignia, el Arduino uno. Algo que tenemos que destacar de este es que no dispone de WIFI integrado y que necesitaremos un módulo externo para dotar al dispositivo de WIFI. Características principales:

- **Microcontrolador:** ATmega328 (5V)
- **Alimentación recomendada:** 7-12V
- **Pines digitales I/O:** 14
- **Pines PWM:** 6
- **Entradas analógicas:** 6
- **Corriente máxima por pin:** 40 mA
- **Memoria Flash:** 32 KB
- **SRAM:** 2 KB
- **EEPROM:** 1 KB
- **Velocidad:**16 MHz
- **Precio:** Alrededor de 20€

La segunda empresa es Pycom[5], que se diferencia con la anterior, entre otras cosas, en que estos dispositivos se programan mediante MicroPython en lugar del tedioso C. Vamos a analizar uno de sus productos estrella, el Lopy:

- **Microcontrolador:** Espressif ESP32 chipset
- **Alimentación recomendada:** 3.3-5V
- **Pines digitales I/O:** 24
- **Entradas analógicas:** 24
- **Corriente máxima por pin:** 40 mA
- **WIFI:** 802.11b/g/n 16mbps
- **Precio:** Alrededor de 35€

Crítica al estado del arte

Después de haber analizado los distintos ecosistemas que existen actualmente en el mercado se han encontrado distintos problemas.

[4] *Arduino - Home*. (2005). Arduino. https://www.arduino.cc/

[5] https://pycom.io/

En primer lugar, pese a que ofrecen soluciones muy completas y válidas, son soluciones poco configurables y que nos tenemos que adaptar a los sensores que existen en cada marca en especifico, y en algunas ocasiones no existe gran variedad.

En segundo lugar, muchas veces estas soluciones no pasan solo por la red domestica. Aunque a mucha gente le guste poder acceder a sus dispositivos domésticos desde cualquier lugar, esto implica un riesgo ya que cualquier persona podría estar obteniendo información de nuestro hogar, incluso podría guardar esta información para analizar nuestros hábitos a largo plazo.

Finalmente, el último inconveniente es el precio de estos componentes. Aunque con el avance del IoT en nuestras vidas cada obtenemos sensores con precios mas competitivos, la realidad es que en muchas ocasiones el precio por cualquier sensor es mucho mas elevado de lo que es el mismo elemento sin conexión a internet.

Propuesta

Después de haber analizado los inconvenientes de los productos comerciales hemos llegado a la siguiente conclusión. Necesitamos un sistema capaz de incorporar una gran cantidad de dispositivos sin importar la empresa que fabrique cada dispositivo.

Además, nuestra solución deberá estar alojada completamente en una red privada para garantizar la privacidad de nuestros datos. Pese a este último requisito, también debería facilitar alguna forma de acceder a estos dispositivos desde el exterior, pero sin exponer los datos del sistema. Por lo tanto, debería existir algún tipo de autenticación.

3. Análisis del problema

En primer lugar, debido a que este proyecto esta obligado a conectar distintos dispositivos por red, y algunos de ellos de forma inalámbrica se debería tener en cuenta la seguridad de esta red. En muchos, cuando hablamos de dispositivos IoT estamos hablando de dispositivos que se conectan mediante wifi a la red de casa. Por lo tanto, en la solución deberemos tener alguna forma de verificar la conexión.

En segundo lugar, al ser dispositivos que vamos a tener en nuestros hogares pueden tener algún tipo de información sensible. Pese a que con los dispositivos que vamos a utilizar para la solución no exista ningún tipo de sensor que obtenga información sensible, se debería tener en cuenta la codificación y protección de datos para que con futuras ampliaciones y usos de otros dispositivos no se exponga esta información.

Después de haber analizado parámetros abstractos de la aplicación vamos a pasara enumerar los requisitos funcionales de esta en la siguiente tabla.

Referencia	Requisito
RF - 1	La aplicación deberá ser capaz de comunicarse con distintos dispositivos
RF – 2	La aplicación deberá ser capaz de mostrar la información.
RF – 3	La aplicación deberá ser capaz de solicitar información de los dispositivos.
RF - 4	La aplicación deberá ser capaz de almacenar la información de los dispositivos

Tabla 1. Requisitos funcionales

Finalmente vamos a enumerar algunos de los distintos requisitos no funcionales que debería tener la aplicación.

Referencia	Requisito
NRF - 1	El tiempo de aprendizaje del sistema debe ser menos a 5 horas
NRF – 2	Toda la información de los sensores tiene que mantenerse en una red privada para su privacidad.
NRF - 3	El sistema tiene que ser capaz de gestionar al menos 10 peticiones por minuto.
NRF – 4	El sistema tiene que recibir nuevos datos cada 2 minutos.

Tabla 2. Requisitos no funcionales

4. Solución propuesta

Este capítulo contendrá la explicación de la evolución del proyecto con cada reunión con el tutor. Una breve descripción de un presupuesto básico de cuánto podría suponer montar un sistema completo con nuestra solución, tanto el coste de materiales como el coste humano de montar el sistema.

Después de esto, se hará una pequeña introducción a la solución propuesta para pasar finalmente a una descripción detallada de la solución propuesta.

Finalmente se explicarán los distintos dispositivos y tecnologías utilizadas para llegar a la solución propuesta.

4.1. Evolución del proyecto

Básicamente durante el proyecto hemos tenido 4 grandes reuniones en las que se han ido añadiendo más requisitos y cambios al proyecto.

En la primera de las reuniones se hizo una pequeña introducción al protocolo MQTT, incluyendo una explicación del modelo publicador-suscriptor y otra explicación de qué son los tópicos. También se hizo entrega de un dispositivo Lopy junto a la placa de extensión Pysense, junto a la entrega se hizo una breve explicación del funcionamiento de los dispositivos y se indicó dónde se podía conseguir documentación sobre ellos.

En la siguiente reunión se plantearon dos pequeñas actividades, ambas actividades incluían poder conectar los dispositivos a un bróker MQTT. La primera de las actividades era enviar periódicamente información sobre los sensores del dispositivo. La siguiente actividad consistía en poder solicitar información al dispositivo sobre los distintos sensores.

