Universidad Zaragoza
1542

Trabajo Fin de Máster

Desarrollo de un ecosistema IoT para la mejora de la eficiencia energética de edificios

Development of an IoT ecosystem for improvement of buildings' energy efficiency

Autor

Jorge Learte Liarte

Director

Alberto Belled Casabona

Co-director

Emiliano Bernués del Río

Universidad de Zaragoza/EINA

2017-2018

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. Jorge Learte Liarte,

con nº de DNI 73220100Z en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Máster, (Título del Trabajo)

Desarrollo de un ecosistema IoT para la mejora de la eficiencia energética de edificios

Development of an IoT ecosystem for improvement of buildings' energy efficiency

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 17-Septiembre-2018

Fdo: Jorge Learte Liarte

A mis padres

Jorge Learte Liarte
20 de septiembre de 2018

DESARROLLO DE UN ECOSISTEMA IoT PARA LA MEJORA DE LA EFICIENCIA ENERGÉTICA DE EDIFICIOS

RESUMEN

La evolución de las Tecnologías de la Información (TI) en el mundo en el que vivimos hace necesario implementar sus avances con el fin de contrarrestar los efectos nocivos que la tecnología provocan en el ecosistema. Para ello, tras estudiar las distintas alternativas tecnológicas, se decide diseñar un sistema de monitorización y control de los gastos de energía excesivos que se producen en un determinado entorno.

El entorno escogido es el bloque de oficinas en el cual se desarrolla el proyecto, tomando como muestra piloto una planta y en concreto como sala piloto una de las salas de reuniones. Sobre los distintos espacios se diseña el despliegue de una red de sensores cuya información se transmite a través de distintos protocolos de comunicación a los nodos de procesado y visualización.

Los gastos a minimizar que abordamos son el gasto de agua y electricidad. Para optimizar el gasto de agua se diseña un sistema de apagado automático de grifos mediante el uso de detectores de ultrasonidos. Por otro lado el ahorro en el gasto de electricidad es más complejo de optimizar ya que se compone tanto del referente al destinado a iluminación como el de la climatización. Para conseguirlo se diseña por una parte un sistema de apagado automático de la iluminación en aseos basado en sistema de detección de movimiento y por otra un sistema basado en una lógica de control que nos permita optimizar el uso de la climatización e iluminación en nuestra planta a estudiar. En el alcance del proyecto no se incluye la implantación total de la red de sensores dado que se trata de un proyecto en fase piloto. Por ello y con el fin de obtener ahorros reales de alguno de los sub-sistemas se realiza también el diseño de sistemas de contra-medidas.

La lógica de control reside en un nodo de procesado que recibe la información a través de los distintos protocolos de comunicación escogidos, ejecuta el algoritmo diseñado para alcanzar las condiciones de confort de una forma eficiente y envía sus salidas a los responsables de los sistema de iluminación, climatización y de elevación de las persianas.

Como valor añadido al proyecto se diseña el nodo de visualización como una aplicación para dispositivos table con el sistema operativo Android. Su finalidad es ser un tablero desde el cual se pueda controlar las variables monitorizadas de los distintos espacio en los que se ha desplegado y el reajuste de las condiciones de confort.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Internet de las cosas

Hoy en día, las TI son cada vez más accesibles para la mayoría de la población. Las últimos estudios de 2018 afirman que el 53 % de la población mundial tiene acceso a Internet (un 7 % más que en 2017) y que el 75 % de ellos usa las redes sociales. Este incremento en el número de usuarios de Internet refleja nuestra enorme necesidad de conectividad, no solo con el resto de la población, sino también con el mundo que nos rodea.

Es en esta realidad de conectividad creciente donde surgió la necesidad de definir un nuevo concepto para referirse a la interconexión digital de los objetos de nuestro entorno a través de Internet. Este concepto fue bautizado por primera vez por Kevin Ashton [1] (creador de estándares de tecnologías tan utilizadas como RFID) como Internet de las Cosas o Internet of Things (IoT) en el año 1999. Esta conceptualización lo convirtió en uno de los promotores de esta tecnología y abanderado en el fomento del uso del Internet de las Cosas en congresos y grandes multinacionales.

Dar una definición de IoT no es algo sencillo. Desde un punto de vista tecnológico, The Global Standards Initiative on Internet of Things (IoT-GSI), de ITU, define IoT como: "Una infraestructura global en el marco de la sociedad de la información que provee servicios a través de la conexión de elementos físicos o virtuales, basada en tecnologías de la información y la comunicación, tanto existentes como en desarrollo".

A pesar de su conceptualización tardía, los primeros casos de elementos conectados los encontramos a finales del siglo XIX. Fue en 1874 cuando, basándose en enlaces de radio de onda corta, se consiguió conectar una estación meteorológica en la cima del Mont Blanc con un laboratorio en París. Décadas después, grandes inventores y pensadores de la época, como Nikola Tesla o Alan Turing, ya anticiparon el crecimiento de la conectividad global. En diversas publicaciones hablaron de un planeta entero que, tan pronto como estuviera interconectado, se convertiría en un gran cerebro lleno de partículas de un todo real que seguirían el proceso normal de aprendizaje de un niño y que cualquier persona podría llevar en su bolsillo. Finalmente, estas expectativas se consumaron cuando, en las décadas de los 60 y 70, en el Departamento de Defensa de EEUU comenzaron a aparecer los primeros protocolos de comunicación que dieron pie a lo que hoy conocemos como Internet [2].

Actualmente, estudios realizados por grandes multinacionales del mundo IT prevén que ese gran cerebro estará formado, en 2020, por unos 30.000 millones de "cosas inteligentes" conec-

tadas a Internet y, que solo cuatro años más tarde, esa cifra se duplicará [3]. Este incremento exponencial del número de elementos conectados a Internet hace que la implementación de las tecnologías IoT en el mundo en el que vivimos ofrezca a la sociedad un amplio abanico de oportunidades en cuanto a nuevos servicios e innovación se refiere.

Los recientes avances en este ámbito aceleran la aparición de plataformas IoT a gran escala. Con ellas recolectamos, procesamos y analizamos los datos en tiempo real para alimentar al ecosistema de soluciones "inteligentes", o Smart Solutions. El uso de esas plataformas para la sensorización de los objetos que nos rodean supondrá, en el futuro inmediato, una revolución en la forma de obtener información. Con ese enorme volumen de datos se podrá, entre otras muchas cosas, optimizar la gestión de la industria (Smart Industry y Smart Energy), del mundo médico (Smart Health), de nuestra propia casa (Smart Home), del mundo agrícola (Smart Farming) o, incluso, de la ciudad en la que vivimos (Smart City).

Los ejemplos de soluciones IoT en la nueva era Smart son innumerables y se extienden a sectores muy diversos.

Como primera implementación se tomó como experiencia piloto el llamado Smart Farming [4], el cual agrupa tanto la agricultura como la ganadería. Este tiene una casuística sumamente imprevisible debido a su alta correlación con el clima y las condiciones ambientales. Por ello, el uso de la tecnología no solo reducirá el factor humano en la supervisión de las plantaciones, sino que además aumentará la productividad y rentabilidad al poder controlar de forma sencilla el estado en el que se encuentran. Para ello, se implementa en la granja la automatización y sensorización de parámetros que le afectan como puede ser la climatología, el estado del aire, de la tierra o incluso el estado de salud de los animales. Con esta información es posible aplicar técnicas como el riego eficiente, el uso específico y preciso de fertilizantes y pesticidas en los cultivos así como de alimentos para los animales.

Todo lo que respecta al mundo de la medicina (Smart Health [5]) tiene como principal propósito ofrecer la mayor calidad posible en la atención, diagnóstico e intervención del paciente. Reducir las grandes colas, el control de los inventarios o un mayor control del estado de los pacientes tanto dentro como fuera del hospital son algunas de las posibilidades que la aplicación de soluciones IoT a la medicina pueden hacer realidad. Al tratarse de un aspecto de suma importancia nunca se buscará sustituir al ser humano sino que se buscará un ahorro de tiempo del personal médico y una atención más completa y personalizada.

Para conseguir estos objetivos vuelve a ser de nuevo necesaria la implantación de una extensa red de sensores, entre los que se pueden incluir wearables, que monitorizan el estado de los pacientes del hospital o de aquellos que han sido de alta y son propensos a necesitar una rápida actuación (tele-medicina). Del mismo modo si extendemos esta red de sensores a la localización real-time tanto del personal médico del hospital como de los equipos médicos y coordinamos todo de una forma eficiente lograremos reducir proporcionalmente a esta eficiencia el tiempo de reacción ante una emergencia.

Otro sector en el que se implementan las soluciones IoT es el de la industria (Smart Industry o Industria 4.0), consolidándose como uno de los campos con mayor índice de implementación en los próximos años.

A día de hoy estas soluciones se centran principalmente en la mejora de la situación de confort de los usuarios: gestión de la temperatura, de la iluminación o incluso de los electrodomésticos. En este caso nuestra red no está formada solo por una red de sensores de la que

extraer las condiciones actuales sino también por dispositivos que mediante su conectividad nos permitan un control remoto o programado de su funcionamiento.

El porcentaje de compañías que incluyen las soluciones IoT en sus procesos crece año tras año. Pero no solo eso sino que también se expande el número de dispositivos conectados en esas soluciones. Esto, unido a la conocida como "revolución de los sensores" en la cual las grandes compañías proveedoras de ellos están invirtiendo grandes cantidades de capital en el desarrollo de novedosos sensores, hace que en los próximos años las previsiones afirmen que la tecnología IoT se convertirá en un negocio que crecerá exponencialmente. El 79 % de aquello que ya han adoptado soluciones IoT prevén que el 50 % de los procesos de negocio incluirán sensorización IoT para 2022. Derivado de esto, el volumen de datos que se maneja será suficientemente grande (Big Data) como para incluir el uso de Inteligencia Artificial (IA) y Machine Learning (ML) para analizar todos esos datos y generar actuaciones inteligentes. Este despunte tecnológico no será pasado por alto por la mayoría de las grandes compañías que, según el 82 % de los usuarios de IoT, buscarán establecer colaboraciones con las industrias para el desarrollo de soluciones IoT que mejoren sus negocios y la capacidad de análisis en sus procesos.

1.2. Entorno de trabajo

Este proyecto se desarrolla en el marco de trabajo de Hiberus Tecnología, una compañía especializada en la consultoría de negocio y la prestación de servicios tecnológicos. Es la compañía de tecnología líder del noreste de España, referente en el mercado español y en pleno proceso de expansión e internacionalización. Hiberus se integra en uno de los principales grupos empresariales del sector de las Tecnologías de la Información y Comunicación (TIC) en España, formado por más de 1.200 profesionales y con más de 120 M de euros de facturación.

El crecimiento de la compañía y las nuevas oportunidades relacionadas con las tecnologías IoT hacen que se crea conveniente investigar dichas tecnologías y desarrollar metodologías de fácil integración en futuros proyectos reales.

1.3. Motivación y objetivos

El objetivo principal de este Trabajo Fin de Máster es la investigación del ecosistema IoT y su aplicación a la mejora de la eficiencia energética de un bloque de oficinas. Para ello, se desarrollará un sistema de monitorización y control de las principales variables, como la luminosidad o la temperatura, para así poder tomar las decisiones de cómo actuar en cada momento sobre los sistemas del edificio (iluminación, climatización,etc...).

El gasto energético excesivo de cualquier empresa supone un descenso de las ganancias de esta, lo que repercute no solo en la empresa sino también en sus trabajadores. Además, un sistema como el que se plantea tendrá también como objetivo la mejora del ambiente en el que se trabaja, lo que se traduce en un aumento de la eficiencia del trabajador y por tanto en la reducción del tiempo necesario para realizar sus tareas, lo que supone un aumento de las ganancias. Aunque en el alcance del proyecto no se incluye su implantación real se desarrolla con la plena confianza de llegar a hacerlo en la sede que Hiberus Tecnología tiene en Zaragoza.

Otro de los objetivo del proyecto, aprovechando que se desarrolla en un entorno fuera de la Universidad de Zaragoza, es el integrarse en un ambiente real empresarial con el fin de desa-

rrollar aptitudes que me ayuden en mi futuro laboral.

La principal motivación que me lleva a realizar este proyecto se basa en el aprendizaje de todo lo relacionado con esta nueva tecnología puntera y la aplicación de este aprendizaje a contribuir, en la medida de lo posible, al planeta en el que vivimos.

1.4. Planificación del proyecto y tiempos empleados

La planificación del proyecto consiste en 5 fases que se describen de forma paralela en la estructura de la memoria que veremos a continuación.

- Fase I. Estudio de la documentación e investigación de las plataformas hardware y software del ecosistema IoT: dispositivos, sensores, actuadores, tecnologías y protocolos de comunicación,...
- Fase II. Desarrollo de metodologías, haciendo uso de los elementos seleccionados en la fase anterior.
- Fase III. Análisis de los principales factores que afectan a la eficiencia energética de un edificio y elección final de las metodologías a implementar con el fin de diseñar los diferentes bloques de sensorización que formarán parte de nuestro sistema.
- Fase IV. Desarrollo de la topología de control de las variables de eficiencia energética y obtención de resultados.
- Fase V. Redacción de la memoria.

1.5. Estructura de la memoria

En cuanto a la estructura de la memoria consta de 5 capítulos y 3 apéndices.

En el Capítulo 1 se presenta una Introducción al IoT junto con la descripción de los distintos objetivos y motivaciones para desarrollar este proyecto junto con su planificación y la estructura de la memoria. En el Capítulo 2 hablaremos de los resultados del estudio de la documentación y de la investigación de las distintas plataformas hardware y software que encontramos en el ecosistema IoT. A pesar de haber realizado una amplia investigación nos centraremos solo en describir aquello que tenga utilidad para las metodologías desarrolladas más adelante. Una vez que conocemos el ecosistema IoT, es momento de analizar en el Capítulo 3 aquellos factores que afectan a la eficiencia energética y de diseñar los bloques de sensorización necesarios, haciendo uso de parte de los elementos descritos en el capítulo anterior. Además, hablaremos de cómo se realiza la gestión de toda la información obtenida. Tras esto, en el Capítulo 4, coordinaremos todos estos bloques mediante una topología en cuyo núcleo se encontrará un nodo de procesado y visualización y presentaremos los resultados obtenidos. Finalmente, en el Capítulo 5, se presentarán las conclusiones y líneas futuras.

En el Apéndice A hablaremos de metodologías que se han desarrollado pero no se han llegado a implementar finalmente en el sistema. En el Apéndice B describiremos brevemente cada uno de los sensores y actuadores de los que se hablan a lo largo de esta memoria. En último lugar el Apéndice C se ilustra la estructura de las placas utilizadas.

Capítulo 2

Ecosistema IoT

Una aplicación IoT puede ser entendida como un conjunto de bloques que cooperan entre sí. Existe una gran variedad de aplicaciones pero, por básicas que estas sean, es inevitable la necesidad de tener un gran número de dispositivos hardware a los que se les permita interactuar con distintos sensores y actuadores, y que interaccionen entre sí mediante tecnologías de comunicación. La información que obtenemos se almacenará y procesará en estos nodos o en una plataforma software. Adicionalmente, en aquellas aplicaciones en las que tenemos un gran número de dispositivos hardware y/o sensores/actuadores es fundamental describir una topología de red asociada a la tecnología de comunicación empleada que nos permita adaptarnos a los requerimientos.

Como en la mayoría de aplicaciones, se distingue una parte hardware (plataforma hardware) y una parte software (plataforma software), que constituye el *"back-end"* de nuestra aplicación y se encarga de aportar la potencia de procesamiento. Siendo estrictos, esta plataforma hardware debería estar formada no solo por los dispositivos encargados de recoger la información de sensores o de provocar un cambio en los actuadores, sino también por los propios sensores o actuadores. Sin embargo, en nuestro caso y con el único fin de dar una visión más clara de los distintos bloques los consideraremos como dos bloques independientes: por un lado hablaremos de la plataforma hardware, haciendo referencia a los dispositivos y, por otro, de sensores/actuadores. Además, de aquí en adelante nos referiremos al conjunto de sensores y actuadores con el término "periféricos", puesto que van a ser dispositivos auxiliares e independientes que se conectarán a una "unidad central" - nuestra plataforma hardware.

Ambas plataforma, hardware y la software, forman la base de la aplicación. El resto de bloques, por su parte, se incorporan a ella con el fin de dotarla de una funcionalidad más completa. Todos ellos trabajan de forma conjunta realizando un trabajo distribuido.

A continuación, se describen estos cuatro bloques básicos: plataforma hardware, periféricos, tecnologías de comunicación y plataforma sofware.

2.1. Plataforma hardware

El gran abanico de posibilidades que nos ofrece el mundo IoT hace que sea necesaria una ardua tarea de investigación acerca de cual es la mejor plataforma hardware. Tras dicha investigación y como paso previo a describir algunas de dichas posibilidades, cabe destacar que no

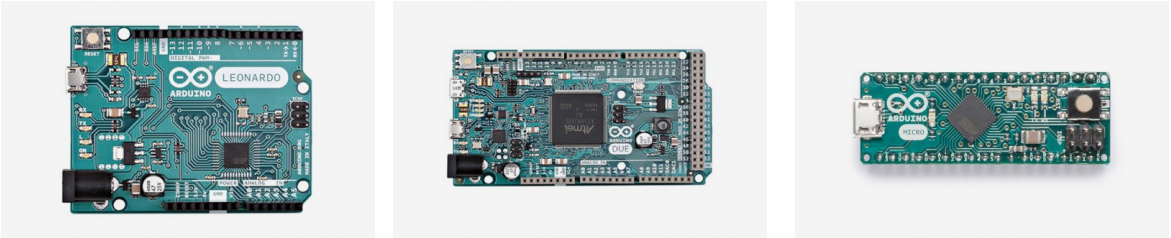
se ha alcanzado una conclusión de cuál es la mejor plataforma hardware, sino que esa conclusión dependerá de la finalidad del proyecto. Cada una de las plataformas hardware dispone de una larga lista de características, algunas de ellas vinculantes para nuestra decisión (número de pines, procesador, dimensiones, etc...) y otras no. Como principales plataformas hardware destacan las conocidas Arduino y Raspberry.

2.1.1. Arduino

Arduino[6] es una plataforma de hardware libre basada en una placa con un microcontrolador, con entradas y salidas tanto analógicas como digitales que permitan la conexión con los periféricos. Cuenta con una amplia comunidad de desarrolladores que la convierten en la más popular del mercado y con un entorno de desarrollo integrado (IDE) propio (Arduino IDE 1.8.7) que facilita el desarrollo de proyectos. Gracias a la extensa documentación, las posibilidades que nos brinda Arduino van desde un ámbito de trabajo puramente educativo hasta el profesional, permitiéndonos construir de una forma realmente sencilla nuestros prototipos básicos.

Existe una gran variedad tanto de placas disponibles como complementos para ellas: (*shields*), permitiendo que estos prototipos se adapten perfectamente a los requisitos funcionales de cada proyecto. Cada una de estas placas se alimenta con el voltaje de alimentación que corresponda tanto de forma externa vía USB como a través de distintos tipos de baterías, siendo la propia placa la que alimenta a los periféricos conectados a sus pines con un valor de tensión que recibe el nombre de voltaje de funcionamiento. A pesar de que este voltaje depende de la placa escogida, es muy común que encontremos pines analógicos de alimentación a otros valores de voltaje distintos (3.3 V o 5 V) para poder conectar todo tipo de periféricos.

A continuación, se presenta una comparativa de los tres modelos de placa sin conectividad que podrían cubrir la mayoría de los proyectos de IoT.



(a) Leonardo **(b)** Due **(c)** Micro

Figura 2.1: Modelos de Arduino sin conectividad

Cada una de estas placas nos permitiría adaptarnos a los requisitos de distintos tipos de proyectos IoT. Una placa Arduino Due se adapta perfectamente a un gran proyecto en el que es necesario un alto número de periféricos y por tanto también un buen procesador y un gran número de pines. Del mismo modo un Arduino Leonardo nos sería útil para una aplicación

[1] I: pin de entrada, O-PWM: pin de salida haciendo uso de Pulse-width modulation
[2] EEPROM: memoria que pertenece en la placa guardada incluso cuando está está apagada.
[3] SRAM: memoria estática donde se guardan las variables en tiempo de ejecución.
[4] Memoria flash: memoria en la que se guarda nuestro programa.

	Leonardo	**Due**	**Micro**
Procesador	ATMega32U4	ATSAM3X8E	ATMega32U4
Reloj (MHz)	16	84	16
Voltaje de alimentación (V)	6-20	6-16	6-16
Voltaje de funcionamiento (V)	5	3.3	5
Intensidad de pines I/O (mA)	40	130	20
Intensidad de pin 3.3V/5V (mA)	40/50	800	20/50
Pines analógicos[1]	12(I) y 7(O-PWM)	12(I) y 12(O-PWM)	12(I) y 8(O-PWM)
Pines digitales (IO)	12	54	12
EEPROM (kb)[2]	1	-	1
SRAM (kb)[3]	2	96	2.5
Memoria flash (kb)[4]	32	512	32
Dimensiones (mm×mm)	68.4x53.4	101.52x53	48x18
Peso (g)	25	36	13

Tabla 2.1: Comparativa placas de Arduino

estándar en la que el número de periféricos no es tan elevado. Finalmente, Arduino Micro[7] tiene exactamente las mismas prestaciones que Arduino Leonardo pero con unas dimensiones mucho menores, adaptándose por tanto a proyectos en los que el espacio que se dispone es reducido.
Puesto que es la única diferencia que existe entre ellas, se podría valorar el uso exclusivo de uno de ellos. Sin embargo, Arduino Leonardo posee una capacidad que no tiene un Arduino Micro: permite incorporar a él un *shield* de Wi-Fi, que dotaría a nuestro sistema de conectividad. Por tanto, en caso de que para el proyecto que desarrolláramos fuera imprescindible utilizar dicho *shield*, sería necesario el uso de Arduino Leonardo. Para el resto de casos y tras un proceso de pruebas se decide utilizar Arduino Leonardo para la fase de diseño del montaje, ya que su mayor tamaño permite un espacio de trabajo con la *protoboard* más limpio, y el uso de Arduino Micro para la fase de prototipado y comercialización.

Hasta ahora se había especificado que las placas descritas no incluyen ningún medio de conectividad. Para disponer de una placa con ella existen distintas opciones: incorporar distintos shields del catálogo de Arduino, usar módulos de comunicación de otros fabricantes o directamente comprar una placa que posea la conectividad deseada.

Esta última opción parece la más simple aunque a menudo no es la más económica. Además, en la mayoría de los casos no existen múltiples placas para cada tipo de conectividad por lo que nuestras posibilidades se verán reducidas a una sola en caso de que el tipo de conectividad a usar quede fijado en las especificaciones. Los posibles tipos de conexión que se nos ofrecen son los siguientes:

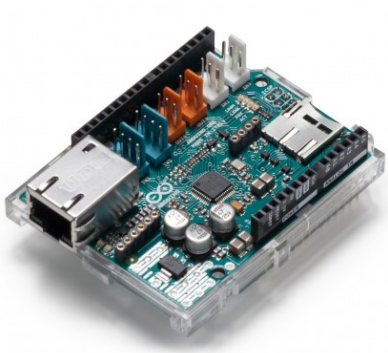
- MKR Wi-Fi 1000: conexión Wi-Fi 2.4GHz
- MKR FOX 1200: conexión SigFox con f_c= 868 MHz

- MKR WAN 1300: conexión LoRa con f_c= 433/868/915 MHz
- MKR GSM 1400: conexión GSM 850 MHz, E-GSM 1900 MHz, DCS 1800 MHz y PCS 1900 MHz

Afortunadamente no es estrictamente necesario usarlas ya que, para ciertos tipos de conexión, existen periféricos que podemos añadir a nuestra placa, como el módulo de Wi-Fi ESP-01, el módulo Bluetooth HC-05 o los módulos de conexión Zigbee.

Adicionalmente, como ya hemos mencionado, también es posible utilizar de los denominados Wi-Fi Shield o Ethernet Shield para dotar a nuestra placa de conectividad a internet de forma inalámbrica o mediante un cable Ethernet. Estos *shields*, que podemos ver en la Figura 2.2, se colocan sobre las placas compatibles ocupando la mayoría de los pines pero permitiéndonos conectar nuestros periféricos a los pines del *shield*.

(a) Wi-Fi Shield



(b) Ethernet Shield

Figura 2.2: Selección de Shields de Arduino

Cierto es que por el alcance del proyecto no todas estas opciones serán estudiadas pero sí aquellas que nos resultan más interesantes para la descripción de metodologías que abarquen distintos tipos de proyectos que se puedan plantear. Entre todas ellas se prioriza el estudio de las opciones que, a priori, mejores características presentan: los periféricos de comunicación (que desarrollaremos más adelante en la la sección 2.3.2) y la placa de Arduino MKR1000:

MKR1000[8] es una placa fabricada por Arduino que combina las prestaciones de una placa Arduino de pequeño tamaño (como Arduino Micro) con las funcionalidades que nos otorgaría un Wi-Fi Shield. La relación entre sus características y su precio la convierten en una solución muy recomendable para proyectos en los que la tecnología de conectividad a utilizar sea Wi-Fi. Su principal peculiaridad es que el voltaje de funcionamiento de sus pines es de 3.3 V, aunque incluye un pin para suministrar alimentación de 5 V, por lo que será necesario el uso de bloques adicionales en caso de que gran parte de nuestros periféricos hagan uso de pines que funcionen a 5 V.

Otra de sus ventajas es que incorpora un adaptador JST macho con el que, al conectarle una batería LiPo de 3.7 V con capacidad superior a 700 mAh, podemos o bien alimentar nuestra placa o cargar la batería en caso de que ya esté siendo alimentada. Esta peculiaridad añadida convertiría este tipo de placa en la idónea para aquellas aplicaciones en las que el dispositivo esté en una posición fija durante un periodo de tiempo y móvil en otro. Para controlar dicha carga se incorpora un LED que nos indica su estado. Finalmente incluye un Real-Time Clock (RTC)

con el que podemos gestionar tiempos: contar periodos de tiempo, programar alarmas,etc. Todo ello sin necesidad de incluir un bloque adicional.

