Proyecto Fin de Máster
Ingeniería Industrial

Integración de Raspberry en Domótica de Sistema HomeKit

Autor: Ana Rocío Racero Valcárcel
Tutor: Jose María Maestre Torreblanca

Dpto. Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2018

Proyecto Fin de Máster
Ingeniería Industrial

Integración de Raspberry en Domótica de sistema HomeKit

Autor:
Ana Rocío Racero Valcárcel

Tutor:
Jose María Maestre Torreblanca
Profesor titular

Dpto. de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2018

Proyecto Fin de Máster: Integración de Raspberry en Domótica de sistema HomeKit

Autor: Ana Rocío Racero Valcárcel

Tutor: Jose María Maestre Torreblanca

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2018

El Secretario del Tribunal

A mi familia

A mi tutor

Agradecimientos

Quisiera dedicar estas líneas a todas aquellas personas que me han ayudado a llegar hasta aquí.

En mi primer lugar a mi familia, por vuestro apoyo y ayuda incondicional que me habéis proporcionado así como por los buenos consejos que siempre me habéis dado y ser un pilar fundamental en mi vida.

A mis amigas por estar ahí cuando lo he necesitado, y los buenos momentos que me han hecho pasar.

A mi tutor del presente proyecto, Pepe Maestre, por su paciencia, tiempo y dedicación que ha tenido, además gracias por el trato cercano que me has dado ya que me ha hecho afrontar el trabajo con el mejor ánimo en momentos difíciles.

Ana Rocío Racero Valcárcel

Sevilla, 2018

Resumen

El objetivo de este proyecto es la integración de Raspberry en domótica con sistema HomeKit de Apple mediante la aplicación Homebridge. Para ello se ha dividido el proyecto en varios capítulos realizando una secuencia de pasos para llevar a cabo dicha integración.

El primer capítulo consta de una breve introducción a la domótica y sus principales características que están llevando al desarrollo de diversas plataformas para casas domotizadas.

El segundo capítulo hace una investigación del sistema HomeKit de Apple, en el que se incluyen todas las características y funcionamiento del sistema, así como un subcapítulo para desarrolladores.

El tercer capítulo está orientado para la explicación de la Raspberry y la aplicación Homebridge que se usado como herramienta de conexión entre la Raspberry y el sistema HomeKit de Apple para dispositivos domóticos no compatibles.

El cuarto capítulo es una breve introducción a la programación de npm usada para aplicaciones de Homebridge.

En el quinto capítulo se hace una breve guía de instalación de la puesta en marcha de la Raspberry y HomeKit.

Por último, se ha desarrollado el último capítulo con distintas aplicaciones de accesorios compatibles y no compatibles con HomeKit.

Abstract

The scope of this project is the integration of Raspberry in home automation with Apple's HomeKit system through the Homebridge application. For this, the project has been divided into several chapters, making a sequence of steps to carry out this integration.

The first chapter consists of a brief introduction to home automation as its main characteristics that are leading to the development of various platforms for home automation.

The second chapter is an investigation of Apple's HomeKit system, which includes all the features and operation of the system, as well as a subchapter for developers.

The third chapter is oriented to the explanation of the Raspberry and the Homebridge application that was used as a connection tool between the Raspberry and Apple's HomeKit system for non-compatible home automation devices.

The fourth chapter is a brief introduction to the npm programming used for Homebridge applications.

The fifth chapter is a brief installation guide for the startup of the Raspberry and HomeKit.

Finally, the last chapter has been developed with different applications compatible accessories and not compatible with HomeKit.

... -translation by Google-

Índice

ÍNDICE DE FIGURAS

1. INTRODUCCIÓN A LA DOMÓTICA

> Se entiende por domótica el conjunto de sistemas capaces de automatizar una vivienda, aportando servicios de gestión energética, seguridad, bienestar y comunicación, y que pueden estar integrados por medio de redes interiores y exteriores de comunicación, cableadas o inalámbricas, y cuyo control goza de cierta ubicuidad, desde dentro y fuera del hogar. Se podría definir como la integración de la tecnología en el diseño inteligente de un recinto cerrado.
>
> *- Wikipedia -*

La domótica tiene su origen en los años setenta cuando comenzaron a aparecer los primeros dispositivos automatizados en edificios basados en la triunfante tecnología X-10. Fue a partir de entonces cuando este concepto empezó a cobrar vida internacionalmente creciendo un gran interés por la búsqueda de la casa ideal. Los primeros sistemas fueron implantados en Estados Unidos y se limitaban a la regulación de la temperatura ambiente de los edificios de oficinas y poco más.

A finales de los 80, principio de los 90, se empezaron a incorporar en los edificios el cableado necesario para facilitar las conexiones de todo tipo de terminales, usando un cableado estándar y tomas repetidas por todo el edificio. A todos los edificios con ese tipo de cableado se les comenzó a llamar edificios inteligentes.

Más tarde todos estos nuevos sistemas destinados a edificios de oficinas comenzaron a instalarse también en viviendas de particulares donde el número de necesidades que había que cubrir era mayor.

El por qué los dispositivos inteligentes han llegado a su mayoría de edad tiene su explicación en la cantidad de ventajas que presentan hoy en día y que hace 10 años eran impensable. Esto es debido a que hace 10 años controlar un dispositivo inteligente era en base al protocolo X-10, que se comunicaba a través de líneas eléctricas lo que complicaba mucho convertir una casa en una casa inteligente.

La domótica trata de integrar todos los dispositivos y no de dar soluciones aisladas. Las principales claves para el sistema domótico es cubrir necesidades ampliando funcionalidades de los hogares a través de la simplicidad en el uso de dichos dispositivos, con un bajo coste y un adecuado servicio de postventa.

Hoy en día en la domótica nos encontramos dos estándares muy implantados en el mercado llamados X10 y Z.Wave. El primer caso como se ha comentado anteriormente fue el primero que se creó y funcionaba mediante la red eléctrica o radiofrecuencia, enviando órdenes a diferentes dispositivos o accesorios. El segundo funciona a través de una red inalámbrica aunque en frecuencias diferentes a una red WIFI, que transmite órdenes en lo que se denomina una red mallada, donde cada accesorio es emisor y receptor ampliando así la cobertura de comunicación. En secciones posteriores se hará una descripción más detalla de los estándares y protocolos de un sistema domótico.

1.1 Aspectos de la domótica

Los servicios que ofrece la domótica se pueden agrupar según diversos aspectos o ámbitos principales:

1.1.1 Ahorro energético

El ahorro energético es un concepto al que se puede llegar de muchas maneras. En muchos casos no es necesario sustituir los aparatos o sistemas del hogar por otros que consuman menos energía sino una gestión eficiente de los mismos.

Algunos ejemplos son los siguientes:

- Climatización y calderas: Para ello se puede programar y zonificar, utilizando un termostato. Se pueden encender o apagar la caldera usando un control de enchufe, mediante telefonía móvil, fija, Wifi o Ethernet.
- Control de toldos y persianas eléctricas: Realizando algunas funciones repetitivas automáticamente o bien por el usuario o manualmente mediante un mando a distancia. Por ejemplo para proteger el toldo del viento se puede usar un sensor de viento que actúe sobre los toldos cuando sea necesario. O para la protección del sol, mediante otro sensor de sol.
- Gestión eléctrica: Se puede racionalizar el uso de cargas eléctricas mediante la desconexión de equipos de uso no prioritarios en función del consumo eléctrico en un momento dado.
- Gestión de tarifas: Es posible derivar el funcionamiento de algunos aparatos a horas de tarifa reducida.
- Uso de energías renovables.

1.1.2 Confort

Se entiende por confort todos aquellos servicios de la domótica enfocados a obtener el bienestar por medio de ciertos dispositivos que trabajen para nosotros ya sea pasivo, activo o mixto. Algunos aspectos que mejoran nuestro confort en un sistema domótico son:

- Iluminación:
  - ✓ Apagado general de todas las luces de la vivienda.
  - ✓ Activación o desactivación de la iluminación por presencia.
  - ✓ Automatización del apagado/encendido en cada punto de luz.
  - ✓ Atenuación de las luces por franja horaria.
- Automatización de todos los distintos sistemas/instalaciones/dotándolos de control eficiente y de fácil manejo.
- Integración del portero al teléfono, o del videoportero al televisor.

- Control vía Internet
- Gestión Multimedia y del ocio electrónico.
- Generación de macros y programas de forma sencilla para el usuario y automatización.

1.1.3 Seguridad

Un aspecto fundamental de los sistemas domóticos consiste en una red de seguridad encargada de proteger tanto los bienes patrimoniales, como la seguridad personal y la vida.

Algunos ejemplos destacados son:

- Alarmas de intrusión (antiintrusión): Se utilizan para detectar o prevenir la presencia de personas extrañas en una vivienda o edificio.
- Detectores y alarmas de detección de incendios.
- Alerta médica y teleasistencia.
- Acceso a cámaras IP.

1.1.4 Comunicaciones

La domótica no puede ser algo aislado, debe permitir la comunicación hacia el exterior y desde el exterior para avisar tanto de los acontecimientos que sucedan en la casa como para poder controlar las funciones en nuestra ausencia.

Para la trasmisión de la información, interconexión y control entre los elementos del sistema domótico se puede usar el cableado propio, el medio más común, o cableado compartido, haciendo uso de redes existentes o inalámbricas con tecnologías de radiofrecuencia o infrarrojo etc. [1] [2]

1.1 Tipos de redes domóticas

Existen cuatro tipos de redes domóticas: centralizada, descentralizada, mixta e híbrida. [3]

- **Centralizada**

La centralizada tiene una unidad central de control que se encarga de administrar todos los accesorios, ya sean sensores o actuadores conectados a la red domótica.

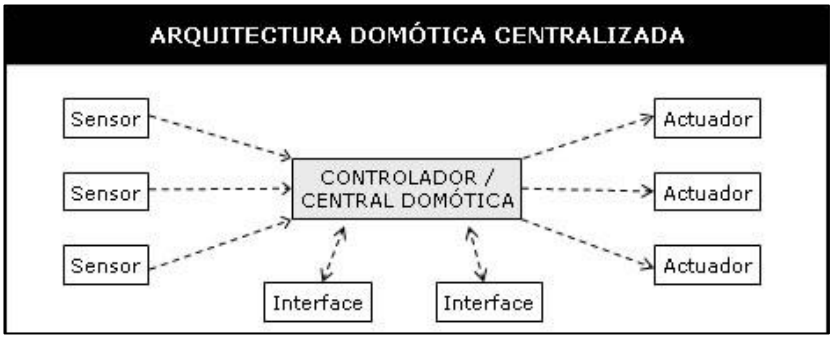

Figura 1. Arquitectura domótica centralizada [4]

- **Descentralizada**

La descentralizada es una variante de la anterior que no necesita unidad central de control porque cada accesorio posee un microcontrolador y cada uno de ellos se une mediante buses actuando de manera independiente.

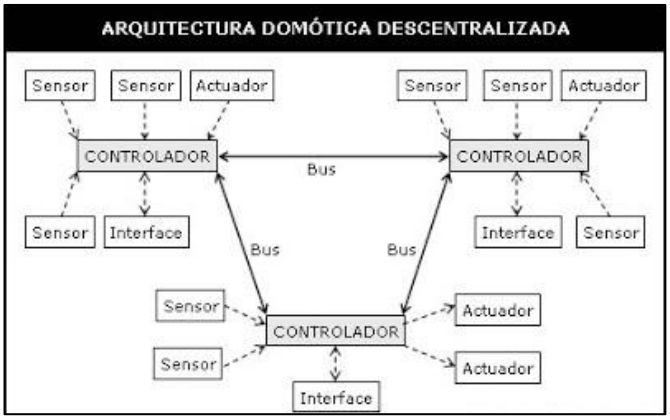

Figura 2. Arquitectura domótica descentralizada [4]

- **Distribuida**

En la arquitectura distribuida los sensores y actuadores actúan como controladores independientes que tienen la capacidad de enviar información y todos los elementos disponen de un acoplador al bus con una interfaz de acceso compartido y cuentan con una inteligencia propia para comunicación bidireccional segura.

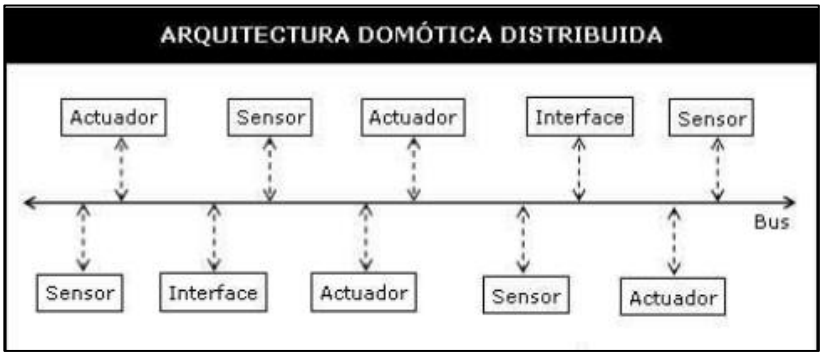

Figura 3. Arquitectura domótica distribuida [4]

- **Híbrida**

La arquitectura híbrida es una combinación de las tres anteriores. Dispone tanto de dispositivos con controladores, controlador central, dispositivos de interfaces, sensores y actuadores para administrar la información tanto recibida como la que se va a enviar.

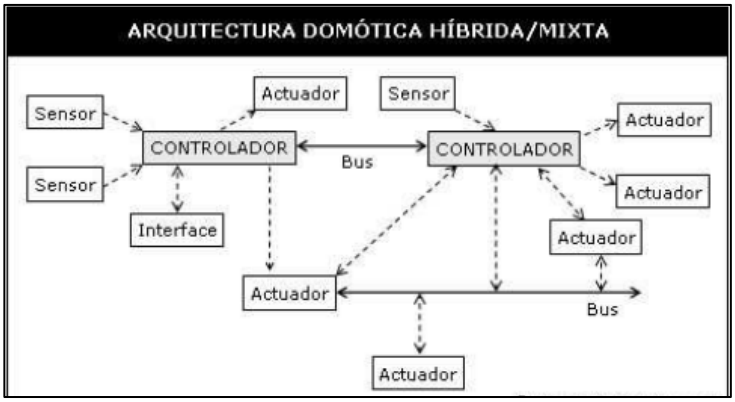

Figura 4. Arquitectura domótica híbrida [4]

Las pasarelas residenciales y el acceso a internet juegan un papel muy importante en un sistema domótico. Esto es debido a que las pasarelas residenciales son las encargadas de adaptar los protocolos y flujos de datos entre redes externas e internas. Así se consigue que varios ordenadores puedan compartir archivos y actuar como cortafuegos adaptando los datos a los protocolos típicos de internet.

El principal objetivo de las pasarelas es la unificación de las todas las redes presentes en una Red doméstica. Tienen la forma de un modem de nueva generación combinando las funciones de router, un hub y modem con acceso a internet. Además garantizan la seguridad de las comunicaciones desde/hacia HAN (Home Area Network) y es gestionable de forma remota. La red HAN se divide en tres tipos de redes; red de control, red de datos y de red multimedia.

1.2 Tecnologías de comunicación

La tecnología en redes domóticas se basan en el uso de protocolos de comunicación y de estándares para la comunicación entre dispositivos. Los más conocidos son los siguientes: [5]

- **X10**: Es un protocolo de comunicaciones para control remoto de dispositivos eléctricos, se usa por ejemplo en enchufes eléctricos sin necesidad de un nuevo cableado. Es de código abierto y el más difundido. Sin embargo, es un protocolo poco fiable frente a ruidos eléctricos, ya que pueden darse interferencias entre ellos.

- **KNX/EIB**: Es un estándar internacional que garantiza interoperabilidad de edificios inteligentes. Es un estándar abierto a nivel mundial para el control tanto de viviendas como de edificios. KNX es un sistema que elimina los problemas que presentan dispositivos aislados dentro de un mismo edificio o vivienda, asegurando que todos los dispositivos hablen un mismo lenguaje común siendo independiente de fabricantes y aplicaciones. Este protocolo tiene también una asociación dedicada a la promoción del mismo.

- **ZigBee**: Es el nombre de la especificación de red mallada de un conjunto de protocolos de alto nivel de comunicación inalámbrica basada en el bajo consumo y recogida en la especificación de radio IEEE 802.15.4. Es usado principalmente en la automatización industrial y en la operación de plantas físicas.

- **OSGi**: Son las siglas de "Open Services Gateway Initiative". Son especificaciones abiertas de software que permiten diseñar plataformas compatibles que puedan proporcionar múltiples servicios. Su origen tuvo lugar para aplicaciones para redes domésticas. Pese a tener su propia arquitectura, ha sido pensada para su compatibilidad con Jini o UPnP.

- **LonWorks**: Este estándar consiste en un conjunto de dispositivos inteligentes que conectan mediante uno o más medios físicos mediante un protocolo común. Tiene una arquitectura basada en nodos programados para enviar mensajes o llevar a cabo acciones de forma autónoma y proactiva.

- **UPnP**: Son las siglas de "Universal Plug and Play". Es un protocolo basado en una arquitectura de software abierta y distribuida que permite la conexión y la comunicación directa de dos o más aplicaciones que se conecten tanto a la red local como a internet. Este protocolo tiene la desventaja que también las aplicaciones maliciosas pueden conectarse al servidor.

- **IEEE**: Son las siglas de "The Institute of Electrical and Electronics Engineers". Es una asociación en el campo de la ingeniería con el objetivo de ayudar al desarrollo de dispositivos electrónicos y eléctricos, permitir su difusión y el uso de estándares que permitan la mayor compatibilidad y más sencillo manejo entre fabricantes.

- **CENELEC**: Son las siglas de "Comité Européen de Normalisation Electrotechnique". Es un comité responsable de la estandarización europea en las áreas de ingeniería eléctrica. Forma parte del sistema europeo de normalizaciones junto con la ETSI y la CEN.

- **CEDOM**: Es la Asociación Española de Domótica cuyo objetivo principal es la promoción de la Domótica. Esta asociación reúne nacionalmente a todos los agentes del sector mediante foros para aumentar la implantación de la domótica y la inmótica a través de la promoción de la tecnología.

1.3 Características de la domótica

Las principales características que han llevado a la domótica este avance en la sociedad son las siguientes: [6]

- ✓ **Activado por voz**

  Los controladores activados por voz son fáciles de usar porque responden a comandos de lenguaje natural como "Siri, apaga la luz de la sala familiar". Algunos ejemplos de controladores inteligentes son los termostatos de control , bombillas o enchufes inteligentes.

- ✓ **Mayor confiabilidad**

  La mayor confiabilidad se debe principalmente a que los dispositivos domóticos cada vez menos se comunican por líneas eléctricas. La mayoría se comunican con los protocolos explicados anteriormente como ZigBee o Z-Wave dejando en el olvido otros como X-10 debido a las ventajas que presentan. Además existen productos como interruptores y salidas de luz que actúan como repetidores de señal, ampliando el alcance de su red. Además, gracias a la comunicación en red mallada cualquier dispositivo es capaz de comunicarse con cualquier otro dispositivo inteligente. Por último, algunos de estos nuevos protocolos no utilizan las mismas frecuencias que los productos de Wi-Fi, por lo que son mucho menos susceptibles a la interferencia inalámbrica.

- ✓ **Accesibilidad**

  La domótica tiene como objetivo principal la accesibilidad para todos y por ello se basa en el concepto de diseño para todos, este concepto es un movimiento que procura satisfacer las necesidades humanas creando diseños accesibles para toda la diversidad humana. El objetivo de esta tecnología es beneficiar mejorando la autonomía personal ya sean destinatarios dependientes o independientes.

- ✓ **Fácil de instalar**

  La fácil instalación proviene que la mayoría de dispositivos inteligentes funcionan con baterías, por lo que no necesitan un adaptador de CA ni conexión a la alimentación de pared. Incluso informan su nivel de batería al concentrador inteligente, por lo que sabrá cuándo es el momento de comprar una batería nueva. Además en su mayoría funcionan de forma inalámbrica por lo que se está prescindiendo ya del cableado para sistemas domóticos.

- ✓ **Sin teclados de seguridad ni alarmas fuertes**

  La casa inteligente de hoy puede programarse para automatizarse después de que cada miembro de la familia se haya ido. Cuando se active la alarma el propietario recibe un mensaje indicando que la alarma se ha activado, y si tiene cámaras instaladas verás que ha causado que se activara la alarma y todo ello con la mejor ventaja de tener todos estos servicios sin coste adicional de una tarifa mensual.

- ✓ **Comunicación en la nube**

  Hoy en día muchos productos se comunican a través de la nube. Un ejemplo de ello es a través de los servicios web IFTTT que pueden realizar acciones condiciones en los dispositivos de tu casa si la marca está asociada con la aplicación. La nube también es capaz de decirle a sus dispositivos inteligentes cuándo ocurre el ocaso todos los días haciendo que sea fácil encender las luces en el momento perfecto durante todo el año.

- ✓ **Apelación masiva**

  Los productos de hogares inteligentes ya no son solo para determinadas marcas. Existen muchísimas marcas que comercializan hoy productos domóticos. Estas compañías también han hecho un buen trabajo explicando cómo estos dispositivos funcionan con otros dispositivos domésticos inteligentes, por lo que han preparado la bomba para otros productos inteligentes para el hogar. En un futuro cercano, es probable que millones de consumidores adicionales creen sus propias casas inteligentes.

- ✓ **Los dispositivos inteligentes hacen más**

  Cada vez los dispositivos inteligentes van teniendo más funcionalidad inmersa en un solo dispositivo. Las salidas inteligentes miden el uso de energía y actúan como repetidores de señal. Algunos termostatos inteligentes también detectan el movimiento, por lo que pueden apagar automáticamente el aire acondicionado cuando estás lejos. Algunos dispositivos inteligentes inalámbricos también informan la temperatura, por lo que podrían detectar un incendio en un área de su hogar que no tiene detector de humo.

- ✓ **Mejor software y servicios:**

  Los primeros Smart Hubs solo funcionaban con software de una sola compañía. Los controladores inteligentes actuales funcionan con aplicaciones, complementos y servicios de una amplia gama de desarrolladores diferentes. Hay aplicaciones disponibles que hacen una amplia gama de cosas, si no puede encontrar una aplicación que haga lo que desea, puede crear la suya o modificar las aplicaciones existentes . La mejor parte es que no necesita un programador experimentado para hacer esto.

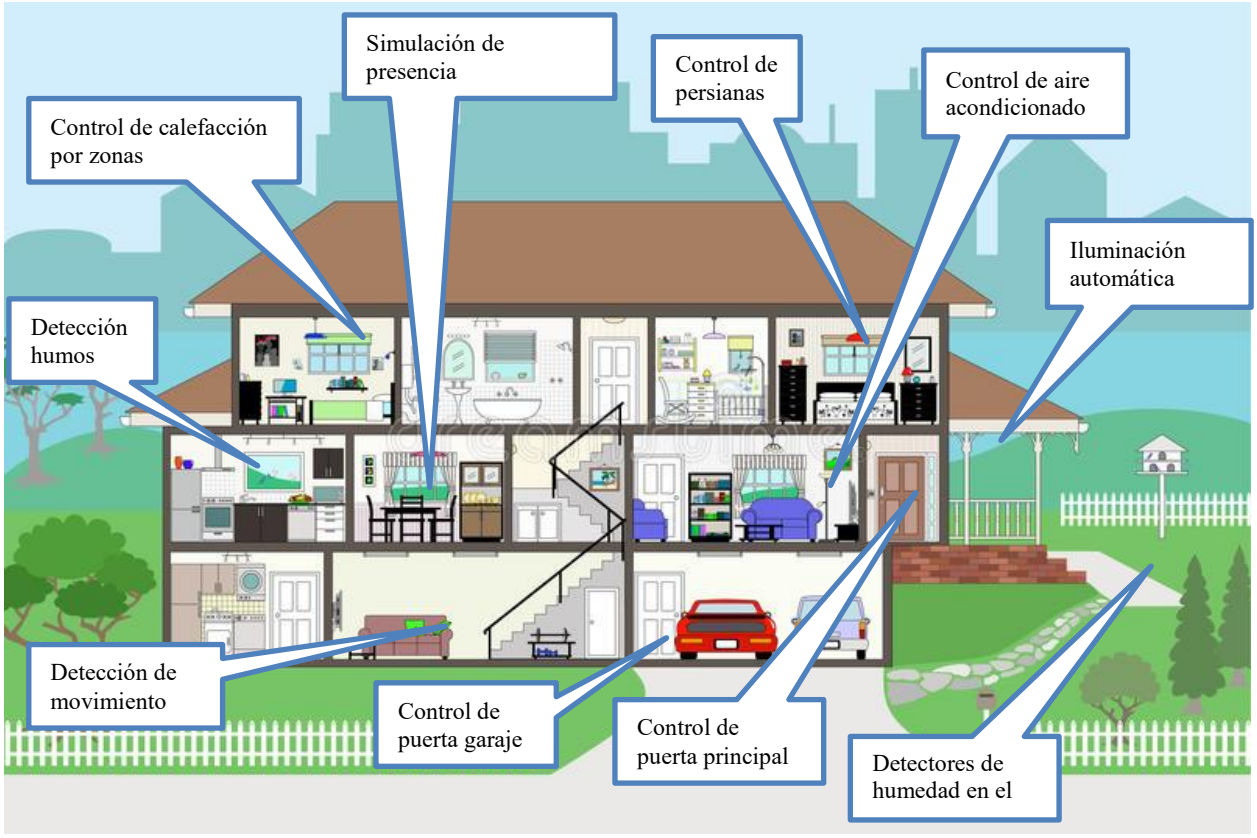

Figura 5. Ejemplo de vivienda automatizada

En relación a una casa inteligente, hoy en día no hay límites, y existe multitud de funcionalidades que cada vez se van desarrollando más. Algunas de las ventajas en cuanto a funcionalidades más significativas de un sistema domótico son las siguientes: [6]

- Hacer que el aire acondicionado suba automáticamente el termostato para que no se encienda después de que todos los inquilinos de la vivienda la hayan abandonado.
- Configura el altavoz para reproducir el sonido de un perro ladrando fuerte cuando hay movimiento y no haya nadie en la casa.
- Hacer que el dispositivo de control abra automáticamente las persianas de una ventana y desactive la alarma cuando el usuario se levante por la mañana.
- Recibir un mensaje de texto después de que el correo haya llegado.
- Usar el teléfono para iniciar un modo de "Buenos días" que ajusta el termostato, enciende una cafetera e inicia la lista de reproducción de música favorita, o un modo "Adiós", que abre la puerta del garaje, enciende el termostato, apaga todas las luces y bloquea las puertas.
- Recibir una notificación en la pantalla del coche si hay un intruso, un incendio o una fuga de agua en la vivienda.
- Usar un sensor para controlar la humedad en el suelo y solo encenderá los rociadores cuando sea necesario.
- Usar el smartphone para controlar de forma remota cualquier dispositivo inteligente del hogar.

