Universidad de Oviedo


# ESCUELA POLITÉCNICA DE INGENIERÍA DE GIJÓN.

## GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

### ÁREA INGENIERÍA DE SISTEMAS Y AUTOMÁTICA

**TRABAJO FIN DE GRADO N.º 18010302**

# EVOLUCIÓN DE SISTEMA DOMÓTICO BASADO EN RASPBERRY PI Y Z-WAVE

**D. Jorge Nieto Palacio**
**TUTOR: D. Antonio Robles Álvarez**

**FECHA: JULIO 2018**

# Índice

# 1. Introducción.

Desde la época de la Revolución Industrial, uno de los principales objetivos de la industria ha sido el de implantar soluciones capaces de automatizar diferentes tipos de procesos, buscando como instancia última abaratar los costes y mejorar la calidad de trabajo de los empleados. Hoy en día el avance de la automatización se hace evidente y la búsqueda de nuevas y más avanzadas soluciones es un hecho [1].

Aun así, hasta finales de los años setenta no se empezó a hacer un hueco a la vivienda como un posible espacio de aplicación de las nuevas técnicas de automatización desarrolladas en la industria. Recibiendo esta aplicación de la automatización en la vivienda el nombre de domótica [2],[3].

Las primeras viviendas que aparecieron estaban basadas en el aún exitoso protocolo de comunicación X-10. Durante los siguientes dos años la comunidad mostró un creciente interés por la búsqueda de una casa automatizada y comenzaron los primeros ensayos avanzados sobre electrodomésticos y dispositivos automáticos para el hogar. Los primeros sistemas comerciales sencillos fueron instalados, aunque todo ello estaba más orientado a edificios de oficinas [2].

Desde entonces, su evolución no se ha detenido, buscando unos objetivos no del todo iguales a los existentes en la industria por lo que las soluciones para ciertos problemas difieren. Gracias a esta evolución su abaratamiento se hace realidad y proyectos que hasta hace poco tiempo presentaban un elevado coste se convierten en opciones reales con un reducido presupuesto. Una importante consecuencia de este abaratamiento es que permite a nuevos usuarios no sólo el consumo, sino el desarrollo de esta tecnología. Consiguiendo así un incremento de proyectos hardware y software totalmente libres para la comunidad de este ámbito.

Un importante protagonista de este mundo es Linux, que engloba varias distribuciones libres como Raspbian, que ganan importancia gracias a la proliferación de

dispositivos de bajo coste como es la Raspberry Pi, cada vez más popular tanto entre usuarios técnicos como un público más común. Raspberry cuenta con un respaldo de una comunidad de usuarios muy grande, que, a su vez, ayudan al desarrollo de nuevos tipos de proyectos. Paralelamente, durante estos últimos años ha venido tomando cada vez más peso en la sociedad el movimiento "maker" o "Do it Yourself" que promueve la idea de desarrollar tus propias tareas en lugar de contratar a un especialista, lo que ha apoyado y reforzado este tipo de nuevas tecnologías.

Todo esto ha desembocado en la aparición de alternativas domóticas libres y de un coste menor, que se presentan como opciones reales frente a las comerciales en un gran número de ocasiones, y que a su vez acerca a más gente este tipo de soluciones.

Lo que se pretende con este trabajo es precisamente aportar una solución para la automatización de una vivienda que se base en el uso de herramientas alternativas libres.

## 1.1.- ANTECEDENTES

Este proyecto se inicia con la intención de implementar un sistema domótico utilizando las herramientas disponibles en el mercado, con el añadido de crear un diseño y desarrollo propio de la instalación completa. Se parte de la disponibilidad de todos los productos hardware, dejando para el desarrollador la tarea de crear una interconectividad entre todos ellos, obteniendo un sistema final capaz de informar sobre diferentes parámetros e interactuar con el usuario. Se plantea la necesidad de informar al usuario del sistema ante inundaciones en su vivienda, ofreciéndole control sobre la llave de paso del agua de dicha vivienda como funciones principales.

Por otro lado, tal y como el nombre del documento indica, se trata de una evolución sobre un trabajo realizado con anterioridad. Este trabajo desarrollaba un sistema domótico también apoyándose en las mismas herramientas hardware y bajo el protocolo de comunicaciones Z-Wave, al igual que el presente proyecto. Asimismo, había llevado a cabo su propia interfaz web, tratando de informar e interactuar con los potenciales usuarios.

El presente trabajo se relaciona con su predecesor ya que utiliza las mismas herramientas hardware y el mismo protocolo de comunicaciones para llegar a un objetivo parcialmente común. Este es el de alcanzar un sistema domótico funcional con su propia interfaz. Sin embargo, difieren en el desarrollo del proceso para conseguirlo y en la expansión de la funcionalidad y mejoras que se han conseguido en este con respecto al anterior.

## 1.2.- OBJETIVOS Y ALCANCE

Como se ha mencionado, este trabajo pretende aprovechar los avances en el mundo de la domótica y mediante alternativas económicamente más viables automatizar un ámbito concreto de una vivienda cualquiera.

Por un lado, se ha profundizado en las diferentes opciones domóticas disponibles en el mercado, haciendo hincapié en aquellas soluciones libres, ya que presentan una gran ventaja económica y son capaces de ser modificadas para así adaptarlas a las necesidades concretas de cada individuo. También se han investigado las opciones hardware en lo que a los sensores y actuadores del sistema respecta, para poder llevar a cabo los objetivos del proyecto.

Estos objetivos del proyecto podrían desglosarse de la siguiente manera:

- Se quiere desarrollar un sistema de control capaz de informar y llevar a cabo diferentes tareas usando como controlador la Raspberry Pi anteriormente mencionada. Esta, permite el manejo de los sensores inalámbricos y actuadores gracias a la compatibilidad de estos con el protocolo de comunicaciones que se pretende usar, llamado Z-Wave. El sistema estará basado en este protocolo.

- Por consiguiente, se pretende que el sistema sea inalámbrico, para dar la mayor movilidad posible a los componentes que lo forman.

- Para ello, se deberán estudiar las maneras de comunicar a los nodos del sistema con el controlador. En este punto, aparecerá la tarjeta RaZberry, que es compatible con la Raspberry Pi y juntos permiten sostener las comunicaciones mediante el protocolo Z-Wave.

- Además, dado que la mayoría de los sistemas domóticos comerciales ofrecen una interfaz web desde ordenadores y dispositivos móviles donde controlar el sistema a nivel de usuario e informar del estado de la vivienda, se pretende desarrollar una totalmente independiente a la que la empresa Z-Wave ofrece. Esto conlleva la posibilidad de añadir nuevas características que otras interfaces no oferten.

- Una característica que se querrá implementar en el sistema es la posibilidad de servir imágenes de la vivienda al usuario. Para ello, se estudiarán las opciones que el mercado ofrece en cuanto a cámaras web compatibles.

- Por otro lado, se quiere alertar al usuario de manera externa a la interfaz web, de forma que no sea necesario el tener que acceder a esta para saber si ha pasado algo. Con notificaciones externas, se entiende lo que puede ser un correo electrónico o alguna APP que se pueda instalar en un smartphone.

Además, como nota final, cabe recordar que este trabajo está enfocado como una evolución sobre una versión anterior llevada a cabo. En este caso, además de realizar su completa ejecución, se pretende añadir mejoras y expandir su funcionalidad, perfeccionando el desarrollo y diseño de la interfaz, optimizando el sistema completo, ofreciendo imágenes tomadas por una cámara y generando notificaciones externas al usuario en caso de alarma por parte de algún sensor. Todas las mejoras serán detalladas en las conclusiones del documento.

## 1.3.- DESCRIPCIÓN DE LA MEMORIA

Esta memoria tiene el objetivo de dar una visión de las tareas que se han llevado a cabo en este Trabajo Fin de Grado. Por tal razón, está formado por las partes que giran en torno a las tareas de investigación realizadas, además del desarrollo real del sistema domótico. También incluye un estudio de la prueba completa de todo el sistema, garantizando su buen funcionamiento. Por último, se detalla un presupuesto en referencia a la ejecución e instalación del proyecto completo. De esta manera, se describirán a continuación los contenidos de la memoria.

En primer lugar, con el capítulo "Requisitos de usuario" se pretende dar una visión global de cómo queremos que la aplicación se comporte de cara al usuario para así entender mejor las soluciones tomadas. Siguientemente, en el capítulo "Herramientas de desarrollo disponible" se estudian los siguientes aspectos y se justifica su elección:

- Tipos de arquitecturas y topologías de los sistemas domóticos.
- Protocolos inalámbricos más comúnmente empleados en la actualidad.
- Todos los componentes hardware empleados en el proyecto.

A continuación, en el capítulo "Implementación del proyecto" nos centramos en el desarrollo de este, entrando en profundidad en los aspectos tenidos en cuenta desde un punto de vista más técnico y especificando su implementación. Se continúa con el capítulo "Prueba del sistema completo", dónde se pone en marcha y se verifica el correcto funcionamiento del sistema domótico en su totalidad, realizando diferentes test abarcando todas sus funcionalidades.

Por último, se aportarán detalles adicionales como una valoración económica en el capítulo "Presupuesto", que nos ayudará a terminar la memoria dando una serie de

conclusiones generales sobre todo el proyecto y explicando las mejoras conseguidas respecto a su predecesor, en el capítulo “Conclusiones”.

Se reservará un espacio final, en el capítulo “Referencias”, para adjuntar todas aquellas fuentes consultadas durante la realización y documentación del proyecto.

# 2. Requisitos de usuario.

Este capítulo guiará al lector a la hora de entender cómo ha de comportarse la aplicación creada y conocer las pautas que han llevado a las soluciones adoptadas. Se comentarán los requisitos del sistema impuestos por el usuario para que así, en capítulos posteriores, a partir del análisis se pueda llevar a cabo el diseño.

Una vez conocidos los objetivos y alcance del proyecto y entrando más en detalle en el comportamiento deseado de la interfaz, se puede decir que esta ha de, por un lado, servir imágenes del lugar de la vivienda donde se sitúe el controlador. El usuario podrá actualizar estas imágenes por medio de la interfaz cuando así lo desee, mediante un botón con tal propósito.

Por otro lado, ha de informar en tiempo real del estado del actuador y de tres parámetros que los sensores Z-Wave ofrecen. Estos tres parámetros son la temperatura ambiente, la detección o no de agua, y el estado de la batería que incorporan. Los parámetros se actualizarán sin retardo apreciable para que el usuario tenga conocimiento de lo que está pasando en su vivienda sin ningún tipo de demora. Además, en caso de detección de agua por parte del sensor, se enviará una notificación al correo del usuario con una imagen adjunta y un mensaje a través de la aplicación para smartphones Pushbullet. Al mismo tiempo, se activará el actuador capaz de cerrar o abrir la válvula de paso de agua de la vivienda, evitando inundaciones. Esta válvula podrá ser manejada también mediante la interfaz de usuario. Se dispondrá de dos botones, uno de cierre y otro de apertura.

El usuario podrá acceder a la interfaz a través de sus credenciales, es decir, a través de su usuario y contraseña previamente entregados. Una vez dentro, la navegación a través de las diferentes páginas será puramente intuitiva mediante una barra de navegación. Dispondrá, asimismo, en todas las páginas, de un botón para cerrar la sesión. Esta interfaz será accesible desde cualquier lugar del mundo con conectividad a internet.

Finalmente, hay que recordar que la comunicación de todo el sistema se ha establecido de manera inalámbrica y por tanto el usuario deberá ser consciente a la hora de situar los dispositivos. Problemas técnicos de este tipo de instalaciones, como son el alcance o interferencias, podrían aparecer.

# 3. Selección de tecnologías.

Este capítulo dará una visión general sobre el estado del arte para así conocer las diferentes opciones de tecnologías referentes a la automatización del hogar. Dado que el objetivo es conseguir un sistema de bajo coste basándose en elementos de hardware y software libre, parece necesario el estudio de diferentes soluciones que permitan la interconexión de los dispositivos que integran el conjunto.

## 3.1.- TIPOS DE ARQUITECTURAS

Normalmente, los sistemas de control se pueden agrupar en tres categorías: centralizados, descentralizados y distribuidos. La domótica, al ser una rama de la automatización y control, puede seguir esta misma clasificación.

### 3.1.1.- Sistemas centralizados

En los sistemas centralizados, existe un controlador central que es el que se encarga de enviar la información a los actuadores e interfaces como se puede ver en la figura 2.1. Si en este caso falta el controlador principal, el sistema deja de funcionar [4].

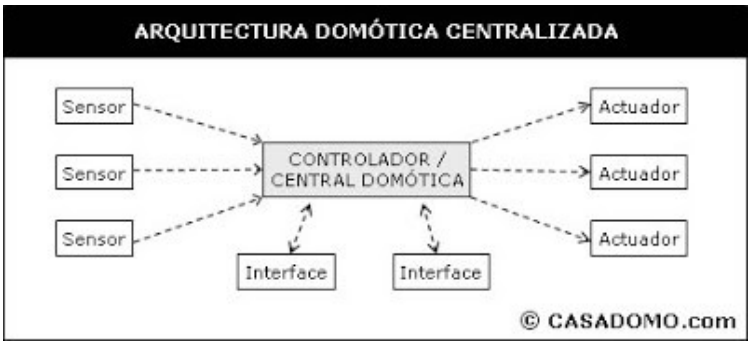


Figura 2.1.- Esquema de arquitectura domótica centralizada [5].

Su funcionamiento se ajusta al de un autómata clásico, siguiendo los típicos pasos de: lectura de entradas, ejecución del programa creado y escritura de salidas.

### 3.1.2.- Sistemas descentralizados

En ellos, existe más de un controlador y todos son interconectables mediante un BUS que se encarga de enviar toda la información entre ellos (figura 2.2.). Cada uno de los controladores se encarga de enviar información a los actuadores, dependiendo de lo que hayan registrado, tanto a través de los sensores como mediante los usuarios.

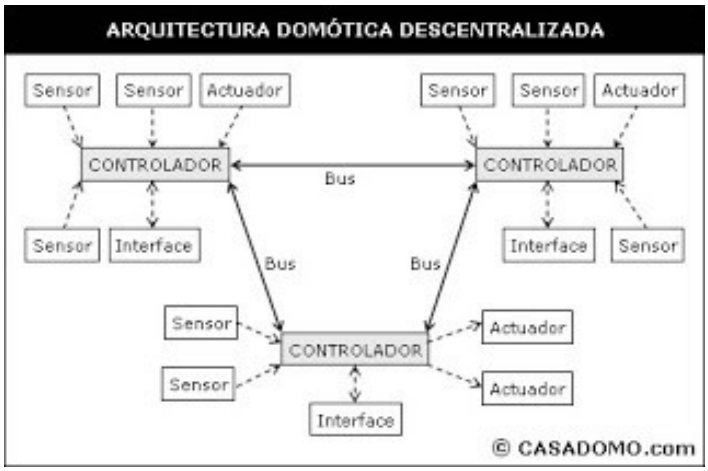


Figura 2.2.- Esquema de arquitectura domótica descentralizada [6].