Procesador	SAMD21 Cortex-M0
Reloj (MHz)	48
Voltaje de alimentación (V)	5
Intensidad de pines y de 3.3V (mA)	7
Pines analógicos	7(I), 1(O[5]) y 12(O-PWM)
Pines digitales	8
EEPROM (kb)	-
SRAM (kb)	32
Flash (kb)	256
Dimensiones (mm x mm)	61.5 x 25
Peso	32 g

Figura 2.3: Arduino MKR1000 y características

2.1.2. Raspberry

Raspberry es otra de las plataformas hardware más conocidas a día de hoy aunque con recursos de computación superiores a los ofrecidos por Arduino. Oficialmente usa sistemas operativos Linux como Raspbian, una distribución open-source surgida de Debian, aunque también puede incorporar UNIX o Windows 10.

Por sus características nos permitiría realizar proyectos de sensorización de mayor envergadura, o bien aprovechar la potencia de procesamiento que ofrece junto con la posibilidad de integrarlo de forma sencilla con distintas plataformas software para el procesado y almacenamiento de datos o para la visualización de estos. Para ello en los últimos meses Google ha lanzado un nuevo sistema operativo compatible con Raspberry, denominado Android Things 1.0. Este sistema permitiría hacer uso de la plataforma Raspberry para realizar el procesado y visualización de datos de nuestros proyectos.

Actualmente existen ocho modelos de Raspberry Pi (1A, 1A+, 1B+, 2B, 3B, 3B+, Zero y Zero W) aunque la buena relación prestaciones-precio de las más recientes hace descartar al resto. Como hicimos con Arduino, se presenta a continuación una comparativa de las que consideramos relevantes en nuestro estudio.

Por la naturaleza de los proyectos de sensorización que se esperan realizar la mayoría de las placas (todas las anteriores a la 3B) han sido descartados por su mala relación prestaciones-precio o porque no poseen tecnologías de conexión.

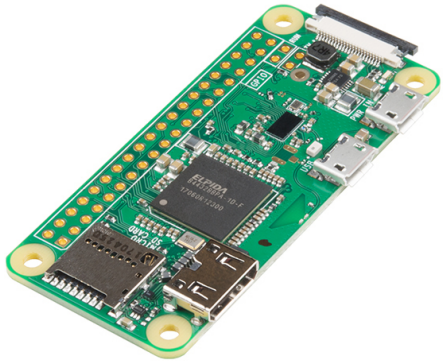
El procesador de las Pi Zero W debería ser suficiente para la gran mayoría de los proyectos que podamos imaginar. Sin embargo y dado que el alcance del mundo IoT es grande no está

[5]O: pin de salida

	Pi 3B+	**Pi Zero W**
Procesador	Broadcom BCM28837B0 de 4 núcleos a 1200MHz	Broadcom BCM2835 de 1 núcleos a 1GHz
RAM	1GB	512MB
Memoria	MicroSD	MicroSD
I/O	40	40
Salidas	Vídeo HDMI y 4xUSB	Vídeo MiniHDMI y 1 USB
Conexiones	Wi-Fi 802.11 b/g/n/ac doble banda, Bluetooth 4.2 y Ethernet 300Mbps	Wi-Fi 802.11n y Bluetooth 4.0
Dimensiones	85.6 mm x 53.98 mm	65 mm x 30 mm

Tabla 2.2: Comparativa de placas Raspberry Pi

(a) Pi 3B+



(b) Pi Zero W

Figura 2.4: Modelos de Raspberry Pi estudiados

de más prever la necesidad de una placa con un procesador superior como el de Raspberry Pi 3B+. Por ello y debido a sus características Raspberry Pi 3B+ debería ser usada en caso de que necesitemos un alto nivel de procesamiento, utilizar una red Wi-Fi 5 GHz o una conexión Ethernet. Por otro lado, Raspberry Pi Zero W se usaría en caso de que el espacio fuera reducido. Finalmente, en caso de que ninguno de los anteriores requisitos se dieran se debería testear en fase de pruebas si alguna de las placas genera algún otro inconveniente no previsto. En caso de que ambas fueran aptas para su uso seleccionaríamos la más adecuada para minimizar el presupuesto del proyecto.

En cuanto al uso de Raspberry como herramienta de procesado la placa escogida es Raspberry Pi 3B+: a día de hoy es suficientemente potente y su calidad-precio es superior al resto.

2.2. Tecnologías de comunicación

Otro de los bloques fundamentales en nuestro ecosistema IoT es la conectividad. Disponer de ella es casi un requisito fundamental puesto que cualquier sensor o actuador debe ser capaz de conectarse o ser conectado en cualquier momento y lugar. Como es de esperar, no existen una única forma de establecer una conexión entre nuestros dispositivos ni tampoco todas ellas

dan cabida a ser exploradas. Se comienza con las más básicas de nuestro día a día, como puede ser una conexión Wi-Fi o Bluetooth. Una vez exploradas estas se decide investigar algunas más ajenas a nuestro conocimiento con el fin de buscar soluciones a los problemas que nos generan las primeras o bien mejorarlas.

2.2.1. Wi-Fi

La conectividad Wi-Fi es por excelencia la tecnología de comunicación predominante en nuestro día a día, permitiendo que los dispositivos habilitados con Wi-Fi puedan conectarse entre sí o a Internet a través de un punto de acceso (AP). La tasa de transmisión que se consigue depende directamente de los parámetros de configuración como el ancho de banda de los canales, la frecuencia o la modulación entre otros. Con el fin de estandarizar, Wi-Fi se rige por los estándares IEEE 802.11. Los más comunes en la actualidad son:

- IEEE 802.11n: estándar más habitual en hogares. Soporta tasas de entre 50 y 600 Mbps en la banda de 2.4 GHz. Para conseguir dicha tasa máxima es necesario el uso de multiplexación por frecuencia Orthogonal Frequency Division Multiplexing (OFDM) (108 sub-portadoras), Multiple-input Multiple-output (MIMO) con 4 antenas, canales de 40 MHz y modulaciones 64-QAM con tasa de código 5/6.
- IEEE 802.11ac: estándar para la banda de 5 GHz que incrementa la tasa a 360 Mbps aunque de modo teórico sería capaz de alcanzar 1.3 Gbps con el uso de un array de 3 antenas. Otra de las ventajas que posee es que la banda de 5 GHz es una banda limpia que no coexisten con otras tecnologías de comunicación como Bluetooth o ZigBee. De igual modo que en el estándar anterior para conseguir la tasa máxima es necesario el uso de OFDM (469 sub-portadoras), MIMO de 4 antenas y canales de 40 MHz, pero en este caso la modulación es 256QAM con tasa de código 3/4.

A través de esta tecnología y haciendo uso de protocolos como User Datagram Protocol (UDP) y M2M podremos comunicar tanto los dispositivos que pertenecen a la misma red como los que no. El funcionamiento de ambos protocolos se detallará en el apartado 2.4.

Cada dispositivo con este tipo de conectividad tiene dos identificadores de gran utilidad. Por un lado, poseen un identificador único de 48 bits a nivel de enlace Media Access Control (MAC) que no solo nos permite identificar al dispositivo, sino también nos podría permitir realizar un control de admisión de los dispositivos en nuestra red. Por otro lado, a nivel de red existe la dirección Internet Protocol (IP), un identificador de 32 bits que nos identifica con respecto al resto de la red (IP pública) o dentro de nuestra red (IP privada).

2.2.2. Bluetooth

Bluetooth es otra de las tecnologías de comunicación más utilizada actualmente. Permite la transmisión de voz o datos y se encuentra incorporado en la gran mayoría de dispositivos móviles, por lo que no solo nos permitirá comunicar las distintas plataformas hardware descritas sino que también incluirá todos aquellos dispositivos móviles que lo incorporen. Los enlaces que se crean entre estos dispositivos son de bajo consumo y se realizan a través de radiofrecuencia, aplicando Adaptive Frequency Hopping (AFH) entre un total de 79 frecuencias distintas en la banda entre 2.4 GHz y 2.48 GHz con intervalos de 1 MHz, lo que nos permite dotar a nuestro sistema de seguridad y robustez, descartando todas aquellas frecuencias en la que se encuentre interferencia. Dependiendo de sus características existen distintos tipos de dispositivos Bluetooth en función de su potencia de transmisión, de la capacidad del canal o del rol que toman

(*master* o *slave*). En función de la potencia de transmisión, y por tanto de su alcance, un dispositivo puede ser de clase 1 (hasta 100m), 2 (hasta 20m), 3 (hasta 1m) o 4 (hasta 0.5m). Dependiendo de la capacidad del canal puede ser de versión 1.2: 1.2 Mbit/s, versión 2.0+EDR: 3 Mbit/s, versión 3.0+HS: 24 Mbit/s o versión 4.0: 32Mbit/s). Dependiendo del rol que tenga en la topología un dispositivo puede actuar como slave si está conectado a un solo dispositivo o como máster si se le permite conectarse a varios, permitiendo que ellos se conecten a él y se produzca un intercambio de información.

Cada dispositivo Bluetooth se identifica mediante una dirección única de 48 bits y mediante un nombre que facilita al usuario seleccionar a qué dispositivo debe conectarse. El procedimiento de conexión de un dispositivo *slave* a un dispositivo *master* se llama *"pairing"* y es tan simple como establecer desde el dispositivo *master* una vinculación con la dirección única del dispositivo *slave*. Una vez realizada dicha vinculación ambos almacenan la dirección del dispositivo al que se han conectado, permitiendo en sucesivas ocasiones la vinculación automática ágil.

2.2.3. ZigBee

Zigbee[9] es el nombre por el que se conoce al conjunto de protocolos de alto nivel de comunicación inalámbrica en Wireless Personal Area Network (WPAN) basados en el estándar IEEE 802.15.4. Al igual que Wi-Fi y Bluetooth utiliza la banda de 2.4 GHz debido a que es una banda libre a nivel mundial, haciendo uso de hasta 16 canales de ancho de banda 5 MHz. Posee una velocidad de transmisión no superior a 250 Kbps en función del dispositivo y su rango de cobertura se encuentra entre 10 y 90 metros. El consumo energético puede llegar a oscilar entre 55 mA y 215 mA dependiendo del modelo del dispositivo y el estado en el que se encuentra (transmitiendo, recibiendo o en espera de datos). Para reducir aún más este consumo el protocolo Zigbee incorpora un cuarto estado (dormido) en el que gran parte del dispositivo está apagado y exclusivamente se mantiene en marcha un *timer* o una lectura digital que, llegado el momento, genere el evento de entrar en uno de los otros tres estados. En cuanto al número de dispositivos que se pueden incluir en cada red Zigbee es de 2^{16} nodos, aunque este número se ve reducido por restricciones físicas de los dispositivos y por el rango frecuencial en el que opera, como la memoria o el ancho de banda disponible. Basado en el nivel físico y el de control de acceso al medio (MAC) cada dispositivo está caracterizado por una dirección única de 64 bits que lo identifican en la red.

Los dispositivos Zigbee se pueden clasificar en tres tipos dependiendo de la función que tengan en la topología de red:

- Coordinador: nodo único con la función de formar la red. Responsable de realizar el control de la red y la conexión de los dispositivos.
- Router: nodo encargado de crear el enrutado de cada paquete de información. Interconecta dispositivos separados en la topología de red.
- Dispositivo final: nodo que suele estar conectado a una batería y se encarga del envío de información al coordinador o router al que está conectado. Puede estar dormido gran parte del tiempo para aumentar la vida de la batería.

Como cualquier tecnología de conexión existen diferentes topologías de red posibles (estrella, malla o árbol), cuyas estructuras se muestran en la Figura 2.5.

Debido sobre todo a su bajo consumo y su facilidad en el uso de topologías de malla es una solución ideal para un gran número de aplicaciones en las cuales la capacidad de transferencia es baja, como por ejemplo en domótica.

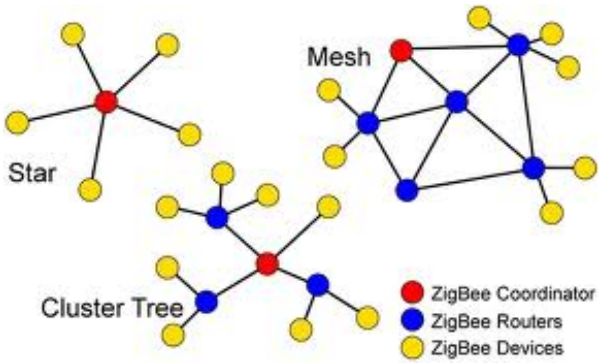


Figura 2.5: Topologías de red de Zigbee

2.3. Periféricos

El uso de una placa con procesador como unidad central no tiene ningún sentido si no se dispone de información que procesar. Por esta razón será necesario que añadamos a nuestra plataforma hardware los periféricos adecuados para cumplir con nuestros requisitos. Diferenciaremos dos tipos de periféricos: por un lado, los sensores y actuadores, que serán los encargados de recoger información del entorno o de actuar en él; por otro, los periféricos que nos permiten el uso de las tecnologías de comunicación para establecer una comunicación con otro dispositivo.

2.3.1. Sensores y actuadores

Como ya hemos dicho, son los encargados de interactuar con el entorno de la plataforma hardware. A menudo son de pequeño tamaño y bajo precio pero constituyen un bloque fundamental sin el cual prácticamente ningún proyecto de IoT podría realizarse. A través del elemento al que están conectados deben ser accesibles en cualquier momento y lugar.

Hasta ahora hemos explicado de forma intuitiva qué es un sensor y qué un actuador. Formalmente un sensor es un elemento que puede medir una propiedad física, como el nivel de luz de una habitación o la distancia a la que se encuentra un objeto, y la convierte en una señal eléctrica. Mediante los pines analógicos o digitales de nuestra placa somos capaces de medir dicha señal.

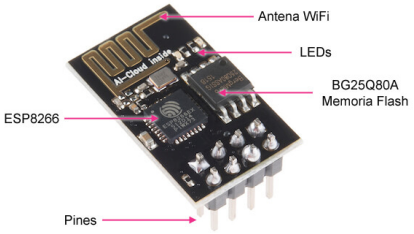
En cuanto a los actuadores, son también elementos del mundo IoT capaces de transformar una señal eléctrica en un cierto comportamiento de cara a su entorno, como el encendido de una luz o la vibración de un zumbador. Al igual que medir una señal nuestra placa también es capaz de generarla por lo que cada uno de los actuadores estará conectado a nuestra placa y su comportamiento dependerá de ella. Sin embargo esta no es una visión completa de los actuadores. Deberemos incluir también en esta categoría todos aquellos elementos sobre los cuales se puede imponer un comportamiento como un mecanismo o incluso una aplicación móvil. Aquellos sensores y actuadores que hemos estudiado se describen en el Apéndice B.

2.3.2. Elementos de comunicación

Asociados a las tecnologías de comunicación explicadas en el apartado 2.2 y partiendo de la hipótesis de que la placa seleccionada no dispone de la conectividad deseada, será necesario incorporar a ella periféricos dedicados a la comunicación, que se describen a continuación.

2.3.2.1. ESP-01 Wi-Fi Module

El ESP-01[10] es un módulo programable fabricado por Espressif que incorpora un microcontrolador ESP8266 y que nos permite dotar a nuestra placa de conectividad Wi-Fi 2.4 GHz (802.11 b/g/n) de una forma relativamente simple y económica.



Protocolo	802.11 b/g/n
Frecuencia	2.4 GHz
Protocolos de red	IPv4 y TCP/UDP/HTTP
Potencia de transmisión	Configurable entre +14 y +20 dBm
Sensibilidad de recepción (dBm)	-91
Voltaje (V)	3.0 - 3.6
Consumo energético medio (mA)[6]	80
Temperatura de operación (°C)	Entre -40 y +125
Dimensiones (mm x mm)	24.75 x 14.5

Figura 2.6: Módulo Wi-Fi ESP-01 y características

Al ser un módulo programable sería posible configurarlo sin necesidad de estar conectado a una placa. Sin embargo dado que nuestro principal propósito con él es el intercambio de información obtenida a través de los distintos periféricos conectados a una placa sí será necesaria dicha conexión. El módulo incluye ocho pines ya soldados que nos facilitan el conexionado con la placa: alimentación, transmisión y recepción, *reset*, pin de encendido del módulo y dos pines digitales GPIO que podríamos utilizar para nuestro propósito si fuera conveniente.

La principal desventaja en el uso de dicho módulo es que su configuración es ligeramente compleja, siendo necesario modificar el conexionado del módulo a la placa para programar por un lado el comportamiento de la conexión Wi-Fi en el módulo y por otro lado el comportamiento con los periféricos en la placa. Una vez configurados ambos, la comunicación módulo-placa se realiza a través de los pines serie de transmisión y recepción de datos dedicados para dicho propósito.

2.3.2.2. Bluetooth HC-05

El HC-05[11] es un módulo de pequeño tamaño que permite establecer comunicación con otro dispositivo vía tecnología Bluetooth.
Por sus características es un módulo Bluetooth cuya capacidad de canal es de versión 2 (tasa

1.2Mbps) y su potencia de transmisión de clase 2 (potencia de transmisión 2.5 mW y alcance hasta 20 m). La conexión con la placa se realiza mediante los pines de transmisor y receptor del mismo modo que ocurría con el ESP-01, pero en este caso existe también una conexión

[6] El consumo energético varía enormemente desde 10μA si está en estado dormido a 170 mA si está transmitiendo con la máxima potencia.

Protocolo	802.15.1
Frecuencia	2.4 GHz con FH Spread Spectrum
Potencia de transmisión	4 dBm
Sensibilidad de recepción	-80 dBm
Voltaje	4-6 V
Consumo energético medio	30 mA
Tasa media (Mbps)	1.2
Rango máximo (m)	hasta 10 m
Modos de funcionamiento	AT o comunicación (master o slave)
Dimensiones (mm x mm)	37.5 x 16.1

Figura 2.7: Módulo Bluetooth HC-05 y características

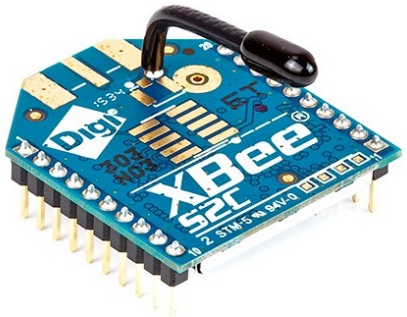
KEY que nos permite, en caso de conectarla a GND, configurar el módulo vía comandos AT. Con el propósito de conocer de una forma sencilla el modo de funcionamiento en el que nos encontramos, el módulo incorpora un LED que posee un modo de parpadeo asociado a cada uno de los estados: configuración por comandos AT (lento), buscando conexión (rápido) o conectado (doble parpadeo). En cuanto a su alimentación, se realiza mediante los pines +5V y GND.

2.3.2.3. Módulos XBee

Los módulos XBee[12] ofrecen una solución más profesional y completa mediante el uso de la tecnología de comunicación Zigbee que describimos anteriormente. Son fabricados por Digi[13], altamente configurables mediante el entorno gráfico XCTU y utilizan la tecnología Zigbee RF para la transmisión y recepción de datos. Su principal utilidad es la implementación en entornos que no incluyan una red inalámbrica a la que conectarnos, generando nosotros una propia. Una de sus principales desventajas se encuentra en su conexión, ya que es altamente recomendable adquirir un elemento intermedio que nos permita conectar nuestro módulo al Arduino, lo que aumenta el precio de nuestro diseño.

Los módulos XBee permiten dos modos distintos de funcionamiento, pudiendo funcionar en ambos o no dependiendo del modelo escogido. En el modo transparente los módulos se configuran mediante comandos y únicamente actúan de pasarela entre los pines transmisores o receptores del módulo y la placa mediante el puerto serie. En cuanto al modo API es un modo mucho más completo que incorpora el tratamiento de los datos permitiendo conocer información tan valiosa en ciertos casos como el origen de nuestra comunicación, así como otras funcionalidades de gran utilidad. La versatilidad que nos ofrecen estos módulos permite que sean configurados de formas muy distintas. En nuestro caso, realizaremos la identificación de los módulos mediante una dirección MY de 16 bits única y de la red mediante el identificador PAN ID de 16 bits que deben compartir unicamente aquellos módulos que pertenezcan a la misma red, lo que nos permitiría incluso tener varias redes distintas coexistiendo. Para el envío de los paquetes usaremos las direcciones de 16 bits Destination Address High (DH) y Destination Address Low (DL), donde DH tomará siempre el valor 0 y DL se corresponderá con la dirección MY del nodo destino (o 0xFFFF en caso de que queramos realizar una transmisión broadcast).

Al contrario que ocurría con los módulos Wi-Fi o Bluetooth, donde no encontrábamos un amplio abanico de opciones, existen una gran cantidad de módulos XBee distintos [?] . Dependiendo de factores como el país en el que va a encontrarse el sistema, el rango de alcance de señal necesario, el consumo que podamos permitir y la topología de red debemos elegir uno u otro. En nuestro caso seleccionamos un módulo XB24C [14] y se configura según el Firmware 802.15.4 TC debido a que incluye una optimización en su consumo, se ajusta a la regulación de gran parte del mundo (Europa, América del Norte y Australia), permite el estudio de distintas topologías así como de distintos modos de funcionamiento (transparente y API) y, además, posee una antena que mejora el rango en exteriores.



Protocolo	802.15.4
Frecuencia	2.4 GHz
Tasa	250 Kbps
Alcance	$\simeq$ 60m en interior $\simeq$ 1km en exterior
Potencia de transmisión	5 dBm
Sensibilidad de recepción	-100 dBm
Modos configurables	AT o API
Memoria flash	32 Kb
Memoria RAM	2 Kb
Voltaje	2.1 - 3.6 V
Intensidad	47 mA (TX) 42 mA (RX) 1.5 μA (idle)

Figura 2.8: Módulo ZG24C y características

2.4. Plataforma software

El análisis, procesado y visualización de los datos constituyen otro de los bloques de nuestro ecosistema IoT. A pesar de que no es expresamente necesario disponer de él, sí es una buena opción incorporarlo en mayor o menor medida puesto que ofrece a nuestro sistema nuevas funcionalidades. Gran parte de las posibilidades que la plataforma software nos ofrece se basan en hacer uso de los tecnologías de comunicación Wi-Fi o Ethernet para conectarnos a la red y, con ello, a multitud de servidores y bases de datos. Es aquí donde encontramos la principal razón para que, aunque también caben en nuestro sistema otras tecnologías de comunicación, las tecnologías Wi-Fi o Ethernet sean necesarias en nuestro sistema en caso de que queramos hacer uso de dicha plataforma software. La cantidad de elementos que pueden ser incluidos en la plataforma software es muy grande. A continuación se destacan aquellos que resultan más interesantes para desarrollar metodologías que abarquen la mayoría de las posibles aplicaciones.

2.4.1. Protocolo UDP

El protocolo UDP[15] es un protocolo del nivel de transporte basado en el intercambio de datagramas a través del nivel de red. Por definición no es necesario el establecimiento de la

conexión, pero por contra no garantiza que los paquetes enviados hayan llegado correctamente al destino. Como ventajas sobresalientes de UDP destacan su rapidez en el envío de mensajes y la facilidad de implementación en Arduino por encima de otros protocolos de transporte como Transmission Control Protocol (TCP)[16], protocolo que deberíamos implementar en caso de querer disponer tanto del establecimiento de como la confirmación de recepción de una forma sencilla. En el Apéndice A.1 se detalla como es posible aprovechar las ventajas que nos ofrece UDP adaptándolo hasta conseguir las funcionalidades del protocolo TCP.

2.4.2. Protocolos M2M

Los protocolos M2M[17], al igual que UDP, permiten el intercambio de información entre dos dispositivos remotos. Son protocolos muy usados en el IoT debido a que consumen muy poco ancho de banda y se pueden utilizar en dispositivos con pocos recursos de Central Processing Unit (CPU) o Random Access Memory (RAM). Algunos ejemplos de estos protocolos son Message Queue Telemetry Transport (MQTT), Constrained Application Protocol (CoAP) o Advanced Message Queuing Protocol (AMQP), etc. A la hora de escoger uno de ellos se plantea el uso de MQTT[18] o de CoAP [19, 20]. Para escoger la opción más adecuada para nuestro proyecto se realizan pequeñas pruebas de concepto con ambas, A pesar de que la documentación hacía pensar que CoAP iba a ser una mejor opción, finalmente nos decantamos por MQTT debido a su mayor rapidez de envío, su versatilidad y su fácil implementación tanto en los sistemas de sensorización como en el nodo central.

MQTT es por tanto un protocolo que funciona a nivel de aplicación asíncrono, basado en TCP y con diferentes niveles de Quality of Service (QoS). El tamaño de sus paquetes varía entre tan solo 2 bytes si no enviamos información y 256 MB, aunque es recomendable el envío de paquetes no superiores a 127 bytes. Su topología es de estrella, donde todos los dispositivos se comunican con un nodo central que funciona como servidor, también llamado broker, que puede encontrarse en remoto compartido o en un dispositivo de nuestra topología. El límite de dispositivos que podemos conectar a un solo nodo central ronda los 10.000, un valor que permite crear topologías de red de gran tamaño. Además las comunicaciones pueden ser bien *One-To-One* o *One-To-Many* ya que la información se envía a un “topic" al que cualquier dispositivo autenticado podría suscribirse y recibir sus cambios. Este modo de funcionamiento lo podemos ver de forma simple en la Figura 2.9. En cuanto a la seguridad funciona con SSL/TLS, los usuarios se autentican con la dupla (*Username*, *Password*) y además el contenido del mensaje se envía encriptado.

2.4.3. Firebase Real-Time database

Otra de las funcionalidades que una aplicación IoT debería tener acceso a una base de datos. La lista de posibilidades es extensa y las razones por las que elegir una u otra son variadas.

En 2016 la compañía Google lanzó una versión renovada y mejorada de la plataforma Firebase, una plataforma que permite crear y desarrollar aplicaciones multiplataforma para dispositivos móviles de forma sencilla. Dentro de esta plataforma se incluye Firebase Real-Time database, que se define como “una base de datos NoSQL alojada en la nube en la que los datos se almacenan en formato JSON y se sincronizan en tiempo real, manteniéndose disponibles incluso cuando la app no tiene conexión" [21]. Su uso es gratuito y su estructura lo hace altamente escalable. En cuanto a su integración con dispositivos de menor tamaño a bajo nivel las lecturas y escrituras se realizan mediante sentencias *GET* o *PUT* de *HTTP*. Esta gran ventaja

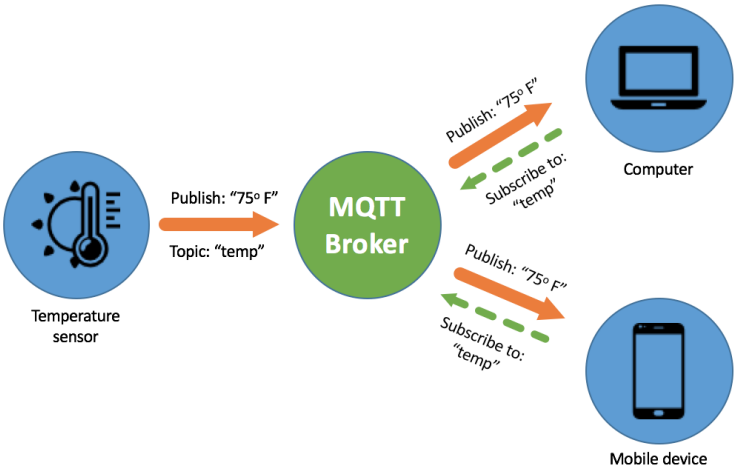


Figura 2.9: Topología simple MQTT

unida a la gran comunidad nos permitirá interactuar con la base de datos directamente desde cualquier plataforma hardware con conectividad a Internet.

