UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADO EN INGENIERÍA DEL SOFTWARE

Aplicación IoT domótica usando Android Things

IoT home automation application using Android Things

Realizado por
Juan Muñoz Carrasco

Tutorizado por
Daniel Garrido Márquez

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA

MÁLAGA, FEBRERO DE 2018

Fecha defensa:

Fdo. El/la Secretario/a del Tribunal

Resumen

En los últimos años está creciendo el número de dispositivos de la vida cotidiana que están conectados entre sí mediante internet, este concepto se denomina Internet de las Cosas. Ante la gran demanda existente en la actualidad, se ha decidido realizar el trabajo de fin de grado utilizando tecnologías IoT mediante dos placas hardware, Arduino y Raspberry con Android Things. Además, para la visualización de los datos se ha añadido la creación de una aplicación móvil para mostrar todos los datos que se leen y se procesan en las placas hardware utilizando un protocolo de comunicaciones IoT como es MQTT.

Este trabajo se centra en la realización de tres aplicaciones distintas, una para una placa Arduino, otra para una placa Raspberry con el nuevo sistema operativo de Google para IoT Android Things, y una última para un dispositivo móvil donde se muestran los datos de los sensores de temperatura, humedad y luminosidad alojados en las placas hardware, y donde poder actuar sobre LEDs y ventiladores.

La elección de dicha aplicación ha sido promovida por la utilización del nuevo sistema operativo para el internet de las cosas Android Things. Tras leer varios artículos se comprobó que había poca información sobre realizar aplicaciones IoT utilizando Android Things. De esta manera se evalúa si es una buena solución para IoT o es mejor utilizar otros sistemas.

Palabras clave:

IoT, MQTT, broker MQTT, Arduino, Android Things, Raspberry, aplicación móvil, sensores.

Abstract

In recent years, the number of devices of daily life that are connected to each other through of Internet is growing, this concept is called the Internet of Things. In view of the great demand currently existing, it has been decided to carry out the end-of-degree proyect using IoT technologies by means of two hardware boards, Arduino and Raspberry with Android Things. In addition, for the visualization of the data has been added the creation of a mobile application to display all the data that read and processed the hardware boards using an IoT communications protocol such as MQTT.

This work focuses in the realization of three different applications, one for an Arduino board, one for a Raspberry board with the new Google operating system for IoT Android Things, and one last for a mobile phone where it displays the sensor data of temperature, humidity and luminosity housed in hardware boards, and where to act on LEDs and fans.

The choice of this application has been promoted by the use of the new operating system for the internet of things Android Things. After reading several articles it was found that there was little information about doing IoT applications using Android Things. In this way, it is evaluated if it is a good solution for IoT or it is better to use other systems.

Keywords:

IoT, MQTT, broker MQTT, Arduino, Android Things, Raspberry, mobile application, sensors.

Índice

1

Introducción

1.1 Motivación

Debido la creciente demanda de sistemas conectados entre sí, se ha querido desarrollar el trabajo de fin de grado sobre alguna aplicación para el mundo del Internet de las Cosas. A esto se sumaba la necesidad de acceder al mundo laboral y encontrar un trabajo relacionado con el Internet de las Cosas que no he encontrado como materia de estudio en los cuatro años en los que se dividen los estudios del Grado de Software.

Por ello, se decidió realizar una aplicación domótica IoT utilizando tecnologías que se han estudiado a lo largo de la carrera pero muy por encima y profundizar en ellas aprendiendo a usarlas realmente. Así surge utilizar Arduino y Android Studio para realizar una aplicación móvil que obtenga datos a tiempo real de distintos sensores.

Se le ha añadido la dificultad de aprender qué es y cómo se usa el nuevo sistema operativo de Google para IoT Android Things. Por lo tanto, aparte de aprender a realizar una aplicación móvil con gráficos y manejo de datos en tiempo real, se ha añadido la dificultad de uso de este sistema operativo para, por ejemplo, leer sensores o comunicarse mediante un protocolo de comunicaciones de IoT.

1.2 Objetivos

El objetivo principal de este trabajo es el desarrollo de una aplicación móvil que muestre datos de sensores en tiempo real y mediante la cual se pueda actuar sobre los mismos. Todo esto depende del desarrollo de dos aplicaciones para las placas Arduino y Raspberry, esta última con Android Things.

Sobre Arduino se realizará un programa que sea capaz de leer sensores tanto analógicos como digitales, conectarse a Wifi y tanto recibir como enviar mensajes por MQTT.

Sobre la realización de la aplicación móvil se comunicará mediante MQTT con las placas hardware para obtener los datos de los sensores y mostrarlos en la aplicación.

Y por último y lo más importante aprender sobre el nuevo sistema operativo de Google, Android Things, destacando por tanto las ventajas y desventajas del mismo. Dicho programa leerá los datos de los sensores y se comunicará por MQTT con la aplicación móvil para proporcionarle dicho datos.

1.3 Estructura de la memoria

Introducción

Se detalla en qué consiste el trabajo y los motivos por los que se ha realizado. Además, se explican las fases del trabajo junto con la estructura de la memoria.

Tecnologías y plataformas utilizadas

En esta sección se detallan las herramientas que se han utilizado para el desarrollo de las aplicaciónes y las tecnologías sobre las que se lanzan las aplicaciones.

Hardware y sensores utilizados

Las placas hardware y los sensores son parte fundamental en el trabajo de fin de grado por lo que esta sección detalla las placas y los sensores necesarios para hacer funcionar el sistema.

Documento general de requisitos

Esta sección describe al detalle las funcionalidades principales que se deberán realizar sobre las aplicaciones. Además, se detallan todos los requisitos no funcionales de los que depende la aplicación. Este documento es esencial en la realización del trabajo de fin de grado porque establece qué deben hacer las aplicaciones y qué limitaciones tiene.

Diseño e implementación

Esta sección describe al detalle el diseño de cómo deben ir conectados los sensores sobre las placas hardware y cómo van a ser las pantallas de la aplicación móvil.

Además, se explica al detalle cómo funcionan las aplicaciones de cada dispositivo. También se detalla cómo se ha implementado cada aplicación junto con las dependencias y las bases de datos utilizadas.

Mantenimiento y pruebas

En esta sección se detallan las labores de mantenimiento que se deben realizar sobre el sistema y posibles mejoras a realizar sobre la aplicación. También se detallan las pruebas que se han realizado sobre las aplicaciones.

Conclusión

Contiene una explicación de los principales problemas que han ido surgiendo durante el desarrollo y llevó más tiempo resolver. Además, se detallan opiniones personales sobre este trabajo de fin de grado y recomendaciones y sugerencias para líneas futuras.

1.4 Metodología de trabajo

Para la realización del trabajo de fin de grado se ha elegido una metodología incremental. De esta manera se han ido desarrollando las distintas funcionalidades de manera incremental probándolas en conjunto conforme se desarrollaba.

Durante la primera fase se realizó la investigación sobre el sistema operativo Android Things para saber cuáles eran los requisitos de uso y cómo poder lanzar una aplicación básica. Posteriormente se realizó el diseño de cómo iban a ir alojados los sensores sobre los dispositivos hardware.

En la segunda fase se realizó el documento general de requisitos para poder tener claro cuales iban a ser las funcionalidades a implementar en cada dispositivo. Una vez realizado lo anterior se pasó a la instalación de todos los sensores y a la implementación básica de los mismos para comprobar que se podía leer correctamente los sensores.

En la tercera fase se desarrollaron las comunicaciones. En este caso la implementación de la conexión Wifi sobre el dispositivo Arduino y posteriormente la implementación de las conexiones MQTT y realizando pequeñas pruebas de que las conexiones funcionaban correctamente.

En la cuarta fase se han desarrollado las principales funcionalidades tras las comunicaciones como el procesamiento de los datos o el formato de enviar o recibir los determinados datos.

Por último, se han desarrollado todas las pantallas junto con las gráficas para la aplicación móvil y el almacenamiento de las temperaturas.

2

Tecnologías y plataformas utilizadas

Debido a que se utiliza un Arduino Genuino Uno y una Raspberry con Android Things se han utilizado tecnologías y lenguajes de programación completamente distintos. A continuación se muestran las plataformas y tecnología utilizadas.

2.1 Sistema operativo Android

Es un sistema operativo diseñado sobre Linux. Está especialmente adaptado para dispositivos móviles y tabletas, pero está diseñado para ser insertado en sistemas empotrados. Es propiedad de Google y actualmente es utilizado por muchas marcas fabricantes de móviles. Una de las características más importantes es que posee un sistema operativo abierto que permite su utilización de una manera más sencilla. Hoy en día es uno de los sistemas operativos más usados en el mundo debido a la venta de móviles con este sistema operativo.

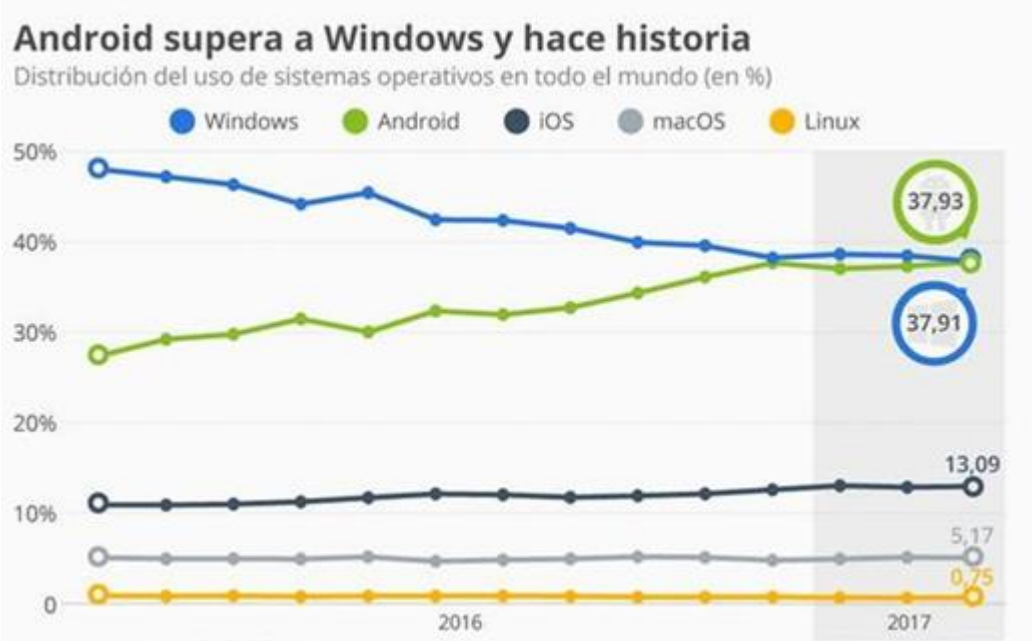

Ilustración 1 Sistemas operativos más usados en el mundo, Fuente: "eldiario.es"

2.2 Android Studio

Es un entorno de desarrollo centrado en la realización de aplicaciones móviles con un sistema operativo Android. Está basado en IntelliJ IDEA.