- Encender un ventilador de techo después de que la temperatura se eleve por encima de un determinado valor especificado por el usuario.
- Cambiar el color de tu iluminación en función de diferentes eventos. Por ejemplo:
    - Si hay una fuga de agua, enciende las luces azules
    - Si hay una alarma de humo, enciende las luces de color rojo
    - Si hay un incendio, enciende todas las luces de mi casa
- Grabar 10 segundos de video antes del inicio de un evento, para que pueda ver qué activó una alarma. Por ejemplo, si alguien golpeara la puerta de la vivienda, vería la puerta abrirse y la cara de la persona al entrar, no solo la parte de atrás de su cabeza.

Sin embargo, pese a todas las ventajas que posee un sistema domótico no podemos dejar en el olvido uno de sus inconvenientes principales que existe: la seguridad. Muchos expertos en seguridad se preocupan de que los piratas informáticos puedan tener acceso a configuraciones de hogares inteligentes no segura, lo que podría ser meramente un inconveniente puede convertirse en algunos casos en un peligro real. Y es que se ha probado que hasta para sistemas que usan tecnología de encriptación, pueden suministrar información a terceros.

Una de las medidas a tomar para prevenir riesgos de ciberataques es activar la encriptación de la red inalámbrica doméstica y desactivar el acceso por internet a los controles de los dispositivos domóticos instalados en la vivienda.

Esto es debido a que IoT (Internet of Thing) va a convertirse en un punto de ataque principal para los piratas principales ya que es nuevo y despierta mucho interés, además amplía de forma masiva el alcance al que pueden acceder los hackers, porque ya no solo alcanzan los ordenadores o móviles sino en todos los dispositivos que nos rodean. El Internet de las Cosas va a hacer que cualquier aparato, por sencillo que parezca pueda estar conectado a Internet y eso requiere un plus de seguridad.

2. HOMEKIT

HomeKit es el nombre del marco de automatización del hogar de Apple para los desarrolladores. Apple presentó HomeKit como parte de iOS 8 en septiembre de 2014. Con HomeKit, los iPhones y iPads tendrán una forma racional de configurar, comunicarse y controlar "el Internet de las cosas" que nos rodean, incluidas las luces, altavoces, sistemas de seguridad, dispositivos conectados y muchos más accesorios. Tanto a nivel local cuando estás en casa como a distancia cuando estás lejos, a través de aplicaciones y a través de Siri, el asistente personal virtual de Apple. [7]

Figura 6. Domótica con HomeKit (Apple) [8]

HomeKit es un marco para comunicarse y controlar los accesorios automatizados que existen en los hogares y que son compatibles con el protocolo de 'protocolo de accesorios HomeKit' de Apple. Las aplicaciones HomeKit permiten a los usuarios añadir accesorios compatibles y configurarlos. Los usuarios también pueden crear acciones para controlar los accesorios (como un termostato o luz), agruparlos y activarlos utilizando Siri. Los dispositivos HomeKit se almacenan en una base de datos que reside en el dispositivo iOS del usuario, que se sincroniza a través de iCloud con otros dispositivos iOS.

La utilidad de HomeKit radica en que muchos accesorios funcionaban con iPhone o con aplicaciones a tercero, pero sin HomeKit, esos accesorios tenían que crear soluciones personalizadas que a menudo eran incompatibles otros accesorios, con diversos grados de seguridad. Esto significa que las luces inteligentes no pueden comunicarse con su termostato, ni ambos pueden conectarse a un servicio unificado que lo controla todo a la vez. Los accesorios de HomeKit no solo pueden comunicarse entre sí, sino que también pueden comunicarse con todos ellos mediante Siri. [9]

Localmente, las aplicaciones HomeKit solo se pueden usar en primer plano. Eso asegura que las personas puedan ver exactamente lo que está sucediendo y cuándo, y nunca tener que preocuparse de que algo ocurra en secreto en segundo plano. La única excepción a esto son los desencadenantes, que le dan a iOS la capacidad de activar un conjunto de acciones. Sin embargo, deben ser configurados expresamente por el usuario para hacerlo.

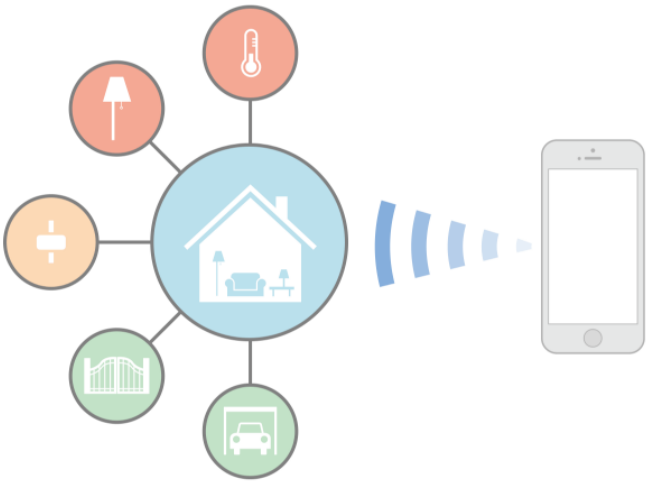

Figura 7. HomeKit & Sistema Domótico [10]

HomeKit se basa en un “Home Manager” y una base de datos común, almacenada en iOS, que contiene toda la información sobre el hogar, sus habitaciones, los accesorios dentro de ellos, y los servicios y características de esos accesorios. El Home Manager proporciona sus punteros a la base de datos común, a la que se accede a través de los hogares. La facilidad de tener todo almacenado en un solo lugar permite una experiencia más uniforme en todas las aplicaciones. Así que, por ejemplo, lo que haya configurado en la aplicación de control de luz, será lo mismo que en la aplicación de control de speaker con el mismo nombre que tenga la casa, el mismo nombre de las habitaciones y los mismos nombre de accesorios.

Home Manager, como su nombre lo indica, permite administrar “Hogares”, incluida la designación de un hogar principal si tiene más de uno. Cada hogar debe tener un nombre único para que pueda especificar a cuál se refiere, incluso a través de Siri.

La parte “Inicio” se compone de "Habitaciones". Las habitaciones también deben tener nombres únicos, pero solo dentro de sus Hogares, por ejemplo, se puede tener "Dormitorio principal" y "Dormitorio de vacaciones". Una vez más, eso es para que pueda referirse a ellos específicamente, y también lo puede hacer Siri.

Las habitaciones se pueden agrupar en "Zonas". Estos podrían incluir, por ejemplo, "Arriba" y "Abajo". Se puede agrupar cualquier cantidad de salas en una zona, y la misma sala puede existir en múltiples zonas. Sin embargo, las zonas también necesitan nombres únicos dentro del hogar, para el usuario y para Siri.

Las habitaciones son las que contienen los "Accesorios". Los accesorios son los dispositivos físicos específicos conectados al iPhone o iPad: básculas, altavoces, cerraduras, luces, enchufes etc. Los accesorios también necesitan nombres únicos dentro de una casa, por lo que el usuario o Siri pueden acceder específicamente a ellos. Por lo tanto una manera de ser designado puede ser "Luces del dormitorio principal" y "Altavoces del dormitorio principal". [7]

HomeKit es una base de datos común a todo el sistema y que cualquier app puede consultar, escribir o programar por lo que no se almacena ningún dato en ningún lugar externo, solo en caso de usarlo, en el Apple Watch donde dicha base de datos permanece en modo espejo con la del iPhone.

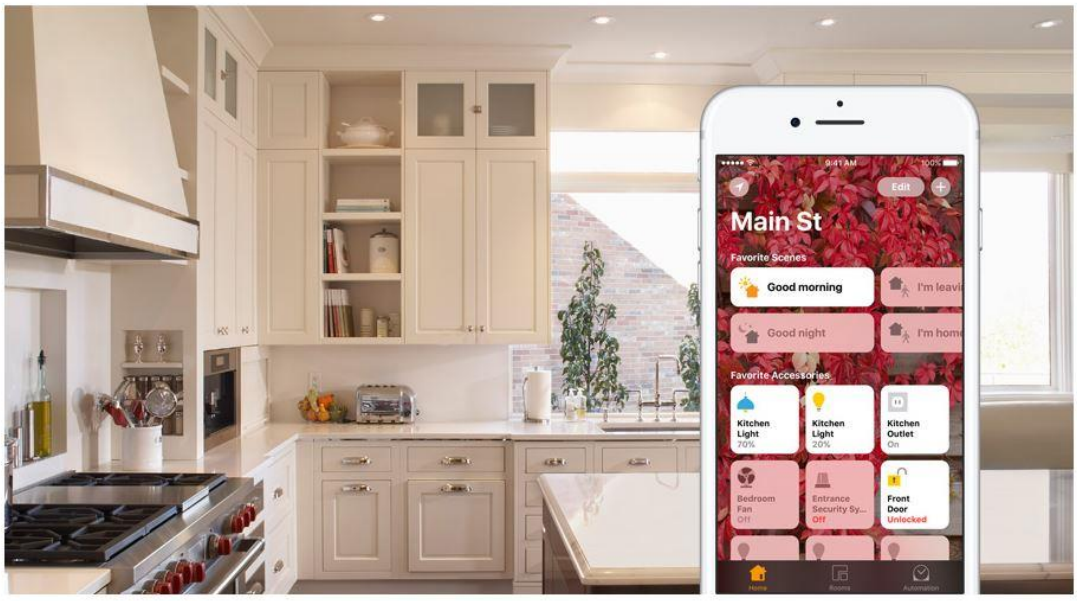

Figura 8. App Home [11]

Actualmente estos son los accesorios que se pueden adquirir según el tipo de producto que se necesite para la casa:

Tipos de productos:

- Alarmas y sensores, calefacción y aires acondicionados, Luces e interruptores, Cámara de video, enchufes, cerraduras…

Cabe destacar que existen más de 50 marcas que comercializan productos compatibles con HomeKit. Algunos de los más comunes se citan a continuación:

- D. Link, Koogek, Elgato, Fibaro, Honeywell, LYFX, Logitech, Nanoleaf, Netatmo …

En estas marcas la más utilizada y accesible es Elgato, pero recientemente ha cambiado de nombre por Eve ya que se ha producido la venta de toda su división de accesorios gamer a Corsair y quedándose únicamente con accesorios de domótica compatibles con HomeKit. En cuanto al ahorro económico destaca la marca Koogek, con la que se ha desarrollado un apartado del presente proyecto. [12]

2.1. Servicios y características

Los accesorios tienen "Servicios". Los servicios representan lo que un accesorio puede hacer, éstos pueden o no tener nombre dependiendo si están destinados a ser utilizados mediante dispositivo o si son los que se accede a través de Siri que necesitarán un nombre único para el hogar al igual que los accesorios en sí. Por ejemplo, una bombilla que se enciende es un servicio que necesita un nombre. Otros servicios incluyen abridores de puertas de garaje, cerraduras de puertas, termostatos, cámaras IP, interruptores y servicios personalizados.

Los servicios se pueden agrupar en "Grupos de servicios". Estos pueden incluir, por ejemplo, "Luces nocturnas" que incluyen luces de la habitación, luces del abre puertas de garaje, luces exteriores y luces de electrodomésticos, o "Altavoces de fiesta" que canalizan audio por toda la casa. Los grupos de servicio pueden incluir cualquier cantidad de servicios de cualquier cantidad de accesorios diferentes, y están destinados a facilitar el control de servicios específicos en una amplia gama de accesorios. El mismo servicio se puede incluir en cualquier cantidad de grupos diferentes, por lo que la misma luz puede aparecer en "Luces nocturnas" y "Horas de juego", pero cada grupo de servicio necesita un nombre único por hogar para el usuario y Siri.

Los servicios también tienen "Características". Las características son la parte interactiva de los servicios, por ejemplo, si una bombilla está encendida o apagada (el estado de encendido) es una característica. No son nombrados, pero Siri los reconoce porque Apple ha definido ciertos tipos, como estado de energía, estado de bloqueo, estado objetivo, brillo, número de modelo, temperatura actual y características personalizadas.

Algunas características son de solo lectura, como la temperatura actual sin embargo, otros son de lectura y escritura, como pedir y restablecer la temperatura y algunos son de solo escritura, como comandos. Por lo tanto, por ejemplo, se puede ordenar a cualquier accesorio que se "identifique" y parpadeará, emitirá un pitido o, de lo contrario, le mostrará o le dirá qué y dónde se encuentra.

Para ayudar a los desarrolladores y fabricantes a pensar fuera de los ajustes preestablecidos, HomeKit permite definir servicios y características personalizadas. Siri no los entiende de forma nativa por la forma en que son definidos los servicios y las características de Apple, pero permite una funcionalidad potencialmente mucho mayor y más diversa.

2.2. Acciones y disparadores

Las acciones cambian las características. Se consideran tipos de acciones, por ejemplo, cierre el garaje, bloquee las puertas, apague las luces, baje la temperatura, entre otros.

Los conjuntos de acciones son conjuntos de operaciones que se ejecutan juntas. Por ejemplo, "Buenas noches" podría asegurar que la puerta de su casa esté cerrada, que las luces de la casa estén apagadas, que el televisor esté apagado y que la cafetera esté lista para cuando el usuario se despierte por la mañana. El "tiempo de juego" podría asegurar que las luces estén en rojo, el sistema de sonido esté en máximo y todo lo demás en la casa esté apagado o silenciado. Los conjuntos de acciones no se ejecutan en un orden específico, todos ocurren tan pronto como pueden, de una vez si es posible. Una vez más, un conjunto de acción necesita un nombre único por hogar para el usuario y Siri.

Los disparadores (triggers) ejecutan conjuntos de acciones en fechas y horas predefinidas. Pueden ser de uso único o pueden configurarse para repetirse y además, pueden tener retrasos incorporados. Los triggers no se pueden utilizar a través de Siri, pero pueden ser ejecutados por iOS en segundo plano a diferencia del resto de HomeKit, y también requieren nombres únicos por hogar.

En conjunto, los conjuntos de acciones y triggers le permiten crear una secuencia de comandos para automatizar el control de todos y cada uno de sus accesorios compatibles con HomeKit. [7]

2.3. Agregar accesorios a Home Kit

Debido a que HomeKit es un framework para desarrolladores y no una aplicación, no hay Home.app para HomeKit. De esta forma, hay Health.app para HealthKit, Photos.app para PhotoKit, Passbook.app para PassKit, etc., cualquier aplicación que se vincule con HomeKit tiene que estar lista y ser capaz de ayudar al usuario a administrar sus accesorios.

Es por ello que si se inicia una aplicación HomeKit y no se detecta "Casa", la aplicación tiene que guiar al usuario a crear y nombrarla, luego crear y nombrar las habitaciones, y luego proporcionar el accesorio para que el usuario pueda encontrarlo y lo agregue accesorios a la casa, posteriormente asignarles un nombre y ubicarlo en la habitación adecuada. HomeKit también puede informar a cualquier aplicación si un accesorio es accesible o no, por ejemplo fuera de rango, fuera de línea, apagado, etc.

Pero para añadir nuevos accesorios a HomeKit se ha de tener en cuenta ciertas reglas para nombrar los accesorios para que sean reconocidos por Siri. A continuación se muestra un listado de las reglas a seguir:

- Los nombres deben ser únicos dentro de su espacio de nombres.
- Los nombres de las casas que pertenecen a un usuario están en un espacio de nombre.
- Un nombre solo puede contener caracteres alfanuméricos, espaciales y de apóstrofo.
- Un nombre debe comenzar y terminar con un carácter alfabético o numérico.
- Los caracteres de espacio y apóstrofes se ignoran en las comparaciones (por ejemplo, home1y home 1 son lo mismo).
- Un nombre no distingue entre mayúsculas y minúsculas.

Pero no todos los accesorios son compatibles con HomeKit, es por ello por lo que se ha desarrollado un accesorio puente denominado Homebridge, el cual hace de enlace entre accesorios incompatibles con HomeKit y una Raspberry. Una vez se agregan los accesorios a Homebridge mediante la programación de la Raspberry, Homebridge se encargará de traducir entre HomeKit y el formato que utilicen cada dispositivo incompatible como se explicarán en capítulos posteriores.

2.4. Accesibilidad

La accesibilidad , también conocida como inclusión, tiene que ver con hacer que el iPhone, el iPod Touch y el iPad funcionen para el mayor número posible de personas. Eso puede incluir personas muy jóvenes, muy viejas, personas principiantes en ordenadores y dispositivos móviles, y también personas con discapacidades y necesidades especiales. Con iOS, Apple ha agregado funciones para ayudar específicamente a las personas con discapacidad visual, incluida la ceguera, el daltonismo y la baja visión, con deficiencias auditivas que incluyen sordera en uno o ambos oídos, con impedimentos de habilidades motoras o físicas, incluyendo coordinación limitada o rango de movimiento y desafíos de aprendizaje, incluyendo autismo y dislexia. También incluye características generales, como Siri y FaceTime que pueden proporcionar un valor significativo para los ciegos o los sordos. Muchas de estas características se pueden encontrar en Configuración, todas se pueden encontrar en el iPhone e iPad. [13]

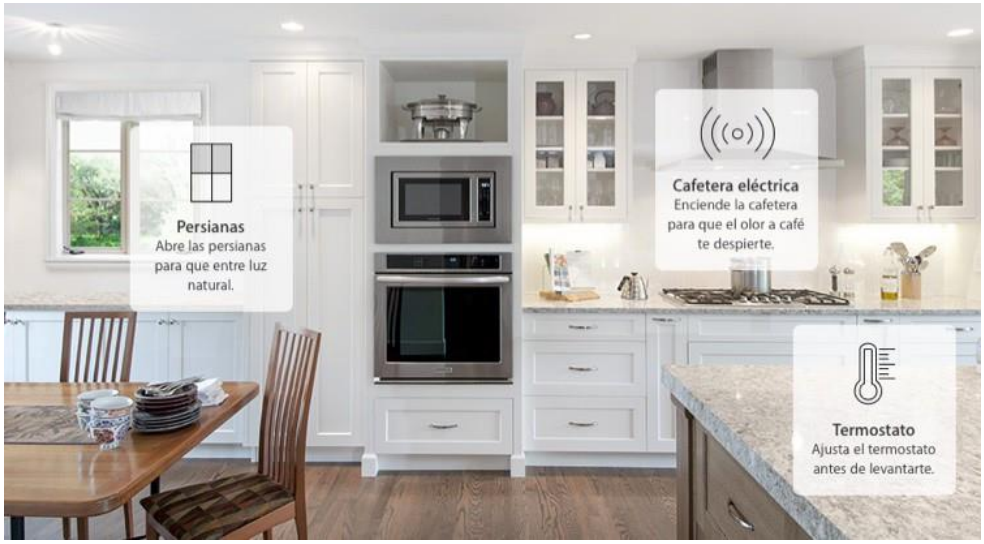

Figura 9: Ejemplo de tipo de automatización en una vivienda [14]

2.5. HomeKit para desarrolladores

Apple ha permitido que los desarrolladores tengan la posibilidad de ampliar el contenido y funcionalidad de sus aplicaciones. Tanto para HomeKit como otras aplicaciones como Cloukit, Healthkiy etc. Apple está permitiendo que los desarrolladores puedan crear sus propias experiencias con la creación de más de 4000 nuevas API que permiten nuevas capacidades y características.

Apple ha facilitado que los desarrolladores trabajen con HomeKit al construir un simulador de accesorios de kits domésticos directamente en Xcode 6. Actúa como un accesorio real y permite a los desarrolladores probar aplicaciones como si estuvieran conectadas a un accesorio real.

El simulador de accesorios HomeKit no viene preinstalado con Xcode, el usuario debe descargarlo e instalarlo. Para ello se deben seguir las siguientes instrucciones.

En primer lugar debe iniciar sesión en su cuenta en developer.apple.com e ir a la sección de Descargas. En la parte inferior de la página, hay un enlace que dice "No ve lo que está buscando, consulte más descargas". Hacer clic en ese enlace y expandir el enlace etiquetado como "Herramientas adicionales para Xcode 9". Aquí se puede encontrar un enlace a un archivo DMG llamado "Herramientas adicionales para Xcode 9.dmg". Descargar este dmg, montarlo y copiar la carpeta "Herramientas de Xcode" en la carpeta de Aplicaciones. En la carpeta "Hardware" de "Herramientas de Xcode", se puede encontrar el simulador de accesorios HomeKit. La ejecución de esta aplicación le abrirá una ventana con el entorno para la programación de HomeKit. Por último usar el menú Archivo para crear tantos Accesorios o Puentes como desee y ya podrá comenzar a programar. [15]

2.1.1 Configuración para desarrolladores de HomeKit

Apple tiene la seguridad como una de sus preferencias por lo que recomienda proteger la privacidad del usuario, por ello en una aplicación de iOS 10.0 o superior y que accede a los datos de configuración de HomeKit el usuario debe declarar estáticamente la intención de hacerlo. Para realizar esa acción es necesario incluir la clave *NSHmeKitUsageDescription* en el archivo de la aplicación y proporcionar una cadena para esta clave. Así cuando su aplicación intenta cambiar los datos de configuración de HomeKit del usuario sin esa cadena correspondiente, la aplicación se cierra. [16]

Al igual que para usuario de la aplicación desde el dispositivo Apple tiene los elementos explicados anteriormente, el desarrollador posee la misma jerarquía básica de contención, siendo posible la programación de cada uno de ellos como desee siempre y cuando se atienda a los requisitos de Apple: [16]

- Las casas (HMHome) son el contenedor de nivel superior y representan una estructura que un usuario generalmente consideraría como una sola casa, sin embargo, los usuarios pueden tener varias casas que están muy separadas, como una casa principal y una casa de vacaciones. Las casas son el objeto central de la organización para HomeKit, sirven como el principal punto de acceso para comunicarse y configurar accesorios.

  Los hogares no se crean directamente, sino que se crea con addHome(withName:completionHandler:)HMHomeManager .

  Para la configuración de inicio se tienen los siguientes comandos:

  - class HMHome: Un hogar y sus accesorios.
  - class HMHomeManager: El gerente de una colección de una o más casas.
  - class HMUser: Una persona en el hogar que puede tener acceso a los accesorios y servicios de control en el hogar.
  - class HMHomeAccessControl: Los privilegios de acceso de un usuario asociado a una casa.
  - class HMAccessControl: Una superclase abstracta para acceder a los privilegios del usuario.

- Las habitaciones (HMRoom) son partes opcionales de las casas y representan habitaciones individuales en el hogar. Las habitaciones no tienen características físicas como tamaño, ubicación, etc. Simplemente son nombres que son significativos para el usuario, como "sala de estar" o "cocina". Los nombres significativos de las habitaciones permiten comandos como, "Siri, enciende las luces de la cocina".

- Las zonas (HMZone) son agrupaciones opcionales de habitaciones en un hogar, por ejemplo, "arriba" y "abajo" estarían representados por zonas. Las zonas son completamente opcionales: las habitaciones no necesitan estar en una zona, pero es útil para agregar salas a una zona, el usuario puede dar órdenes a Siri, como, "Siri, enciende todas las luces de la planta baja".

  Para el diseño de zonas y habitaciones se tienen los siguientes comandos:

  - class HMZone: Una colección de habitaciones que los usuarios generalmente consideran una sola área.

  - class HMRoom: Una habitación en una casa.

- Los accesorios (HMAccessory) se instalan en los hogares y se asignan a las salas y son los dispositivos físicos de automatización del hogar, como un abridor de puerta de garaje. En el caso que el usuario no configure ninguna habitación, HomeKit le asigna accesorios a una habitación predeterminada especial para el hogar.

  Para configurar los accesorios y las características se tienen los siguientes comandos:

  - class HMAccessoryBrowser: Un navegador de red utilizado para descubrir nuevos accesorios.

  - protocol HMAccessoryBrowserDelegate: Un conjunto de métodos utilizados para notificar a un navegador accesorio delegado de nuevos accesorios.

  - class HMAccessory: Un accesorio de domótica en el hogar, como un abridor de puerta de garaje o un termostato.

  - class HMAccessoryProfile: Un perfil implementado por un accesorio.

