Escola d'Enginyeria de Telecomunicació i Aeroespacial de Castelldefels
UNIVERSITAT POLITÈCNICA DE CATALUNYA

# TRABAJO FINAL DE GRADO

**TÍTULO DEL TFG: Monitorización de personas dependientes en interiores**

**TITULACION: Grado en Ingeniería Telemática**

**AUTOR: Albert Álvarez Mirón**

**DIRECTORA: Dolors Royo Vallés**

**FECHA: 3 de septiembre del 2019**

**Título:** Monitorización de personas dependientes en interiores

**Autor:** Albert Alvarez Mirón

**Directora:** Dolors Royo Vallés

**Fecha:** 3 de septiembre del 2019

**Resumen**

El proyecto *ElderPi* nace con el objetivo de poder detectar la posición de una persona dentro de su vivienda sin la necesidad de llevar ningún elemento identificatorio consigo misma. Está enfocado para aquellas personas de edad avanzada o dependientes que viven solas y pasan largos periodos de tiempo sin la supervisión de sus cuidadores. Además, este proyecto permite detectar anomalías en su comportamiento diario y avisar a las personas responsables de su cuidado en caso de ser necesario.

La evolución demográfica para los próximos años deja entrever un futuro, no tan lejano, donde una población envejecida con pocos hijos será el nuevo estándar de núcleo familiar. Esto quiere decir, que la carga de mantener y cuidar a los progenitores por parte de sus hijos se verá incrementada. Por ese motivo, en algunas familias es posible que no dispongan de los recursos necesarios para una correcta supervisión. Debido a estas previsiones, es necesario disponer de un sistema que permita tener controlados en todo momento.

Este sistema se ha construido a partir de materiales de bajo coste y código libre que busca poder llegar a todas aquellas personas que lo necesiten. Su sencilla instalación y mantenimiento es ideal para usuarios poco familiarizados con la tecnología. Además, gracias al uso de tecnologías inalámbricas no es necesario realizar reformas en la vivienda.

La arquitectura del proyecto está formada, en primer lugar, por un servidor, el cual se ocupa de centralizar el flujo de datos entre el usuario y la red de sensores. Por otro lado, una serie de pequeños dispositivos sensores encargados de medir y notificar los cambios pertinentes al servidor se agrupan formando una red colaborativa con un fin bien definido. Finalmente, el usuario puede acceder a los datos desde cualquier navegador con conexión a internet con una visualización simple y clara.

**Title:** Monitoring of dependent people indoors

**Author:** Albert Álvarez Mirón

**Director:** Dolors Royo Vallés

**Date:** September 3rd, 2019

## Overview

The ElderPi project was born with the aim of being able to detect the position of a person inside their home without the need to carry any identifying element with themselves. It is focused for those elderly or dependent people who live alone and spend long periods of time without the supervision of their caregivers. In addition, this project allows you to detect anomalies in your daily behavior and notify the people responsible for your care if necessary.

The demographic evolution for the next few years suggests a future, not so distant, where an aging population with few children will be the new family nucleus standard. This means that the burden of maintaining and caring for parents on the part of their children will be increased. For this reason, in some families it is possible that they do not have the necessary resources for proper supervision. Due to these forecasts, it is necessary to have a system that allows to be controlled at all times.

This system has been built from low cost and open source materials that seeks to reach all those who need it. Its simple installation and maintenance is ideal for users unfamiliar with technology. In addition, thanks to the use of wireless technologies it is not necessary to carry out reforms in the home.

The architecture of the project is formed, first, by a server, which is responsible for centralizing the flow of data between the user and the sensor network. On the other hand, a series of small sensor devices responsible for measuring and reporting changes relevant to the server are grouped together forming a collaborative network with a well-defined purpose. Finally, the user can access the data from any browser with an internet connection with a simple and clear visualization.

*Agradecimientos*

*Estas palabras van dedicadas a todas aquellas personas que han colaborado en la realización de este proyecto. En primer lugar, a la responsable de la tesis, Dolors Royo, por su apoyo académico y difusión ante organizaciones y entes públicos con tal de obtener la financiación necesaria para llevar a cabo la primera prueba piloto en un entorno real.*

*A mi pareja y su padre por ser los impulsores de la idea original junto a la primera aproximación de ElderPi durante las vacaciones de Navidad. Agradecer el apoyo de mis padres y familiares durante el desarrollo y su disposición en todo momento.*

*Por último, este trabajo tiene una mención especial, a mi abuela, Maravillas Sánchez Marín quien accedió a ser monitorizada y prestó su vivienda durante las pruebas. Además de ayudar a conformar los requisitos necesarios del sistema según sus necesidades.*

# ÍNDICE

# INTRODUCCIÓN

En el presente documento se intenta dar respuesta a un problema que, cada vez más, sufren muchas familias en todo el mundo. Nos referimos a aquellas que no disponen de los recursos necesarios para atender las 24 horas a personas de edad avanzada o dependientes. Por ese motivo, muchas veces estas personas se quedan horas, o en el peor de los casos, pueden llegar a estar días sin la supervisión de la persona encargada de esa tarea por diversos motivos.

Este producto, el cual se ha bautizado como *ElderPi*[1], tiene un objetivo muy claro, poder monitorizar la ubicación de una persona dentro de su casa sin la necesidad de llevar elementos identificatorios como balizas u otros métodos de localización en interiores. De esta forma poder analizar patrones de comportamiento y detectar conductas anómalas y actuar en consecuencia.

A lo largo de esta investigación y desarrollo han surgido diferentes intereses e inquietudes. En primer lugar, un marcado componente social, el cual busca facilitar y mejorar la calidad de vida de las personas. Permitiendo al cuidador disponer de más flexibilidad horaria y a la persona monitorizada gozar de cierta independencia en los casos en que sea posible. A continuación, se ha procurado dar salida a una inquietud de carácter económico, el cual busca ofrecer a la sociedad un producto de bajo coste con el fin de acercar la tecnología a todas las familias que lo necesiten sin importar su estrato social y/o formativo. Finalmente, y no por ello menos importante, el interés académico que este proyecto puede llegar a suscitar, pues procura mantener la corriente filosófica de los proyectos de *código abierto* donde cualquier persona puede obtener el código fuente y los diseños necesarios con tal de poder aportar mejoras y críticas constructivas.

La estrategia a seguir para el desarrollo de este producto ha estado la de preparar un prototipo simple pero funcional en el cual se han ido añadiendo módulos y funcionalidades que han desembocado en una posible solución para el posicionamiento de interiores la cual se detallará más adelante. Cabe destacar que en esta primera versión del proyecto se ha optado por una arquitectura descentralizada con una base de datos alojada en un servidor local en cada vivienda. Por tanto, cada usuario es responsable del trato que hace de sus propios datos y tiene control absoluto sobre ellos.

[1] Código disponible en https://github.com/alberti-tu/ElderPi

Con el objetivo de tener un orden lógico a la hora de exponer los distintos conceptos e ideas en este proyecto se ha optado por redactar este documento en capítulos donde se irá desde las capas más externas al detalle de los diferentes componentes que intervienen en el proceso y siguiendo la línea de flujo que siguen los datos desde que se generan hasta que llega al usuario final. Por tanto, la distribución del documento es:

- Capítulo 1: Contexto Social

En él se expondrán las distintas causas que han originado un cambio de tendencia en la relación de los hijos respecto al cuidado de sus progenitores y porque es necesario esta herramienta que se expone en este documento.

- Capítulo 2: Red de sensores

Aquí se discutirá sobre como se definen actualmente las redes de sensores, que son y que implementación se ha decidido emplear en este proyecto desde una perspectiva a alto nivel.

- Capítulo 3: Sensor

Aquí nos encontramos con uno de los 2 pilares maestros en los que se basa este proyecto, las pequeñas fuentes generadoras de datos que serán necesarias para poner en funcionamiento nuestra red de sensores. A diferencia del capítulo anterior aquí sí se entrará en mayor detalle para explicar la electrónica que hace posible la generación de los datos.

- Capítulo 4: Servidor

La segunda pieza maestra de ese rompecabezas se encuentra aquí, en el servidor, la piedra angular de todo el proyecto que permite ser un punto de encuentro entre humanos y sensores. Será un apartado diverso y con mucho detalle donde se hablará de seguridad, protocolos, algoritmos…

- Capítulo 5: Cliente web

En este capítulo se discutirá como se consigue llevar los datos desde el servidor hasta la persona que los necesite y también el como se muestra para que sea amigable, robusto y útil.

- Capítulo 6: Entorno de producción

En esta sección se pondrá a prueba todo el sistema para comprobar el funcionamiento en un entorno real y que resultados extraemos de ellos, si son satisfactorios o no y en qué medida.

- Capítulo 7: Propuestas de mejoras

Para concluir, en esta sección se detallarán los próximos objetivos de este proyecto para así poder seguir creciendo y mejorando poco a poco.

# CAPÍTULO 1. CONTEXTO SOCIAL

A continuación, se expondrán los principales motivos por los cuales se está produciendo un cambio demográfico a escala global. Para ello, se emplearán dos índices para estudiar este efecto, la distribución demográfica de la sociedad y la fertilidad (número de hijos promedio por cada mujer).

## 1.1 DEMOGRAFÍA

Diversos estudios apuntan a que la población mundial cada vez está más envejecida a causa de diversos factores como pueden ser el incremento de la esperanza de vida (mejor calidad de vida y sanidad) y una disminución del número de hijos que conciben las familias (familias más pequeñas y se forman a una edad más tardía). Estudios como el realizado por el CSIC [1], sitúan que en el año 2050 más del 30% de la población tendrá más de 65 años (actualmente ronda el 20%).

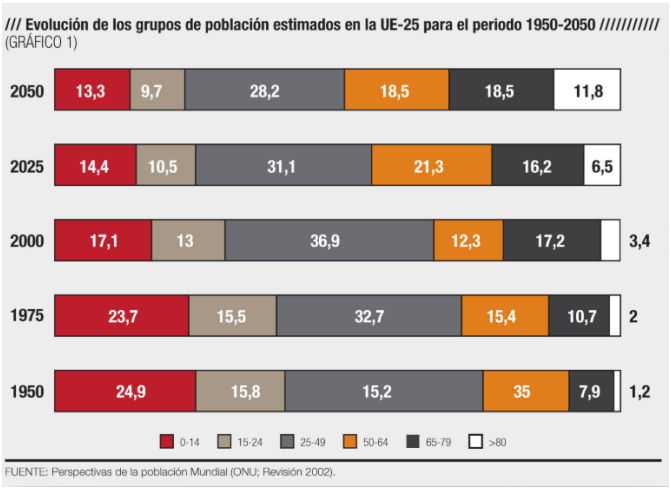



**FIG 1.** *Evolución de la franja de edad en la UE (1950 – 2050)*

Particularizando el tema a España, esta tendencia se agrava si consideramos a las personas que viven en hogares unipersonales (es decir, aquellos hogares que únicamente disponen de un único habitante) y tienen más de 65 años. Según los últimos datos disponibles del Instituto Nacional de Estadística [2] entre el año 2017 y 2018 este colectivo ha aumentado en 76000 personas (3.9%). Siendo mayoritariamente mujeres las integrantes de este grupo demográfico (para la franja de edad de mayores de 85 años se sitúa en un 42.7% a las mujeres que viven solas frente al 23.6% de los hombres). Por tanto, en estos datos podemos apreciar que un gran número de ancianos en España afrontan la vejez en soledad.

## 1.2 FERTILIDAD

La fertilidad de define como el número medio de hijos que una mujer tiene y, según datos del Banco Mundial [3], la fertilidad ha descendido drásticamente en los últimos 60 años. Estos datos suponen un cambio de tendencia en la composición de la familia tal y como la conocemos en la actualidad. Se prevé que en unos años el estándar de familia será el de unos progenitores envejecidos con una media de 1 o 2 hijos por núcleo familiar frente a los 5 que nacían en 1960.