En la tercera reunión se planteó ya como se tenía que comportar el dispositivo, que en este caso era una combinación de las 2 actividades anteriores. Siendo posible para este enviar información de forma periódica de su estado y si es necesario poder solicitarle la información en un momento específico.

En la última reunión se sugirió la inclusión de algún sistema más complejo para tener más control sobre los datos obtenidos desde el Lopy, teniendo en cuenta que todos los datos obtenidos por este tipo de dispositivo son métricos se sugirió utilizar Grafana.

4.2. Presupuesto

En este presupuesto se van a evaluar los dos tipos de gastos, los materiales y el coste de las horas de trabajo.

En los gastos materiales se tendrán en cuenta sólo los materiales específicos para el desarrollo de la solución, sin tener en cuenta costes de redes adicionales, suponiendo que en la mayoría de las casas ya existen estos elementos.

Elemento	Precio	Cantidad
Lopy1.0r	38.45€	1
Pysense V1.1	29.65€	1
Raspberry Pi 4 Modelo B 4GB	86.44€	1
Total	154.54€	

Tabla 3. Presupuesto hardware

Estos gastos sólo incluyen la instalación de un solo dispositivo MQTT y la de el dispositivo que va a usarse como broker.

Respecto al coste de las horas de trabajo, vamos a tener en cuenta que el salario medio de un ingeniero informático que acaba de terminar el grado ronda entre los 18.000 y los 22.000 euros brutos. Por lo que se van a utilizar los 18000 euros para sacar el coste por hora que es de aproximadamente 7 euros.

También para las horas del coste de trabajo solo vamos a tener en cuenta la instalación de un nuevo entorno en el que tendremos un solo dispositivo y un broker configurado. En este cálculo no tendremos en cuenta el coste de las horas de investigación previas a dicha instalación

Actividad	Horas	Precio
Instalar software en Lopy	1	7€
Instalar SO en Raspberry Pi	1	7€
Instalar broker MQTT en Raspberry Pi	1	7€
Configurar paneles en Home Assistant	3	21€
Instalar InfluxDb	1	7€
Instalar Grafana	1	7€
Configurar paneles Grafana	2	14€
Total	10	70€

Tabla 4. Coste de horas

Después de tener estos gastos aproximados de todos los costes posibles podemos tener un presupuesto aproximado de una instalación doméstica básica.

Tipo	Precio
Material	154.54€
Horas de trabajo	70€
Total	224.54€

Tabla 5. Coste total

4.3. Arquitectura del sistema

Para explicar la arquitectura del sistema podemos recurrir al avance por las distintas etapas del proyecto, coincidiendo estas con las reuniones realizadas con el tutor.

El primer sistema que se realizó era sólo capaz de enviar información desde el dispositivo al broker de forma periódica sin tener capacidad de recibir información de otros dispositivos o del mismo broker.

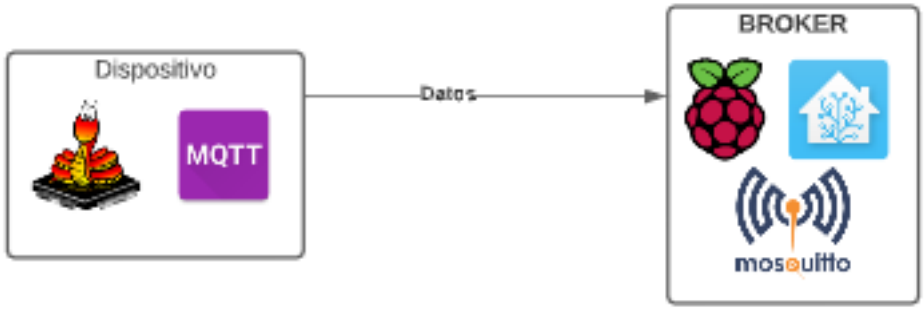

Ilustración 2. Primera aproximación al problema

La siguiente integración nos va a permitir hacer peticiones desde el mismo broker, que gracias a la incorporación de Home Assistant[6] nos va a facilitar una interfaz gráfica en la cual podemos añadir botones para realizar distintas acciones.

Ilustración 3. Segunda aproximación al problema

En la última de las modificaciones se va a permitir la persistencia de datos y también la visualización de forma más precisa de estos datos, tanto para analizarlos después de obtenerlos como una visualización en tiempo real.

[6] https://www.home-assistant.io/

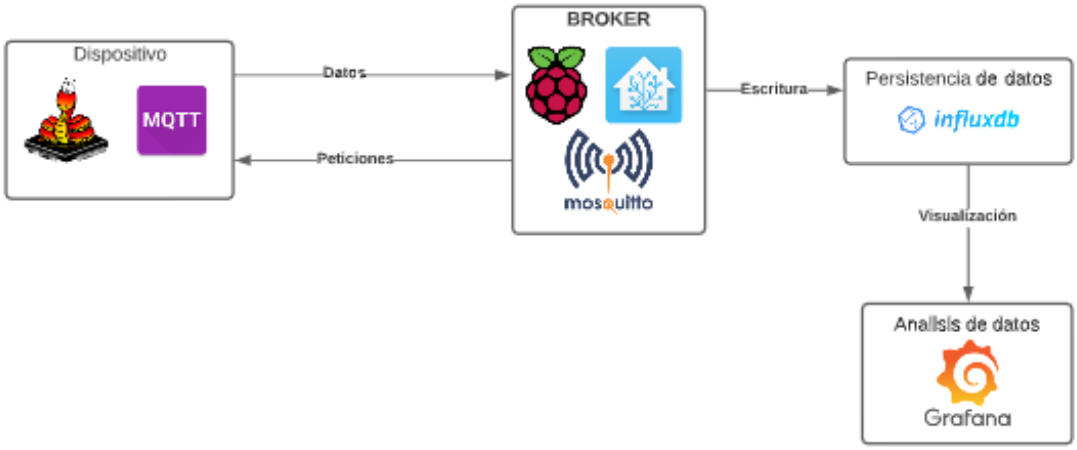

Ilustración 4. Aproximación final al problema

4.4. Diseño detallado

Para esta sección vamos a partir con la información del apartado anterior y vamos a precisar detalles más concretos tanto de el dispositivo, el broker, la aplicación y todas las conexiones entre los distintos elementos.