Esta herramienta es perfecta para el desarrollo porque facilita entre otras cosas importación de librerías o la creación de vistas unidas a clases Java. Posee un emulador para aplicaciones que facilita poder mostrar por consola errores producidos en la aplicación a la vez que puedes interaccionar con la aplicación. El lenguaje de programación que soporta la plataforma es Java ,Kotlin o C++. Para cualquier desarrollador es una gran plataforma de desarrollo porque existe mucha información para desarrollar aplicaciones en dicha plataforma.

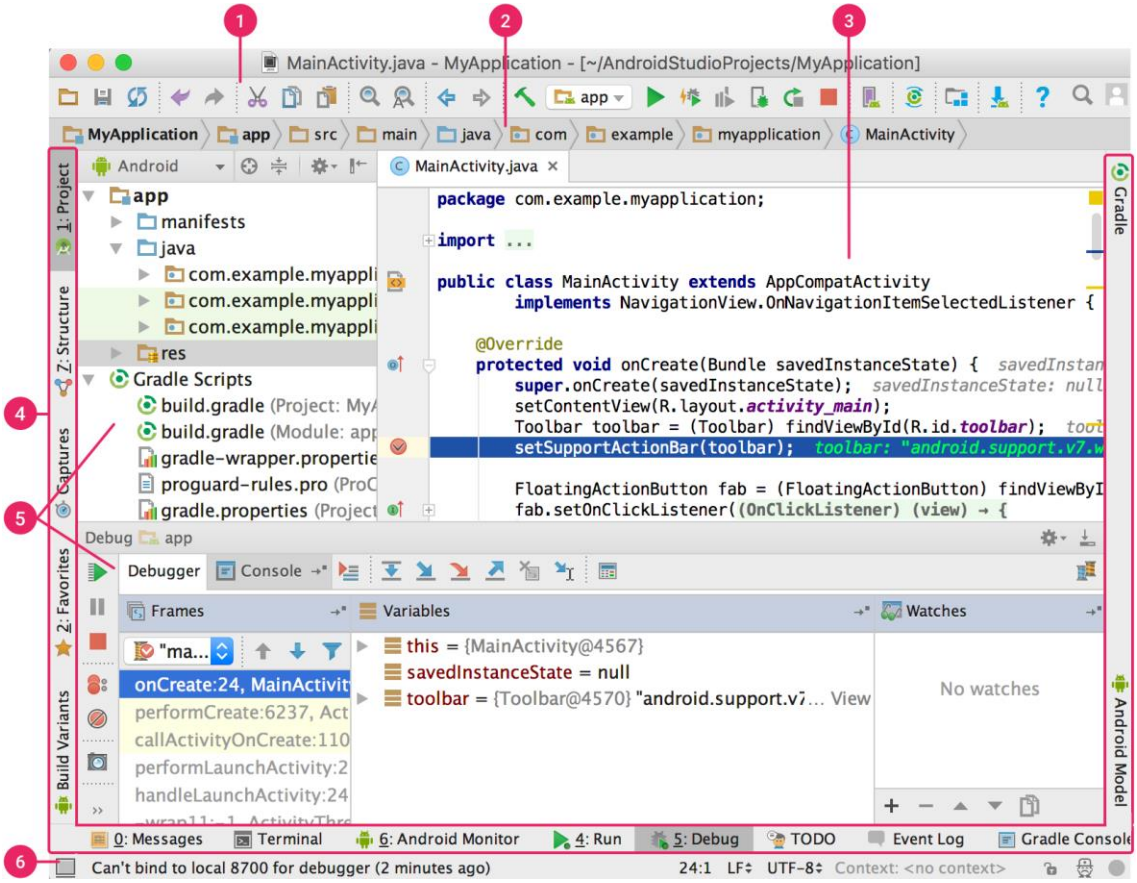

Ilustración 2 Ventana principal de Android Studio, Fuente: "https://developer.android.com"

2.3 Android Things

Es un sistema operativo creado específicamente para la utilización de sensores sobre una placa hardware. Está basado en el sistema operativo Android y su funcionamiento es muy similar con la diferencia que está diseñado para potenciar el funcionamiento de las librerías para las conexiones hardware.

La herramienta funciona de la misma manera que cuando realizas una aplicación móvil con la única diferencia de que no se crean vistas .xml.

A la hora de lanzar la aplicación podemos utilizar el emulador igual que con una aplicación móvil o bien crear un APK e instalarlo sobre el sistema operativo Android Things.

2.4 Arduino IDE

Para la programación de cualquier versión hardware de Arduino necesitamos este IDE para poder instalar el programa. Dicho IDE posee las herramientas para poder programar e instalar dicho programa para todo tipo de Arduinos y sus respectivas versiones. El lenguaje principal de programación es C, aunque posee librerías especiales para realizar acciones específicas. Es una herramienta muy potente y muy sencilla de utilizar, basta con conocer mínimamente en lenguaje de programación C.

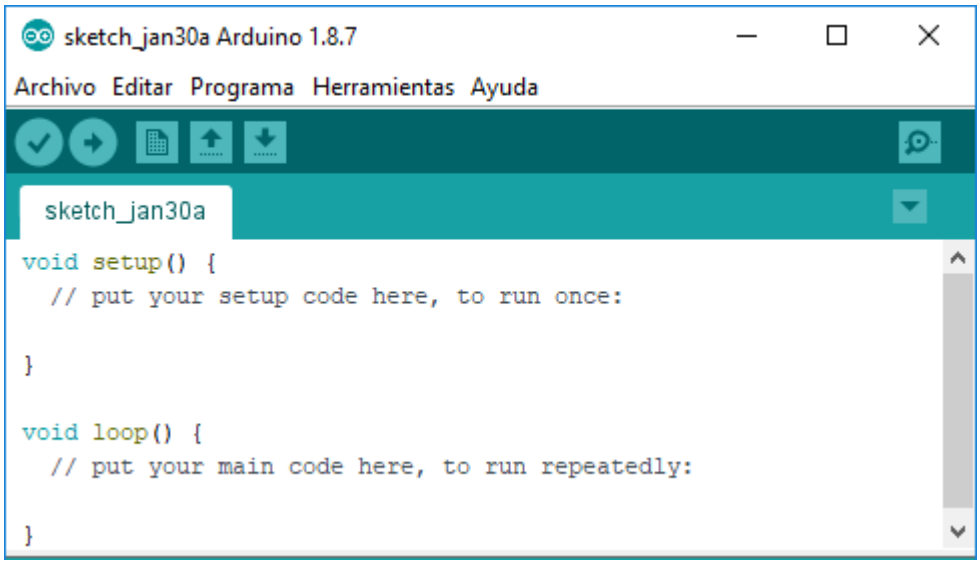

Ilustración 3 Estructura de un programa de Arduino, Fuente: "https://www.arduino.cc/"

2.5 Lenguaje de programación C

Es el lenguaje sobre el que se programa Arduino. Es un lenguaje de alto nivel pero cercano al bajo nivel. Fue diseñado para la programación de sistemas operativos pero en la actualidad se utiliza para todo tipo de software. como por ejemplo, para sistemas empotrados. Es muy útil para programar sistemas que están cercanos al hardware como es nuestro caso, ya que, una de las principales ventajas es la velocidad del mismo y la poca memoria que se necesita para poder ejecutar programas en C.

2.6 Lenguaje de programación JAVA

Es el lenguaje principal para programar sobre Android Studio. Es un lenguaje de alto nivel concurrente y orientado a objetos que necesita de una máquina virtual para ser compilado y ejecutado. Fue diseñado para que el desarrollador tuviese las menores dependencias posibles. Su gran utilidad es que una vez compilado se puede ejecutar sobre cualquier dispositivo que soporte la máquina virtual Java. Es un lenguaje bastante usado entre desarrolladores debido a la gran cantidad de librerías que posee, y que permiten no tener que implementar a mano cada operación que se quiera realizar.

2.7 MQTT

Es un protocolo de comunicaciones máquina a máquina mayormente utilizado para el Internet de las Cosas. La principal característica de este protocolo de comunicaciones es el poco consumo que produce en los dispositivos que lo utilizan. Utiliza una arquitectura de tipo estrella.

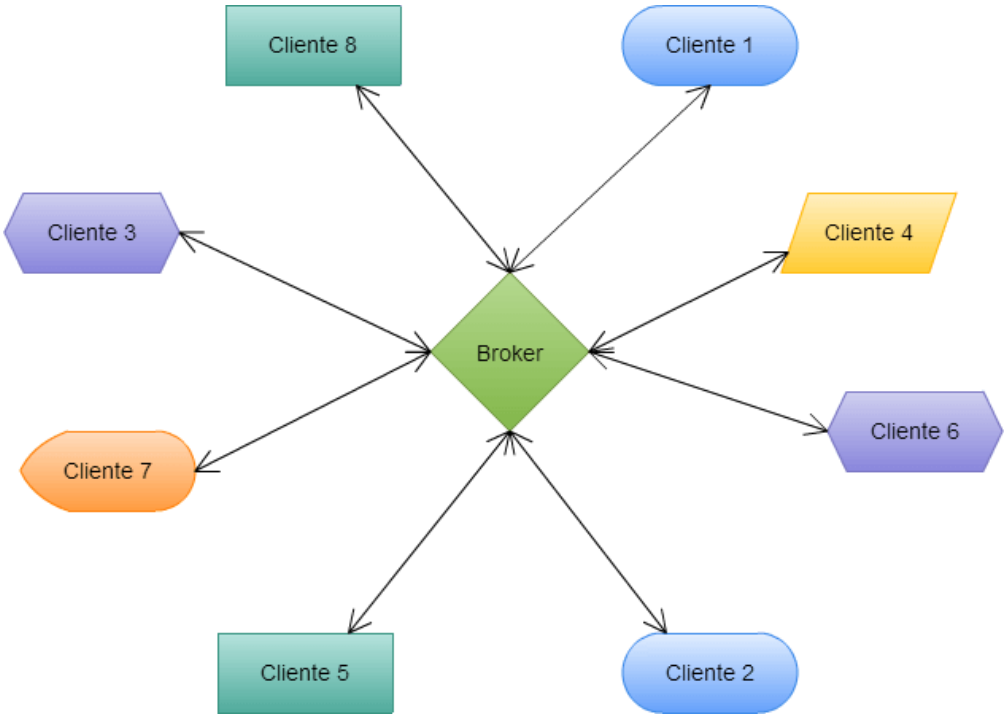

Ilustración 4 Estructura de las conexiones MQTT, Fuente: https://geekytheory.com

De tal manera que un Cliente si quiere enviar datos debe enviar a una dirección específica de MQTT denominada topic ("string/string"), y si quiere recibir datos debe subscribirse a otro topic.

3

Hardware y sensores utilizados

2.1 Raspberry Pi 3 B

Es una placa hardware que funciona como un mini ordenador. Se instala un sistema operativo sobre una tarjeta micro SD y cuando se enciende funciona como un ordenador sobre el sistema operativo instalado. Incluye entradas USB, HDMI y entrada

Ethernet, además posee WIFI integrado. La gran cualidad que posee dicha placa es que al ser un mini ordenador se puede hacer casi lo mismo que en uno normal y alojarlo donde se quiera.

Se ha elegido esta placa debido a que el sistema operativo Android Things se puede instalar sobre dicha placa añadiendo una tarjeta micro SD superior a 8 Gb.