### 3.1.3.- Sistemas distribuidos

En este tipo de sistemas, cada uno de los actuadores y sensores funciona como un controlador que tiene la capacidad de actuar y enviar datos al sistema según la información que recibe de otros dispositivos. Estos sistemas son los que más nos podemos encontrar en el mundo de la domótica y, es que, en este tipo de arquitecturas, eliminamos la necesidad de contar con uno o varios controladores como se refleja en la figura 2.3.

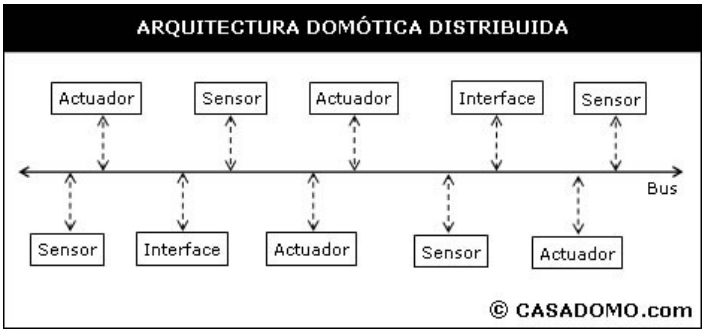


Figura 2.3.- Esquema de arquitectura domótica distribuida [7].

## 3.2.- TOPOLOGÍAS

Los sistemas distribuidos pueden, a su vez, clasificarse en diferentes topologías. Las más comunes se muestran en la figura 2.4. [8].

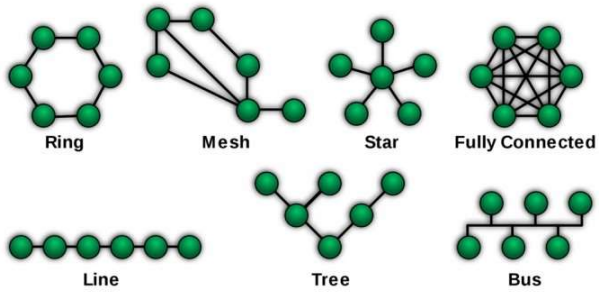


Figura 2.4.- Tipos de topologías para sistemas distribuidos [9].

- **Anillo**: En la que cada nodo tiene una conexión de entrada y otra de salida, que lo comunican con los nodos contiguos.

- **Malla**: La información puede tomar varios caminos, de modo que si un nodo falla se puede seguir intercambiando información.

- **Estrella**: Los nodos están conectados hacia uno central (Host). Topología de un sistema centralizado.

- **Árbol**: Los nodos están conectados entre sí de manera jerárquica.

- **Bus**: Los nodos están conectados a un canal central llamado bus.

## 3.3.- PROTOCOLOS DOMÓTICOS

Existen gran cantidad de protocolos domóticos con los que se puede llevar a cabo el presente proyecto, por lo que resultará interesante mencionar las diferentes opciones de las que se dispone. Este estudio, se centrará en aquellos protocolos concebidos para la comunicación inalámbrica, dado que su arquitectura es mucho más interesante en el ámbito de este proyecto, aunque hoy en día la mayor parte están concebidos para permitir la comunicación sin cables.

Cabe destacar que cualquier opción inalámbrica va a tener que lidiar con el problema de la autonomía de las baterías, por lo que su consumo es un aspecto crítico a la hora de la elección. Sin embargo, este tipo de soluciones son más flexibles gracias a la ausencia de cables y, por tanto, de una instalación.

### 3.3.1.- ZigBee

ZigBee es un estándar de comunicaciones inalámbricas diseñado por la ZigBee Alliance, estandarizado y de soluciones que pueden ser implementadas por cualquier fabricante. ZigBee está basado en el estándar IEEE 802.15.4 de redes inalámbricas de área personal (*Wireless Personal Area Network*, WPAN) y tiene como objetivo las aplicaciones que requieren comunicaciones seguras con baja tasa de envío de datos y maximización de la vida útil de las baterías.

ZigBee es promovida por la ZigBee Alliance, la cual es una comunidad internacional de más de 100 compañías como Motorola, Mitsubishi, Philips, Samsung, Honeywell, Siemens, entre otros, cuyo objetivo es habilitar redes inalámbricas con capacidades de control y monitoreo confiables, de bajo consumo energético y de bajo costo, que funcione

vía radio y de modo bidireccional. Todo ello basado en un estándar público global que permita a cualquier fabricante crear productos que sean compatibles entre ellos.

Desde sus anuncios, ZigBee ha gozado de gran expectativa, incluso corrían los rumores que se trataba del reemplazo de Bluetooth, y no es para menos pues, por ejemplo, el nodo ZigBee más completo requiere en teoría cerca del 10% del software de un nodo de Bluetooth o Wi-Fi típico. Esta cifra baja al 2% para los nodos más sencillos, pero el tamaño de código en sí es bastante mayor y se acerca al 50% del tamaño del de Bluetooth. No obstante, ZigBee no ha surgido para reemplazar a Bluetooth, pues sus campos de acción son distintos.

El alcance de las radios ZigBee tiene rangos de 10 a 75 metros y una velocidad de transmisión de 250 Kbps. En el entorno del hogar, las frecuencias menores a 2.4 GHz se propagan dos o más veces más lejos debido a la menor absorción de los materiales de construcción. Además, permite operar en redes de gran densidad. Esta característica, ayuda a aumentar la confiabilidad de la comunicación, ya que cuantos más nodos existan, mayor es el número de rutas alternativas para enviar los mensajes y garantizar que un paquete llegue a su destino [10].

### 3.3.2.- EnOcean

Se trata de una tecnología de comunicación inalámbrica de bajo consumo sin estándar internacional. Sin embargo, la alianza EnOcean mantiene el protocolo y garantiza su interoperabilidad. Trabaja en la frecuencia de 868 MHz, por lo que no existen problemas con la saturada banda de los 2.4 GHz. Su velocidad de transmisión es de 120 Kbps y tiene un rango de unos 300 metros en condiciones ideales y unos 30 metros en hogares [11].

Una ventaja muy importante de esta tecnología es la de permitir a los sensores trabajar sin tener activada la recepción por radio constantemente. Esto implica que pueden trabajar sin batería, obteniendo la energía del ambiente o de la interacción con el usuario, resultando en sensores con un mantenimiento mínimo y con un consumo inexistente [12].

Una desventaja digna de mención es que no se ofrece la comunicación bidireccional entre el controlador y sus módulos, por lo que no hay manera de saber si los comandos de ejecución han sido enviados a no ser que el usuario vea la consecuencia de manera directa (por ejemplo, el encendido o apagado de una luz). Por otro lado, en comparación con otros sistemas se podría decir que su coste es más elevado.

### 3.3.3.- Z-Wave

Z-Wave es una tecnología de comunicación inalámbrica sin estandarizar oficialmente, no obstante, existe la Alianza Z-Wave que garantiza la interoperabilidad de los dispositivos que se usan. El estándar es cerrado, por ello, para acceder a él es necesario ser miembro. Aun así, es fácil encontrar documentación sobre este protocolo de comunicaciones.

Entre todos los protocolos anteriores, Z-Wave ocupa el mejor puesto. La fuerza de Z-Wave radica en el gran número de productos de diferentes fabricantes, así como la comunicación inalámbrica robusta, bidireccional y segura. También se compone de chips de baja potencia y bajo coste, ya que fueron diseñados originalmente en 2003 para aplicaciones de viviendas residenciales.

Al igual que EnOcean, trabaja en la banda de los 868 MHz, evitando la saturada banda de los 2.4 GHz. Puede llegar a trabajar a 40 Kbps con unos rangos de alcance de hasta 30 metros en hogares.

Es una tecnología de red mallada donde cada controlador o módulo en la red es capaz de enviar y recibir comandos a través de paredes o pisos y utilizar nodos intermedios para esquivar obstáculos o espacios muertos. Esta libertad de conectividad facilita el comienzo con un paquete básico, pudiendo añadir con el tiempo componentes adicionales, personalizando los sistemas de energía y seguridad, únicos para una casa y al gusto de cada uno [12].

En la actualidad se ha evolucionado a Z-Wave+ que mejora ciertos aspectos con respecto al anterior, principalmente el consumo. Este protocolo inalámbrico es posiblemente el más conocido y empleado en la actualidad en aplicaciones de domótica, consiguiendo una buena relación características/precio claramente más competitivo que ZigBee o EnOcean.

### 3.3.4.- Protocolo escogido para el proyecto

Hasta este punto, las principales características de diferentes protocolos de comunicación inalámbrica han sido analizas. Se considera que no necesariamente la opción escogida es la mejor en todos los casos, simplemente, se ha dado prioridad a ciertos aspectos que se consideran primordiales en este proyecto en concreto.

En este caso, la elección de Z-Wave como protocolo de comunicaciones óptimo se ha llevado a cabo teniendo en cuenta diferentes características descritas a continuación.

A diferencia de ZigBee, Z-Wave, al igual que EnOcean, evita usar la saturada banda de los 2.4 GHz lo que evita interferencias que podrían entorpecer el rendimiento del sistema. Por otro lado, y dado que es posiblemente el más conocido, Z-Wave ofrece mucha más compatibilidad de productos que EnOcean, en parte gracias a la Alianza Z-Wave, la cual establece un control restricto y continuo de empresas colaboradoras.

Además, cabe resaltar la fuerte encriptación de Z-Wave, protegiendo el intercambio de información en la instalación domótica. Z-Wave ofrece igualmente una comunicación bidireccional, confirmando siempre los mensajes enviados por el remitente una vez que han llegado al destinatario, incrementando aún más la seguridad del sistema. Cabe destacar que, ZigBee comparte estas dos últimas características mencionadas.

Por último, se debe mencionar una última característica que vuelve a compartir con ZigBee. Esta, es la posibilidad de usar un nodo concreto como repetidor, haciendo que cuantos más nodos haya en el sistema, más fuerte se vuelve su comunicación, pues los mensajes tendrán varios caminos para escoger a la hora de transmitirse.

## 3.4.- RASPBERRY PI

El controlador del sistema permitirá que este opere correctamente, recibiendo información de los sensores, actuando consecuentemente, permitiendo la captura de imágenes y albergando la interfaz de usuario. Dado que se pretende crear un sistema domótico libre, no se entrará a valorar aquellos controladores comerciales puesto que no son modificables, su compatibilidad es reducida y su coste es mucho mayor que el de controladores libres.

Por otro lado, los dispositivos programables no están diseñados específicamente para funcionar como controladores, pero sus altas capacidades de ser programados permiten que se ajusten a diferentes tipos de proyectos, entre ellos el nuestro. Asimismo, gracias a estas características se pueden conseguir proyectos más flexibles, personalizables y baratos que con controladores comerciales.

Existe más de una solución, pero para este trabajo la Raspberry Pi se ajusta a las especificaciones concretas que se pretenden conseguir. No incluye un soporte nativo para el protocolo Z-Wave, pero gracias a la tarjeta RaZberry (de la que se hablará más adelante), compatible con la Raspberry y que la propia compañía Z-Wave vende, se pueden sostener las comunicaciones del controlador con sensores y actuadores. Raspberry y RaZberry juntas forman el controlador de nuestro sistema domótico Z-Wave.

La Raspberry Pi, es una placa computadora de bajo coste desarrollada por la fundación Raspberry. Presenta una gran aceptación mundial y se ha convertido en una de las herramientas principales para la gente que sigue la cultura DIY (*Do It Yourself*). Para este proyecto se usa el modelo Raspberry Pi 2.0 B mostrada en la figura 2.5.

Figura 2.5.- Placa Raspberry Pi [13].

Este modelo cuenta con un procesador Broadcom BCM2836 quad-core ARM Cortex-A7, que funcionan a 900 MHz. Tiene una memoria RAM de 1GB, 4 puertos USB y 40 pines GPIO. Además de un puerto HDMI y Ethernet, salida de audio analógica y digital y adaptadores para cámara (CSI) y display (DSI) también integrados [14].

El diseño no incluye disco duro y se entrega sin sistema operativo, pero cuenta con una ranura para tarjeta MicroSD sobre la que instalarlo. Existen diferentes sistemas como, por ejemplo, Raspbian, Pidora, Ubuntu o Windows, según las diferentes necesidades del desarrollador.

Su tamaño, potencia, versatilidad y su escaso consumo de 3W, la convierten en una excelente opción como controlador de nuestro sistema domótico.

## 3.5.- RAZBERRY

La placa RaZberry (figura 2.6.) se conecta a los pines 1-10 de la Raspberry y correctamente configurada pasa a crear un controlador Z-Wave funcional [15]. Esta placa se alimenta directamente de los pines de la Raspberry Pi y cuenta con una antena integrada, además de dos diodos LED para informar de la transmisión de datos. La instalación de esta placa en la Raspberry Pi se realiza con la finalidad de utilizar el software Z-Way, que

implementa los protocolos Z-Wave y permite crear aplicaciones propias mediante un API basada en peticiones HTTP.

Figura 2.6.- Raspberry Pi + RaZberry (rodeada de rojo) [16].

## 3.6.- SENSORES

Los sensores del sistema son los encargados de detectar cambios en el ambiente y generar la respuesta oportuna. Estos sensores transforman magnitudes físicas en eléctricas mandando una señal que el controlador pueda entender.

Para este proyecto en concreto, queremos un sensor capaz de detectar la temperatura y la presencia o no de agua en el ambiente. Una compañía que fabrica productos totalmente compatibles con Z-Wave y que oferta un sensor con estas dos funcionalidades unidas es Fibaro. Este sensor (figura 2.7.) es resistente al agua y detecta la misma a nivel del suelo cuando se encentra inundado, activando tanto una alarma luminosa como sonora.

Figura 2.7.- Fibaro Flood Sensor [17].

Además, informa sobre la temperatura ambiente y es totalmente inalámbrico, funcionando gracias a su batería, la cual tiene una vida estimada de dos años. Cabe resaltar que económicamente resulta más barato que la compra de dos sensores por separado. En este trabajo se han utilizado dos.

## 3.7.- ACTUADOR

El actuador del sistema se encargará de cerrar la llave de paso del agua de la vivienda cuando alguno de los sensores detecte agua o cuando el usuario así lo desee. Recibe directamente las órdenes del controlador, previamente programadas.