Como hemos dicho esta base de datos se integra dentro del ecosistema de Firebase del que podemos integrar en nuestro sistema otras funcionalidades como la autenticación para dotar de seguridad a nuestra base de datos o el uso de reglas para delimitar los permisos de acceso de cada dispositivo a ella.

2.4.4. IFTTT

IFTTT (si ocurre "esto", entonces haz "esto") es un tipo de servicio web que permite la automatización de acciones en función de eventos ocurridos en otro servicio. La plataforma IFTTT, que relaciona ambos servicios, es y debe ser segura debido a la información que maneja. La plataforma es gratuita aunque dependiendo de los servicios que hacemos uso existen ciertos límites, lo que limita pero no impide su uso.

En nuestro caso de estudio del proyecto que se plantea nos interesa generar acciones en función de eventos ocurridos en nuestros dispositivos. Por tanto el evento reactivo que debemos tratar se enviará unicamente mediante un servicio llamado Webhooks, que no es otra cosa que el envío de una Web-Request mediante una sentencia POST HTTP. La estructura de esta sentencia y el paso de datos en el cuerpo de la petición se observa en la Figura 2.10 junto con el modo de actuar en dos de los servicios sobre los que podemos actuar y que más interés

tienen para nuestro proyectos: el envío de una notificación vía e-mail y el almacenamiento de mediciones en Google Sheets.

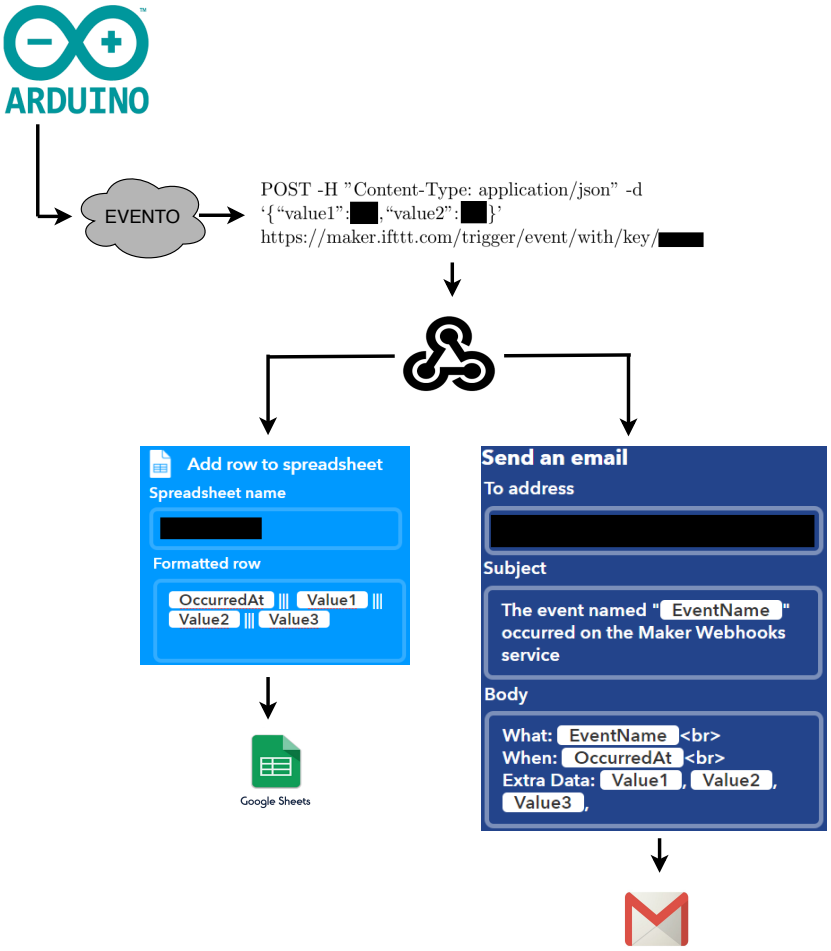


Figura 2.10: Ejemplo de aplicación de IFTTT

Capítulo 3

Diseño del entorno de eficiencia energética

Las previsiones en cuanto al crecimiento de número de dispositivos conectados constituye la principal razón para el desarrollo de la tecnología de IoT. Como aspecto negativo, este incesante incremento supone un gran número de dispositivos fabricados cada año, con los efectos dañinos para el ecosistema que ello conlleva. Todo dispositivo fabricado contamina desde el momento en el que comienzan a fabricarse sus componentes y el propio dispositivo, pasando por la producción de la energía necesaria para cargar su batería periódicamente hasta el momento en el que, en la mayoría de los casos, sus componentes no son reciclados adecuadamente una vez acabó su ciclo de vida.

La tecnología contribuye a la destrucción del mundo en el que vivimos, aunque por suerte este problema es parte de la solución. Una solución que comienza con el uso de estos avances tecnológicos aplicados a la monitorización, mediante redes de sensores, diversos ecosistemas que abarcan desde el ámbito doméstico (Smart Home) hasta una ciudad al completo *(Smart City)*. De entre los posibles ecosistemas, tomamos como ejemplo un espacio de oficinas con el fin de conseguir un aumento de la eficiencia energética en él. Comenzaremos con una pequeña sala de reuniones con una perspectiva escalable que nos permita aplicarlo a un bloque entero de oficinas.

3.1. Definición del entorno y necesidades

Hoy en día el consumo eficiente de energía se ha convertido en un gran reto para las grandes compañías, tanto por el ahorro a nivel económico como por la reducción de la contaminación, para lo cual la aplicación de soluciones IoT es su mejor aliado. En nuestro caso particular partimos de la base de que a diario el consumo de electricidad y agua en cualquier espacio en el que viven, trabajan o se divierten las personas es superior al necesario por diversas razones. Por un lado, hay un derroche energético voluntario motivado por querer encontrarse en una situación de confort superior a la estrictamente necesaria. Por otro lado, también existe un gasto involuntario y por tanto completamente innecesario debido a que no es sencillo el desarrollo de una lógica de control de sistemas como la climatización o iluminación sin una correcta monitorización de los parámetros ambientales de las distintas zonas del espacio. Es aquí donde resulta interesante diseñar una solución IoT, la cual debe consistir en la monitorización de nuestro espacio junto con la correcta lógica de control eficiente y su correspondiente actuación sobre los sistemas asociados[22]. Para conseguirlo, haremos uso de la plataforma hardware Arduino, cuyas principales ventajas son el amplio abanico de productos que ofrece y su alta compatibilidad con la gran mayoría de los sensores comercializados. En cuanto a la interconexión entre

plataformas hardware, nos decantamos principalmente por el uso de Wi-Fi como la tecnología de comunicación más adecuada, ya que consideramos que en cualquiera de los posibles espacios a implementar existirá y que su calidad de señal será más que aceptable. Puntualmente se planteará el uso también de la tecnología Bluetooth para ciertas funcionalidades que veremos más adelante. Apoyándonos sobre esta conexión Wi-Fi y de acuerdo a lo razonado en el capítulo anterior, explotaremos las comunicaciones M2M a través del uso de MQTT y emplearemos como base de datos Firebase Real-Time junto con su funcionalidad de autenticación. En cuanto al espacio concreto a implementar, se escoge como modelo el propio bloque de oficinas en el que se desarrolla este proyecto.

Con todo este ecosistema los dos frentes de ahorro más significativos son el consumo de agua y la electricidad, tanto de la iluminación como de la climatización, aunque dicho ahorro no se debe buscar a cualquier precio. Cualquier acción realizada no debe comprometer la productividad de ninguna de las personas que se encuentra trabajando, así que se analizarán minuciosamente estas acciones buscando incluso mejorar la situación de confort de esos trabajadores. Además, cabe destacar que durante la realización de este proyecto no resulta viable la modificación de la instalación de agua o electricidad de las zonas a prototipar por lo que los valores de ahorro se deberán estimar de la forma más objetiva posible y se deberá presuponer cual será la instalación que se realizará en la fase de implementación. Para tratar de obtener datos totalmente experimentales de la parte de la lógica de control se realizará también el diseño y toma de datos de otras sensorizaciones con las que poder extraer valores reales del ahorro en el consumo. Sin embargo, estos nuevos diseños no abarcarán el total de los casos de ahorro por lo que parte de ellos se deberán estimar de una forma totalmente teórica.

A continuación, en los apartados 3.1.1 y 3.1.2, detallamos las variables a monitorizar y cómo a través de ellas obtenemos el ahorro en nuestro consumo. Será en 3.4 donde veremos como se obtiene la información.

3.1.1. Consumo de agua

En lo referente al consumo de agua, descartamos actuar sobre inodoros y nos centramos en el ahorro del agua proveniente de los grifos instalados en los aseos. Si pensamos en nuestra vida cotidiana podemos darnos cuenta de que no existe una regla fija del tiempo que invertimos en las labores de higiene personal y, por tanto, en la cantidad de agua que consumimos. Este consumo supera, de forma inconsciente, el estrictamente necesario. Para solventarlo se realiza en primer lugar un estudio sobre los hábitos de consumo de las personas que trabajan en las oficinas, para desarrollar posteriormente un sistema de apagado automático de los grifos instalados en los aseos. Este sistema consistirá fundamentalmente en el apagado del grifo transcurrido un determinado tiempo, que viene determinado por el estudio, tras la detección de que el usuario se ha situado frente al grifo.

3.1.2. Consumo de electricidad: climatización e iluminación

Con relación al consumo de electricidad existen múltiples variables a controlar y sobre las que actuar en las diversos zonas del espacio. En nuestro estudio consideramos por un lado las zonas de aseos y por otro las zonas de reuniones, teniendo siempre una visión escalable a un espacio de mayor amplitud como es la zona de trabajo de una oficina.

En primer lugar, tratamos la zona de los aseos, en la cual también controlaremos la iluminación. Es muy común en grandes bloques de oficinas que el encendido o apagado de la iluminación se realice manualmente a través de interruptores, lo que hace que esta se pueda

mantener encendida incluso cuando no hay nadie. Con el fin de solucionar esto se diseñará un sistema de apagado automático de la iluminación mediante sensores de infrarrojos que nos permiten detectar si existe movimiento. Además, como dijimos con anterioridad no se encuentra dentro del alcance del proyecto la modificación de la instalación eléctrica por lo que en este caso diseñaremos también otro bloque que nos permita calcular durante cuánto tiempo la iluminación está realmente encendida con el sistema de iluminación actual. Con ello, el cálculo del ahorro es trivial.

Por otro lado, tratamos pequeñas zonas de reuniones que se encuentran aisladas de una zona de trabajo más amplia. Las razones de tomar estas pequeñas zonas como piloto son el no alterar el ambiente de trabajo y la facilidad para adaptar la sensorización de esta zona de reuniones a la zona de trabajo. La zona de trabajo, que es mucho más amplia, se concibe como una agrupación de otras zonas de tamaño similar a nuestra “zona piloto" en las que no es necesaria una sensorización tan completa. Por ello una vez dispongamos de una zona de reuniones totalmente monitorizada y haya sido establecida la lógica de control, la adaptación a otras zonas más simples no será compleja.

Respecto a la iluminación, la actuación radica en el control de las luminarias, presuponiendo que la instalación en la implementación del sistema constará de iluminación regulable de forma remota. Gracias a ello, en función de la lógica se establecerá como se debe regular cada una de las luces de la zona. Respecto a la climatización, nos basaremos en la monitorización para establecer la cantidad de energía necesaria para regular la temperatura de nuestro espacio. Como actuación añadida se tendrá también en cuenta el efecto de las persianas motorizadas que afectará al nivel de luminosidad de la zona y aislará el interior del exterior del calor por radiación y transmisión.

Para nuestro bloque de lógica de control se diseña un bloque funcional que, en función de unas entradas determinadas, genere las correspondientes salidas. Estas entradas se basan no solo en la monitorización, sino también en otros valores constantes como el número de personas de la zona a estudiar o la calidad de las ventanas. En nuestro caso, la información que nos interesa extraer de la sensorización y que formará parte de estas entradas es: la temperatura, la iluminación (tanto del interior como del exterior), la humedad en cada momento y si las ventanas o puertas están abiertas. Para cada una de estas información será necesario elaborar un bloque funcional con los correspondientes sensores.

3.2. Selección de los sensores

Dado que el mundo del IoT está en crecimiento existen múltiples sensores para cada variable a medir. Tras estudiar para cada una de ellas de una forma teórica distintas opciones[23, 24] se realiza una selección de sensores con el fin de comparar de forma práctica sus prestaciones. A la hora de seleccionar los sensores, se establecerá una relación entre la precisión de la medida y su precio que nos permita escoger el sensor adecuado en cada situación. Por contra, en otros casos no se considera relevante la comparación de varios sensores o bien porque las opciones son escasas o bien porque la diferencia de precio entre ellas, para unas prestaciones similares, es suficientemente grande como para que sean descartadas.

Como herramienta para determinar la precisión de las medidas se realiza un estudio estadístico cuya fuente de datos es la sensorización de las distintas variables durante 24 horas, un tiempo muy amplio pero necesario para evaluar la precisión en los distintos rangos de valores que tomarán las variables. Puesto que no se tiene como referencia cuáles son los valores a los

cuales se deberían aproximar los datos, tomaremos como referencia los valores medios en cada momento. A partir de ellos obtendremos la desviación de cada una de las medidas de los sensores con respecto a la media a lo largo de tiempo. El sensor con mayor precisión será aquel cuya desviación sea menor a lo largo del tiempo, para lo cual calculamos la media y desviación típica de las desviaciones de cada sensor. Aquel que tenga menor media será el sensor que más se aproxima a nuestros valores de referencia; además, será necesario comprobar que la desviación típica obtenida también sea mínima para asegurar que las medidas obtenidas se aproximan a la referencia a lo largo del tiempo. Adicionalmente y dado que los datos climatológicos de temperatura y humedad de las distintas estaciones meteorológicas que hay en cada ciudad se publican diariamente, tomamos los datos de la más cercana a nuestro emplazamiento, que se encuentra aproximadamente a diez kilómetros, como referencia en cuanto a la tendencia que los datos deberían seguir.

Luminosidad

En el caso de la luminosidad se comparan dos sensores (TSL2561[25], BH1750[26]) junto con un simple fotorresistor GL55[27]. Se realiza un diseño de conexionado común para estos tres sensores y se extraen los datos de luminosidad durante 24 horas en un ambiente de exterior. El esquema de conexionado se muestra en la Figura 3.1. En la Figura 3.2 se comparan los resultados.

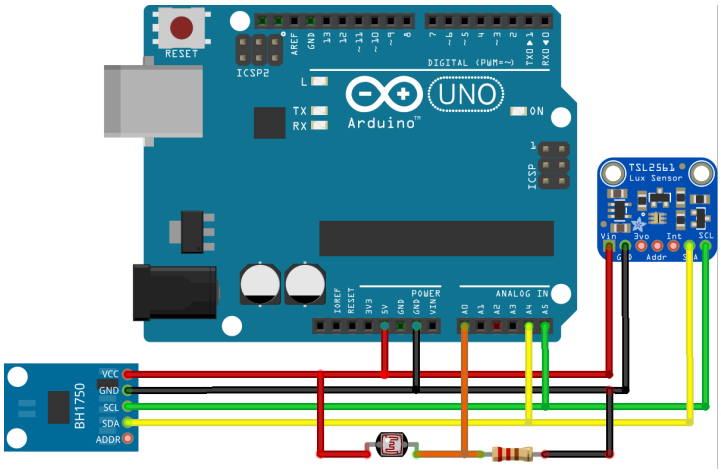


Figura 3.1: Esquema de conexiones del bloque de selección de sensores de luminosidad

Al contrario de lo que ocurre con la temperatura o la humedad, para comparar la luminosidad no se dispone de ninguna referencia. Por ello la decisión se debe tomar en función de las hojas de características y de los resultados obtenidos sin posibilidad de realizar grandes estadísticas. En la comparativa de los resultados se termina de certificar que el fotorresistor no aporta medidas fiables puesto que, aunque la tendencia que sigue es la correcta, los resultados no son correctos. Respecto a los otros dos sensores, se obtienen resultados similares en ambos casos. Al realizarse las medidas en una habitación de cara al exterior se puede apreciar que a partir del medio día los datos registrados son elevados puesto que la radiación solar recibida es directa. Gracias a ello con este estudio somos capaces de seleccionar el sensor más adecuando tanto para la sensorización de la luminosidad exterior como la interior. Para ambos casos se obtiene la desviación porcentual de la media obtenida de estos dos sensores respecto a la medida de cada sensor. En el rango de luminosidad que encontraremos en una oficina (entre los 200 y los 500 lux) las desviaciones son suficientemente parecidas como para escoger el sensor

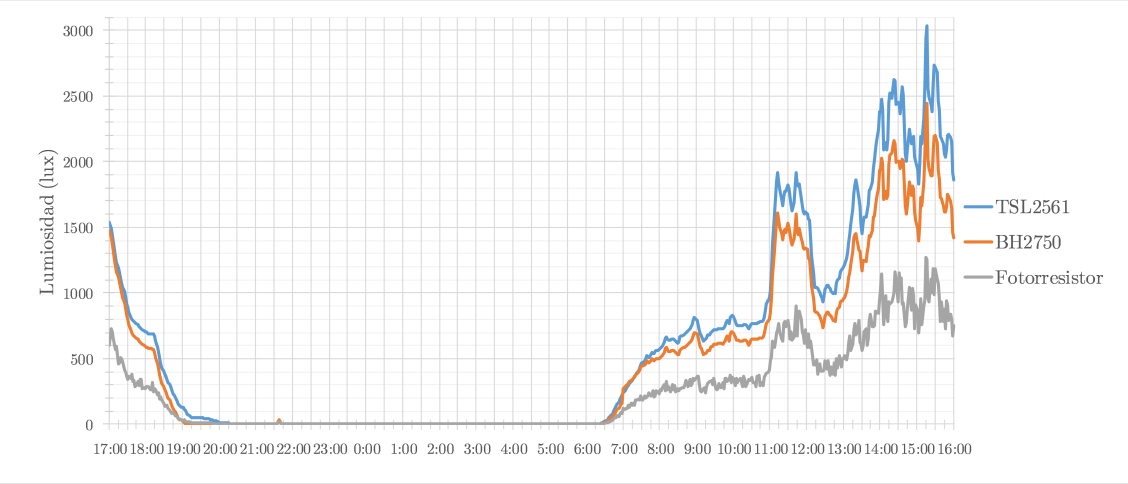


Figura 3.2: Comparación de los resultados de la selección de sensores de luminosidad

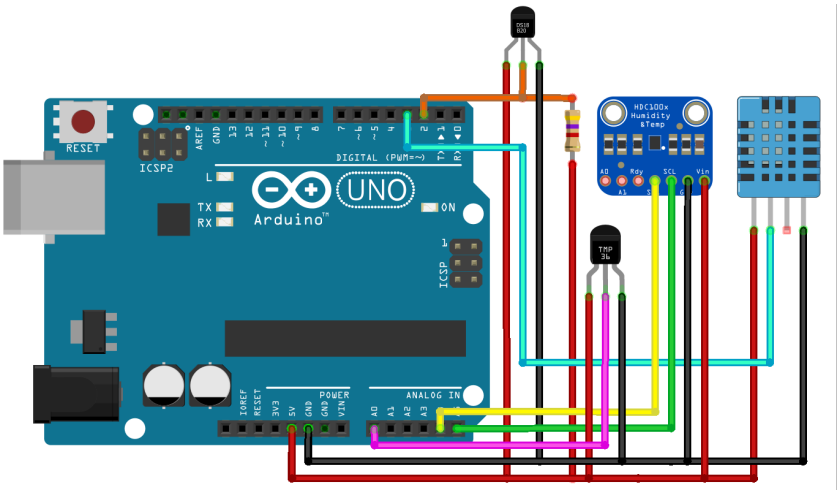


Figura 3.3: Esquema de conexiones del bloque de selección de sensores de temperatura y humedad

BH1750 por tener un precio ligeramente inferior. En cuanto al rango de luminosidad exterior la desviación máxima es del 12.16 y 16.1 % para los sensores TSL2561 y BH1750 respectivamente, por lo que se escoge el primero para ello.

Temperatura y humedad

En cuanto a la temperatura y la humedad, gran parte de los sensores son capaces de tomar medidas de ambas variables por lo que con un mismo bloque podemos realizar ambas comparativas. Utilizaremos cuatro sensores: los sensores DS18B20[28] y TMP36[29], que nos permitirán obtener exclusivamente el dato de temperatura, y los sensores HDC1008[30] y DHT11[31] de los que obtendremos tanto temperatura como humedad. Al igual que en el caso anterior recogemos datos durante 24 horas en el mismo ambiente. La Figura 3.3 muestra el esquema de conexionado conjunto, las Figuras 3.4 y 3.5 los resultados de la comparativa de temperatura y humedad respectivamente y en la Tabla 3.1 encontramos las estadísticas respecto a la temperatura de las que hablábamos.

Respecto a la temperatura, si observamos los cuatro sensores utilizados siguen el patrón de la referencia. Sin embargo dos de ellos son los que, además de seguirlo, tienen unas medidas que

se parecen mucho entre sí frente a los otros dos, que tienen unas medidas mucho más dispersas. Por mayor seguridad se calculan las correspondientes estadísticas tomando como referencias tanto la media de los valores tomados en cada instante como la referencia de la estación meteorológica.

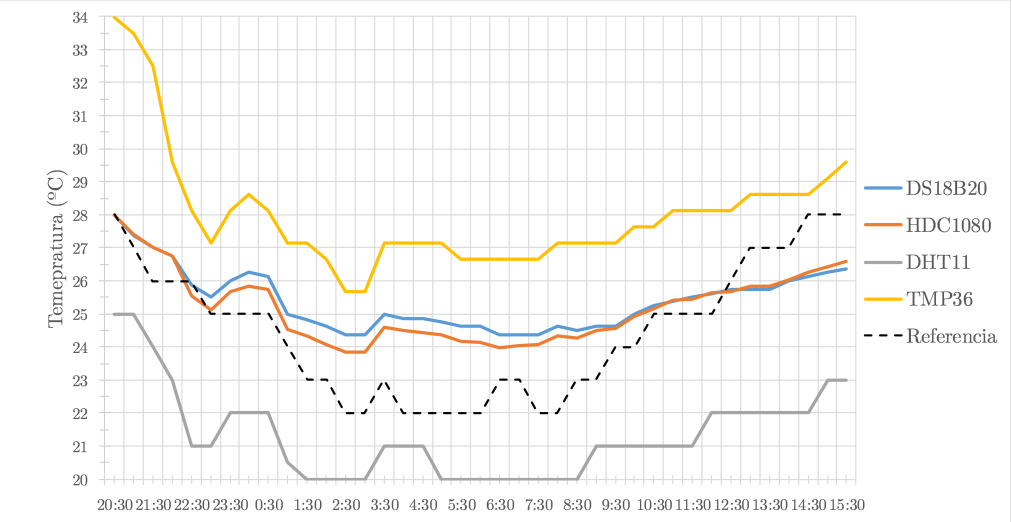


Figura 3.4: Comparación de los resultados de la selección de sensores de temperatura

Los resultados obtenidos son claros, siendo los sensores HDC1080 y DS18B20 los más aptos para la monitorización de la temperatura. Ambos poseen la menor media y desviación típica, tanto respecto a la referencia promedio como a la referencia de la estación. Entre ellos es el sensor HDC1080 el que otorga una mejor precisión, convirtiéndose en el sensor de temperatura que utilizaremos para tomar las medidas del interior. En cuanto al DS18B20, su buena precisión y su capacidad impermeable lo convierten en el escogido para la sensorización de la temperatura exterior.

	Respecto a la media		**Respecto a la referencia**	
	Media	**Desviación típica**	**Media**	**Desviación típica**
DS18B20	0.53	0.23	1.39	0.84
HDC1080	0.33	0.17	1.16	0.7
DHT11	3.65	0.26	3.41	1.22
TMP36	3.03	0.64	3.54	1.43

Tabla 3.1: Estadística sensores de temperatura

En cuanto a la humedad la toma de la decisión es mucho más simple dado que, al tratarse de un valor que no afectará a la eficiencia energética, solo se estudian dos sensores que nos aporten dicha información. El estudio que se realiza no tiene como objetivo determinar cual es el sensor de mayor precisión sino que se plantea como una verificación de la idea preconcebida de que las prestaciones del HDC1080 son altamente superiores a las del DHT11.

Tomando como referencia el dato de humedad de la estación meteorológica, resultan mucho más fiables los datos obtenidos del sensor HDC1080 frente a los del DHT11, teniendo también en cuenta la precisión de las medidas en cuanto a la temperatura se refiere. Es por ello que,

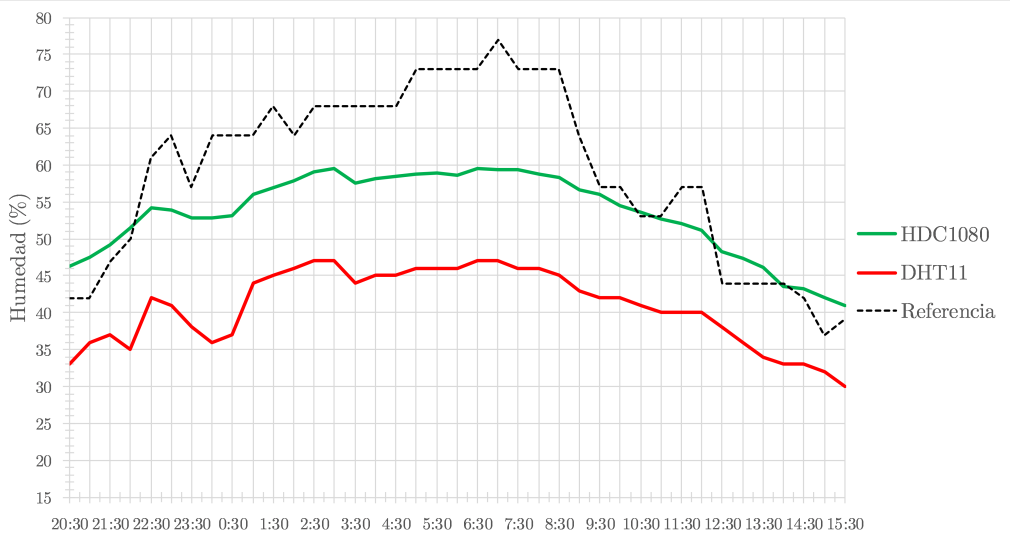


Figura 3.5: Comparación de los resultados de la selección de sensores de humedad

además de para la sensorización de la temperatura, también utilizaremos el HDC1080 como sensor de humedad a partir de ahora.