Ilustración 5 Placa Raspberry utilizada en el proyecto, Fuente "www.raspberry.org"

2.2 Arduino Genuino Uno

Es una placa hardware que se basa en un microprocesador y el entorno de desarrollo Arduino IDE. Posee entradas analógicas y digitales integradas. Es una placa sencilla y fácil de usar para aprender y poder realizar todo tipo de actividades. Además existen todo tipos de sensores y otros materiales compatibles con dicha placa, lo que permite poder utilizar Arduino para casi cualquier funcionalidad. Es muy fácil de utilizar para comunicaciones I2C, que son necesarias para el desarrollo de este trabajo de fin de grado.

Ilustración 6 Placa Arduino utilizada en el proyecto, Fuente: "www.arduino.cc"

2.3 Arduino ESP8266

Es una placa hardware que funciona igual que Arduino y tiene integrada una placa WIFI. Necesita de una librería especial que viene integrada en la propia placa y que se descarga sobre Arduino IDE para poder funcionar. Trabaja a 3,3 V por lo que necesita sensores capaces de trabajar con esta alimentación. La principal desventaja respecto a la anterior placa es que solo posee una entrada analógica, pero la facilidad del uso de

WIFI la hace una placa muy potente y muy ágil para el desarrollo de aplicaciones que necesiten Internet.

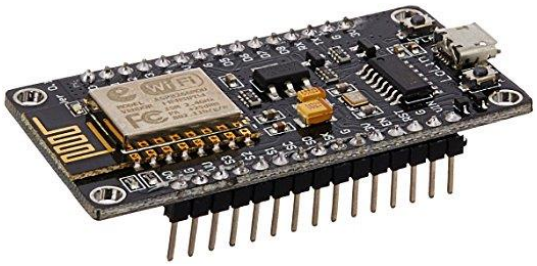

Ilustración 7 Placa Arduino con Wifi utilizada en el proyecto, Fuente: "www.amazon.com"

2.4 Sensor de temperatura y humedad DHT11

Es un sensor que mide tanto la temperatura como la humedad. Puede utilizarse con un voltaje de 3.3 o 5 V pero el fabricante recomienda 5 V debido a que las caídas de tensión pueden producir un mal funcionamiento del sensor. La característica principal de este sensor es que es digital lo que facilita su implantación en sistemas que no posean entradas analógicas.

Ilustración 8 Sensor de temperatura y humedad DHT11, Fuente: "www.amazon.com"

2.5 Sensor de temperatura DS18B20

Es un sensor de temperatura muy útil para sistemas con pocos pines analógicos, ya que es digital. Sirve tanto para temperatura ambiente como para líquidos. Este sensor trabaja entre 3 y 5 V lo cual es perfecto para el Arduino ESP8266 que trabaja a 3,3 V. Puede medir temperaturas de entre -55ºC y 125ºC debido a que es especial para líquidos.

La petición del valor del sensor se realiza mediante el bus 1-Wire, debido a que internamente posee un pequeño control de errores. De esta manera el sensor es más fiable pero antes de leer un valor indicar mediante este bus necesita una indicación de que se va a realizar una petición de datos.

Ilustración 9 Sensor de temperatura DS18BS20, Fuente"ww.amazon.com"

2.6 Fotorresistor

Es un sensor que se basa en un resistencia que al incrementar la misma la intensidad disminuye. Se lee por lo tanto desde un pin analógico esta variación. Este sensor es muy efectivo si la variación de luz es grande y se hace de manera progresiva, ya que, si la variación de luz es muy rápida el sensor no es capaz de detectarlo. Una de las ventajas o desventajas de este sensor es que es necesario inclinar el sensor por el que se detecta la luz hacia el foco luminoso para notar la variación en la luminosidad.

Ilustración 10 Fotorresitor, Fuente"ww.amazon.com"

2.7 Led

Un led es un diodo que emite luz usando una unión p-n. De esta manera es muy sencillo general luz con muy poca intensidad. Es bastante eficiente para dispositivos con poco voltaje o que necesiten poco consumo.

Ilustración 11 LEDs para Arduino y Raspberry, Fuente"www.amazon.com"

2.8 Ventilador L9110

Este ventilador está especialmente diseñado para ser insertado sobre un Arduino gracias al poco voltaje que necesita para hacer girar las hélices con potencia. Necesita 5V para funcionar a pleno rendimiento pero sí se le suministra menos alimentación lo único que ocurre es que las hélices giran con menos potencia. Según el fabricante puedes poner un objeto hasta a 20cm de distancia del ventilador.

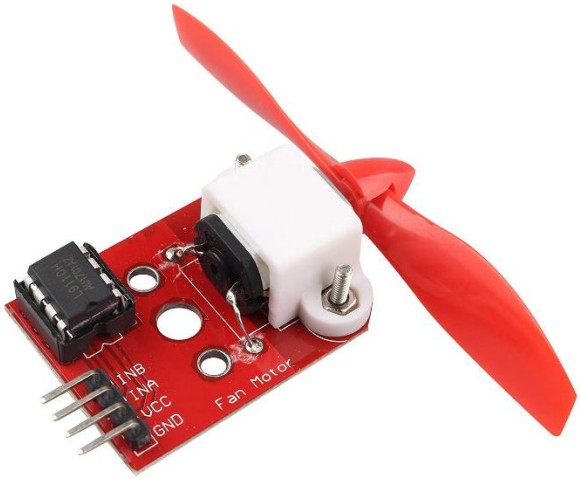

Ilustración 12 Ventilador L9110, Fuente"www.amazon.com"

4

Documento general de requisitos

4.1 Objetivos

El principal objetivo es conocer más en profundidad cómo funciona el Internet de las Cosas mediante sensores alojados en dispositivos hardware.

Como objetivo destacado se aprenderá el funcionamiento del nuevo sistema operativo para IoT de Google Android Things. Además, se utilizará la placa Arduino para conocer más en profundidad esta plataforma. Se llevará acabo la lectura de distintos sensores a través de estos dos dispositivos hardware y de esta manera conocer las ventajas y las limitaciones de estos potentes sistemas.

También se realizará una aplicación móvil Android para visualizar los datos aprendiendo por tanto a realizar visualizaciones gráficas y dinámicas.

4.2 Alcance

El sistema estará limitado por una aplicación móvil Android desde donde el usuario podrá acceder a los datos de los sensores. Por lo tanto, el usuario no podrá realizar nada más allá de lo que le proporcione la aplicación.

4.3 Directivas de negocio

4.3.1 Descripción del problema

Se realizará el trabajo de fin de grado y para ello se necesita una aplicación domótica IoT donde se controlen los sistemas hardware mediante una aplicación móvil Android.

La principal dificultad de esta aplicación radica en el sistema operativo Android Things alojado en una Raspberry, debido a la poca información que existe para

desarrollar dicha aplicación IoT con sensores analógicos y controlados desde este sistema operativo.

Se deberá realizar la lectura y el envío correcto de los datos de los sensores desde Arduino y desde Android Things. Se utilizará el protocolo de comunicaciones, bastante utilizado para IoT, MQTT. La dificultad radicará en poder insertar este protocolo tanto en los dos tipos de dispositivos hardware como en la aplicación móvil Android, de tal manera que puedan comunicarse los sistemas entre sí.

Por último, la aplicación móvil Android será la encargada de pedir datos de los sensores en tiempo real y mostrarlos gráficamente para poder observar los datos de una manera más sencilla y vistosa.

4.3.2 Descripción del producto

El producto final está orientado al aprendizaje y a la realización del trabajo de fin de grado. Con la aplicación se conseguirá una nueva manera de comunicación IoT y conocer las ventajas y los inconvenientes del sistema operativo Android Things.

El producto final será una aplicación móvil capaz de obtener datos de las placas hardware. Además se proporcionará los códigos de los programas de los sistemas hardware y de la aplicación móvil.

4.4 Descripción de los usuarios

Cualquier persona que posea un móvil Android con versiones iguales o superiores a 4.4 Kit Kat podrá usar el sistema y comunicarse con los dispositivos IoT a través de la aplicación. El sistema está orientado a un uso doméstico y será usado por los habitantes del domicilio, garaje, fábrica, hangar o similar donde se instalen las placas hardware.

4.5 Entorno de despliegue

4.5.1 Entorno para la implementación del sistema actual

Se utilizará un Arduino ESP8266 programados en C, una Raspberry Pi B 3 en la que se instalará Android Things programado en Java conectada por I2C con un Arduino R3 Genuino Uno, y una aplicación móvil Android.

4.5.2 Aplicaciones colaboradoras

Se utilizará HiveMQ, Broker MQTT, para la comunicación entre los distintos sistemas externo al producto.

4.5.3 Paquetes comerciales

Los frameworks Android Studio y Arduino IDE.

Se utilizará una salida a internet mediante Wifi o Ethernet y el protocolo de comunicaciones máquina a máquina MQTT.

Las librerías que necesiten los distintos dispositivos IoT que se utilizarán para la aplicación.

4.6 Suposiciones y dependencias

4.6.1 Factores externos que tienen un efecto en el producto, pero no son restricciones obligatorias

Compatibilidad con los distintos sistemas operativos tanto de los sensores como de las tecnologías.

4.6.2 Suposiciones que asume el desarrollador entorno al proyecto

Correcto funcionamiento del bróker MQTT que se utilizará para las comunicaciones.

La conexión a internet vía Wifi será correcta a través del Router para los dispositivos hardware.

Los datos de los sensores serán considerarán correctos si el margen de error entre sensores se considera probable. Es decir, se considerará incorrecto un valor de un sensor de temperatura de 85 grados, pero no una diferencia de 23 a 26 grados.

4.7 Precio y coste

La compra de los materiales no deberá superar los 60 euros que se ofrecen en la beca universitaria. La aplicación móvil será gratis.

4.8 Licencias e instalación

La instalación de la aplicación móvil se realizará a través de un APK en dispositivos Android o bien a través de la plataforma Google Play con la correspondiente licencia de dicha plataforma.

La instalación de los sensores se realizará según la información proporcionada por el fabricante.

4.9 Requisitos funcionales

- Se medirán sensores de temperatura, humedad y luminosidad tanto desde la placa Arduino como de la placa Raspberry.
- El usuario podrá encender y apagar los leds situados en la placa Arduino y en la placa Raspberry desde la aplicación móvil.
- El usuario podrá encender y apagar los ventiladores situados en la placa Arduino y en la placa Raspberry desde la aplicación móvil.

- El usuario podrá visualizar los datos de temperatura, luminosidad, humedad de los distintos dispositivos alojados en distintos puntos de la casa.
- El usuario podrá pedir datos en tiempo real mediante botones de temperatura, luminosidad, humedad de los distintos dispositivos.
- El usuario podrá visualizar datos históricos de las mediciones de temperatura realizadas por el mismo.
- El usuario podrá establecer que a una temperatura determinada el ventilador se mantenga encendido hasta que alcance la temperatura determinada por el usuario.
- El usuario podrá establecer que a una temperatura determinada el ventilador se mantenga apagado hasta que alcance la temperatura determinada por el usuario.