  - protocol HMAccessoryDelegate: Un conjunto de métodos que define el método de comunicación para las actualizaciones de estado de los accesorios a sus delegados.

  - class HMAccessoryCategory: La categoría principal para un accesorio HomeKit.

  - class HMCharacteristic: Una característica específica de un servicio, por ejemplo, si una luz está encendida o apagada, o a qué temperatura está configurado un termostato.

- Los servicios (HMService) son los servicios reales proporcionados por un accesorio. Los accesorios tienen servicios controlados por el usuario, como una luz, y servicios que son para su propio uso, como un servicio de actualización de firmware. HomeKit está más preocupado por los servicios controlables por el usuario.

  Un único accesorio puede tener más de un servicio controlable por el usuario. Por ejemplo, la mayoría de los abridores de puertas de garaje tienen un servicio para abrir y cerrar la puerta, y otro servicio para la luz en el abre-puertas de garaje.

Los comandos para programar los servicios son los siguientes:

- class HMService: Un servicio provisto por un accesorio.
- class HMServiceGroup: Una colección de servicios accesorios.

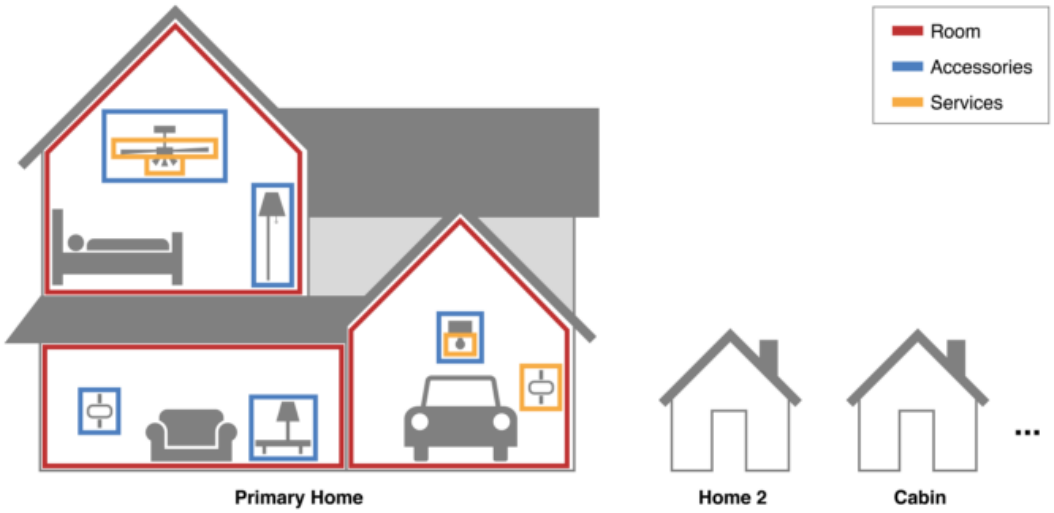

Figura 10. Distribución Home [10]

Apple recomienda una serie de pautas a tener en cuenta para los desarrolladores de APIs para HomeKit [10]:

- Hacer que la configuración de los accesorios sea rápida, intuitiva y automática siempre que sea posible. Lo ideal es que la aplicación detecte los accesorios lo más rápido posible.
- Indicar a los usuarios que asignen los accesorios a habitaciones una vez instalados y posteriormente sugerir la incorporación a una habitación y si no existe, dar la opción de crearla.
- Los accesorios deben ser de fácil identificación, y consecutivamente dar a conocer la manera de cada uno de ellos se diferencien.
- Las personas quieren interactuar con los accesorios de forma rápida por lo que hay que proporcionar múltiples formas de ubicar accesorios.
- Es importante que el usuario este informado de los estados de los accesorios con precisión y ayude al usuario a resolver problemas por lo que si la aplicación no puede acceder a un accesorio no debe inventarse el estado, sino sugerir por ejemplo que el accesorio no está disponible en ese momento.

2.6. Seguridad y privacidad

Apple tiene como mayor objetivo la seguridad y la privacidad. Es por ello que existe un cifrado de extremo a extremo entre los dispositivos conectados y todos los accesorios compatibles con HomeKit.

Cada usuario de HomeKit genera dos claves, una pública y otra privada, Ed25519, en su acceso a la aplicación, dichas claves son las que garantizan la identidad y la seguridad de la aplicación iOS. Esas dos claves también son usadas para para autenticar dispositivos iOS entre ellos mismos y, entre accesorios y dispositivos iOS. Si el accesorio se restaura con la configuración original de fábrica, se genera un par de claves nuevas. [17]

Las claves mencionadas anteriormente se recogen en el llavero y solo se incluyen en él los respaldos encriptados del llavero, utilizando el llavero iCloud para la sincronización entre dispositivos.

El fabricante del accesorio proporciona un código de 8 dígitos que junto con el uso del protocolo de contraseña remota segura se establece una relación entre un dispositivo iOS y un accesorio de HomeKit. Durante la configuración se verificará la certificación MFi del accesorio. [17]

Mediante el programa MFi (Made for iPhone) de Apple se asegura que los fabricantes cumplan una serie de medidas de seguridad muy específicas. Algo importante pues hace que las comunicaciones entre usuario y dispositivo sean cifradas de punto a punto. Evitando así, que un tercero pueda interceptar, falsificar las comunicaciones o robar datos.

El programa MFi funciona de tal forma que cada producto que una empresa quiera certificar con Apple, ésta tiene que verificar todas las interfaces de programación de aplicaciones (API) de un prototipo inicial además de asegurarse de que la aplicación de terceros cumple con los estrictos requisitos de la App Store. Cada accesorio certificado por el programa MFi debe posee la etiqueta que aparece en la siguiente imagen:

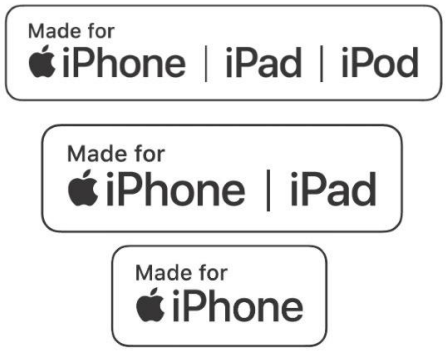

Figura 11. Logo certificación MFI [18]

Por suerte, Apple decidió cambiar las reglas de HomeKit con iOS 11.3. A partir de esta actualización, Apple habilitaba la autenticación de accesorios mediante software, eliminando el requerimiento de un chip que se especificaba en el programa MFi específico para ello.

La encriptación con claves derivadas de la identidad de HomeKit es la base fundamental de la seguridad del almacenamiento local de datos. Además, los datos de HomeKit se almacenan con la clase de protección de datos "Protegido hasta la primera autenticación de usuario". Los datos de HomeKit sólo se incluyen en respaldos encriptados descartando otros que no los tengan como por ejemplos iTunes. [17]

En cuanto a la configuración de la seguridad de HomeKit para el usuario tenemos dos requisitos relacionados: la autentificación de dos factores, para el ID de Apple e iCloud Keychain. Este tipo de autentificación es un ritual de seguridad que te obliga a obtener un código en un dispositivo confiable, por ejemplo en un iPhone cada vez que inicias sesión en el ID Apple en otro dispositivo. Es una solución imperfecta porque a veces suele ocasionar bucles entre los dispositivos, o puede que se hayan introducidos las credenciales y el dispositivo simplemente se quede colgando en el limbo esperando confirmación. El otro paso es habilitar iCloud Keychain, que es una función de Apple basada en la nube para almacenar de forma segura contraseñas y otros datos de seguridad. Estas dos funciones conjuntas hacen que HomeKit almacene y encripte los códigos de su dispositivo en iCloud Kaychain y mantengan la información de manera segura.

Tal vez no ahora, pero sí de cara a un futuro cercano, estos dispositivos domésticos cada vez estarán más presentes y por tanto, todo dispositivo compatible con HomeKit ofrecerá mayor seguridad y privacidad.

En el entorno de HomeKit entra en juego el uso de Apple TV con la versión de sistema 7.0 o superior para el control remoto y automatización de los accesorios instalados en HomeKit, lo que se conoce como HomeKit Hub. Un Hub puede ser tanto un Apple Tv como cualquier dispositivo iOS que nunca salga de casa. Esto es así porque es la solución que implementó Apple para mantener ese nivel de seguridad cuando conectamos desde fuera de casa siendo algo que garantiza la encriptación y sirve como obstáculo frente a usos y usuarios no autorizados. Si lo tenemos, sólo habrá que iniciar sesión en ambos productos con el mismo Apple ID. Si todo está correcto podremos acceder a ellos allá donde estemos.

Pero a partir de iOS 9 Apple ha incorporado una tecnología tunneling que permite, a través de iCloud, conectar directamente con los dispositivos desde fuera de casa que estos estarán unidos también a la nube de Apple. Incluso en este caso la comunicación es encriptada punto a punto y Apple no puede saber de ninguna forma que dato está pasando por su sistema.

HomeKit usa iCloud para compartir sus emparejamientos y configuraciones, lo que significa que hay que iniciar sesión con la misma cuenta de iCloud en tu HomeKit Hub que con la que se ha iniciado sesión en el dispositivo que se va a sacar fuera de casa.

El acceso remoto a iCloud está planteado curiosamente de forma que Apple no pueda determinar de qué accesorio se trata o que información o notificación se está enviando mientras que los accesorios pueden controlarse y seguir enviando las notificaciones. HomeKit no remite información de la casa a través del acceso remoto a iCloud.

El acceso remoto a iCloud es mediante el cual se autentican mutuamente dispositivos iOS cuando un usuario envía un comando y los datos se encriptan mediante el mismo procedimiento descrito para conexiones locales. Apple por tanto cuando esos contenidos son encriptados ya no puede verlo y los identificadores de iCloud registrados durante el proceso de configuración están basados en el direccionamiento iCloud.

Figura 12. Presentación Home + iCloud [19]

El proceso que siguen los accesorios compatibles con HomeKit a través del acceso remoto tiene su origen en la configuración inicial de los accesorios. Posteriormente el transcurso de envío de datos comienza cuando el usuario inicia sesión en iCloud.

A continuación, el accesorio debe firmar un reto mediante el coprocesador de autenticación integrado en todos los accesorios diseñados para HomeKit que es solicitado por el dispositivo iOS. El accesorio también genera claves de curva elíptica prime256v1, y la clave pública se envía al dispositivo iOS junto con el reto firmado y el certificado X.509 del coprocesador de autenticación. Desde el servidor de iCloud se solicita mediante las claves anteriores un certificado. El certificado se recoge en el accesorio, pero no contiene ninguna información de identificación sobre el mismo, salvo el hecho de que tiene permitido acceder al acceso remoto a iCloud de HomeKit. También existe un depósito en el cual se envían datos de información de URL y otras informaciones necesarias para conectarse al servidor remoto, pero esta información no es específica para ningún usuario y va alternándose para mejorar así la seguridad de encriptación de los dispositivos. [17]

Existen dos elementos que presenta un accesorio cuando se conecta al servidor de acceso remoto HomeKit, su certificado y una tarjeta. La tarjeta se obtiene de otro servidor de iCloud y no es única para cada accesorio. Dicha tarjeta solicitada por un accesorio incluye su fabricante, modelo y versión de firmware en la solicitud. Esta solicitud no contiene información de identificación del usuario ni de la casa. [17]

Los accesorios se conectan al servidor de acceso remoto a iCloud a través de HTTP/2, asegurado mediante TLS 1.2 con AES-128-GCM y SHA-256. El accesorio mantiene abierta la conexión al servidor de acceso remoto a iCloud, de manera que pueda recibir mensajes entrantes y enviar respuestas y notificaciones salientes a los dispositivos iOS. [17]

Touch ID sin duda también jugará un papel importante en la casa inteligente de la manzana, ya que junto con iCloud forman una forma segura de acceso a los dispositivos Apple y por tanto a nuestro centro de control domótico Apple.

2.7. Base de datos de HomeKit

Existe una base de datos HomeKit por hogar. Las bases de datos están sincronizadas de manera segura con los dispositivos iOS de un usuario y con los dispositivos iOS de invitados que se les ha permitido el acceso a la casa, tal como se puede ver con la siguiente figura. Estas bases de datos contienen la información más reciente para que cada usuario pueda observar los cambios que se producen.

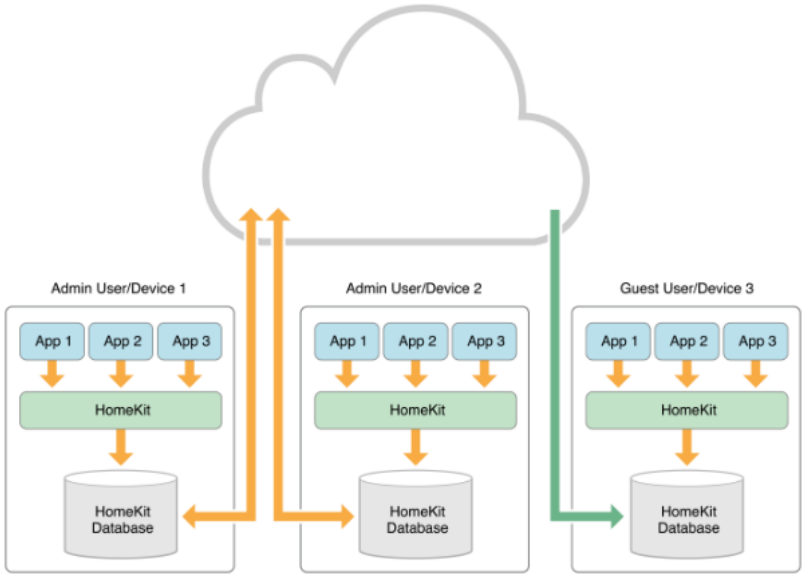

Figura 13. Arquitectura de la base de datos HomeKit [20]

Siri tiene acceso a esta base de datos permitiéndole dar órdenes como "Oye Siri, enciende las luces del salón", y más atractivo aún, da lugar a crear grupos de accesorios, servicios y comandos para llevar a cabo operaciones realmente elaboradas de un modo muy simple y con control por voz. [20]

2.8. Métodos de delegación HomeKit

La delegación es un patrón simple y poderoso en el que un objeto en un programa actúa en nombre de, o en coordinación con otro objeto. El objeto delegado guarda una referencia al otro objeto, el delegado, y en el momento apropiado le envía un mensaje. El mensaje informa al delegado de un evento que el objeto delegado está por realizar o que acaba de gestionar. El delegado puede responder al mensaje actualizando la apariencia o el estado de sí mismo u otros objetos en la aplicación, y en algunos casos puede devolver un valor que afecte cómo se va a realizar un evento inminente. El principal valor de la delegación es que le permite personalizar fácilmente el comportamiento de varios objetos en un objeto central. [21]

HomeKit usa el patrón de diseño de la delegación para notificar a su aplicación los cambios de los accesorios de HomeKit. En general, si su aplicación invoca una determinada acción con cambios en los estados de los accesorios y esto es llevado a cabo con éxito, el mensaje de delegado es enviado a otras aplicaciones HomeKit que se ejecutan en los mismos o en dispositivos iOS remotos. Las aplicaciones incluso pueden ser ejecutadas por usuarios invitados en sus dispositivos iOS. Si la aplicación inicia un cambio, el mensaje de delegado no se envía a su aplicación y por lo tanto, es necesario agregar un código tanto al controlador de finalización como al método de delegado asociado para volver a cargar los datos y actualizar las vistas según sea necesario. Si el diseño de una casa cambia significativamente, hay que volver a cargar toda la información sobre esa casa. En el caso del controlador de finalización, se debe verificar si el método es exitoso antes de actualizar la aplicación. HomeKit también invoca métodos delegados para notificar a su aplicación los cambios en el estado de la red doméstica. [21]

A continuación se muestra un ejemplo del funcionamiento de la delegación de HomeKit, si (1) en respuesta a una acción del usuario, (2) su aplicación invoca *addRoomWithName:completionHadler* y no ocurre ningún error, (3) el controlador de finalización debe (4) actualizar las vistas de la página de inicio. Si tiene éxito, HomeKit (5) envía el home:*didAddRoom:mensaje* a los delegados del hogar en otras aplicaciones. Por lo tanto, su implementación del *home:didAddRoom:metodo* también debería (6) actualizar las vistas de la casa. [21]

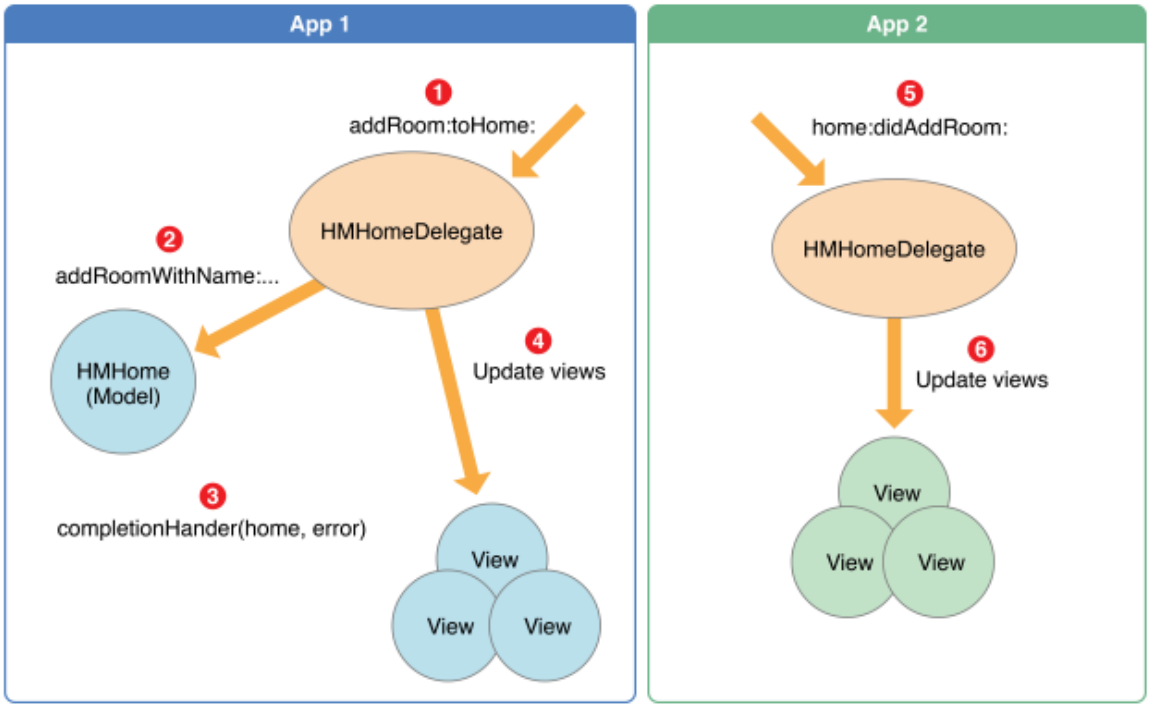

Figura 14. Método de delegación HomeKit [21]

Las aplicaciones deben estar en primer plano para recibir estos mensajes de delegado. Los cambios no se procesan mientras la aplicación está en segundo plano. Si la aplicación está en segundo plano mientras que otra aplicación agrega con éxito una habitación a un hogar, la aplicación no recibirá *un home:didAddRoom:mensaje*. Cuando la aplicación aparece en primer plano, la aplicación recibe un *homeManagerDidUpdateHomes:mensaje* que indica a la aplicación que vuelva a cargar todos sus datos. [20]

2.9. Protocolo de accesorios

La especificación del protocolo de accesorios HomeKit (versión no comercial) se puede descargar desde la página del desarrollador de HomeKit Apple. La especificación HAP es un PDF de 256 páginas con toda la información necesaria sobre la creación de accesorios habilitados para HomeKit que pueden comunicarse con dispositivos Apple. Existen ciertos requisitos que debe cumplir el accesorio para utilizar el marco HomeKit, pero no son tan rigurosos como los requisitos para accesorios comerciales HomeKit.

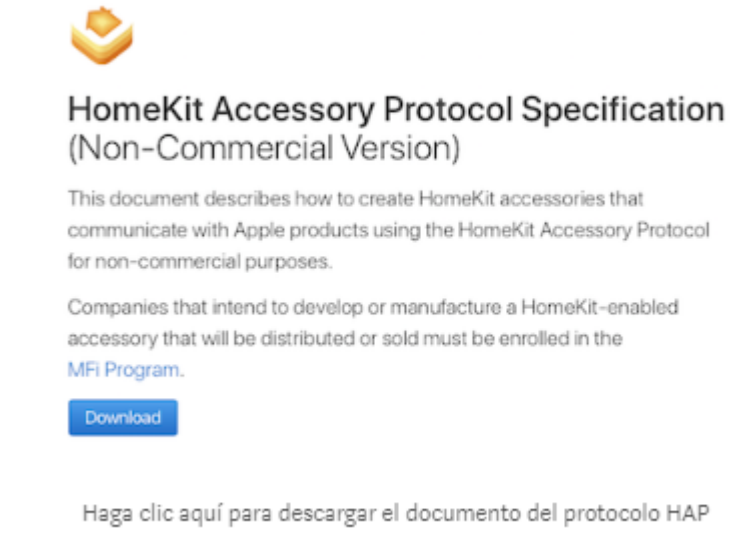

Figura 15. Protocolo Accesorios HomeKit. [22]

A diferencia de un accesorio HomeKit no comercial, un accesorio HomeKit que será distribuido o vendido debe incorporar el Coprocesador de Autenticación Apple, obtener la certificación Wi-Fi Alliance o la certificación Bluetooth SIG dependiendo del transporte utilizado, y completar la certificación HomeKit bajo el Programa MFi. A nivel de usuario, un accesorio no comercial utilizará un proceso diferente para incorporar un accesorio basado en IP a la red, y presentará un cuadro de diálogo de advertencia en iOS que el usuario debe confirmar antes de continuar.

Además del uso de Hap los accesorios HomeKit basados en IP deben implementar Bonjour para permitir el descubrimiento del accesorio.

Bonjour, también conocida como red de configuración cero, permite la detección automática de dispositivos y servicios en una red local que usan protocolos IP estándar. Facilita la publicación y resolución de servicios de red con una interfaz de programación sofisticada y fácil de usar. Bonjour no solo se basa en estándares abiertos de internet, la implementación también está disponible como código abierto bajo la licencia de Apache 2.0. [23]

La especificación del protocolo de accesorios HomeKit ha sido abierta recientemente por Apple a todos los desarrolladores. La especificación del protocolo de accesorios HomeKit es el medio por el cual los accesorios para el hogar conectados se comunican con los dispositivos de Apple a través de HomeKit. En otras palabras, puede hacer sus propios accesorios habilitados para HomeKit y controlarlos con Siri y la aplicación Home para iOS.

Hasta ahora, solo las compañías que tenían la intención de distribuir sus accesorios habilitados para HomeKit comercialmente podían hacer uso de la especificación del protocolo de accesorios HomeKit de Apple. Los desarrolladores tenían que solicitar licencias a través del programa MFi de Apple y someterse a rigurosas pruebas para reclamar la codiciada insignia Works with Apple HomeKit. Ahora sin embargo, si quieres crear accesorios habilitados para HomeKit para fines no comerciales puedes hacerlo.

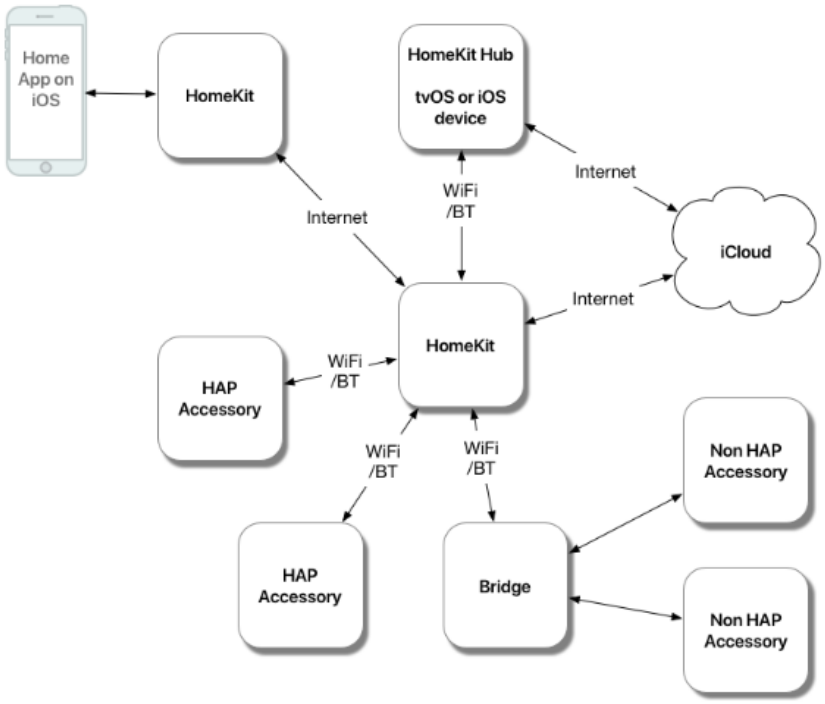

Figura 16. Esquema protocolo HomeKit [22]

HomeKit Accessory Protocol (HAP) es el protocolo patentado de Apple que permite que los accesorios de terceros en el hogar, por ejemplo, luces, termostatos y cerraduras de las puertas y los productos de Apple se comuniquen entre sí. HAP admite dos transportes, IP y Bluetooth LE. La información provista en la Especificación del Protocolo Accesorio HomeKit (Versión no comercial) describe cómo implementar HAP en un accesorio que el usuario crea para uso no comercial y que no será distribuido o vendido. [22]

HomeKit habla únicamente HomeKit Accessory Protocol (HAP), que se ejecuta sobre un BLE / Bluetooth Smart o una pila HTTP / TCP / IP. Si un accesorio no es compatible con HAP directamente, se necesita una puerta de enlace para usarlo en HomeKit. [24]

HomeKit administra la colección de accesorios en un hogar definido, y permite interactuar los servicios provistos por esos accesorios usando aplicaciones iOS o Siri. HomeKit se encarga de la detección y configuración de accesorios / puentes, utilizando HAP para comunicarse con esos dispositivos accesorios y puertas de enlace.