A la hora de seleccionar uno, la Auto Válvula GR-105 (figura 2.8) es la solución más proliferada en el mercado. Este actuador se alimenta directamente de la toma eléctrica de la vivienda gracias a un adaptador de 12V y 1A incluido con su compra. Puede estar abierta o cerrada, tardando de 5 a 10 segundos en cambiar de un estado a otro, y recibe y envía la información de manera inalámbrica [18].


Figura 2.8.- Auto Válvula GR-105 [19].

## 3.8.- CÁMARA

Dado que se ha escogido como controlador la Raspberry Pi, y la misma incluye un adaptador para cámara, una buena opción es usar la Pi Cámara. Se trata de una cámara

comercializada por la propia empresa Raspberry, mostrada en la figura 2.9. Aunque no da las mismas prestaciones que algunas cámaras que podemos conectar vía USB, al pertenecer a la misma empresa el controlador y la cámara, existen librerías que facilitan el control y supone que no sea necesario descargar nuevos programas para su instalación.

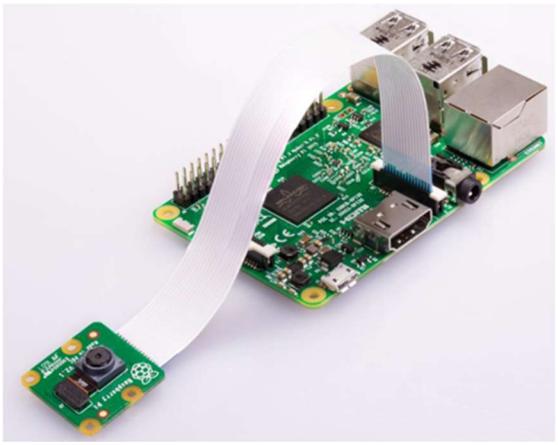

Figura 2.9.- Pi Cámara [20].

La cámara permite tanto grabar vídeo como tomar imágenes y funciona con todos los modelos de Raspberry Pi gracias al adaptador que estas disponen. Se trata de un modelo muy versátil y muy usado en el mundo Raspberry y aplicaciones de seguridad del hogar, por lo que dispone de numerosa información en la web sobre sus posibles usos. Existen librerías para su rápido y fácil manejo, convirtiéndola en una muy buena opción para el proyecto [21]. Cabe destacar que al ir unida al controlador habría que ubicar éste en la zona deseada de la vivienda. Para ello, se dispone de un adaptador Wi-Fi que facilita esta tarea. Este adaptador será descrito a continuación.

## 3.9.- ADAPTADOR WI-FI

Para que el controlador y, por consiguiente, el sistema funcione es necesaria una conexión a internet. La Raspberry Pi cuenta con adaptador para cables Ethernet, pero para darle mayor movilidad al sistema también cuenta con la posibilidad de añadir un adaptador Wi-Fi vía USB para tener conexión inalámbrica a la red.


Figura 2.10.- Pi Cámara [22].

Nos podemos encontrar con numerosas opciones, pero una buena es la que proporciona EDIMAX (figura 2.10.) ya que su adaptador no requiere de la instalación de drivers y es compatible con el sistema operativo Linux (usado en nuestro caso). Por lo tanto, se ha decidido usar el adaptador N-150 [23] que además está especialmente diseñado para la Raspberry Pi. Aunque para un rendimiento eficiente se recomienda usar la conexión vía ethernet, se ha dotado al sistema de esta opción en caso de necesidad.

## 3.10.- CONCLUSIONES DEL ESTUDIO

Como se ha visto, la domótica es la transposición de la automatización industrial al mundo doméstico. Empleándola, se pretende recoger la información de los dos sensores situados en la vivienda para conocer la temperatura y la ausencia o no de agua en la zona del sensor. El usuario tendrá la opción de actuar directamente a través de la interfaz sobre una válvula, cerrando o abriendo la llave de paso de agua de la vivienda. En caso de detección de agua por parte de los sensores, la válvula será cerrada automáticamente, informando al usuario mediante un mensaje a través de la APP Pushbullet y también con un e-mail que adjuntará una foto del momento en el que el sensor se ha activado.

En cuanto a la arquitectura y sabiendo por los requisitos de la aplicación que ésta va a ser inalámbrica, se tratará de un sistema centralizado, donde los sensores y actuadores se comunicarán con el nodo central (controlador), creando una topología en estrella.

# 4. Diseño e implementación.

En este capítulo se llevará a cabo la explicación de los pasos dados para llegar al objetivo final, que es el de obtener un sistema domótico Z-Wave funcional y con acceso a su propia interfaz web.

## 4.1.- MONTAJE

Antes de comenzar con la explicación, se considera oportuno para el mejor entendimiento del lector, mostrar el montaje final del propio controlador para que el sistema completo pueda llegar a funcionar.

El montaje del sistema (figura 3.1.) se realiza en varios pasos. Por un lado, la Raspberry ha de estar alimentada, para ello se necesita una fuente de alimentación micro USB. Es posible la utilización de un cargador de smartphone, asegurándose previamente que éste ofrezca una salida de 5 voltios y 1 amperio.

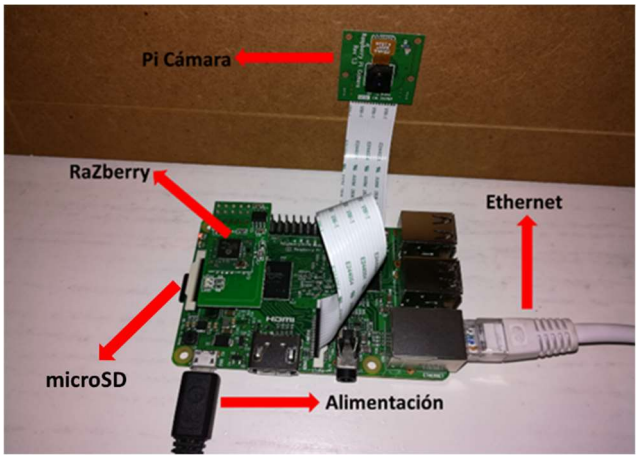


Figura 3.1.- Montaje completo del controlador.

La conexión a internet se puede realizar vía wifi o usando directamente un cable ethernet desde el rúter a la entrada habilitada para tal propósito. Se han configurado ambas maneras, pero para un mejor rendimiento del sistema se decide conectarlo vía ethernet.

Existe una ranura lateral donde insertar la tarjeta microSD, en este caso, de 16 GB. El conexionado de la cámara se realiza mediante un conector CSI (*Camera Serial Interface*).

Por último, para dejar el sistema funcional y operativo, ha de conectarse la tarjeta RaZberry que sostiene las comunicaciones con los sensores y actuador mediante el protocolo Z-Wave. Esta tarjeta se conecta en los pines GPIO 1-10 de la Raspberry tal como indica el fabricante. La correspondiente descarga del software de Z-Wave es necesaria para su funcionamiento.

## 4.2.- INSTALACIÓN DEL SISTEMA OPERATIVO

El primer paso en la instalación del sistema operativo es la elección de éste. Para la Raspberrry Pi, Raspbian, una distribución basada en Linux ha sido la escogida. Este es el sistema operativo oficial y recomendado por la Fundación Raspberry.

Para poder utilizar una de las versiones de Linux en la Raspberry se disponen de tres alternativas. Adquirir una tarjeta SD con la distribución deseada preinstalada, configurar una manualmente o utilizar la herramienta NOOBS (*New Out Of the Box Software*) que ofrece Raspberry.

NOOBS está especialmente pensada para aquellos usuarios con menos conocimientos, ya que provee una instalación guiada y sencilla de la distribución escogida. Entre ellas, ofrece Raspbian, Arch Linux o Pidora. El inconveniente es que se requiere de un monitor y teclado para realizar esta primera instalación, de lo cual no se disponía.

La sencilla solución a esto es grabar en la tarjeta de memoria la distribución deseada directamente desde un ordenador. Se deberá:

- Descargar la imagen Raspbian que ofrece Raspberry [24].

- Instalar un software para poder grabar la imagen de Raspbian. Se ha utilizado Etcher pero podría valer cualquier otro [25].

- Usando este programa, se selecciona el archivo .zip que contiene la imagen de Raspbian y se graba.

Un paso muy importante después de esto es que se ha de crear dentro de la tarjeta SD un archivo de texto vacío de texto llamado ssh.txt. Esto se debe a que, al manejar la Raspberry desde otro ordenador se necesita un protocolo de comunicación cliente-servidor que permita conectarse a ella de manera remota. Si se ha realizado lo anterior de manera correcta, y una vez insertada la tarjeta SD en la Raspberry ya conectada a internet, Raspbian encontrará el archivo ssh.txt, habilitará el protocolo SSH y borrará ese archivo.

## 4.3.- CONFIGURACIONES INICIALES

Para poder trabajar con la Raspberry Pi, al no disponer de teclado y monitor que conectarle, se ha decidido usar un entorno gráfico con el que acceder a la Raspberry desde otro ordenador. Con este fin se ha usado la aplicación VNC (figura 3.2.), pero primero ha necesitado de su instalación.

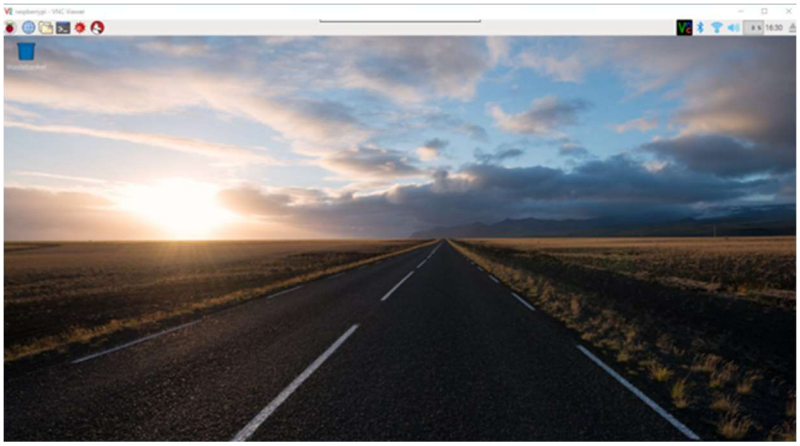

Figura 3.2.- Entorno gráfico VNC.

Putty es una consola de comandos que utiliza la conexión SSH [26]. Para establecer una conexión con la Raspberry mediante ésta, se necesita conocer su IP local, la cual es dinámica. Para ello, existe otro software llamado Advanced IP Scanner [27], que proporciona las direcciones IP de todos los elementos conectados a la red local. Sabiendo la IP que se le ha asignado en ese momento a la Raspberry y que el usuario y contraseña por defecto son "pi" y "raspberry" respectivamente, se ha podido iniciar sesión a través de Putty.

Para proceder a la instalación de VNC en la Raspberry simplemente se utilizan una serie de comandos insertados mediante la consola Putty. Por otro lado, ha de instalarse también en el ordenador desde el que se desea acceder a la Raspberry Pi la aplicación correspondiente (VNC Viewer). Para tal fin, en la página oficial de VNC, ofrecen su descarga gratuita.

Una vez que se ha conseguido acceder a la Raspberry mediante el entorno gráfico VNC, el siguiente paso es asignar a la misma de una IP estática con el objetivo de prescindir de usar Advanced IP Scanner cada vez que queramos conectarnos. Además, una IP estática será estrictamente requerida por las especificaciones del proyecto, ya que facilita las conexiones SSH y permite que el envío de información al servidor se realice siempre correctamente.

Para ello, es necesario editar el archivo dhcp.conf, configurando dos IP. Una para la conexión ethernet y otra para poder usar la conexión Wi-Fi mediante el adaptador. Además, habrá que editar el archivo WPA Supplicant, mediante el cual se especificarán los credenciales de conexión a la red local [28]. Una vez hecho esto, la Raspberry puede conectarse directamente vía SSH al entorno gráfico VNC Viewer directamente, sin requerir el uso de Putty y con una IP estática.

## 4.4.- CONFIGURACIÓN DE LA TARJETA RAZBERRY E INCLUSIÓN DE NODOS AL SISTEMA

Como antes se ha mencionado, Z-Way (software de Z-Wave) permite crear aplicaciones propias mediante una API basada en peticiones HTTP. Estas peticiones son atendidas en la Raspberry por el servidor web de Z-Wave. Para que las comunicaciones de los sensores/actuador con la Raspberry sean posibles es necesaria la tarjeta RaZberry.

La instalación se realiza de manera simple. Una vez conectada la RaZberry es necesario usar un comando que descargara todo el software Z-Way como se indica en [29]. Una vez hecho esto, el controlador Z-Wave basado en la Raspberry se encontrará operativo y el siguiente paso es el de incluir los nodos en la red Z-Wave para poder utilizarlos.

Z-Wave ofrece una aplicación web que actúa como interfaz de usuario para visualizar la red de nodos Z-Wave, permitiendo a los usuarios manejar sus equipos domóticos a través de la API sin necesidad de elaborar su propio software. Para incluir los nodos del sistema se hará uso de esta interfaz.

Para acceder a ella (figura 3.3.) es necesario introducir en un buscador la IP de la Raspberry junto al puerto 8083, que es el de Z-Wave (http://IPESTÁTICA:8083). Se pide crear un usuario y contraseña para poder acceder a la interfaz en futuras ocasiones. Una vez dentro, en el apartado de configuración, se puede añadir nuevos elementos de la siguiente manera: se selecciona la opción "*add new"*, se clica en "*add new Z-Wave Device*" y, por último, "*identify it automatically"*. Una vez realizado, se selecciona la opción "empezar la inclusión" y durante este tiempo el sensor/actuador se puede incluir. Para ello, en caso del sensor, hace falta clicar tres veces en un botón interior, al cual se accede abriendo su tapa. En caso de la válvula, el botón se visualiza desde el exterior.

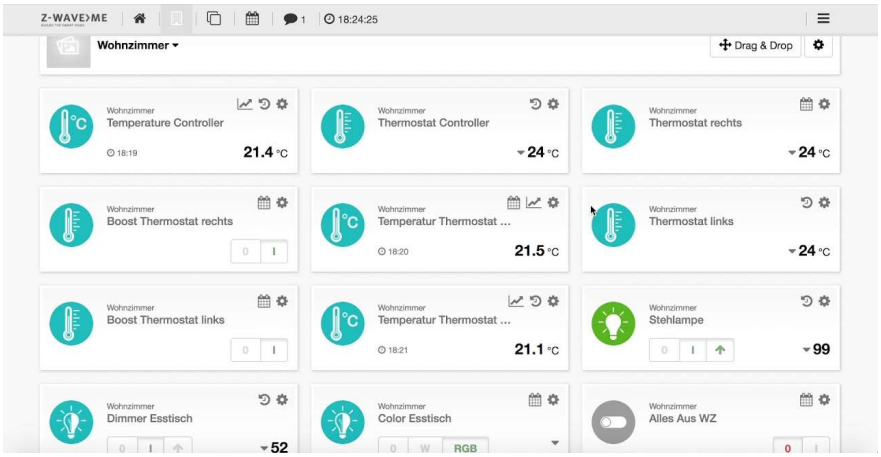


Figura 3.3.- Interfaz de usuario proporcionada por Z-Wave [30].