Antes de pasar a especificar el comportamiento concreto de los distintos elementos nos va a ser imprescindible entender el protocolo MQTT. Como ya hemos mencionado anteriormente el MQTT está basado en un modelo publicador-suscriptor. A este patrón le tenemos que añadir un intermediario que es lo que denominamos broker, la función de este broker es poner en contacto a los distintos publicadores y suscriptores.

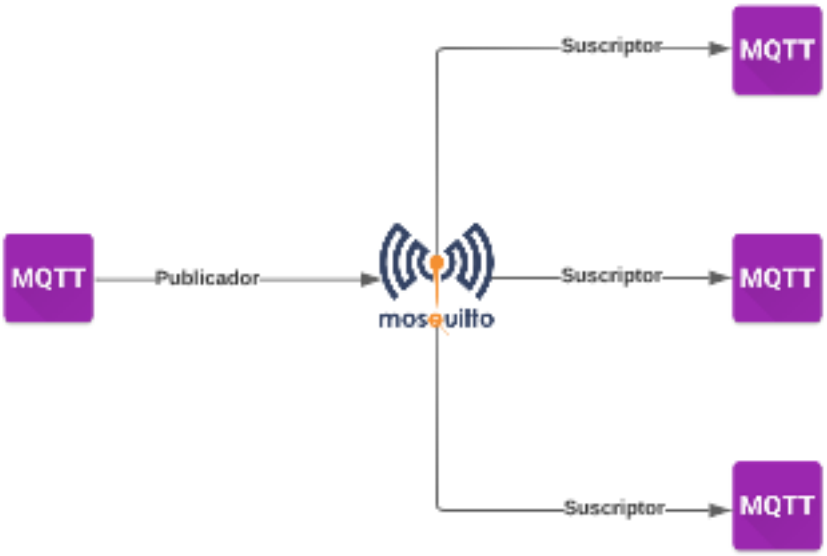

Ilustración 5. Ejemplo básico publicador suscriptor

También cabe destacar que un mismo dispositivo MQTT puede ser a la vez publicador y suscriptor, esto suele ser lo más habitual para poder enviar y recibir información de otros sensores y reaccionar a la información obtenida.

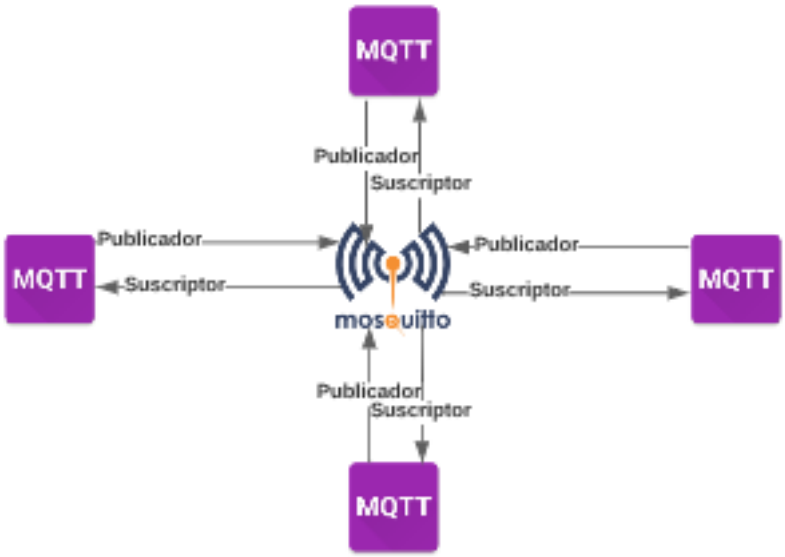

Ilustración 6. Ejemplo completo modelo publicador suscriptor

También tenemos que conocer cómo se suscriben los instrumentos, y es mediante tópicos. Un tópico MQTT es una cadena de texto que el bróker utiliza para filtrar los mensajes de cada cliente. Por lo tanto, si tenemos un tópico llamado *pizza* y queremos ver los mensajes que se envían en este tópico nos tendremos que suscribir a él. Los tópicos se pueden dividir en niveles mediante una barra, por lo que normalmente nos encontraremos con tópicos multinivel ya que puedes concretar más el tipo de mensajes.

En estos tópicos multinivel existen comodines, el primero de ellos es el símbolo + el cual se utiliza para sustituir un solo nivel del tópico por lo que si tenemos los tópicos *restaurante/pizza* y *restaurante/paella* podemos suscribirnos a los mensajes de los dos suscribiéndonos al tópico *restaurante/+* . El otro comodín es el símbolo #, el cual nos servirá para suscribirnos a todos los tópicos que empiecen por algo en concreto por ejemplo si tenemos los tópicos restaurante/pizza, restaurante/paella y el tópico restaurante/camarero/terraza nos podemos suscribir a todos utilizando el tópico restaurante/# mientras que si utilizáramos el símbolo + solo nos estaríamos suscribiendo a los 2 primeros.

Tópico \ Suscripción	restaurante\+\carbonara	restaurante\pizza\#
restaurante\pizza\carbonara	Suscrito	Suscrito
restaurante\espagueti\boloñesa	No suscrito	No suscrito
restaurante\pizza\barbacoa	No suscrito	Suscrito
restaurante\espagueti\carbonara	Suscrito	No suscrito

Tabla 6. Ejemplo tópicos

Ahora que ya hemos visto unos ejemplos de tópicos genéricos, vamos a especificar los tópicos utilizados en nuestra aplicación. Estos tópicos son multinivel con el siguiente diseño *edificio/habitación/dispositivo/parámetro* para los tópicos que nos envían información y el siguiente diseño para los tópicos que necesitan algún tipo de acción del dispositivo *edificio/habitación/dispositivo/parámetro/acción.*

Cliente MQTT

Después de haber resumido cual es el funcionamiento de MQTT ya podemos describir el funcionamiento exacto de nuestro cliente, en este caso el dispositivo utilizado es un LoPy con la placa Pysense incorporada. La placa nos facilita sensores de temperatura, humedad, luminosidad, presión, altitud, balanceo e inclinación, a parte de facilitarnos conexión WIFI.