Apple también ha desarrollado otro protocolo que permite la transmisión inalámbrica entre dispositivos audios, videos, pantallas de dispositivos y fotos denominado AirPlay. Originalmente se llamaba AirTunes porque solo se admitía el audio digital, pero se le cambió el nombre a AirPlay cuando se agregaron más características.

AirPlay se compone de un conjunto patentado de protocolos que le permite usar el ordenador Mac o dispositivo móvil iOS para transmitir medios a través de una red wifi. Pero Apple ha continuado su evolución y recientemente ha llegado el Airplay2 que ofrece además un sistema multiroom de forma que podemos reproducir el mismo tema en diferentes equipos y habitaciones o un tema en cada altavoz. Otra de las nuevas ventajas es la posibilidad de transmitir de forma inalámbrica el audio del televisor a diferentes altavoces al mismo tiempo.

Airplay2 ha llegado completamente integrado con HomeKit, por lo que aparecerán los dispositivos compatibles que se tengan en la app Home, y además viene pensado en la compatibilidad con distintas marcas y dispositivos más allá de los propios fabricados por Apple.

Así, los fabricantes que lo deseen podrán actualizar sus equipos para hacerlos compatibles con Airplay 2 o bien lanzar nuevos productos que vengan ya preparados desde el primer momento dispuestos a interactuar entre sí. Algunas de las marcas que ya han señalado su disposición a usar el nuevo protocolo son Bang & Olufsen, Bluesound, Bose, Bowers & Wilkins, Denon, Libratone, Marantz, Marshall, Naim, Pioneer o Sonos, por lo que parece que la nueva versión del protocolo nos dará mucho juego en los próximos años. [25]

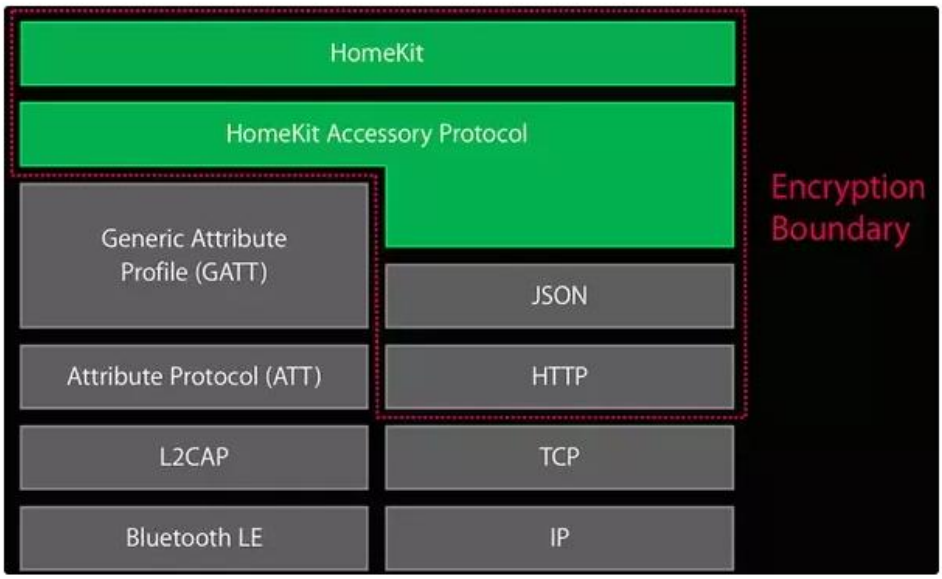

Figura 17. Jerarquía encriptación HomeKit [24]

2.10. Inconvenientes HomeKit

El mayor inconveniente que presenta HomeKit es la distancia a central de un elemento, ya que puede hacer que todo vaya bien o que nada funcione. Para Home existe la aplicación HomeScan para ver la señal de Bleutooth que se recibe de cada accesorio en el dispositivo Apple. Esta aplicación es muy útil por ejemplo para saber si un sensor de puertas de llegada tiene alcance hasta la central de HomeKit donde se encuentra ubicado.

Pero no todas las marcas usan la misma forma de conectarse, la marca Koogeek opta por una conexión wifi de 2,4 GHz, más estable y sin problema de alcance pero no apta para aparatos que funcionan a pilas. Philips usa su propio protocolo pero necesita que se añada puentes para conectarlo con HomeKit y otras marcas usan Bleutooth, como elGato, que también es muy bueno para bajos consumos en dispositivos que funciona con pilas.

Uno de los grandes inconvenientes de las casas inteligentes es la red Bleutooth, usada por la mayoría de dispositivos con sistemas de batería de casa inteligentes. Bleutooth a veces puede causar retrasos o esperas, y el diseño o el tamaño de algunas casas pueden requerir que exista más un dispositivo Apple Tv como centro de control. HomeKit ha disminuido en cierta medida este inconveniente tomando medidas para mejorar el rendimiento al incluir el último estándar Bluetooth 5.0, en el Apple Tv 4K, sin embargo sigue siendo el eslabón más débil al que deben seguir enfrentándose.

Otro de los principales inconvenientes es que los accesorios instalados en ciertas ocasiones se muestran como no disponible cierto tiempo y luego se resuelven por si solos y vuelven a estar disponibles. Esto puede ser debido a la red wifi utilizada en la vivienda por lo que será necesario la reconfiguración del mismo para que se pueda tener una casa domotizada funcionando correctamente.

2.11. Futuro de HomeKit

Sin embargo el problema de la incompatibilidad de HomeKit tiene los días contados. Apple ha entrado recientemente en “The Thread Group”, un grupo de empresas que tratan de estandarizar y mejorar la comunicación y el funcionamiento de dispositivos inteligentes en viviendas. Esto nos da una sospecha de los esfuerzos que Apple está realizando con HomeKit. Google con Nest o Amazon con Alexa ya están dentro de Thread, por lo que pronto se podrá controlar Alexa desde HomeKit sin ningún inconveniente.

The Thread Group ofrece las siguientes características:

- Instalación simple utilizando un Smartphone, ordenador o Tablet.
- Compatible con más de 250 accesorios ahora mismo.
- Seguridad en la red y el protocolo gracias al cifrado AES.

3. RASPBERRY

3.1. Introducción

Una Raspberry es un ordenador del tamaño de una tarjeta de crédito originalmente diseñada para la educación pero que con el paso del tiempo está teniendo cada vez más utilidades.

Los comienzos de la Raspberry se remontan al año 2006, con los primeros diseños basados en el microcontrolador Atmel ATmega 644. En el año 2009 fue creada la fundación Raspberry en Reino Unido en la Universidad de Cambridge como una asociación de caridad, regulada por comisión de caridad de Inglaterra y Gales. [26]

El objetivo del creador Eben Upton era crear un dispositivo de bajo costo que mejorara las habilidades de programación y comprensión del hardware a niveles preuniversitarios. Pero gracias a su pequeño tamaño y precio accesible, ha sido adoptado por fabricantes de electrónica para proyectos que requieren más que un microcontrolador para llevarlos a cabo. El primer prototipo patentado por Eben se montó en un módulo del mismo tamaño que una memoria USB, tenía un puerto USB en un extremo y un puerto HDMI en el otro.

Figura 18. Prototipo Raspberry Pi

El siguiente paso en la historia de la Raspberry fue la fabricación de 50 placas alfa en Agosto de 2011, y fue entonces cuando se eligió definitivamente el logotipo oficial de la fundación. Después de las placas alfa comenzaron a salir los primero modelos A, con 128 MB de memoria RAM, aunque desde sus primeras ventas hasta el lanzamiento oficial pudieron aumentar su capacidad hasta 256MB. Este primer modelo únicamente tenía un puerto USB. Fue el modelo B con dos puertos USB y un puerto para Ethernet el que tuvo mayor acogida en el mercado de ventas.

Ambos modelos primitivos tenían un SOC Broadcom BCM2835, chip gráfico VideoCore IV y procesador ARM11 de un núcleo a 700 MHz, aunque podía hacerse hasta 1000 MHz. También contienen salidas de video RCA, HDMI y DSI para panel LCD, mientras que para salidas de audios tiene un conector Jack de 3.5 mm. Además contiene 8 GPIO que da lugar a una ventaja entre dispositivos electrónicos.

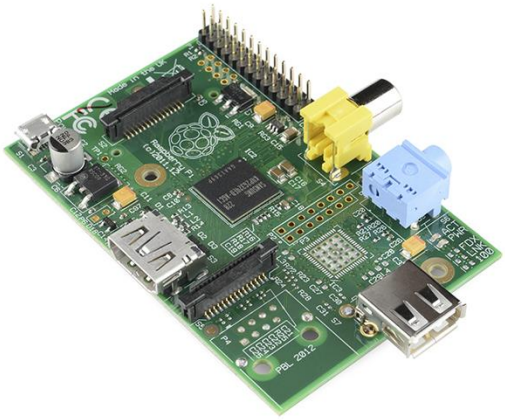

Figura 19. Raspberry pi I Modelo A [26]

Figura 20. Raspberry pi I Modelo B [26]

Debido a la gran acogida de estos dispositivos electrónicos se siguió aumentando sus funcionalidades y se creó la Raspberry pi II. La principal variación con respecto al último modelo anterior fue la potencia del motor de cálculo que se aumentó en gran medida. Estos modelos tienen instalados un SOC Broadcom BDM2836, un procesador ARM Cortex A7 de cuatro núcleos a 900 MHz y un 1 GB de RAM. Sin embargo el chip gráfico se mantuvo el mismo que anterior. También aumentaron el número de puerto USB hasta 4 y el número de pines GPIO a 17.

Figura 21. Raspberry pi II model B [27]

Por último tenemos la Raspberry pi III cuyo principal objetivo era un cambio en la conectividad del dispositivo y por ello se incluyó el Bluetooth 4.1 y wifi. El procesador se mejoró con un SOC Broadcom BCM287, y un procesador ARMv8 de cuatro núcleos a 1.2 GHz de 64 bits. Mientras que todas las demás características se mantuvieron iguales.

Figura 22. Raspberry pi III model B [27]

Por otra parte existen los modelos Zero de Raspberry, estos tienen las funcionalidades de la Raspberry pi I explicadas anteriormente pero destacan por su pequeño tamaño y bajo coste.

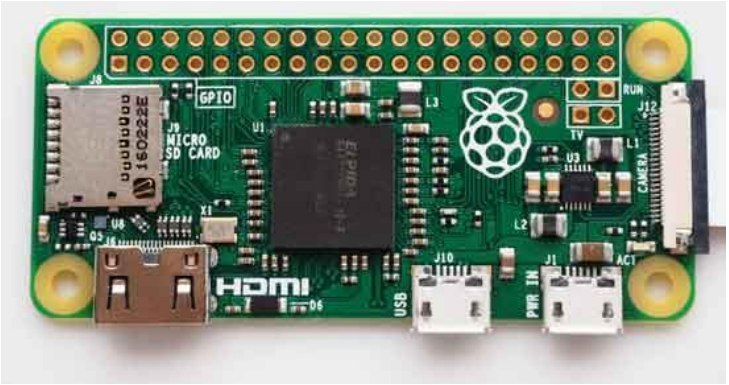

Figura 23. Raspberry pi Zero [27]

La Raspberry es más lenta que un ordenador pero aun así es un ordenador completo que puede ejercer todas las funcionalidades de un ordenador Linux con un bajo consumo de energía.

El hardware es abierto, a excepción del chip principal, el Broadcomm SoC (System On a Chip), que ejecuta muchos de los componentes principales de la placa: CPU, gráfico, memoria, controlador USB...etc. Muchos de los proyectos hechos con la Raspberry también están abiertos y bien documentados, y son cosas que puede modificar cualquier persona.

3.2. Sistemas operativos

El sistema operativo para el que fue diseñada la Raspberry fue Linux pero sin embargo, la opción más popular es Raspbian, del sistema operativo Debian y Pidora, que se basa en el sistema operativo Fedora. No obstante hay otras opciones de sistemas operativos. A continuación se mencionan alguno de ellos. [28]

- Raspbian

Raspbian es un sistema operativo libre basado en la distribución de Linux denominada Debian y optimizado para el hardware de una Raspberry. Los desarrolladores Mike Thomponso y Peter Green fueron los publicadores de una primera versión de la portabilidad de Debian. Desde Junio de 2012 Raspbian se ha convertido en el sistema operativo por excelencia de la Raspberry debido al continuo desarrollo, y esto ha conllevado a un aumento en el repositorio.

Figura 24. Logo Raspbian [29]

- Kali Linux

Kali Linux es otro sucesor de Debian. Sus creadores fueron Mati Ahoni y Devon Kearns, trabajadores de la compañía Offensive Security. Se publicó con el objetivo de que fuera usado para fines de pruebas de seguridad y penetración de sistemas informáticos y redes pero también puede usarse como software habitual. Debido a su centralización en la seguridad, este sistema operativo realiza actualizaciones de seguridad y software desde el repositorio de Debian.

Figura 25. Logo Kali Linux [30]

- Pidora

Pidora es una combinación de Fedora, creada por el "Centre Development of Open Tecnology". Tiene dos elementos diferenciales que son un paquete estándar del proyecto Fedora y aplicaciones modificadas y escritas de nuevo. La principal funcionalidad de Pidora es el llamado *mode headless*, que permite acceder al miniordenador sin tener que usar una pantalla.

Figura 26. Logo Pidora [28]

- Windows 10 IoT Core

En 2015 Microsoft lanzó su propio Sistema Operativo para los nuevos dispositivos del Internet de las cosas. Este sistema tiene su fundamento en "Universal Windows Platform" que permite escribir aplicaciones para dispositivos propios. Está orientada principalmente para desarrolladores y aficionados a la programación que quieren conectar aparatos del día a día con internet. Un aspecto a favor es que pueden controlarse dispositivos como electromotores gracias a la modulación por ancho de pulsos. Tiene descarga y uso gratuito pero no permite el cambio en el núcleo del sistema.

Figura 27. Logo Window 10 IoT Core [31]

- Ubuntu Core

Es una de las distribuciones Linux más sonadas. Este sistema operativo se puede usar tanto en un ordenador personal como en un servidor. Es un software basado en Debian en 2004 y sus principales características son su elevada adaptabilidad y usabilidad. Destaca también porque cada paquete de software representa una unidad individual, subsanando así las posibles brechas de seguridad en actualizaciones.

Figura 28. Logo Ubuntu Core [32]

- RISC OS

Primitivamente denominado Arthur, es un sistema operativo desarrollado para el ordenador Arquímedes basado en arquitectura ARM. Fue creado por Castle Tecnology y se trata de un sistema operativo que cuesta dinero y necesita una licencia de pago para su utilización. El software no es usual ya que usa la técnica del drag and drop (arrastrar y soltar). Los archivos no se abren directamente si no que se arrastran hasta la ventana del programa para que se abran.

Figura 29. Logo RISC OS Pi [33]

- RetroPie

Fue llevado a cabo por defecto sobre Raspbian, este sistema operativo transforma la Raspberry Pi en una consola para jugar a distintos juegos. El software RetroArch ofrece la posibilidad de tener acceso a la API Libretro, mediante la cual se configuran diferentes emuladores. Con Kodi, esta distribución de Raspberry proporciona la utilidad de ver películas o escuchar músicas como centro multimedia.

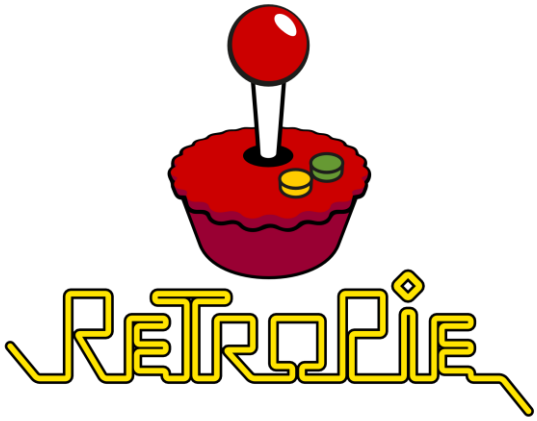

Figura 30. Logo Retropie [34]

- Arch Linux ARM

Este sistema operativo apareció en 2009 en la portabilidad de distribución de Linux para procesadores ARM. Tiene varias versiones que abarcan desde 2012 hasta 2015. Arch Linux ARM proporciona al usuario un control total y responsabilidad sobre el sistema basado en una arquitectura básica y además realiza una actualización continua del sistema denominado ciclo de rolling release, esto significa que, en lugar de lanzar grandes actualizaciones en un determinado momento, los desarrolladores van publicando pequeñas actualizaciones regularmente.

Figura 31. Logo Arch Linux ARM [35]

3.3. Homebridge

Homebridge es un servidor de código abierto diseñado por Nick Farina, que aprovecha una implementación de Node.js del servidor de accesorios HomeKit por Khaos Tian. Es un servidor de NodeJS que se puede ejecutar en una red doméstica y que emula la API de HomeKit de iOS. Mediante plugins, que son módulos aportados por la comunidad, proporcionan un puente básico de HomeKit a varias APIs de terceras partes suministradas por fabricantes. [36]

El sistema HomeKit de Apple para comunicarse con dispositivos domóticos comenzó muy lentamente pero actualmente está en constante expansión. Es por ello que Homebridge está recobrando tanta importancia en los últimos días, ya que es el puente que se necesitaba para eliminar la barrera entre Apple y otras marcas e integrar varios dispositivos de distintas marcas en una sola aplicación.

Homebridge actúa como centro entre los dispositivos, siendo posible controlar dispositivos muy diversos en cuando a fabricantes desde una misma aplicación. Tiene una base código fuente totalmente abierta y con capacidad de crear sus propios complementos.

Para que Homebridge sea de utilidad necesita estar ejecutándose en un dispositivo que permanezca encendido todo el tiempo en segundo plano. Idealmente dicho dispositivo no debe consumir mucha energía siendo una posible solución: la Raspberry, que es la usada para este proyecto.

Una gran ventaja que tiene Homebridge es la integración que tiene Apple con su "asistente" Siri, lo cual permite utilizar un Apple TV, iPad, iPhone, Apple Watch, e, incluso, los auriculares AirPods para poder controlar todos los dispositivos inteligentes que podamos tener en nuestro casa.

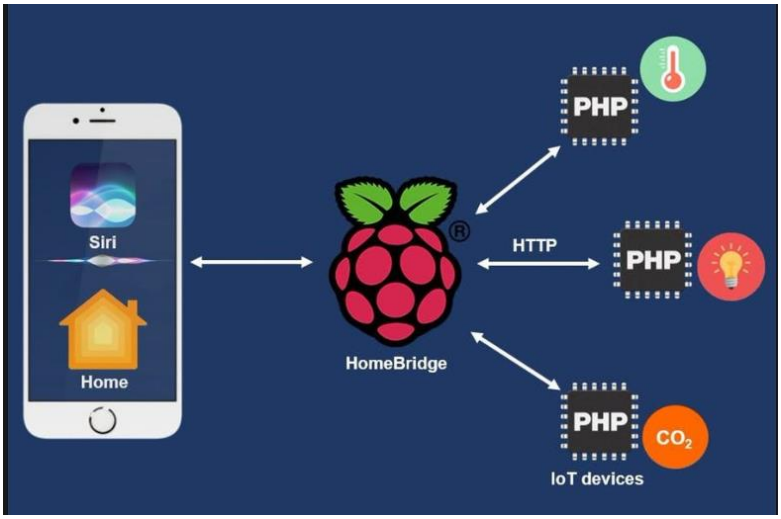

Figura 32. Esquema de conexión iOS y Raspberry [37]

4. NOCIONES BÁSICAS DE PROGRAMACIÓN

4.1 ¿Qué es npm?

Npm se encarga de gestionar módulos y aplicaciones para node.js para crear, compartir y reutilizar modelos en nuestras aplicaciones. Fue creado en 2009 como un proyecto de código abierto para ayudar a los desarrolladores de JavaScript a compartir fácilmente los módulos de código empaquetados. Una vez instalado permite utilizar pequeños módulos e incluso aplicaciones completas. [38]

Npm tiene tres componentes principales: la web, la interfaz de línea de comandos (CLI) y el registro.

- La web se puede usar para encontrar paquetes, configurar perfiles o gestionar otros aspectos como por ejemplo crear organizaciones para el acceso privado a paquetes.
- El CLI se usa desde su ejecución en un terminal y es la forma en la que se interacciona con npm.
- El registro es una amplia base de datos públicas de software de JavaScript.

Una tarea principal para trabajar con npm es conocer el archivo xxx.json. Este archivo se debe crear en la raíz.

Los desarrolladores de código abierto de todos los continentes usan npm para compartir y tomar prestados paquetes, y muchas organizaciones usan npm para administrar el desarrollo privado también.

Con este entorno se puede crear diferentes elementos como:

- Servidores Web
- Aplicaciones en tiempo real
- Juegos para varios jugadores
- Aplicaciones de una sola página
- API basadas en JSON
- Aplicaciones en línea de comandos

Npm dispone de paquetes para dispositivos móviles, IoT, dront-end, back-end, robótica etc, todo lo que necesita para construir nuevas cosas.

4.2 ¿Qué es un módulo o paquete en node.js?

Un paquete es un archivo o directorio que describe algo por medio del archivo *package.json*, mientras que un módulo es cualquier archivo o directorio que puede ser cargado por *node.js*. [39]

Un paquete puede ser alguno de los siguientes casos:

- Una carpeta que contiene un programa descrito por un archivo *package.json*.
- Un tarball comprimido que contiene (a).
- Una url que se resuelve en (b).
- Un <name>@<version>que se publica en el registro con (c).
- Un <name>@<tag>que apunta a (d).
- Un <name>que tiene una etiqueta satisfactoria (e).
- Una gitURL que, cuando se clona, da como resultado (a).

Por otro lado los casos que pueden ser módulos son los siguientes:

- Una carpeta con un archivo *package.json* que contiene un campo *main*.
- Una carpeta con un archivo *index.js* en ella.
- Un archivo JavaScript.

En la mayoría de los casos los paquetes npm que se usan en el programa *node.js* se cargan y se convierten en módulos, pero no es condición indispensable que un paquete sea un módulo. Hay que destacar que casi todos los paquetes npm contienen muchos módulos dentro de ellos. Un ejemplo de paquetes que no son módulos son los paquetes *cli,* que únicamente contienen una interfaz de línea de comandos ejecutables y no contiene un campo *main* para usar en los programas *node.js.*

El archivo *package.json* define el paquete y la carpeta node_modules es el lugar donde *node.js* busca los módulos.

En el contexto de un programa Node, *module* también es lo que se carga desde un archivo. Por ejemplo, en el siguiente comando se puede decir que la variable req se refiere al módulo request. Pero si el módulo no está bien creado con los correspondientes archivos *index.js* y *package.json* el módulo no podrá ser un argumento para require.

```
var req = require('request')
```

Un módulo en *node.js* es simplemente una unidad de código relacionado con un único fin.

Un paquete o módulo es una forma encapsular y extraer de nuestro proyecto principal: librerías, componentes, funcionalidades, etc.

Un módulo puede estar compuesto por uno o más archivos. *Node.js* tiene un sistema simple de carga de módulos y los archivos y los módulos tienen una correspondencia de uno a uno, un módulo puede depender de uno o más módulos de Node.js.

Existen 2 tipos de módulos: los módulos principales y los módulos de archivo [40]

- ✓ **Módulos principales**

  Los módulos principales o también llamados nativos son definidos en el código fuente de node en la carpeta lib/ y tienen la preferencia de carga frente a cualquier otro módulo.

- ✓ **Módulos de archivo**

  Los módulos de archivo son módulos definidos por el usuario y por lo general se registran en NPM para su uso.

4.3 Conceptos básicos de un módulo de archivo en node.js

Primero, es necesario definir una estructura de trabajo para nuestro módulo npm, se llamará mynewmodule en este caso: [40]

```
mynewmodule/
        bin/
        source/
        dist/
        test/
        .npmignore
        package.json
        README.md
```

Dónde:

- La carpeta bin/: Es la que contendrá las funciones con las que se podrá ejecutar nuestro módulo en la línea de comandos.
- La carpeta source/: Es la que contendrá nuestro código fuente.
- La carpeta dist/: Es la que contendrá nuestro código Javascript final que es el resultado de la compilación del código fuente.
- La carpeta test/: Es la que contendrá el código JavaScript final de las pruebas unitarias que tenga nuestro módulo (no son obligatorias).
- El archivo .npmignore: Este archivo es necesario para que npm sepa que archivos de nuestro repositorio ignorar al instalar y descargar nuestro módulo en ordenador cliente.
- El archivo package.json: Este archivo es necesario para que npm sepa toda la información relevante de nuestro proyecto. Por ejemplo: nombre, versión, script principal, scripts secundarios, ruta principal de acceso mediante terminal, autor, tipo de licencia, dependencias, dependencias de desarrollo y mucho más.