Cabe mencionar que, un problema encontrado a la hora de acceder a esta interfaz ha estado relacionado con la actualización por parte de Z-Wave del sistema. Debido a que en un principio existían problemas de compatibilidad entre la interfaz y el nuevo sistema operativo de Raspberry (Raspbian Stretch), ha sido necesaria la instalación de una librería adicional mediante el comando wget por consola. Esta librería está disponible en [31] y tiene su última modificación el 2018-03-29 a las 13:10. Una vez hecho esto, se ha podido acceder a ella sin complicaciones.

Se ha corroborado que, efectivamente, la interfaz de usuario proporcionada por Z-Wave responde ante cambios en los sensores y permite abrir o cerrar la llave de paso mediante la auto válvula. A cada nodo, se le asigna automáticamente un ID único para poder trabajar con él a la hora de realizar nuestra propia automatización del sistema.

Después de este proceso, la red Z-Wave está lista para proseguir con su automatización usando la interfaz web creada en este proyecto.

## 4.5.- MONTAJE DEL SERVIDOR WEB EN LA RASPBERRY

Antes de proceder a automatizar el proceso de envío de datos del sensor y actuador, se necesita crear un servidor en la Raspberry que permita alojar la interfaz de usuario, así

como una base de datos para almacenar los diferentes valores recibidos. El servidor debe estar online de forma permanente para que el usuario pueda hacer uso de la interfaz.

Una de las opciones más proliferadas en la web dada su alta popularidad, es lo que comúnmente se conoce como LAMP (Linux, Apache, MySQL y PHP o Python), se trata de las cuatro tecnologías más conocidas para la creación de un servidor de páginas web completo.

- **Linux** como sistema operativo del conjunto y previamente instalado.
- **Apache** funciona de servidor web.
- **MySQL** como base de datos.
- **PHP/Python** son lenguajes de programación para crear la lógica dentro de la página web.

Se han escogido estas opciones por la principal razón de ser las más documentadas. Notar que para el diseño de páginas web hacen falta otras dos tecnologías de las cuales se hablará más adelante: **HTML** y **CSS**.

La instalación del servidor, se ha llevado a cabo ejecutando una serie de comandos en la consola disponible en el entorno de Raspberry (VNC Viewer), los cuales no serán detallados pues son de fácil acceso a través de la web, por ejemplo, en [32]. Para el manejo de la base de datos se ha decidido usar el software phpMyAdmin, que permite trabajar con ella de forma cómoda. Para acceder a este software, basta con introducir la IP de la Raspberry Pi seguida de phpMyAdmin en el buscador web (IPESTÁTICA/phpmyadmin). El acceso requiere de usuario y contraseña, los cuales han sido configurados previamente durante su instalación.

## 4.6.- AUTOMATIZACIÓN DEL ENVÍO DE DATOS AL SERVIDOR

El software Z-Way se ocupa de la comunicación con los nodos Z-Wave, actualizando los valores de cada uno de ellos y enviando los comandos correspondientes [33]. La parte más importante de Z-Way es el núcleo de Z-Wave. Este núcleo, usa una API para comunicarse con los nodos del sistema. En este proyecto, la API seleccionada se ocupará, principalmente, de enviar los datos recogidos por los sensores y actuador al servidor web, así como de enviar las órdenes procedentes de la interfaz web a la auto válvula para su correcto funcionamiento.

Z-Way presenta numerosas API (*Application Programming Interface)*, todas ellas parcialmente diseñadas unas en otras (ver figura 3.4.), que nos ayudarán a automatizar este proceso. Básicamente son cuatro:

- **Z-Wave Device API (zDev)**. Ésta tiene a su vez dos versiones. La JSON API y la C Library API. Ambas, son capaces de comunicarse con Z-Way y las interfaces de usuario usando un servidor interno. Todo ello usando direcciones URL (http://IPESTÁTICA:8083/ZWaveAPI/Run/devices[x].instances[y].commandClasses[z].*).

  Esta API nos facilita dos funciones. La primera es la llamada *function classes*, que permite el manejo de la red como puede ser el incluir y excluir nodos y el enrutado de los mismos. La segunda, llamada *command classes*, se ofrece mediante la zDev API y hace que se ejecuten órdenes como la apertura o cierre de una válvula o recopilación de información, mediante los comandos SET o GET.

- **Third Party Technology APIs**. Implementan la misma lógica que la zDev API, pero para otras tecnologías de comunicación inalámbrica como EnOcean.

- **JavaScript API (JS API)**. Permite, además de ejecutar funciones basadas en direcciones URL, la creación de módulos JavaScript que son ejecutados dentro de Z-Way. Con la zDev API, se puede acceder a parámetros y funciones a través de un navegador (http://TUIP:8083/JS/Run/zway.devices[x].*), pero no ofrece órdenes lógicas superiores. Este inconveniente es solventado por la JS API, haciendo el proceso automático mediante módulos que son ejecutados continuamente.

- **Virtual Device API (vDev)**. Esta API permite un acceso puramente informativo mediante direcciones URL.

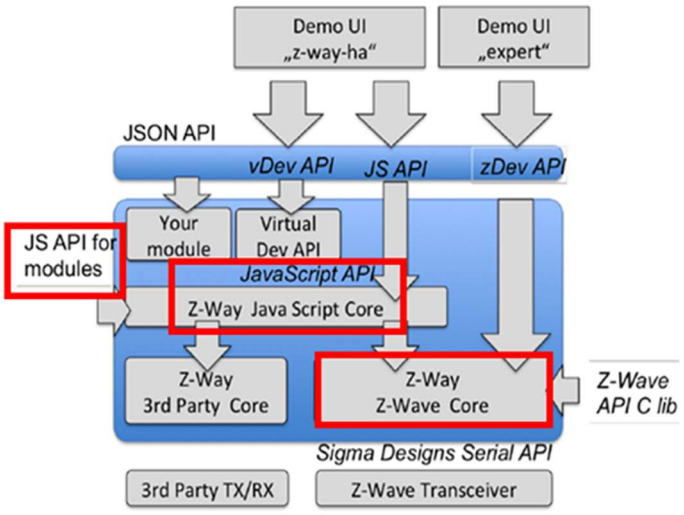


Figura 3.4.- Diferentes API que ofrece Z-Wave [33].

En primera instancia, la C Library API ha sido la opción escogida, puesto que es un lenguaje con el que el autor estaba bastante familiarizado. Debido a la carencia de información, tanto en la web, como en la documentación asociada a Z-Wave, se ha terminado optando por el camino de la JavaScript API, la cual está mucho más documentada y nos permite llegar a la misma solución, posiblemente de manera más sencilla.

### 4.6.1.- Automatización mediante la JavaScript API

Como se ha explicado, esta API, puede usarse para acceder a los parámetros o dar órdenes a los nodos a través de direcciones URL tal y como con la zDev API, pero, además, permite crear tus propios módulos para automatizar el proceso.

Se ha desarrollado un programa usando esta API con la finalidad de poder enviar los valores del sensor y válvula a la base de datos. Se ofrece la posibilidad de obtener los valores que se van a enviar por medio de una función, de tal modo que sólo se activa cuando algún valor del sensor/actuador cambia. De esta manera, se evita enviar datos innecesarios cada cierto tiempo, ya que supondría que éstos no han cambiado. Por ejemplo, el sensor podría pasarse mucho tiempo sin detectar agua y el envío continuo del mismo dato sería redundante. Esta función sólo es ofrecida por la API de Javascript y se llama *bind*.

Con todo esto llevado a cabo, el programa creado llamado binding.js, ha de ubicarse en la carpeta *automation*, a su vez dentro de la carpeta del servidor Z-Wave (carpeta *z-way-server*), en la Raspberry Pi, para que el programa se esté ejecutando constantemente. Contiene tres funciones *bind* para cada sensor de inundación, cada una apuntando al sensor y a su respectivo valor: una para la temperatura, otra para la detección de agua y otra para la batería. Además, existe otra apuntando a la auto válvula. Estas funciones, cuando se detecta un cambio en los valores de cualquier nodo, ejecutan el código que hay dentro de ellas, que básicamente envía los valores a la dirección del servidor web deseada por el método POST, para más adelante enviarlos a la base de datos.

La estructura de estas funciones sigue una organización igual para cada situación. Se comienza llamando a la función *bind*, que apuntará al nodo, y en el caso del sensor de inundación, al valor que se desea recoger (temperatura, detección de agua o batería). Si se produce un cambio en algún valor, se entrará a realizar el código de la función *bind* correspondiente. Para apuntar a cada nodo se usa, por un lado, el ID que Z-Wave le da a cada nodo al agregarlo al sistema y accesible mediante la interfaz de usuario proporcionada por Z-Wave. Por otro lado, las ya mencionadas *command classes*, que permiten traducir los datos enviados por los nodos y hacerlos entendibles. Las *command classes* están definidas

en la Z-Way Developers Documentation [33] y son diferentes dependiendo del elemento que se emplee.

Como ejemplo, en caso de la función para la temperatura del sensor, se llama a la función *bind* de la siguiente manera:

```
zway.devices[2].instances[2].commandClasses[49].data[1].val.bind(function(){})
```

Donde *devices* e *instances* es el ID asignado al sensor, concretamente al sensor de temperatura dentro del mismo. El número 49 para la *command classes* es el asignado para los sensores multinivel, que permite leer, entre otros, los datos enviados por el sensor usado para este proyecto. Finalmente, con *val* y *data*, nos enviará el valor ya traducido de la temperatura que se ha detectado.

Una vez que se ha entrado en la función, se implementará el código cuya finalidad es enviar los datos a nuestro servidor web y, en última instancia, a la base de datos, para así poder hacer uso de ellos. Para tal fin, se configuran una serie de opciones que describirán las características del envío. Entre ellas, cabe destacar que se ha de especificar la URL de envío a nuestra interfaz, el método, que en este caso se usa POST y, por último, se declara la variable para poder manejarla con el código de la interfaz y enviarla a la base de datos. Por último, se especifica que las opciones anteriormente descritas han de ejecutarse como una solicitud http.

Continuando con el ejemplo de la temperatura del sensor, la parte del envío quedaría:

```
try {
        var options = {};
        options.url = "http://192.168.0.30/portada1.php";
        options.method = "POST";
        options.data = "temperature="+this.value;
        var res = http.request(options);}
```

En caso de error en el envío, se ha decidido que se muestre por la consola de la Raspberry y se especifique el tipo de error mediante:

```
Catch(err) {debugPrint ("Failed to send data: "+ err);}
```

Es importante mencionar que, en el caso del envío de datos de la auto válvula, existía un problema, y es que se enviaban dos datos. El primero de ellos correspondía al estado actual de la válvula, mientras que el segundo, representaba el estado al que cambiaba, siendo éste el deseado. Para invalidar el primer envío simplemente se ha añadido un comando:

```
if (type & 0x20) return;
```

Este comando se ha escrito, lógicamente, antes de proceder a la parte del envío y evita el problema anteriormente mencionado. Para más detalles sobre el código del proceso de automatización mediante la JavaScript API se sugiere consultar los anexos.

Por último, en lo referente a este apartado, se ha de añadir que para que este programa (binding.js) se ejecute, se debe añadir al final del programa principal (main.js), en la carpeta *automation*, una sentencia: executeFile('binding.js'). De esta forma, cada vez que el servidor Z-Wave se inicia, cargará este programa principal y a su vez el que se ha diseñado.

## 4.7.- RECIBIR DATOS EN EL SERVIDOR WEB Y ENVÍO A LA BASE DE DATOS

Para recibir los datos enviados desde la JavaScript API se ha creado una página PHP cuyo nombre (portada1.php) coincide con el escogido para enviar los datos desde el programa binding.js. Este archivo está en la carpeta del servidor web de la Raspberry y se ocupa de crear una conexión con la base de datos y enviar los valores procedentes de Z-Way a la misma. En el momento que detecte la llegada de un envío por parte del programa Javascript se deben guardar en dicha base de datos para que así no se pierdan.

Con la intención de escribir el código de la página más fácilmente, se ha hecho uso de un editor de código llamado Notepad++ [34], que puede trabajar con distintos lenguajes, entre ellos PHP, HTML y CSS, necesarios para la creación de la interfaz. La funcionalidad de este editor es que permite trabajar desde tu propio ordenador y al mismo tiempo estar conectado directamente con la Raspberry para poder guardar los archivos en el directorio deseado. Al no tener privilegios de usuario root en la Raspberry ha sido necesario editar la carpeta donde se guardan las páginas de la interfaz, para así poder crear nuevos archivos en ella. Esta carpeta se ubica en el directorio /var/www/html de la Raspberry, y es donde deben guardarse todos los archivos que formen parte de la interfaz de usuario para que ésta funcione correctamente. Estas son las páginas que se muestran al buscar la dirección IP de la Raspberry (192.168.0.30) en un navegador dentro de la red local. Para poder utilizarla se usan dos comandos en la consola [35]:

```
sudo chown root : root -R /var/www/
sudo chmod 777 -R /var/www/
```

De este modo, conseguimos cambiar los privilegios de esta carpeta y así poder trabajar con ella.

Por otro lado, para crear la correspondiente tabla en la base de datos es necesario darle, de nuevo, los privilegios a nuestro usuario. Para ello, recurrimos directamente a phpMyAdmin, ya que dentro de la pestaña “Cuentas de usuarios” existe la posibilidad de dar privilegios globales al usuario que se escoja [36]. Con todo esto, se ha creado una base de datos llamada sensor, con su correspondiente tabla llamada valores (figura 3.5.), a la que irán a parar todos los datos de los nodos del sistema. Esta tabla se compone de cuatro columnas:

- El **ID** identificador de cada elemento.

- El **objeto**, es decir, el nombre del componente que envía el dato (auto válvula, sensor de temperatura, batería…).

- El **valor** del dato que es recibido.