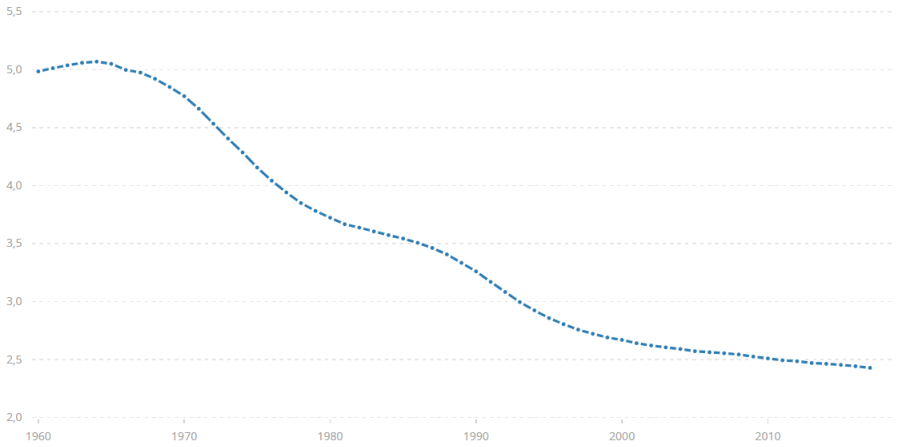



**FIG 2**. *Evolución de la fertilidad mundial (1960 – 2017)*

Sin embargo, otro de los mayores problemas a los que se enfrenta la sociedad en su vejez (a parte de la soledad que se exponía en el apartado 1.1) es a la falta de atención por parte de sus hijos. A veces, por falta de medios y otras por falta de conciliación laboral y familiar. Hasta hace unos años los hijos tenían mayor facilidad al poder repartirse las tareas entre los distintos hermanos mientras que ahora, en ser cada vez menos miembros los que componen la unidad familiar, supone una carga de trabajo mayor para cada hijo.

Para el caso de España la tendencia es la misma, sin embargo, aquí la fertilidad ha descendido desde los 2,9 hijos por mujer (1960) a los 1,7 (2017). Esta tendencia hace que en muchos hogares la persona dependiente pueda pasar horas o, en el peor de los casos, días sin recibir atención por parte de sus cuidadores. Esta situación hace que las personas dependientes queden a merced de padecer un accidente doméstico (caídas, desmayos…) y no poder comunicarse con el exterior hasta que alguien se percate del suceso.

La solución en estos casos es la de poder monitorizar a esa persona en todo momento de forma telemática y automática para saber si todo transcurre correctamente. Siendo este el pilar principal en el que se sustenta este proyecto de telemetría doméstica, comprobar el estado de una persona y avisar lo antes posible frente a anomalías.

# CAPÍTULO 2. RED DE SENSORES

En esta sección del documento se expondrán las características y requisitos necesarios para poder crear una red de sensores [4] capaces de recoger la información de su entorno y hacerla llegar hasta una base de datos donde será procesada y filtrada para, a posteriori, ser mostrada al usuario final.

## 2.1 CARACTERÍSTICAS PRINCIPALES

Una *red de sensores* consiste en interconectar pequeños aparatos electrónicos autónomos (nodos) los cuales son capaces de medir magnitudes físicas (temperatura, luz, humedad, distancia…) y entre todos ellos conseguir llevar a cabo una tarea común bien definida.

Los nodos más externos suelen disponer de un limitado *hardware*, una CPU a unos pocos megahercios y una interfaz radio de baja potencia. Estos microprocesadores cuentan con la capacidad de reducir drásticamente su consumo durante largos periodos de tiempo para poder extender la vida útil de sus fuentes de alimentación (baterías, pilas o placas fotovoltaicas). Para que estas máquinas sean realmente eficientes se debe llegar a un compromiso entre las limitaciones impuestas por su *hardware* y un *software* que permita una correcta gestión de los recursos.

Cuando diversos nodos se interconectan los unos con los otros se forman las ya mencionadas redes de sensores. A este tipo de intercambio de información se le conoce como comunicaciones máquina a máquina o por sus siglas en inglés M2M. Este tipo de implementaciones suelen estar basadas en una topología de red dinámica a partir de redes inalámbricas multisalto[2]. Sin embargo, la simplicidad de nuestra red, sus dimensiones físicas y la localización de sus elementos hace que no sea necesario mantener un algoritmo de enrutamiento dinámico (se detallará en la sección 2.3).

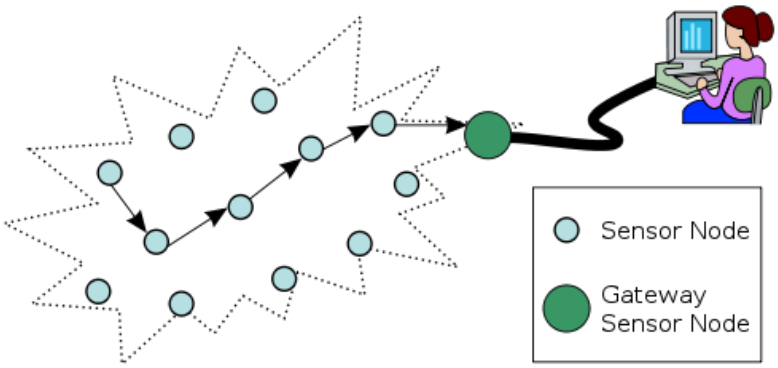



**FIG 3.** *Red de sensores multisalto*

Por último, cabe destacar que el tiempo de ida y vuelta (RTT) entre el sensor y el servidor está en el orden de milisegundos mientras que entre el servidor y el sensor no está definido y varía en función del ciclo de actividad del sensor.

[2] Algunos ejemplos de protocolos multisalto son AODV, DSDV y EWMA

## 2.2 TECNOLOGIAS INALÁMBRICAS

En la actualidad existen diferentes arquitecturas que permiten la creación de redes inalámbricas donde las más usuales de encontrar son las propuestas bajo las normas del *Institute of Electrical and Electronics Engineers* (IEEE):

- IEEE 802.11 (Wi-Fi) [5]
- IEEE 802.15.1 (Bluetooth) [6]

O bien, la propuesta de la ZigBee Alliance que aglutina una serie de protocolos de alto nivel:

- ZigBee

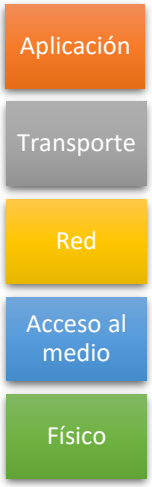



**FIG 4.** *Pila de protocolos TCP/IP*

Cabe destacar que todas esas tecnologías han adaptado su estructura para hacerse compatibles bajo IP[3] en la capa de red.

Por ello vamos a analizar qué ventajas e inconvenientes nos ofrece cada una de las propuestas anteriormente citadas.

### 2.2.1 WI-FI

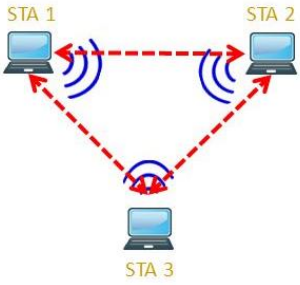
El primer estándar Wi-Fi se publicó en 1997 y actualmente es la tecnología inalámbrica más utilizada en el mundo, gracias a su interoperabilidad con gran cantidad de fabricantes. Opera en la banda IMC de 2,4GHz y 5GHz a través de un protocolo CSMA/CA[4] el cual permite repartir el acceso a un medio compartido (aire). Sin embargo, estas bandas están empezando a saturarse por la gran cantidad de usuarios que acceden a ellas reduciendo el rendimiento de la red.

Las redes Wi-Fi permiten establecer dos modos de uso, el primero y más utilizado, el modo infraestructura. En esta arquitectura la red se debe diseñar e implementar *a priori* gracias a unos dispositivos especiales llamados “puntos de acceso”, los cuales permiten centralizar el tráfico de paquetes y después redirigirlos hacia sus respectivos destinos. Permite el escalado de la red y es el modo de uso más adecuado para redes permanentes. En segundo lugar, encontramos las redes ad-hoc, se caracterizan por una arquitectura distribuida, no tienen una planificación previa y suelen tener un reducido ciclo de vida para la cual se busca realizar una tarea específica. En este modo de uso son los propios nodos los encargados de enrutar el tráfico de la red. Suelen tener un despliegue rápido y precisan de poco mantenimiento, pero a partir de un cierto número de nodos conectados la red tiene un mal escalado.

Cabe destacar que Wi-Fi comparte la capa 2 con las tramas Ethernet por lo que estas redes son perfectamente compatibles entre ellas.

---

[3] Especificaciones de IP disponibles en RFC 791 (IPv4, 1981) y RFC 2460 (IPv6, 1998)
[4] Acceso Múltiple por detección de portadora y prevención de colisiones



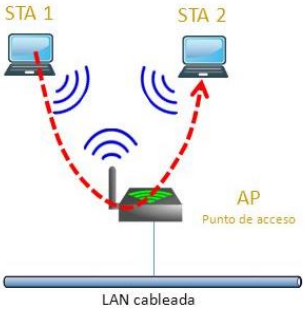




**FIG 5.** *Red Wi-Fi en modo ad-hoc* **FIG 6.** *Red Wi-Fi en modo infraestructura*

Sin embargo, uno de los puntos más desfavorables de esta tecnología es, en primer lugar, el bajo rendimiento de la red debido al *performance anomaly*, donde la velocidad media de la red se ajusta a la velocidad del dispositivo con una tasa de envío más reducida[5]. Y, en segundo lugar, la gran cantidad de pausas que existen en la transmisión y sus ventanas de transmisión aleatorias para evitar colisiones.

### 2.2.2 BLUETOOTH

En 1995 se presenta el estándar IEEE 802.15.1 enfocado a las redes inalámbricas de área personal, las cuales han tenido un gran éxito para las comunicaciones en aparatos de uso cotidiano como manos libres, teclados, relojes inteligentes… Está basado en un protocolo TDMA[6], el cual asigna a cada dispositivo un espacio de tiempo para realizar la transferencia de información que deseen, y en saltos frecuenciales para evitar colisiones con otras tecnologías las cuales también operan en la banda IMC de 2,4GHz.

Una de sus mayores ventajas es la simplicidad de su arquitectura maestro-esclavo en modo ad-hoc. En sus primeras versiones, las conexiones se realizaban en forma de estrella siendo el dispositivo maestro el centro de la red (piconet) y los dispositivos esclavos a su alrededor. Cabe destacar que un dispositivo puede pertenecer a diversas piconets de forma simultánea y actuar de puente entre ellas (a la interconexión de dos o más piconets se le denomina scatternet). Sin embargo, por cada piconet que se añade a la red, el rendimiento de esta decaía en un 10% de media.

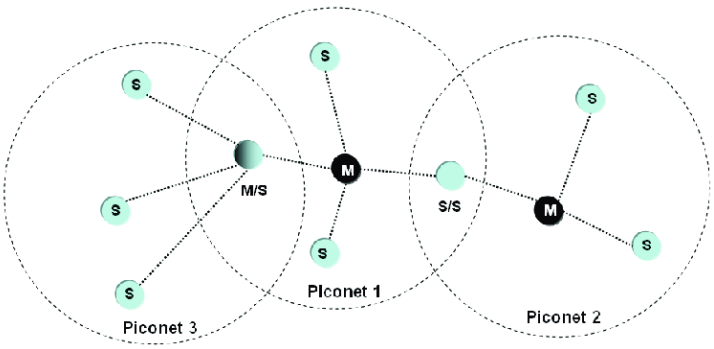



**FIG 7**. *Scatternet formada por 3 piconets Bluetooth*

[5] Dispositivo antiguo o muy alejado de su punto de acceso
[6] Acceso al medio por división de tiempo

No obstante, en 2015 se publica Bluetooth Mesh, que se basaba en la 4ª versión de Bluetooth (también conocido como Bluetooth Low Energy) permitiendo crear redes malladas que permitían ser más robustas además de mejorar el rendimiento general de la red. Al estar basadas en Bluetooth v4, el ahorro y la optimización de la batería era uno de sus principales objetivos, una característica muy importante a la hora de implementar redes de sensores. Esta tecnología, según su estándar permite crear redes de hasta 1km de diámetro.

Sin embargo, la principal desventaja que presenta esta tecnología es su tasa de envío, siendo esta de 1 Mbps. Para transmisiones cortas no debería suponer mayor problema, pero si se requiere una transferencia de mayor cantidad supone tener la interfaz radio más tiempo encendida y puede afectar a la duración de la autonomía del dispositivo.

### 2.2.3 ZIGBEE

El origen de ZigBee es algo distinto a las dos últimas tecnologías comentadas, se basa en el estándar para tecnologías inalámbricas de área personal (IEEE 802.15.4) el cual fue presentado en 2003 tras una agrupación entre diversos fabricantes los cuales buscaban una solución para el ámbito de la domótica.

Se caracteriza por disponer de una pila de protocolos propia y se puede personalizar la capa de aplicación con los denominados “perfiles”, una serie de funciones estandarizadas listas para ser utilizadas por los dispositivos. Algunos ejemplos son los perfiles para el control remoto de luces, cámaras de seguridad… También permiten agrupar los distintos dispositivos maestro-esclavo en topologías de estrella, árbol y malla con un bajo consumo eléctrico.