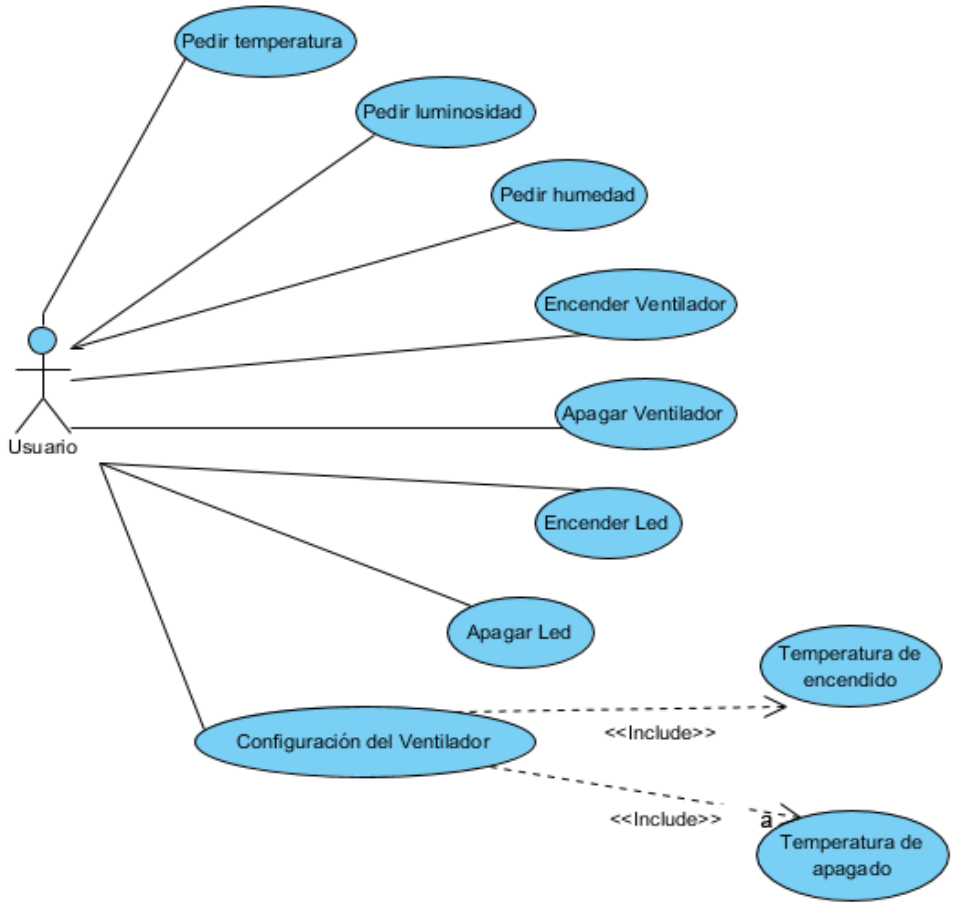

Ilustración 13 Diagrama de casos de uso

4.10 Precedencia y prioridades

Basándose en que la realización del trabajo de fin de grado requiere de un estudio previo de Android Things, lo primero será el aprendizaje de dicha herramienta.

En segundo lugar, es necesario comunicar mediante MQTT los dispositivos hardware con la aplicación móvil Android, ya que sin esto el sistema no funcionaría.

Por último, la compra de materiales y de sensores, y de la comprobación de que funcionan en los dispositivos hardware será crucial para el desarrollo del trabajo.

4.11 Requisitos no funcionales

- Los dispositivos hardware deberán conectarse a Internet.
- Los dispositivos hardware se conectarán a HiveMQ (Broker MQTT).
- Los dispositivos hardware recibirán datos de la aplicación móvil por el protocolo de comunicaciones MQTT.
- Los dispositivos hardware enviarán datos de los sensores a la aplicación móvil por el protocolo de comunicaciones MQTT.

4.11.1 La interfaz

- La aplicación móvil deberá proporcionar gráficas donde se puedan visualizar los datos de los sensores.
- La aplicación móvil tendrá varias pantallas para que el usuario elija que sensor quiere ver.
- El nombre de la aplicación será AplicacionIoTDomotica.
- Tendrá varias ventanas para visualizar los distintos sensores
- La aplicación móvil deber tener claros accesos a los datos de los sensores

4.11.2 Estilo de producto

La aplicación móvil tendrá colores sencillos y un diseño que destaque la funcionalidad por encima de la vista. Debe ser intuitiva y en un lenguaje claro.

4.11.3 Requisitos de velocidad

La comunicación de la aplicación móvil con los dispositivos hardware dependerá de la comunicación MQTT y de la velocidad del bróker MQTT. La velocidad de respuesta de la aplicación móvil respecto a la petición de un usuario será adecuada para que el usuario no espere más de 1 segundo.

4.11.4 Requisitos de fiabilidad y disponibilidad

Los datos solo estarán disponibles cuando los dispositivos hardware estén conectados.

4.11.5 Requisitos de capacidad

Solo la aplicación móvil almacenará datos. Se almacenarán solo los datos de sensores de temperatura.

4.11.6 Entorno físico

La aplicación móvil irá alojada en un móvil Android. El sistema operativo Android Things irá alojado en una Raspberry Pi B 3. Por último, se utilizará un Arduino ESP8266.

4.11.7 Soporte

La aplicación móvil estará limitada por la versión del sistema operativo actual. Los dispositivos hardware serán programados para ser automáticos de tal manera que el usuario no pueda modificar nada de los mismos.

4.11.8 Dificultad para el mantenimiento del producto

Para cualquier cambio en la aplicación móvil se deberá evaluar el tiempo de realización del cambio. Una vez evaluado se actualizará la aplicación.

Para un cambio en los dispositivos hardware, si es de software se evaluará el tiempo de realización y se actualizará el código, si es hardware se tendrá que evaluar el tiempo de instalación y de programación.

4.11.9 Requisitos de portabilidad

La aplicación móvil solo se podrá instalar en otros dispositivos Android con versión de sistema operativo igual o superior a Android 4.4 Kit Kat.

4.11.10 Fiabilidad del sistema

Cualquier usuario podrá instalar la aplicación móvil y usarla.
Solo el desarrollador de la aplicación podrá realizar modificaciones en la misma.

4.11.11 Requisitos de Integridad de los datos

Se guardarán los datos de la temperatura de manera que el usuario pueda obtener cuando lo solicite.

5

Diseño e implementación

Tras la elección del tema del trabajo de fin de grado, se eligió el método para la realización de las diferentes fases del desarrollo. Se eligió la metodología incremental. De esta manera se han ido implementando los distintos sistemas de manera incremental, se empezó por implementar sistemas sencillos en el hardware, pasando por la implementación de las comunicaciones y terminando por la aplicación móvil completa.

5.1 Arduino ESP8266

5.1.1 Diseño del hardware

La conexión de los sensores se ha realizado según la información de los fabricantes de cada sensor quedando de la siguiente manera:

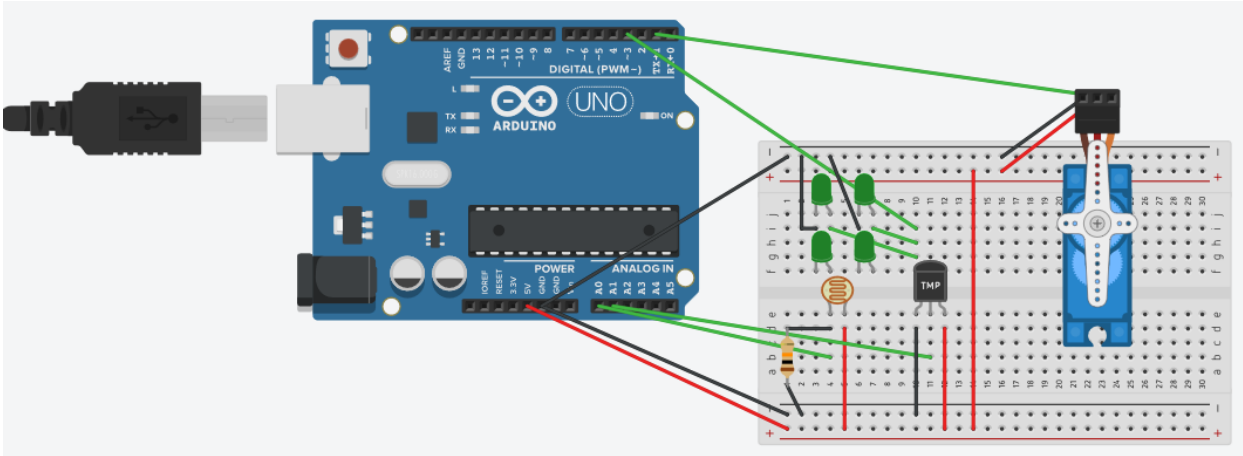

Ilustración 14 Diseño de la instalación de los sensores

Los sensores utilizados en el dispositivo y usados para la realización del diseño han sido: DS18B20, Fotorresistencia, Leds y Ventilador L9110. Como placa principal para este diseño se ha utilizado la placa hardware Arduino ESP8266 con Wifi integrado.

5.1.2 Implementación

El funcionamiento de dicho sistema se separa en tres funcionalidades, la conexión Wifi, el protocolo MQTT y la lectura de los sensores.

La conexión Wifi se basa en un SSID y en una Contraseña preestablecida. El sistema se intenta conectar y si no lo consigue espera 5 segundos y lo vuelve a intentar. No se contempla un proceso sin conexión Wifi debido a que es necesaria la conexión a Internet para el funcionamiento del dispositivo.

```
void configurarWifi() {
  delay(100);
  Serial.println();
  Serial.print("Intento de conexión Wifi a: ");
  Serial.println(ssid);

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

La conexión MQTT se realiza de forma parecida a Wifi. Se establece la conexión mediante un nombre de Cliente a un Servidor MQTT preestablecido, y posteriormente se establece la subscrición a los topic. Cada cierto periodo establecido se comprueba si está conectado y en caso de no estarlo se vuelve a reconectar con el Servidor.

```
void reconnect() {
  while (!client.connected()) {
    Serial.print("MQTT -> Intentando la conexion...");
    if (client.connect(nombreClienteMqtt)) {
      Serial.println("Conectado");
      client.subscribe(subscripcion);
      client.subscribe(subscripcionConfiguracionMaxima);
      client.subscribe(subscripcionConfiguracionMinima);
```

Una vez establecida la conexión MQTT se establece una función a la cual se llamará cuando llegue algún mensaje al topic subscrito. Esta función es la encargada de comprobar y leer el mensaje, y actuar en consecuencia o de averiguar cuál es el topic que se ha detectado y actuar.

```
client.setServer(servidorMqtt, 1883);
client.setCallback(llegadaDeMensaje);
```

Si el mensaje indica la petición del valor de un sensor se enviará dicho valor por otro topic al que esta subscrita la aplicación móvil. Si indica una acción sobre un sensor de escritura, ventilador o led, se realizará la acción indicada que en este caso es encender o apagar el sensor.