Otros sensores

Adicionalmente utilizaremos sensores de ultrasonidos HC-SR04[32] para calcular la distancia a la que se encuentra un objeto, sensores PIR[33] para detectar movimiento en una zona concreta y sensores magnéticos[34] que nos permitan detectar si puertas o ventanas están abiertas.

Una descripción más detallada de las características (voltaje de alimentación, de funcionamiento, pines,...) de cada uno de estos sensores se encuentra en el Apéndice B.

3.3. Gestión de datos, procesado y visualización

Toda la información obtenida debe ser procesada y visualizada adecuadamente, para lo cual se diseñan los nodos de procesado y visualización.

La gestión de los datos corresponde a una base de datos Firebase Real-time y al protocolo M2M MQTT. Es necesario diferenciar qué información se transmite por cada uno de ellos debido a que la base de datos tiene una capacidad y número de peticiones finitos. Para ello, diferenciamos entre aquella información que debe ser exclusivamente almacenada para su visualización, para lo que haremos uso de la base de datos de Firebase, y aquella cuyo principal propósito es ser una entrada del bloque de procesado a través del protocolo MQTT. En cuanto a las salidas de este nodo, se gestionarán también por esa vía. Parte de esas entradas y salidas se utilizarán también en el nodo de visualización.

La base de datos Firebase no necesita de ningún soporte físico que la albergue y, mediante las funcionalidades de autenticación y el uso de reglas, limitamos quién puede realizar lecturas o escrituras en cada uno de los nodos. La estructura de árbol que posee, la cual convierte la base de datos en escalable, se muestra en la Figura 3.6.

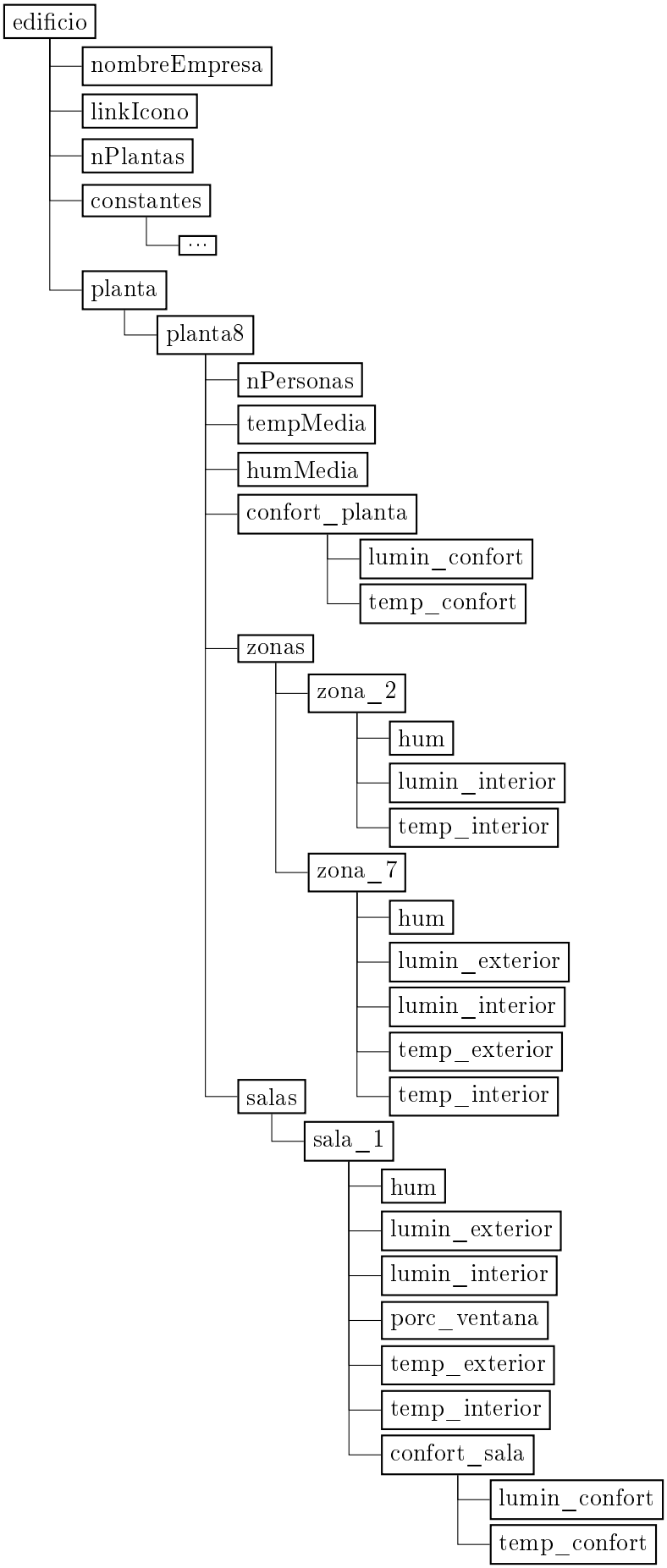


Figura 3.6: Estructura base de datos Firebase

Por simplicidad se prescinde de detallar el nodo “constantes" donde se encuentran un gran número de constantes empíricas obtenidas de la documentación para los cálculos de la lógica de control. La razón de guardarlas en nuestra base de datos y no definirlas en la propia lógica es que al ser empíricas se han obtenido de diversos estudios y puede ser interesante realizar modificaciones en el futuro para ajustarlas a un caso más realista. La posibilidad de realizar estos cambios de forma remota sin necesidad de modificar el código de la lógica dota a la base de datos de gran potencial.

Como se comentó en la explicación del protocolo MQTT existen tres roles diferenciados: clientes que publican información, clientes que se suscriben para recibirla y un “broker". El broker se aloja en la misma Raspberry Pi 3B+ en la que se encuentra el nodo de procesado, mientras que los clientes que publican y se suscriben serán las plataformas hardware que forman parte de sensorización y actuación respectivamente. Como resulta obvio, el nodo de procesado tomará los dos roles de cliente: por un lado se suscribirá para recibir toda la información de los sensores y por otro publicará las acciones para ser recibidas por los actuadores. Adicionalmente, el nodo de visualización también se suscribirá para mostrar parte de la información recogida por los sensores como las acciones enviadas desde el nodo de procesado.

El protocolo utilizado no fija la necesidad de preconfigurar una estructura en árbol de “topics" sobre los que podemos escribir. Sin embargo, es necesario tener claro el nodo sobre el que cada uno de los clientes debe publicar o suscribirse para evitar colisiones. Al igual que ocurría con Firebase se diseña una estructura para MQTT que permita su escalabilidad en un futuro. El resultado lo podemos ver en la Figura 3.7.

3.4. Diseño de la sensorización y de los bloques que la forman

Para controlar cada una de las variables de las que hemos hablado anteriormente es necesario definir un bloque funcional que nos permita extraer la información correspondiente.

Control de apagado de grifos en aseos

El objetivo de este bloque es controlar el apagado de los grifos de los aseos. Para conseguirlo utilizamos sensores HC-SR04, con los que medimos la distancia entre el grifo y la persona que lo usa. Analizando esta variable y definiendo un umbral que depende de la instalación podremos obtener la variable binaria de si hay alguien frente al grifo o no y con ello calcular durante cuánto tiempo el grifo ha sido utilizado. En nuestro caso consideramos que hay dos grifos en cada aseo, por lo que serán necesarios dos sensores.

En una primera fase obtendremos este tiempo de permanencia frente al grifo y lo almacenaremos a través de IFTTT en un Google Sheet. Tras tomar datos durante un tiempo suficiente, los analizamos y elaboramos una estadística, a partir de la cual definiremos el tiempo que un grifo debería estar en funcionamiento. Para la segunda fase y basándonos en ese tiempo calculado se diseña un bloque de sensorización similar al anterior, pero que en este caso incorporará la gestión de apagado del grifo. La máquina de estados con la que aplicamos la lógica de control necesaria es común para ambas fases (3.8), donde la condición *persona=1* es la variable binaria antes mencionada.

En la primera fase nos podemos encontrar en tres estados (*“off"*, *“detectado"* y *“no_ detectado"*), con los que controlamos los eventos generados cuando la persona se coloca frente al grifo y cuando se aleja, permitiéndonos extraer el tiempo que ha pasado e incorporando un sistema de

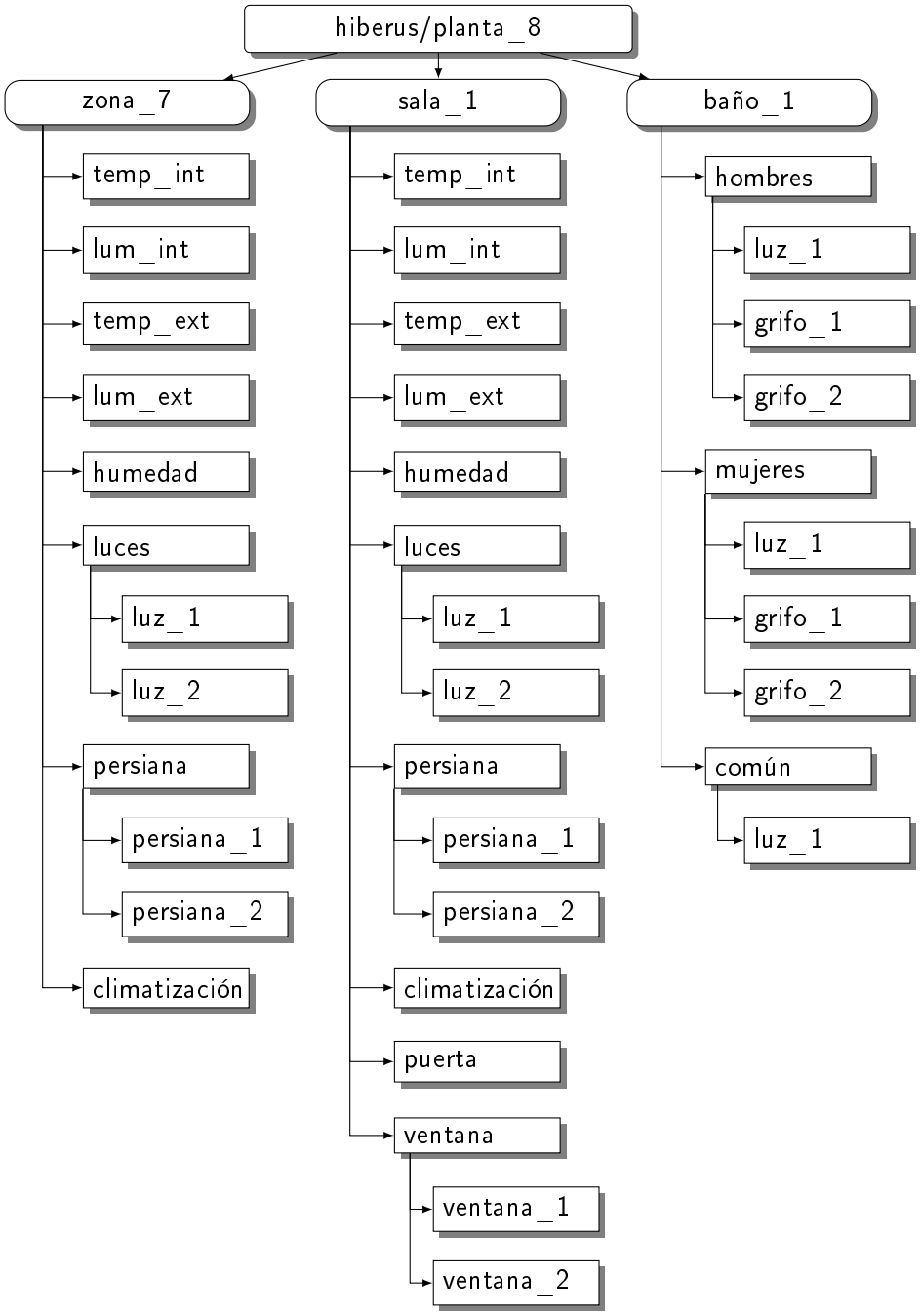


Figura 3.7: Estructura MQTT

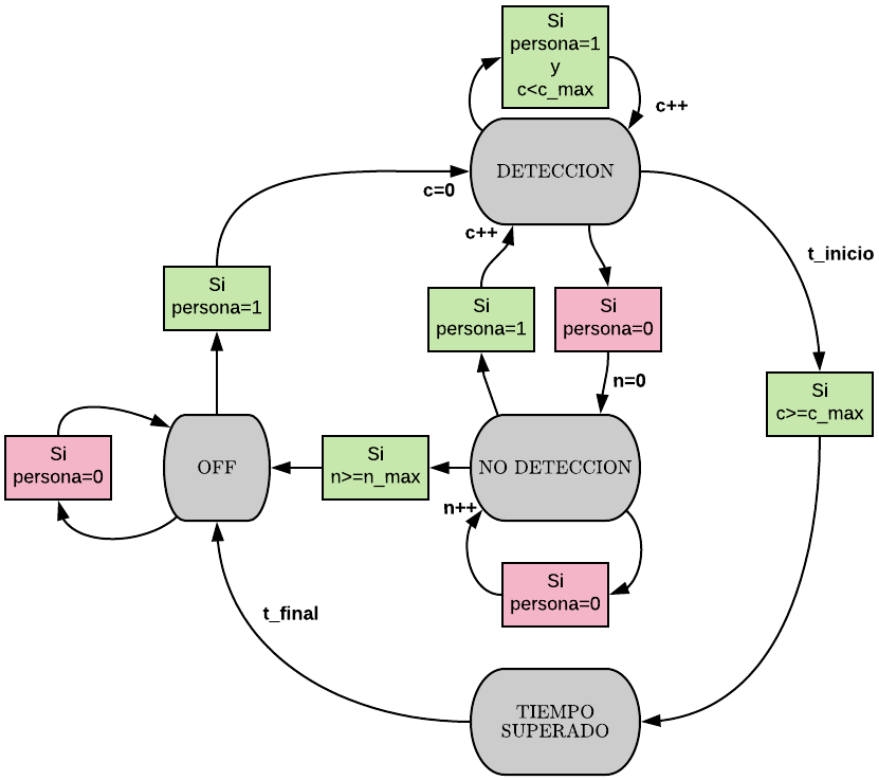


Figura 3.8: Diagrama de estados control de apagado de grifos en aseos

control que certifique que realmente la persona ya no se encuentra frente al grifo.

Para una segunda fase se añade un nuevo estado (*t_ superado*) que nos permite apagar el grifo no solo si la persona ya no se encuentra haciendo uso de él sino también cuando ha transcurrido un tiempo superior al definido. Del mismo modo, en esta segunda fase también recabaremos los periodos de tiempo con el fin de cerciorarnos de que nuestra lógica de control está funcionando correctamente o incluso para realizar ajustes. En cuanto a los eventos de encender o apagar el grifo, se envíarán mensajes MQTT al correspondiente “topic", quedando pendiente en una futura implementación que sea el propio Arduino quien active o desactive el grifo.

En la Figura 3.9 se muestra el esquema de conexiones de este bloque. En principio se plateó el uso de un ESP-01 tanto para las labores de conectividad como de obtención de los datos de los sensores. Sin embargo, tras decidir que un mismo bloque controle la actividad de dos grifos el número de pines necesario asciende a cuatro. Esta opción no es por tanto posible, ya que el número de GPIOs que tiene el módulo ESP-01 es solo dos. Por ello utilizaremos para la conectividad Wi-Fi el módulo ESP-01 y una placa Arduino para obtener los datos de los sensores. En cuanto a dicha placa durante la fase de pruebas se utiliza un Arduino Leonardo debido a su mayor tamaño y facilidad para las conexiones y será en la fase de prototipado cuando hagamos uso de Arduino Micro. Ambos nos permitirán, por un lado, la alimentación con 3.3 V del módulo Wi-Fi y con 5 V al sensor de ultrasonidos y, por otro, realizar las tareas de debug durante la fase de pruebas.

El Arduino es el encargado de obtener por tanto la información de ambos sensores y proce-

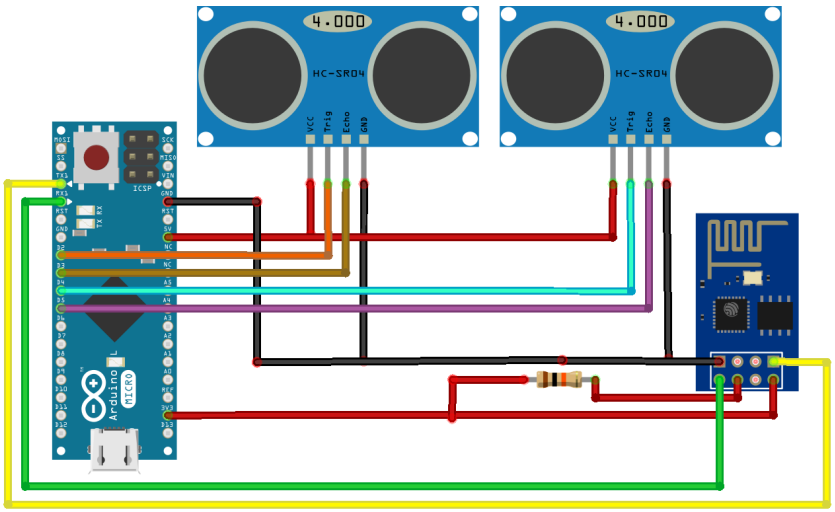


Figura 3.9: Esquema de conexión del bloque de ahorro en agua de los aseos

sar sus correspondientes máquinas de estados. Los eventos generados por ella (mensajes de "on" y "off" del grifo o los tiempos calculados) se envían mediante el puerto serie al módulo, que los transmite a través de internet haciendo uso de los correspondientes protocolos de comunicación.

Control del encendido y apagado de la iluminación en zonas de aseos

Referido al ahorro en el consumo de electricidad en las zonas de aseos se diseña un bloque con el fin de controlar el encendido y apagado de la iluminación. Para ello hacemos uso del sensor de infrarrojos PIR Motion Sensor con el que podemos detectar si existe movimiento en la zona que nos interese mediante la señal digital que recibimos en nuestra placa (3.3 V si detecta movimiento y 0 V si no lo detecta).

Al igual que ocurría en el bloque anterior es necesario elaborar una máquina de estados que certifique tanto que es necesario encender la iluminación como que podemos apagarla una vez no haya ningún usuario. La máquina de estados que hemos elaborado consta unicamente de cuatro estados (*"off"*, *"movimiento"*, *"on"* y *"esperar"*) con los que controlamos el encendido de la iluminación cuando detectamos por primera vez movimiento, el mantenimiento de ella mientras sigamos detectando movimiento y el apagado tras un tiempo de no detección. A modo de estudio estadístico y de cálculo del ahorro se incorpora al bloque la funcionalidad de registrar el tiempo de permanencia vía IFTTT.

Cálculo del tiempo que permanece encendida la iluminación de un aseo

Siguiendo con el ahorro en el consumo de la electricidad en las zonas de aseo, se realiza el diseño de un sub-bloque añadido al bloque anterior que nos permita calcular de una forma objetiva este ahorro. Para ello, haremos uso de un fotorresistor que, aunque tiene una baja precisión para calcular el nivel de iluminación, nos permite saber si la luz está encendida o no. El fotorresistor GL55, como su nombre indica, funciona como una resistencia variable en función de la luz que recibe. Para obtener este nivel de luz debemos calcular el valor de resistencia que toma y, a partir de su documentación, obtener el valor de luminosidad al que equivale. Para ello mediremos el voltaje en un divisor resistivo formado por el fotorresistor y una resistencia de 10 KΩ y mediante la ecuación de la Figura 3.11, que describe la conversión, obtenemos el

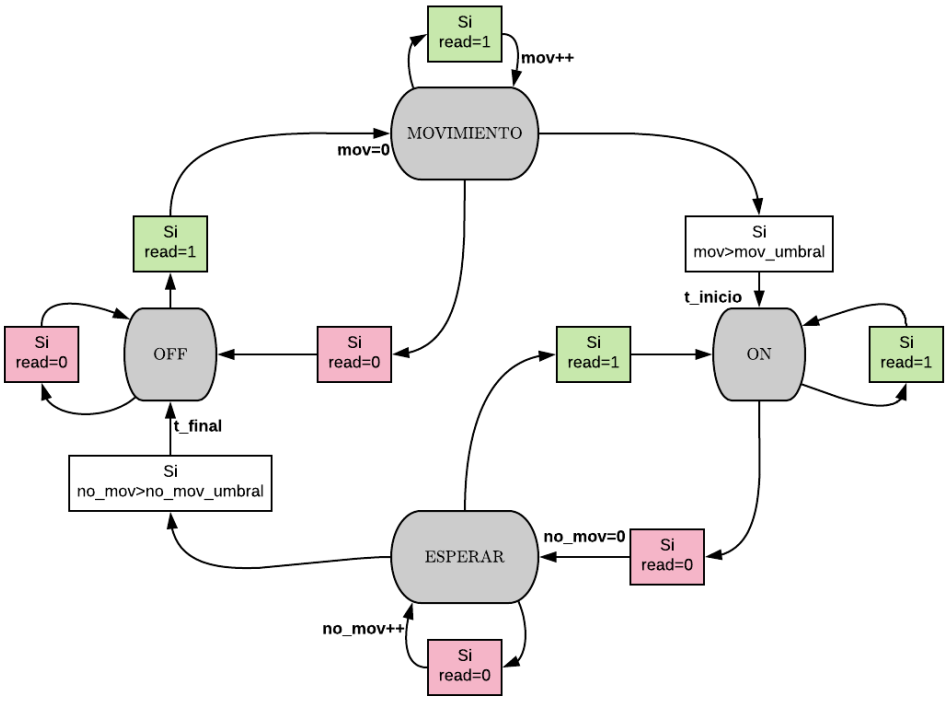


Figura 3.10: Diagrama de estados control de encendido y apagado de la iluminación en zonas de aseos

valor de luminosidad medido en lux.

$$Luminosidad = \frac{V \cdot R_{osc} \cdot 10}{R_{luz} \cdot R_c \cdot (1024 - V)}$$

- V: voltaje obtenido en el rango 0 - 1024
- R_{osc}: resistencia en oscuridad = 1.25 MΩ
- R_{luz}: resistencia en plena luz = 14 KΩ
- R_c: resistencia de calibración = 10 KΩ

Figura 3.11: Ecuación de obtención del valor de luminosidad del fotorresistor GL55

Con esta información y haciendo uso de un reloj en tiempo real obtendremos la duración de los periodos que la luz se mantiene encendida. Esta información la almacenamos también en un Google Sheet mediante IFTTT. Pasado un tiempo suficiente de toma de datos analizamos la información almacenada, tanto por este bloque como por el anterior, para obtener el tiempo que nuestro sistema de iluminación ha estado encendido y el tiempo que hubiera estado encendido en caso de instalar nuestro bloque de control de la iluminación, siendo trivial el cálculo del porcentaje de ahorro.

Por la simplicidad de ambos bloques se decide instalarlos en un mismo bloque prototipo, que podemos ver en la Figura 3.12. En este caso, debido a que necesitamos un número de pines superior a los que nos ofrece ESP-01 y a que los voltajes de alimentación y funcionamiento de los periféricos que vamos a conectar se ajustan a la especificación, se hace uso de una placa Arduino MKR1000. Además, como ventaja adicional de esta elección, podemos hacer uso del RTC interno que la propia placa incorpora y que nos permitirá calcular los periodos de tiempo de un modo más sencillo y preciso.

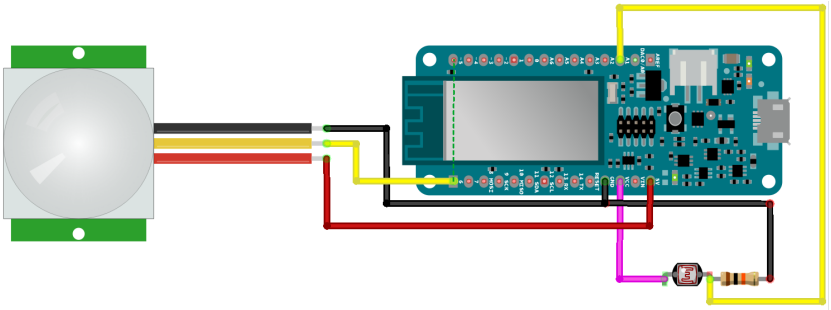


Figura 3.12: Esquema de conexión del bloque de ahorro en iluminación de los aseos

Sensorización de temperatura, humedad y luminosidad interior

En cualquier espacio al que se quiere aplicar una política de eficiencia energética es necesario monitorizar la temperatura y la luminosidad. Como valor añadido monitorizamos también la humedad para su visualización en el correspondiente nodo. Debido al ahorro económico, espacial y de ancho de banda se decide incluir las tres monitorizaciones en un solo bloque, formado por una placa Arduino, un sensor de temperatura y humedad HDC1080 y un sensor de luminosidad BH1750. En cuanto a la placa Arduino, la escogida es la MKR1000 debido a su reducido volumen, su procesador y su conectividad WiFi, que nos evita incluir un módulo extra. Aprovechando esta comunicación almacenaremos la información obtenida en la base de datos de Firebase y también, con una frecuencia mucho mayor, la enviaremos vía MQTT al nodo de procesado.

Adicionalmente, debido a la necesidad de recibir datos vía Bluetooth que veremos a continuación, se incorpora un módulo Bluetooth HC-05 que actúa como *slave*. El esquema de conexión de este bloque se muestra en la Figura 3.13. En él podemos ver que tanto el sensor HDC1080 como el sensor BH1750 comparten pines de transmisión y recepción de la información SDA y SCL aunque no por ello generarán conflictos, puesto que las direcciones de ambos son distintas.