Ilustración 7. Lopy y sus sensores

Al contar con estos sensores se ha optado por asignarle tópicos con la siguiente información, el edificio asignado es home, la habitación asignada es myroom, como dispositivo se le ha asignado LoPy y después se ha añadido cada uno de los sensores. Aparte de estos tópicos que son los que el LoPy tiene capacidad para publicar también se ha suscrito al dispositivo a los respectivos tópicos de cada sensor con la acción get para facilitar la petición en ese momento del valor de cualquiera de los sensores.

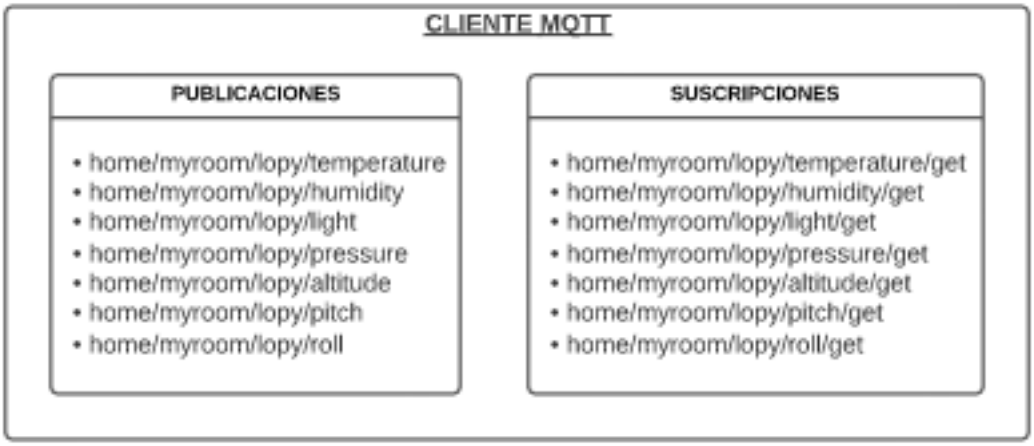

Ilustración 8. Tópicos cliente MQTT

Cabe destacar que este dispositivo se programa en MicroPython, el cual nos permite tener aplicaciones multihilo. Gracias a poder soportar más de un hilo se ha podido tener un hilo que es capaz de enviar los valores que están registrando los dispositivos y otro hilo que está esperando la solicitud del valor de cualquier dispositivo para poder enviarla en el momento.

Servidor

El servidor va a ser el instrumento que nos va a permitir alojar todas las aplicaciones que nos van a servir para conectar todos los dispositivos y analizar todos los datos obtenidos por estos dispositivos.

En primer lugar, el dispositivo que vamos a utilizar como servidor es una Raspberry Pi 4 model B con 4gb de RAM. Este servidor va a tener instalado el sistema operativo Home Assistant, el cual nos va a permitir conectar distintos dispositivos IoT, tanto por MQTT como dispositivos de distintos ecosistemas comerciales como Amazon Alexa o Samsung Smarthings.
Este sistema operativo nos permite, mediante Docker, instalar distintos tipos de aplicaciones que normalmente están relacionadas con el IoT o con la gestión de servidores por lo que nos va a facilitar el trabajo permitiéndonos desplegar la mayoría de las funcionalidades que necesitamos de forma casi automática. También cabe destacar que, aunque el propio servidor es accesible desde línea de comandos, este sistema operativo nos va a facilitar una aplicación web completamente configurable disponible en el puerto 8123 de este dispositivo.

El siguiente elemento que nos va a ofrecer este dispositivo es el broker MQTT, en este caso estaremos utilizando el broker [7]Mosquitto, que como cualquier otro broker MQTT va a estar disponible en el puerto 1883. Después de instalar estas dos aplicaciones y aplicar un poco de configuración, la cuál será explicada en los siguientes apartados, ya tendremos la primera aproximación al problema.

[7] https://mosquitto.org/

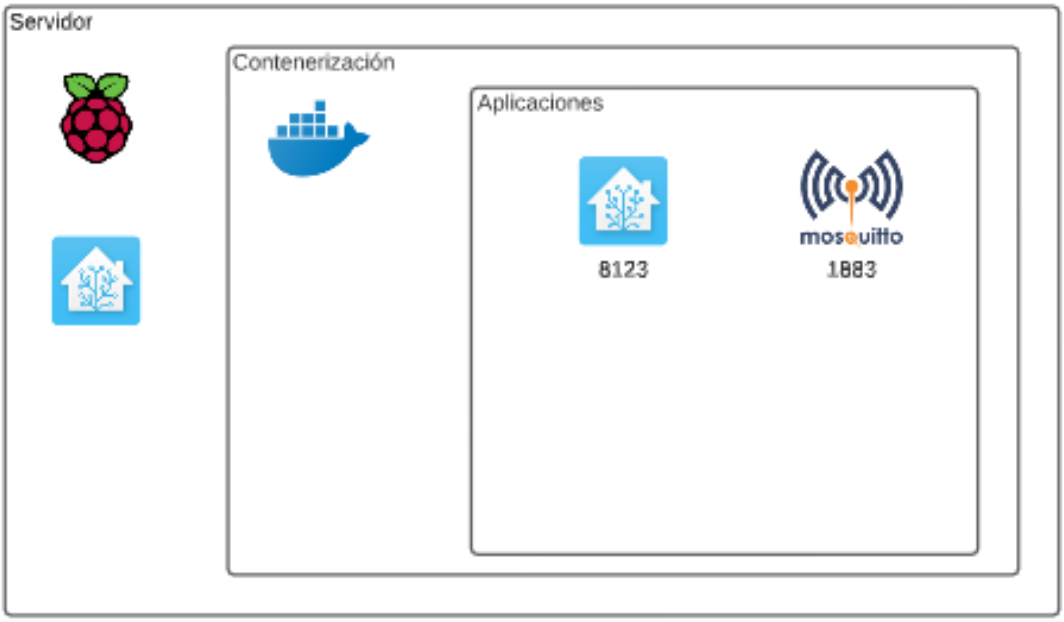

Ilustración 9. Ejemplo primera aproximación al problema

Para la siguiente aproximación nos falta incluir persistencia de datos y una plataforma para poder analizar los datos de forma más exhaustiva que en la propia aplicación de Home Assistant. Para solventar la persistencia de datos vamos a instalar, con el instalador automático de Home Assistant, la base de datos InfluxDB. Home Assistant y InfluxDB tienen una integración casi automática que va a registrar los valores que obtengamos desde nuestros sensores. Esta base de datos estará disponible en el puerto 8086 de nuestro servidor.