Sin embargo, cuenta con una tasa de envío bastante reducida de tan solo 250 kbps. Con lo cual las comunicaciones deben de ser concisas o el tiempo que un dispositivo se encuentra despierto será demasiado largo.

En las redes ZigBee se puede optar por el uso de balizas, las cuales permiten asignar un espacio temporal a cada dispositivo que quiera transmitir previo registro ante el coordinador de la red. De forma periódica el coordinador envía una baliza (también llamada supertrama) que permite notificar a un dispositivo de la red si tiene mensajes pendientes para entregarle y organizar las comunicaciones entre los dispositivos y el coordinador con tal de evitar interferencias. Este método es muy recomendable cuando el coordinador está alimentado por baterías pues al ser un método determinista el consumo de energía se puede planificar con bastante exactitud.

Por otro lado, se puede evitar el uso de tales balizas y las redes ZigBee pasan a ser redes de acceso al medio aleatorias donde el coordinador debe permanecer a la escucha ante las posibles comunicaciones del resto de dispositivos. En esta implementación las transmisiones pueden sufrir interferencias y debe existir un mecanismo de retransmisión.

Para cerrar esta sección mostramos el siguiente cuadro resumen con las ventajas e inconvenientes entre las distintas tecnologías inalámbricas que hemos analizado.

| | Ventajas | Desventajas |
|---|---|---|
| **Wi-Fi** | Tecnología muy extendida<br>Alta tasa de envío<br>Integración con Ethernet | Performance Annomaly<br>Tiempos de espera |
| **Bluetooth** | Bajo consumo de energía<br>Redes dinámicas<br>Distintos tipos de topologías | Protocolo de enrutamiento<br>Baja tasa de envío |
| **ZigBee** | Pila completa de protocolos<br>Uso de supertramas | Poco personalizable<br>Baja tasa de envío |

**FIG 8.** Ventajas e inconvenientes entre tecnologías inalámbricas

Bluetooth Mesh es, para nuestro proyecto, el mejor candidato para ser la tecnología para implementar en la capa física de la red de sensores. Sin embargo, basaremos nuestra red en el estándar Wi-Fi porque es la que proporciona mayor escalabilidad y simpleza ya que está muy presente en el día a día del usuario final.

## 2.3 ARQUITECTURA DE RED

Para poder realizar la tarea de monitorizar una vivienda es necesario el despliegue y configuración de los distintos dispositivos que intervendrán. A continuación. También se evaluará la seguridad y autenticación de usuarios/sensores en la red.

En este tipo de implementaciones, el servidor ocupa un papel central en la red siendo el nexo de unión entre la red de sensores y el usuario final. Para ello, el servidor debe disponer de al menos una interfaz inalámbrica Wi-Fi[7] que se pueda configurar como punto de acceso. De esta forma, se creará una nueva red sin cables donde todos los sensores irán conectados y enviarán directamente la información al servidor formando una estructura en estrella[8]. El direccionamiento de esta nueva red se puede realizar a partir de un servidor DHCP local (en nuestro caso, se asigna el rango IP 10.0.42.0/24). Además, se ha levantado un *firewall NAT* con una serie de normas que permite el tráfico saliente de la red, pero no el que se inicia desde fuera. Con estas normas el servidor puede autenticar la procedencia de los paquetes a partir de la interfaz en que los recibe. Un ataque de MITM es posible si la clave Wi-Fi se ve comprometida, por ese motivo debe de ser lo más aleatoria posible.

[7] Asignar la red al canal menos saturado para evitar posibles interferencias.
[8] Si la cobertura Wi-Fi no es la adecuada en algunas zonas de la vivienda se puede expandir la red con repetidores.

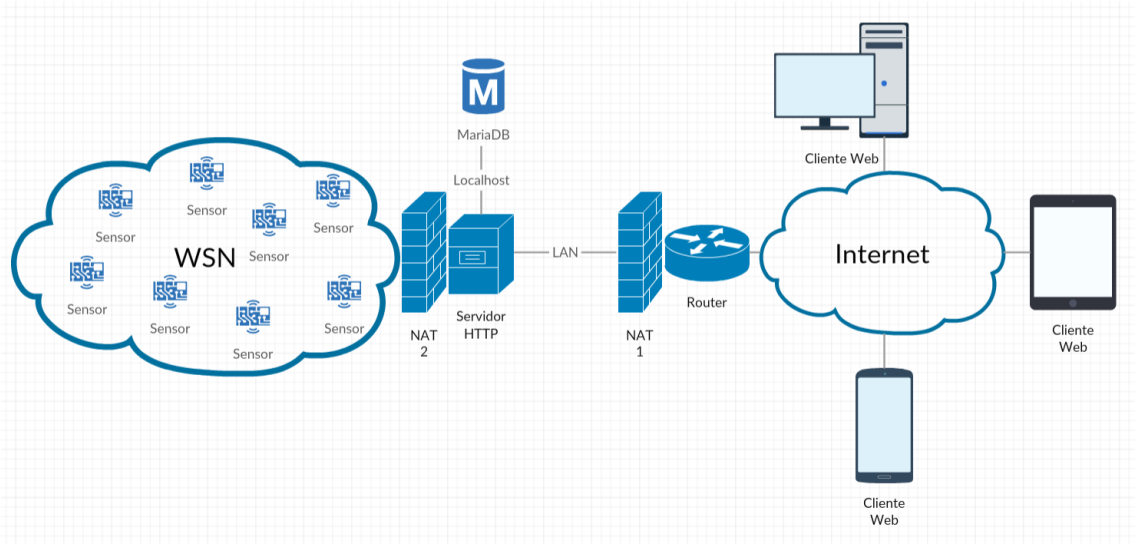



**FIG 9.** *Topología por defecto para ElderPi*

Como se detallará más adelante, la placa de desarrollo de los sensores no permite el uso de HTTPS y la seguridad de la información de la red irá cifrado únicamente con la seguridad WPA2 que proporciona Wi-Fi. Por este motivo, se debería restringir el acceso a esta red únicamente a los sensores que la forman.

Paralelamente el servidor debe tener acceso a Internet mediante el *router* que proporciona su ISP. Esta conexión se realizará siempre sobre HTTPS a partir de otra interfaz de red, ya sea mediante cable Ethernet[9] u otra conexión Wi-Fi. Para poder exponer los servicios al resto de Internet se deben modificar la configuración de nuestro *router*. Es decir, redirigir el tráfico TCP[10] de entrada hacía la IP que tengamos asignada a la interfaz LAN del servidor (se recomienda el uso de IP estática). De esta forma si conocemos la IP pública del *router* y nos conectamos a los puertos que se han redireccionado podremos establecer una conexión segura desde Internet hacia nuestro servidor.

*Nota: Muchos operadores ISP no disponen de direccionamiento público estático[11], por ese motivo, se recomienda para estos casos el uso de entidades externas que proporcionan Dominios de Nombres Dinámicos (DDNS) para así identificar nuestra IP pública dinámica bajo un nombre estático. Ejemplos de proveedores de DDNS gratuitos [7] son:*

- *No-IP*
- *Con-IP*
- *Duck DDNS*
- *FreeDNS*

[9] El uso de cable trenzado permite obtener un ancho de banda y estabilidad mayores que Wi-Fi.
[10] Asignar puertos para conexiones entrantes HTTP y HTTPS sobre TCP.
[11] La dirección IP pública cambia cada cierto tiempo (ejemplo: expira al cabo de 1 semana).

# CAPÍTULO 3. SENSOR

En los siguientes apartados se explicará la base tecnológica y funcional de los dispositivos que intervienen en la medición, procesado y transmisión de información hasta llegar al servidor.

## 3.1 PLACA DE DESARROLLO

El elemento central de nuestro sensor es la placa de desarrollo *Particle Photon* [8], un módulo dotado de un microprocesador con un sistema operativo FreeRTOS[12] que maneja los eventos del sistema, una interfaz de red inalámbrica (Wi-Fi), conexiones GPIO para adherir circuitos electrónicos auxiliares y un sensor PIR de presencia para establecer los diferentes estados del sensor.

En primera instancia el dispositivo inicia su *set up* para cargar el sistema operativo y el código[13] de ejecución. A continuación, se hace una lectura del estado del sensor y se le aplica un algoritmo booleano para establecer la siguiente acción. En caso de que el resultado sea "falso", el dispositivo desconecta la CPU y entra en un profundo sueño para ahorrar la máxima energía posible hasta que exceden 10 segundos o llege una señal de *wake up* para iniciar una nueva lectura. Por el contrario, si el resultado es "verdadero", se activa la interfaz Wi-Fi y se hace un escaneo de red para establecer la conexión con el *Acces Point* más cercano. Si logra establecer la conexión transmite los datos al servidor, apaga la interfaz radio y entra en modo ahorro de energía. De no ser así, el sensor desecharía el paquete de información, entraría en modo ahorro de energía y esperaría a tener de nuevo algo que notificar al servidor.

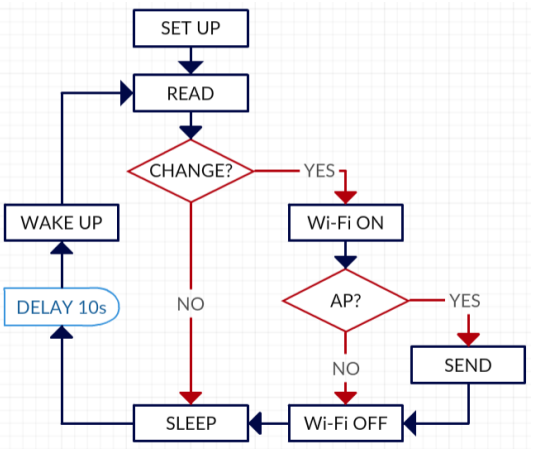



**FIG 10.** *Diagrama de flujo del sensor*

Una primera aproximación del mecanismo para notificar cambios de estado al servidor sería realizar una transmisión al entrar y otra al salir una persona de una habitación a otra. Sin embargo, este método es ineficiente desde el punto de vista energético y consistencia de la información ya que para cada cambio se deben realizar dos transmisiones. Como se supone que solo habrá una persona en la vivienda nos permite asegurar que esta persona no puede estar en dos habitaciones al mismo tiempo, por tanto, basta con notificar el cambio al servidor desde el sensor que detecta la entrada de una persona en su área de acción. Este protocolo permite reducir el número de transmisiones necesarias a la mitad, aumentando la autonomía de los sensores y evitando posibles interferencias debido a que el sensor que notifica la salida y el sensor que notifica la llegada del individuo transmitirían al mismo tiempo.

[12] Sistema operativo de tiempo real para objetos inteligentes.
[13] Para consultar el código en C, ver *Anexo 1: Código del sensor*.

Desde el punto de vista de álgebra booleana solo se realizará la transmisión de información entre sensor y servidor cuando su estado pase de “falso” a “verdadero”, es decir, cuando entre una persona en la habitación. Además, esta transmisión va acompañada de una marca temporal y permite saber el tiempo que ha transcurrido en la anterior habitación.

| Estado actual (p) | Nuevo estado (q) | Envío (r) |
|---|---|---|
| Falso | Falso | Falso |
| Falso | Verdadero | Verdadero |
| Verdadero | Falso | Falso |
| Verdadero | Verdadero | Falso |

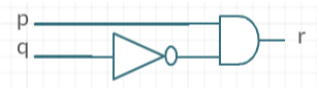



**FIG 11.** *Tabla de la verdad para el algoritmo de transmisión y esquema lógico*

Una vez definido en qué circunstancias se enviarán los datos falta ahora definir el protocolo de transmisión y que información se requiere. La comunicación entre sensor y servidor se realizará a través del método POST de HTTP[14] ya que al ir sobre el protocolo de transporte TCP nos asegura la correcta recepción de la información en el destino y de no ser así nos permite realizar retransmisiones e integridad en él mensaje. La información que se envía en el cuerpo del mensaje se codifica con el estándar JSON con los campos “deviceID” y “battery” siendo el identificador único del cual dispone cada *Photon* y el nivel de batería en tanto por ciento respectivamente.

