Escuela Politécnica Superior de Jaén

Grado en Ingeniería Informática

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Jaén
Departamento de Informática

Trabajo Fin de Grado

Análisis y despliegue de una plataforma IoT para el Smart Lab del CEATIC

Alumno: José Francisco Gay Medina

Tutores: Dra. Macarena Espinilla Estévez
D. Daniel Zafra Romero

Dpto: Informática

Septiembre, 2018

Análisis y despliegue de una plataforma IoT para el Smart Lab del CEATIC

José Francisco Gay Medina

Septiembre 2018

Universidad de Jaén
Escuela Politécnica Superior de Jaén
Departamento de Informática

Dra. Macarena Espinilla Estévez, tutora,
y *D. Daniel Zafra Romero*, co-tutor,
del Trabajo Fin de Grado titulado:

Análisis y despliegue de una plataforma IoT para el Smart Lab del CEATIC,

que presenta *D. José Francisco Gay Medina*,
autorizan su presentación para defensa y evaluación en la
Escuela Politécnica Superior de Jaén.

Jaén, septiembre de 2018

El alumno:	Los tutores:
José Francisco Gay Medina	Macarena Espinilla Estévez
	Daniel Zafra Romero

"Dar las gracias es la forma más elevada de pensamiento;
ser agradecido, es la mejor forma de felicidad"
Gilbert Keith Chesterton

Agradecimientos

Este trabajo ha sido posible gracias a todas las personas que, a lo largo de este tiempo, no solo he tenido el privilegio de conocer sino también de trabajar junto a cada uno de ellos.

Para comenzar, me gustaría ensalzar la labor de mis tutores, Macarena Espinilla Estévez y Daniel Zafra Romero, a quienes no solo agradezco la ayuda prestada, la atención constante y el notable esfuerzo sino que también me gustaría dedicarles el presente trabajo. Macarena, gracias por descubrirme el maravilloso mundo del IoT y la inteligencia ambiental.

A mi familia, sin vosotros, nada de esto hubiera sido posible. Gracias por apoyarme incondicionalmente, gracias por levantarme cuando me he caído y gracias por el aliento que me habéis dado siempre y que estoy seguro que me lo seguiréis dando.

A mis amigos, a todos ellos, gracias por enseñarme tantas y tantas cosas en mi día a día. Especialmente a María, pero también a Gloria, a Francisco Javier, y por supuesto a los cinco mejores compañeros de clase que uno puede tener. Gracias por formar parte de estos maravillosos cuatro años. Sin vuestro apoyo, tampoco hubiera sido posible.

GRACIAS.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Justificación

En septiembre de 2015 fue inaugurado el Smart Lab de inteligencia ambiental, denominado actualmente UJAmI (University of Jaén and Ambient Intelligent) [1], del Centro de Estudios Avanzados en Tecnologías de la Información y Comunicación (CEATIC) [2].

Dicho Smart Lab es un espacio donde los objetos cotidianos están conectados en red, siendo capaces de recoger información, procesarla y compartirla. Los recientes avances en investigación permiten a los objetos ser capaces de tomar decisiones por sí mismos, a partir de la información que faciliten otros aparatos o sensores con el fin de facilitar o mejorar la vida a los ocupantes.

En el Smart Lab subyacen dos conceptos emergentes que en los últimos años están transformando la forma de entender la tecnología. El primero es el Internet de las Cosas [3], un paradigma donde los objetos cotidianos se encuentran conectados a la red, permitiendo interactuar y comunicarse entre ellos. El segundo, es el concepto de ambiente inteligente [4] donde el entorno que nos rodea se adapta a las personas que lo habitan gracias a los múltiples dispositivos de sensores y actuadores, los protocolos de comunicación y, finalmente, los procesos de

razonamiento.

En una superficie aproximada de $25m^2$ se desarrolló el Smart Lab [5] donde se encuentran distribuidas las zonas habituales de una vivienda: un recibidor, un salón con puesto de trabajo, una cocina y un dormitorio con un aseo integrado. Además del equipamiento tradicional de una vivienda como el televisor, los electrodomésticos o la cama, se han incorporado múltiples dispositivos inteligentes como altavoces o luces controlables de forma remota y dispositivos wearables, que permiten, por ejemplo, monitorizar el sueño, las señales vitales o las emociones.