- La **fecha** y hora en la que ha sido recibido ese valor.

| | id | objeto | valor | fecha |
|---|---|---|---|---|
| Editar Copiar Borrar | 300 | Auto Válvula | ABIERTA | 2018-03-30 16:43:49 |
| Editar Copiar Borrar | 299 | Sensor de inundación 2 | OFF | 2018-03-30 16:43:19 |
| Editar Copiar Borrar | 298 | Auto Válvula | CERRADA | 2018-03-30 16:43:12 |
| Editar Copiar Borrar | 297 | Sensor de inundación 2 | ON | 2018-03-30 16:43:10 |

Figura 3.5.- Tabla valores de la base de datos sensor.

Finalmente, para almacenar los valores en la tabla creamos el código PHP correspondiente. Este código se basa en el uso de sentencias preparadas, las cuales se usan para ejecutar sentencias parecidas a las de SQL pero con alta eficiencia [37]. Éstas son estrictamente necesarias en este proyecto, ya que, de no usarlas, se ha corroborado que los datos enviados desde la JS API serían incapaces de llegar a la base de datos creada, perdiéndose por el camino. Asimismo, estas sentencias previenen inyecciones en la base de datos, un método usado por hackers para entrar al sistema.

El esquema utilizado para esto se podría dividir en tres partes. La primera de ellas consiste en crear la conexión con la base de datos, asegurándose que no hay ningún tipo de error. Para ello, asignamos a una serie de variables las credenciales de acceso y el nombre de la base de datos a la que nos queremos conectar.

```
$conn = new mysqli($servername, $username, $password, $dbname);
```

Una vez hecho esto, preparamos el envío de los datos a la tabla valores mediante la función prepare(), que especifica a qué columnas de la tabla se van a enviar datos, los cuales por el momento no son conocidos y de ahí las interrogaciones. Además, mediante la función bind_param() enlazamos las variables con los valores que vamos a recibir. Las “s” hacen referencia a que los tres valores serán enviados como *string*.

```
$stmt = $conn->prepare("INSERT INTO valores (objeto, valor, fecha) VALUES (?, ?, ?)");
$stmt->bind_param("sss", $objeto, $valor, $fecha);
```

Por último, y continuando con el ejemplo del sensor de temperatura, recibimos los datos de la JS API y se los asignamos a las variables. Para esto, en caso de que se detecte el envío de datos del servidor Z-Wave mediante el método POST, se procederá a verificar que nodo envía los datos.

```
if($_SERVER['REQUEST_METHOD'] == 'POST'){
//SENSOR TEMPERATURA VALOR
        if(isset($_POST['temperature'])){
```

Si el nodo que envía los datos es, por ejemplo, el de temperatura, se verificará que el objeto que envía el valor coincide en nombre con el asignado en el programa binding.js. Es decir, se comprobará que el objeto que envía el valor se llama temperature.

En ese caso, se procede a fijar los valores a las variables que se van a enviar y guardar en la base de datos.

```
$objeto = "Sensor de temperatura 1";
$fecha;
$variable = round((float)$_POST['temperature'],1);
$valor = $variable . "&deg;C";
}
```

Por un lado, objeto, asignándole en este caso, la referencia al sensor de temperatura. La fecha, que se ha decido enviar vacía, ya que se ha configurado en phpMyAdmin de tal manera que en el momento que la base de datos reciba algo la columna de la fecha se actualice. Por último, variable, que se utilizará para redondear el dato a una cifra decimal para así guardarlo definitivamente en valor.

Con todo esto, el último paso es ejecutar la sentencia de envío a la base de datos mediante $stmt->execute().

De igual manera, se realiza el envío de los demás datos variando únicamente a la hora de establecer los valores a las variables. Cabe mencionar que, en los casos de la auto válvula y la detección de agua, el nombre con el que se recibe el dato de la JS API puede ser 255 o 0 en el caso de detección de agua, o true o false en el caso de la auto válvula. Esto se traduce vía PHP en on, si se recibe 255, o en off, si por el contrario se recibe un 0.

```
$variable = $_POST['sensor_level'];
        if ($variable=="255"){
                $valor = "ON";}
        else if ($variable == "0"){
                $valor = "OFF";}
```

En el caso de la auto válvula, true significaría que la válvula está abierta y false que está cerrada.

Para más detalles en cualquier código descrito en este apartado, consultar el código comentado de los anexos.

## 4.8.- MANEJO DE LA VÁLVULA DESDE LA INTERFAZ DE USUARIO

Para poder llevar a cabo la apertura y cierre manual de la válvula desde la interfaz desarrollada, así como el cierre automático cuando se detecte agua por medio de cualquier sensor, se ha vuelto a hacer uso de las API que Z-Wave ofrece.

Como bien se ha mencionado con anterioridad, los nodos de Z-Wave pueden ser controlados mediante peticiones HTTP. Estas peticiones las ofrecen tanto la Z-Wave Device API como la JavaScript API [33]. Con ambas llegaríamos al mismo propósito, pero con una

dirección URL un tanto diferente en cada caso, por lo que es totalmente indiferente el uso de una u otra API.

1. URL usando la Z-Wave Device API: http://IP:8083/ZWaveAPI/Run/devices[].instaces[].commandClasses[]

2. URL usando la JavaScript API: http://IP:8083/JS/Run/zway.devices[] .instaces[].commandClasses[]

Donde *devices*, *instances* y *commandClasses* harán la misma referencia que en el programa binding.js creado con anterioridad. Por último, debe hacerse uso de la función Set, que indicará el estado al que se desea que la auto válvula pase.

En este caso, se ha decidido usar la zDev API. La petición HTTP de cierre o apertura de la válvula con su ID completo quedaría:

```
http://192.168.0.30:8083/ZWaveAPI/Run/devices[8].SwitchBinary.Set()
```

El ID asignado a la auto válvula por Z-Way es el 8 y, dado que ésta no está compuesta por más subsensores, no es necesario usar el argumento *instances*. Por otro lado, el valor de Set será true o false en caso de que se quiera abrir o cerrar la válvula respectivamente.

Hasta este punto, se ha conseguido generar la petición HTTP al servidor Z-Wave mediante una de sus API, pero se necesita una manera de ejecutar este comando desde el código PHP de la interfaz diseñada. Con este propósito, entra en juego cURL, una herramienta muy útil para hacer peticiones HTTP y multiplataforma. Es posible utilizarla usando la terminal de la Raspberry y mediante PHP gracias a la librería libcurl, la cual viene con la instalación de PHP [38]. Esta librería nos permitirá conectarnos y comunicarnos con el servidor Z-Wave, pudiendo ejecutar las peticiones HTTP de la auto válvula.

Dado que la petición HTTP no puede ser ejecutada sin antes haber iniciado sesión, el programa en cuestión se encargará de iniciar sesión como usuario de Z-Wave, para así poder ejecutar la petición de cierre o apertura de la auto válvula. Para ello, la estructura de ha de ser la siguiente [39]:

1. Se debe inicializar una sesión cURL usando curl_init(). Esta función contendrá la URL a ejecutar, en nuestro caso, la descrita anteriormente apuntando a la auto válvula. Tendrá dos posibilidades, una para el cierre y otra para la apertura.

```
function control_valvula($url){
$ch=curl_init($url);
```

2. Han de establecerse todas las opciones para la transferencia usando curl_setopt(). Dada la necesidad de configurar varias se usa curl_setopt_array().

```
$opciones=array(CURLOPT_CONNECTTIMEOUT => 100,
CURLOPT_TIMEOUT => 100,
CURLOPT_USERPWD => "admin:jo95ni2011",
CURLOPT_POST => true,
CURLOPT_POSTFIELDS => "login=admin&password=jo95ni2011",);
curl_setopt_array($ch,$opciones);
```

Con la variable opciones, estableceremos las diferentes configuraciones. Como seguridad, configuramos CONNECTTIMEOUT, que especificará el número de segundos antes de desconectar la sesión y así asegurar que el servidor Z-Wave recibe la petición HTTP. TIMEOUT para establecer el tiempo máximo para ejecutar las funciones cURL. Mediante USERPWD asignamos las credenciales para poder iniciar sesión en nuestro usuario Z-Wave y, por consiguiente, poder ejecutar la petición HTTP. Finalmente, se

establece mediante POST y POSTFIELDS el envío de la petición HTTP y de las credenciales, respectivamente.

3. Ejecutamos la sesión con curl_exec($ch).

4. Finalizamos la sesión con curl_close($ch).

Con todo ello, se ha creado el programa curl.php, que nos permite ejecutar la función control_válvula($url) en cualquier parte de nuestra interfaz de usuario siempre y cuando llamemos antes a este programa.

A modo de ejemplo, cada vez que deseemos controlar la auto válvula desde la interfaz el código diseñado tendrá que contener, por un lado, el archivo curl.php importado mediante require_once ('/var/www/html/curl.php'). Por otro lado, se hará uso de la función creada en este archivo. Para ello, en caso de querer abrir la auto válvula ha de escribirse:

```
control_valvula("http://192.168.0.30:8083/ZWaveAPI/Run/devices[8].SwitchBinary.Set(true)")
```

En el caso de querer cerrarla simplemente se sustituiría true por false.

## 4.9.- NOTIFICACIÓN EXTERNA EN CASO DE DETECCIÓN DE AGUA POR PARTE DEL SENSOR

En caso del envío de detección de agua por parte de binding.js a portada1.php, se ha decidido automatizar el envío de un mensaje vía e-mail y Pushbullet [39] utilizando Python y PHP. Pushbullet es una aplicación de mensajería que no supone las limitaciones que WhatsApp y Telegram han implicado y, por las cuales, no se han utilizado en este proyecto.

### 4.9.1.- Mensaje mediante Pushbullet

En el caso de WhatsApp, existe una librería para Python, que permite mandar mensajes a un smartphone sin requerir de otro número de teléfono. Telegram no contaba con esta opción, por lo que fue descartado. Finalmente, se ha decidido prescindir también de WhatsApp debido a que esta librería no está aceptada por la empresa y, en caso de usarla, se corre el riesgo de bloqueo del número de teléfono. Por todo esto, se ha acabado usando Pushbullet, pues permite trabajar con total libertad, creando nuestro programa con el editor Python y sin requerir de número de teléfono.

Para ello, se ha creado una cuenta en Pushbullet y se ha descargado la aplicación en nuestro smartphone para recibir las notificaciones. Una vez hecho esto, se ha creado el siguiente archivo ejecutable (pushbullet.sh) para poder llamarlo desde Python [39]:

```
#!/bin/bash
API="o.RFRwQtZzE3Fh4MVvmSK6eql2gyBnJkMo"
MSG="$1"
curl -u $API: https://api.pushbullet.com/v2/pushes -d type=note -d title="Alerta Z-Wave!" -d body="$MSG"
```

En él, introducimos la *API key* localizada en la pestaña ajustes de la aplicación y definimos el tipo de mensaje que se va a enviar, especificando su título y dejando el mensaje a ser escrito desde el programa Python. Este archivo se ha guardado en el directorio /home/pi/Program. Al carecer este de permisos de ejecución se le han tenido que dar mediante los comandos en la consola:

```
cd Program
chmod 755 pushbullet.sh
```

Por último, se ha generado el programa Python que llamará al ejecutable pushbullet.sh y definirá el mensaje a enviar. Este programa se ha ubicado con el resto de las páginas de la interfaz de usuario en /var/www/html.

```
import os

os.system('/home/pi/Program/pushbullet.sh "Se ha detectado agua en su hogar gracias a su sensor Z-Wave"')
```

El resultado final es una notificación a través de esta aplicación como puede verse en la figura 3.6.

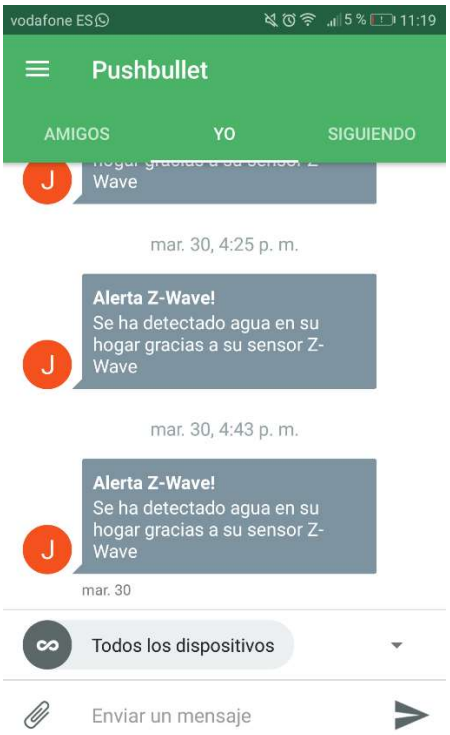


Figura 3.6.- Mensaje a través de Pushbullet.

### 4.9.2.- Mensaje mediante e-mail

Por otro lado, se ha generado un programa Python, capaz de enviar un mensaje al e-mail del usuario del sistema domótico. Además, este mensaje, adjunta una imagen tomada por la Pi Cámara, cuyo proceso de automatización se describirá más adelante. Por ahora, para el correcto entendimiento del lector, cabe mencionar que esta imagen es guardada en el mismo directorio que todas las páginas de la interfaz.

Antes de crear el programa, se ha dado de alta una cuenta de Gmail desde la que se enviarán los mensajes al usuario. Esta cuenta, ha de ser específicamente de Gmail, ya que la

librería usada en el programa está pensada solo para esta compañía. Sin embargo, Google no permitirá el inicio de sesión a través de esta librería porque ha marcado este tipo de inicio de sesión como "menos seguro". Para solucionar este problema, se ha configurado en los ajustes de la cuenta de Google el permitir a aplicaciones menos seguras su uso. Con todo esto, se ha procedido al desarrollo del programa en cuestión.

Para esto, se ha comenzado por importar la librería SMTP, un módulo de Python utilizado para enviar e-mails y que maneja el envío y el correo electrónico de enrutamiento entre los servidores de correo [40].

```
import smtplib
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
from email.mime.base import MIMEBase
from email import encoders
```

Una vez definida la librería y sus módulos, se ha establecido el remitente y destinatario del correo, así como el asunto y texto a mostrar.

```
fromaddr = "jorgenpalacio.tfg@gmail.com"
toaddr = "jorgenpalacio@hotmail.com"
msg = MIMEMultipart()
msg['From'] = fromaddr
msg['To'] = toaddr
msg['Subject'] = "Alerta Z-Wave!"
body = "Se ha detectado agua en su hogar gracias a su sensor Z-Wave. Se adjunta una imagen del estado actual de la habitacion."
msg.attach(MIMEText(body, 'plain'))
```

Seguidamente, se ha especificado el nombre de la imagen a adjuntar y tras una serie de configuraciones se ha juntado al mensaje mediante msg.attach(part).

```
filename = "image1.jpg"
attachment = open(filename, "rb")
part = MIMEBase('application', 'octet-stream')
part.set_payload((attachment).read())
encoders.encode_base64(part)
part.add_header('Content-Disposition', "attachment; filename= %s" % filename)
msg.attach(part)
```

Por último, se ha definido el puerto de servidor Gmail y la ejecución del envío.

```
server = smtplib.SMTP('smtp.gmail.com', 587)
server.starttls()
server.login(fromaddr, "jo95ni2011")
text = msg.as_string()
server.sendmail(fromaddr, toaddr, text)
server.quit()
```

Se ha de destacar que, ya que el envío de ambas notificaciones es simultáneo, se ha decido juntar tanto el programa de Pushbullet como el del e-mail en un mismo script Python llamado envioemail.py alojado, como siempre, en /var/www/html.