```
void llegadaDeMensaje(char* topic, byte* mensaje, unsigned int length) {
  Serial.print("Mensaje recibido: ");
  for (int i = 0 ; i < length ; i++) {
    Serial.print((char)mensaje[i]);
  }
```

Por último, el sistema se ha subscrito a dos topic por los que se recibirán la temperatura por la que el ventilador deberá estar siempre encendido hasta que baje, y la temperatura mínima por la que el ventilador deberá estar siempre apagado hasta que suba. Cuando se establezca por la aplicación móvil los datos recibidos se actualizarán sobre el sistema. De esta manera habrá una función que esté constantemente comprobando lo anterior. La implementación de dicha funcionalidad se ha realizado utilizando el siguiente diagrama de estado:

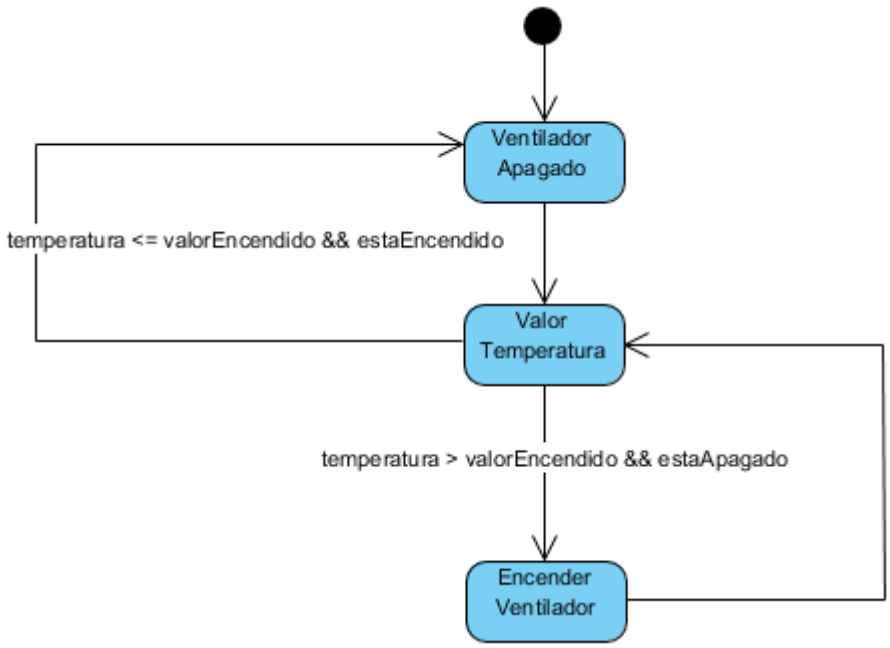

Ilustración 15 Diagrama de estados para la función controladora del ventilador

5.2 Raspberry con Android Things

5.2.1 Diseño del hardware

Debido a que desde la placa Raspberry no se pueden leer sensores analógicos se ha utilizado un Arduino conectado mediante el protocolo I2C a la Raspberry. Además ciertos sensores digitales requieren de librerías especiales por lo que se ha realizado con ellos el mismo procedimiento. De esta manera se ha realizado el diseño de la placa quedando de la siguiente manera:

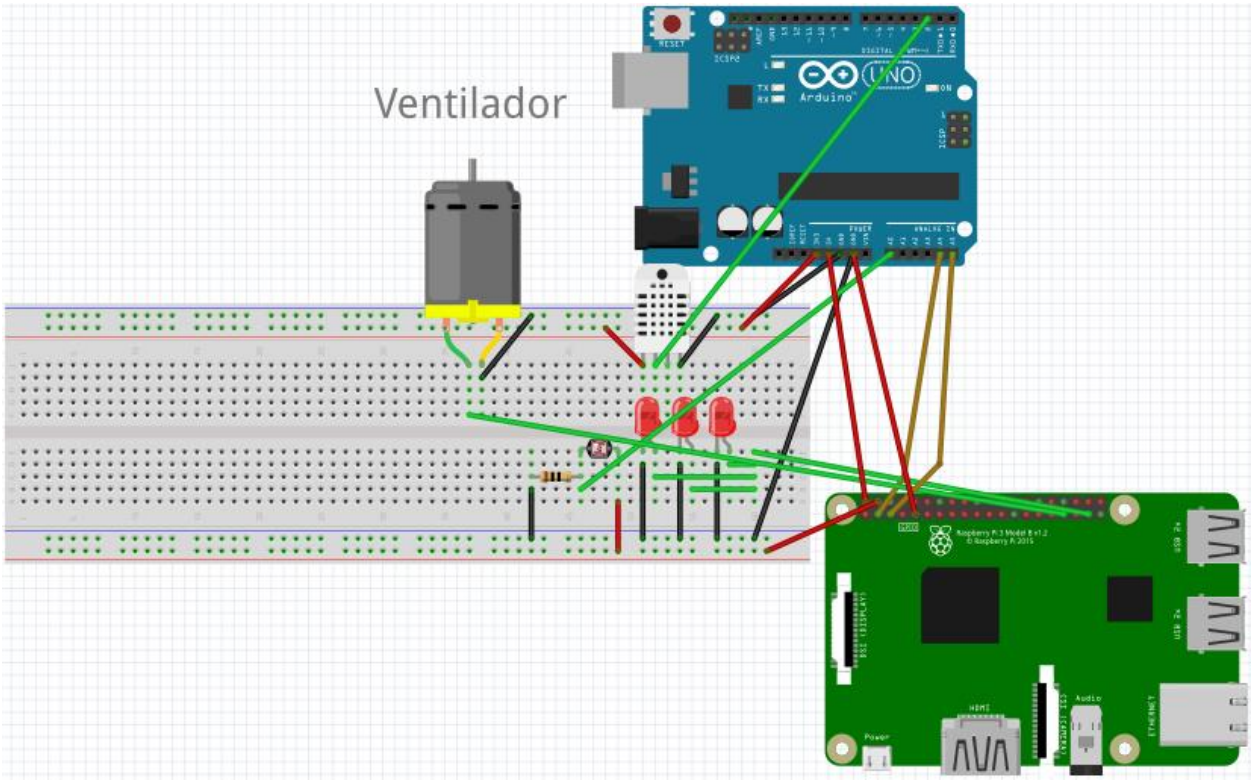

Ilustración 16 Diseño de la instalación de los sensores y la comunicación I2C

Los sensores utilizados en el dispositivo y usados para la realización del diseño han sido: DHT11, Fotorresistencia, Leds y Ventilador L9110. Como placa principal para este diseño se ha utilizado la placa hardware Raspberry Pi 3 B conectada con Arduino Genuino Uno.

5.2.2 Implementación

El funcionamiento de dicho sistema es muy parecido al anterior de Arduino pero está realizado en Java y basado en el sistema Android para la realización de aplicaciones. La Raspberry se conecta mediante Ethernet a internet por lo que no es necesaria la conexión vía Wifi.

La conexión MQTT se realiza mediante el patrón Singleton utilizando la librería "org.eclipse.paho" para realizar dicha conexión sobre un broker MQTT. Además se ha añadido un método conectar() porque si durante un tiempo no se ha realizado ninguna acción sobre el Cliente MQTT se desconecta del broker para realizar publicaciones a un topic determinado. De esta manera cada vez que vamos a publicar sobre un topic primero llamamos a dicha función y luego publicamos.

```
public static MQTTConnection getMQTTConnection(Context context) {
    if (mqttConnection == null) {
        return new MQTTConnection(context);
    } else {
        return mqttConnection;
    }
}
```

```
public boolean conectar() {
    if (client == null || !client.isConnected()) {
        client = new MqttAndroidClient(context, Constantes.URI,
                clientId);
    }
    subcribirse();
    return client.isConnected();
}
```

La lectura de los sensores de temperatura, humedad y luminosidad se realizan por parte de Arduino pero se piden por parte de la Raspberry. Cuando llega una petición de un sensor por parte de la aplicación Móvil se utiliza la comunicación por el PIN I2C. La placa Arduino está programada para que se ejecute una función que lee todos los sensores conectados al mismo y los envía en un buffer como un String hacia la Raspberry.

```
public void leerSensores() throws IOException {
    byte[] buffer = new byte[14];
    puerto.readRegBuffer(direccion, buffer, buffer.length);
    String datosString = new String(buffer);
```

Una vez establecida la conexión MQTT se establece una función a la cual se llamará cuando llegue algún mensaje al topic subscrito. Esta función es la encargada de leer el mensaje y actuar en consecuencia o de averiguar cuál es el topic que se ha detectado y actuar.

```
mqttConnection.getClient().setCallback(new MqttCallback() {
    @Override
    public void connectionLost(Throwable cause) { mqttConnection.conectar(); }

    @Override
    public void messageArrived(String topic, MqttMessage message) throws Exception
        String mensaje = new String(message.getPayload());
        Log.d(TAG,  msg: "" + mensaje);
        switch (mensaje) {
```

Si el mensaje indica la petición del valor de un sensor se enviará dicho valor por otro topic al que esta subscrita la aplicación móvil. Si indica una acción sobre un sensor de escritura, ventilador o led se realizará la acción y se enviará un mensaje de correcta realización de la acción a la aplicación.