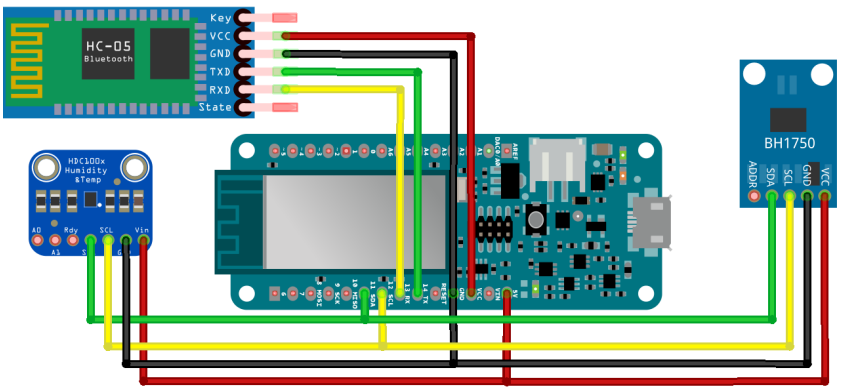


Figura 3.13: Esquema de conexión del bloque de monitorización de luminosidad, temperatura y humedad

Detección de apertura o cierre de puertas y ventanas

Otra variable que debe ser analizada para optimizar el uso de la energía de climatización es el control de si una puerta o ventana está abierta o no, ya que en función de las condiciones climatológicas puede producir un efecto positivo o negativo en nuestro sistema. Imaginemos por ejemplo dos casos opuestos: por un lado, una situación en la cual la temperatura exterior al entorno a estudiar es superior a la interior y nuestro interés es aumentar la temperatura en un espacio, siendo entonces favorable que las ventanas de dicho espacio se encuentren abiertas para así evitar el uso de la calefacción; por otro lado imaginemos la misma situación pero en este caso siendo nuestro interés disminuir la temperatura, resultado ilógico que mientras se utiliza la climatización las ventanas se encuentren abiertas introduciendo así más calor al espacio.

Por todo ello, se diseña un bloque que controle el estado de puertas y ventanas con el único fin de servir de alerta a nuestro sistema. Para conseguir dicho objetivo hacemos uso de un sensor magnético, cuyo funcionamiento se detalla en el Apéndice B, junto con una placa Arduino Micro escogida fundamentalmente por su pequeño tamaño. Para comunicar este bloque tanto con el nodo de procesado como con el de visualización se incluye en él un módulo Bluetooth, que actuará como “master", enviando los cambios en el estado de la puerta o ventana al bloque descrito en el apartado 3.4. Este bloque será el encargado de retransmitir la información vía MQTT al nodo de procesado. El esquema de conexiones de este bloque se muestra en la Figura 3.14.

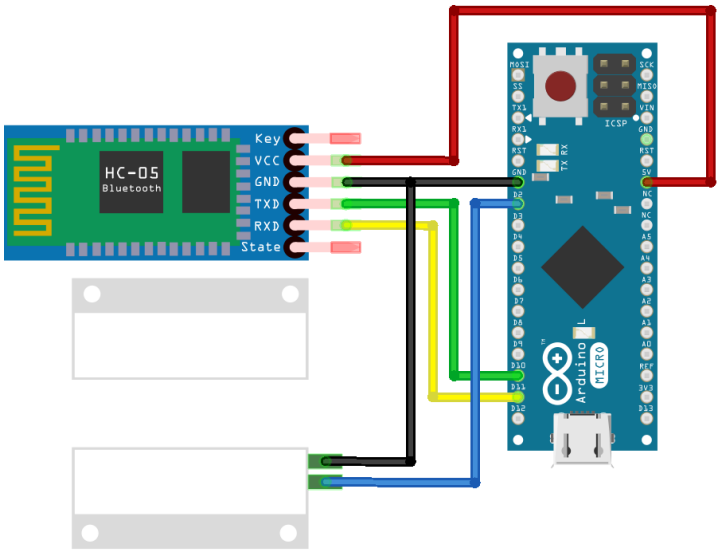


Figura 3.14: Esquema de conexión del bloque de control de apertura y cierre de puertas o ventanas

Sensorización de temperatura y luminosidad exterior

Siguiendo las justificaciones del bloque de sensorización de las condiciones interiores se diseña el equivalente para las exteriores haciendo uso de los sensores DS18B20 y TSL2561. La información se envía del mismo modo que en la sensorización interior. En la Figura 3.15 podemos ver el diseño de este bloque. En este diseño el sensor DS18B20 se encuentra sin encapsular, aunque en la implementación real se utiliza la versión impermeable.

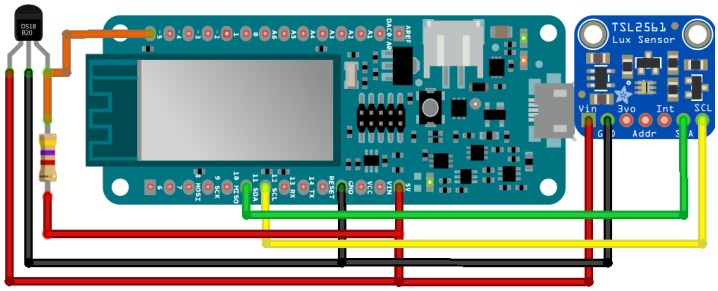


Figura 3.15: Esquema de conexión del bloque de monitorización de luminosidad y temperatura exterior

Capítulo 4

Aplicación y resultados

El entorno sobre el cual se colocan los bloques de sensorización se sitúa en el nuevo edificio Hiberus Ecosystem situado en la ciudad de Zaragoza. Este edificio consta de un total de nueve plantas, de las que la empresa posee una parte de ellas. En concreto, el área en el que se desarrolla el proyecto se sitúa en la planta ocho, por lo que tanto las zonas piloto como las zonas a sensorizar se encuentran en ella. La planta posee una superficie de aproximadamente 900 m^2 distribuidos entre una zona de trabajo, que ocupa la mayoría del espacio, pequeñas salas que se emplean para la realización de reuniones o sesiones de grupo, despachos y aseos. La Figura 4.1 ilustra la distribución de la octava planta, junto con la disposición de los distintos sensores, que detallaremos más adelante.

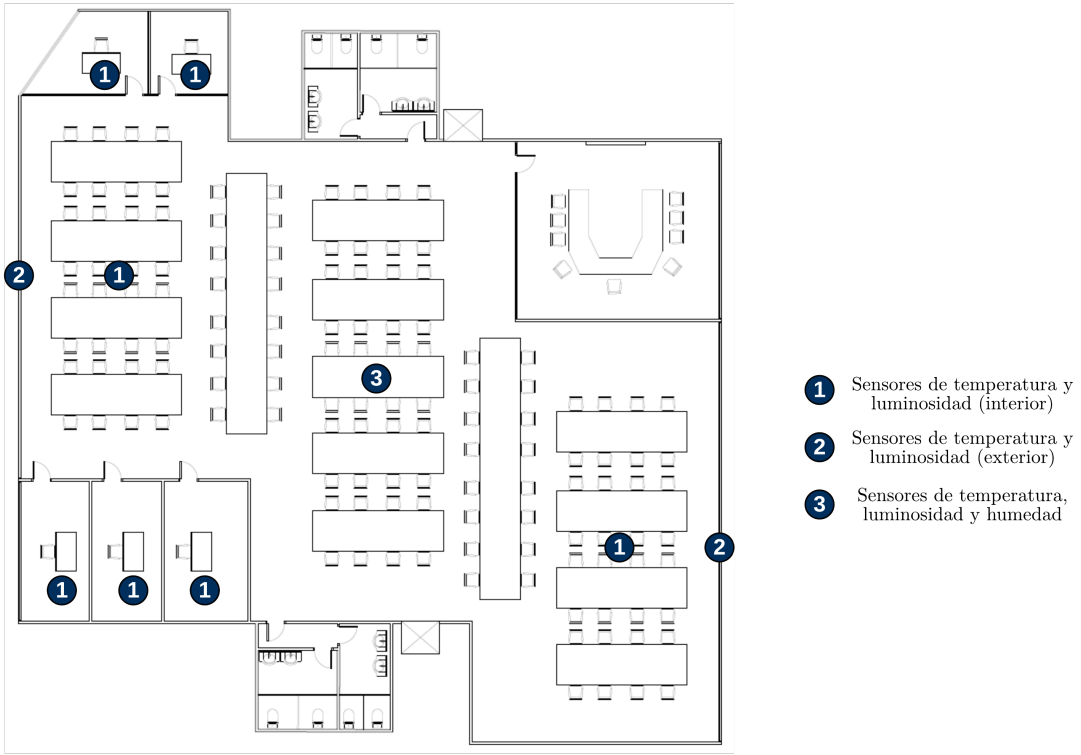


Figura 4.1: Distribución del espacio de la planta 8

Los laterales de la planta, que dan al exterior, poseen un sistema de cristaleras cubiertas por finas persianas, lo que favorece la entrada de la iluminación exterior, pero por contra tam-

bién la filtración de calor por radiación. Las cristaleras están orientadas una hacia el norte y la opuesta hacia el sur, información que será relevante posteriormente cuando queramos calcular el gasto energético por las condiciones del exterior. La iluminación consta de luminarias con distribución uniforme, cada una de ellas compuestas por dos tubos fluorescente de 36 W de 1.2 metros de largo especiales para tiendas u oficinas. En cuanto a cada espacio de aseos, como resulta obvio, están compuestos por dos aseos (hombres y mujeres).

Como introducíamos en el Capítulo 3 nuestro objetivo es monitorizar una de las zonas de un modo completo para así poder extrapolar el diseño que realicemos a otras zonas en las que no fuera necesaria una sensorización tan completa. De entre todas las zonas, escogemos para realizar el diseño la sala de mayor tamaño: una zona de aproximadamente 35 m^2 que únicamente tiene una mesa y un proyector, cuyo uso principal es la realización de reuniones o sesiones de grupos, y en nuestro caso constituye la “sala piloto".

4.1. Implementación de los bloques en el entorno

En el Capítulo 3 se realizó el diseño de los distintos bloques funcionales a implementar en nuestro sistema de acuerdo con las necesidades, pero no se realizó la distribución de ellos en el entorno a estudiar. En la Figura 4.1 podemos ver la hipotética distribución de sensores que se realizaría en una futura implementación. En cada una de las zonas en las que se distribuiría nuestra planta se instalarían sensores para obtener la temperatura y luminosidad interior salvo en la zona central, en la cual incluiríamos también la sensorización de la humedad. Esta información será empleada en el nodo de visualización pero no tendrá utilidad en el nodo de procesado. En cuanto a las zonas calificadas como exteriores se instalan adicionalmente sensores de temperatura y luminosidad exterior.

Las zonas de la planta en las que sí hemos desplegado nuestra red de sensores se muestran en las Figuras 4.2 y 4.3, correspondiendo con la sala piloto y las zonas de aseos respectivamente.

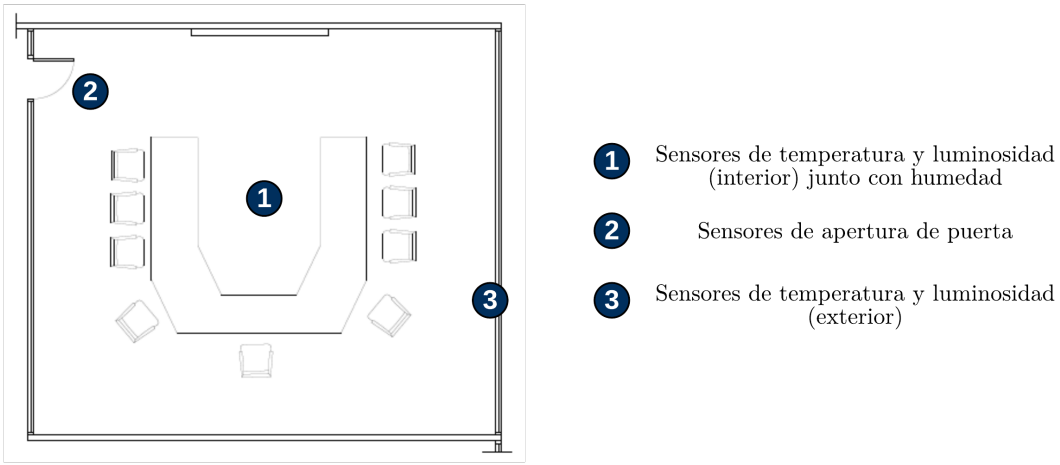


Figura 4.2: Distribución de la zona piloto y de los sensores que hay en ella

La distribución de nuestra sala piloto es equivalente a la de una de las zonas en las que se dividirá nuestra planta, siendo necesario obtener los mismos parámetros que comentamos anteriormente. La única diferencia es que en este caso añadimos un nuevo parámetro corres-

pondiente al estado de la puerta y cuya finalidad ya se detalló anteriormente.

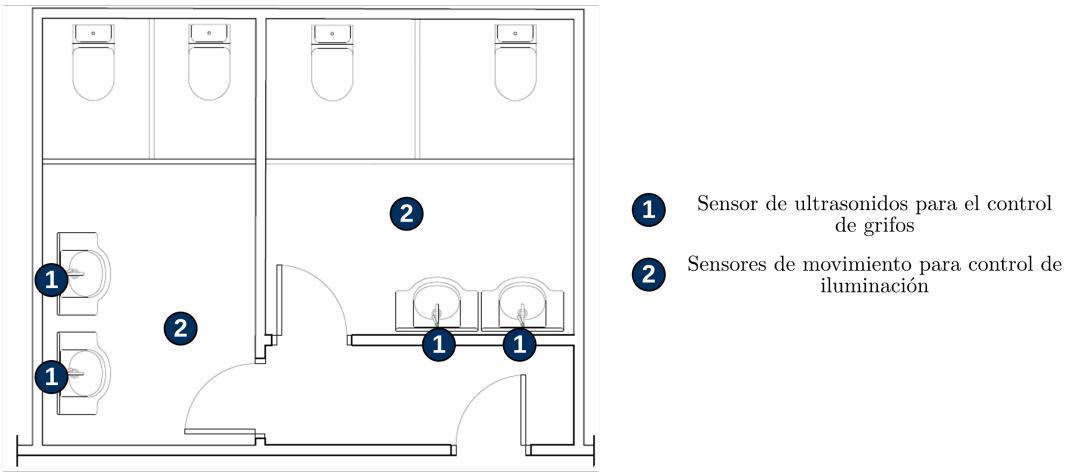


Figura 4.3: Distribución de la zona de aseos y de los sensores que hay en ella

Relativo a la zona de aseos (Figura 4.3) se instalan sensores de ultrasonidos y de movimiento tanto en los aseos masculinos como femeninos para el control del encendido o apagado tanto de los grifos como de la iluminación.

4.2. Diseño del software para cada bloque

En esta sección describiremos el código software que cada uno de los bloques que hemos distribuido en nuestro entorno debe ejecutar.

Control del apagado de grifos en aseos

De acuerdo al diseño de conexionado del bloque que ilustrábamos en la Figura 3.9, consta de una placa Arduino que se comunica mediante el puerto serie con un módulo Wi-Fi ESP-01, siendo necesario el desarrollo de software diferenciado para cada uno de ellos.

En cuanto al diagrama de software de la placa de Arduino de la Figura 4.4, tras configurar tanto la comunicación por puerto serie como los sensores de ultrasonidos, ejecutamos la máquina de estados ilustrada en la Figura 3.8 para cada uno de los sensores de ultrasonidos de forma secuencial, seguido de un tiempo de espera de 200 ms. Si la salida de alguna de ellas coincidiera con los estados de “DETECCIÓN" o de “OFF" ejecutamos las funciones *encender_grifo()* o *apagar_grifo()* respectivamente, que envían al módulo la correspondiente información.

En el otro lado de la comunicación se encuentra el módulo ESP-01, cuyo diagrama de software se muestra en la Figura 4.5, encargado de la comunicación inalámbrica. Previo a la recepción de la información es necesario configurar la conexión Wi-Fi junto con las plataformas software MQTT y IFTTT. Una vez tenemos la capacidad de retransmitir la información esperamos a recibirla a través del puerto serie y, en función de lo recibido, enviamos los mensajes MQTT o generamos el evento IFTTT.

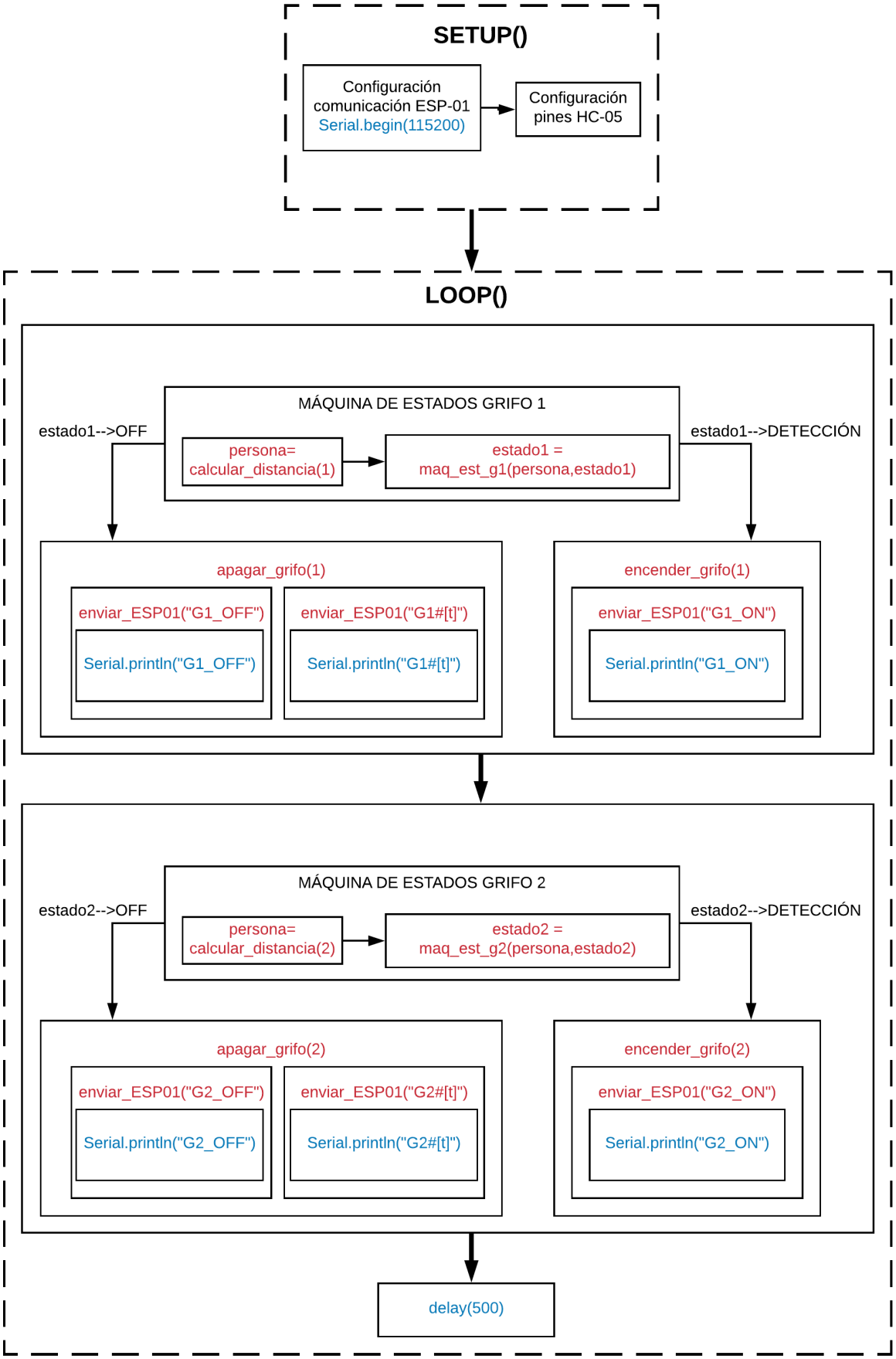


Figura 4.4: Esquema de software del bloque de control de apagado de los grifos en aseos: Arduino

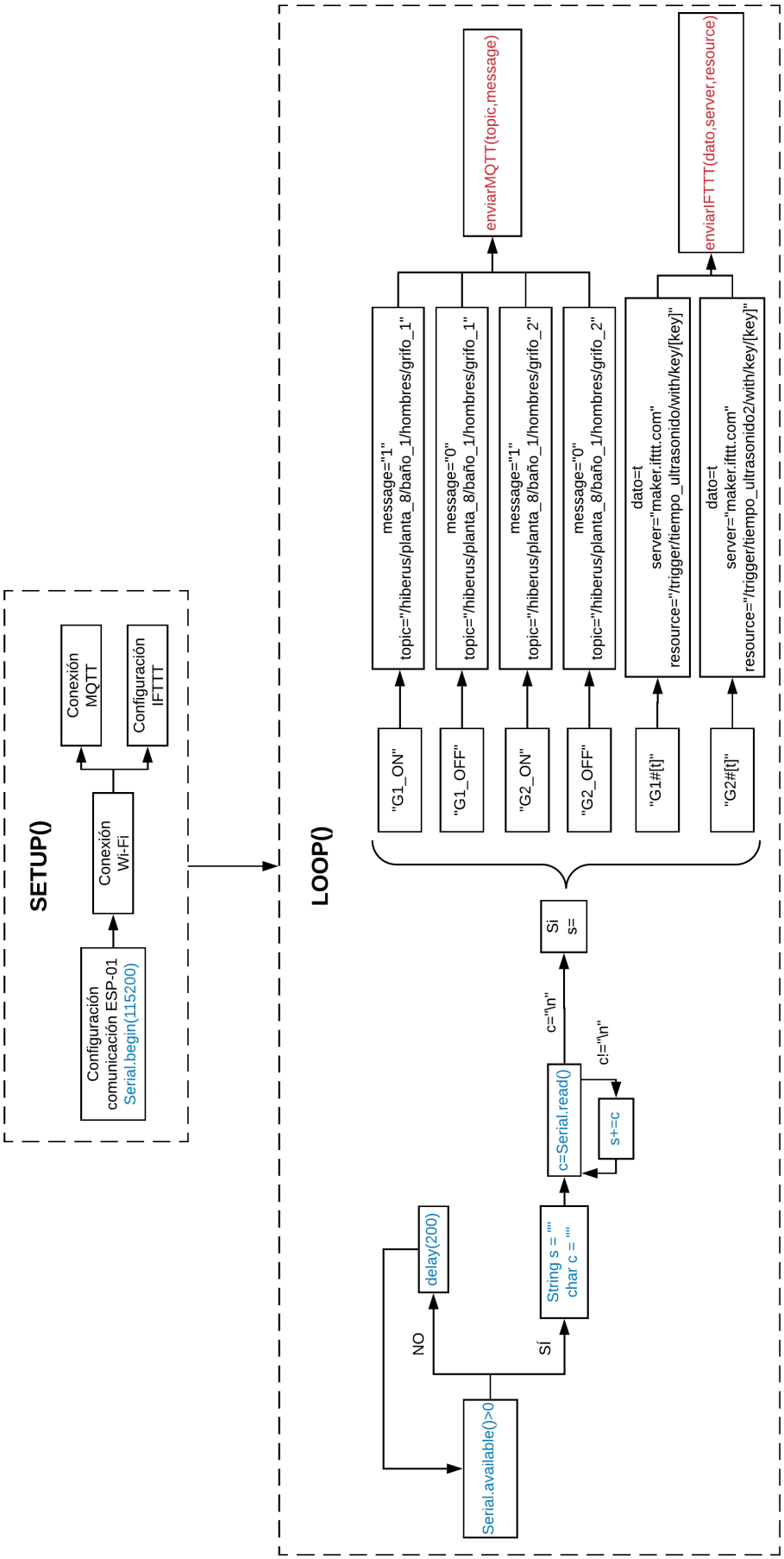


Figura 4.5: Esquema de software bloque de control de apagado de los grifos en aseos: ESP-01

Control de la iluminación en aseos

Este bloque, que se corresponde con la Figura 3.12, consta unicamente de una placa Arduino MKR1000, en la cual se integra tanto la lógica de control como la comunicación inalámbrica. Al igual que ocurría en el diagrama anterior, es necesario establecer la conexión Wi-Fi y de las plataformas software MQTT y IFTTT. La lógica de control consta de las máquinas de estados del sensor PIR (Figura 3.10) y la máquina de estados del sensor de luminosidad. Esta última, que no se encuentra ilustrada por su sencillez, consiste en alternar entre los estados de encendido y apagado dependiendo de si el valor de luminosidad es superior o inferior a un determinado umbral. Ambas se ejecutan de forma secuencial y, en función de sus salidas, se envían los correspondientes mensajes a las plataformas software.

Sensorización de temperatura, humedad y luminosidad interior

Otro de los bloques a implementar en nuestro sistema es el de la sensorización de las condiciones climáticas y luminosas de cada una de las zonas a estudiar haciendo uso del conexionado de la Figura 3.13. Previo a la obtención de los datos, y al igual que en los casos anteriores, es necesario configurar las conexiones Wi-Fi junto con las plataformas software MQTT y Firebase así como configurar los dos sensores a utilizar: el sensor de temperatura y humedad HDC1080 y el sensor de luminosidad BH1750. Adicionalmente es necesaria también la configuración de la comunicación Bluetooth, a través de la cual recibiremos la información del estados de las puertas.

Esta información se recibe por el puerto serie para lo cual es necesario leerlo con una frecuencia alta y, en caso de recibir algún mensaje, analizarlo. En caso de que el mensaje recibido contenga la cabecera “puerta" obtendremos el estado de la puerta que lo acompaña y se enviará vía MQTT al correspondiente topic. A continuación y con una frecuencia menor, marcada por *t_sens*, tomaremos los datos de temperatura y humedad con la función *readSensor()* del HDC1080 y el valor de luminosidad del BH1750 mediante *luxometro.readLightLevel()*. Estos tres valores se envían por MQTT a cada uno de sus topics y, con una frecuencia aún menor controlada por *t_firebase*, lo publicamos en la base de datos de Firebase.

Detección de apertura o cierre puertas y ventanas

Finalmente, en el bloque anterior mencionábamos que existía una comunicación Bluetooth a través de la cual obtendríamos el estado de la puerta. El bloque que vamos a describir es el encargado de transmitir dicha información y se corresponde con el conexionado de la Figura 3.14. Para ello y en primer lugar, debemos configurar la comunicación a través del puerto serie y el sensor magnético. Una vez la comunicación esté activa tomaremos la medida del sensor aproximadamente cada tres segundos y, dependiendo del estado en el que se encontraba la puerta con anterioridad, transmitiremos vía Bluetooth los mensajes correspondientes para indicar que la puerta se ha abierto o cerrado.