## 3.2 SENSOR DE PRESENCIA

El sensor de presencia que se ha utilizado pertenece a la familia de detectores PIR o infrarrojos pasivos. La principal característica de este tipo de sensores es que no requieren de una interacción con el exterior, tan solo se limitan a recibir las ondas infrarrojas cercanas. Estas ondas electromagnéticas se producen por el movimiento de los electrones en los cuerpos con una temperatura superior a los 0ºK tal y como indica la ley de desplazamiento de Wien, la cual establece la longitud de onda para la cual la energía de emisión de ese cuerpo es máxima.

$$\lambda_{max} = \frac{0{,}0028976\ \mathrm{m}\cdot{}^{\circ}\mathrm{K}}{\mathrm{T}}$$

Por tanto, si la temperatura corporal media de un ser humano ronda los 36ºC. El pico de energía se sitúa cerca de los 10µm (banda infrarroja).

$$\lambda_{max} = \frac{0{,}0028976}{(36+273)^{\circ}\mathrm{K}}\ \mathrm{m}\cdot{}^{\circ}\mathrm{K} \leftrightarrow \boldsymbol{\lambda_{max} = 9{,}38\ \mu m}$$

El dispositivo HC-SR501 [9] consta de dos partes bien diferenciadas. La primera, una lente de Fresnell, fabricada a partir de polietileno de alta densidad que permite el paso de ondas con una longitud de onda entre 8 y 14 µm. Permite concentrar la radiación con un ángulo de 110° en el centro del encapsulado. La

[14] Aún no se ha diseñado una librería HTTPS estable para esta plataforma de desarrollo.

segunda es el sensor PIR, un receptor con dos fotodiodos separados a una cierta distancia. La diferencia de potencial a la salida de los diodos se aumenta con un amplificador operacional diferencial. Si esa diferencia supera un cierto límite sitúa la salida del sensor a un “1” lógico durante un periodo de tiempo ajustable entre 2,5 segundos y 5 minutos según se configure. Sin embargo, una vez superado ese tiempo la salida queda bloqueada a 0 voltios durante 2,5 segundos antes de poder tomar una nueva muestra. Este bloqueo hace que la salida del sensor no se pueda conectar directamente a la entrada del conversor ADC del *Photon* porque se interpretaría mal su significado.

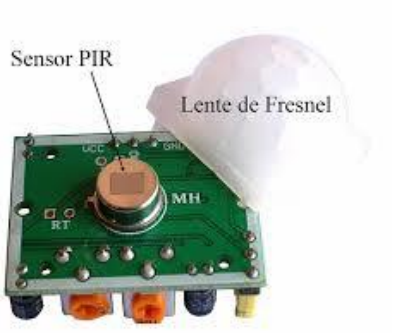



**FIG 12.** *Sensor HC-SR501*

El fallo radica en que después de detectar a una persona el bloqueo de la salida haría que el *Photon* leyese un “0” y entendería que esa persona se ha marchado a otra habitación. En cambio, en el siguiente ciclo detectaría un “1”, pues la persona sigue ahí, y según el algoritmo de estados debería notificar al servidor, una notificación que se repetiría en cada ciclo de detección. Si se ajusta la salida a su tiempo mínimo se reduce el retardo en la medida, pero la autonomía del dispositivo se ve comprometida. Sin embargo, con el tiempo de exposición en su máxima duración la autonomía mejora respecto el caso anterior, pero se aumenta el retardo en la medida.

Por ese motivo, en el apartado 3.3 se definirá un circuito auxiliar que permite corregir esta deficiencia del sensor.

## 3.3 CIRCUITO AUXILIAR

Como se explica en el apartado anterior, debido al diseño del que dispone el receptor PIR se precisa de un circuito electrónico auxiliar que permita mantener un “1” lógico durante todo el tiempo que una persona permanezca en una habitación con tal de reducir el número de transmisiones al servidor. Así se podrá aprovechar lo mejor de cada caso, precisión en la medida y aumento de la autonomía del sensor.

El circuito auxiliar se sitúa entre la salida del sensor PIR ($V_{in}$) y la entrada al pin de lectura del *photon* ($V_o$). El elemento principal de este circuito es un condensador electrolítico ($C_1$) que almacena la energía en el semiciclo que el sensor está con su salida a “1” y cede su energía cuando el sensor se establece en “0”. La resistencia $R_1$ permite regular el tiempo de carga ($t_1$) mientras que la resistencia $R_2$ ajusta el tiempo de descarga del condensador ($t_2$). Para finalizar el diodo ($D_1$) bloquea la corriente del condensador al sensor PIR cuando este está en el estado de bloqueo a modo de protección. La condición de diseño establece que la tensión de salida ($V_o$) se debe mantener a “1” como mínimo el tiempo de bloqueo del sensor PIR.

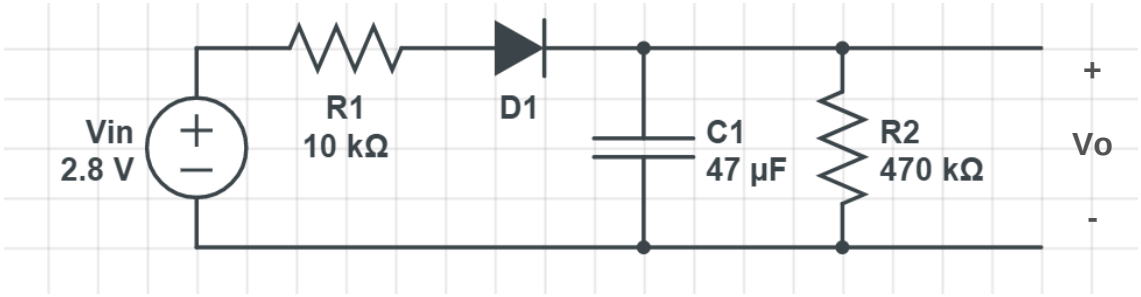



**FIG 13.** *Circuito electrónico auxiliar*

Según las especificaciones del fabricante del *photon* la tensión de entrada ha de ser superior a 1,6 V para considerar un "1" lógico y por debajo de 1,4 V es un "0" lógico. Entre 1,4 y 1,6 V es una zona de transición y mantiene la última salida conocida. La tensión de salida ($V_o$) respecto del tiempo se puede describir con las siguientes ecuaciones:

- Semiciclo de carga $\quad V_o = (V_{in} - V_{R1} - V_{D1}) \cdot (1 - e^{-\frac{t}{R_1 \cdot C_1}})$
- Semiciclo de descarga $\quad V_o = (V_{in} - V_{R1} - V_{D1}) \cdot e^{-\frac{t}{R_2 \cdot C_1}}$

Para verificar que el circuito de la figura 13[15] cumple la condición de diseño basta con verificar que el tiempo de descarga del condensador es mayor al tiempo de bloqueo del sensor PIR (5 segundos) cuando la tensión de salida es de 1,4V.

En primer lugar, calculamos la caída de tensión en la resistencia $R_1$ ($V_{R1}$) en el semiciclo de carga y una duración de 2,5 segundos.

$$V_o = (V_{in} - V_{R1} - V_{D1}) \cdot (1 - e^{-\frac{t_1}{R_1 \cdot C_1}}) \leftrightarrow V_{R1} = V_{in} - V_{D1} - \frac{V_o}{1 - e^{-\frac{t_1}{R_1 \cdot C_1}}} \leftrightarrow$$

$$\leftrightarrow V_{R1} = 2.8\,V - 0.3\,V - \frac{2.3\,V}{1 - e^{-\frac{2.5\,s}{10\,k\Omega \cdot 47\,\mu F}}} \leftrightarrow V_{R1} = 165\,mV$$

A continuación, calculamos el tiempo que necesita el condensador para descargarse hasta alcanzar los 1,4 V (cambio de "1" a "0" lógico) a su salida.

$$V_o = (V_{in} - V_{R1} - V_{D1}) \cdot e^{-\frac{t_2}{R_2 \cdot C_1}} \leftrightarrow t_2 = -R_2 \cdot C_1 \cdot \ln\frac{V_o}{V_{in} - V_{R1} - V_{D1}} \leftrightarrow$$

$$\leftrightarrow t_2 = -470\,k\Omega \cdot 47\,\mu F \cdot \ln\frac{1.4\,V}{2.8\,V - 0.165\,V - 0.3\,V} \leftrightarrow \mathbf{t_2 = 8.35\,s}$$

Como se puede ver el tiempo de descarga es mayor que el de bloqueo, por tanto, este circuito es capaz de suplir los efectos que originarían falsos negativos en el sistema.

[15] Los valores nominales de los componentes se pueden consultar en el Anexo 2: CIRCUITO AUXILIAR

## 3.4 CONSUMO ENERGÉTICO

Con el objetivo de construir un dispositivo que se adapte a cualquier ambiente de una vivienda es necesaria la incorporación de una batería o fuente de alimentación que suministre la energía necesaria al sensor para medir y transmitir la información hasta el servidor.

A la hora de escoger la ubicación de los sensores puede surgir dos posibilidades. La primera, el sensor dispone de una toma de corriente accesible la cual suministrará toda la energía que el dispositivo demande de forma continua. Este es el escenario ideal pues el único inconveniente que puede ocasionar es un corte temporal de la red eléctrica que apagaría inmediatamente el sensor imposibilitando el envío de información, sin embargo, este corte también afectaría al servidor haciendo caer toda la red de sensores de la vivienda hasta subsanar el problema. Y, en segundo lugar, conectar una batería a cada sensor para así obtener un dispositivo autónomo capaz de colocarse en cualquier estante o mueble de la vivienda. Esta segunda opción requiere de la creación de una optimización del código con el objetivo de reducir el consumo eléctrico y así extender la vida útil de la batería.

En una placa de desarrollo como es *Photon*, el elemento que más consume es la interfaz de red inalámbrica[16] por tanto, el objetivo es el de mantener desconectada la radio el mayor tiempo posible. En segundo lugar, se puede apagar la unidad de procesado[17] pues no se necesita realizar mediciones del estado del sensor continuamente.

Para calcular el consumo medio del dispositivo es necesario hacer una media ponderada del consumo medio de cada estado. Si expresamos el consumo como una función según el número de paquetes transmitidos obtenemos la siguiente ecuación.

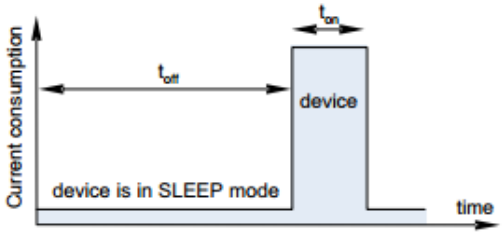



**FIG 14.** *Consumo medio respecto el tiempo*

$$I_A = \frac{I_{MAX} \cdot t_{ON} \cdot n + (t_S - t_{ON}) \cdot I_{IDLE}}{t_S} \leftrightarrow I_A = (I_{MAX} - I_{IDLE}) \cdot \frac{t_{ON} \cdot n}{t_S} + I_{IDLE}$$

Donde $I_{MAX}$ corresponde con el consumo medio cuando se transmiten datos al servidor, $I_{IDLE}$ es el consumo medio con la CPU y la radio desconectada, $t_{ON}$ es el tiempo que permanece encendida la radio, $t_S$ representa el tiempo total que se está midiendo y n es el número de paquetes que se han enviado durante el tiempo $t_S$. Donde n ha de cumplir que:

$$\forall n \in N \cup \{0\} \mid n \leq \frac{t_S}{t_{ON}}$$

[16] Consumo medio con la interfaz de red activada: 80 mA
[17] Consumo medio con la CPU desconectada: 80 µA

Como se explicaba anteriormente, el objetivo de estos dispositivos es el de acoplarles una batería que suministre energía para su funcionamiento. Para este proyecto se ha utilizado una batería LiPo de 2.000mAh [10] debido a su reducido tamaño e integración con el adaptador de batería [11] y *Photon*. La agrupación de estos elementos permite hacer un seguimiento del estado y la carga de la batería pudiendo avisar al usuario en caso de quedarse sin energía el dispositivo.

Se realizaron dos pruebas para comprobar el impacto de la gestión de los recursos que consumía el sensor. Las cuales consistía en registrar el tiempo de autonomía de la batería en un escenario real. En el primer caso no se aplicó ningún tipo de restricción en cuanto al uso de CPU y la interfaz de red permaneció encendida en todo momento. El dispositivo permaneció conectado 25 horas y 36 minutos. Por tanto, el consumo medio equivale a:

$$I_A = \frac{C_{BAT}}{t_S} \leftrightarrow I_A = \frac{2000\ \text{mAh}}{25{,}6\ \text{h}} \leftrightarrow \mathbf{I_A = 78{,}125\ mA}$$

Sin embargo, cuando se restringió el uso de la CPU y la interfaz de red únicamente de encendía cuando era estrictamente necesario se registró una duración de 20 días, 16 h y 24min.

$$I_A = \frac{C_{BAT}}{t_S} \leftrightarrow I_A = \frac{2000\ \text{mAh}}{496{,}4\ \text{h}} \leftrightarrow \mathbf{I_A = 4{,}029\ mA}$$

Esto se debe a que en el primer caso el ciclo de trabajo[18] correspondía a un 100% mientras que en el segundo se redujo a poco más de un 5%.