La notificación final al correo del usuario se muestra en la figura 3.7.


Figura 3.7.- Notificación con imagen adjunta al correo del usuario.

## 4.10.- CONFIGURACIÓN DE LA PI CÁMARA

Para realizar las capturas de la imagen que será enviada como notificación y a la cual se podrá acceder y actualizar mediante la interfaz de usuario, se ha usado la cámara desarrollada por la empresa Raspberry. Dado que es fabricada por la misma empresa, no existe ningún problema de compatibilidad y es conectada mediante la ranura CSI (*Camera Serial Interface*) de la Raspberry.

La Raspberry Pi Cámara requiere de una configuración inicial que habilite su uso. Se lleva a cabo desde el propio entorno VNC en la opción de configuración de la Raspberry, más concretamente, habilitando la opción *Camera* dentro de la pestaña *Interfaces* como puede verse en la figura 3.8.

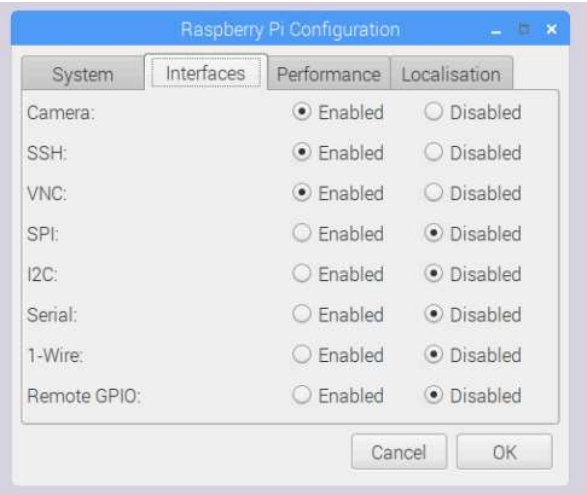


Figura 3.8.- Opciones de configuración.

Una vez llevado a cabo esto, se ha creado un programa en Python encargado de capturar una imagen del lugar y guardarla en el servidor web, cuando la orden de dicha captura se ejecute. Es decir, cuando se presione el botón de tomar imagen en la interfaz que más tarde se explicará, o cuando cualquiera de los sensores detecte agua.

Este programa, se encarga de sacar una foto haciendo uso de las librerías existentes.

```
import picamera
from picamera import PiCamera, Color
import time
```

Además, se configuran una serie de ajustes para mejorar su calidad, como son el brillo y un texto informativo en la imagen.

```
camera=picamera.PiCamera()
camera.start_preview()
camera.brightness = 50
camera.annotate_text_size = 50
camera.annotate_background = Color('red')
camera.annotate_foreground = Color('white')
camera.annotate_text = "Asi esta tu habitacion"
```

Finalmente, tomamos la imagen y cerramos el programa. Ésta se guarda en el servidor web bajo el nombre de image1.jpg, la cual se actualiza, borrando la anterior, cada vez que este pequeño programa se ejecute.

```
camera.capture('image1.jpg')
camera.stop_preview()
camera.close()
```

El programa se ha nombrado panorámica.php, con la intención de llamarlo cada vez que el botón para tomar una imagen se active en la interfaz. Por otro lado, este código se ha incluido en el programa de notificaciones envioemail.php, ya que cuando se detecte agua y se envíe una imagen adjunta al correo del usuario, ésta ha de estar actualizada. Ambos programas se encuentran detallados en los anexos.

## 4.11.- DISEÑO Y DESARROLLO DE LA INTERFAZ DE USUARIO

En este punto, todo está debidamente implementado como para proceder a desarrollar la parte del proyecto que va a interactuar con el usuario, es decir, la interfaz web. Ésta se compone de cinco páginas, que serán descritas brevemente a continuación, para luego proceder a su explicación en detalle:

1. Página que muestra las imágenes tomadas por la cámara. El usuario puede actualizarla por medio de un botón cuando así lo desee. Esta página toma el nombre de **portada2.php**.

2. La página principal llamada **portada0.php,** para mostrar la información actual de todos los sensores, así como para controlar la auto válvula mediante dos botones.

3. Otra dedicada exclusivamente a mostrar un historial con los últimos 15 datos recibidos por parte de sensores y actuador. Se llama **portada1.php**.

4. Una página meramente informativa, llamada **logo.php**, que describe el motivo de este trabajo y su autor.

5. Por último, **index.php**, que es la página que corresponde al inicio de sesión mediante las credenciales de acceso. En caso de no haber iniciado sesión todas las demás páginas, redirigen al usuario a ésta.

La estructura HTML de todas las páginas es muy similar, por lo que no se entrará en detalles individuales ya que resultaría una explicación demasiado extensa y repetitiva. Por el contrario, si se explicará en detenimiento la lógica PHP implementada.

Para el diseño HTML se ha hecho uso de la herramienta Bootstrap [40], que nos ofrece plantillas para conseguir un desarrollo web más rápido y simple. Ofrece diseños de plantillas basadas en HTML y estilos CSS, así como varios plugin JavaScript. Con ello y con la ayuda de w3schools [41] y sus tutoriales online, se ha conseguido generar el esqueleto de las páginas que componen la interfaz de usuario.

Todas las páginas contienen una imagen diseñada con la ayuda de PowerPoint y a partir de otras imágenes. Asimismo, todas ellas menos la de inicio de sesión, contienen una barra de navegación a las diferentes páginas, así como una opción de cierre de sesión. Esto se muestra en la figura 3.9.

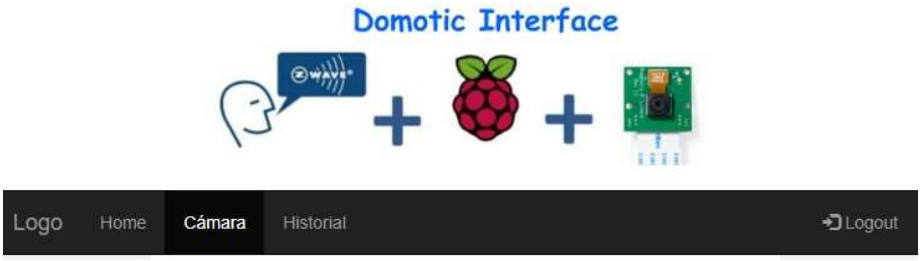


Figura 3.9.- Cabecera de la interfaz de usuario.

Ha de recordarse que para la realización de la interfaz se ha usado el anteriormente mencionado editor de código Notepad++.

### 4.11.1.- Página que sirve imágenes

Esta página se compone de un botón bajo el nombre de Actualizar Imagen, como puede verse en la figura 3.10. En el caso de ser pulsado, el nombre de envío escogido (panorámica) será enviado mediante el método POST. El código PHP desarrollado en la misma página será el encargado de detectar este envío y llamar al programa panorámica.py, que hará a la Pi Cámara sacar la foto.

```
//Funcion PHP fotos
if($_POST[panoramica]){
$a- exec("sudo python /var/www/html/panoramica.py");
```

```
sleep(1);
header("Location:http://192.168.0.30/portada2.php");
echo $a;}
```

Además, se espera un segundo para volver a recargar la página con la nueva imagen mediante la instrucción header().


Figura 3.10.- Página web de la imagen (http://192.168.0.30/portada2.php).

Por último, cabe mencionar que esta página, al servir imágenes, puede ser que tarde en actualizarse a la imagen nueva, puesto que, al ser un objeto pesado, el caché de los navegadores la retiene. Para solucionar este problema, se recomienda cada vez que se quiera actualizar la imagen, pulsar las teclas Ctrl+F5, ya que estas refrescarán la caché dejando paso a la nueva imagen.

### 4.11.2.- Página principal

Esta página se conecta a la base de datos, concretamente, a la tabla valores y muestra el estado actual de todos los componentes de ambos sensores, así como el estado de la auto válvula. Además, dispone de dos botones, que mediante la función cURL manejan la auto válvula a gusto del usuario. Los valores de los sensores que se muestran son la temperatura

ambiente, la batería y la detección de agua por parte de estos. Todos estos detalles pueden verse en la figura 3.11.

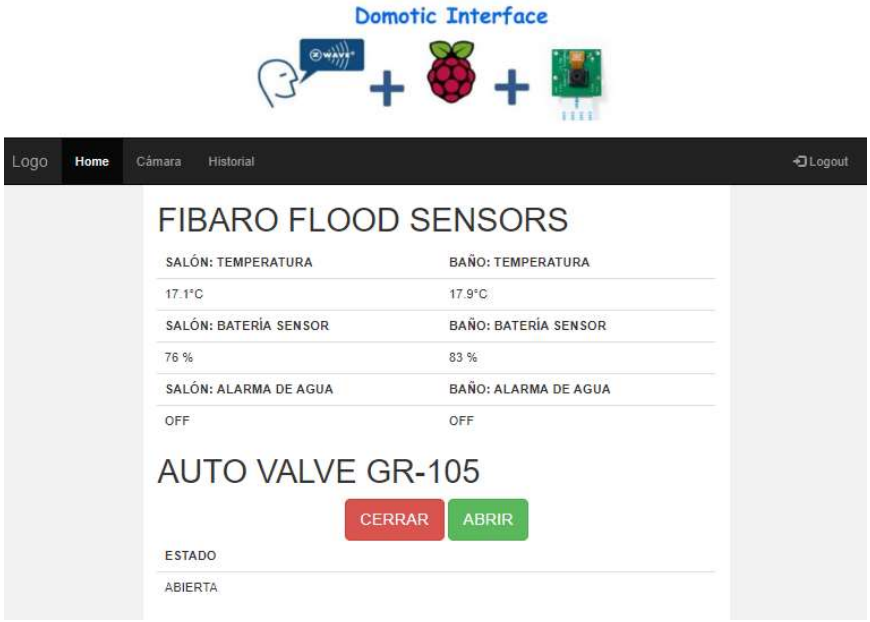


Figura 3.11.- Estado actual de cada parámetro (http://192.168.0.30/portada0.php).

Los datos se han mostrado en forma de tabla. Para ello, se ha hecho una consulta a la base de datos de tal manera que proporcionase el último valor que se haya recibido de cada parámetro. Por ejemplo, para saber la temperatura que está midiendo uno de los sensores la consulta selecciona el dato valor para el objeto Sensor de temperatura 1:

```
$sql = "SELECT valor FROM valores WHERE objeto = 'Sensor de temperatura 1' ORDER BY id DESC LIMIT 1";
```

Siendo la instrucción “ORDER BY id DESC LIMIT 1” utilizada para coger el último valor. Previamente, se ha creado una conexión con la base de datos, pues sin ella, no se podría llevar a cabo ninguna consulta.

Por otro lado, para poder manejar la válvula se han creado dos botones, los cuales, en caso de clic, envían su propia señal POST al código PHP de la página. Éste, es el encargado de ejecutar la función control_valvula() creada mediante cURL.

```
//EJECUTAR CURL
if($_POST[close]){
control_valvula("http://192.168.0.30:8083/ZWaveAPI/Run/devices[8].SwitchBinary.Set(false)");
sleep(1);
header("Location:http://192.168.0.30/portada0.php");}
if($_POST[open]){
control_valvula("http://192.168.0.30:8083/ZWaveAPI/Run/devices[8].SwitchBinary.Set(true)");
sleep(1);
header("Location:http://192.168.0.30/portada0.php");}
```

Si la señal POST detecta close, se utilizará false para el cierre de la válvula en la petición HTTP, por el contrario, si se detecta open, se utilizará true. En ambos casos, se refresca la página automáticamente para mostrar el nuevo valor de la auto válvula. Para ello, ha de hacerse uso del archivo curl.php creado, importándolo mediante:

```
require_once ('/var/www/html/curl.php');
```

### 4.11.3.- Página con el historial

Se encarga de mostrar los últimos 15 sucesos que los nodos del sistema domótico han detectado, además de enviar los datos procedentes del programa binding.js a la base de datos como se ha explicado con anterioridad en el presente documento. Ahora nos centraremos en la parte de consulta a la base de datos.

Para ello, una vez conectados a la base de datos se ha hecho una consulta en la que se seleccionan los últimos 15 sucesos, independientemente del nodo que los envíe. De ahí el uso en este caso de SELECT *.

```
$sql = "SELECT * FROM valores ORDER BY fecha DESC LIMIT 15";
$result = $conn->query($sql);
        foreach($result as $row){
                echo"<tr>";
```

```
                echo"<td>" . $row['objeto'] . "</td>";
                echo"<td>" . $row['valor'] . "</td>";
                echo"<td>" . $row['fecha'] . "</td>";
                echo"<tr>";}
$conn->close();
```

Esta consulta, se ordena a modo de tabla en tres columnas. Objeto para el nombre del nodo, valor del estado del nodo y la fecha en que se ha producido dicho cambio.

Por otra parte, en el código responsable del envío a la base de datos, se ha añadido la ejecución en segundo plano de un código PHP escrito en un nuevo archivo llamado e-mailycURL.php. Esta ejecución, se realiza en el caso de recibir que uno de los dos sensores haya detectado agua.

```
exec("php e-mailycURL.php > /dev/null 2>&1 &");
```

El programa, al que se llama en segundo plano, cierra la auto válvula y ejecuta el programa Python envioemail.py, que envía las notificaciones externas adjuntando una imagen. Este se muestra a continuación.

```
//EJECUTAR CURL
control_valvula("http://192.168.0.30:8083/ZWaveAPI/Run/devices[8].SwitchBinary.Set(false)");
//MANDAR PUSHBULLET Y MANDAR EMAIL
$a- exec("sudo python /var/www/html/envioemail.py");
echo $a;
```

Se ha decidido ejecutarlo en segundo plano mediante la instrucción “> /dev/null 2>&1 &” ya que, al no hacerlo el sistema, no respondía en tiempo real a las órdenes del código. Seguramente, debido a que el proceso abarcaba demasiado código para que todo se ejecutase de manera fluida.

El resultado final de esta página se muestra en la figura 3.12.


Figura 3.12.- Página de historial de sucesos (http://192.168.0.30/portada1.php).