Son inagotables las aplicaciones que pueden llevarse a cabo en el Smart Lab. Las aplicaciones más prometedoras son aquellas centradas en la asistencia tecnológica [6], que permitan apoyar a las personas en sus actividades diarias, facilitar y mejorar la experiencia de uso, trabajo, juego o vida en general y aquellas capaces de proporcionar información relevante en el momento y lugar necesario para tomar las decisiones adecuadas en tiempo real.

Inicialmente, la comunicación de la información de algunos dispositivos fue integrándose en una plataforma IoT de desarrollo propio denominada "Sistema Web para la Monitorización de Ambientes Inteligentes" [7]. Dicha plataforma IoT posee características de investigación muy interesantes relacionadas con el campo de investigación de reconocimiento de actividades [8]. Sin embargo, presenta limitaciones desde el punto de vista de compatibilidad con dispositivos, módulos de terceros que incrementen su funcionalidad, comunicación con plataformas de terceros, etc.

Actualmente, existe en el mercado un amplio abanico de plataformas IoT consolidadas que facilitan la integración de múltiples dispositivos. Este aspecto es relevante debido al gran número de dispositivos inteligentes y protocolos de comunicación que se encuentran presentes en el Smart Lab y que va aumentado año tras año.

Por tanto, ante la constante aparición y adquisición de nuevos dispositivos, así

como de nuevos protocolos de comunicación relacionados con el paradigma de IoT, es imprescindible desplegar una plataforma IoT en el Smart Lab que permita tanto la gestión y configuración de dispositivos como la recolección y procesamiento de los datos generados por estos dispositivos.

Además, no podemos olvidar que el Smart Lab del CEATIC se encuentra en un contexto académico que nace con la especial vocación de investigar y desarrollar propuestas y aplicaciones que mejoren la calidad de vida de las personas y que puedan ser transferidas a otros Smart Labs académicos con el mismo fin. Así, la plataforma IoT deberá de contar con unas necesidades específicas de investigación de acuerdo a las estrategias propias del CEATIC.

1.2. Propuesta

El objetivo principal de este trabajo fin de grado es seleccionar y desplegar una Plataforma IoT para el Smart Lab del CEATIC, considerando los dispositivos inteligentes que alberga el Smart Lab y sus necesidades específicas de recolección y procesamiento de datos.

1.3. Objetivos

De acuerdo a la propuesta de este trabajo fin de grado, se plantean los siguientes objetivos.

1. Revisar los conceptos de Internet de las Cosas, Inteligencia Ambiental y Smart Labs.

2. Analizar las distintas plataformas IoT existentes y los dispositivos inteligentes que alberga el CEATIC.

3. Analizar la compatibilidad de los dispositivos inteligentes del CEATIC y las

plataformas IoT existentes para seleccionar la plataforma IoT que mayor dispositivos integre.

4. Desplegar la plataforma IoT seleccionada en el Smart Lab del CEATIC.
5. Implementar la funcionalidad adicional en la plataforma IoT desplegada para cubrir las necesidades específicas del CEATIC.
6. Realizar los manuales asociados.
7. Redactar una memoria que recoja todo el trabajo desarrollado.

1.4. Planificación del trabajo

En este apartado se describe la planificación del trabajo, tanto desde el punto de vista del tiempo necesario para su realización como de los costes que se requerirán para poder llevar a cabo su desarrollo.

1.4.1. Estimación de tiempos

Para este propósito vamos a tener en cuenta una estimación optimista de la duración para cada una de las tareas que tendremos que abarcar a lo largo del desarrollo de este proyecto. Para ello hemos realizado la siguiente tabla que ilustra cada una de las tareas y el tiempo estimado para las mismas, dando como resultado un tiempo de aproximadamente 5 meses (800 horas a 8 horas por día) para poder llevar a cabo la propuesta del proyecto.

Tarea	Horas
Tarea 1. Revisión de IoT, inteligencia ambiental y smart labs	60
Tarea 2. Análisis de plataformas IoT	80
Tarea 3. Análisis de dispositivo IoT del Smart Lab	50
Tarea 4. Elección de plataforma IoT	10
Tarea 5. Análisis en profundidad de la plataforma IoT y necesidades	170
Tarea 6. Despliegue de plataforma IoT	55
Tarea 7. Análisis, despliegue y configuración de módulos	65
Tarea 8. Análisis, diseño y desarrollo de funcionalidad adicional	40
Tarea 9. Redacción de memoria y manuales	270
Total	800

Tabla 1.1: Estimación de tiempos

Diagrama de Gantt

Un diagrama de Gantt es una representación gráfica de las tareas que constituyen un proyecto y su duración a lo largo de un periodo determinado de tiempo. Es una muy buena herramienta para controlar el seguimiento de las tareas y comprobar si estamos cumpliendo con el progreso estimado del proyecto, como se puede observar en la Figura 1.1.