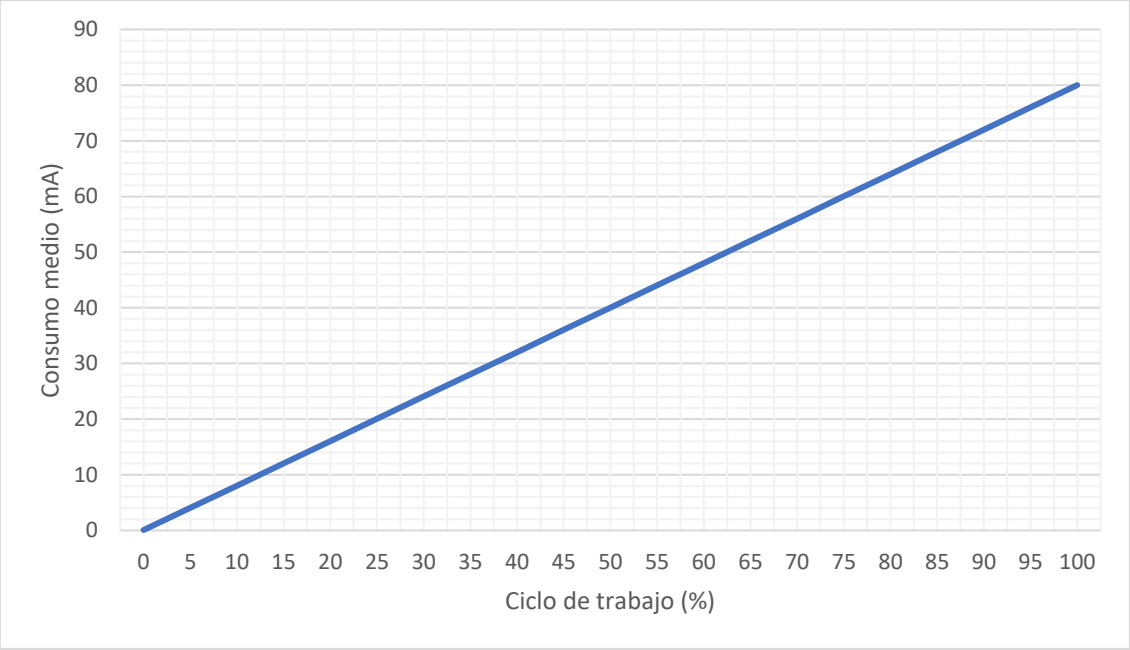



**FIG 15.** *Consumo medio según su ciclo de trabajo*

[18] Se define como la porción de tiempo que el sensor está activo sobre el tiempo total de funcionamiento.

# CAPÍTULO 4. SERVIDOR

A continuación, se explicará las características y funciones que se requieren en el servidor, el cual es el elemento central de la red.

## 4.1 ENTORNO DE EJECUCIÓN

La máquina que se ha escogido para alojar el código fuente del proyecto es una Raspberry Pi 3B [12] con un sistema operativo Raspbian[19]. La elección de este equipo respecto a otro viene dada por su reducido coste, disponer de serie dos interfaces de red (Ethernet y Wi-Fi) y un bajo consumo eléctrico pues la aplicación que ha de ejecutar no precisa de grandes prestaciones.

A nivel de aplicación se ha utilizado NodeJS [13] en su versión 10.16.0 LTS. Este entorno modular para JavaScript nos permite crear de forma fácil y rápida aplicaciones y entornos web. El propósito de utilizar esta tecnología es el de ofrecer una serie de rutas mediante HTTP (para las comunicaciones con el sensor) y HTTPS (para las comunicaciones con el usuario).

Mediante el gestor de paquetes NPM se han podido añadir librerías de código ya existentes como son:

- *Express*: Gestiona las rutas y funciones que se ejecutan en los puertos de escucha.
- *JWT-simple*: creación y validación de tokens los cuales se utilizan en la autenticación y autorización de los usuarios en el sistema.
- *MariaDB*: interfaz del controlador de la base de datos, debe tener permisos de lectura y escritura.
- *Moment*: permite manipular fechas en formato UNIX a modo de marcas temporales, permite saber si el token de un usuario ha expirado.
- *Network*: permite acceder a la configuración de la interfaz de red y aplicar máscaras para conocer la procedencia de los paquetes de información, se utiliza en la autenticación del sensor.
- *Nodemailer*: permite enviar mensajes automáticos a la bandeja de entrada de un usuario.
- *Socket.io*: establece un túnel TCP bidireccional entre el servidor y cliente para el intercambio de información, permite el cifrado de datos mediante certificados digitales.

Durante la inicialización del código, el controlador de la base de datos[20] realiza una conexión de prueba con la base de datos para comprobar si existe o se debe crear una nueva. A continuación, se establece la interfaz por la cual entrarán los paquetes IP procedente de la red de sensores y finalmente se abren los puertos 80 y 443 para iniciar los servidores HTTP y HTTPS.

[19] Distribución Linux basada en Debian y optimizada para esta placa de prototipado.
[20] Todo lo relacionado con la estructura de la base de datos se desarrollará en el apartado 4.2.

## 4.2 BASE DE DATOS

Una base de datos permite guardar una gran cantidad de información ordenada de tal forma que sea accesible y manipulable de forma eficiente. En nuestro caso, se ha optado por el uso de una base de datos relacional ya que al tener una estructura predefinida permite gestionar los datos de una forma más simple respecto otras implementaciones las cuales están basan en el modelo de documentos como es MongoDB.

Un buen controlador de código abierto es MariaDB, la cual pertenece a la familia de bases de datos relacionales (basados en tablas), ofrece compatibilidad y superioridad en cuanto a procesado de peticiones complejas sobre MySQL. En este proyecto se ha creado una base de datos que almacena cuatro tablas en ella.

La primera tabla se llama *users*. Su función es la de registrar y autorizar el acceso de los clientes finales al sistema a partir de un nombre de usuario único y su correspondiente contraseña. En esta primera versión, únicamente se registra al usuario administrador.

A continuación la tabla *sensors* se encarga de mantener el estado actual de la red de sensores. Permite identificar de forma inequivoca al sensor a partir de su ID, conocer el porcentaje de batería restante, asignar un tiempo máximo que una persona puede estar en esa habitación y a que hora se detecto la última actividad. Además, de forma opcional, se puede asignar un nombre para identificar al sensor desde la interfaz de usuario.

```
MariaDB [ElderPi]> show columns from users;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| username | varchar(64) | NO   | PRI | NULL    |       |
| password | varchar(64) | NO   |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
2 rows in set (0.02 sec)

MariaDB [ElderPi]> show columns from sensors;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| deviceName | varchar(24) | YES  |     | NULL    |       |
| deviceID   | varchar(24) | NO   | PRI | NULL    |       |
| battery    | int(11)     | NO   |     | NULL    |       |
| expiration | int(11)     | NO   |     | NULL    |       |
| timestamp  | datetime    | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)

MariaDB [ElderPi]> show columns from history;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| deviceName | varchar(24) | YES  |     | NULL    |       |
| deviceID   | varchar(24) | NO   |     | NULL    |       |
| duration   | int(11)     | NO   |     | NULL    |       |
| timestamp  | datetime    | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)

MariaDB [ElderPi]> show columns from notifications;
+-------+--------------+------+-----+---------+-------+
| Field | Type         | Null | Key | Default | Extra |
+-------+--------------+------+-----+---------+-------+
| email | varchar(128) | NO   | PRI | NULL    |       |
+-------+--------------+------+-----+---------+-------+
1 row in set (0.01 sec)
```

**FIG 16.** *Tablas de la base de datos*

En tercer lugar, la tabla *history* guarda el registro de todos los cambios de estado de la red de sensores para después mostrarlos al usuario junto al tiempo que de dedicó en cada estancia de la vivienda.

Finalmente, la tabla *notifications* guarda una lista de direcciones de correo electrónico donde se enviaran todos los mensajes del sistema a sus usuarios. Las notificaciones se detallarán en el apartado 4.4.

## 4.3 FUNCIONAMIENTO

El flujo de los datos y su procesado es un apartado fundamental para el desarrollo de este proyecto. Por ello se arrojará un poco más de luz en este apartado y se explicarán los detalles de los algoritmos que se ejecutan dentro del servidor.

En primer lugar, cuando el servidor recibe una transmisión entrante por el puerto 80 de la interfaz designada para la red de sensores se procede a autenticar el origen de esa información. Aplicando la máscara de subred sobre la dirección IP origen y conociendo la interfaz de entrada se puede averiguar su procedencia. En caso de tener un resultado negativo se desecha el paquete, en caso contrario, se envía a la siguiente función, comprobar cuál fue el último cambio notificado correctamente. Si el sensor que ha enviado la notificación coincide con la última transmisión se rechaza el nuevo paquete, en cambio, si esta nueva petición procede de un sensor distinto se procede a actualizar el estado del sistema. Se introduce una nueva línea en la tabla *sensors* con toda la información necesaria si no se encuentra el identificador del sensor en la base de datos, o bien, se actualizan los parámetros de batería y fecha de dicho sensor con la última información disponible. Con esta acción se pretende tener una fotografía global actualizada del estado de la red con una entrada única para cada sensor.

A continuación, se actualiza la última línea insertada de la tabla *history* añadiendo la marca temporal que resulta de restar el momento en que se detectó la entrada en la habitación junto al momento en que la abandonó. Finalmente, se inserta una nueva entrada en la misma tabla con la posición del usuario dejando el campo *duration* vacío a espera de una nueva actualización del estado de la red. Esta tabla permitirá graficar el comportamiento del usuario para en un futuro obtener patrones de comportamiento.

## 4.4 NOTIFICACIONES AL USUARIO

A veces el sistema debe comunicarse con el usuario, pero este puede que en ese momento no esté visualizando la interfaz web donde se muestran este tipo de alertas. Por ello, se ha diseñado un sistema de notificaciones por correo electrónico. En esta primera versión tan solo se envían los avisos del exceso de tiempo en una habitación de la vivienda o bien cuando un sensor detecta que el nivel de batería ha descendido de un nivel preestablecido y se debe proceder con una recarga.

Previamente el sistema debe tener guardada una lista de correos electrónicos a los cuales se les enviarán las notificaciones. Esta lista puede contener las direcciones de diversos familiares cercanos de la persona que se quiere monitorizar para que todos estén avisados. Esta lista se guarda en la tabla *notifications* y está marcada para que no haya repetición de correos electrónicos. Todos deben de ser únicos. Mediante la librería *nodemailer* se puede configurar una cuenta de correo electrónico para enviar mensajes de forma automática al ejecutar una función.

Las notificaciones por exceso de tiempo en una habitación son las más críticas y prioritarias pues son las encargadas de dar la voz de alarma cuando existen comportamientos anómalos en la rutina de la persona monitorizada. Su funcionamiento se basa en crear un temporizador a partir del tiempo que cada sensor tiene asignado en el campo *expiration* de la tabla *sensors*. Si en ese periodo de tiempo no se ha recibido una transmisión conforme el usuario se ha desplazado a otra estancia de la vivienda se procede a enviar un correo electrónico con el motivo de la alarma y el lugar del incidente (si el sensor tiene configurado un nombre aparecerá este en vez del identificador para hacer más fácil su localización). En caso de recibir una transmisión de otro sensor antes de finalizar el temporizador este se reiniciará con los parámetros de la nueva localización (tiempo y sensor).

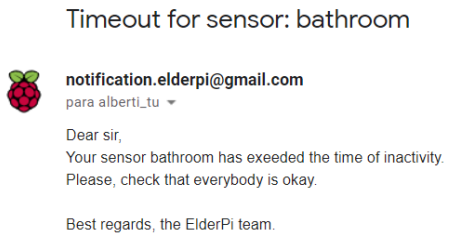



**FIG 17.** *Alerta por inactividad*

De esta forma cuando el usuario recibe este tipo de alerta puede ponerse en contacto con la persona dependiente o directamente ir en persona y verificar lo sucedido. Para un correcto funcionamiento del sistema, y evitar falsos positivos, es necesario configurar los tiempos de expiración acordes al tiempo que una persona puede estar en una habitación sin salir de ella. Por ejemplo, el sensor encargado del dormitorio no debería tener configurado menos de 8 o 9 horas pues es lo que debería permanecer una persona si no se despierta a mitad de la noche. Sin embargo, pasar más de 2 horas consecutivas en el baño podría considerarse un comportamiento anómalo y además es un espacio susceptible de caídas y resbalones.