Sensorización de temperatura y luminosidad exterior

Este bloque de sensorización es paralelo al de sensorización de las condiciones interiores. De igual modo debemos configurar las conexiones tanto a la red Wi-Fi como a las plataformas software MQTT y Firebase. Una vez la comunicación esté preparada tomaremos los valores de temperatura y luminosidad de ambos sensores con las correspondientes funciones y los enviaremos a la lógica de control y a la base de datos con la correspondiente frecuencia.

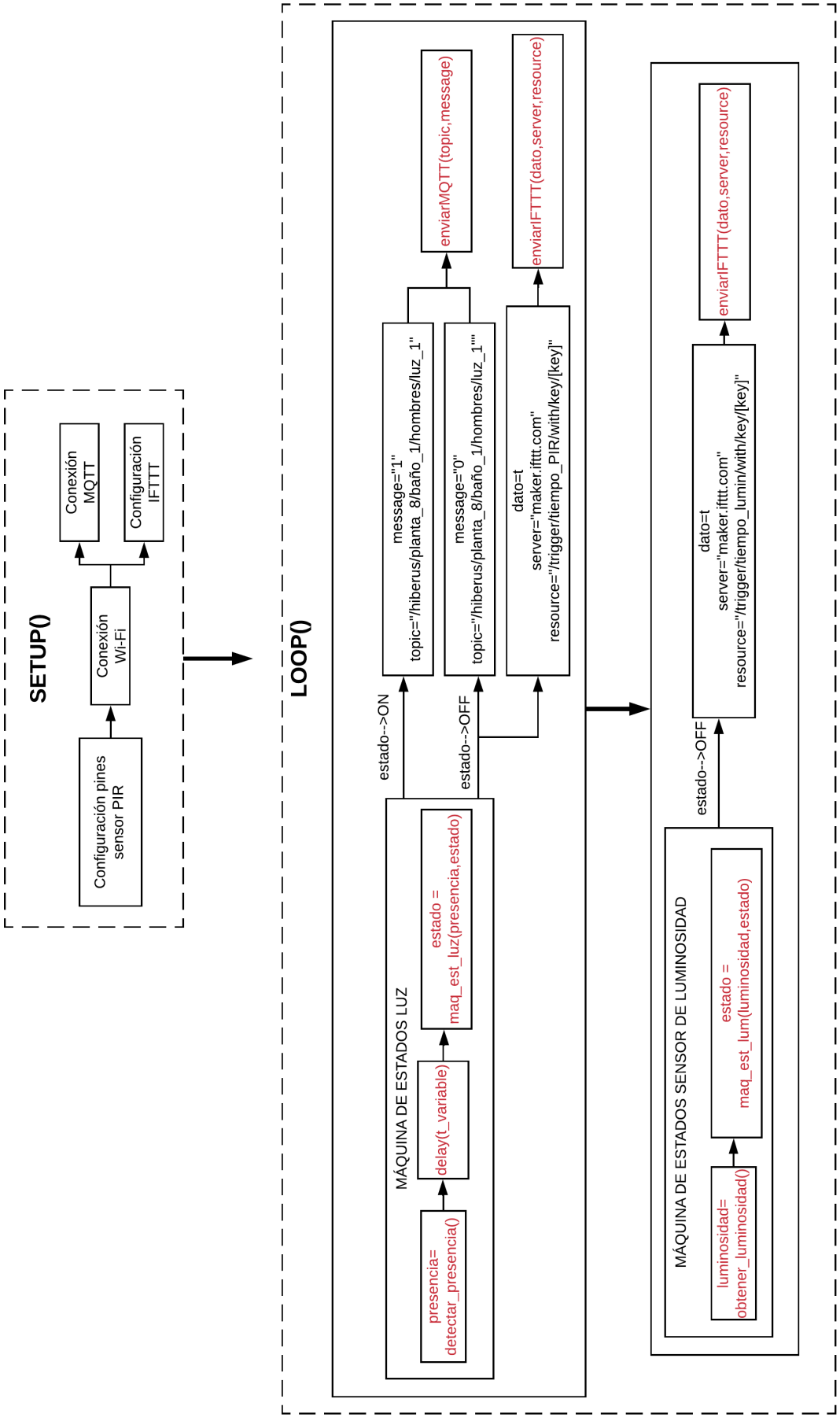


Figura 4.6: Esquema de software del bloque de control de la iluminación en aseos

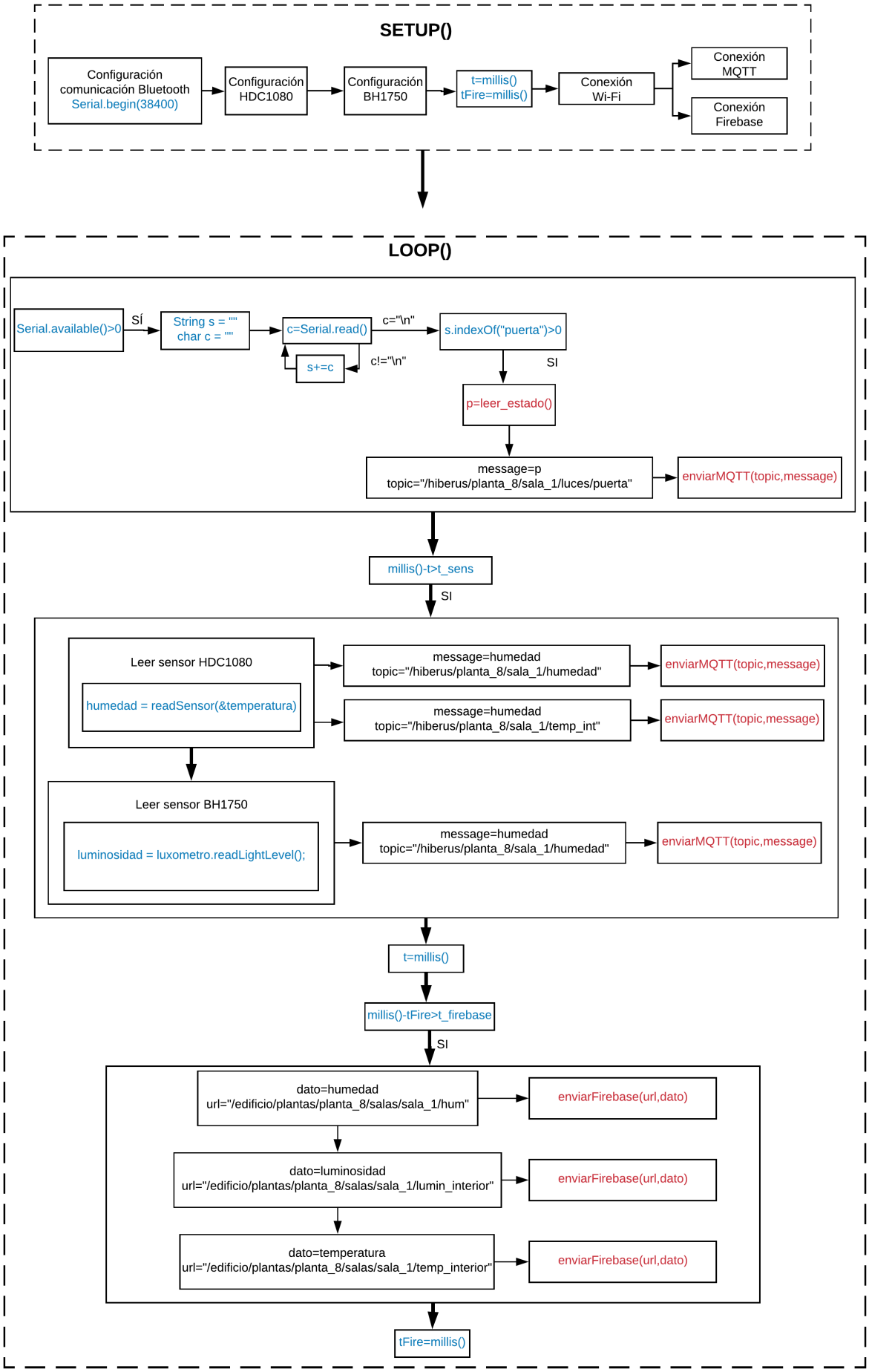


Figura 4.7: Esquema de software del bloque de sensores de temperatura, humedad y luminosidad

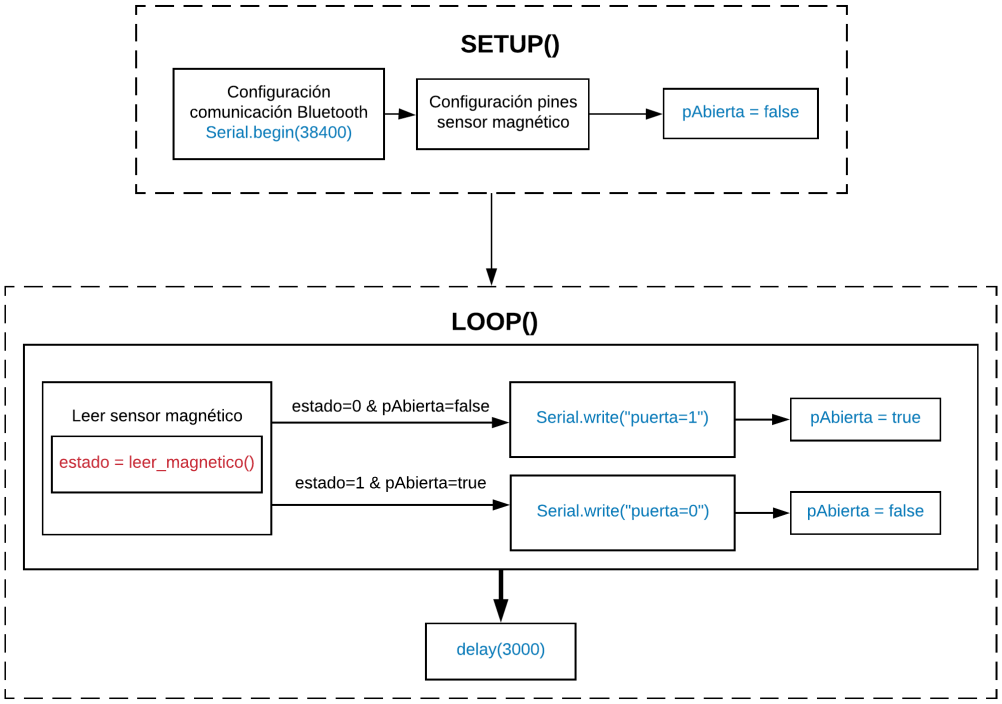


Figura 4.8: Esquema de software del bloque de control del estado de la puerta

4.3. Definición del nodo de procesado

Todo lo relativo a la lógica de control se sitúa en el nodo de procesado, encargado del proceso de recolección de los datos, procesado y toma de las decisiones oportunas como el ajuste de la climatización, la iluminación y la posición de las persianas.

Estas decisiones se toman fundamentalmente en función de la aportación de energía necesaria en la climatización y la iluminación[35]. La potencia necesaria para enfriar o calentar un espacio (Ecuación 4.1) consta de diversas contribuciones. En el proceso de enfriar o calentar el aire existen dos tipos de calor: el necesario para aumentar o disminuir la temperatura del aire (calor sensible) y el necesario para evaporar o condensar el agua presente en el aire (calor latente). De las distintas contribuciones de la potencia necesaria solo tenemos en cuenta el calor latente en el caso de la ventilación y la ocupación puesto que son los dos únicos casos en que se mezclan aires con diferentes humedades relativas. El gasto térmico mínimo por las contribuciones de calor es:

$$W_{min} = W_{trans} + W_{rad} + W_{vent} + W_{ocup} + W_{ilum} + W_{equipos} \tag{4.1}$$

A continuación se detalla como se obtiene cada una de ellas:

• Transmisión: calor ganado o perdido por la transmisión de calor a través de distintas superficies: paredes, techo, suelo o ventanas. La aportación de cada una de ellas depende de su superficie, de su coeficiente de transmitancia térmica ($k_{T,i}$) y de la diferencia de temperatura entre los espacios que separa.

$$W_{trans} = \sum S_i \cdot k_{T,i} \cdot \Delta T_{ext-\mathrm{int},i} \tag{4.2}$$

• Radiación: calor aportado por la radiación solar a través de las ventanas. El calor por radiación depende de múltiples factores: la calidad de los cristales, el coeficiente de radiación

($k_{R,i}$) proporcional al nivel de luminosidad y de si la superficie de la ventana está cubierta por un material aislante. Este último resulta de vital importancia en nuestro estudio, puesto que se relaciona con la posición de las persianas que cubren las ventanas.

$$\begin{aligned} W_{rad} &= \sum (S_{cristal} \cdot k_{R,i} \cdot k_{cristal} + S_{persiana} \cdot k_{R,persiana} \cdot k_{cristal} \cdot k_{persiana}) \\ &= \sum (S_{ventana} \cdot \%_{ventana} \cdot k_{R,i} \cdot k_{cristal} + S_{ventana} \cdot (1 - \%_{ventana}) \cdot k_{R,i} \cdot k_{cristal} \cdot k_{persiana}) \end{aligned} \tag{4.3}$$

• Ventilación: aportación de calor a nuestro entorno a través de la renovación de aire del local. Según las normativas esta renovación debe ser de alrededor de 35 $m^3/h{\cdot}persona$. Su cálculo depende únicamente del número de personas, puesto que el volumen de aire y los coeficientes sv y sl, que corresponden a las contribuciones de calor sensible y latente respectivamente, son constantes.

$$\begin{aligned} W_{vent} &= W_{vent\_sensible} + W_{vent\_latente} \\ &= V_{aire} \cdot nPersonas \cdot \Delta T_{conf-\text{int}} \cdot (sv + sl) \end{aligned} \tag{4.4}$$

• Ocupación: aportación de calor que generan las propias personas. Al igual que en la ventilación tiene tanto componente sensible como latente y su valor es constante puesto que solo depende de dos coeficientes constantes ($k_{sensible}$ y $k_{latente}$) y de la superficie de la planta.

$$W_{ocup} = W_{ocup\_sensible} + W_{ocup\_latente} = S_{planta} \cdot (k_{sensible} + k_{latente}) \tag{4.5}$$

• Iluminación: calor aportado por los sistemas de iluminación. Su valor depende del número luces encendidas, del consumo de cada una de ellas y del coeficiente de tipo de iluminación (k_{tipo_ilum}).

$$W_{ilum} = nLuces \cdot W_{luz} \cdot k_{tipo\_ilum} \tag{4.6}$$

• Equipos: aportación de calor de sistemas como los equipos informáticos. Su valor es constante dado que depende del coeficiente de carga de los equipos ($k_{equipos}$) y de la superficie de la planta.

$$W_{equipos} = S_{planta} \cdot k_{equipos} \tag{4.7}$$

Por sus capacidades multiparadigma, se selecciona como lenguaje de programación Python. Mediante su entorno de desarrollo Pycharm se elabora un *script* de Python que realiza todas las tareas del nodo de procesado. Una de sus ventajas es que la integración, tanto con la base de datos como con el protocolo MQTT, no es excesivamente compleja. En el caso de la base de datos usaremos el SDK para Python [36], que nos permite realizar todas las tareas de conexión, autenticación, lectura y escritura. En cuanto a MQTT, hacemos uso de unas librerías *open-source* desarrolladas por Eclipse (Paho-MQTT[37]) para todos los lenguajes de programación, incluido Python. Gracias a su buena implementación podemos realizar todas las tareas de conexión, suscripción y publicación de un modo realmente sencillo.

Haciendo uso de todo esto y tras obtener todos los datos, tanto vía MQTT como de la lectura de la base de datos de Firebase, se calculan las aportaciones por transmisión, ventilación, ocupación y equipos, que no dependen de ninguna de las variables a ajustar. Aquellas que sí son dependientes (radiación, transmisión y calor producido por la iluminación) junto con el propio gasto energético de los sistemas de iluminación formarán el sistema de ecuaciones a optimizar. Para ello se deben definir las variables en función de las cuales se va a realizar la minimización: el número de luces y el porcentaje de ventana no cubierta por la persiana. Sin embargo no

todos los posibles valores obtenidos en la minimización son válidos para nuestro intereses. Es necesario fijar ciertas condiciones a cumplir, basadas en que la iluminación satisfaga la iluminación de confort y en que una parte de ella, que se establecerá en función de la luminosidad exterior, sea luz natural.

La función de minimización junto con las condiciones a cumplir y los límites superiores e inferiores son los siguientes:

$$\begin{array}{l} min \quad W_{transmisión} + W_{radiación} + W_{iluminacionón} \\ min \quad W_{energtico\ por\ iluminación} \end{array} \tag{4.8}$$

$$nLuces_{centro} \cdot F_{luz} \geq 0.6 \cdot S_{planta} \cdot E_v \tag{4.9}$$

$$nLuces_{norte} \cdot F_{luz} + S_{ventana} \cdot \ \%vent_{norte} \cdot E_{v,ventana} \geq 0.2 \cdot S_{planta} \cdot E_v \tag{4.10}$$

$$nLuces_{sur} \cdot F_{luz} + S_{ventana} \cdot \ \%vent_{sur} \cdot E_{v,ventana} \geq 0.2 \cdot S_{planta} \cdot E_v \tag{4.11}$$

donde F_{luz} es la medida de potencia luminosa percibida en lumen y E_v es la medida de incidencia mínima de confort que debe tener la luz sobre una superficie en lux=lm/m^2. Respecto a las constantes numéricas representan el porcentaje de la planta que corresponden a la zona central, norte y sur.

$$0 \leq nLuces \leq nLuces_{\text{máx}} \tag{4.12}$$

$$\%vent_{\text{mín}} \leq \%vent \leq 1 \tag{4.13}$$

Una vez optimizada, calculamos el gasto energético descrito en la Ecuación 4.1, que se corresponde con el gasto mínimo para compensar las distintas fuentes de calor, mientras que para calentar o enfriar el entorno deberemos hacer un aporte energético extra negativo o positivo respectivamente. Cada una de las salidas de la lógica de control (número de luces que deben estar encendidas, gasto energético necesario para la climatización y el estado de las persianas) se envía a su correspondiente actuador (los sistemas de iluminación, climatización y motores de las persianas) vía MQTT. La frecuencia con la que se aplica esta lógica de control dependerá del edificio en el que se implemente, aunque esta lógica es también reactiva ante las condiciones de confort.

4.4. Definición del nodo de visualización

Como valor añadido al proyecto se decide desarrollar un nodo de visualización. Consiste en una aplicación Android para dispositivos tablet de 9 o 10 pulgadas, para lo cual utilizamos el entorno de desarrollo integrado Android Studio. La aplicación realiza conexiones con Firebase y con el protocolo MQTT. Para realizar ambas conexiones hacemos uso de la API Firebase de Android, que nos permite una sencilla integración así como la lectura y escritura de los datos, y de las librerías *open-source* Paho-MQTT.

La aplicación se plantea como un tablero desde el cual la persona responsable de cada entorno pueda controlar las distintas variables, así como configurar las condiciones de confort, siendo esta por tanto la principal funcionalidad. Dado que, como hemos dicho con anterioridad, no se monitoriza todo el entorno del edificio la aplicación solo mostrará los datos de aquellas

salas o zonas en las que el sistema ha sido instalado. Sin embargo, el desarrollo se realiza de modo que su escalabilidad en un futuro sea sencilla.

La Figura 4.9 muestra el diagrama de flujo de la aplicación, en cuya pantalla de inicio (Figura 4.10) se muestra el eslogan actual de Hiberus “Crecemos contigo".

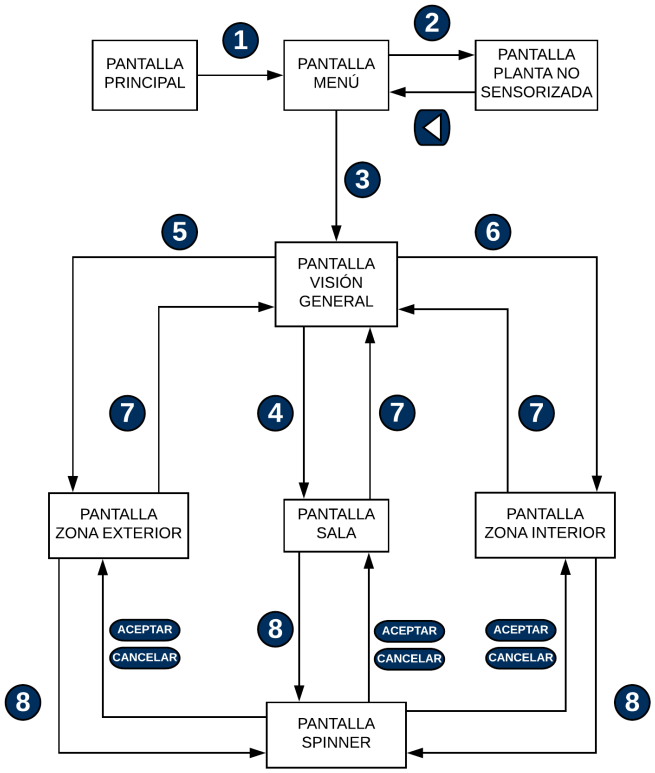


Figura 4.9: Diagrama de flujo de la aplicación Android

En esta misma pantalla, al hacer *click* en (1), se despliega un menú lateral (Figura 4.11) que lista las distintas plantas de nuestro edificio y sobre el cual podemos escoger de qué planta queremos visualizar la información.

Actualmente el único entorno que se encuentra sensorizado es la planta ocho por lo que en caso de seleccionar otra planta ((2)) encontraremos un mensaje de “La planta no se encuentra disponible en este momento". En caso de seleccionar la planta ocho ((3)) accedemos a una pantalla en la que se muestran una imagen en 2D de la distribución junto con los datos de temperatura, humedad y luminosidad media de ella (Figura 4.12).

Sobre esta distribución del entorno se incluyen unos botones que, al hacer *click* sobre ellos, nos muestran distintas pantallas con la información de cada una de las zonas. Hasta el momento solo existen tres botones sobre los que hacer *click*((4), (5) y (6)) que corresponden con nuestra sala piloto, una zona exterior o una zona interior respectivamente. La información que se muestra en cada una de las zonas definidas en la introducción de este capítulo se ilustra en la Tabla 4.1. La visualización de la información, que proviene o bien de la base de datos de Firebase o se recibe vía MQTT, es reactiva ante cambios sin necesidad de refrescar la página.

Figura 4.10: Pantalla de inicio de la aplicación Android

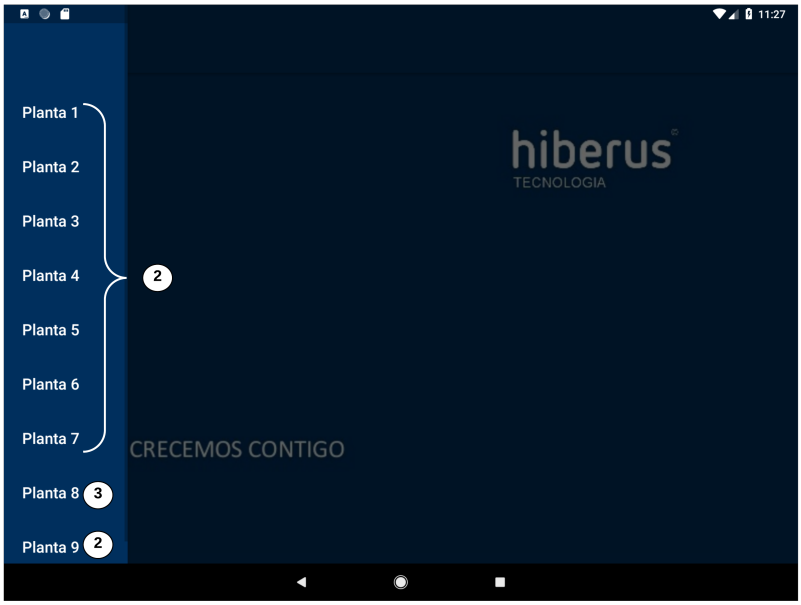


Figura 4.11: Menú de selección de la planta a visualizar en la aplicación Android

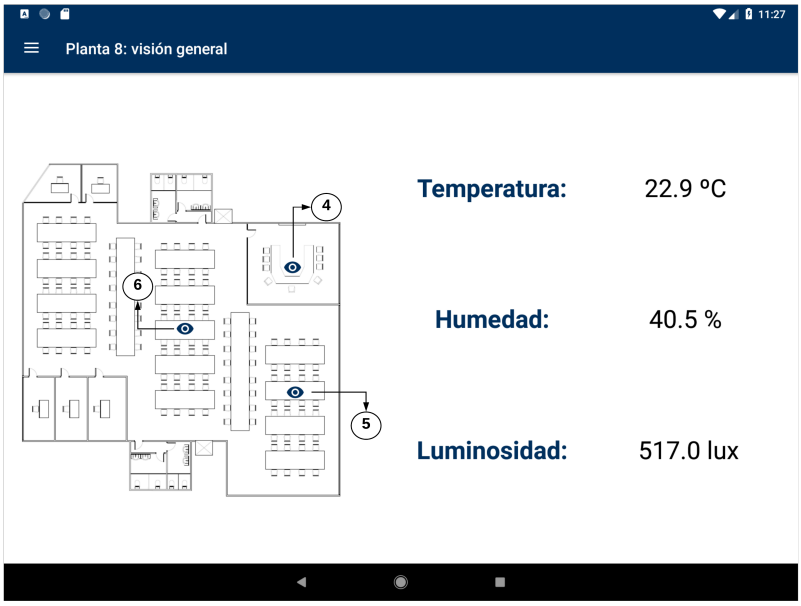


Figura 4.12: Pantalla de información general de una planta sensorizada en la aplicación Android