Para acabar con las aplicaciones disponibles en nuestro servidor nos falta incluir Grafana, que también cuenta con una integración en Home Assistant y se puede configurar fácilmente para obtener datos de nuestra base de datos.

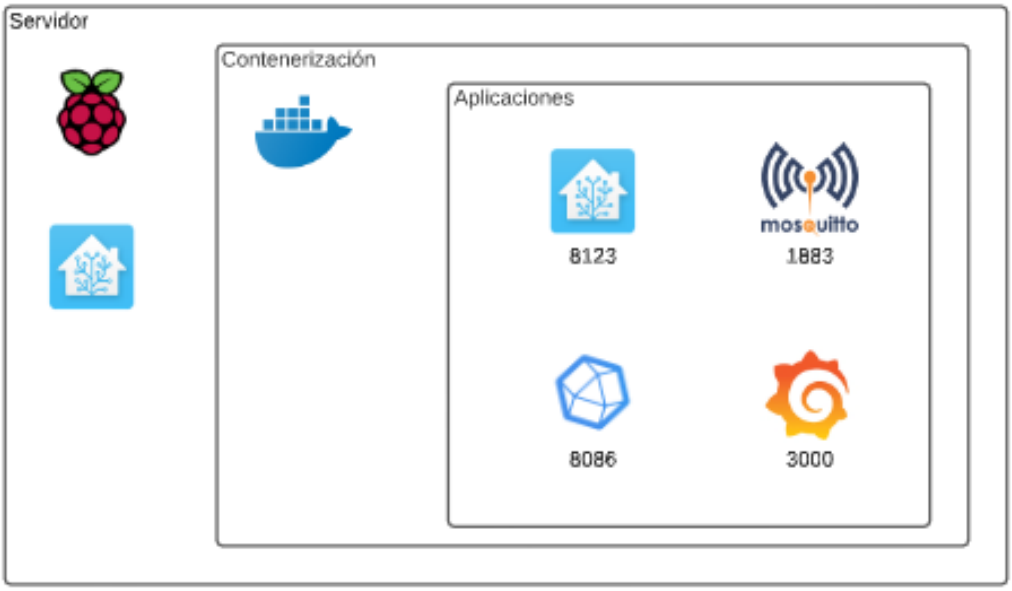

Ilustración 10. Aproximación final parte servidor

Aunque no sea uno de los usos más habituales del servidor en algunas situaciones también puede ser utilizado como cliente MQTT para enviar y recibir información a distintos clientes MQTT.

Panel de instrumentos

El último dispositivo, es el que nos facilitará la interfaz gráfica, que será explicada en apartados posteriores. Aunque con este dispositivo podamos acceder a cualquier aplicación de nuestro servidor indicando el puerto correspondiente, solo utilizaremos la aplicación de Home Assistant ya que mediante la creación de paneles personalizados podremos acceder al resto de integraciones si nos fuera necesario.

Cabe destacar que como panel de instrumentos nos servirá cualquier dispositivo que pueda acceder a páginas web o que sea capaz de instalar la aplicación cliente de Home Assistant.

Conexión entre dispositivos

Finalmente nos queda explicar la conexión entre los distintos elementos del sistema. Este sistema está centralizado en un único servidor y los clientes no pueden comunicarse entre ellos, por lo tanto, todas las comunicaciones que hagamos pasan antes por el servidor.

En primer lugar, vamos a explicar los contactos que tiene el servidor con elementos externos a él. En este caso los únicos contactos que tiene con el exterior son el puerto 1883 para comunicarse con los clientes MQTT y el otro contacto que tiene es la conexión con el panel de instrumentos mediante el puerto 8123.

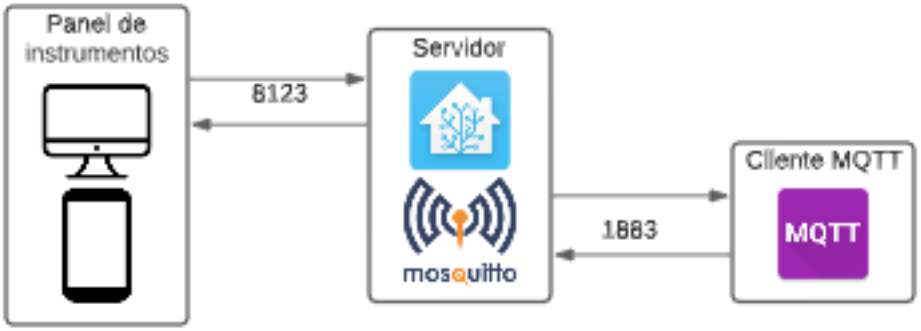

Ilustración 11. Ejemplo de la comunicación entre dispositivos

En segundo lugar, cabe destacar la conexión de los distintos elementos internos del servidor. Internamente el servidor tiene 4 elementos: Home Assistant, Mosquitto MQTT, InfluxDB y Grafana. Home Assistant está conectado directamente con el resto de los elementos. Primero está conectado con Mosquitto MQTT que actúa como broker MQTT, después está conectado con InfluxDB que actúa como base de datos y finalmente está conectado con Grafana que permite la generación de gráficos con los datos de la base de datos. Adicionalmente Grafana e InfluxDB están interconectados para poder obtener los datos desde Grafana para generar los gráficos.

Ilustración 12. Ejemplo de comunicación entre aplicaciones

4.5. Tecnología utilizada

En este apartado vamos a hablar sobre las tecnologías utilizadas en el proyecto que pueden tener más relevancia tanto en el proyecto en sí como para proyectos futuros.