- El archivo README.md: Este es un archivo con extensión .md, por defecto es el archivo llamado por npm para mostrar un resumen y/o instrucciones de instalación de nuestro módulo o paquete *node.js*.

4.4 Cómo encontrar y seleccionar paquetes en npm

Para encontrar paquetes ponemos en el buscador de la página web de npm la palabra clave que describa lo que queremos realizar. [40]

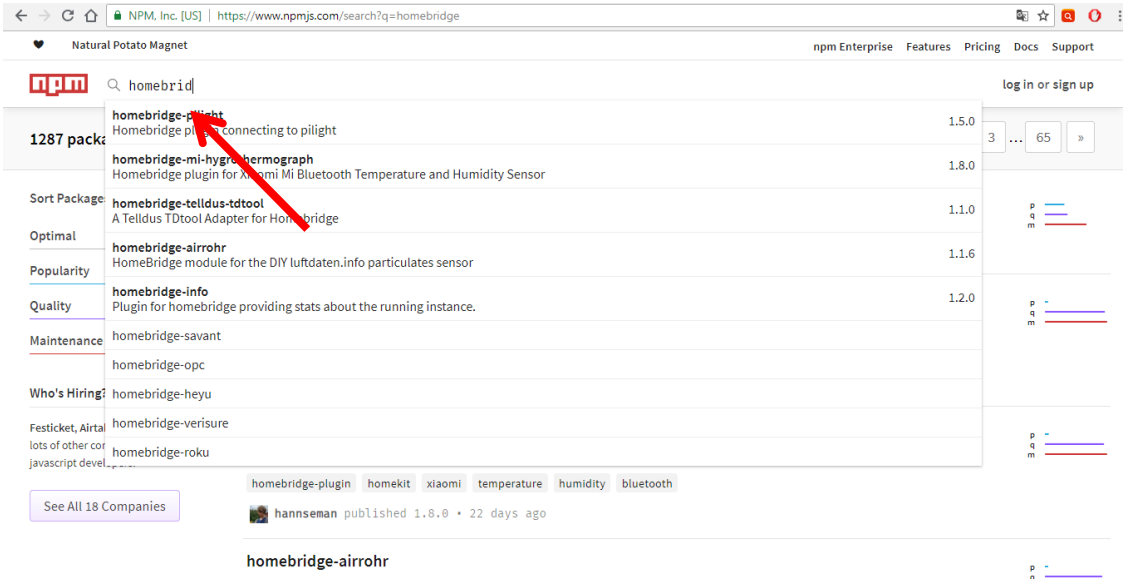

Figura 33. Interfaz web npm

Obtenemos diversos números paquetes que pueden ser similares entre ellos por el nombre o por la funcionalidad pero para ello npm hace un análisis de los mismos a la derecha atendiendo a los siguientes criterios:

✓ **Popularidad**

La popularidad indica cuántas veces se ha descargado el paquete. Este es un buen indicador de paquetes que otros han encontrado que son especialmente útiles, pero no infalibles.

✓ **Calidad**

La calidad incluye consideraciones tales como la presencia de un archivo *README*, la estabilidad, las pruebas, las dependencias actualizadas, el sitio web personalizado y la complejidad del código.

- ✓ **Mantenimiento**

  El mantenimiento clasifica los paquetes según la atención prestada por los desarrolladores. Los paquetes que se mantienen con mayor frecuencia tienen más probabilidades de funcionar bien con las versiones actuales o futuras de npm, por ejemplo.

- ✓ **Optimal**

  Optimal combina los otros tres criterios de una manera significativa.

Cuando eliges un paquete, aparece más información. Esta información la escriben los autores del paquete para que los detalles varíen. Aquí es donde puedes descubrir cómo usar el paquete y además, existen desarrolladores a menudo que proporcionan información de contacto.

Al hacer una búsqueda en npm se obtienen cientos de complementos Homebridge, éste por sí solo no es muy útil, y donde realmente brilla es su sistema de complementos respaldado por la comunidad y abierto para todos.

4.5 Introducción a node.js

Node.js es un entorno de programación en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor (pero no limitándose a ello) basado en el lenguaje de programación ECMAScript, asíncrono, con E/S de datos en una arquitectura orientada a eventos y basado en el motor V8 de Google

- Wikipedia -

Al ser un entorno multiplataforma está presente en los siguientes S.O: Windows, Mac OS X, GNU/Linux, Solaris, FreeBSD, OpenBSD, WebOS en servidores NAS (Synology y QNAP) y Raspberry. [41]

Antes de *node.js* hubo varios intentos de establecer un entorno de ejecución para JavaScript fuera de los navegadores. Pero finalmente *node.js* y npm se convirtieron en el entorno de ejecución de JavaScript en el servidor.

Existe también el intérprete de node, el REPL (Read Eval Print Loop), escribiendo el comando node en el terminal nuestra consola pasa a ser una consola de JavaScript. El REPL es una consola que ejecuta cada expresión en JavaScript que le demos y devuelve el resultado. [42]

Por ejemplo si ejecutamos

```
> console.log('Hola Mundo')
```

La consola devuelve:

```
'Hola Mundo'
undefined
```

Tal como se puede ver en la siguiente imagen:

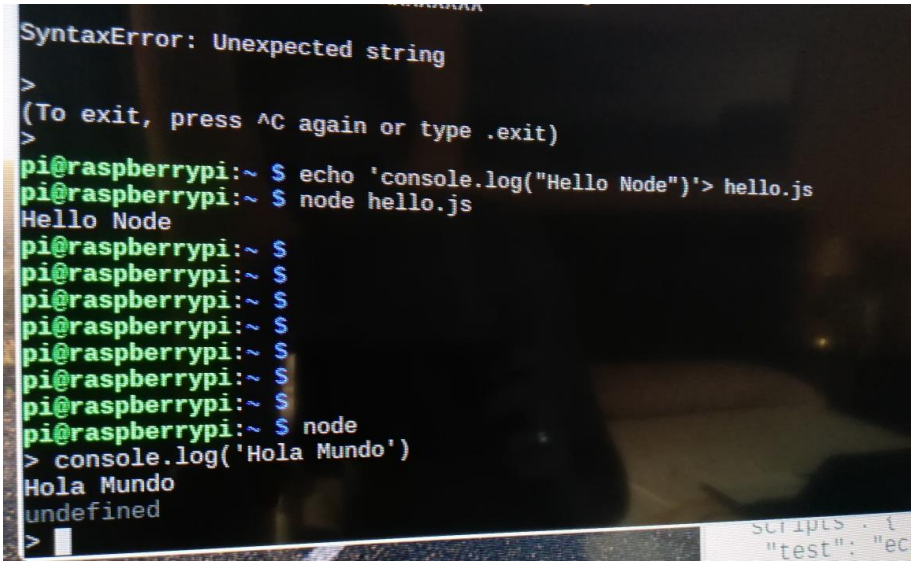

Figura 34. Ejemplo 1 Introducción node

El REPL es muy útil para probar pequeñas funciones y expresiones, ya que es inevitable tener una duda de código mientras se programa y con esto nos daríamos cuenta al probarlo de cualquier fallo.

Node no es solo el REPL, también podemos ejecutar ficheros siendo cada fichero JavaScript es un módulo para *Node.js*. Este es uno de los puntos fuertes de node al haberse mantenido flexible gracias a tener una librería estándar muy pequeña. Aquí es donde también se encuentra inmerso el mundo npm, que como se comentó anteriormente se trata de un repositorio centralizado de módulos para node.js La filosofía de *node.js* y npm es la de módulos pequeños que hagan una sola cosa. Esto hace el lenguaje más fácil de componer, reordenar y modificar y tiene un gran potencial.

Node.js se instala en nuestro sistema junto con npm y se puede usar para instalar cualquier módulo de forma global o local en nuestro proyecto.

Si creamos una carpeta nueva y ejecutamos el comando `npm init` el programa nos hará unas preguntas sobre el proyecto y creará un fichero *package.json* con la configuración mínima.

Ejecutando `npm install` se instalará cualquier paquete de registro. Si queremos hacer uso o crear servidores web existe la librería express.

Una vez instalado *express.js* podemos crear el fichero *index.js*.

A continuación se muestra un ejemplo de lo explicado anteriormente.

Fichero *index.js*

```
const express = require('express')
const app = express()
app.get('/', function (req, res) {
  res.send('Hola desde Node!')
})
app.listen(3000, function () {
  console.log('Servidor creado y escuchando en el puerto 3000!')
})
```

Al ejecutar:

```
node index.js
```

Se obtiene la siguiente respuesta en el terminal:

```
Servidor creado y escuchando en el puerto 3000!
```

La siguiente imagen muestra lo explicado anteriormente:

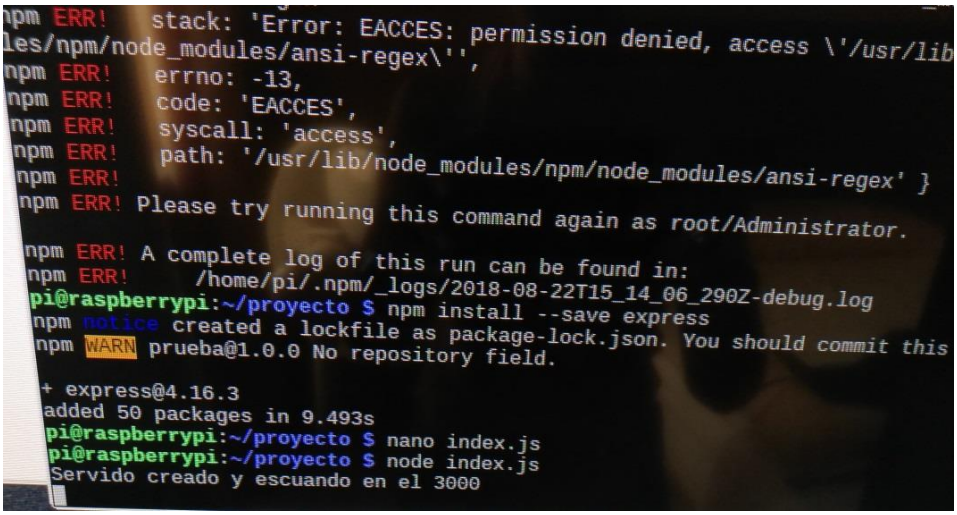

Figura 35. Ejemplo 2 Introducción node

Si se abre el navegador y tecleamos “localhost:3000” se verá el mensaje “ hola desde node”.

Node destaca en el modelo de concurrencia al usar un modelo con un único hilo de ejecución para todas las conexiones que se ejecuten con el servidor, siendo las redes basadas en múltiples hilos más ineficientes, ya que además este tipo de redes suele bloquear el sistema cuando no se usa mientras que *node.js* se mantiene en reposo. La ventaja de compartir un único hilo entre todas las solicitudes que se atienden es que se pueden usar aplicaciones altamente concurrentes, con esto se disminuye el principal inconveniente de las aplicaciones web que era el número máximo de conexiones que podía manejar un servidor.

Mientras que el principal inconveniente radica en que este modelo requiere de módulos adicionales para escalar la aplicación con el número de núcleos de procesamiento de la máquina en la que se esté ejecutando.

Otro aspecto de node es el Chrome V8, el entorno de ejecución de JavaScript creado para Google. Este entorno es un software libre escrito en C++ y compila el código fuente JavaScript.

Node usa lo que se conoce como la programación orientada a eventos, con JavaScript como lenguaje base que da lugar al uso de funciones y cierres anónimos. Destaca por su altísima velocidad y la forma tan sencilla que tiene para la creación de aplicaciones web escalables.

5. GUÍA DE INSTALACIÓN

5.1 Puesta en marcha Raspberry

Para la puesta en marcha de la Raspberry debemos realizar los siguientes pasos. En primer lugar si no tenemos instalado el sistema operativo debemos instalarlo nosotros, para ello se puede descargar de la página web oficial de Raspberry. Para este proyecto se ha usado el sistema operativo Raspbian, aunque como se ha comentado antes existen otros compatibles. Seguidamente extraer la tarjeta SD de la Raspberry e introducirla en el ordenador. El sistema operativo es una imagen Raspbian que debemos escribir sobre la tarjeta SD para ello se puede hacer uso de la aplicación Win32DiskImager, la imagen que hemos descargado será la que debamos escribir con la aplicación.

Una vez terminado el proceso volvemos a introducir la tarjeta miniSD en la Raspberry y ya la tenemos lista para usar con el sistema operativo Raspbian.

Para el uso de Raspberry desde el pc se conecta un cable de Ethernet desde la Raspberry al ordenador. En la configuración de redes del ordenador es necesario asignarle una IP fija al puerto de Ethernet como se muestra en las siguientes imágenes:

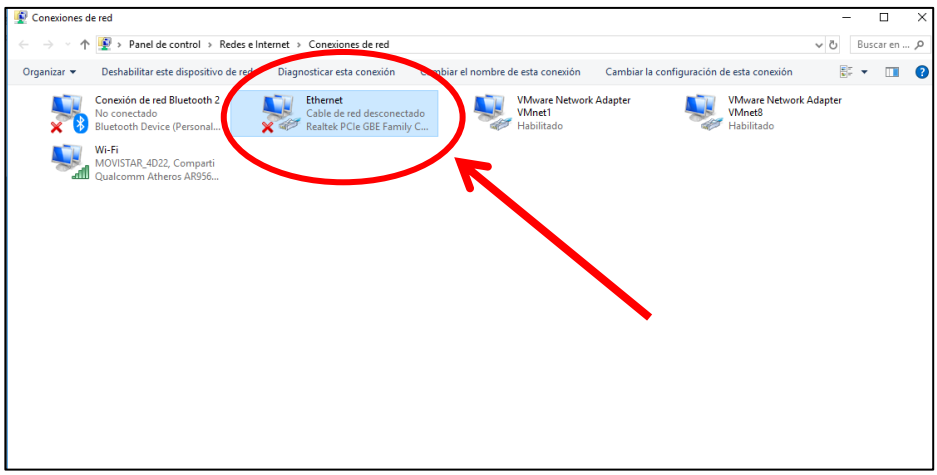

Figura 36. Paso 1: Configuración Ethernet

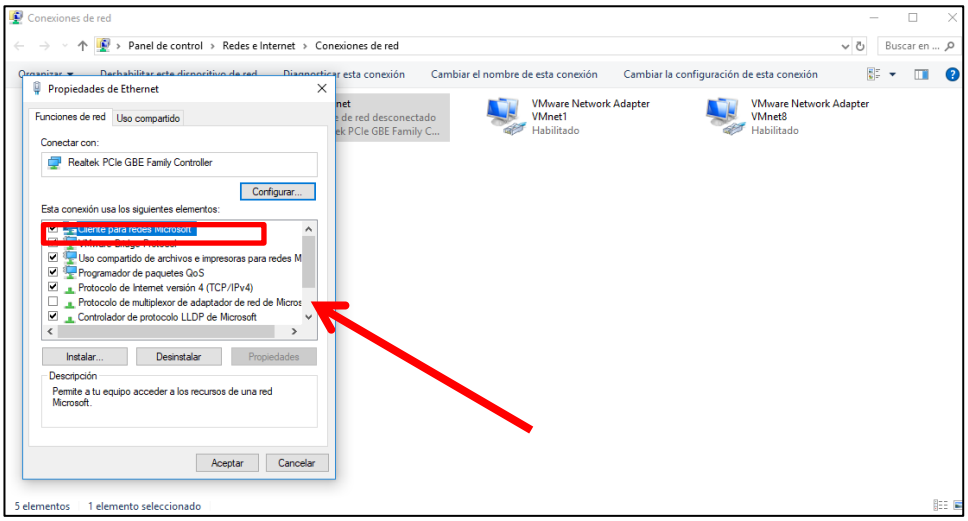

Figura 37. Paso 2: Propiedades IPv4

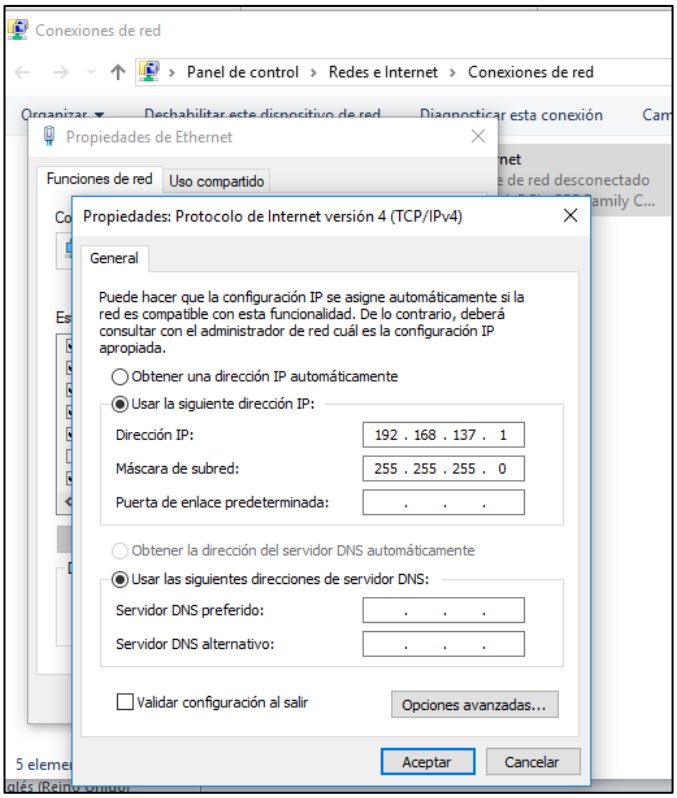

Figura 38. Paso 3: Actualización IPv4

Una vez realizado el cambio de IP fija hay que asignarle una IP fija también a otro puerto para la Raspberry. Para asignarle una IP fija a la Raspberry hay que introducir la tarjeta SD en el ordenador y buscar el archivo *cmdline.txt* que se ha creado con el sistema operativo y posteriormente se introduce al final del archivo "IP=192.168.138.5" nótese el cambio del último número. De esta forma podemos conectar la Raspberry y el ordenador mediante el cable de Ethernet.

Si queremos que nuestra Raspberry tenga acceso a internet desde el ordenador es necesario marcar nuestra red Wifi de uso compartido con los siguientes pasos:

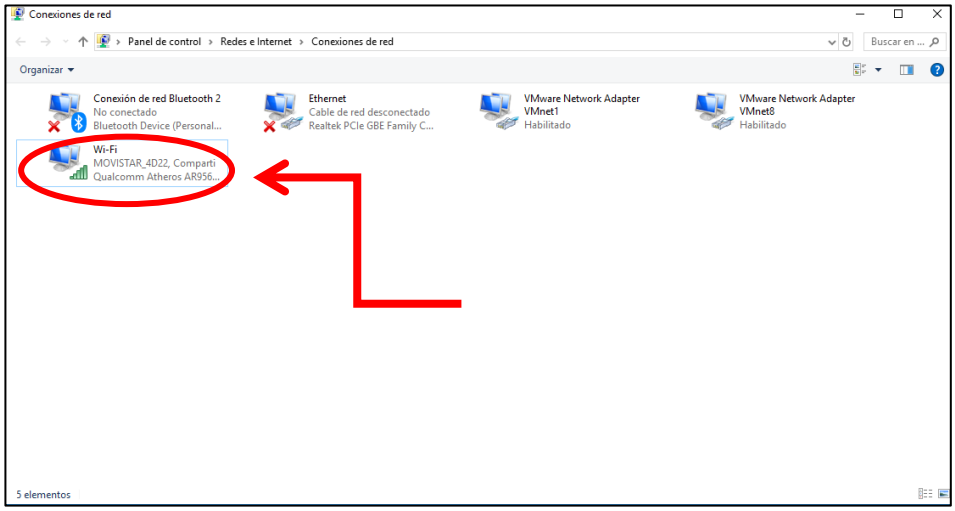

Figura 39. Paso 1: Configuración internet en Raspberry

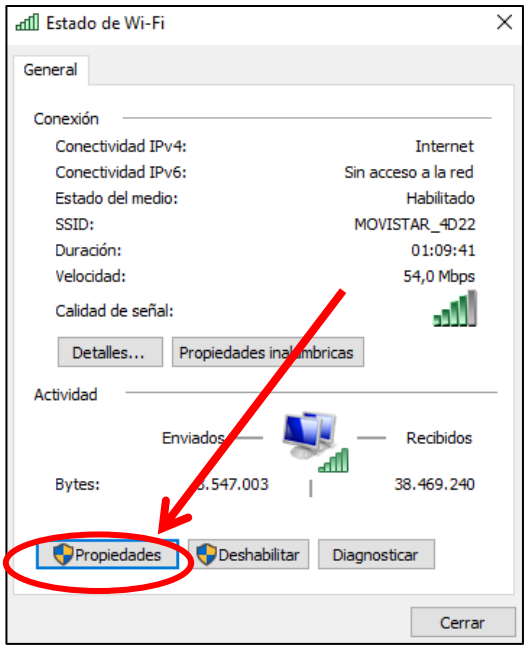

Figura 40. Paso 2: Configuración internet en Raspberry

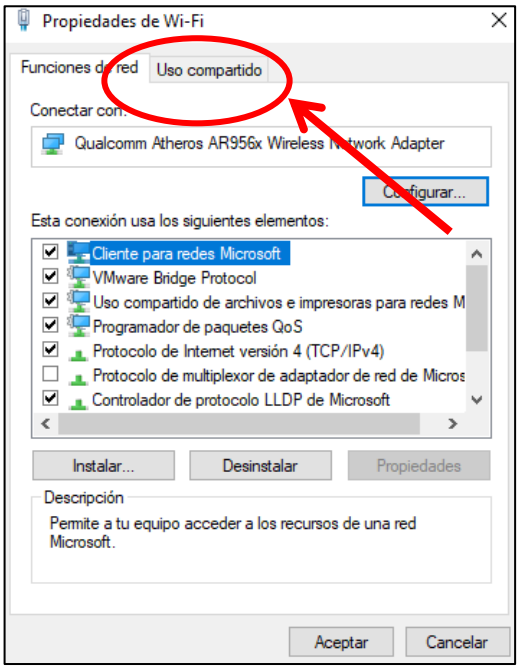

Figura 41. Paso 3: Configuración internet en Raspberry

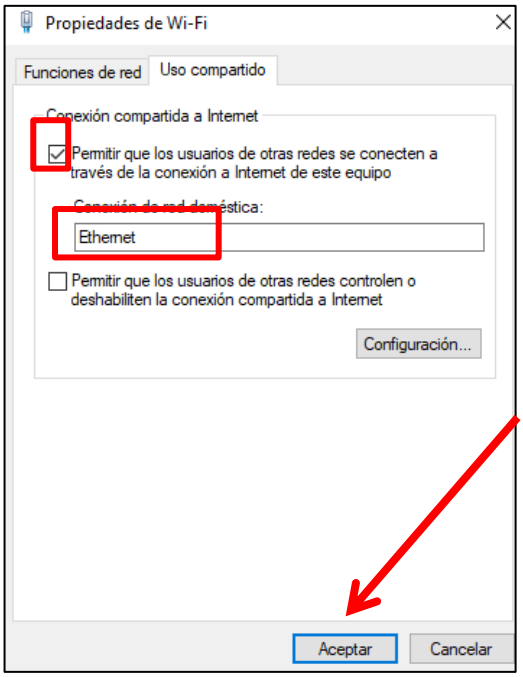

Figura 42. Paso 5: Configuración internet en Raspberry

Para acceder a la Raspberry y poder programar y/o visualizar el sistema operativo desde el ordenador existen dos opciones: SSH o VNC. Sin embargo, si se ha instalado el sistema operativo por primera vez la Raspberry tendrá deshabilitada ciertas características de fábrica que no permitirán acceder mediante SSH o VNC desde el ordenador. Para solucionar este problema antes, hay que conectar la Raspberry a la una pantalla mediante HDMI y ejecutar el sistema operativo y en el menú configuración activar SSH y VNC.