```
//Encender Led
case Constantes.MENSAJE_LUMINOSIDAD_ENCENDER_LED_THINGS:
    sensores.setLed(true);
    break;
//Apagar Led
case Constantes.MENSAJE_LUMINOSIDAD_APAGAR_LED_THINGS:
    sensores.setLed(false);
    break;
```

Por último, el sistema se ha subscrito a dos topic por los que se recibirán la temperatura por la que el ventilador deberá estar siempre encendido hasta que baje, y la temperatura mínima por la que el ventilador deberá estar siempre apagado hasta que suba. Cuando se establezca por la aplicación móvil los datos recibidos se actualizarán sobre el sistema. De esta manera habrá una función que esté constantemente comprobando lo anterior. La implementación de dicha función se ha realizado utilizando el siguiente diagrama de estado:

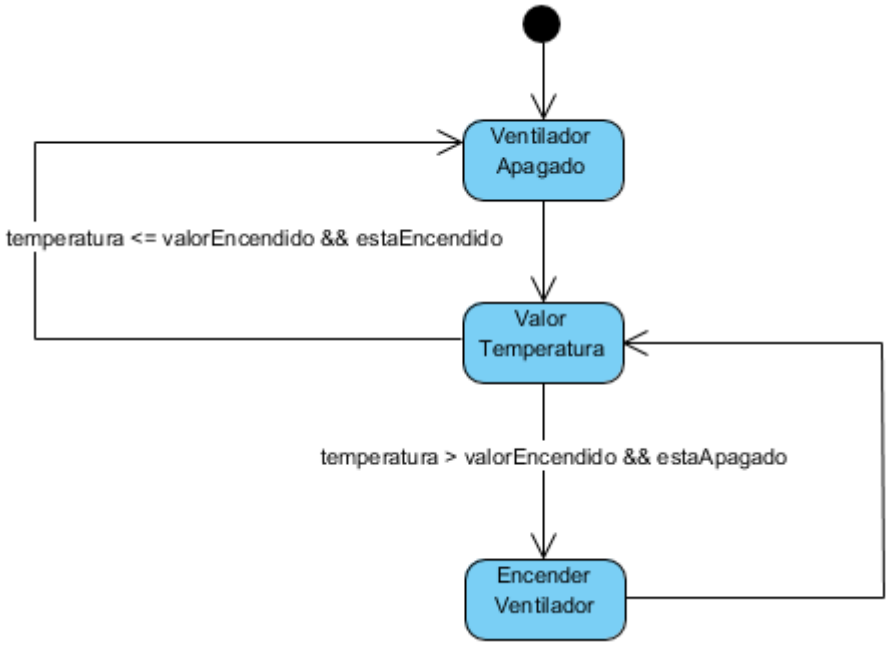

Ilustración 17 Diagrama de estados de la función controladora del ventilador.

5.3 Aplicación móvil

5.3.1 Prototipo de la interfaz

Utilizando lo estipulado en los requisitos se ha realizado un diseño tipo boceto de la interfaz de la aplicación móvil.

La aplicación móvil está separada en diferentes pantallas independientes entre ellas salvo la pantalla inicial de la aplicación desde la cual se accede al resto como se puede ver en la *Ilustración 18*. En la *Ilustración 19* podemos ver la pantalla para poder configurar la temperatura a la que el ventilador se encenderá y/o apagará automáticamente.

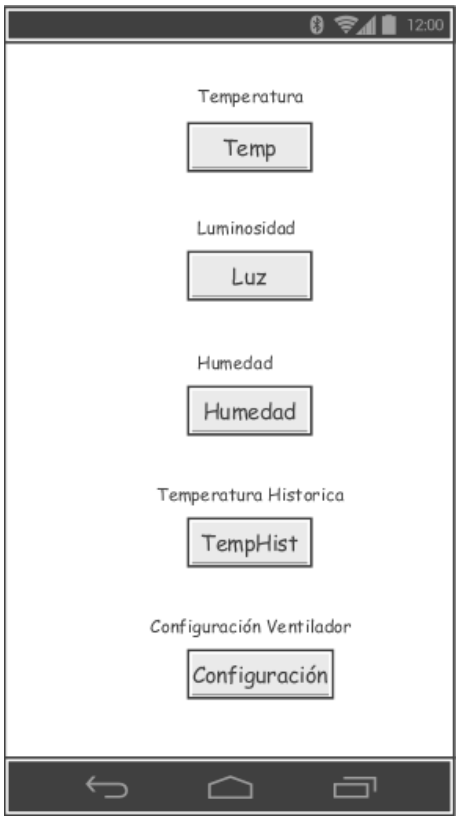

Ilustración 18 Pantalla de inicio

Ilustración 19 Pantalla de Configuración del ventilador

Sobre el sensor de temperatura se muestran los datos en dos pantallas. La *Ilustración 20* se puede observar la pantalla principal para mostrar los datos de la temperatura en tiempo real y para encender o apagar el ventilador. En la *Ilustración 21* se pueden ver los datos de las temperaturas que se han ido midiendo por días.

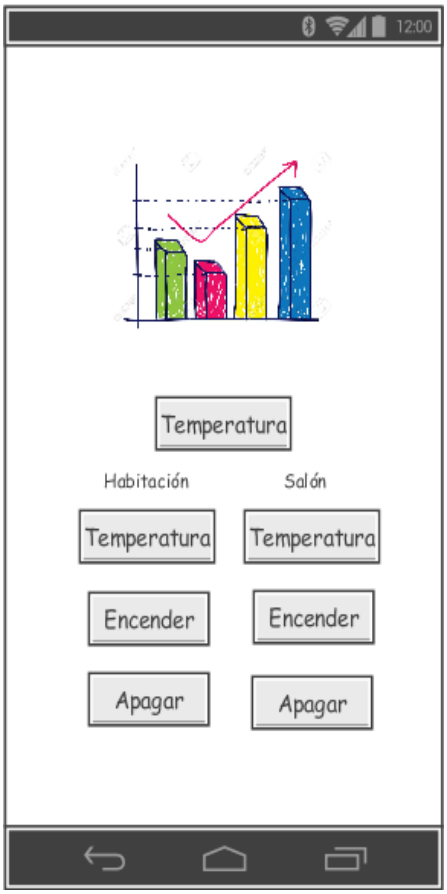

Ilustración 20
Pantalla temperatura

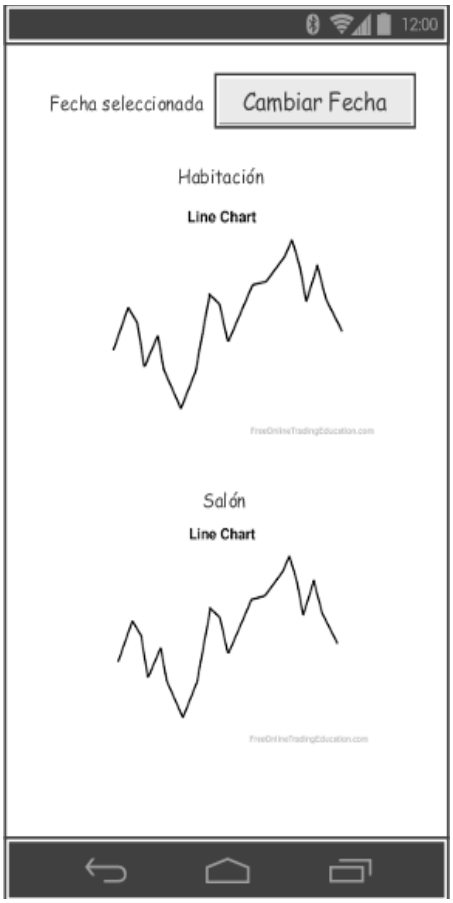

Ilustración 21 Pantalla temperatura histórica

Respecto a la humedad al solo poseer un sensor para la misma se ha diseñado una sola pantalla para dicho sensor, como se muestra en la *Ilustración 22*. Respecto a la luminosidad se mostrará con una gráfica y a la vez se podrá tanto encender los Leds como apagarlos (*Ilustración 23)*.

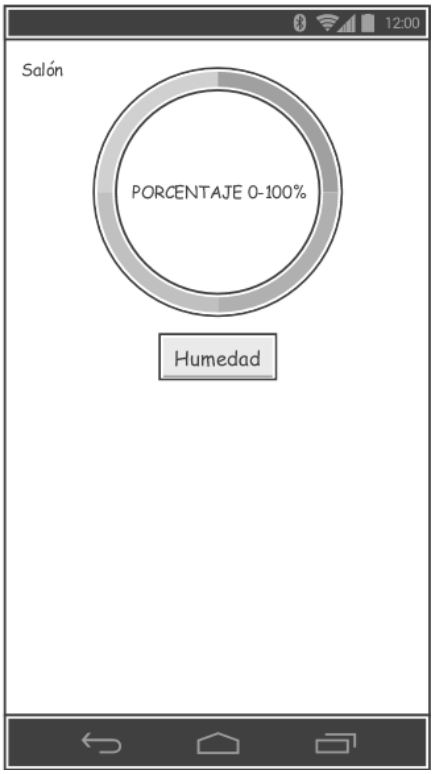

Ilustración 22 Pantalla del sensor de humedad

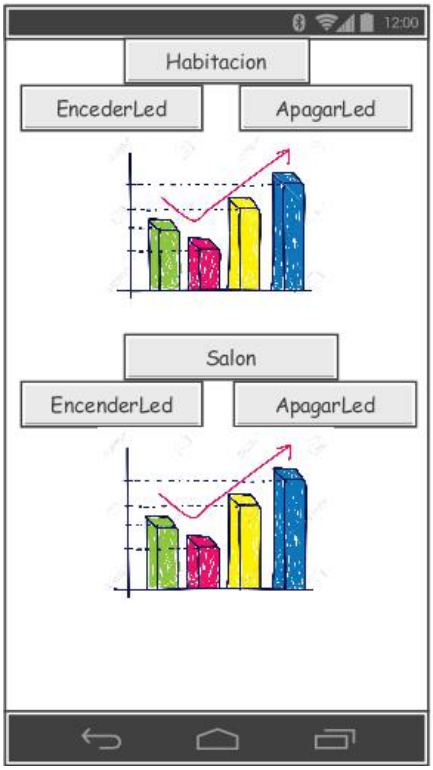

Ilustración 23 Pantalla de luminosidad y de los LEDs

5.3.2 Diseño de la interfaz

Cuando se lanza la aplicación la pantalla principal de la aplicación (*Ilustración 24)* muestra diferentes botones con nombres clave para acceder a la información de los sensores en función de la acción que quiera realizar el usuario.

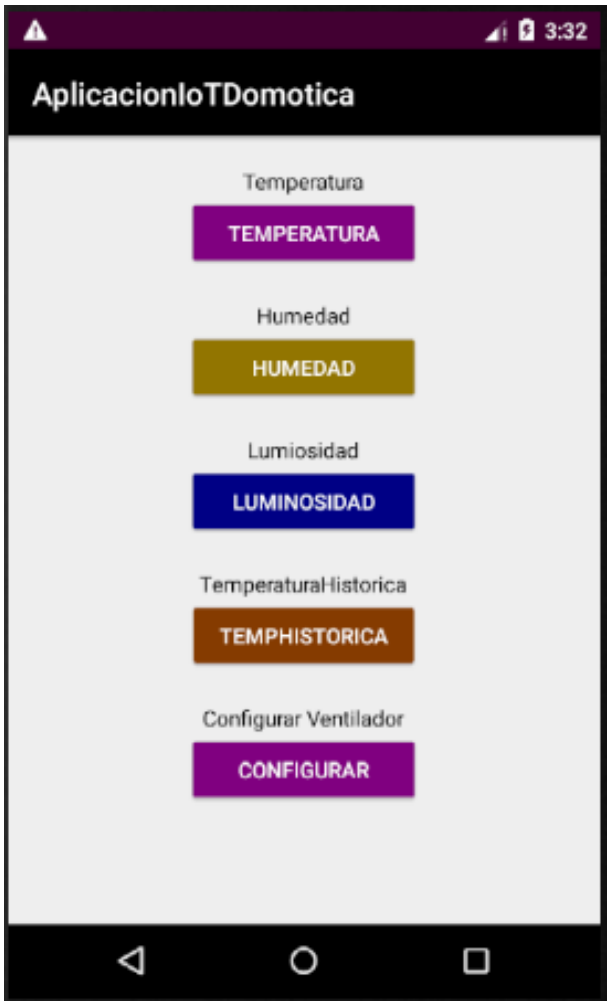

Ilustración 24 Pantalla principal de la aplicación

Para actuar directamente sobre los sensores es necesaria la conexión a Internet, ya que, la conexión MQTT necesita de la misma. Para ello se ha añadido una alerta para indicar que es necesaria la conexión a Internet, tanto al abrir la aplicación como cuando se abre una pantalla para la actuación directa y en tiempo real sobre los sensores (*Ilustración 25)*.

Ilustración 25 Alerta de que no hay conexión a Internet.

Al seleccionar sobre temperatura se mostrará una pantalla donde se mostrará la temperatura (*Ilustración 26)* de los dos lugares de la casa donde están alojadas las placas hardware. Dentro de esta pantalla podrá pedir la temperatura de los dos sensores a la vez o de cada uno por separado pudiendo así comparar las temperaturas de los dos lugares.

Además dentro de esta pantalla hay dos botones para cada habitación donde podrán tanto encender como apagar un ventilador para poder bajar la temperatura de la habitación. Cuando se realice una de las acciones anteriores los dispositivos hardware enviarán un mensaje por un topic determinado indicando que se ha realizado con éxito, cuando la aplicación lo recibe muestra un mensaje temporal por pantalla indicando la acción que se ha realizado con éxito.

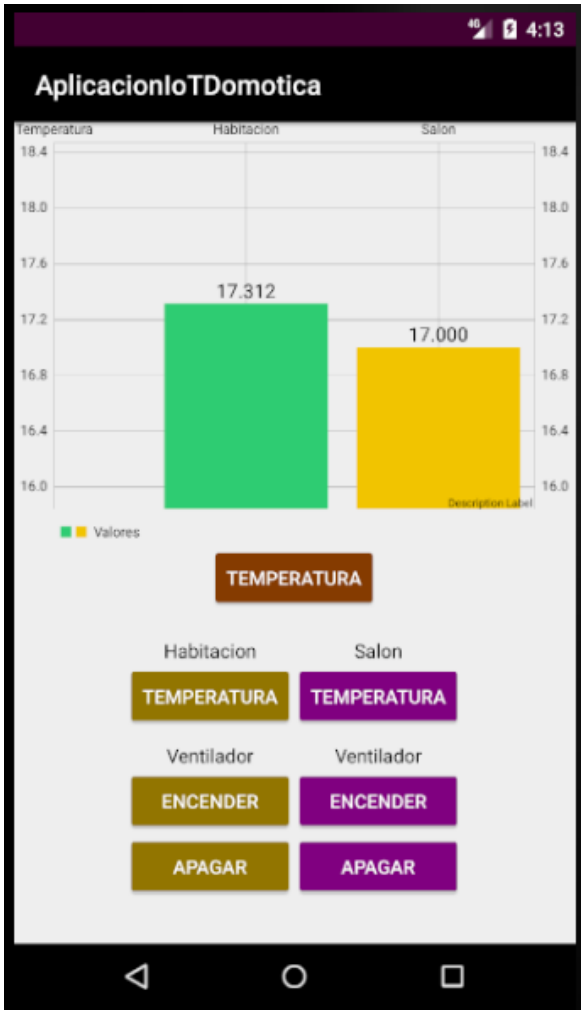

Ilustración 26 Pantalla de la temperatura de la aplicación móvil

El usuario tendrá la opción para poder ver la temperatura que se ha ido obteniendo cada vez que clicaba sobre la ventana de temperatura. Para ello cuando clica en temperatura histórica se mostrará dicha información (*Ilustración 27)*.

Primero se mostrará una ventana con un calendario para que el usuario elija el día que desea ver y posteriormente se mostrarán los datos de temperatura que se tomaron dicho día para cada uno de los sensores (*Ilustración 28)*. Además el usuario podrá elegir otro día clicando sobre el botón "cambiar fecha".

Ilustración 27 Pantalla elección del día

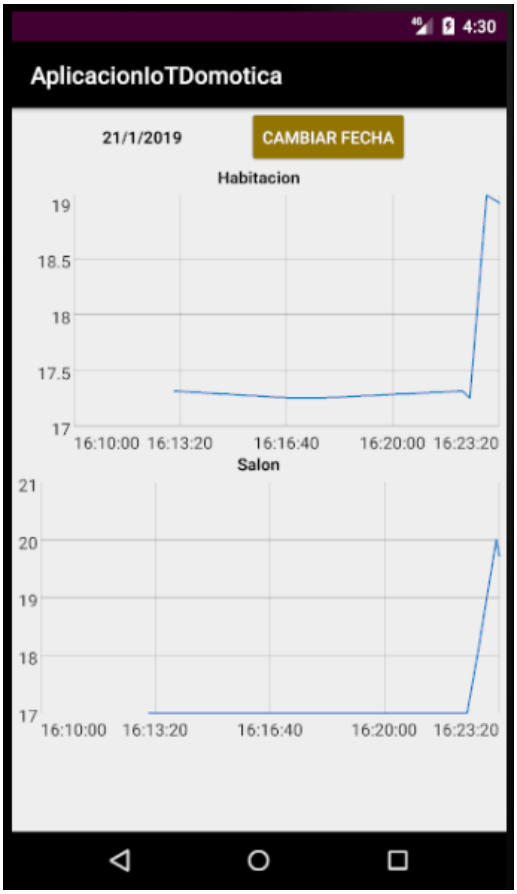

Ilustración 28 Pantalla de temperatura histórica

Si el usuario selecciona sobre la sección de humedad podrá visualizar la humedad que hay en la habitación donde está alojada la placa Raspberry con tan solo clicar en el botón "humedad", como muestra la *Ilustración 29*.

También si el usuario quiere saber la cantidad de luz que hay en una habitación puede saberlo con solo clicar sobre el botón de la habitación que quiere (*Ilustración 30)*. Además puede encender o apagar las luces alojadas en cada dispositivo clicando sobre los botones de encender o apagar. Cuando se realice la acción sobre el dispositivo este emitirá un mensaje de que se ha realizado correctamente, cuando la aplicación reciba el mensaje mostrará por pantalla un mensaje temporal indicando la acción que se ha efectuado con éxito.

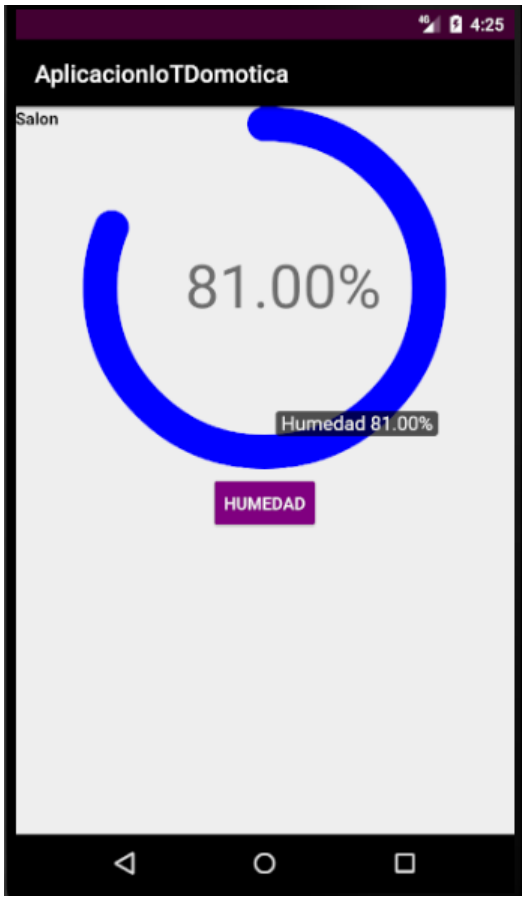

Ilustración 29 Pantalla de humedad

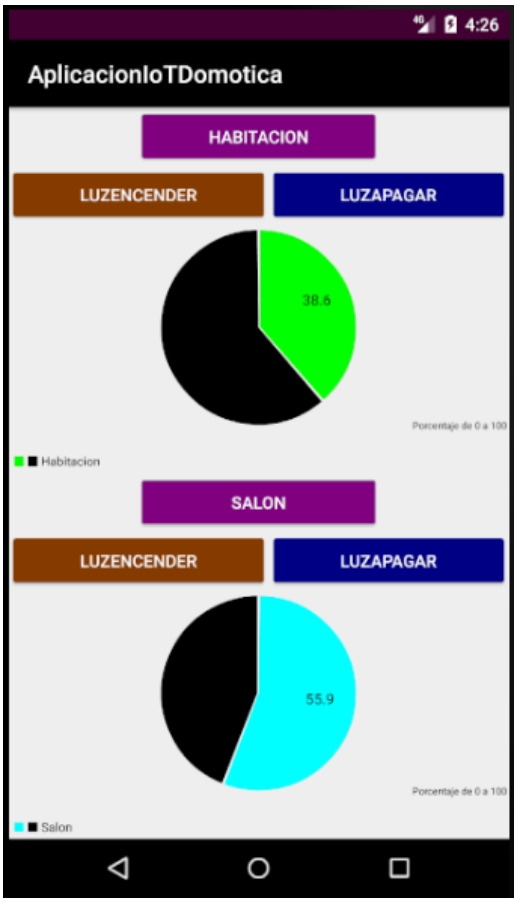

Ilustración 30 Pantalla de luminosidad y LEDs

Por último, el usuario podrá configurar para cada ventilador la temperatura a la que se encenderá automáticamente o la temperatura a la que se mantendrá apagado siempre (*Ilustración 31)*. Para saber si se ha realizado la operación correctamente el dispositivo correspondiente mandará un mensaje a la aplicación si se ha realizado con éxito, aplicación mostrará por tanto que se ha realizado la configuración correctamente.

Ilustración 31 Pantalla de configuración del ventilador

5.3.3 Implementación

Empezaremos hablando de las vistas. Estás se crean como un fichero .xml y se asocian a una clase Java, de tal manera que cuando nosotros lanzamos esa clase automáticamente se lanza la vista asociada a la misma.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_luminosidad);
```