1.4.2. Análisis de costes

Realizar un estudio sobre el presupuesto que vamos a necesitar para llevar a cabo un proyecto es una fase obligatoria en la etapa de planificación. Además, en la mayor parte de los casos puede resultar una parte fundamental para poder llevar a cabo el proyecto, ya que si el coste total estimado es muy alto puede suponer la cancelación del proyecto. En el presupuesto que podemos observar en la Tabla 1.2 se valoran económicamente los costes que se necesitan para llevar a

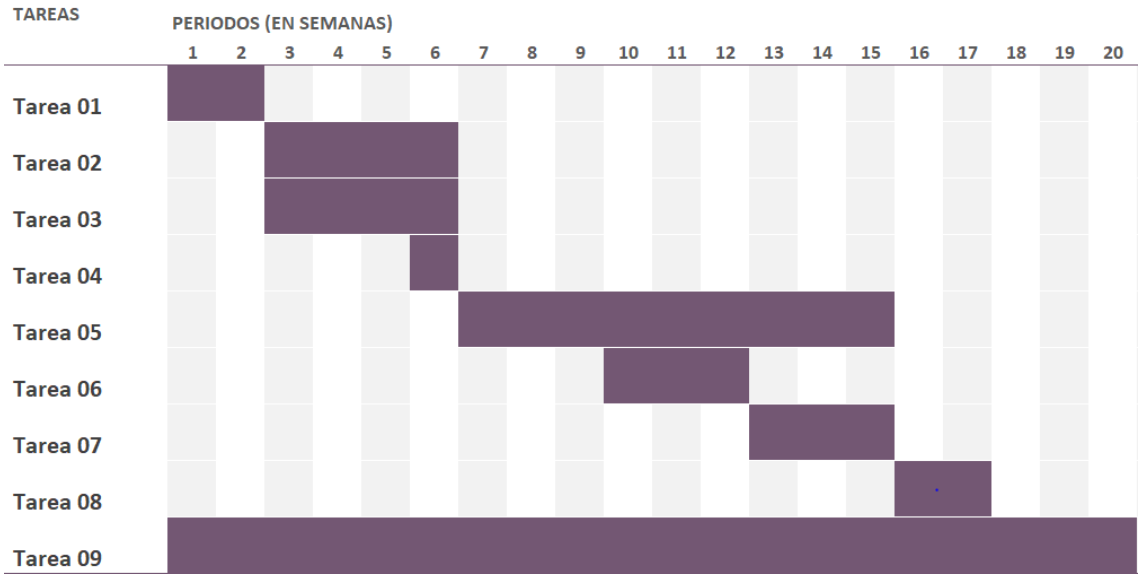

Figura 1.1: Diagrama de Gantt

cabo el desarrollo, y en él se incluyen tanto los costes de hardware y licencias de software como los costes de mano de obra de los desarrolladores. En dicho coste, no se incluirán los dispositivos que alberga el Smart Lab del CEATIC.

Tarea	Salario mensual	Meses	Coste total
Hardware	-	-	711,15€
Análisis y despliegue	1.181,17€	5	5.905,85€
Licencias software	-	-	12,99€
Coste bruto			6.629,99€
IVA (21 %)			1.392,29€
Coste total			8.022,28€

Tabla 1.2: Análisis de costes

Por tanto, el presupuesto final para llevar a cabo nuestro proyecto es de 8.022,28€ aproximadamente, pero debemos tener en cuenta que la mayor parte de dicho presupuesto se destinaría a la mano de obra necesaria para su análisis y despliegue. El coste del mantenimiento de la plataforma IoT, una vez desplega-

da, alcanzaría el valor de 390,00€ por mes, que corresponde a 5 horas de trabajo para su actualización y testeo por semana.