- Raspberry Pi: Es un microcomputador de bajo coste desarrollado por la fundación Raspberry Pi. Pese a que su primer objetivo era facilitar a todas las personas el uso de la informática también ha sido muy útil, debido a su reducido tamaño y coste, para aplicaciones como robótica.
- Lopy: Es un microcontrolador compacto de la compañía Pycom que nos permite distintos tipos de conexión inalámbrica y que es utilizado en una amplia gama de productos de IoT.
- Pysense: Placa de expansión compatible con un microcontrolador Lopy, también fabricado por la compañía Pycom. Esta placa nos incluye sensores de luz ambiente, presión, humedad, acelerómetro, temperatura y nos incluye el puerto de conexión USB con acceso serial.
- Home Assistant: Sistema operativo de código abierto que está diseñado para ser el sistema de control central de un sistema demótico.
- Docker: Sistema de contenerización de software de código abierto, capaz de proporcionar una capa de abstracción superior al de una máquina virtual.
- Github: Plataforma, de la compañía Microsoft, que nos permite alojar todo tipo de proyectos mediante el sistema de control de versiones de Git.
- Git: Sistema de control de versiones, el cual es software libre, que nos permite versionar todo tipo de proyectos de forma eficiente.
- Python: Lenguaje de programación de código abierto, interpretado y multiparadigma. Este lenguaje está muy extendido tanto en el sector educativo como en el sector del IoT. Aunque en este proyecto no está presente en la solución final ha sido utilizado para probar la comunicación entre clientes MQTT.
- MicroPython: Implementación del lenguaje Python optimizada para ser usada en microcontroladores.
- Yaml: Lenguaje de serialización de datos que es utilizado normalmente para guardar configuración de aplicaciones o para transmitir datos en formato de texto.
- Visual Studio Code: Editor de código, de la compañía Microsoft, el cuál es gratuito y de código abierto. Al ser de código abierto cuenta con una amplia cantidad de extensiones hechas por la comunidad.

- Pymakr: Extensión disponible en varios editores de código que vamos a utilizar para subir el código MicroPython a nuestro microcontrolador.
- Vim: Editor de código desde terminal que en muchas ocasiones es útil para modificar configuración de servidores de forma eficiente.
- Mosquitto MQTT: Broker MQTT de la empresa Eclipse, es de bajo consumo y compatible con todo tipo de dispositivos.
- InfluxDB: Base de datos de código abierto y no relacional. Esta aplicación está programada con el lenguaje Go por lo que soporta concurrencia de forma eficiente.
- Grafana: Aplicación que nos sirve para generar gráficos desde gran variedad de fuentes y tipos de datos distintos.

5. Desarrollo de la solución propuesta

Pese a que en la propuesta inicial se sugiere utilizar Thunkable para desarrollar la aplicación, durante la investigación descubrimos que la nueva versión de esta herramienta no era compatible con los módulos de cliente MQTT. Tras intentar utilizar la versión antigua de Thunkable nos dimos cuenta de que solo era capaz de generar aplicaciones para dispositivos Android y empezamos a buscar alternativas. En este proceso encontramos Home Assistant, aunque este solo ofrecía soluciones para dispositivos IoT era compatible con todo tipo de dispositivos. Por lo tanto, aunque Thunkable ofrece más posibilidades en cuanto a desarrollo de aplicaciones, nos decantamos por Home Assistante ya que nos ofrece mayor compatibilidad con dispositivos y con las tecnologías necesarias para este proyecto.

La siguiente investigación que se realizó fue para elegir el broker MQTT que se iba a utilizar. Los dos brókeres que analizamos fueron Mosquitto y RabbitMQ, en esta elección nos decantamos rápidamente por Mosquitto ya que contaba con una integración casi automática con Home Assistant y las características generales eran muy similares.

Después de tener claro cuales iban a ser los principales componentes de software del proyecto se empezó a experimentar con los mensajes MQTT, para esto se utilizaron clientes desarrollados en Python para entender cómo funcionaban realmente los tópicos y los distintos usos que le podíamos dar a estos para ordenar la información y generar una comunicación clara y efectiva entre los distintos dispositivos.

Respecto a los tópicos se buscaron las mejores prácticas en guías oficiales para poder organizarlos. En esta etapa nos surgió la duda de cómo utilizar los tópicos para pedir a los dispositivos información sobre sus estados, problema que resolvimos añadiendo un nivel de tópico extra para añadir la acción que necesitábamos del dispositivo.

Por último, en la parte del servidor solo nos queda el porqué de la inclusión de Grafana en la ecuación. Aunque en la propuesta inicial no se incluía ningún tipo de software externo de monitorización, tras ver que Home Assistant no tenía ningún tipo de monitorización mediante gráficos y que tenía una increíble facilidad para incluir una aplicación como Grafana que permite formar gráficos de todo tipo a partir de datos numéricos.

En la parte del cliente, lo primero que se hizo fue investigar sobre el propio dispositivo y su lenguaje de programación. En principio se hicieron ejercicios que no estaban propiamente relacionados con el proyecto como encender leds con el Lopy para ganar familiaridad tanto con el nuevo lenguaje como el dispositivo. Tras estos pequeños ejercicios ya se realizaron las dos primeras aproximaciones, la primera que simplemente envía información de los sensores periódicamente, la segunda solo envía información bajo demanda y finalmente la solución final combina el comportamiento de las anteriores mediante el control de hilos. El código versionado de estas soluciones está disponible en Github.

6. Implantación

Este proyecto solo se ha implantado en entornos de pruebas ya que al ser parte de un sistema mas amplio tampoco hacía falta integrarlo en un sistema de producción. Pese a esto el sistema funciona tal y como estaba planeado tras el desarrollo de la solución final.

Aunque durante el diseño nunca se pensó sobre que tipo de dispositivo íbamos a utilizar para alojar la aplicación en cuanto empezamos a necesitar probar los sensores y ver como se comportaba el sistema durante periodos de tiempo que pudieran darnos información suficiente para saber que el dispositivo era capaz de mostrar valores correctos, nos dimos cuenta de que necesitábamos una maquina dedicada. Tras esta decisión nos planteamos utilizar una maquina virtual o alojarlo en una Raspberry Pi. La decisión fue fácil ya que utilizar la Raspberry Pi nos acercaba más al entorno de las IoT, por lo que se opto por esta.