Por otra parte, tanto la petición de los valores de los sensores como la acción sobre los mismos se realiza mediante el protocolo MQTT explicado anteriormente. Se utiliza el patrón Singleton para crear una sola conexión MQTT sobre un broker, esto es debido a que se utiliza la librería *paho-mqtt* proporcionada por eclipse y solo permite tener un Cliente por aplicación. Pero para poder utilizar dicho protocolo es necesaria la

conexión a Internet, por ello como se ha mostrado anteriormente se realiza una comprobación sobre la conexión y se informa al usuario si no existe dicha conexión.

Además, cuando el Cliente creado lleva un tiempo sin realizar ninguna acción automáticamente se desconecta por lo que se ha creado una función conectar() que permite conectarse al broker antes de realizar ninguna acción para evitar problemas. Cuando se realice una nueva conexión es necesario volver a indicar los topic a los que debe subscribirse para obtener datos mediante ellos. Esto es debido a que al desconectarse se suprimen todos los datos asociados al cliente creado.

```
private MQTTConnection(Context context) {
    id = MqttClient.generateClientId();
    if (client == null || !client.isConnected()) {
        client = new MqttAndroidClient(context, URI,
                id);
    }
    this.context = context;
    subcribirse();
}
```

Para comunicarse mediante MQTT se establece una función a la que se llama si llega un mensaje a algún topic a los que la aplicación está subscrita. Esta función se encarga de todo el procesado de los datos llegados de las placas hardware, suministrando la información necesaria a las funciones encargadas de pintar las gráficas o de informar al usuario de posibles errores.

```
private void llegadaDeMensajes() {
    mqttConnection.getClient().setCallback(new MqttCallback() {
        @Override
        public void connectionLost(Throwable cause) { mqttConnection.conectar(); }

        @Override
        public void messageArrived(String topic, MqttMessage message) throws Exception
```

Cuando el usuario clica sobre un botón para actuar sobre un sensor o pedir datos del mismo primero se comprueba si hay conexión con el broker MQTT, y luego se realiza una publicación MQTT sobre el topic especificado. Dependiendo del sensor y la petición del usuario sobre el mismo esta publicación se realiza por un topic u otro, es decir, si por ejemplo pide encender el ventilador se publicará la petición por un topic, pero si la acción es pedir la temperatura de la habitación se realizará por otra. De esta manera se envían y se reciben datos por topics distintos evitando que colapse un topic específico.