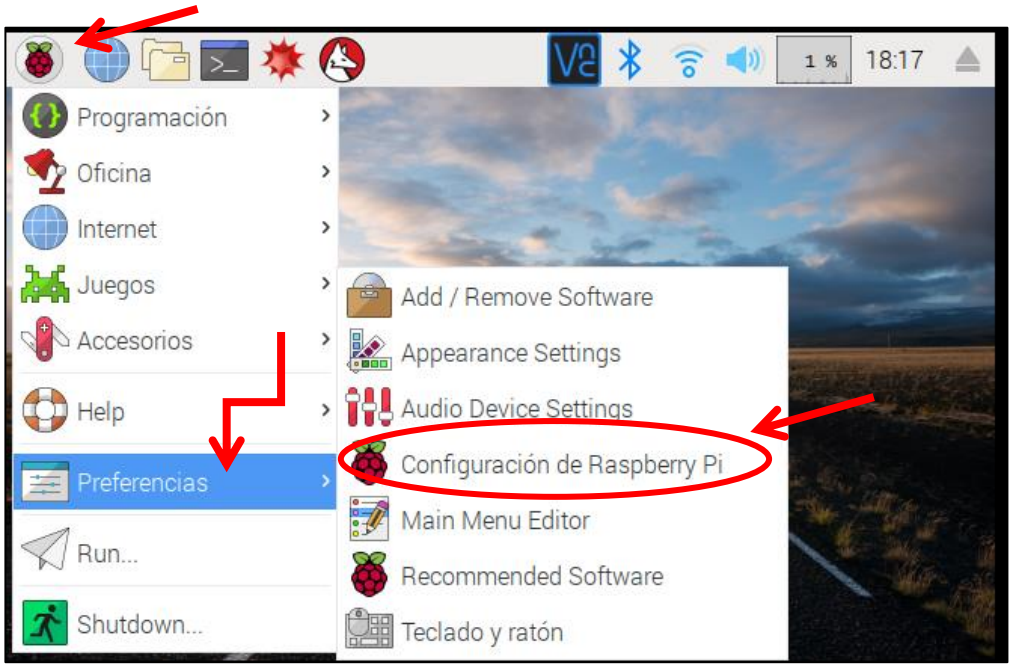

Figura 43. Paso 1: Configuración SSH y VCN

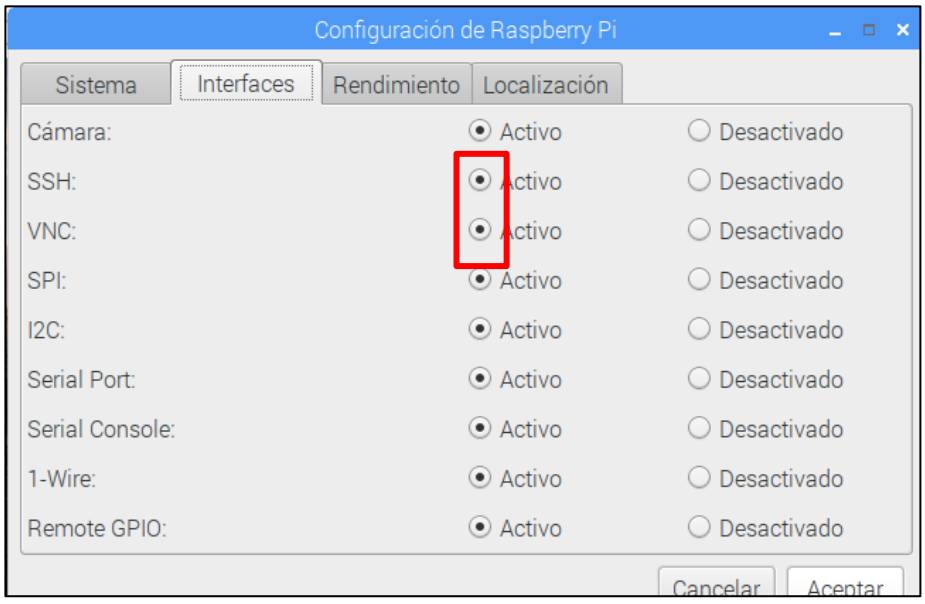

Figura 44. Paso 2: Configuración SSH y VCN

Además aparecerá un aviso alertando que el usuario y contraseña de la Raspberry es el que viene por defecto y esto puede ser un riesgo, haciendo clic en la advertencia redigirá al cambio de las credenciales. Por defecto éstas son *usuario: pi* y *contraseña: Raspberry*.

Una vez activado el acceso a las credenciales ya se puede conectar la Raspberry al ordenador y llevar acabo los siguientes pasos para el uso de la misma a través de cualquiera de las dos interfaces:

- SSH

Para el acceso mediante SSH es necesario descargar una aplicación que nos permita acceder a una interfaz de comunicación entre ambos, en este proyecto se ha descargado el software Putty. Una vez descargado y ejecutado aparecerá en la pantalla del ordenador la siguiente imagen, en el campo Host Name (or Ip address) introducimos la IP que se le ha asignado a la Raspberry.

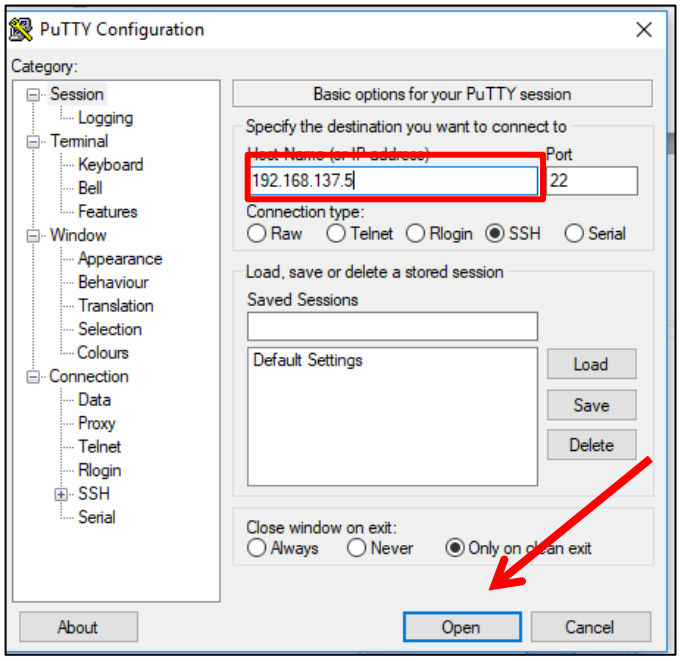

Figura 45. Configuración PuTTy

Una vez introducido la IP hay que introducir las credenciales de acceso a las Raspberry, que si se han cambiado habrá que introducir las nuevas y sino las que vienen por defecto.

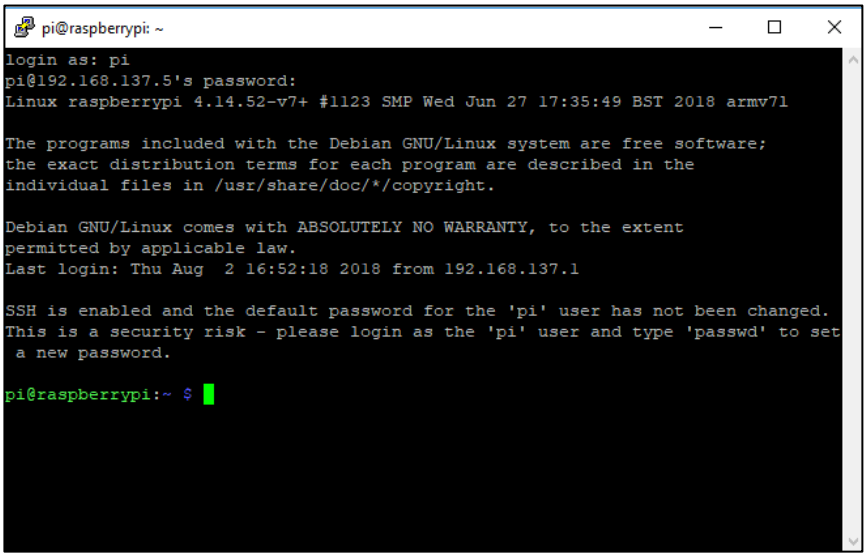

Figura 46. Inter faz PuTTy

- VNC

La aplicación VNC es más completa que SSH ya que permite visualizar el sistema operativo y además permite acceder al mismo terminal Linux para ejecutar comandos línea a línea.

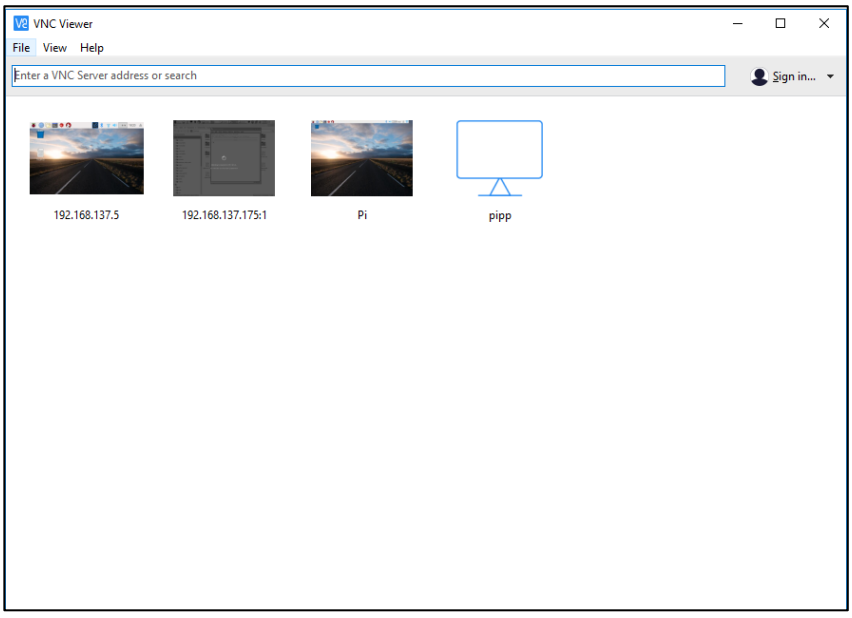

Figura 47. Interfaz VNC

Una vez aparezca en la pantalla la interfaz de la aplicación hacemos clic en *File→New connection* e introducimos de nuevo la IP de la Raspberry y debe aparecer la siguiente imagen:

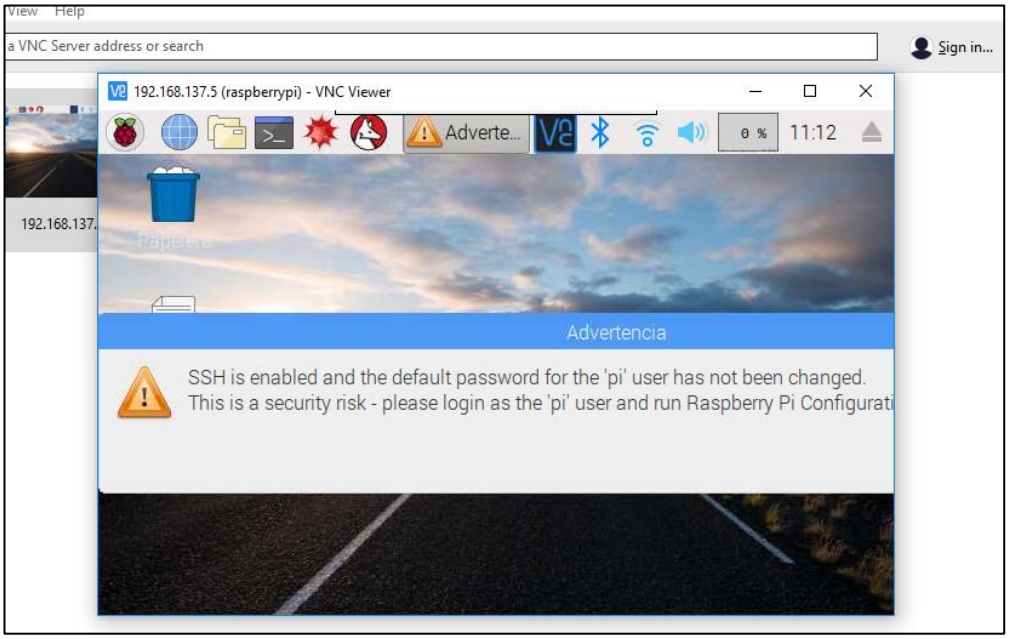

Figura 48. Interfaz VNC

Esta interfaz presenta la ventaja de que se puede acceder de una manera fácil a cualquier parte del sistema operativo, ya sean directorios, carpetas o configuración. Sin embargo al ser una conexión remota a escritorio tiene el inconveniente de un comportamiento más lento que una conexión física o una conexión SSH al tener la necesidad de transmitir una gran cantidad de datos principalmente por el envío de señal de video. Por último VNC es más inseguro que SSH y la comunicación es más susceptible de ser interceptada.

5.2 Instalación Homebridge en Raspberry

La instalación de Homebridge es muy sencilla y para ello se deben seguir los siguientes pasos:

El primer paso es tener la Raspberry actualizada para ello ejecutar:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

Para el uso de Homebridge hay que tener instalados los paquetes git y make en nuestra Raspberry, Git es un sistema de control de versiones (VCS) para rastrear cambios en archivos y coordinar cambios entre varias personas que trabajan en la misma base de código, para ello escribimos en la línea de comandos:

```
 sudo apt-get install git make
```

También es necesario instalar g++:

```
sudo apt-get install g++
```

El siguiente paso es instalar Node.js para poder instalar después los paquetes necesarios en la Raspberry y los servicios de avahi para permitir configuraciones en red, para ello ejecutar los siguientes:

```
sudo curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash -
```

El paquete que se instalará a continuación se trata de una especie de multicast de DNS que se utilizará dentro de *HAP-NodeJS* (implementación NodeJS de HomeKit Accessory Server que usa Homebridge). Este paquete permite notificar y descubrir servicios y servidores a los programas que se están ejecutando en una red local sin una configuración previa determinada. Esto se puede usar por ejemplo para conectarse a una red y encontrar impresoras de la misma.

```
sudo apt-get install libavahi-compat-libdnssd-dev
```

Ahora es el turno de instalar el gestor de módulos npm, herramienta que se ha explicado anteriormente. Si al ejecutar el comando da error puede ser por que se instalado ya al instalar node.js.

```
sudo apt-get install npm
```

Y por último instalar Homebridge:

```
sudo npm install -g --unsafe-perm homebridge
```

Para ejecutarlo únicamente hay que escribir en la línea de comando:

```
homebridge
```

Al ejecutar Homebridge debe aparecer la siguiente pantalla:

Figura 49. Pantalla iniciación Homebridge

Al aparecer el error que se encuentra en la imagen anterior: el archivo *config.json* no ha sido encontrado es debido a que para que Homebridge funcione correctamente es necesario crear un archivo *config.json*, que será el que archivo que contengan los accesorios y/o plataformas de los accesorios que se quieren conectar con HomeKit a través de Homebridge.

Como prueba de ejemplo, Homebridge trae un archivo *config.json* de ejemplo que se puede utilizar. Este archivo *config-sample.json* se encuentra en una carpeta diferente a la que debe estar *config.json* y está ubicado en el siguiente directorio:

```
usr/lib/node_modules/homebridge
```

Una vez copiado el archivo *config-sample.json* se le cambia el nombre por *config.json* y se debe ubicar dentro de `.homebridge,` que está dentro de la carpeta de inicio pi. En Raspbian, la ruta sería la siguiente:

```
home/pi/.homebridge/config.json
```

Si la Raspberry se encuentra conectada al ordenador e intentamos introducir el código de Homebridge como accesorio en la App Home nunca lo reconocerá y esto es porque, para que el dispositivo Apple detecte Homebridge como accesorio hay que ejecutar Homebridge y que la Raspberry se encuentre conectado a la red por cable.

Una posibilidad es ejecutar Homebridge mediante SSH en la pantalla del ordenador teniendo la Raspberry conectada al ordenador mediante un cable de Ethernet, posteriormente desconectar el cable de Ethernet y conectarlo al router, pero con esta opción ya se deja de visualizar en pantalla el terminal Linux y por tanto no se puede ver la respuesta de éste cuando se ejecuta cierta orden desde HomeKit. Otra opción es conectar la Raspberry mediante cable HDMI a una pantalla y mediante cable de Ethernet al router. Esta segunda opción permite realizar modificaciones de código a la vez que se va probando que Homebridge funciona correctamente en HomeKit.

Si el router es poco accesible para poder mantenerlo conectado como se ha explicado antes existe una alternativa, los PLC de TP-Links con conexión Ethernet. Estos dispositivos se conectan a la corriente y mediante la línea eléctrica transmite la señal de red y al conectar el cable de Ethernet desde la Raspberry al PLC ésta funciona como si estuviera conectada directamente al router.

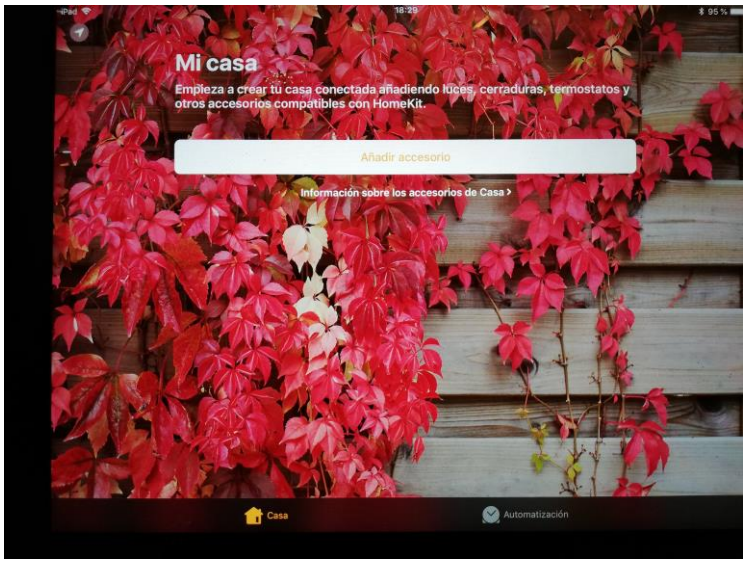

Figura 50. Paso 1: Agregar accesorio Home

Una vez se obtiene el código del archivo *config.json* de Homebridge ya se puede agregar como accesorio en la App Home de nuestro dispositivo Apple. Para ello acceder a la aplicación Home del dispositivo Apple y hacer clic en añadir accesorio. Hay dos opciones para añadir accesorio, haciendo una foto al código QR que se genera en el terminal Linux cuando se ejecuta Homebridge o mediante el código que aparece también en el terminal.

Si todo funciona correctamente Home debe reconocer directamente Homebridge como un accesorio que se encuentra cerca de su alcance tal como se puede ver en la siguiente figura:

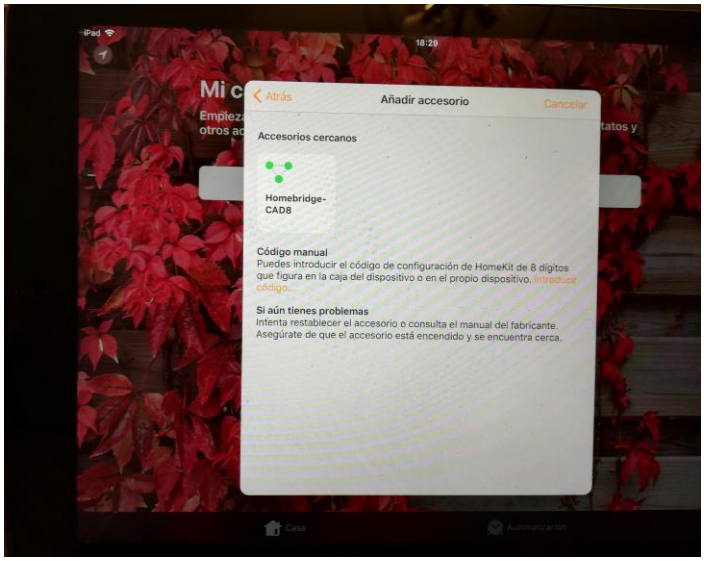

Figura 51. Paso 2: Accesorio detectado

Hacemos clic sobre el accesorio y tecleamos el código del accesorio. Home te da la posibilidad de realizar una primera caracterización del dispositivo dentro de las posibles ubicaciones creadas en la aplicación, aunque posteriormente ésta podrá modificarse

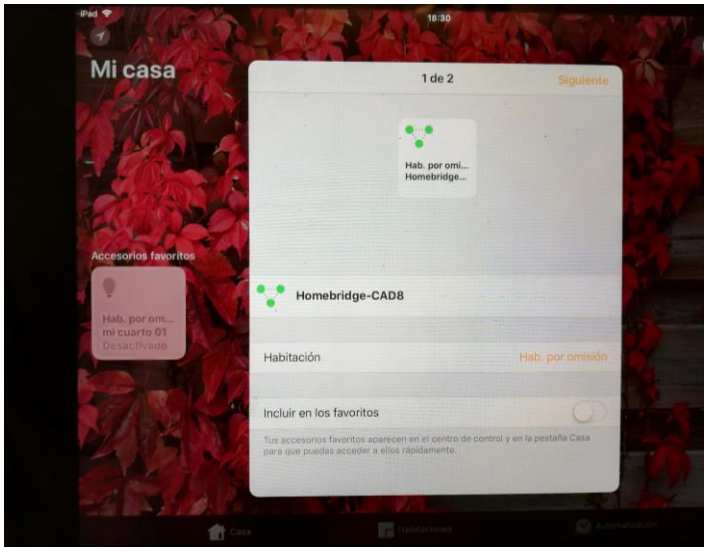

Figura 52. Paso 3: Configuración Homebridge

Una vez realizado el paso anterior tendremos Homebridge instalado correctamente y podremos ver Homebridge como accesorio y además los propios dispositivos que hayamos configurado en Homebridge dentro del archivo *config.json.*

Posteriormente, cuando ya se ha instalado Homebridge debemos instalar los plugins de los accesorios que se desea conectar con HomeKit. En el apartado Aplicaciones se verá de forma detalla varios ejemplos de configuraciones de distintos dispositivos a través de Homebridge. Es importante al instalar los plugins de asegurarse de seguir las guías correspondientes de cada uno de ellos después de instalarlos ya que estos plugins han sido creados por diferentes personas y cada uno de ellos les ha dado una configuración distinta.

Un error muy común es que iOS no reconozca Homebridge como accesorio, esto puede ser debido a dos cosas.

- El servidor Homebridge cree que ha sido emparejado, pero iOS piensa lo contrario para ello eliminar la carpeta persist y accesories del directorio donde se encuentra *config.json*
- El dispositivo iOS ha hecho que Homebridge se haya quedado atascado, encuentra la base de datos pero está inactivo, para ello cambiar alguna letra de username en el archivo *config.json*

Y a pesar de la gran potencia, Homebridge sigue teniendo ciertas limitaciones como que solo puede tener 100 accesorios debido a un límite establecido por HomeKit y otra limitación se produce al cambiar el nombre de un accesorio que ya ha sido agregado a la aplicación, ya que se debe cambiar también de nombre tanto en Homebridge como en HomeKit.

5.3 Arranque de la aplicación Homebridge

Una comodidad para el usuario es el arranque automático de Homebridge. La Raspberry debe estar encendida constantemente y por consiguiente conectada a la red y esto puede ser susceptible de posibles reinicios por ejemplo. si se va luz, por lo que si desea un sistema autónomo debemos realizar la configuración de los siguientes pasos.

Existen dos métodos para el arranque del programa Homebridge en segundo plano cada vez que se encienda la Raspberry: [43]

1. Usar la pantalla: para ello es necesario instalar la pantalla con el comando

   ```
   sudo apt-get install screen.
   ```

   Luego editar el archivo: nano /etc/rc.local y agregarla siguiente línea de código al final del archivo que estamos editando:

   ```
    /usr/bin/screen -d -m -U -S homebridge /usr/bin/Homebridge
   ```

2. Configurándolo como un servicio init.d. Este tipo es un poco más complejo pero es más eficiente. Primero se crea el archivo para el servicio con el comando:

   ```
   sudo nano /etc/init.d/Homebridge.
   ```

   Posteriormente hay que copiar el código del Anexo A en el archivo creado.

El siguiente paso es establecer los permisos para ese archivo y para ello ejecutamos:

```
sudo chmod 755 /etc/init.d/Homebridge
```

```
sudo update-rc.d homebridge defaults
```

Por último se autoriza al usuario a ejecutar esa secuencia de comandos en el arranque:

```
sudo visudo
```

Y agregar las siguientes líneas al final de la sección

```
# Allow members of group sudo to execute any command
```

```
homebridge ALL=(ALL) NOPASSWD: /etc/init.d/homebridg
```

```
homebridge ALL=(ALL) NOPASSWD: /usr/bin/Homebridge
```

```
homebridge ALL=(ALL) NOPASSWD: /usr/local/bin/Homebridge
```

```
homebridge ALL=(ALL) NOPASSWD: /usr/bin/npm
```

También se puede iniciar la aplicación manualmente con el comando

```
sudo /etc/init.d/homebridge start
```

5.4 Compartir controles en HomeKit

La plataforma y todos los accesorios que se integren en la App Home están enlazado mediante el ID de Apple del dueño del dispositivo, de forma que solo el dueño del dispositivo puede acceder al control de los accesorios. Pero surge el inconveniente de que el dueño quiera que un invitado pueda acceder a los controles de Home, y Apple ha tenido en cuenta esta casuística y hay un modo de realizarlo. [44]

Para permitir que otros usuarios de Apple tengan acceso a una aplicación home de otro dispositivo Apple es necesario abrir la aplicación Home y hacer clic en el icono de la brújula que hay en la parte superior derecha, una vez abierta aparecerán una serie de opciones de configuración de HomeKit para el propio usuario.

Entre las opciones que Apple presenta hay que buscar la sección Personas y hacer clic en invitar. De esta forma se abrirá la opción para añadir a cualquier persona a la aplicación Home para que pueda realizar cambios cuando desee. Estas personas se añaden a través de su ID de Apple.