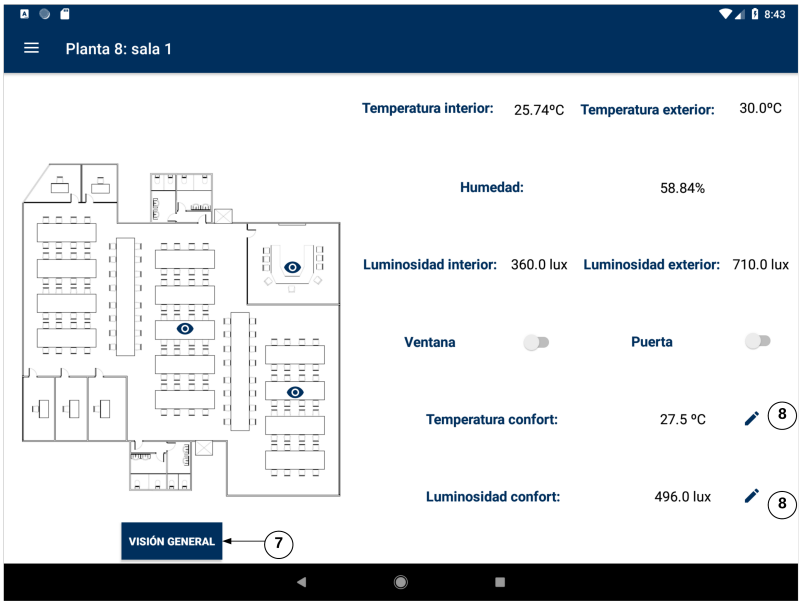


Figura 4.13: Pantalla con la información de una sala sensorizada en la aplicación Android

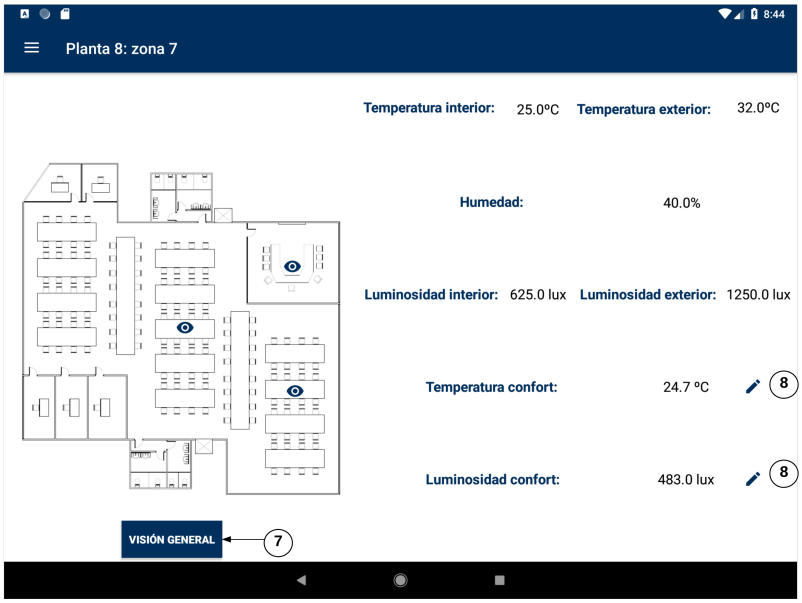


Figura 4.14: Pantalla con la información de una zona exterior sensorizada en la aplicación Android

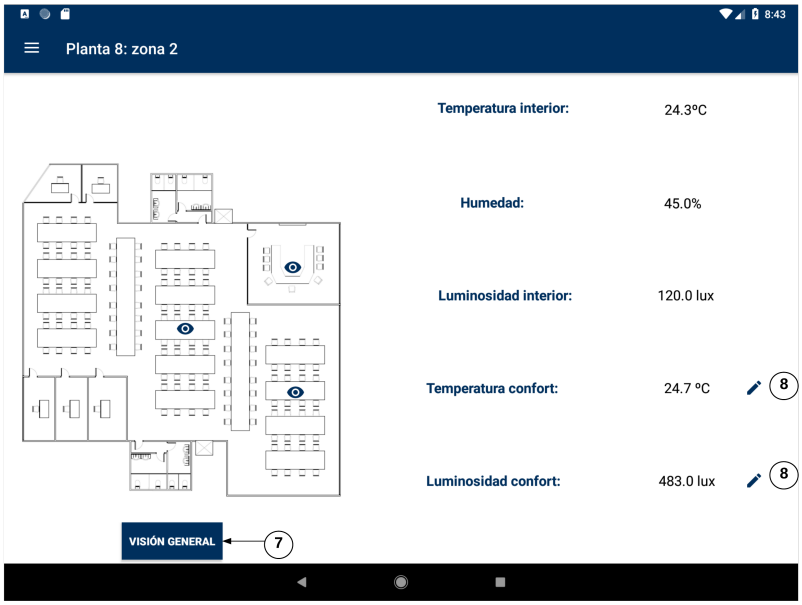


Figura 4.15: Pantalla con la información de una zona interior sensorizada en la aplicación Android

	Sala	Zona exterior	Zona interior
Temperatura interior	✓	✓	✓
Temperatura exterior	✓	✓	✗
Humedad	✓	✓	✓
Luminosidad interior	✓	✓	✓
Luminosidad exterior	✓	✓	✗
Temperatura de confort	✓	✓	✓
Luminosidad de confort	✓	✓	✓
Estado de ventanas	✓	✗	✗
Estado de puertas	✓	✗	✗

Tabla 4.1: Datos a mostrar dependiendo del tipo de zona

Otra funcionalidad con la que dotamos a la aplicación consiste en poder modificar las condiciones de confort de la sala piloto así como de la planta en general. Cualquier cambio en ellas provocará una reacción en la lógica de control que reajustará las variables para conseguir el confort. Para realizar esta modificación debemos hacer *click* en (8) en la pantalla que deseemos, lo que hará aparecer una ventana emergente (Figura 4.16).

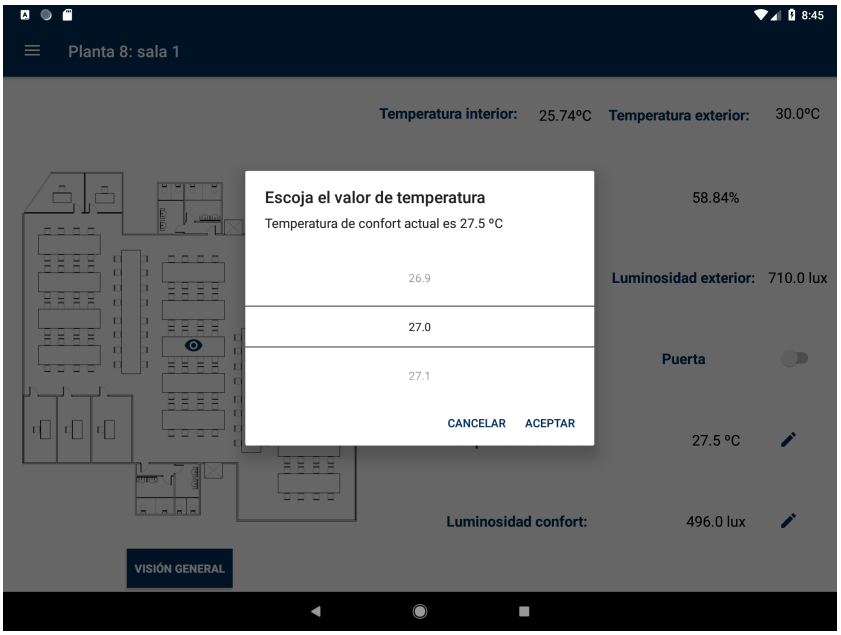


Figura 4.16: Pantalla de modificación de las condiciones de confort en la aplicación Android

4.5. Resultados

Tras el diseño, el montaje y el desarrollo software tenemos todo lo necesario para obtener algunos resultados de nuestro sistema, principalmente de las implementaciones en las zonas de aseos y en la sala piloto.

En las zonas de aseos analizamos la información recabada del sistema de encendido o apagado de los grifos y de la iluminación. Para el ahorro de agua de los grifos se recogen en torno

a 900 mediciones a lo largo de una semana, las cuales se analizan con la herramienta Microsoft Excel y visualizamos en la Figura 4.17.

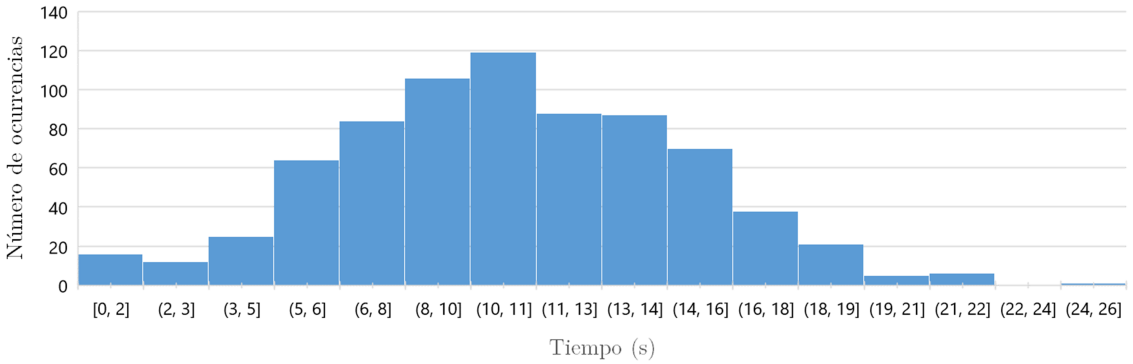


Figura 4.17: Resultados obtenidos del bloque de encendido y apagado de grifos

El análisis que se lleva a cabo consiste en obtener la media, desviación típica y la mediana, dato que se emplea para establecer el umbral temporal a partir del cual apagaremos el grifo. Estas mismas medidas se usan para estimar, en caso de haber implementado el sistema de apagado de grifos automático, el ahorro obtenido. Definimos el tiempo ahorrado como la diferencia temporal entre la medida y la mediana, siempre y cuando sea superior a cero. En caso de no ser superior no existiría ahorro puesto que en la implementación actual el usuario cerraría manualmente el grifo y en la implementación futura el sistema de apagado automático se accionaría al no detectar al usuario. Con el fin de realizar un mejor análisis tenemos también en cuenta la situación en la cual el tiempo fijado para un usuario concreto es insuficiente y decide volver a accionar el grifo, suponiendo que esto ocurre cuando la medida es superior a dos veces la mediana. El tiempo ahorrado, por tanto, se obtendrá de la resta entre la medida y dos veces la mediana. Tanto los valores estadísticos como los ahorros obtenidos se muestran en la Tabla 4.2.

Media	11.55
Desviación típica	3.73
Mediana	11.49
Porcentaje ahorro en el caso 1[1]	16.1 %
Porcentaje ahorro en el caso 2[2]	15.2 %

Tabla 4.2: Resultados bloque encendido y apagado de grifos

El ahorro de tiempo obtenido está en torno al 15.5 %, lo que se traduce en un ahorro equivalente en la factura del agua.

Referido al ahorro en la iluminación de la zona de aseos se recogen durante una jornada laboral por un lado, las medidas del tiempo que la iluminación se encuentra encendida y, por otro, los tiempos que los distintos usuarios hacen uso del baño. Con toda esta información se elabora la correspondiente estadística con el fin de obtener un valor objetivo del ahorro

[1] Caso 1: No tenemos en cuenta si dentro de la misma medida el usuario activa dos veces el grifo.
[2] Caso 2: Tenemos en cuenta si dentro de la misma medida el usuario activa dos veces el grifo.

que supondría la implementación del sistema. La Figura 4.18 representa el histograma de los tiempos de uso mientras que el tiempo total que la iluminación permanece encendida es de 11 664 segundos, un 36 % del tiempo.

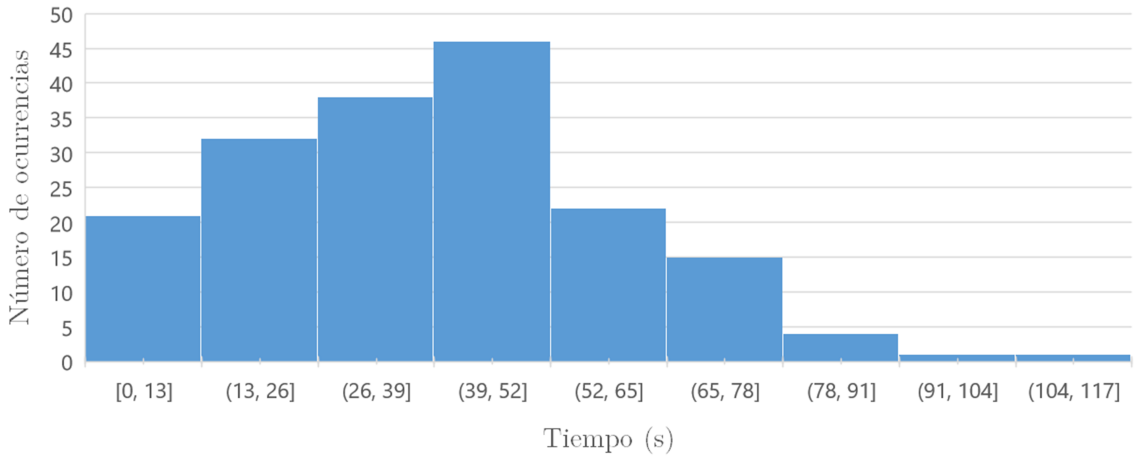


Figura 4.18: Resultados obtenidos del bloque de encendido y apagado de la iluminación en aseos

La media de tiempo que los usuarios hacen uso del baño es de 33.36 segundos con una desviación típica de 18.83 segundos, un valor un tanto elevado que refleja la gran dispersión de los datos. El total del tiempo que la iluminación estaría encendida en caso de implementación es de 6074 segundos equivalente al 18.74 %, lo que supondría un ahorro en el gasto de un 47.93 %.

Respecto a la sala piloto, no existe una manera objetiva de conocer el ahorro obtenido. Sin embargo, sí es posible ilustrar distintos resultados derivados de la toma de datos y el procesamiento de la información.

La lógica de control, como vimos en apartados previos, consta de una ecuación de gasto energético mínimo que a su vez se divide en múltiples contribuciones. Para calcular cada una de ellas es necesario disponer de distintos coeficientes empíricos de las dimensiones de la zona a estudiar y de otras constantes. En nuestro caso los valores que hemos utilizado se muestran en las Tablas 4.3 , 4.4 y 4.5 respectivamente.

Coeficientes	
$k_{T,suelo}$	1.2
$k_{T,techo}$	1.2
$k_{T,ventana}$	5.8
$k_{R,norte}$	241
$k_{R,sur}$	38
$k_{cristal}$	0.88
$k_{persiana}$	0.45
$k_{sensible}$	6
$k_{latente}$	3.79
k_{tipo_ilum}	1.25
$k_{equipos}$	4.5

Tabla 4.3: Valores constantes de la lógica de control

Dimensiones		
Ventana	Ancho (m)	9
	Alto (m)	2.3
Sala	Ancho (m)	6
	Largo (m)	9
	Alto (m)	2.3

Tabla 4.4: Dimensiones empleadas en la lógica de control

Otras constantes	
nPersonas	8
Factor sv $(W \cdot m^3/h \cdot K)$	0.35
Factor sl $(W \cdot m^3/h \cdot K)$	0.8
W_{luz} (W)	36
renAire $(m^3/h \cdot persona)$	30

Tabla 4.5: Otras constantes de la lógica de control

La optimización de la ecuación se realiza con el fin de maximizar el ahorro para alcanzar unas condiciones de confort. Estas condiciones se fijan en función de diversos estudios. La temperatura de confort según [38] debe situarse entorno a 22 °C durante el invierno y 24.5 °C en verano para un ambiente de trabajo de actividad ligera esencialmente sedentaria. En cuanto a la luminosidad, según [39], para este ambiente debe ser de 500 lux.

Para calcular el gasto necesario es indispensable conocer las condiciones en las que nos encontramos en cada momento. En este caso vamos a ilustrar una de las optimizaciones en un momento concreto de un día durante la jornada laboral. Las condiciones en las que nos encontramos se muestran en la Tabla 4.6. Dos de ellas, los valores la temperatura de la planta superior e inferior, se han estimado suponiendo que la diferencia de temperatura que existe entre plantas es pequeña.

Variables del entorno	
Temperatura interior	24.7
Temperatura exterior	31.2
Temperatura planta superior	24.6
Temperatura planta inferior	24.9
Temperatura zona colindante	24.9
Luminosidad exterior	825

Tabla 4.6: Variables extraídas de la sensorización utilizadas en la lógica de control

Tras la optimización, en la Tabla 4.7, podemos ver el gasto energético mínimo, el estado óptimo de las persianas y el número de luces que deberían estar encendidas.

Resultados de la optimización	
Número de luces	8
Porcentaje de ventana sin persiana (%)	63.46
W_{minimo} (kW)	3.87

Tabla 4.7: Resultados obtenidos tras la optimización en la lógica de control

Capítulo 5

Conclusiones y líneas futuras

5.1. Conclusiones

La voluntad de las grandes empresas de generar políticas beneficiosas para el medio ambiente brinda la oportunidad de aplicar los avances tecnológicos a ellas. Esto, unido al incesante incremento de usuarios y dispositivos conectados a Internet, lo hacen aún más fácil. En nuestro caso nos planteamos como objetivo la reducción del consumo de electricidad y agua en un espacio de oficinas. Esta es solo una de las múltiples aplicaciones a las cuales podemos aplicar el IoT en el entorno en el que nos encontramos. Sin embargo y dada la limitación en tiempo que el proyecto tiene nos debemos focalizar en una única aplicación.

A lo largo de esta memoria se han analizado distintas posibilidades en cuanto a hardware, tecnologías de comunicación y software a utilizar, para escoger los elementos del ecosistema IoT a diseñar. Con el fin de conseguir un ahorro en los consumos de electricidad y de agua y de acuerdo a las necesidades de nuestro espacio, se diseñan los distintos bloques funcionales con los correspondientes sensores desde el conexionado hasta el desarrollo del código que deben ejecutar. Estos diseños y la tecnología de comunicación con la que cada bloque se asocia derivan de las conclusiones de los diversos estudios que se realizan. Con todo ello y haciendo uso de las herramientas de análisis pertinentes obtenemos buenos resultados tanto en las zonas de aseos como en la planta a estudiar.

En el caso de los aseos el ahorro en el uso de agua está en torno al 15.5 % y en cuanto a la electricidad destinada a la iluminación es del 47.93 %. Tanto en la planta como en la sala piloto resulta imposible estimar de forma cuantitativa el valor del ahorro sin la implementación real. Sin embargo resulta razonable pensar que un control centralizado de las variables del entorno que derive en el reajuste del modo en el que se ilumina o se climatiza desprenderá un ahorro suficientemente significativo.

Todos estos resultados nos permiten llegar a la conclusión de que los objetivos se han cumplido. Por otro lado, este ahorro, que no es despreciable, nos permitiría compensar en un periodo de tiempo breve la inversión realizada en la modificación de los sistemas de iluminación, persianas, climatización y grifos, así como de los gastos derivados del proyecto.

Más allá de cuantificar el ahorro, las decisiones en cuanto a las tecnologías de comunicación escogidas y a las plataformas software utilizadas se consideran acertadas puesto que dotan a nuestro sistema de una infraestructura sencilla y centralizada, en la que cada bloque actúa de forma aislada pero cooperativa con el núcleo de la infraestructura.

En cuanto al nodo de procesado, este toma un papel fundamental en el proyecto ya que constituye el núcleo de la sensorización sin el cuál no se podría tomar decisiones que favorezcan a todo el entorno. Tanto el lenguaje como la plataforma escogidas se adaptan perfectamente a las necesidades, por lo que se consideran adecuados.

Finalmente, el nodo de visualización, aunque se enmarca como un valor añadido, se considera un gran acierto. La visualización transforma el proyecto de algo que se ejecuta de un modo invisible a algo tangible integrado con el ecosistema de la empresa. Además, permite controlar que el sistema está funcionando como se espera de una forma sencilla y objetiva.

5.2. Líneas futuras

A la vista de que los resultados obtenidos son positivos, este proyecto podría servir como punto de partida para otros proyectos futuros.

La línea futura principal consiste en la implementación real del sistema completo en una de las plantas y la obtención de los valores de los ahorros que se obtienen. A continuación, dando por hecho que el ahorro será positivo, se extrapolaría al resto de plantas sin necesidad de aumentar el número de nodos de procesado. El desarrollo del software propio del nodo de procesado debería ser adaptado para ejecutar varios *"threads"* simultáneos donde en cada uno de ellos se procesaría la información relativa a una planta. Finalmente, el nodo de visualización se ha diseñado con esta visión de futuro, siendo únicamente necesario modificar el software para adquirir la información del resto de plantas.

Por otro lado, cabe también la posibilidad ya no solo de extender la red de sensores dentro de la planta, dividiéndola en zonas de menor tamaño para aumentar la precisión, sino también de integrar sensores que nos permitan obtener otras variables del entorno, como por ejemplo la calidad del aire.

Apéndice A

Metodologías desarrolladas

A.1. Intercambio de mensajes por UDP

La principal justificación de este proyecto se fundamenta en el desarrollo de unas metodologías versátiles que abarquen todo tipo de proyectos. Es por ello que a la hora de elegir entre estudiar los protocolos UDP y TCP o profundizar en mayor medida en solo uno de ellos se prefiere esta últimalínea. Como ya se justificó en apartados anteriores y tras realizar múltiples pruebas se observa que la velocidad con la que se envían los paquetes UDP es superior a la de TCP por lo que se decide experimentar en simular el protocolo TCP mediante el envío de paquetes UDP. Hay que tener en cuenta que en ambas implementaciones se hace uso de librerías de las cuales se desconoce el nivel de optimización.

Como nuestra principal meta es la versatilidad de las metodologías desarrolladas, se cree conveniente dejar de lado la idea de explorar ambas vías y centrarse exclusivamente en el uso del protocolo UDP para el intercambio de mensajes entre dispositivos. Por simplicidad en la configuración de la red dejamos fijos los puertos tanto del coordinador (puerto 300) como del *end-device* (puerto 200). Basándonos en él se genera un bloque con múltiples funcionalidades:

- Establecimiento de conexión mediante el envío de mensajes con dirección broadcast y puerto 200 periódicos cada $t_{broadcast}$, seguidos de la recepción por parte del coordinador del correspondiente OK y envío de confirmación OK/ACK. Una vez establecida la conexión el coordinador y el *end-device* guardan las IPs del dispositivo con el que ha establecido la conexión.

- Mantenimiento de conexión: envío de mensaje CONTROL periódico desde el coordinador cada $t_{control}$. En caso de no recepción durante un $t_{max\_control}$ el dispositivo se desconecta y se reinicia el proceso de establecimiento de conexión.

- Intercambio de mensajes entre dos dispositivos: envío de mensajes a las IPs guardadas en la fase de establecimiento de conexión o pre-configuradas en caso de que no se implementara esa funcionalidad.

- Confirmación de recepción de mensajes mediante el envío de mensaje ACK a la dirección origen por parte del *coordinator*.

- Re-envío de paquetes perdidos en caso de que no se reciba el correspondiente ACK pasado un tiempo t_{ACK}.

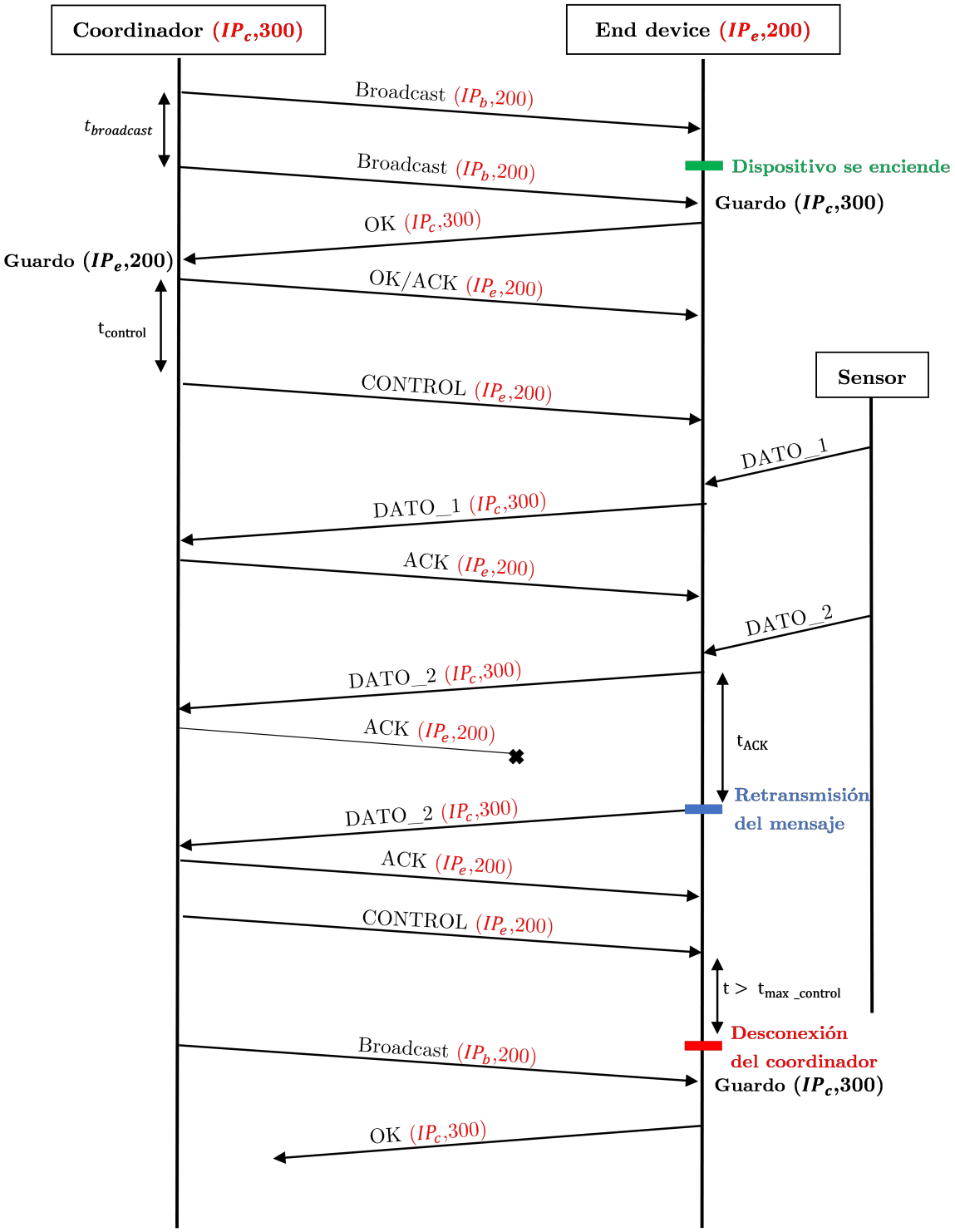


Figura A.1: Diagrama de conexión y transmisión de paquetes UDP

En la Figura A.1 se muestra el diagrama de envío de paquetes abarcando todas estas funcionalidades. Durante el desarrollo de otras metodologías surge la necesidad de desconectar de forma remota alguno de los dispositivos ya bien porque no queremos recibir más información de él, no es un dispositivo que tengamos registrado,... Para ello se define un nuevo comando OFF que se enviará en dicho caso al *end-device*, provocando que este borre la dirección IP que tenía guardada y, por tanto, reinicie el procedimiento de establecimiento de conexión. Un ejemplo de ello se muestra en la Figura A.2, donde el coordinador recibe un mensaje de una IP no registrada y le contesta con un mensaje OFF para forzar la reconexión. De igual modo

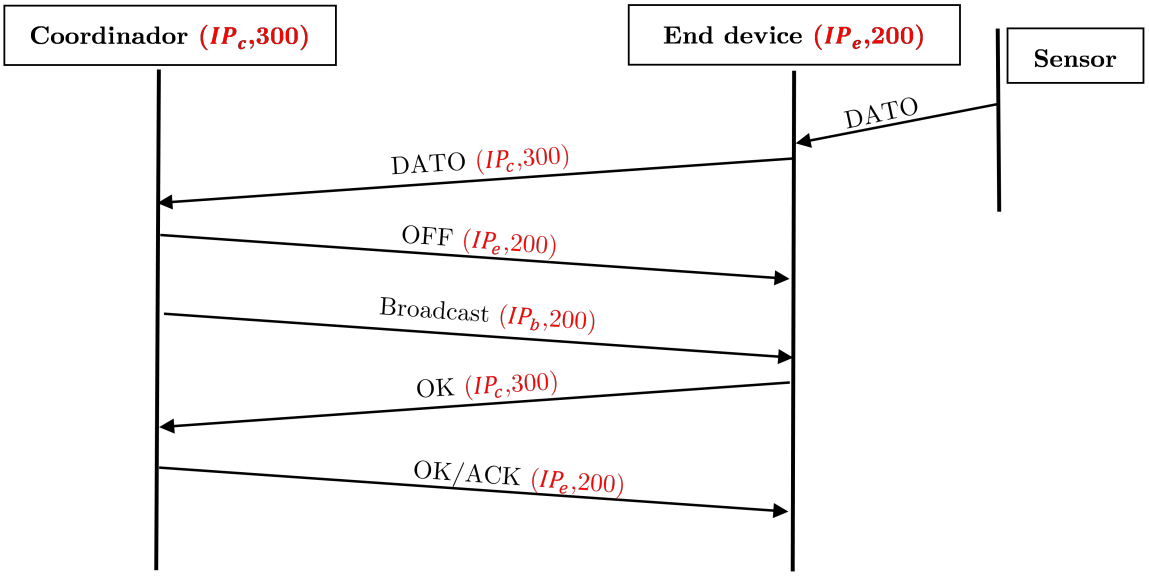


Figura A.2: Diagrama de desconexión de usuario

basándose en esta plantilla es posible de un modo muy sencillo la incorporación de nuevas funcionalidades a través de la configuración nuevos comandos en cualquiera de los dispositivos.