5.3.4 Gráficas y librerías

Para la realización de las gráficas se han utilizado las siguientes librerías:

- *MPAndroidChart*: es específica para la realización de gráficas sencillas y dinámicas. La gran utilidad de esta librería se encuentra en la gran cantidad de gráficas que posee y la sencillez de su uso. Se ha utilizado como gráfico de barras en la pantalla de temperatura y en la ventana de luminosidad como gráfico circular.

- *GraphView*: es otra librería especializada en la realización de gráficas. Una de las características de dicha librería es la posibilidad de insertar como coordenada X una fecha. En este caso se utiliza en la pantalla de temperatura histórica para poder mostrar la hora a la que se tomó la medida de la temperatura para poder observar la evolución.

- *Android-DecoView-Charting*: es otra librería especializada en la realización de gráficas dinámicas sobre todo circulares. Es muy útil para realizar gráficas dinámicas con efectos visuales atractivas para el usuario y pudiendo realizar acciones sobre las mismas. Se ha utilizado como gráfica circular y con efectos visuales sobre la pantalla de humedad.

Se ha utilizado para el funcionamiento de la aplicación:

- *paho.service.android*: es un librería especial para la creación de clientes MQTT suministrada por eclipse. Además es recomendada por las plataformas que poseen broker MQTT para su utilización. Posee una forma muy sencilla de enviar y recibir mensajes utilizando pocos recursos energéticos.

5.3.5 Almacenamiento

Se ha implementado una base de datos para el almacenamiento de las temperaturas medidas. Se ha utilizado una base de datos SQLite encargada de almacenar los datos necesarios. Dicho sistema de gestión de bases de datos relacional es muy útil para el almacenamiento en dispositivos empotrados. Además se utiliza el lenguaje SQL para ejecutar consultas y es de código libre.

Para poder mostrar la información sobre la gráfica se ha almacenado la fecha en la que se realiza la medición, el valor de la temperatura y un valor lógico para indicar si pertenece a un dispositivo hardware u a otro.

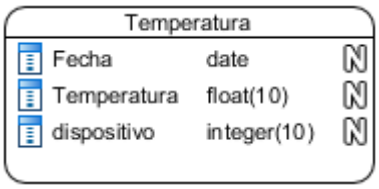

Ilustración 32 Diseño de la base de datos

Para la obtención de los datos de un determinado día se ha realizado un método que realiza una consulta usando las funciones de SQLite en Android, de tal manera que los datos se obtienen en forma de lista y ordenados en el tiempo.

Para evitar almacenar datos erróneos antes de llamar a la función que guardará los datos se comprueban que no sean datos incorrectos del sensor de temperatura. Si son erróneos se mostrará una alerta indicando que el valor es incorrecto.

6

Mantenimiento y pruebas

6.1 Mantenimiento

La ejecución de este proyecto se ha realizado en función del documento de requisitos establecido, por lo que cualquier persona que quiera saber cuál es el funcionamiento del sistema puede acceder al documento.

Sobre las placas hardware se han realizado los diseños de instalación de los sensores, por lo que cualquier modificación afectará al funcionamiento de los mismos teniendo que modificar el código para la placa modificada.

Se ha añadido al código de cada sistema constantes con los datos principales de las aplicaciones de tal manera que si se necesita modificar cualquiera de estos datos solo hay que modificar dichas constantes sin tocar las funcionalidades.

El mantenimiento de los dispositivos se hará por separado. Lo único que une a los distintos dispositivos es un mismo broker MQTT por lo que si se quiere modificar la funcionalidad de algún dispositivo no será necesaria la modificación de los demás dispositivos.

Las labores de mantenimiento sobre el sistema actual son:

- Comprobar que el broker HiveMQ para las comunicaciones MQTT sigue en funcionamiento.
- Comprobar que las placas hardware siguen funcionando y conectadas tanto a Wifi como al broker MQTT.
- Sobre la aplicación móvil comprobar si los datos de los sensores siguen siendo coherentes, comprobar que las distintas versiones del sistema operativo Android han producido errores en la aplicación y hay que actualizarla.
- Posible mejoras tanto en los dispositivos hardware como en la aplicación móvil.

6.2 Pruebas

Tras la decisión de la realización del proyecto mediante una metodología incremental en cada fase se han ido realizando pruebas conjuntas del funcionamiento de la sección implementada en cada fase. De tal esta manera se han desarrollado las siguientes pruebas.

- *Conexión de los sensores:* se conectaron todos los sensores y se realizaron pruebas de lectura y escritura para comprobar la correcta conexión de los mismos.
- *Conexión Wifi y MQTT:* sobre Arduino se realizaron pruebas de la correcta conexión Wifi sobre el router. Para los tres dispositivos se comprobó la correcta conexión al broker MQTT y la comunicación entre los dispositivos.
- *Procesamiento y almacenamiento de datos:* una vez que las comunicaciones estaban correctamente se realizó la implementación del procesamiento de los datos para su uso posterior en la aplicación móvil. Se realizaron pruebas mostrando por consola los datos recibidos y procesados para su uso posterior.
- *Realización de las pantallas:* con los datos ya procesados se pasó al desarrollo de las gráficas para mostrar los datos en tiempo real, y a la realización de cada una de las pantallas que el usuario utilizará para actuar sobre los sensores. Así que, se realizó una prueba de funcionamiento íntegro de cada una de las pantallas de la aplicación.

Para terminar se ha realizado una prueba de uso real de la aplicación por parte de un usuario comprobando que todas las respuestas se producen correctamente, y que las placas actúan correctamente ante la petición del usuario.

7

Conclusiones

7.1 Problemas y dificultades encontradas

Los principales problemas han venido de la utilización de Android Things. La instalación fue complicada debido a que dejaba la tarjeta SD inutilizada y si se instalaba mal o querías cambiar algo había que formatear la SD y Windows Defender no dejaba hacerlo. Además otra dificultad añadida a la instalación fue entender cómo poder lanzar

aplicaciones en el dispositivo con Android Things debido a que primero había que conectar el portátil con los comandos de Android "adb", una vez hecho esto podías lanzar la aplicación desde Android Studio. Sumado a esto se añadía la dificultad de poder lanzar la aplicación de tal manera que se quedase funcionando sobre el dispositivo, cosa que no he conseguido realizar debido a que las indicaciones que proporciona Android no son las correctas para poder realizarlo.

Android Things es muy potente respecto a que puedes hacer lo mismo que con una aplicación móvil. Pero para mí tiene un problema muy grande y es que lo que en Arduino haces en pocas líneas sobre Android necesitas muchas más. Además se suma al ser sobre Java si necesitas usar una librería para leer datos de un sensor no puedes realizar más allá de lo que te permite la librería. Cuando leí por primera vez pensé que sería más facil utilizar dicho sistema pero no fue así. Para cualquier persona que quiera utilizar dicho sistema operativo es recomendable que primero lo pruebe y sea capaz de estimar el tiempo para cada tarea, porque de lo contrario es probable que tarde más de lo previsto.

Sobre la aplicación móvil la principal dificultad fue entender el funcionamiento de las comunicaciones MQTT con la librería PAHO. Además nunca había utilizado gráficas para realizar aplicaciones de ningún tipo por lo que saber cómo insertar y como mostrar datos era una dificultad.

7.2 Internet de las Cosas

Este nuevo concepto de la informática creo que es muy interesante por lo que haber aprendido muchísimo más sobre el internet de las cosas creo que es muy útil. Además con el paso de los años está creciendo mucho más gracias a que casi todos los dispositivos electrónicos ya poseen dispositivos de comunicaciones integrados en ellos.

7.3 Recomendaciones para líneas futuras

Respecto a Android Things también comentar que en mi opinión no es útil para utilizarlo en el mundo del internet de las cosas. Cualquier cosa realizada con Arduino es muchísimo más simple, sencillo y rápido que con Android Things. La única utilidad real es que puedes utilizar toda la potencia que posee Android junto con las librerías que se pueden usar dentro para realizar aplicaciones más complicadas. Pero para leer sensores y procesarlos fuera de dicho sistema es mucho mejor Arduino.

Para alguna línea futura quizás sea útil realizar una investigación más a fondo sobre dicho sistema operativo usando todos los kit que ofrece Android para utilizar Android Things. También puede ser muy bueno realizar una librería para poder manejar más tipos de sensores o poder utilizar otros dispositivos hardware distintos a los que ofrece Android.

8

Referencias

Frank Ableson, Charlie Collins. *Android, Guía para desarrolladores*, 2010

Google LLC. Google Developers. web: https://developer.android.com/things/

Google LLC. Google Developers. web: https://developer.android.com/

Google LLC. Android Things. web: https://androidthings.withgoogle.com

GitHub Inc. GitHub. web:https://github.com/

Sandro Kock. December 2, 2015. HiveMQ. web: https://www.hivemq.com/blog/mqtt-client-library-enyclopedia-paho-android-service/

Arduino LLC. Arduino. web: https://www.arduino.cc/

Luis del Valle. Programar Facil. web: https://programarfacil.com/blog

Raspberry Pi Foundation. Raspberry Pi. web: https://www.raspberrypi.org/

Jonas Gehring, 2019. GraphView, web: http://www.android-graphview.org/

Philipp Jahoda. MPAndroidChart. web: https://github.com/PhilJay/MPAndroidChart

DecoView. Brent. web: https://github.com/bmarrdev/android-DecoView-charting

Apéndice A
Manual de Instalación

Requerimientos:

Para la instalación y puesta en funcionamiento de este trabajo de fin de grado es necesario la puesta en funcionamiento de las dos placas hardware con los respectivos sensores y poseer un teléfono móvil Android con versiones iguales o superiores a 4.4 Kit Kat.

Procedimiento:

Para la instalación del sistema operativo Android Things sobre una Raspberry será necesaria una tarjeta microSD de al menos 8 GB con las versiones actuales en 2019. Tras esto basta con seguir los pasos referenciados en la web de desarrolladores de android para la instalación de la imagen del sistema operativo en el siguiente enlace: https://developer.android.com/things/hardware/raspberrypi

Una vez instalado el sistema operativo bastará con conectar la Raspberry a la corriente y a un cable Ethernet de datos. Una vez realizado esto se puede abrir el proyecto de AndroidThings con Android Studio o bien entrar dentro de dicho proyecto y ejecutar los siguientes comandos:

cd AndroidThings
./gradlew assembleDebug
adb install -g -r app/build/outputs/apk/app-debug.apk
adb shell am start com.example.juan.androidthings /.MainActivity

Para la instalación sobre Arduino de la aplicación solo habrá que instalar Arduino IDE y sobre esta herramienta cambiar el SSID y la Constraseña de Wifi por el del router que se vaya a utilizar para dicho sistema.

Para la instalación de la aplicación si no se encuentra alojada en Play Store bastará con instalar el APK de la aplicación dándole los permisos solicitados por Android para la instalación. Si esto no es posible, se puede cargar el proyecto AplicaciónMóvil sobre Android Studio y lanzarlo sobre su emulador.