La otra notificación para la cual está preparado el sistema es la de realizar un seguimiento del nivel de batería individualizado de cada sensor conectado a la red. Cuando el nivel de batería se sitúa por debajo del 5% el sensor activa la interfaz radio para enviar un aviso al servidor indicando su identificador, una vez completada la transmisión el sensor vuelve a desconectarse para ahorrar energía mientras el servidor envía un correo electrónico a todas las direcciones que se encuentran en la tabla *notifications*, de igual manera que ocurría con las notificaciones por la finalización del temporizador. Si consideramos el consumo medio que se calculó en el apartado 3.4 (4,029 mAh) el tiempo de autonomía que este sensor dispone antes de dejar de funcionar es:

$$t_S = \frac{C_{BAT}}{I_A} \cdot \rho_{BAT} \leftrightarrow t_S = \frac{2000\ \text{mAh}}{4{,}029\ \text{mA}} \cdot 0{,}05 \leftrightarrow \mathbf{t_S = 1\ día\ 19\ h\ 40\ min}$$

# CAPÍTULO 5. CLIENTE WEB

El último elemento que conforma este proyecto es la aplicación web donde se envían los datos que ha captado el servidor procedente de la red de sensores. Su cometido es administrar el flujo de datos entre servidor y cliente para así mostrarlos al usuario de una forma ordenada y clara.

Se ha optado por desarrollar una aplicación web basada en Angular 2+, un cliente liviano, no bloqueante y de código abierto que permite ser alojado dentro del servidor el cual se envía al usuario tras realizar una petición GET a través del navegador de Internet. Esta tecnología permite el uso de componentes HTML5 ajustables al tamaño de la pantalla y, por tanto, permite visualizar los datos de forma más amigable tanto en ordenadores de sobremesa como en dispositivos móviles. Gracias a la versatilidad de las aplicaciones web, el usuario únicamente debe tener un dispositivo con conexión a Internet y conocer la dirección IP pública a la cual está conectado su servidor para así tener acceso a los datos.

Su jerarquía interna está basada en la concatenación de componentes, los cuales están formados por un fichero HTML5 para visualizar el contenido, un fichero JavaScript (o TypeScript) que administra el comportamiento de la página y un fichero CSS para dar estilos y color al renderizado.

## 5.1 TRANSMISIÓN DE LA INFORMACIÓN

Angular presenta la información a partir del modelo-vista vista-modelo [14]. El cual permite establecer un canal bidireccional que permite al usuario modificar en tiempo real la información que se le presenta. Esto quiere decir que el modelo y la vista son dependientes entre sí, están sincronizados. Esta característica permite crear aplicaciones orientadas a eventos en tiempo real tal y como se explica a continuación.

### 5.1.1 ARQUITECTURA CLIENTE-SERVIDOR

La mayoría de los intercambios de información entre el cliente y servidor se realizan con la estructura básica del protocolo HTTP donde el cliente inicia la comunicación y el servidor responde con una única respuesta. Algunos ejemplos de uso son: el envío del cliente tras la primera conexión al servidor, la solicitud de claves tras el inicio de sesión, actualización de la configuración en los sensores…

Este método es eficiente debido a que no almacena la sesión en memoria una vez ha finalizado y será empleado en todas las comunicaciones que inicie el cliente. Sin embargo, si ocurre algún evento en el servidor no será posible notificarlo hasta que el cliente realice una nueva petición, de tal forma que no se puede conocer el

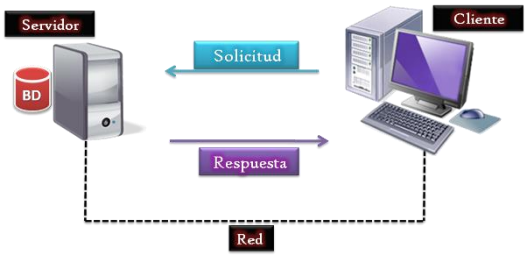



**FIG 18.** Estructura cliente-servidor

tiempo que transcurrirá entre la emisión del evento y el momento de la recepción por parte del usuario. Esta deficiencia obliga a la necesidad de crear un nuevo canal de comunicación que permita gestionar de forma eficiente la emisión y recepción de eventos.

### 5.1.2 COMUNICACIÓN BIDIRECCIONAL

Como se detalla en el apartado anterior, el protocolo HTTP no permite el intercambio de datos si no es el cliente quien realiza la solicitud, relegando al servidor a un mero gestor de peticiones. Para permitir al servidor la capacidad de iniciar él las comunicaciones se puede establecer lo que se conoce como *web socket*. Este tipo de comunicaciones está orientado a conexión y se basa en la idea de crear dos[21] sesiones TCP las cuales no se dan por finalizadas y pasan a memoria, de tal forma ambos extremos pueden introducir flujos de datos asíncronamente que serán recibidos en el otro extremo de la tubería. En el gestor de paquetes de NodeJS (NPM) se pude descargar la librería bajo el nombre de *Soket.io* [15].

Esta librería gestiona los *web sockets* a partir de la emisión y recepción de eventos, los cuales pueden ser acompañados por datos complementarios en el cuerpo del mensaje[22]. El uso de eventos permite no bloquear el código y permite realizar un buen escalado de los recursos al aumentar el número de usuarios conectados.

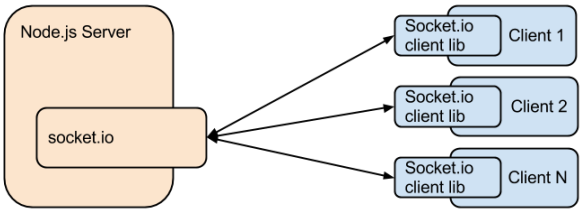



**FIG 19.** Estructura de web socket

La creación de este tipo de canales bidireccionales nos permite crear un medio de difusión y notificación de todos los cambios que ocurran en el servidor hacia los dispositivos conectados. De esta forma el usuario no ha de estar refrescando la aplicación, es ella la que se actualiza automáticamente.

Este sistema se utiliza para actualizar la posición del individuo monitorizado y publicar eventos relacionados con los sensores (expiración del temporizador por inactividad y batería baja en los sensores).

[21] Una sesión con origen en el cliente y con destino el servidor y otra sesión en sentido contrario.
[22] Soket.io admite cifrado a partir de certificados SSL (los mismos que se usan para HTTPS).

## 5.2 INTERFAZ DE USUARIO

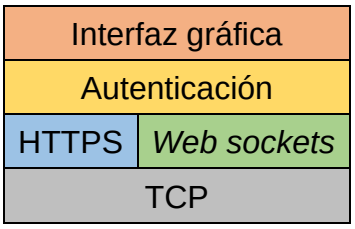



**FIG 20.** Estructura cliente web

La interfaz de usuario es el punto de encuentro entre el usuario y el sistema. Esta aplicación web está organizada en distintos niveles. En la base, una vez hemos recibido los segmentos de información a nivel de transporte (TCP), encontramos el gestor de comunicaciones. Encargado de administrar, en paralelo, las conexiones HTTPS y los canales *web socket*. A continuación, se sitúa el gestor de autenticación, una capa encargada de permitir, o denegar, el acceso a aquellos usuarios autorizados. Finalmente, en la cumbre de la aplicación, se sitúa la interfaz gráfica. Un conjunto de componentes que permite visualizar la información y recoger las interacciones con el usuario. A continuación, se detallarán cada uno de ellos.

### 5.2.1 INICIO DE SESIÓN

Cuando un usuario intenta conectarse al sistema y no dispone de una clave válida es redirigido a esta página. En ella el usuario ha de introducir el nombre de usuario y su contraseña. Esta información es enviada al servidor donde se comprueba con la tabla *users* de la base de datos.

Si el resultado es favorable, se devuelve la respuesta con una clave que permite identificar al usuario en todas las peticiones que realice durante toda la sesión. Una vez que el usuario ha guardado la clave en la memoria del navegador es redirigido a la página principal de la aplicación.

### 5.2.2 PÁGINA PRINCIPAL

Al iniciar esta vista, automáticamente se establece un canal a través de *web sockets* por donde el servidor enviará el estado de la red de sensores para mostrarla en formato de tabla. Si ocurriese algún cambio este se modificaría automáticamente sin la intervención del usuario.

En dicha tabla se muestran los sensores en orden descendente, siendo el primero de la tabla el último en notificar la detección del individuo. Cada fila contiene el identificador numérico único de cada sensor (o si se ha configurado, un nombre más amigable a dedición del usuario), el nivel de batería en el momento de la última transmisión realizada y la marca temporal cuando se envió la información.

### 5.2.3 CONFIGURACIÓN DE LOS SENSORES

Debido a que no se destina el mismo tiempo[23] a todos los espacios de la vivienda surge la necesidad de configurar cada sensor con un tiempo de expiración distinto además de configurar nombres más cotidianos a los sensores y así identificar con más facilidad el estado de la red.

En esta vista se puede seleccionar un sensor que esté conectado a la red mediante un desplegable y se rellena el formulario con los datos devueltos por el servidor. El usuario puede modificarlos (nombre y tiempo de expiración) y posteriormente guardarlos.

### 5.2.4 CONFIGURACIÓN DE LAS NOTIFICACIONES

Uno de los aspectos más importantes es la recepción de las notificaciones (procedentes de la expiración de algún temporizador de presencia como por batería baja). En esta vista el usuario dispone de una entrada de texto donde puede introducir todos los correos electrónicos donde desea recibir las notificaciones. Por ejemplo, se pueden introducir las direcciones de todas las personas responsables del cuidado del individuo monitorizado.

Una vez introducidas, aparecen automáticamente en forma de lista con la opción de poder eliminar las direcciones que ya no son necesarias.

### 5.2.5 HISTORIAL

Finalmente, en esta última vista se muestra el registro ordenado de todos los cambios de habitación del individuo. En ellos aparece la hora en que fue registrado, la habitación y la duración de su estancia. De esta forma se puede mostrar al usuario una tabla y un gráfico de barras con el tanto por ciento que dedica esa persona a cada habitación. Gracias a esta información se pueden crear patrones de conducta que ayudan a identificar las rutinas del individuo.

Estos patrones ayudan al gestor del sistema establecer límites para identificar anomalías o posibles problemas y de esta forma ofrecer mejores servicios.

[23] Un individuo de media no pasa el mismo tiempo en el recibidor que en el salón o el dormitorio.

# CAPÍTULO 6. ENTORNO DE PRODUCCIÓN

En el presente capítulo se detallará y evaluará *ElderPi* en un entorno real. El caso de estudio será en una vivienda unipersonal donde vive una mujer de edad avanzada. En primer lugar, hay que analizar la distribución de la vivienda para determinar el número de sensores que hacen falta para cubrir toda la vivienda.

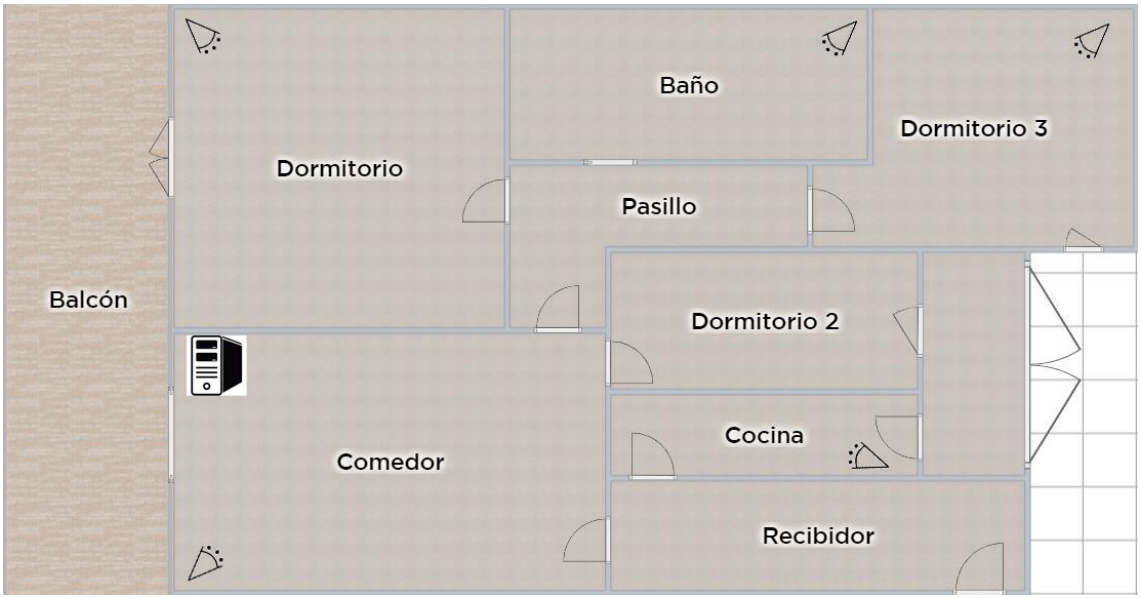



**FIG 21.** Distribución de la red de sensores en la vivienda de estudio

La distribución de los sensores tal y como se muestra en la figura 21 permite reducir a la mitad el número de sensores necesarios para monitorizar la vivienda respecto del número de espacios que hay, ya que, la distancia entre sensores es de dos estancias para cualquier desplazamiento que se quiera realizar. Por tanto, siempre que se salga de una estancia sin sensor se pasará a otra que sí dispondrá de uno, y viceversa. El criterio de no ubicar sensores en ciertos espacios es debido a que son espacios de tránsito y no se le suele dedicar mucho tiempo en ellos.