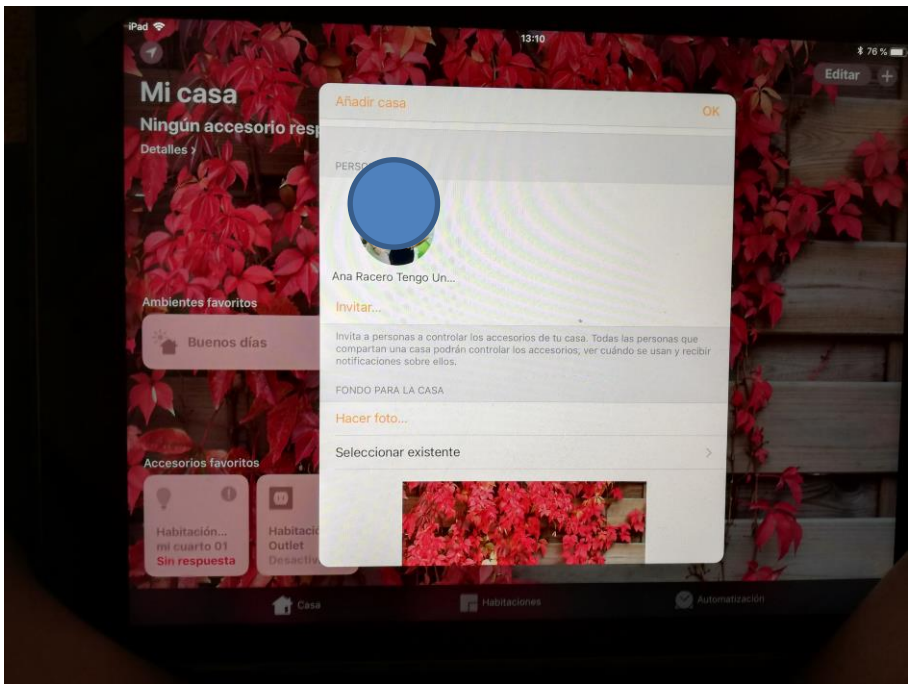

Figura 53. Configuración para compartir controles

Una vez seleccionado el invitado hacer clic en "Ok". El invitado debe tener la aplicación Home instalada en iOS y le aparecerá la invitación y al aceptarla éste podrá ver en su aplicación todos los accesorios de la casa que está invitado.

La sección donde hemos agregado al invitado también tiene la opción de eliminar al invitado, pulsando en el avatar se puede ver la opción de retirar su acceso a los dispositivos de la casa.

Los invitados pueden tener la opción de tener permisos de edición que puedan modificar ambientes o automatizaciones.

5.5 Automatización en HomeKit

La App Home permite la automatización de nuestro sistema domótico de forma muy sencilla. A continuación, se muestran los pasos a seguir para automatizar por ejemplo el encendido de la una luz todos los días entre semana a una determinada hora.

Hacer clic en automatización→ Crear nueva automatización o símbolo "+" situado arriba a la derecha si ya tenemos alguna automatización creada. Home da a elegir entre 4 opciones para cuando quieres que ocurra esa automatización:

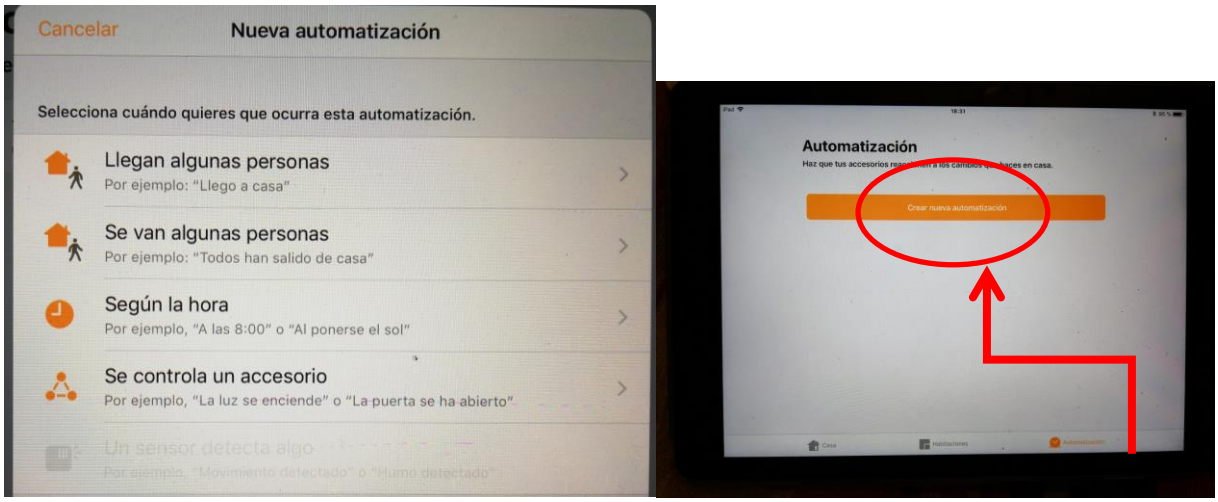

Figura 54. Paso 1: Automatización Home

Al hacer clic en 'Según la hora' aparece la configuración que Home te permite modificar para ese tipo de automatización.

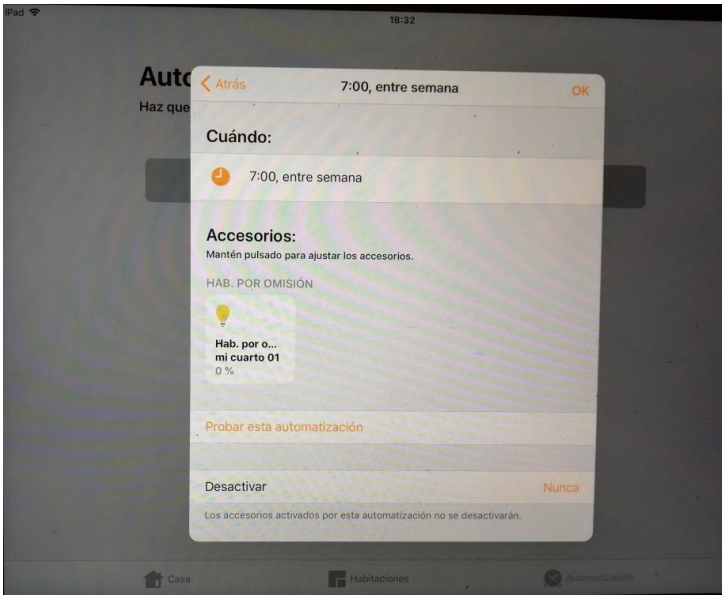

Figura 55. Paso 2: Automatización Home

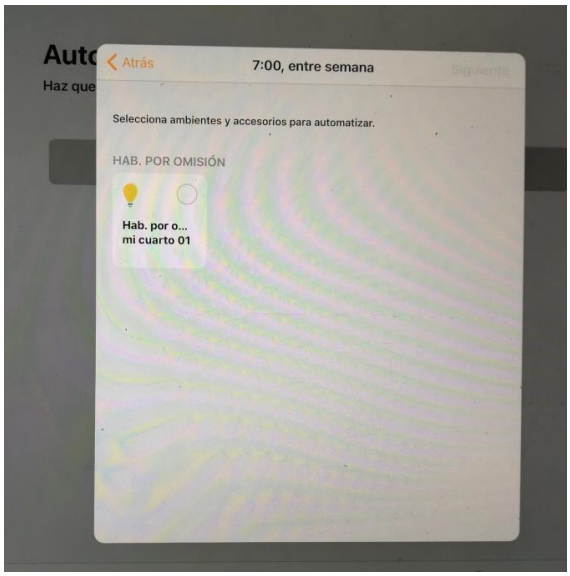

Figura 56. Paso 3 Automatización Home

Y aunque ésta es solo un ejemplo existen infinidad de posibles automatizaciones, ya que dependiendo del accesorio que vayamos automatizar tenemos unas u otras posibilidades que ofrece la aplicación Home para cada dispositivo.

6. APLICACIONES

En la siguiente imagen se muestra un esquema de las posibilidades que existen para hacer de HomeKit la App para una casa domotizada. Tenemos accesorios compatibles que únicamente hay que agregarlos a la aplicación mediante un código pero por otro lado existen accesorios no compatibles, para los cuales se ha usado un puente denominado Homebridge para hacerlos compatibles con la aplicación Home.

En los siguientes apartados se desarrollan distintos accesorios compatibles y no compatibles con HomeKit así como la instalación de cada uno de ellos.

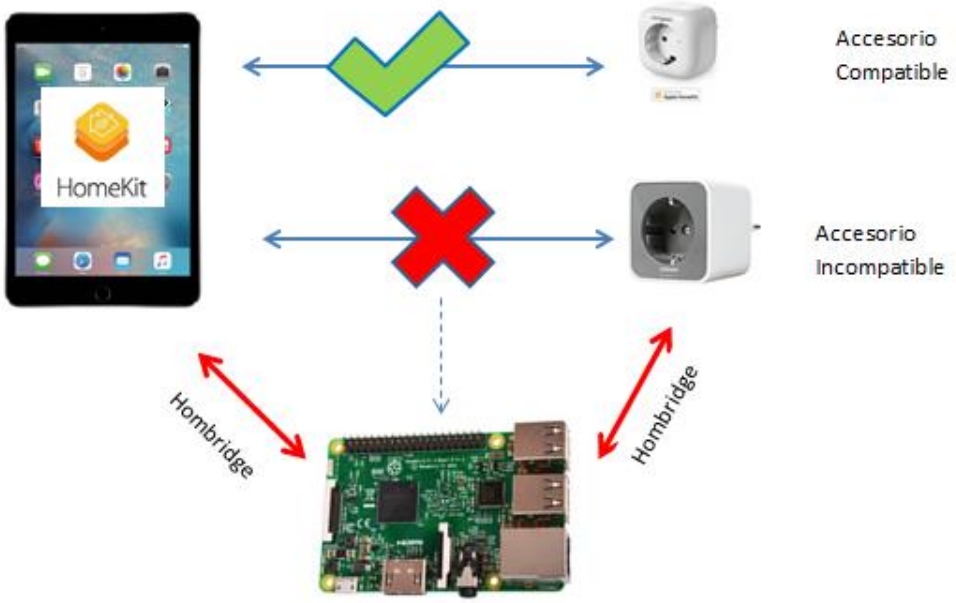

Figura 57. Esquema conexiones Accesorios & HomeKit

6.1 Desarrollo de switch artificial

A continuación se detallan los pasos a seguir para la creación de un módulo npm con la funcionalidad de un switch. [45]

En primer lugar hay que crear un nuevo repositorio que contenga un archivo *package.json* para administrar las dependencias y un archivo *index.js* que contendrá la lógica del accesorio. El archivo package se rellenará automáticamente cuando se da de alta el plugin creado en npm mientras que el arhivo *index.js* se debe rellenar con lo que se va a explicar a continuación.

Se hace la siguiente suposición con respecto a la API de conmutación:

- Las solicitudes GET realizadas a /api/status devuelve un valor booleano que representa el estado actual del conmutador. Al hacerlo, leerá la característica *On* del interruptor.
- Las solicitudes POST realizadas para /api/order contienen un booleano que representa el estado de destino del conmutador y que activarán la acción correspondiente. Al hacerlo actualizará la característica *On* del interruptor

El nuevo plugin de Homebridge contendrá un nuevo accesorio con dos servicios:

- *AccessoryInformation* servicio, requerido para cada accesorio, cualquiera que sea el tipo, contiene la información de transmisión relacionada con el mismo dispositivo.
- *Switchservicio*, correspondiente al cambio real. Este servicio tiene una única característica *On* que será booleana.

Para ello hay que introducir el accesorio dentro de Homebridge, esto se hace con el objeto de JavaScript mySwitch que muestra a continuación:

```
const Service, Characteristic;

module.exports = function (homebridge) {
  Service = homebridge.hap.Service;
  Characteristic = homebridge.hap.Characteristic;
  homebridge.registerAccessory("switch-plugin", "switchome", mySwitch);
};
```

La lógica del núcleo construida en Hap-node.js y Homebridge se encuentra dentro de *get services*, función prototipo de mySwitch.

Para ello es necesario instanciar los siguientes conceptos:

- Un servicio *AccessoryInformation* que contiene:
  - Una característica *Manufacturer*
  - Una característica *Model*
  - Una característica *SerialNumber*
- Un *Switchservicio* que contiene:
  - una característica *On*: la única característica requerida de este servicio.

A diferencia de la característica de *AccessoryInformation*, que son legibles y se pueden configurar en la inicialización del complemento, la característica *On* es de escritura y requiere un getter y un setter.

Ahora se procede a escribir la lógica del *on* dentro de la función prototipo dedicada a mySwitch. Las solicitudes get devuelven el estado actual del conmutador y las solicitudes post envían el estado que se desea que evolucione el conmutador. Para las solicitudes http se usan los módulos request y url.

El código que hay introducir en el archivo se encuentra detallado en el Anexo B.

Una vez creado el archivo *index.js* con la información necesaria para que nuestro accesorio funcione se procede a registrar un nuevo usuario en la web oficial de npm:

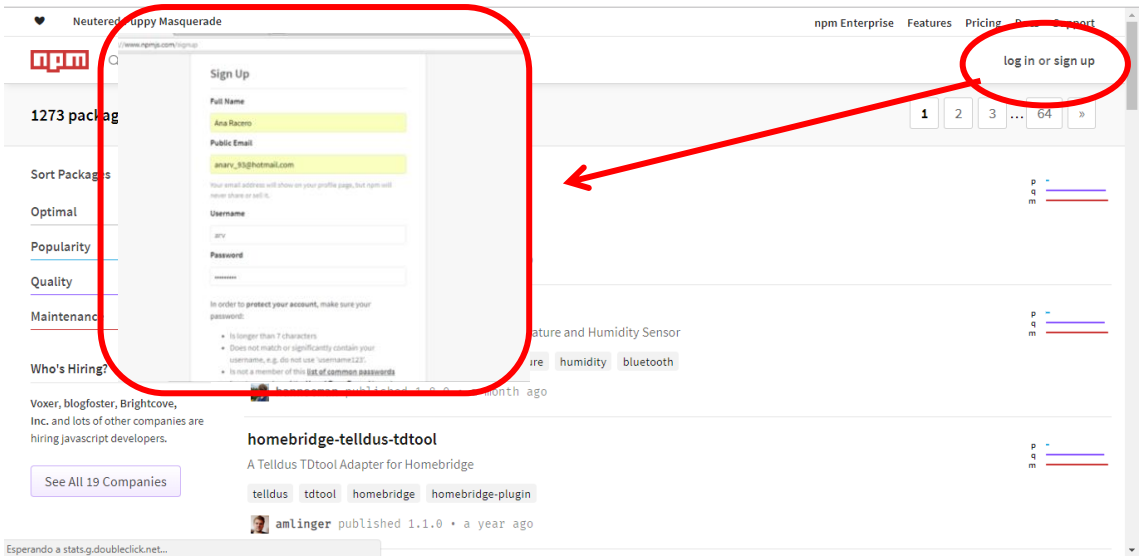

Figura 58. Interfaz web npm

Posteriormente hay que iniciar el usuario npm creado desde la Raspberry para ello ejecutar el comando `npm adduser`. Este comando pedirá el usuario, la contraseña y correo electrónico que se haya usado para darse de alta en npm.

Una vez registrado en npm y con la sesión iniciada en la Raspberry ya se puede publicar el nuevo módulo. Para ello se crea una nueva carpeta y se escribe el comando `npm init` en el terminal para que npm cree un nuevo fichero *package.json*. Hay que verificar que este fichero tenga una versión válida.

Cuando se tiene la carpeta con los archivos necesarios para la creación del módulo se ubica el puntero en la carpeta y ya únicamente se debe escribir en la línea de comando `npm publish`.

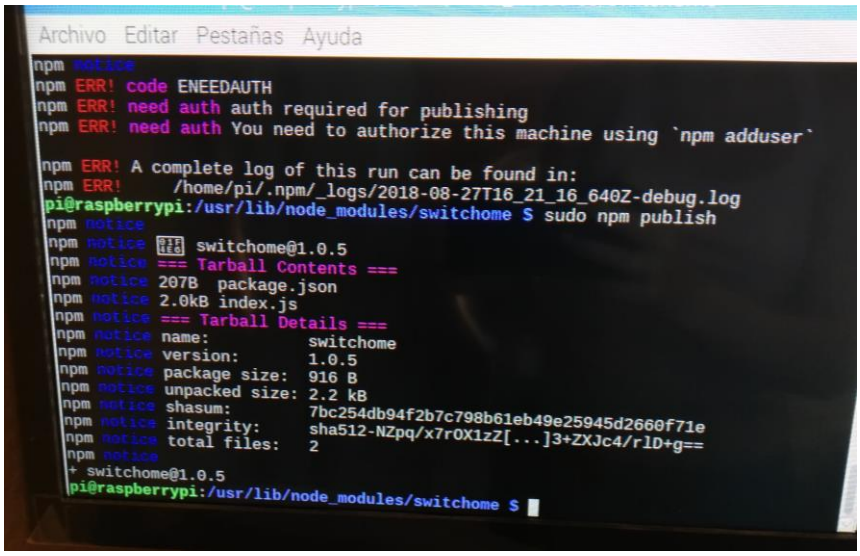

Figura 59. Publicación módulo npm

Si en un momento dado se quiere modificar el módulo ya que se necesite agregar funcionalidades o quitárselas, se debe volver a publicar para que los cambios queden guardados y para ello es necesario ir al fichero package.json y modificar la versión del módulo y volver a ejecutar el comando `npm publish`.

Si el módulo generado se desea que se esté presente en el sistema, al igual que cualquier otro plugin creado por otro usuario debemos ejecutar:

```
Sudo npm install -g 'nombredelmodulo'
```

En la siguiente imagen se puede ver el ejemplo realizado con el módulo creado switchome:

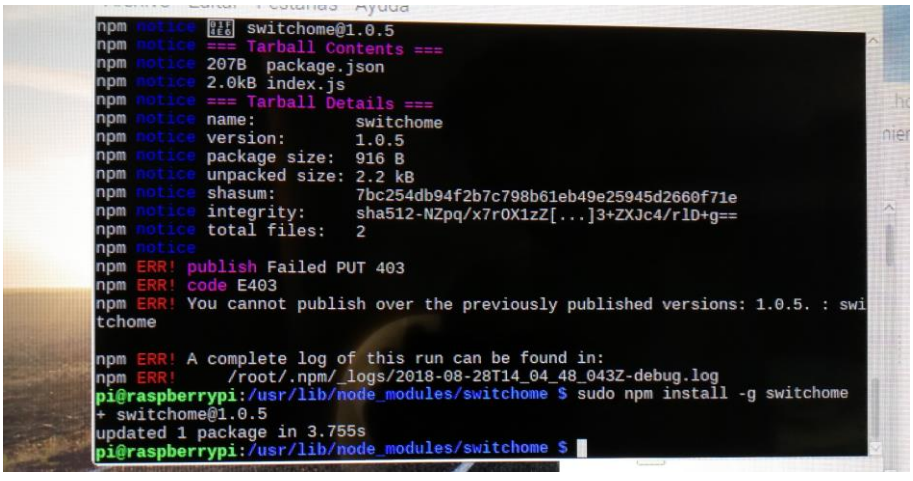

Figura 60. Instalación modulo npm

6.2 Ejemplo Belkin WeMo

En este apartado se va a desarrollar un ejemplo de conexión mediante Homebridge de unas luces inteligentes no compatibles con HomeKit. El pack consta de dos bombillas inteligentes y un WeMo Link para conectar las bombillas con la aplicación mediante la red wifi. El nombre predeterminado de la WeMo Link es Wemo.Link.xxx donde xxx son los últimos tres dígitos del número serie del dispositivo. Con el dispositivo Android conectamos a la red wifi de WeMo link, iniciamos la aplicación y ésta pedirá la red wifi de la vivienda así como la contraseña, y entonces el WeMo Link quedará enlazado con el wifi del hogar.

Otros productos WEMO incluyen el Interruptor WEMO Insight, interruptor WEMO y el kit WEMO Interruptor + Motion, todos ellos habilitados para operar con tecnología Wifi con la WEMO App gratuita para dispositivos Android y iOS. A través de la WEMO App es posible controlar múltiples dispositivos electrónicos, dándole una visibilidad inmediata de cada dispositivo conectado.

Las bombillas WeMo Smart LED proporcionan una luz cálida y brillante similar a una tradicional incandescente de 60 vatios, pero con una diferencia importante: sólo consumen 10 vatios de energía y producen muy poco calor. De esta forma, con estas bombillas se puede reducir la cantidad de energía que consumida, y por consiguiente, la factura de la luz.

Según el fabricante este kit de WeMo tiene una vida útil de 23 años basado en un uso promedio diario de 3 horas al día. Con estas bombillas también es posible regular la intensidad luminosa de las mismas directamente desde tu Smartphone

Este kit tiene una gran ventaja para cuando estamos fuera de casa ya que podemos simular la apariencia de ocupación en la casa cuando en realidad no hay nadie. Todos los dispositivos como explicaremos posteriormente pueden ser controlados para encenderlos y apagarlos automáticamente. Además, entre otras cosas, permite crear un programa de funcionamiento diario para tu iluminación doméstica. Puedes programar una de tus luces para que se encienda automáticamente al atardecer, reduzca su intensidad cuando ves una película o se apague cuando sales de casa, todo ello desde un Smartphone o Tablet.

Las tecnologías en general, pero sobre todo aquellas que disponen de conexión a internet, son vulnerables y necesitan una protección adecuada. La seguridad d estos dispositivos domótica es algo realmente importante. Detectaron un fallo en estos dispositivos WeMo en enero pero ha sido solucionado. Para tener estos fallos soluciones se debe actualizar la aplicación WeMo.

Figura 61. Kit Bombilla Belkin WeMo

Para ello descargar el plugin de la marca en la Raspberry ejecutamos el comando:

```
sudo npm install -g homebridge-platform-wemo
```

Al ejecutar ese comando se crea una carpeta la cual contiene el archivo index.js con la programación del módulo que se ha instalado y un archivo *package.json* con la información del módulo como versión, nombre...etc.

Una vez instalado el plugin de la marca comercial WeMo es necesario actualizar el archivo *config.json* perteneciente a la configuración que se le quiere dar a Homebridge. Para la configuración del archivo *config.json* se debe atender a las explicaciones que el creador da en el archivo README.txt. En este ejemplo se ha usado la configuración que se muestra a continuación en la cual se declara una bombilla inteligente Belkin WeMo. El nombre que se le asigna al accesorio debe ser el mismo que tenga que en la aplicación.

```
{
    "bridge": {
        "name": "Homebridge",
        "username": "CC:22:3D:E3:CE:30",
        "port": 51826,
        "pin": "031-45-154"
    },
    "description": "This is an example.",
    "accessories": [
    ],
    "platforms": [
                {
            "platform" : "BelkinWeMo",
            "name" : "mi cuarto 01"
        }
    ]
```

La declaración del dispositivo como accesorio o plataforma depende como lo haya declarado el creador del plugin.

En este caso el archivo *index.js* muestra la declaración de los accesorios de Belkin WeMo como plataforma.

Una vez instalado el plugin y actualizado el fichero *config.json* se reinicia la Raspberry y se vuelve a ejecutar Homebridge obteniendo así el accesorio "mi cuarto 01" en la aplicación Home de nuestro dispositivo Apple.

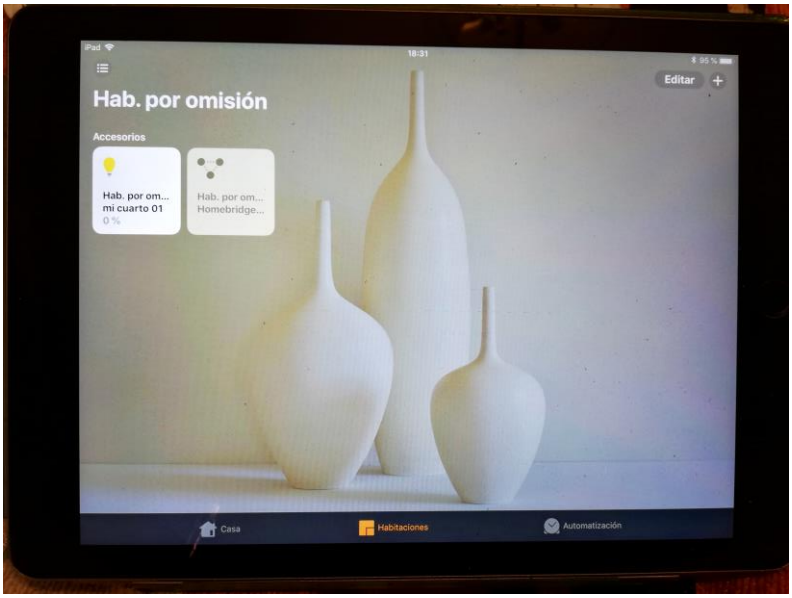

Figura 62. Interfaz de Home con accesorio Belkin

6.3 Plugin para persianas domotizadas

En este apartado se ha llevado a cabo la interconexión de persianas domóticas y la aplicación Home de Apple. Estas persianas usan el acceso a distintas urls para las funciones de subir, bajar o parar y no pueden hacerlo mediante ninguna aplicación que pueda ser compatible con Apple, debido a la incompatibilidad se ha usado el servidor Homebridge para lograr la conexión haciendo uso de un plugin de npm denominado homebrige-blinds.

Para instalar el plugin en nuestra Raspberry escribimos en la ventana de comando:

```
sudo npm install -g homebridge-blinds
```

Y como en todos los plugins de Homebridge, para que el dispositivo nos aparezca en Homebridge como accesorio, debemos actualizar el archivo *config.json* con la siguiente información que también es proporcionada por el creador del plugin:

```
{
    "bridge": {
        "name": "Homebridge",
        "username": "CC:22:3D:E3:CE:30",
        "port": 51826,
```

```
        "pin": "031-45-154"
    },
    "description": "This is an example.",
    "accessories": [
    ],
    "accesories": [
{
      "accessory": "BlindsHTTP",
      "name": "Window",
      "up_url": "http://1.2.3.4/window/up",
      "down_url": "http://1.2.3.4/window/down",
      "stop_url": "http://1.2.3.4/window/stop",
      "motion_time": "<time your blind needs to move from up to down (in mill
iseconds)>",
      "http_method": "PUT",
      "trigger_stop_at_boundaries": false
    }
    ]
```

El comando `trigger_stop_at_boundaries` es el que permite elegir si se debe usar un comando de parada o no al mover las persianas. En general las persianas se paran cuando llegan a su fin y si así es el comando debe ser igual a “false”.