### 4.11.4.- Página con información sobre el autor y trabajo

Se compone de código HTML únicamente, ya que no hay necesidad de interactuar con la base de datos, ni con el usuario, ni con la JavaScript API. Es una página estática y puramente informativa, como puede verse en la figura 3.13.

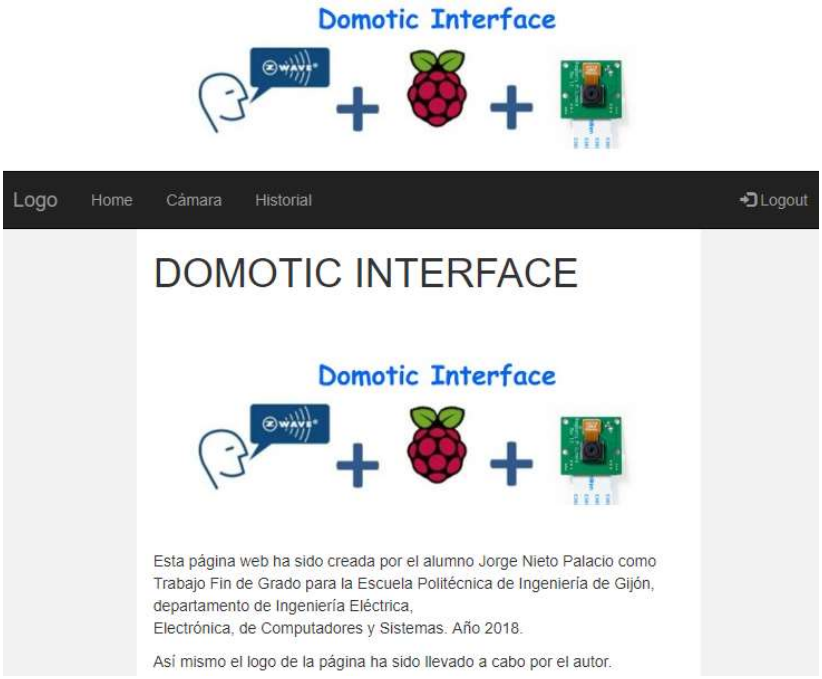


Figura 3.13.- Página informativa (http://192.168.0.30/logo.php).

### 4.11.5.- Página de inicio de sesión

Para elaborar el sistema de inicio de sesión mediante usuario y contraseña, se ha vuelto a hacer uso de la base de datos sensor. Esta vez, se ha creado otra tabla llamada usuarios, donde se han guardado el usuario y contraseña a utilizar en el sistema como puede verse en la figura 3.14.


Figura 3.14.- Tabla usuarios de la base de datos sensor.

Una vez definidos, se ha elaborado la página index.php, mostrada en la figura 3.15. Al llevar este nombre, cuando buscas la interfaz de usuario en un buscador web, automáticamente te redirige a la página que tenga por nombre index.php, sin necesidad de configurar nada. Por ese motivo, se ha reservado ese nombre para esta página en concreto.


Figura 3.15.- Página de inicio de sesión (http://192.168.0.30/index.php).

Ésta, se compone de dos espacios para ingresar el usuario y contraseña y de un botón, para una vez ingresados, acceder a la interfaz. El espacio para ingresar la contraseña, al ser de tipo *password*, representa las contraseñas por puntos, para que no sean visibles. Cuando el botón de entrar sea accionado se verificarán dos cosas:

- Que ambos campos contengan texto.

- Que las credenciales de acceso coincidan con las guardadas en la base de datos.

En caso de que algo de esto no se confirme, una ventana emergente correspondiente a cada caso aparecerá, habiendo tres tipos de notificaciones. Una que informa que, o bien el usuario o la contraseña son incorrectos, y dos que informan que campo no se ha cumplimentado.

Si todo está correctamente cumplimentado, al accionar el botón de entrar se redirigirá al usuario a la página principal de la interfaz, es decir, a portada0.php. Por otro lado, si las credenciales son las correctas, serán enviadas por el método POST a una nueva página llamada login.php. Esta página se encargará de crear un inicio de sesión mediante la

instrucción session_start(), el cual nos servirá para verificar en todas las páginas que la sesión está iniciada, ya que de lo contrario se redirigirá a index.php. La verificación de inicio de sesión se consigue añadiendo en todas las demás páginas de la interfaz el siguiente código PHP:

```
session_start();
if(isset($_SESSION['user'])){}
else {header("location:index.php");}
```

Para romper la sesión se dispone en todas las páginas del anteriormente mencionado botón de logout, que llama al archivo salir.php. Este archivo deshace la sesión y redirige a index.php.

```
session_start();
unset($_SESSION['user']);
header("location:../index.php");
```

Para mayor detalle en todos los programas consultar los anexos.

## 4.12.- CONFIGURACIÓN DE UN NOMBRE DE DOMINIO

Hasta este punto, la interfaz de usuario sólo es accesible dentro de la red local y mediante la dirección IP asignada a la Raspberry Pi. Dado que queremos poder acceder a nuestro sistema domótico desde cualquier lugar del mundo, merecerá la pena buscar una solución. Asignar un nombre de dominio a una dirección IP no suele ser algo gratuito, sin embargo, existen opciones menos estéticas y más laboriosas que sí lo son.

Una opción conocida es NO-IP [42], un proveedor de DNS, tanto de pago como gratuito. NO-IP también ofrece certificados SSL, que garantizan mayor seguridad a nuestro sitio web por su fuerte encriptación del contenido. Este servicio sería de pago irremediablemente. Al ser nuestro caso el gratuito, tendremos el inconveniente de que en

nuestra dirección IP aparecerá el nombre del host que lo ofrece que, en este caso, es ddns.net. Además, habrá que activar la dirección cada 30 días si no queremos que el nombre de dominio sea borrado.

Hemos dado de alta una cuenta en este sitio web y creado el nombre de dominio para nuestra interfaz. Éste es www.domoticinterface.ddns.net.

Para configurar NO-IP en la Raspberry tendremos que ejecutar por consola el siguiente comando:

```
wget http://www.no-ip.com/client/linux/noip-doc-linux.tar.gz
```

Una vez hecho esto y tras seguir una serie de pasos (documentados en [43]) para el arranque automático de NO-IP al encender la Raspberry, sólo quedaría abrir los puertos de nuestro rúter para poder acceder desde cualquier otra red externa. Esto requiere acceder al rúter mediante el navegador web para configurarlo. Sabiendo su IP accedemos y especificamos, dentro de la opción “redirección de puertos”, en la pestaña “internet”, la dirección IP de la Raspberry en el puerto público 80. Además, activamos la opción DNS especificando nuestro nuevo nombre de dominio, para que así sea accesible desde internet.

# 5. Prueba del sistema completo.

Una vez el sistema domótico está completamente operativo y la interfaz de usuario totalmente desarrollada, se han llevado a cabo diferentes pruebas con el objetivo de corroborar su buen funcionamiento. Todos estos test se han realizado dentro de una vivienda de 120 metros cuadrados. Sin mover el controlador, se han ido cambiando de lugar tanto los sensores como la auto válvula, desde unas distancias mínimas hasta lo más separados posible. Este testeo, se ha realizado utilizando la interfaz tanto desde un ordenador como desde un smartphone. Se ha corroborado que:

- Ante cambios en los sensores la interfaz web se actualiza.

- Si algún sensor detecta agua, la auto válvula responde cerrándose. Toda la interfaz se actualiza y las correspondientes notificaciones vía e-mail y Pushbullet son recibidas. Además, la Pi Cámara toma una foto y ésta se actualiza en la interfaz y se adjunta en el e-mail.

- La auto válvula se puede controlar mediante la interfaz.

- Se pueden actualizar las imágenes mediante la interfaz.

- El cierre e inicio de sesión funcionan correctamente.

- La interfaz es accesible fuera de la red local y mediante su nombre de dominio www.domoticinterface.ddns.net.

- La navegación dentro de la interfaz de usuario es fluida y responde adecuadamente ante todos los cambios que el usuario origine.

Todo esto, se ha verificado desde distintas distancias dentro de la vivienda, no encontrando ningún tipo de error incluso en las distancias más largas.

# 6. Presupuesto.

A lo largo de este capítulo, se llevará a cabo un desglose de los costes asociados a este proyecto.

## 6.1.- COSTES DIRECTOS

A la hora de evaluar los costes directamente imputables al proyecto, una ventaja es el haber utilizado en gran medida el software libre descrito a lo largo del documento, que representa un considerable ahorro en cuanto a coste de licencias. Además, al haber configurado nuestro propio controlador nos ha permitido añadirle más versatilidad al proyecto y más funcionalidad, todo ello sumado al ahorro que supone el uso de este.

### 6.1.1.- Coste del personal

Este proyecto ha sido llevado a cabo íntegramente por una única persona. Dicha persona, puede ser categorizada profesionalmente como un ingeniero técnico con un contrato por obra y servicio. Partiendo de esta base, se ha deducido el coste asociado al proyecto de dicha persona en la tabla 5.1, para así poder estimar el coste horario. Para ello, se ha estimado un sueldo bruto mensual para el trabajador de 3500 € y sabiendo que el proyecto ha durado 10 meses se obtiene un sueldo total de 35000 €, cuyo coste asociado a la empresa vendrá incrementado por impuestos tales como el IRPF, el impuesto sobre el Seguro de Desempleo, la Seguridad Social, etc. El porcentaje que supone la totalidad de estos impuestos se ha estimado de manera aproximada en un 30%. El sueldo bruto del trabajador se ha considerado teniendo en cuenta los salarios existentes en el mercado.

| Concepto | Cantidad (€) |
|---|---|
| Sueldo bruto (10 meses) | 35000 |

| | |
|---|---|
| Impuestos (30%) | 10500 |
| Coste total | 45500 |

Tabla 5.1.- Coste anual del personal.

Con esto, se ha procedido a estimar el coste horario que supone el trabajador. Para ello, se han calculado los días laborales de los 10 meses que ha comprendido el proyecto como viene reflejado en la tabla 5.2. Estos meses comprenden desde octubre de 2017 a julio de 2018.

| Días en 10 meses | Días de fines de semana | Festivo | Días laborales |
|---|---|---|---|
| 304 | 87 | 9 | 208 |

Tabla 5.2.- Días laborales anuales.

Con toda esta información, se ha dividido el coste del personal entre el número de días laborales obteniendo un coste de 218,75 $\frac{€}{día}$ . A su vez, suponiendo una jornada laboral de 8 horas diarias se ha establecido el coste horario del trabajador, que resulta de 27,34 $\frac{€}{hora}$.

Para, finalmente, calcular el coste directo asociado al personal se ha establecido una distribución temporal de las tareas que han compuesto el proyecto en la tabla 5.3., obteniendo una referencia temporal de este.

| Tareas | Duración (horas) |
|---|---|
| Investigación | 107 |
| Desarrollo del sistema | 168 |
| Pruebas | 13 |

| | |
|---|---|
| Documentación y gestión | 84 |
| TOTAL | 372 |

Tabla 5.3.- Distribución temporal de tareas.

Para terminar, el coste del personal es la multiplicación del coste horario por el número de horas que ha llevado el proyecto. La cifra asciende a 10170,48 €.

### 6.1.2.- Amortización del equipo informático

El desarrollo del proyecto se ha llevado a cabo utilizando un ordenador portátil Lenovo G500. Es por ello, por lo que se debe considerar el coste de amortización asociado, pues como se ha mencionado anteriormente, no existen costes asociados al software o sistemas operativos.

Para estimar su amortización, se utilizará el método lineal simple, por ser éste el más habitual. Los cálculos se han detallado en la tabla 5.4. Su cálculo se lleva a cabo dividiendo el precio del portátil entre su vida útil, teniendo en cuenta un valor residual nulo.

| **Lenovo G500 (€)** | **Vida útil (años)** | **Coste de amortización (€/año)** |
|---|---|---|
| 428.20 | 5 | 85.64 |

Tabla 5.4.- Amortización del equipo.

Dado que el proyecto ha durado 10 meses y el valor calculado es en un año, este coste de amortización se reduce a 71,37 €.

### 6.1.3.- Coste del material empleado

El coste de los materiales empleados (tabla 5.5.) representa a los elementos que han sido utilizados durante el desarrollo del proyecto y que van a ser parte del producto final.

| Componente | Precio Unitario (€) | Unidades | Importe (€) |
|---|---|---|---|
| Raspberry Pi | 34,12 | 1 | 34,12 |
| RaZberry | 61,59 | 1 | 61,59 |
| Cámara | 25,40 | 1 | 25,40 |
| Sensor de inundación | 50,90 | 2 | 101,80 |
| Auto válvula | 52,63 | 1 | 52,63 |
| microSD | 6,90 | 1 | 6,90 |
| | TOTAL | | 282,44 |

Tabla 5.5.- Costes de la instalación.

### 6.1.4.- Costes directos totales.

Los costes directos totales serán la suma de todos los costes planteados con anterioridad. La recopilación de estos se muestra en la tabla 5.6.

| Costes | Precio (€) |
|---|---|
| Costes del personal | 10170,48 |
| Costes de amortización | 71,37 |
| Coste del material empleado | 282,44 |
| TOTAL | 10524,29 |

Tabla 5.6.- Costes directos totales.

## 6.2.- COSTES INDIRECTOS

Los costes indirectos son aquellos asociados al proceso productivo del proyecto que no son imputables de forma directa a éste. Como son, la electricidad, calefacción, internet, y otros servicios de carácter general. Para su cálculo, se aplicará un porcentaje del 15 % de los costes directos. Éste se trata de un valor de registro habitual en proyectos de Investigación y Desarrollo según la Resolución del 5 de noviembre de 2013, de la Secretaría de Estado de Investigación, Desarrollo e Innovación [44].

Aplicando dicho porcentaje los costes indirectos ascienden a 1578,64 €.

## 6.3.- COSTES TOTALES Y PRESUPUESTO

Para finalizar, los costes totales de ejecución material los podemos calcular como suma de costes directos e indirectos como se muestra en la tabla 5.7.

| Costes | Precio (€) |
|---|---|
| Directos | 10524,29 |
| Indirectos | 1578,64 |
| TOTAL | 12102,93 |

Tabla 5.7.- Costes totales de ejecución material.

Sin embargo, estos costes totales no incluyen ni el IVA ni el beneficio industrial, los cuales son porcentajes aplicables al coste total de ejecución material. En la tabla 5.8. se recogen tales resultados. Se ha escogido un 6 % de beneficio tal y como se recoge en el Boletín Oficial del Estado [45].

| Concepto | Coste (€) |
|---|---|
| Total de ejecución material | 12102,93 |
| Beneficio industrial (6 %) | 726,18 |
| IVA (21 %) | 2541,61 |
| TOTAL OBRA | 15370,72 |

Tabla 5.8.- Presupuesto de ejecución material.