En segundo lugar, el servidor se colocará cerca del *router* para minimizar el despliegue de infraestructura Ethernet. Además, como este servidor actuará a la vez de punto de acceso para los sensores es recomendable situarlo en un punto equidistante de todos los nodos para mejorar la cobertura Wi-Fi de la red.

A continuación, se detallarán los pasos que hace falta seguir para instalar y poner en marcha la red de sensores.

## 6.1 INSTALACION

La instalación de estos dispositivos ha sido pensada para que sea un proceso fácil, rápido y simple. Los materiales necesarios son:

- 1 Raspberry Pi 3B
- Cargador micro USB 2.5A
- Cable Ethernet Cat 5e
- Tantos sensores como se necesiten para la vivienda

En primer lugar, conectamos la Raspberry Pi mediante un cable Ethernet al *router* y desde una terminal descargamos desde GitHub el código fuente del proyecto.

```
# Descarga del código y ejecución de la instalación
git clone github.com/alberti-tu/ElderPi
cd ElderPi
chmod +x setup.sh
sudo ./setup.sh
```

El fichero *setup.sh*[24] instalará automáticamente todos los programas necesarios para la ejecución del servidor. Durante la instalación pedirá al usuario la información necesaria para generar los certificados SSL y poder disponer de comunicaciones cifradas, después se reiniciará la máquina. A continuación, hay que acceder al gestor de red (desde la interfaz gráfica, esquina superior derecha) y configurar la interfaz de red inalámbrica *wlan0* en modo *hotspot*. Para la red hay que configurar un nombre SSID y una contraseña. Además de habilitar un servidor DHCP para direcciones IPv4.

A continuación, se ejecutará el servidor con el siguiente comando en la terminal.

```
# Entramos en la carpeta raíz del servidor y lo iniciamos
cd ElderPi/ElderPi-server
npm start
```

Si todo es correcto se deberá ver por pantalla que se ha creado correctamente la base de datos (si no existía previamente). Para verificar el cliente basta con acceder a la IP de la Raspberry a través de un navegador web. Si los certificados eran autofirmados saltará un error de privacidad, el cual se puede evitar añadiendo una excepción al navegador. Sin embargo, para evitar ataques maliciosos se deberá obtener unos certificados válidos en el futuro.

[24] Ver ANEXO 3: FICHERO DE INSTALACIÓN

Para finalizar la instalación hace falta vincular los sensores a la Raspberry. Para ello es necesario descargarse la aplicación móvil Particle[25] y seguir los pasos que allí se describen. Cuando se solicita conectarse a una red Wi-Fi, se debe vincular el sensor a la red que acabamos de crear. Por último, se debe cargar el código fuente en el sensor, el cual está disponible en el ANEXO 1: CÓDIGO DEL SENSOR.

Para verificar que el sensor se ha vinculado correctamente, al pasar una persona por delante de él se debe crear una nueva entrada en la tabla *sensors*, la cual es accesible desde la pantalla principal en el cliente web.

## 6.2 RENDIMIENTO Y PRUEBAS

El principal objetivo de estas pruebas es comprobar el correcto funcionamiento y rendimiento de la red de sensores. Para ello te utilizará la herramienta Wireshark, con la cual se puede analizar los diferentes intercambios de información en la red.

En esta prueba se medirán diferentes parámetros que permitan modelar el comportamiento de la red, como son el tiempo de detección del sensor, el ancho de banda necesario y el tiempo de servicio en el servidor.

### 6.2.1 TIEMPO DE REACCIÓN

Uno de los parámetros más importantes para el modelado de este sistema es la correlación que mantienen los datos en el momento de ser enviados respecto el momento en el que fueron generados. Esta demora viene principalmente ocasionada por la necesidad de ahorro de energía, el cual se consigue gracias a la desactivación de la CPU principal, sin embargo, una vez se detecta al individuo se debe reactivar ese microprocesador y realizar la transferencia de datos al servidor.

Las diferentes etapas que retrasan la emisión de la información son:

- Detección del individuo
- Reactivación del microprocesador e interfaz de red
- Acceso al canal Wi-Fi
- Configuración de los parámetros de red (ARP, DHCP…)
- Transmisión de la información (HTTP)
- Confirmación del servidor (ACK)

Según las diferentes muestras que se han obtenido, el tiempo medio fue de 15 segundos y una variancia de ±0,2 segundos (±1,33%) entre la detección del individuo hasta la confirmación del servidor. Este reducido tiempo de reacción permite ignorar su impacto en la configuración de los temporizadores (horas) para los sensores ya que se encuentra en varios ordenes de magnitud inferior.

[25] https://play.google.com/store/apps/details?id=io.particle.android.app&gl=ES

### 6.2.2 TIEMPO DE TRANSMISIÓN

Otro elemento permite caracterizar el intercambio de información entre sensor y servidor es la cantidad de información que se debe transmitir y el tiempo que se requiere para su envío. A partir de una captura de Wireshark realizada sobre la interfaz de red del servidor a la que está conectado el sensor se puede observar todos los protocolos de red que intervienen, sus tiempos de envío y la cantidad de información que contienen.

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 2 | 0.002179925 | b8:27:eb:13:2b:05 | e0:4f:43:f4:10:c9 | EAPOL | 113 | Key (Message 1 of 4) |
| 3 | 0.007213988 | e0:4f:43:f4:10:c9 | b8:27:eb:13:2b:05 | EAPOL | 139 | Key (Message 2 of 4) |
| 4 | 0.008534130 | b8:27:eb:13:2b:05 | e0:4f:43:f4:10:c9 | EAPOL | 141 | Key (Message 3 of 4) |
| 5 | 0.012944397 | e0:4f:43:f4:10:c9 | b8:27:eb:13:2b:05 | EAPOL | 113 | Key (Message 4 of 4) |
| 6 | 0.017596745 | b8:27:eb:13:2b:05 | e0:4f:43:f4:10:c9 | EAPOL | 153 | Key (Group Message 1 of 2) |
| 7 | 0.023163145 | e0:4f:43:f4:10:c9 | b8:27:eb:13:2b:05 | EAPOL | 113 | Key (Group Message 2 of 2) |
| 8 | 0.082988150 | e0:4f:43:f4:10:c9 | ff:ff:ff:ff:ff:ff | ARP | 42 | Gratuitous ARP for 0.0.0.0 (Request) |
| 9 | 0.085980463 | 0.0.0.0 | 255.255.255.255 | DHCP | 350 | DHCP Discover - Transaction ID 0x2a |
| 10 | 0.087010869 | 10.42.0.1 | 10.42.0.170 | DHCP | 342 | DHCP Offer - Transaction ID 0x2a |
| 11 | 0.096308637 | 0.0.0.0 | 255.255.255.255 | DHCP | 350 | DHCP Request - Transaction ID 0x2a |
| 12 | 0.104250534 | 10.42.0.1 | 10.42.0.170 | DHCP | 342 | DHCP ACK - Transaction ID 0x2a |
| 13 | 0.113822154 | e0:4f:43:f4:10:c9 | ff:ff:ff:ff:ff:ff | ARP | 42 | Who has 10.42.0.1? Tell 10.42.0.170 |
| 14 | 0.113868247 | b8:27:eb:13:2b:05 | e0:4f:43:f4:10:c9 | ARP | 42 | 10.42.0.1 is at b8:27:eb:13:2b:05 |
| 15 | 0.119319128 | 10.42.0.170 | 10.42.0.1 | TCP | 58 | 61640 → 80 [SYN] Seq=0 Win=4608 Len=0 MSS=1152 |
| 16 | 0.119483397 | 10.42.0.1 | 10.42.0.170 | TCP | 58 | 80 → 61640 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 |
| 17 | 0.120728124 | 10.42.0.170 | 10.42.0.1 | TCP | 54 | 61640 → 80 [ACK] Seq=1 Ack=1 Win=4608 Len=0 |
| 18 | 0.120988746 | 10.42.0.170 | 10.42.0.1 | TCP | 58 | 61640 → 80 [PSH, ACK] Seq=1 Ack=1 Win=4608 Len=4 [TCP segment of a reassembled PDU] |
| 19 | 0.121048121 | 10.42.0.1 | 10.42.0.170 | TCP | 54 | 80 → 61640 [ACK] Seq=1 Ack=5 Win=29200 Len=0 |
| 20 | 0.122659302 | 10.42.0.170 | 10.42.0.1 | HTTP | 208 | POST /sensor HTTP/1.0 (application/json)Continuation |
| 21 | 0.122744666 | 10.42.0.1 | 10.42.0.170 | TCP | 54 | 80 → 61640 [ACK] Seq=1 Ack=159 Win=30016 Len=0 |
| 22 | 0.137672167 | 10.42.0.1 | 10.42.0.170 | HTTP | 184 | HTTP/1.1 200 OK |
| 23 | 0.143941946 | 10.42.0.1 | 10.42.0.170 | TCP | 54 | 80 → 61640 [FIN, ACK] Seq=131 Ack=159 Win=30016 Len=0 |
| 24 | 0.145696667 | 10.42.0.170 | 10.42.0.1 | TCP | 54 | 61640 → 80 [ACK] Seq=159 Ack=132 Win=4477 Len=0 |

**FIG 22.** Tráfico de red entre un sensor y el servidor

Como se puede observar, después de iniciar la interfaz de red del sensor, este solicita al servidor el intercambio de claves para el acceso a la red Wi-Fi mediante el protocolo WPA-2 (paquetes de 2 a 7), a continuación, se negocian las direcciones IP del sensor mediante DHCP (paquetes de 9 a 12) y la incorporación de las direcciones MAC a sus tablas ARP (paquetes 13 y 14). Finalmente, se realiza la transmisión de datos a nivel HTTP (sobre TCP) el cual se encuentra en el paquete número 20 bajo el método *POST* a la URL */sensor*. Tras un tiempo se recibe una respuesta afirmativa en el paquete 22 procedente del sensor.

Tras este análisis, únicamente se intercambian 392 Bytes de información útil a nivel de aplicación sobre los 3.226 Bytes totales (eficiencia del 12%). La comunicación tiene una duración de 527 ms donde 15 ms corresponden al tiempo de servicio[26] del servidor. Con estos datos la velocidad de transmisión es:

$$v_{TX} = \frac{s}{t_{TX}} \leftrightarrow v_{TX} = \frac{3226\ \text{Bytes} \cdot 8\ \frac{\text{bits}}{\text{Bytes}}}{527\ \text{ms}} \leftrightarrow \mathbf{v_{TX} = 48,97\ kbps}$$

[26] Diferencia entre la llegada de un paquete al servidor y el envío de su respuesta.

# CAPÍTULO 7. PROPUESTA DE MEJORAS

Una vez superado con éxito los objetivos establecidos en el primer despliegue de este sistema bajo un entorno real, cabe preguntarse. ¿Y ahora qué? Para contestar a esta pregunta es necesario avanzar hasta la segunda generación de esta red de sensores, *ElderPi 2.0*, y definir cuáles serán las líneas maestras que darán forma a ese nuevo proyecto. Algunas propuestas se detallan a continuación.

## 7.1 SISTEMA DE SEGURIDAD

En la primera versión de *ElderPi* no se dispone de un mecanismo capaz de desactivar los temporizadores que vigilan el estado del individuo, pudiendo generar falsos positivos en los siguientes casos:

1. El individuo recibe la visita de otras personas.
2. El individuo abandona la vivienda (sale a pasear, a hacer la compra, etc).