Después de haber hecho la instalación inicial del servidor, pensábamos que todo se iba a instalar de forma automática con el instalador automático que integra Home Assistant pero la realidad fue algo distinta ya que la mayoría de aplicaciones solo nos permitían integrar elementos muy esenciales desde sus instaladores automáticos. Después de descubrir esto tuvimos que aprender a modificar archivos de configuración con editores de texto desde el terminal. Esto fue debido a que para implantar elementos más avanzados era necesario hacer configuración en ficheros de tipo Yaml.

```
sensor:
  - platform: mqtt
    state_topic: "home/myroom/lopy/temperature"
    name: "home_myroom_lopy_temperature"
    unique_id: "home_myroom_lopy_temperature"
    device_class: "temperature"
  - platform: mqtt
    state_topic: "home/myroom/lopy/humidity"
    name: "home_myroom_lopy_humidity"
    unique_id: "home_myroom_lopy_humidity"
    device_class: "humidity"
  - platform: mqtt
    state_topic: "home/myroom/lopy/light"
    name: "home_myroom_lopy_light"
    unique_id: "home_myroom_lopy_light"
    device_class: "illuminance"
  - platform: mqtt
    state_topic: "home/myroom/lopy/pressure"
    name: "home_myroom_lopy_pressure"
    unique_id: "home_myroom_lopy_pressure"
    device_class: "pressure"
  - platform: mqtt
    state_topic: "home/myroom/lopy/altitude"
    name: "home_myroom_lopy_altitude"
    unique_id: "home_myroom_lopy_altitude"
  - platform: mqtt
    state_topic: "home/myroom/lopy/pitch"
    name: "home_myroom_lopy_pitch"
    unique_id: "home_myroom_lopy_pitch"
  - platform: mqtt
    state_topic: "home/myroom/lopy/roll"
    name: "home_myroom_lopy_roll"
    unique_id: "home_myroom_lopy_roll"
```

Ilustración 13. Ejemplo configuración YML

En la parte del cliente MQTT la implantación fue bastante similar a lo planeado anteriormente. El único problema fue que en ocasiones la extensión Pymakr, que nos permitía subir el código, dejaba de funcionar y no teníamos posibilidad de aplicar cambios en el dispositivo.

7. Pruebas

Respecto a las pruebas, no se pueden realizar ningún tipo de pruebas de carga ya que solo disponemos de un dispositivo, por lo que esto para el sistema no supone ningún tipo de problema. Los problemas de comunicación de red tampoco suponen problemas ya que se trata de una red privada con poco tráfico.

Tras saber que en el sistema de pruebas que hemos preparado no se puede comprobar la eficiencia de este solo nos quedaría comprobar si cumple con las funcionalidades acordadas.

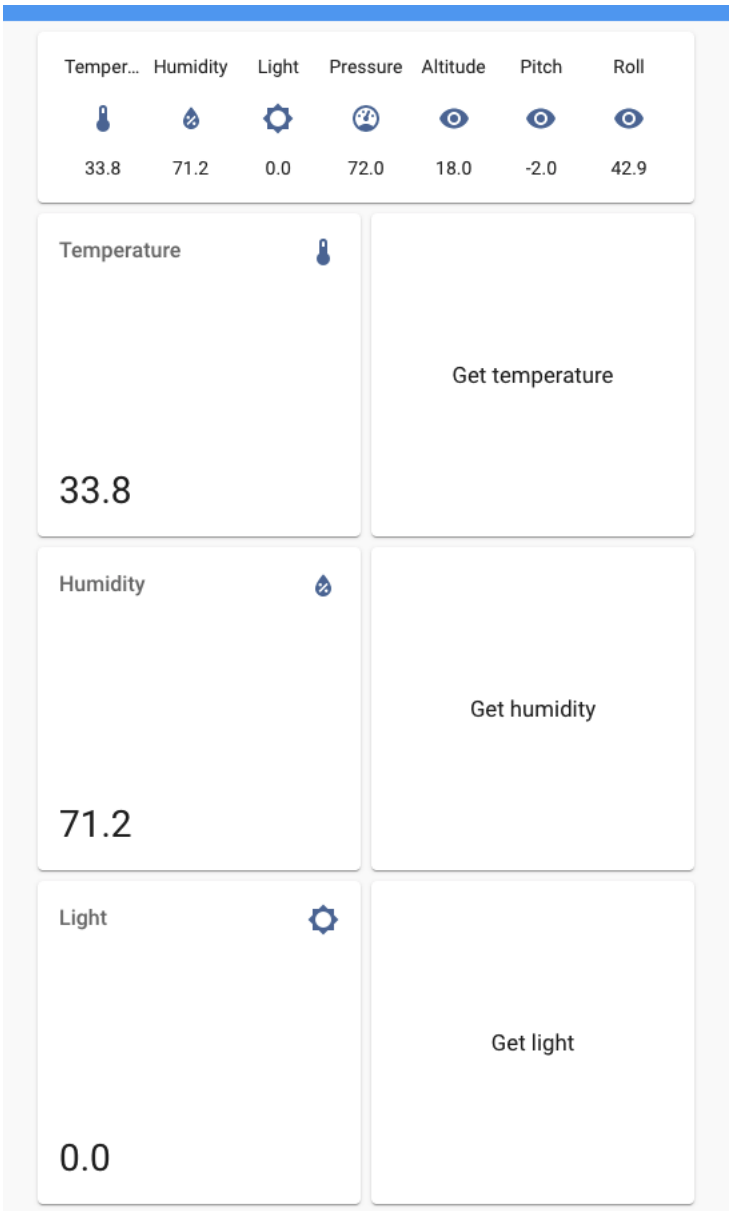

Ilustración 14. Ejemplo panel principal de la aplicación

En la imagen anterior se puede ver en el panel superior la información obtenida por los sensores en el momento actual. Después de haber probado hemos descubierto que la información obtenida en alguno de ellos, en este caso la altitud, no encaja con la

realidad, pero parece ser un problema del tipo de sensor por lo que no lo entenderemos como fallo de la solución propuesta.

Después de ver el comportamiento del panel superior, que existe a modo de resumen del estado de los dispositivos, podemos ver que en cada apartado existe un botón para cada tipo de sensor. Este comportamiento se asocia a la petición que realizaríamos en servidor para recargar la información que tenemos del valor de este sensor en concreto. También podemos concluir que cumple con su función

8. Conclusiones

Como conclusión, tras haber podido completar el objetivo del proyecto tanto la parte de comunicación entre dispositivos como la parte de representación y gestión de datos.

En lo que respecta a la comunicación de dispositivos, después de haber diseñado el modelo con el que se iba a identificar cada dispositivo no ha existido ningún problema para la comunicación entre distintos dispositivos.