Finalmente, se puede decir que el presupuesto total del proyecto es de QUINCE MIL TRESCIENTOS SETENTA EUROS CON SETENTA Y DOS CÉNTIMOS.

# 7. Conclusiones.

Es sabido que el uso de la domótica cada vez tiene más relevancia, no obstante, su lenta evolución en comparación con otras tecnologías se traduce en un precio elevado para este tipo de instalaciones. Aunque esta situación, parece estar revirtiéndose en los últimos años gracias a la aparición de proyectos libres que ponen de manifiesto el interés en facilitar el acceso a más usuarios.

Este proyecto se ha desarrollado en este marco de trabajo, intentando implantar un sistema capaz de informar al usuario e interactuar con él, facilitando las tareas de vigilancia y cuidado del hogar. Todo ello, con un presupuesto competitivo que convierta a la solución dada en una opción viable.

Las herramientas utilizadas a lo largo del diseño, así como las tecnologías empleadas en el mismo siguen, casi en su totalidad, licencias sin copyright con un carácter gratuito y estandarizado que hacen que su expansión, de así quererse, no sea problemática a la vez que se han mantenido los costes reducidos.

A nivel técnico, se han conseguido buenos resultados en cuanto a fiabilidad y eficiencia en el sistema desarrollado. La elección del protocolo Z-Wave junto con la Raspberry Pi, ha sido acertada para llevar a cabo el propósito de implementar un sistema domótico mediante herramientas libres y de bajo coste. El motivo principal es que gracias a la tarjeta RaZberry, que sostiene el protocolo de comunicación nodos-controlador, se puede llevar a cabo este objetivo de manera relativamente cómoda.

Además, el sistema final cuenta con una interfaz web que es capaz de servir tanto datos de los sensores y actuador, como de interactuar con el usuario mediante la auto válvula o de manera que pueda sacar una foto en el instante que lo desee del lugar donde haya colocado la cámara. Ya que, aunque se ha estado usando un cable ethernet para su desarrollo, también se ha configurado Wi-Fi en la Raspberry por medio de un adaptador, lo cual da más libertad a la hora de situar el sistema controlador + cámara.

En cuanto al presupuesto, señalar como el verdadero coste de este proyecto corresponde con el esfuerzo humano realizado. Dicho coste alcanza un 84 % de los costes totales de ejecución material, como se aprecia en la figura 6.1.

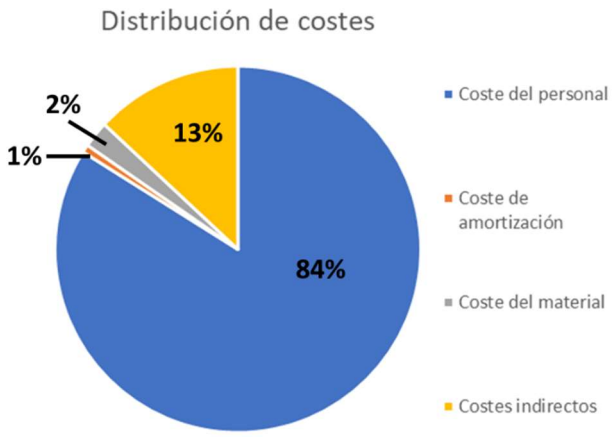


Figura 6.1.- Distribución de costes del proyecto.

Por otra parte, los materiales empleados sí que cumplen el objetivo planteado al principio de conseguir una instalación domótica barata siendo el peso de estos costes de un 2 %.

De cara a las mejoras introducidas en el sistema en su conjunto con respecto al anterior, se pueden destacar, sobre todo, los siguientes puntos:

- Se ha mejorado todo el diseño de la interfaz de usuario, proporcionando la información del sistema más clara y facilitando la navegación a través de ésta. Además, todos los valores que se muestran están debidamente actualizados para que el usuario tenga conocimiento en tiempo real.

- Se ha introducido una página para poder servir imágenes mediante la Raspberry Pi Cámara y se ha decido adjuntar una en el envío del correo al usuario.

- Incluye la automatización del cierre de la auto válvula cuando alguno de los dos sensores detecta agua. Para ello, ha sido necesaria una ejecución en segundo plano de esta pequeña orden, ya que de lo contrario el sistema respondía con retraso.

- La interfaz de usuario ha sido dotada de un nombre de dominio mediante el cual es accesible desde cualquier buscador web en cualquier dispositivo y parte del mundo.

- Se ha simplificado el manejo de todos los datos enviados por los nodos del sistema. Estos valores, se guardan en una misma tabla en la base de datos, indistintamente de que sensor provengan. Simplemente, se ejecuta una orden de consulta más elaborada, evitando trabajar con muchas tablas y agilizando el sistema.

- Se ha añadido una notificación al smartphone del usuario mediante la aplicación Pushbullet.

- Los valores en tiempo real de los nodos del sistema, han sido incluidos en la interfaz para que el usuario tenga conocimiento de ellos.

- Se ha mejorado el envío de los datos por parte de la auto válvula a la interfaz web. Se ha evitado el envío del primer dato de ésta, que hacía referencia al estado del cual la válvula partía y no a su estado final. De esta manera, la base de datos carecerá de valores inútiles y las consultas a ésta serán más fáciles.

Con todo esto, se cree que el resultado del sistema domótico final ha sido capaz de cumplir los objetivos impuestos en un principio, consiguiendo diseñar una solución sencilla de cara al usuario y completa.

# 8. Referencias.

[1] "Automatización industrial, historia." [Online]. Available: https://es.wikipedia.org/wiki/Automatización_industrial. [Accessed: 16-Dec-2017].

[2] "A Brief History of Home Automation." [Online]. Available: https://www.slideshare.net/ProgressiveAutomations/infographic-web. [Accessed: 16-Dec-2017].

[3] "Domótica y su historia." [Online]. Available: https://www.slideshare.net/IsabellaRodriguez22/domtica-y-su-historia. [Accessed: 15-Dec-2017].

[4] "Sistemas domóticos centralizados, descentralizados y distribuidos | Hogartec." [Online]. Available: http://hogartec.es/hogartec2/sistemas-domoticos-centralizados-descentralizados-y-distribuidos/. [Accessed: 31-May-2018].

[5] "DOMOTICA: ARQUITECTURA CENTRALIZADA." [Online]. Available: http://antoniopendolema.blogspot.com/2013/04/arquitectura-centralizada.html. [Accessed: 01-Jul-2018].

[6] "DOMOTICA: ARQUITECTURA DESENTRALIZADA." [Online]. Available: http://antoniopendolema.blogspot.com/2013/04/arquitectura-desentralizada.html. [Accessed: 01-Jul-2018].

[7] "Sistemas de Información: Comparación Sistema Centralizado vs Distribuido." [Online]. Available: http://domotica-ubiobio.blogspot.com/2015/03/comparacion-sistema-centralizado-vs.html. [Accessed: 01-Jul-2018].

[8] "Topología de red: malla, estrella, árbol, bus y anillo - Culturación." [Online]. Available: http://culturacion.com/topologia-de-red-malla-estrella-arbol-bus-y-

anillo/. [Accessed: 31-May-2018].

[9] "Topologia de rede – Wikipédia, a enciclopédia livre." [Online]. Available: https://pt.wikipedia.org/wiki/Topologia_de_rede. [Accessed: 01-Jul-2018].

[10] "Sistemas domóticos existentes, tipos y estándares." [Online]. Available: https://domoticasistemas.com/tienda/tutoriales/1_sistemas-existentes-tipos-y-estandares.html. [Accessed: 02-Jun-2018].

[11] "EnOcean: comunicación inalámbrica a un bajo consumo | Ricardo Vega." [Online]. Available: https://ricveal.com/blog/enocean/. [Accessed: 03-Jun-2018].

[12] "PROTOCOLOS O SISTEMAS MAS IMPORTANTES EN LA ACTUALIDAD | PROTOCOLOS O SISTEMAS MAS IMPORTANTES DE LA DOMÓTICA." [Online]. Available: http://miblognuevoayudaparatodos.blogspot.com/2016/08/protocolos-o-sistemas-mas-importantes.html. [Accessed: 02-Jun-2018].

[13] "Raspberry Pi 2 Model B - Raspberry Pi." [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-2-model-b/. [Accessed: 01-Jul-2018].

[14] "Probamos la nueva Raspberry Pi 2: A fondo." [Online]. Available: https://www.xatakahome.com/trucos-y-bricolaje-smart/probamos-la-nueva-raspberry-pi-2-a-fondo. [Accessed: 03-Jun-2018].

[15] "RaZberry – Z-Wave.Me." [Online]. Available: https://z-wave.me/products/razberry/. [Accessed: 03-Jun-2018].

[16] "Z-Wave.Me Introduces RaZberry Solution - SMAhome." [Online]. Available: https://mysmahome.com/news/3587/z-wave-me-introduces-razberry-solution/.

[Accessed: 01-Jul-2018].

[17] "FIBARO Z-Wave Flood Sensor • Smart Home Automation." [Online]. Available: https://smarthome.hancan.com.au/product/fibaro-z-wave-flood-sensor/. [Accessed: 01-Jul-2018].

[18] "Motor Z-Wave para llave general de Agua/Gas de 1/4 de giro - GR Smart Home." [Online]. Available: http://zwave.es/GR-105. [Accessed: 05-Jun-2018].

[19] "Compre Válvula Automática Z Wave Gr 105 Wireless 868. Control 42mhz Válvula De Gas Natural Y Riego De Jardín Por Control Remoto Del Teléfono A $108.87 Del Wildeer | Dhgate.Com." [Online]. Available: https://m.es.dhgate.com/product/z-wave-auto-valve-gr-105-wireless-868-42mhz/409610011.html. [Accessed: 01-Jul-2018].

[20] "Camera Module V2 - Raspberry Pi." [Online]. Available: https://www.raspberrypi.org/products/camera-module-v2/. [Accessed: 01-Jul-2018].

[21] "Camera Module V2 - Raspberry Pi." [Online]. Available: https://www.raspberrypi.org/products/camera-module-v2/. [Accessed: 04-Jun-2018].

[22] "NetScout Edimax n150 Wi-Fi and Bluetooth USB Adapter | Comms Express." [Online]. Available: https://www.comms-express.com/products/edimax-n150-wi-fi-and-bluetooth-usb-adapter/. [Accessed: 01-Jul-2018].

[23] "EDIMAX - Adaptadores inalámbricos - N150 - 150Mbps Wireless IEEE802.11b/g/n_ nano USB Adapter." [Online]. Available: http://www.edimax.es/edimax/merchandise/merchandise_detail/data/edimax/es/wireless_adapters_n150/ew-7811un/. [Accessed: 05-Jun-2018].

[24] "Download Raspbian for Raspberry Pi." [Online]. Available:

https://www.raspberrypi.org/downloads/raspbian/. [Accessed: 17-Dec-2017].

[25] "Etcher." [Online]. Available: https://etcher.io/. [Accessed: 17-Dec-2017].

[26] "Download PuTTY - a free SSH and telnet client for Windows." [Online]. Available: http://www.putty.org/. [Accessed: 17-Dec-2017].

[27] "Advanced IP Scanner – Explorador de redes de descarga gratuita." [Online]. Available: http://www.advanced-ip-scanner.com/es/. [Accessed: 17-Dec-2017].

[28] "Set up Wifi on Raspberry Pi." [Online]. Available: https://www.lifewire.com/usb-wifi-adapter-raspberry-pi-4058093. [Accessed: 07-Jun-2018].

[29] "RaZberry Download." [Online]. Available: https://razberry.z-wave.me/index.php?id=24. [Accessed: 17-Dec-2017].

[30] "Smart heating with Raspberry Pi + Z-Wave + Eurotronic Spirit - YouTube." [Online]. Available: https://www.youtube.com/watch?v=qi7cHd-W3So. [Accessed: 01-Jul-2018].

[31] "Index of /debian/pool/main/o/openssl." [Online]. Available: http://ftp.debian.org/debian/pool/main/o/openssl/. [Accessed: 05-Jul-2018].

[32] "Install & Configure PHP7, MySQL/MariaDB, phpMyAdmin for PHP7 Web Development on Debian 9 - YouTube." [Online]. Available: https://www.youtube.com/watch?v=EmrWLj2CE00. [Accessed: 18-Dec-2017].

[33] M. Team, "Z-Way Developers Documentation," no. c, p. 107.

[34] "Notepad++ Home." [Online]. Available: https://notepad-plus-plus.org/. [Accessed: 18-Dec-2017].

[35] "server - How to add permission to write to /var/www for my FTP client - Ask Ubuntu." [Online]. Available: https://askubuntu.com/questions/334478/how-to-add-permission-to-write-to-var-www-for-my-ftp-client/334523. [Accessed: 21-Jun-2018].

[36] "PhpMyAdmin - Crear usuario y base de datos." [Online]. Available: http://www.aprendeinformaticaconmigo.com/phpmyadmin-crear-usuario-y-base-de-datos/. [Accessed: 21-Jun-2018].

[37] "PHP Prepared Statements." [Online]. Available: https://www.w3schools.com/PhP/php_mysql_prepared_statements.asp. [Accessed: 18-Dec-2017].

[38] "Tutoriales programación: Herramienta cURL en el interprete de comandos." [Online]. Available: http://programandolo.blogspot.com/2013/08/herramienta-curl-en-el-interprete-de.html. [Accessed: 26-Jun-2018].

[39] "PHP: Ejemplo de curl básico - Manual." [Online]. Available: http://php.net/manual/es/curl.examples-basic.php. [Accessed: 26-Jun-2018].

[40] "Enviar correos electrónicos en Python con SMTP." [Online]. Available: https://code.tutsplus.com/es/tutorials/sending-emails-in-python-with-smtp--cms-29975. [Accessed: 27-Jun-2018].

[41] "W3Schools Online Web Tutorials." [Online]. Available: https://www.w3schools.com/. [Accessed: 28-Jun-2018].

[42] "My No-IP." [Online]. Available: https://my.noip.com/#!/dynamic-dns. [Accessed: 24-Mar-2018].

[43] "Configurar no-ip para Raspberry Pi y de paso, qué es no-ip | RealDroidES."

[Online]. Available: https://www.realdroid.es/2016/10/29/configurar-no-ip-para-raspberry-pi-y-de-paso-que-es-no-ip/. [Accessed: 24-Mar-2018].

[44] "BOE Costes indirectos.pdf." .

[45] Educación and M. Cultura y Deporte, "(BOE) Boletín oficial del estado," pp. 82662–82675, 2012.