En el primer caso, interesa detener el temporizador de toda la red. Este sistema está dimensionado para detectar a una única persona en la vivienda, además, al detener el sistema porque se hayan más individuos estos pueden dar la voz de alarma en caso de ser necesario y reactivar el sistema una vez se marchen todos. Sin embargo, en el segundo caso, la vivienda queda vacía y si se excede el tiempo del último sensor que haya detectado la presencia del individuo se notificará al usuario por inactividad, siendo este un falso positivo. Para solventar esta situación, se debería notificar al sistema sobre la salida o entrada del individuo en la vivienda.

Es interesante la idea de configurar la red de sensores en un modo de alerta cuando el individuo abandona la vivienda. De esta forma, si alguien entra de forma sospechosa será detectado y notificado a los responsables del sistema. Para llevar a cabo esta mejora bastaría con modificar el código en el sensor e instalar unos módulos NFC que se comuniquen vía Wi-Fi con el servidor. El individuo al pasar con sus tarjetas NFC podría activar o desactivar el modo alerta.

## 7.2 MEJORAS EN LA RED DE SENSORES

Actualmente los sensores únicamente disponen de un módulo capaz de detectar la presencia de seres vivos mediante la luz infrarroja que estos cuerpos desprenden. Pero a la hora de realizar la comunicación con el servidor se podría aprovechar para enviar información del ambiente como la temperatura, humedad, la cantidad de luz, etc.

De esta forma se podría detectar si en la habitación donde se encuentra el individuo cumple con los parámetros recomendados.

## 7.3 CENTRALIZACIÓN DEL SERVICIO

Sin duda mantener el servicio distribuido donde cada usuario es responsable del mantenimiento de su red de forma autónoma conlleva una serie de inconvenientes. Una mala gestión del software o de las medidas de seguridad puede dejar la puerta abierta a posibles atacantes y tener acceso a datos sensibles del usuario.

Por ese motivo, en la segunda versión de ElderPi se debe intentar migrar el procesado de datos a unos servidores que certifiquen el buen uso de estos y donde profesionales del sector garanticen a aplicación de las medidas de seguridad que consideren oportunas. Esta migración permitirá reducir la dificultad de la instalación pues ya no será necesario tener acceso a la configuración y redireccionamiento de los puertos en el *router* o el mantenimiento del servidor.

Por último, los sensores deberán cambiar el protocolo de transferencia de datos pues ahora, esos paquetes deberán salir de la red LAN y son susceptibles de sufrir ataques. Debido a esta posible injerencia se deberá asegurar una comunicación segura extremo a extremo mediante el uso de HTTPS en substitución del actual HTTP.

## 7.4 APLICACIÓN DE INTELIGENCIA ARTIFICIAL

En la era actual se generan cada día ingentes cantidades de datos, y tener la capacidad de procesarlos para extraer correlaciones entre ellos o patrones de comportamientos añaden un valor a esta información. Gracias al uso de técnicas de inteligencia artificial se pueden generar modelos o patrones de comportamientos capaces de predecir sucesos con gran exactitud y precisión.

Una vez recopilados una cierta cantidad de información procedente de los usuarios de este sistema se podría modelar su comportamiento con una función matemática que detecte conductas que se salgan de su rutina y dinámicamente pueda dar la alerta correspondiente tal y como sucede en la versión actual de ElderPi.

Para poder aplicar este concepto, en primer lugar, se debe definir las variables de entrada al sistema. Ya que los datos de entrada y de salida son todos binarios, los clasificadores basados en redes bayesianas[27] [16] son capaces de crear modelos con un alto porcentaje de aciertos pudiendo ser actualizados dinámicamente con nuevos datos sin tener que rehacer el modelo desde cero.

[27] Modelo basado en probabilidades combinadas.

# CONCLUSIONES

Dentro del análisis anteriormente expuesto, cabe recalcar la importancia de adecuar la tecnología a las necesidades de la sociedad para que, de esta forma, se pueda avanzar hacia un mundo más justo y equitativo. A su vez, se ha de reducir el impacto de barreras socioeconómicas al acceso de la tecnología, buena muestra de ello son todos aquellos proyectos que nacen bajo el concepto de código libre, los cuales son accesibles para todo aquel usuario que esté interesado. Por ese motivo, *ElderPi* basa toda su infraestructura y arquitectura bajo esta forma de crear tecnología.

Tras el estudio realizado sobre la evolución demográfica se puede vislumbrar un futuro donde las máquinas y las nuevas tecnologías jugarán un papel crucial para la vida diaria de las personas. El envejecimiento de la población es un hecho y se debe afrontar con una perspectiva clara del modelo de sociedad que habrá en el futuro.

Este proyecto pretende ser un sistema autónomo, en constate evolución, que permita velar por la seguridad y bienestar de la ciudadanía cuya situación impida una completa independencia de terceras personas. Situando al individuo en el centro del proyecto se consigue abordar la solución de una manera mucho más eficaz, ya que, *ElderPi* nace con un propósito claro y único. El cuál logra con un sistema simple y económico.

La mayor ventaja de este sistema es su versatilidad que permite adaptarse a cualquier vivienda sin importar su tamaño o distribución. Además, el poder utilizar diferentes sistemas de alimentación para los sensores o una combinación entre ellos ensancha el concepto de versatilidad permitiendo mejorar su tolerancia a fallos con el fin de obtener una gran disponibilidad del producto a lo largo del tiempo. Finalmente, y con la implementación de las distintas mejoras que se proponen para la primera revisión del proyecto, *ElderPi 2.0*, se podrá obtener una serie de mejoras que harán viable la implantación de este sistema en las viviendas de los usuarios a un nivel de producción.

# ANEXOS

## ANEXO 1: CÓDIGO DEL SENSOR

```
#include <HttpClient.h>
#include <SparkFunMAX17043.h>

SYSTEM_MODE(MANUAL);

HttpClient http;
http_header_t headers[] = {  { "Content-Type", "application/json" },  { NULL, NULL }   };

http_request_t request;
http_response_t response;

bool precense = false;  // Presence sensor status
bool alert = false;     // Low battery threshold

void setup() {
    Serial.begin(9600);

    request.ip = IPAddress(10,42,0,1);
    request.port = 80;

    response.body = "NULL";

    pinMode(D5, INPUT);     // Precense data

    lipo.begin();
    lipo.quickStart();
    lipo.setThreshold(5);
}

// Send data to the server
void postRequest(String path) {
    request.path = path;
    double battery = (double) lipo.getSOC();
    if(battery > 100) battery = 100;
    request.body = "{\"deviceID\":\"" + System.deviceID() + "\", \"battery\":" + battery + "}";

    http.post(request, response, headers);
}

// Check the sensor and update the state
bool readStatus(){
    bool changeStatus = false;
    bool newStatus = false;

    newStatus = digitalRead(D5);

    //true if presence goes from 0 to 1
    if((precense == 0) && (newStatus == 1))  changeStatus = true;

    precense = newStatus;    // Save the presence sensor status
    return changeStatus;
}

// Check the battery level threshold and update the state
bool readAlert(){
    bool changeStatus = false;
    bool newStatus = false;

    newStatus = lipo.getAlert();

    // true if battery level threshold goes from 0 to 1
    if((alert == 0) && (newStatus == 1))  changeStatus = true;

    alert = newStatus;    // Save the battery status
    return changeStatus;
}
```

```
// Main program
void loop() {
    System.sleep(D4, RISING, 10);   // D4 Wake up pin

    int status = ((int) readStatus() + (int) readAlert() * 2);

    // Send data
    if(status > 0) {
        WiFi.on();
        WiFi.connect();

        // Connection stablished
        if(WiFi.ready()) {
            switch (status) {
                case 1:
                    postRequest("/sensor");
                break;
                case 2:
                    postRequest("/sensor/alert");
                break;
                case 3:
                    postRequest("/sensor");
                    postRequest("/sensor/alert");
                break;
            }
        }

        WiFi.off();
    }
}
```

## ANEXO 2: CIRCUITO AUXILIAR

| COMPONENTE | VALOR NOMINAL |
|---|---|
| Sensor PIR | $V_{in}$ = 3.3V  $V_o$ = 2.8V |
| C1 - Condensador | C = 47µF |
| D1 - Diodo | $V_D$ = 0.3V |
| R1 – Resistencia | 10kΩ |
| R2 - Resistencia | 470kΩ |

## ANEXO 3: FICHERO DE INSTALACIÓN

```
#!/bin/bash

if [ "$(id -u)" -ne 0 ] ; then echo "Please run as root" ; exit 1 ; fi


echo "Updating packages..."

apt update -y

apt upgrade -y


echo "Installing NodeJS and NPM..."

curl -sL https://deb.nodesource.com/setup_10.x | sudo -E bash -

apt install nodejs -y
```

```
echo "Updating NPM and Angular to the latest version..."

npm install -g npm

npm install -g @angular/cli


echo "Installing programs..."

apt install mariadb-server screen -y

mysql --user="root" --database="mysql" --execute="update user set plugin='' where User='root';
flush privileges;"


echo "Changing the Network Manager"

apt install network-manager network-manager-gnome openvpn \openvpn-systemd-resolved network-
manager-openvpn \network-manager-openvpn-gnome -y

apt purge openresolv dhcpcd5 -y

ln -sf /lib/systemd/resolv.conf /etc/resolv.conf


echo "Cleaning packages..."

apt clean -y

apt autoremove -y


cd ElderPi-client || exit

npm install

npm run-script build

cd ..


cd ElderPi-server || exit

npm install

mkdir certificate

cd ..


openssl req -nodes -new -x509 -keyout ElderPi-server/certificate/server.key -out ElderPi-
server/certificate/server.cert -days 365


echo "Process finished!"

echo "Now reboot..."


reboot
```

# BIBLIOGRAFÍA

[1] Fernández, J, Parapar, C, Ruiz, M. “El envejecimiento de la población”, *CSIC*. Disponible en:
http://www.fgcsic.es/lychnos/es_es/articulos/envejecimiento_poblacion

[2] “Encuesta continua de hogares. Año 2018”, *INE* (2019). Disponible en:
http://www.ine.es/prensa/ech_2018.pdf

[3] “Tasa de fertilidad total”. *Banco Mundial* (2018). Disponible en:
https://datos.bancomundial.org/indicador/sp.dyn.tfrt.in?end=2017&start=1960&view=chart

[4] Iacono, L, Godoy, P, Marianetti, O, García, C, Párraga, C. “Estudio de la Integración entre WSN y redes TCP/IP” (2012), *Universidad de Montevideo*. Disponible en:
http://www.um.edu.uy/docs/5_estudio_de_la_integracion_entre_WSN_redes%20TCP_IP.pdf

[5] “IEEE 802.11”. Disponible en:
https://mentor.ieee.org/802.11/documents

[6] “IEEE 802.15”. Disponible en:
https://mentor.ieee.org/802.15/documents

[7] Velasco, Rubén. “Los mejores DDNS gratis alternativos a No-IP” (2018), *RedesZone*. Disponible en:
https://www.redeszone.net/2018/11/04/mejores-ddns-gratis/

[8] “Photon Datasheet” (2019), *Particle*. Disponible en:
https://docs.particle.io/datasheets/wi-fi/photon-datasheet/

[9] “Sensor infrarrojo de movimiento PIR HC-SR501” (2017), *Punto Flotante S.A.* Disponible en:
https://puntoflotante.net/MANUAL-DEL-USUARIO-SENSOR-DE-MOVIMIENTO-PIR-HC-SR501.pdf

[10] Wang. “585460 LI-POLYMER BATTERY PACKS” (2006), *Spark Fun*. Disponible en:
https://www.sparkfun.com/datasheets/Batteries/UnionBattery-2000mAh.pdf

[11] Jimblom. “Photon Battery Shield Hookup Guide”. *Spark Fun*. Disponible en:
https://learn.sparkfun.com/tutorials/photon-battery-shield-hookup-guide

[12] “Raspberry Pi 3B”. *Raspberry Pi Foundation*. Disponible en:
https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

[13] “Node.js v10.16.0 Documentation”. *NodeJS Foundation*. Disponible en:
https://nodejs.org/dist/latest-v10.x/docs/api/

[14] Blanco, Nacho. "*¿Qué patrón usa Angular? MVC o MVVM*" (2018). Open Webbinars. Disponible en:
https://openwebinars.net/blog/que-patron-usa-angular-mvc-o-mvvm/

[15] "*Soket.io 2.0 is here*". Open Collective. Disponible en:
https://socket.io/

[16] Golightly, Andrew. "*Class NaiveBayesMultinomialUpdateable*". Weka. Disponible en:
http://weka.sourceforge.net/doc.dev/weka/classifiers/bayes/NaiveBayesMultinomialUpdateable.html