Una vez realizado los pasos anteriores conectamos reiniciamos la Raspberry y ejecutamos Homebridge. En nuestro dispositivo Apple es conveniente eliminar Homebridge si ya lo tenemos instalado de antes y volverlo a instalar con los nuevos dispositivos que se hayan agregado ya que si no, puede que la aplicación Home no los detecte correctamente. En la siguiente figura se muestra la aplicación Home, tanto con las luces del apartado anterior como con las ventanas domóticas de éste:

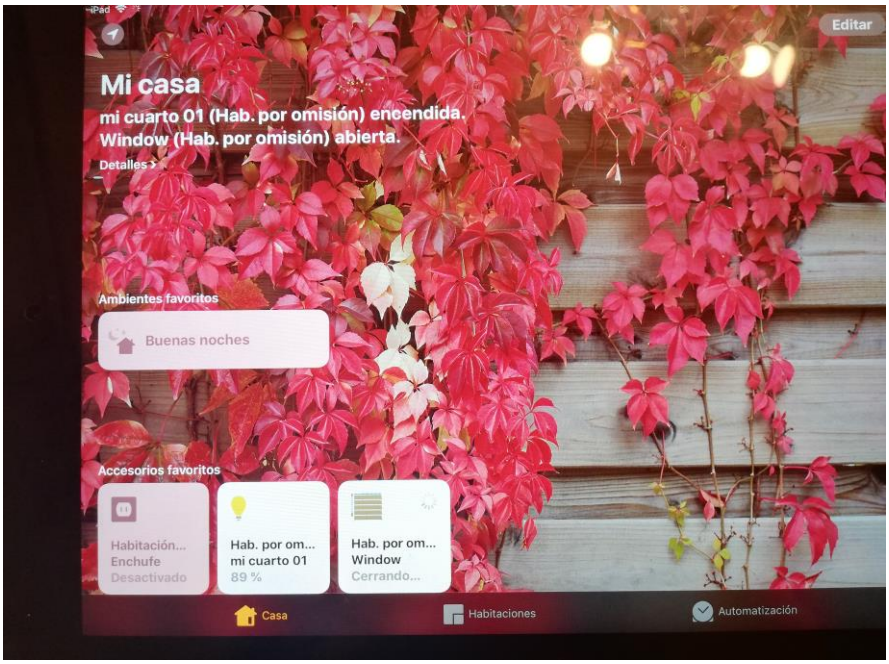

Figura 63. Interfaz Home con accesorio Belkin y persianas

6.4 Homebridge-roomba

En este apartado se va a desarrollar la instalación mediante Homebridge del robot domestico Roomba. Al igual que con el caso anterior Roomba ya tiene un plugin creado en el repositorio npm por lo que se hará uso del mismo para el desarrollo de este apartado.

Para la instalación de este plugin escribimos lo siguiente en la ventana de comando:

```
Sudo npm install -g homebridge-roomba
```

Una vez instalado el plugin para Homebridge es necesario confirmar la dirección IP a la que está conectado Roomba con la aplicación oficial de la misma. Abrimos el "iRobot Home App", seleccionar + → configuración → configuración de Wifi → Detalles de wifi del robot. Extraer la dirección IP (192.168.xxx.xxx).

Figura 64. Robot Roomba

El siguiente paso es obtener una contraseña y blid, para ello hay que ir al directorio donde se ha instalado el módulo, */usr/var/lib/node_modules,* entrar en el directorio *homebridge-roomba* y escribir en la ventana de comando:

```
Sudo npm run getpassword  <Roomba IP address>
```

Se mostrará el siguiente mensaje:

```
Make sure your robot is on the Home Base and powered on (green lights on).

Then press and hold the HOME button on your robot until it plays a series of
tones (about 2 seconds).

Release the button and your robot will flash WIFI light.

Then press any key here...
```

Si todo ha salido correctamente se mostrará el siguiente mensaje:

```
Robot Data:

{ ver: '2',

  hostname: 'Roomba-xxxxxxxxxxxxxxx',

  robotname: 'Your Roomba's Name',

  ip: '192.168.xx.xx',

  mac: 'xx:xx:xx:xx:xx:xx',

  sw: 'vx.x.x-x',

  sku: 'R98----',

  nc: 0,

  proto: 'mqtt',

  blid: '0123456789abcdef' }

Password=> :1:2345678910:ABCDEFGHIJKLMNOP <= Yes, all this string.
```

Con esta información actualizamos el archivo *config.json* de configuración de Homebridge:

```
{

    " plataforma " : " Roomba " ,

    " nombre " : " Roomba " ,

    " robots " : [

      {

        " nombre " : " Nombre del robot en la aplicación de inicio " ,
```

```
        " blid " : " 123456789abcdefg " ,
        " contraseña " : " : 1: 2345678901: ABCDEFGHIJKLMNOP " ,
        " dirección " : " 192.168.x.xx "
      }]}
```

El proceso puede fallar por varios motivos:

- Roomba y Homebridge no están conectados a la misma red wifi.
- Roomba no está cargado.
- Confirmar versión de node.js.

6.5 Hombrdige-eedomus

Eedomus es un controlador usado en el sistema domótico capaz de controlar accesorios mediante http. Eddomus es incompatible con Apple por lo que también tiene desarrollado un plugin para Homebridge de interconexión entre la Raspberry y Apple.

Para la instalación de este plugin escribimos lo siguiente en la ventana de comando:

```
Sudo npm install -g homebridge-roomba
```

Para la configuración del plugin hay que saber las credenciales del eedomus: ip, api_user y api_secret

Figura 65. Controlador eedomus

La configuración del archivo *config.json* requerida para este plugin es la siguiente:

```
{
    "bridge": {
        "name": "Homebridge",
```

```
        "username": "CC:22:3D:E3:CE:30",
        "port": 51826,
        "pin": "031-45-154"
    }
    "description": "This is an example.",

    "credentials": {
        "ip" : "your_local_eedomus_ip",
        "api_user" : "your_eedomus_api_user",
        "api_secret" : "your_eedomus_api_secret"
    },

    "accessories": [
       {
           "accessory": "eedomus",
           "name": "Devide name",
           "periph_id": "device API id",
           "service": "Switch"
       },
```

La limitación de este plugin se encuentra en que solo tiene 4 tipos de servicios disponibles para programar en *config.json*: Switch, WindowCovering, TemperatureSensor, Light.

6.6 Accesorio compatible con HomeKit

En este apartado, en contraposición con los anteriores, se va a desarrollar la instalación de un accesorio que si es compatible con HomeKit. Como se verá a continuación esta instalación de accesorios es mucho menos laboriosa pero los dispositivos que lo permiten son más caros.

El accesorio es un enchufe Koogek inteligente que al ser compatible con HomeKit trae en el envoltorio la certificación de Apple así como el código de activación del accesorio.

Figura 66. Accesorio Koogek

En primer lugar debemos conectar el enchufe a la red. Éste comenzará a parpadear con una luz roja que, pasados unos segundos, pasará a ser verde. En la pantalla inicio de home hacer clic en añadir accesorio y Home detectará el accesorio tal como se ve en la siguiente figura:

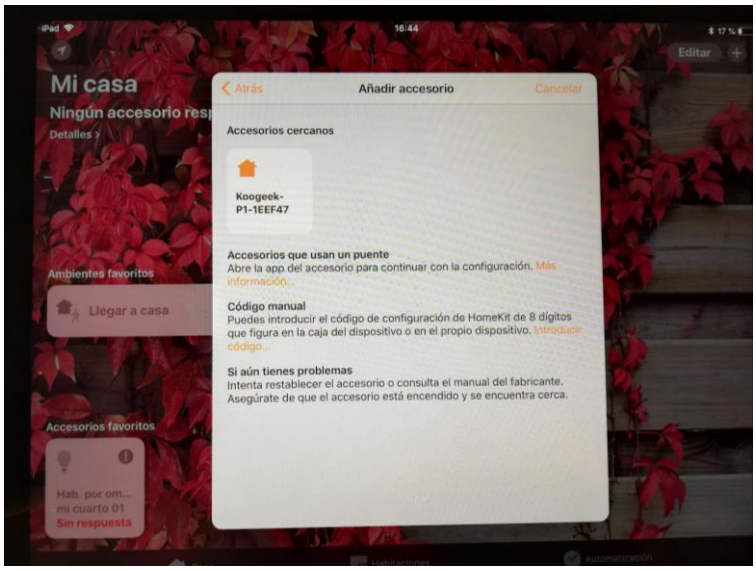

Figura 67. Instalación accesorio Koogek

Haciendo clic sobre el accesorio pedirá la clave de activación del accesorio y una advertencia para dar permiso de añadir el dispositivo a la red wifi.

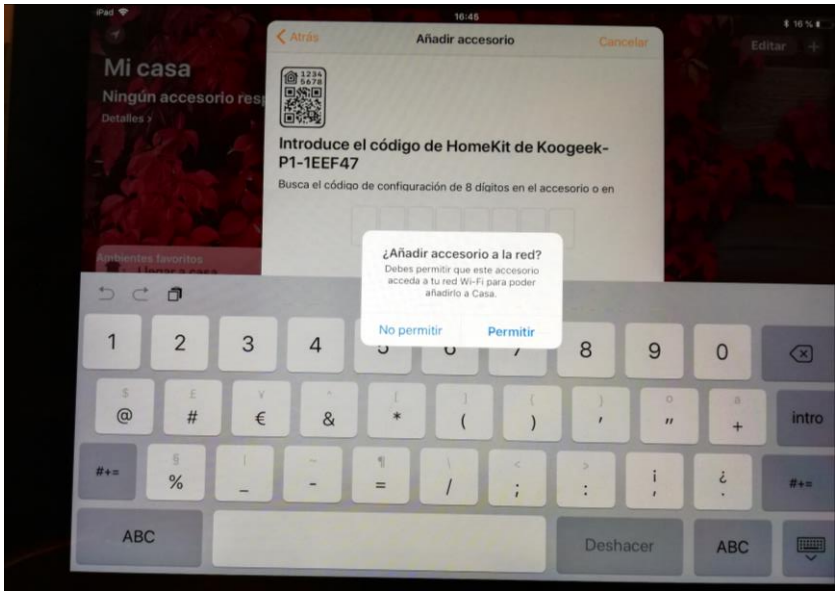

Figura 68. Instalación accesorio Koogek

Cuando se hayan realizado estos pasos tendremos nuestro accesorio listo pasa usar.

Si deseamos reiniciar el dispositivo de fábrica tiene un botón en la parte posterior para ello.

Además esta marca tiene su propia aplicación para el uso de sus dispositivos domóticos. Esta aplicación tiene una interfaz similar a la de Home como se puede ver en la siguiente imagen:

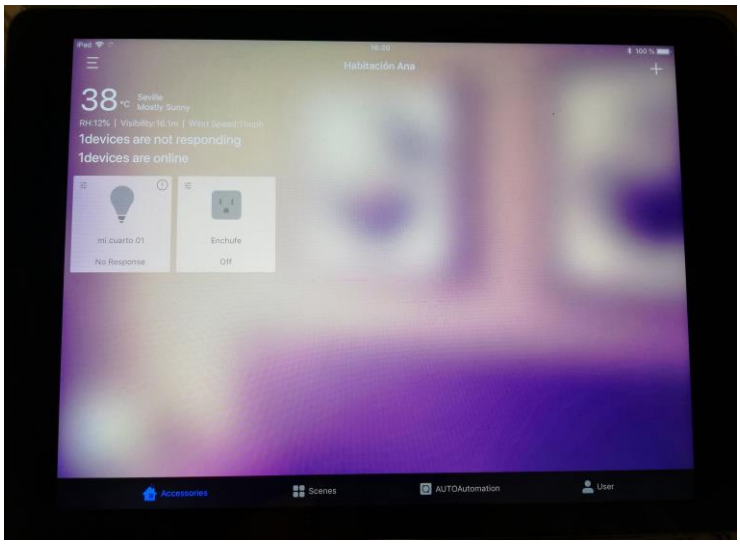

Figura 69. Interfaz Koogek

Además al estar integrada con HomeKit ambas aplicaciones están enlazadas y cualquier cambio que se produzca en los dispositivos desde cualquiera de las dos aplicaciones ambas guardarán ese cambio de estado del dispositivo. Además en dicha aplicación pueden aparecer también otros dispositivos que no

son de la marca pero que estén instalados en el dispositivo iOS.

Todas las automatizaciones que se tengan presente en la App Home también aparecen en esta aplicación y pueden ser modificadas desde ésta y Home también será modificada automáticamente.

6.7 Plugins más usados para Homebridge en domotica

Plugins

- ✓ **Cmd4**

  Es un complemento para Homebridge que te permite ver de qué se tratan Homebridge y HomeKit, sin tener realmente ningún accesorio

  Cmd4 viene con un archivo *config.json* completamente poblado y documentado que apunta a un archivo *State.js* completamente poblado y configurado que coloca en un subdirectorio Cmd4Scripts de su directorio .Homebridge. El complemento Cmd4 lee y comprende el archivo config.json que contiene todos los tipos posibles de HomeKit. Cuando señala HomeKit a Homebridge, todos los dispositivos se llenan, y entre el plugin homebridge-cmd4 y el archivo de comando State.js, HomeKit actúa y se comporta si realmente tiene el Accesorio que realmente no tiene.

- ✓ **homebridge-cmdswitch**

  Este complemento le permite ejecutar comandos de interfaz de línea de comandos (CLI) a través de HomeKit. Esto significa que puede ejecutar un comando simple como ping, shutdown o wakeonlan simplemente diciendo a Siri para hacerlo. Un ejemplo de uso de este complemento sería encender su PS4 o HTPC, verificar si está encendido e incluso apagarlo cuando termine.

- ✓ **homebridge-http**

  Admite dispositivos https en la plataforma Homebridge y proporciona una devolución de llamada legible para obtener las características de "Encendido" y nivel de brillo en HomeKit.

- ✓ **homebridge-tplink-smarthome**

  Complemento dedicado para accesorios Tp-Link. Permite conectar los accesorios tplink a la aplicación HomeKit sin necesidad de la aplicación Tp-Link.

- ✓ **homebridge-better-http-rgb**

  Este plugin controla luces led mediante encendido, apagado y control de luminosidad a través de https.

- ✓ **Homebridge-alexa**

  Soporte completo para todos los dispositivos Amazon Alexa, incluidas las soluciones echo 2nd Generation y basadas en software.

- ✓ **homebridge-nest-cam**

  Con este plugin es posible usar una Nest Cam como cámara IP en HomeKit con Homebridge.

ANEXO A

Código de arranque automático Homebridge.

```
#!/bin/sh
### BEGIN INIT INFO
# Provides:
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Start daemon at boot time
# Description:       Enable service provided by daemon.
### END INIT INFO

dir="/home/homebridge"
cmd="sudo /usr/bin/homebridge"
user="homebridge"

name=`basename $0`
pid_file="/var/run/$name.pid"
stdout_log="/var/log/$name.log"
stderr_log="/var/log/$name.err"

get_pid() {
    cat "$pid_file"
}

is_running() {
    [ -f "$pid_file" ] && ps -p `get_pid` > /dev/null 2>&1
}

case "$1" in
    start)
    if is_running; then
        echo "Already started"
    else
        echo "Starting $name"
        cd "$dir"
        if [ -z "$user" ]; then
            sudo $cmd >> "$stdout_log" 2>> "$stderr_log" &
        else
            sudo -u "$user" $cmd >> "$stdout_log" 2>> "$stderr_log" &
        fi
        echo $! > "$pid_file"
        if ! is_running; then
            echo "Unable to start, see $stdout_log and $stderr_log"
            exit 1
        fi
    fi
    ;;
```

```
    stop)
    if is_running; then
        echo -n "Stopping $name.."
        kill `get_pid`
        for i in 1 2 3 4 5 6 7 8 9 10
        # for i in `seq 10`
        do
            if ! is_running; then
                break
            fi

            echo -n "."
            sleep 1
        done
        echo

        if is_running; then
            echo "Not stopped; may still be shutting down or shutdown may
have failed"
            exit 1
        else
            echo "Stopped"
            if [ -f "$pid_file" ]; then
                rm "$pid_file"
            fi
        fi
    else
        echo "Not running"
    fi
    ;;
    restart)
    $0 stop
    if is_running; then
        echo "Unable to stop, will not attempt to start"
        exit 1
    fi
    $0 start
    ;;
    status)
    if is_running; then
        echo "Running"
    else
        echo "Stopped"
        exit 1
    fi
    ;;
    *)
    echo "Usage: $0 {start|stop|restart|status}"
    exit 1
    ;;
esac

exit 0
```

ANEXO B

Código para archivo index.js

```
mySwitch.prototype = {
  getServices: function () {
    let informationService = new Service.AccessoryInformation();
    informationService
      .setCharacteristic(Characteristic.Manufacturer, "My switch
manufacturer")
      .setCharacteristic(Characteristic.Model, "My switch model")
      .setCharacteristic(Characteristic.SerialNumber, "123-456-789");

    let switchService = new Service.Switch("My switch");
    switchService
      .getCharacteristic(Characteristic.On)
        .on('get', this.getSwitchOnCharacteristic.bind(this))
        .on('set', this.setSwitchOnCharacteristic.bind(this));

    this.informationService = informationService;
    this.switchService = switchService;
    return [informationService, switchService];
  }
};

const request = require('request');
const url = require('url');

function mySwitch(log, config) {
  this.log = log;
  this.getUrl = url.parse(config['getUrl']);
  this.postUrl = url.parse(config['postUrl']);
}

mySwitch.prototype = {

  getSwitchOnCharacteristic: function (next) {
```

```
    const me = this;
    request({
        url: me.getUrl,
        method: 'GET',
    },
    function (error, response, body) {
      if (error) {
        me.log('STATUS: ' + response.statusCode);
        me.log(error.message);
        return next(error);
      }
      return next(null, body.currentState);
    });
  },

  setSwitchOnCharacteristic: function (on, next) {
    const me = this;
    request({
      url: me.postUrl,
      body: {'targetState': on},
      method: 'POST',
      headers: {'Content-type': 'application/json'}
    },
    function (error, response) {
      if (error) {
        me.log('STATUS: ' + response.statusCode);
        me.log(error.message);
        return next(error);
      }28
      return next();
    });
  }
};
```

BIBLIOGRAFÍA

[1] http://domotica1003.weebly.com/medios-de-transmisionbus.html. [En línea].

[2] «https://es.wikipedia.org/wiki/Dom%C3%B3tica,» [En línea].

[3] «http://www.estamoscreandovida.com/airos-el-sistema-que-permite-a-los-aisoy1-sentir-y-emocionarse/,» [En línea].

[4] «CasaDomo,» [En línea]. Available: http://www.casadomo.com.

[5] «Control Domótico,» [En línea]. Available: https://riunet.upv.es/bitstream/handle/10251/28786/memoria.pdf?sequence=1.

[6] «HomeKit,» [En línea]. Available: https://mostly-tech.com/tag/homekit/.

[7] «HomeKit,» [En línea]. Available: https://www.imore.com/homekit-ios-8-explained.

[8] «Apple Coding,» [En línea]. Available: https://applecoding.com/wp-content/uploads/2016/01/apple-homekit-security-risk1.jpg.

[9] «HomeKit,» [En línea]. Available: https://www.imore.com/homekit-faq.

[10] «Apple Developer,» [En línea]. Available: https://developer.apple.com/design/human-interface-guidelines/homekit/overview/experience/.

[11] «Distrito Apple,» [En línea]. Available: http://www.distritoapple.com/aplicaciones-mac/home-apple-te-ayuda-controlar-hogar/.

[12] «HomeKit,» [En línea]. Available: https://www.apple.com/es/shop/accessories/all-accessories/homekit.

[13] «Accesibilidad Apple,» [En línea]. Available: https://www.imore.com/accessibility-iphone-ipad.

[14] «Domótica,» [En línea]. Available: http://blog.macnificos.com/apple-homekit/reviews/domotica.

[15] «HomeKit,» [En línea]. Available: https://medium.com/@vincecoetzee/home-alone-with-homekit-part-i-91667264f563.

[16] «Apple Developer,» [En línea]. Available: https://developer.apple.com/documentation/homekit.

[17] «iOS Security Guide,» [En línea]. Available: http://www.joinjp.com/mx/business/site/docs/iOS_Security_Guide.pdf.

[18] «iPhoneros,» [En línea]. Available: https://iphoneros.com/63358/apple-actualiza-el-logo-mfi-que-encontraras-en-accesorios-para-iphone-certificados.

[19] «DigitalTrends,» [En línea]. Available: https://www.digitaltrends.com/home/ios-9-apple-homekit-icloud/.

[20] «HomeKit,» [En línea]. Available: https://developer.apple.com/library/archive/documentation/NetworkingInternet/Conceptual/HomeKitDeveloperGuide/RespondingtoHomeKitDatabaseChanges/RespondingtoHomeKitDatabaseChanges.html#//apple_ref/doc/uid/TP40015050-CH5-SW2.

[21] «Apple Developer,» [En línea]. Available: https://developer.apple.com/library/archive/documentation/General/Conceptual/DevPedia-CocoaCore/Delegation.html#//apple_ref/doc/uid/TP40008195-CH14-SW2.

[22] «HomeKit,» [En línea]. Available: https://medium.com/@vincecoetzee/home-alone-with-homekit-part-i-91667264f563.

[23] B. Developer. [En línea]. Available: https://developer.apple.com/bonjour/.

[24] «Protocol HomeKit,» [En línea]. Available: https://www.quora.com/What-protocol-does-HomeKit-use-to-communicate-with-its-devices-How-does-it-discover-devices-over-Wi-Fi.

[25] «Airplay,» [En línea]. Available: https://www.xatakahome.com/servicios-y-aplicaciones-de-audio/protocolo-airplay-2-esta-aqui-integrando-sonido-multiroom-ecosistema-apple.

[26] «Blog Teslabem,» [En línea]. Available: https://teslabem.com/blog/historia-de-raspberry-pi/.

[27] «Blog LuisLlamas,» [En línea]. Available: https://www.luisllamas.es/modelos-de-raspberry-pi/ .

[28] «Sistemas Operativos,» [En línea]. Available: https://www.1and1.es/digitalguide/servidores/know-how/10-sistemas-operativos-para-raspberry-pi/ .

[29] «wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Raspbian.

[30] «Blog arpentechnologies,» [En línea]. Available: https://arpentechnologies.com/es/blog/software/que-es-kali-linux/ .

[31] «Blog,» [En línea]. Available: https://i.blogs.es/bb8f8e/win10iot3/450_1000.jpg .

[32] «Web,» [En línea]. Available: http://www.creditlenders.info/wp-content/uploads/snappy-logos-ubuntu-core-ubuntu-design-blog.png.

[33] «Silicon,» [En línea]. Available: https://www.silicon.es/wp-content/uploads/2012/11/raspberry.jpg .

[34] «Blog Petrockblock,» [En línea]. Available: http://blog.petrockblock.com/wp-content/uploads/2015/06/RetroPieLogo2015Download.png .

[35] «Logo ArchLinux ARM,» [En línea]. Available: https://3.bp.blogspot.com/-Fx3b_2yv8-k/URjSEOM_noI/AAAAAAAAKEE/ZwvY89DAaT8/s1600/Arch+Linux+ARM.png .

[36] «HomeKit and Homebridge,» [En línea]. Available: https://www.filipeflop.com/blog/automacao-residencial-com-apple-homekit-homebridge-parte-1/.

[37] «Geeky-gadgets,» [En línea]. Available: https://www.geeky-gadgets.com/diy-homekit-enabled-projects-11-04-2018/.

[38] «Gestor módulos,» [En línea]. Available: https://www.jmramirez.pro/tutorial/npm-gestor-modulos-node-js/.

[39] «Npm Web,» [En línea]. Available: https://docs.npmjs.com/getting-started/packages.

[40] «Entorno node.js,» [En línea]. Available: https://www.jmramirez.pro/tutorial/instalacion-del-entorno-node-js/.

[41] «Instalación node,» [En línea]. Available: https://www.jmramirez.pro/tutorial/instalacion-del-entorno-node-js/.

[42] «Introducción Node,» [En línea]. Available: http://juanmirod.github.io/2017/08/09/introduccion_nodejs.html.

[43] Homebridge. [En línea]. Available: https://worthen.media/homebridge-server-on-raspberry-pi/.

[44] «HomeKit Controles,» [En línea]. Available: https://www.applesfera.com/tutoriales/como-compartir-los-controles-de-homekit-con-tus-invitados-en-casa.

[45] «HomKit and Siri,» [En línea]. Available: https://blog.theodo.fr/2017/08/make-siri-perfect-home-companion-devices-not-supported-apple-homekit/.

[46] «Wikipedia,» [En línea]. Available: https://es.wikipedia.org/wiki/Raspberry_Pi.

[47] «Creación modulos npm,» [En línea]. Available: https://medium.com/@peraferrer/como-

crear-un-m%C3%B3dulo-npm-6baef161a96.

[48] «Raspberry,» [En línea]. Available: https://opensource.com/resources/raspberry-pi.

[49] «WiKipedia Domótica,» [En línea]. Available: https://es.wikipedia.org/wiki/Dom%C3%B3tica.