A.2. Bluetooth

Otra metodología desarrollada pero en este caso sí utilizada finalmente en el proyecto es la comunicación Bluetooth. En ella, hay dos módulos HC-05 conectados cada uno de ellos a una placa Arduino y ejerciendo cada uno el rol *"master"* o *"slave"*. La principal desventaja de esta comunicación con estos módulos es que es uno a uno, siendo imposible establecer una conexión simultánea entre varios *slaves* y un único *máster*.

Emparejamiento de los dispositivos

Para realizar la comunicación es necesaria una fase previa de emparejamiento de ambos dispositivos. Esta configuración se realiza mediante comandos AT a través de una comunicación serie entre el ordenador al que conectamos placa y el módulo, utilizando como pasarela la propia placa. El conexionado del módulo con la placa tanto para la configuración de máster y *slave* se muestra en la Figura A.3

Dispositivo slave

La secuencia de comandos para la configuración del módulo *slave* son los siguientes:

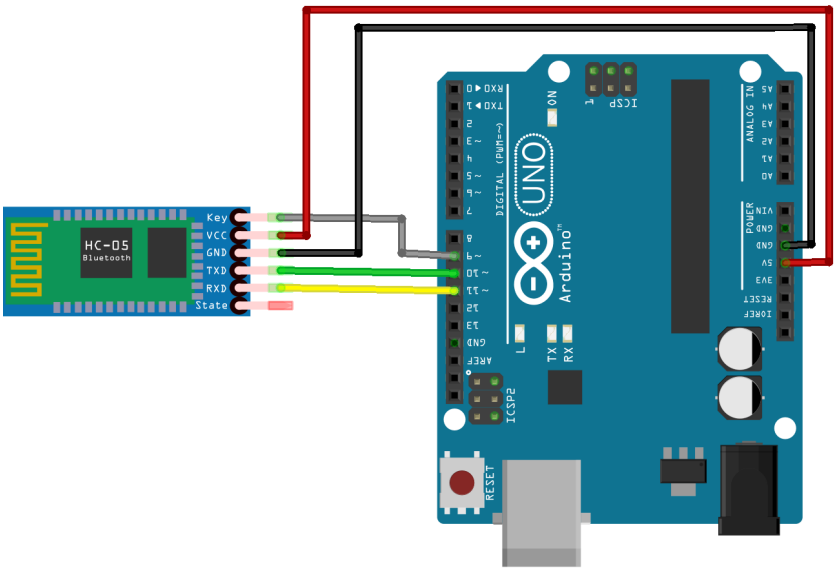


Figura A.3: Esquema de conexiones para la configuración de los módulos Bluetooth

- **AT**: esperaremos como respuesta un OK, que verifica que el conexionado y la comunicación serie es correcta.
- **AT+ROLE=0** para configurar el rol como *"slave"*.
- **AT+ROLE?** para verificar que el rol se ha configurado.
- **AT+ADDR?** para obtener el UUID del módulo al cual se deberá conectar el *"master"*.

Dispositivo master

La secuencia de comandos para la configuración del módulo *master* son los siguientes:

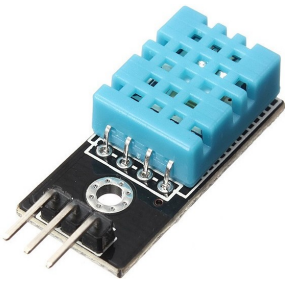
- **AT**
- **AT+ROLE=1** para configurar el rol como *"master"*.
- **AT+ROLE?**
- Existen dos opciones:
  - **AT+BIND=[UUID del *master* separado por comas]** para emparejarlo con un único *slave*.
  - **AT+CMODE=1** para conectarse al primer *slave* disponible

Una vez realizado el emparejamiento debemos desconectar la alimentación de cada una de las placas junto con la conexión entre los pin "KEY" y cada una de las placas. Al reconectar la alimentación de ambas placas el parpadeo de los LEDs de ambos módulos debe ser doble, lo que corresponde con el estado de emparejamiento.

Una vez realizado el emparejamiento podemos transmitir información de modo bidireccional haciendo uso en nuestros sketch de Arduino de los comandos *Serial.write()* y *Serial.read()*.

Apéndice B

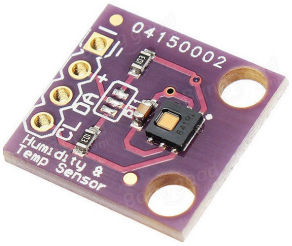
Descripción de sensores utilizados



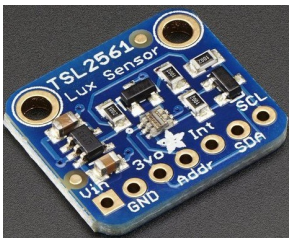
DHT11 [31]. Sensor de temperatura y humedad. Permite medir la temperatura (en °C y F) entre -40 °C y 80 °C con una precisión de 0.5 °C y la humedad (en %) entre 0 y 100 % con una precisión en el rango entre 2 y 4 %. Su frecuencia de muestreo máxima es de 0.5 Hz (1 muestra cada dos segundos). Posee tres pines: dos de ellos se utilizan para la alimentación (5 V y GND) y el restante para la toma de datos por un pin digital haciendo uso de los métodos que nos proporciona la librería $<DHT.h>$.

TMP36 [29]. Sensor de temperatura analógico ampliamente utilizado por sus buenas prestaciones y su sencillez. Su circuito consta de una resistencia y un diodo Zener entre el voltaje positivo de 5 V y tierra, donde el voltaje de corte es proporcional a la temperatura. Para tomar el dato debemos medir la salida de voltaje y realizar la correspondiente conversión lineal. El rango de temperaturas que podemos medir es desde los -40°C hasta los 100°C. La conexión con nuestra placa se realiza mediante la alimentación (5 V y GND) y un pin analógico para la toma de datos.

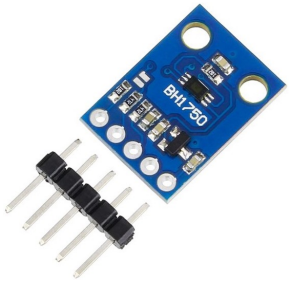
DS18B20 [28]. Sensor de temperatura de bajo precio aunque con un funcionamiento interno complejo que otorga altas prestaciones. Su rango de medición estable va de los -10 °C y 85 °C con una precisión máxima configurable de 0.0625 V. Sus principales ventaja son por un lado que se comercializa con una sonda impermeable que nos permite medir la temperatura exterior o incluso de líquidos. Por otro que funciona con señales digitales lo que nos permite tener largos cables entre el sensor y la placa sin que el deterioro de la señal nos afecte. Su conexión con la placa se realiza mediante el bus 1-Wire que permite, en caso necesario, usar un único conector compartido para comunicación y alimentación gracias a un condensador que almacena energía mientras la línea de datos está en alto. Además gracias a que cada sensor posee un identificador de 48 bits único podríamos conectar 2^{48} sensores a la misma línea de datos. El sensor posee tres pines: dos para alimentación (5 V y GND) y otro para la línea de datos con el que obtenemos la información haciendo uso de la librería *<DallasTemperature.h>*.



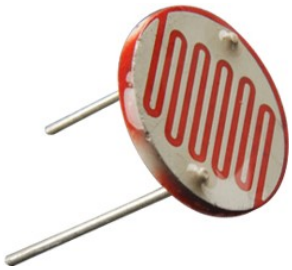
HDC1080 [30]. Sensor de temperatura y humedad de bajo consumo y buenas prestaciones. La precisión en cuanto a la humedad es del 2 % en el rango 10-80 % y en la temperatura de 0.2 °C en el rango -20 a 85 °C. La comunicación con la placa se realiza mediante el bus de datos I2C con una resolución de hasta 14 bits. En cuanto a la alimentación se puede realizar en el rango 2.7 V-5 V. Adicionalmente tiene dos pines de dirección que nos permitiría tener cuatro direcciones distintas del bus I2C.



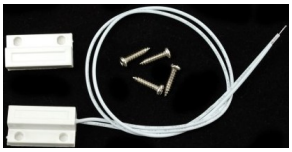
TSL2561 [25]. Sensor de luminosidad del espectro infrarrojo y visible ideal para situaciones con grandes cambios de luminosidad: desde 0.1 a 40.000 lux con una resolución mínima de 1 lux. Su comunicación con la placa se realiza mediante el bus de datos I2C (SDA y SCL) a través de la librería *<TSL2561.h>* y podemos alimentarlo tanto a 3.3 V como a 5 V. Para solventar conflictos entre sensores con la dirección del bus I2C incluye un pin con el que podemos configurar tres direcciones distintas si los conectamos a la alimentación, a tierra o lo dejamos sin conectar. Posee tres modos distintos de funcionamiento con distintos compromisos entre tiempo de medición y resolución de la medida.



BH1750 [26]. Sensor de luminosidad con una respuesta espectral similar a la del ojo humano. Incorpora un conversor ADC de 16 bits por lo que su rango de medición abarca desde 0.11 a 100.000 lux en 2^{16} niveles. La comunicación con la placa se realiza mediante el bus de datos I2C a través de la librería *<BH1750.h>* y su alimentación se puede realizar tanto a 3.3 V como a 5 V. La dirección del bus es 0x23, aunque podemos modificarla conectando el pin de dirección.



Fotorresistor [27]. Resistencia dependiente de la luz que varía desde valores muy bajos cuando la luz es intensa y aumenta conforme se reduce el nivel de luz. Puede ser utilizada como sensor de luminosidad de baja precisión. Para obtener el valor de luminosidad se debe obtener el valor de resistencia mediante un divisor resistivo y a través de la documentación obtener el valor de luminosidad al que corresponde.



Reed-Switch [34]. Sensor electromagnético para controlar si puertas o ventanas se encuentran abiertas o cerradas. Su funcionamiento se basa dos elementos ferromagnéticos que al estar próximos generan una fuerza que provoca el cierre del circuito eléctrico. Su vida útil es limitada aunque oscila en torno a 100 millones de conmutaciones, lo que no supondrá un problema. Consta de dos pines, uno de los cuales se conecta a tierra y el otro a un pin digital de nuestra placa, que tendrá un nivel de tensión bajo cuando el elemento en el que se coloca esté cerrada.

HC-SR04 [32]. Sensor de proximidad basado en el uso de ultrasonidos a 40 Hz. Permite obtener la distancia a la que se encuentra un objeto entre 2 cm y 2 m con una resolución de 1 cm. Utiliza los fundamentos de radar para el envío de un pulso y la espera a recibir el eco con el fin de calcular la distancia a través de la diferencia temporal entre los pulsos. Posee cuatro pines: dos de ellos para la alimentación (5 V y GND) y otros dos para controlar la emisión del pulso y leer la recepción del eco.

PIR Motion Sensor [33]. Sensor de movimiento compuesto por un sensor infrarrojo pasivo capaz de detectar niveles de radiación infrarroja. Tanto su tamaño como su consumo son reducido y es fácil de usar. Posee dos potenciómetros con los que podemos ajustar la sensibilidad y el delay. Posee un ancho de haz de 110º x 70º y un alcance de hasta 6 metros. Posee tres pines: dos de ellos para la alimentación (5V y GND) y un tercero que conectamos un pin digital para la toma de datos.

Apéndice C

Pin-out de las placas Arduino utilizadas

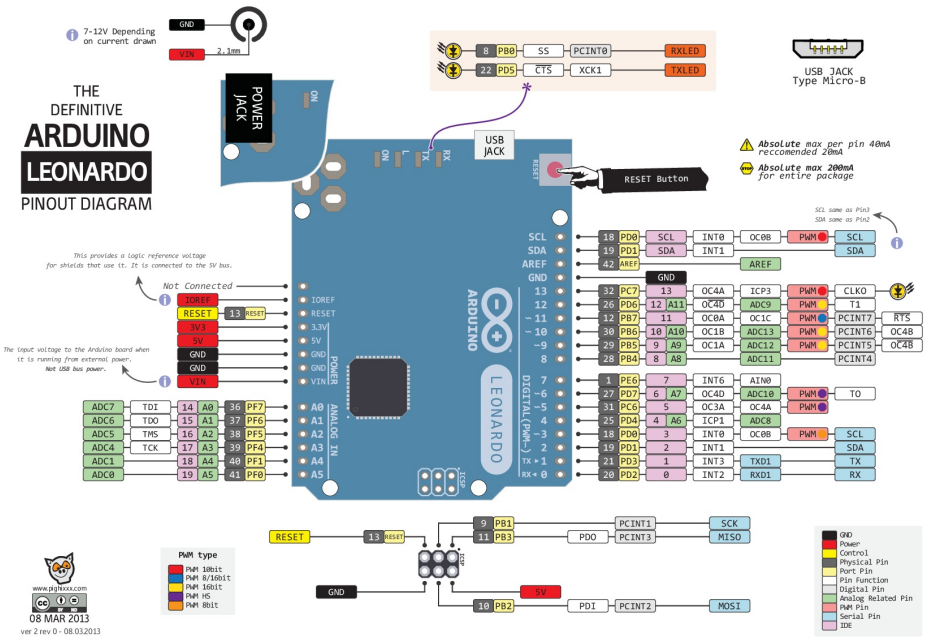


Figura C.1: Pin-out Arduino Leonardo

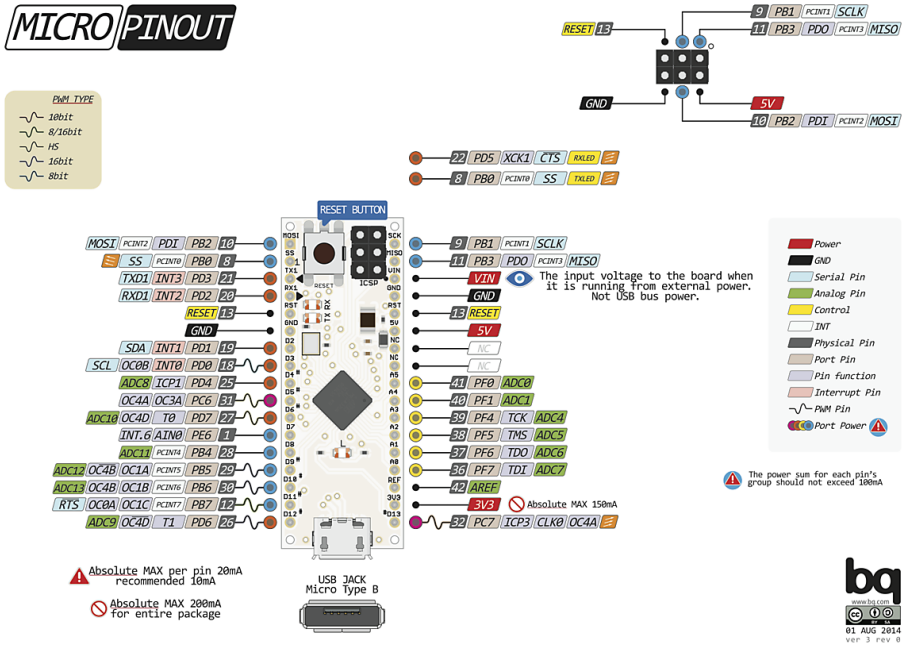


Figura C.2: Pin-out Arduino Micro

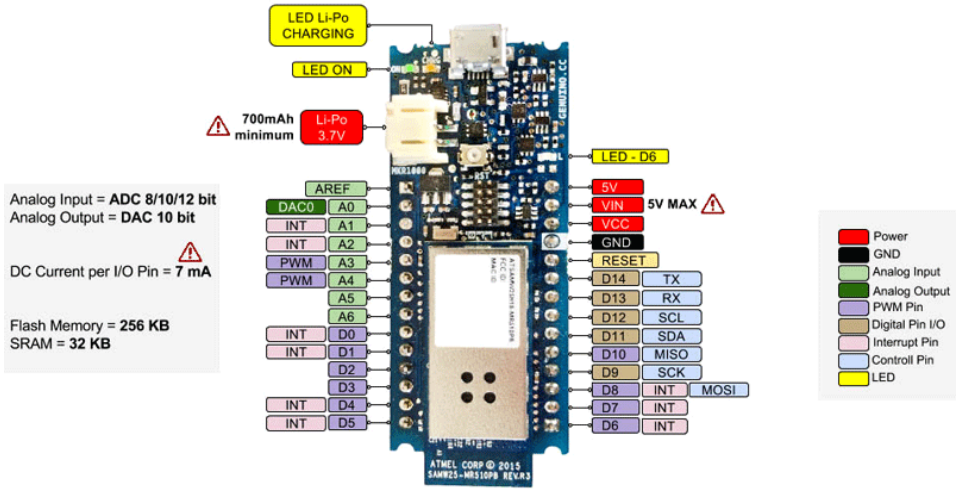


Figura C.3: Pin-out Arduino MKR1000

Lista de acrónimos

AFH Adaptive Frequency Hopping

AMQP Advanced Message Queuing Protocol

CoAP Constrained Application Protocol

CPU Central Processing Unit

DH Destination Address High

DL Destination Address Low

IA Inteligencia Artificial

IFTTT If-This, Then-That

IoT Internet of Things

IP Internet Protocol

MIMO Multiple-input Multiple-output

MAC Media Access Control

M2M Machine-to-Machine

ML Machine Learning

MQTT Message Queue Telemetry Transport

OFDM Orthogonal Frequency Division Multiplexing

QoS Quality of Service

TCP Transmission Control Protocol

TI Tecnologías de la Información

TIC Tecnologías de la Información y Comunicación

RAM Random Access Memory

RTC Real-Time Clock

UDP User Datagram Protocol

WPAN Wireless Personal Area Network

Bibliografía

[1] Kevin Ashton et al. That 'internet of things' thing. *RFID journal*, 22(7):97–114, 2009.

[2] Origen el Internet of Things. http://www.bcendon.com/el-origen-del-iot/,(Último acceso: Julio 2018).

[3] Digital in 2018 - We Are Social. https://wearesocial.com/blog/2018/01/global-digital-report-2018, (Último acceso: Julio 2018).

[4] Andreas Kamilaris, Feng Gao, Francesc X Prenafeta-Boldú, and Muhammad Intizar Ali. Agri-iot: A semantic framework for internet of things-enabled smart farming applications. In *Internet of Things (WF-IoT), 2016 IEEE 3rd World Forum on*, pages 442–447. IEEE, 2016.

[5] Mirza Mansoor Baig and Hamid Gholamhosseini. Smart health monitoring systems: an overview of design and modeling. *Journal of medical systems*, 37(2):9898, 2013.

[6] Arduino. https://www.arduino.cc/,(Último acceso: Julio 2018).

[7] Arduino Micro. https://store.arduino.cc/arduino-micro,(Último acceso: Agosto 2018).

[8] Arduino MKR1000. https://store.arduino.cc/arduino-mkr1000,(Último acceso: Agosto 2018).

[9] Zigbee. https://www.zigbee.org/,(Último acceso: Junio 2018).

[10] Módulo ESP-01 (Espressif Systems).

[11] Datasheet HC-05. http://cesaretopia.com/wp-content/uploads/2017/03/Modulo-Bt.pdf.(Último acceso: Agosto 2018).

[12] Digi Electronics - Comparación de familias. https://www.digi.com/pdf/chart$_{\text{x}}$bee$_{\text{r}}$f$_{\text{f}}$eatures.pdf,(Último acceso: Julio 2018).

[13] Digi XBee. https://www.digi.com/xbee,(Último acceso: Julio 2018).

[14] Digi Electronics - Datasheet ZB-24C. https://www.digi.com/pdf/ds$_{\text{x}}$bee$_{\text{z}}$igbee.pdf,(Último acceso: Julio 2018).

[15] User Datagram Protocol. RFC 768, August 1980.

[16] Transmission Control Protocol. RFC 793, September 1981.

[17] Geng Wu, Shilpa Talwar, Kerstin Johnsson, Nageen Himayat, and Kevin D Johnson. M2m: From mobile to embedded internet. *IEEE Communications Magazine*, 49(4), 2011.

[18] MQTT. http://mqtt.org/,(Último acceso: Agosto 2018).

[19] Zach Shelby, Klaus Hartke, and Carsten Bormann. The Constrained Application Protocol (CoAP). RFC 7252, June 2014.

[20] Dinesh Thangavel, Xiaoping Ma, Alvin Valera, Hwee-Xian Tan, and Colin Keng-Yan Tan. Performance evaluation of mqtt and coap via a common middleware. In *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2014 IEEE Ninth International Conference on*, pages 1–6. IEEE, 2014.

[21] Firebase Realtime Database. https://firebase.google.com/docs/database/?hl=es-419,(Último acceso: Agosto 2018).

[22] Francois Jammes. Internet of things in energy efficiency: The internet of things (ubiquity symposium). *Ubiquity*, 2016(February):2, 2016.

[23] Intorobotis: How to pick the best temperature sensor for your Arduino project. https://www.intorobotics.com/pick-best-temperature-sensor-arduino-project/.

[24] Akram Syed Ali, Zachary Zanzinger, Deion Debose, and Brent Stephens. Open source building science sensors (osbss): A low-cost arduino-based platform for long-term indoor environmental data collection. *Building and Environment*, 100:114–126, 2016.

[25] Adafruit - TSL2561 Luminosity Sensor. https://learn.adafruit.com/tsl2561/overview,(Último acceso: Septiembre 2018).

[26] Datasheet: Mouser - BH1750 Ambient Light Sensor. https://www.mouser.com/ds/2/348/bh1750fvi-e-186247.pdf,(Último acceso: Septiembre 2018).

[27] Datasheet: GL55 - Photoresistor . https://www.kth.se/social/files/54ef17dbf27654753f437c56/GL5537.pdf,(Último acceso: Septiembre 2018).

[28] Datasheet: DS18B20 - 1-Wire Digital Thermometer. https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf,(Último acceso: Septiembre 2018).

[29] Datasheet: TMP36 - Low Voltage Temperature Sensor. http://ctms.engin.umich.edu/CTMS/Content/Activities/TMP35$_3$6$_3$7.pdf,(Último acceso: Septiembre 2018).

[30] Datasheet: HDC1080 - Temperature & Humidity Sensor I2C . https://training.ti.com/how-interface-hdc1080-humidity-and-temperature-sensor-arduino-using-i2c,(Último acceso: Septiembre 2018).

[31] Datasheet: DHT11 - Humidity & Temperature Sensor. https://www.mouser.com/ds/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf,(Último acceso: Septiembre 2018).

[32] Datasheet: HC-SR04 - Ultrasonic Ranging Module. "https://www.mouser.com/ds/2/813/HCSR04-1022824.pdf",(Último acceso: Septiembre 2018).

[33] Datasheet: PIR Motion Sensor. https://cdn-learn.adafruit.com/downloads/pdf/pir-passive-infrared-proximity-motion-sensor.pdf,(Último acceso: Septiembre 2018).

[34] Datasheet: Magnetic Proximity Sensor. http://switches-sensors.zf.com/us/wp-content/uploads/sites/7/2012/05/141215$_{\mathrm{M}}$P2018$_{\mathrm{M}}$agnetic$_{\mathrm{P}}$roximity$_{\mathrm{S}}$ensor$_{\mathrm{D}}$atasheet.pdf,(Último acceso: Septiembre 2018).

[35] Click renovables. Cálculo de la potencia de un aire acondicionado: ratios y método de cargas térmicas.

[36] SDK Firebase para Python. https://firebase.google.com/docs/reference/admin/python/,(Último acceso: Agosto 2018).

[37] Paho-MQTT. https://www.eclipse.org/paho/,(Último acceso: Agosto 2018).

[38] Instituto Nacional de Seguridad e Higiene en el trabajo (INSHT). Ntp 501: Ambiente térmico: inconfort térmico local.

[39] Instituto Nacional de Seguridad e Higiene en el trabajo (INSHT). Iluminación en el puesto de trabajo. criterios para la evaluación y acondicionamiento de los puestos.