Por lo que se refiere a la gestión y representación de datos, era la parte en la que menos conocimientos teníamos y por lo tanto existió una investigación mas extensa. En la primera aproximación se descubrió que la aplicación sugerida para realizar la aplicación era incompatible con los requisitos, en este caso eran poderse comunicar con MQTT y ser compatibles para dispositivos Android y IOT. Después de descubrir este inconveniente cambiamos el sistema para que fuera compatible con estos requerimientos.

Tras solucionar este inconveniente, encontramos dos problemas extra. El primero de ellos es que el sensor que incorpora el Lopy para detectar la presión atmosférica, en ocasiones si estamos en el interior de un edificio no sabe detectar la presión real del ambiente y por tanto no nos indica con precisión ni la altura a la que nos encontramos ni la presión que existe. El último problema es que en ocasiones el sistema de representación de datos reinicia los paneles existentes en el y desparecen.

Sobre estos problemas, no conseguimos solucionar ninguno de los dos problemas, el primero de ellos porque es un problema completamente ajeno al desarrollo de la solución y el segundo de ellos no conseguimos encontrar ningún patrón en el borrado de estos paneles por lo que no conseguimos reproducir dicho fallo.

Después de haber analizado las conclusiones técnicas del proyecto nos gustaría analizar la conclusión personal de dicho proyecto. Este proyecto ha sido una forma muy interesante de iniciarme en el mundo de los dispositivos IoT, ya que pesa solo disponer de un dispositivo, este dispositivo permitía bastantes desarrollos distintos. Adicionalmente se me ha permitido investigar y poner en práctica algunas tecnologías que no estaban establecidas en el proyecto inicial, por lo que ha sido muy enriquecedor e interesante.

8.1 Relación del trabajo desarrollado con los estudios cursados

Después de haber cursado los cuatro cursos del grado y la especialización en tecnologías de la información, se podría decir que hemos obtenido una gran cantidad de conocimientos, tanto conocimientos técnicos, como legales, como de organización de proyectos. Se podría decir que este proyecto consta de una gran amplitud, aunque no se haya llegado a profundizar demasiado en ningún campo de los que existían en el proyecto, por lo que se va a pasar a mencionar las competencias que se han desarrollado en este proyecto.

Primero, antes de empezar a programar se ha tenido que diseñar un sistema capaz de permitir la comunicación entre distintos dispositivos, por lo que ha sido necesario la realización de diagramas para organizar el proyecto. Aunque en los borradores no se

han llegado a utilizar diagramas que encajen completamente con los modelos aprendidos si que se han aproximado suficiente.

En cuanto a la programación, esta competencia esta presente en todos los estudios. Tras haber cursado este grado y haber hecho las prácticas en una empresa que se dedica al desarrollo del software, he obtenido conocimientos para realizar software que se adecue a los requisitos planificados, documentado correctamente y que pueda ser reutilizable. Respecto a la reutilización se debería destacar el uso de un gestor de versiones, en este caso Git, que si fuera necesario haber trabajado en equipo nos lo hubiera permitido.

También me gustaría destacar que este proyecto también obliga a poder mantener una comunicación con el tutor, y a tener la capacidad y conocimiento suficiente para defender algún tipo de desarrollo tecnológico que realicemos en el futuro.

Finalmente, después de haber analizado las competencias obtenidas tanto en el proyecto como previamente en el grado, consideramos que tienen una gran relación y nos ha permitido desarrollar competencias tanto técnicas como sociales que resultan muy necesarias en el mundo laboral.

9. Trabajos futuros

Tras finalizar este proyecto hay algunos puntos que nos hubiera gustado examinar con mas profundidad. Entre ellos la seguridad de la aplicación y la automatización de tareas.

El primero de ellos pensamos que una buena aproximación sería la de incluir certificados para que todas las conexiones al servidor donde se aloja Home Assistant tengan que ser de forma segura. Otra aproximación sería solo permitir conexiones desde fuera de la red interna mediante VPN.

Respecto a la automatización de tareas, teniendo un solo dispositivo no tenía mucha utilidad automatizar tareas de un dispositivo desde el lado del servidor. Por lo tanto, para esto necesitaríamos incluir otro tipo de dispositivos como bien podría ser una bombilla inteligente y utilizar el mismo Lopy para saber si la bombilla debería estar encendida o no.

Tras haber mencionado las ampliaciones que consideramos mas importantes, también nos gustaría incluir la mejora de los paneles para la representación de datos. Debido a que este requeriría una menor investigación no lo consideramos de tanto interés.

Pese a haber mencionado solo 3 líneas de investigación consideramos que el campo del IoT es increíblemente amplío y por lo tanto existen infinidad de ampliaciones para este proyecto o investigaciones paralelas a este.

10. Referencias

Marketing Liberty Seguros. (2021, 23 marzo). Los cinco ecosistemas para convertir tu hogar en una casa inteligente. Blog de Génesis. https://blog.genesis.es/los-cinco-ecosistemas-para-convertir-tu-hogar-en-una-casa-inteligente/

Colaboradores de Wikipedia. (2021, 10 agosto). *Internet de las cosas*. Wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Internet_de_las_cosas

Wikipedia contributors. (2021, 29 agosto). *MQTT*. Wikipedia. https://en.wikipedia.org/wiki/MQTT

Hernández, L. D. V. (2020, 29 mayo). *Guía de introducción a MQTT con ESP8266 y Raspberry Pi*. Programar fácil con Arduino. https://programarfacil.com/esp8266/mqtt-esp8266-raspberry-pi/

Team, D. (2021, 9 mayo). *4 Key IoT Protocols – Learn In Great Detail*. DataFlair. https://data-flair.training/blogs/iot-protocols/

Sierra, Y. (2020, 15 enero). *Grafana vs. Kibana: ¿cuáles son las diferencias y cuál es mejor?* #ADNCLOUD. https://blog.mdcloud.es/grafana-vs-kibana-diferencias/

Pycom Ltd. (2021, 9 junio). *Pycom - Next Generation Internet of Things Platform*. Pycom. https://pycom.io

Team, T. H. (2019). *MQTT Topics & Best Practices - MQTT Essentials: Part 5*. HiveMQ. https://www.hivemq.com/blog/mqtt-essentials-part-5-mqtt-topics-best-practices/