1.5. Estructura de la memoria

Esta sección se dedica a describir la estructura del trabajo, de forma que el lector tenga una visión general de los contenidos que se verán en los sucesivos capítulos:

- **Capítulo I: Introducción.** En este capítulo se describen los motivos que nos han llevado a abarcar este proyecto, la propuesta, los objetivos específicos que nos hemos marcado y también la planificación del trabajo a realizar hasta su conclusión.

- **Capítulo II: Preliminares.** Este capítulo está dedicado a introducir el tema central del proyecto y, por tanto, veremos todas las bases teóricas necesarias, las cuales cubren los conceptos de Internet de las Cosas, Ambientes Inteligentes y Smart Labs.

- **Capítulo III: Análisis y elección de la plataforma IoT**. En el siguiente capítulo de este trabajo se va a realizar un análisis de las plataformas IoT actuales, considerando un punto de vista genérico y un punto de vista orientado a investigación. Posteriormente, se analizarán el conjunto de dispositivos inteligentes que alberga el Smart Lab del CEATIC, considerando sus protocolos de comunicación y su integración en las plataformas IoT analizadas. Finalmente, en base a los análisis realizados, se escogerá la plataforma a desplegar en Smart Lab del CEATIC.

- **Capítulo IV: Despliegue de plataforma IoT**. En este capítulo se exponen las necesidades específicas a cubrir por la plataforma IoT y se presenta

el esquema de la plataforma IoT a desplegar, sus componentes y la funcionalidad a implementar, de acuerdo a dichas necesidades indicadas.

- **Capítulo V: Conclusiones y línea de trabajo futura**. Para finalizar haremos un breve recopilatorio de los aspectos más importantes que hemos estudiado y realizado a lo largo del trabajo. Finalmente expondremos la posible ampliación que se pueda llevar a cabo.

- **Anexo I. Manual de instalación y de mantenimiento**. En este anexo se describe el proceso de instalación de la plataforma IoT desplegada junto al proceso de mantenimiento que requiere la plataforma.

Capítulo 2

Preliminares

2.1. Introducción

Uno de los objetivos principales de este trabajo es desplegar una plataforma IoT en el Smart Lab de inteligencia ambiental del CEATIC de la Universidad de Jaén e incluir las funcionalidades necesarias de acuerdo a sus necesidades específicas. El Smart Lab del CEATIC es un smart lab académico que nace con la especial vocación de investigar y desarrollar propuestas y aplicaciones que mejoren la calidad de vida de las personas. Para ello, dicho Smart Lab se enfoca en la Inteligencia Ambiental bajo el paradigma de Internet de las Cosas. Al ser un Smart Lab académico centrado en el usuario final, es esencial que la plataforma IoT que se despliegue pueda ofertar ciertas características y funcionalidades que faciliten compartir la información entre ellos.

Este capítulo ofrece una visión general sobre IoT e Inteligencia Ambiental, describiendo los pilares sobre los que se fundamenta: dispositivos inteligentes, protocolos y estándares de comunicación, así como plataformas IoT. Además, se proporciona una perspectiva de los principales Smart Lab académicos con los que interacciona el Smart Lab del CEATIC para finalmente indicar las principales propuestas y aplicaciones que se desarrollan en ellos.

2.2. Internet de las Cosas

Desde sus inicios, Internet ha ido incorporando nuevos actores conforme los avances tecnológicos lo han permitido. Primero fueron los computadores, después los dispositivos móviles (smartphones) [9], más tarde las propias personas a través de las redes sociales [10] y, recientemente, los objetos cotidianos, en lo que se ha denominado el paradigma de Internet de las Cosas o *IoT* [3] derivado de sus siglas en inglés (Internet of Things).

Bajo este paradigma, se engloba que todos los objetos cotidianos más comunes se pueden conectar a Internet debido a los grandes avances tecnológicos de los últimos tiempos, ofreciendo nuevos servicios a los usuarios.

Aunque el término se acuñó *a posteriori*, el Internet de las Cosas tiene su origen en un artículo para la revista Scientific American titulado "El computador del siglo XXI" de Mark Weiser en 1991 [11]. A partir de ese artículo, Weiser vaticinaba que se utilizaría en el próximo siglo en las oficinas, hogares y entornos exteriores del futuro con el fin de optimizar recursos, utilizarlos de una forma más eficiente en pos de mejorar la experiencia del usuario. Esta relación entre entorno humano y máquina fue bautizada por Weiser como computación ubicua y lo que quería contextualizar con ello era la incrustación de la computación en el día a día de las personas.

Bajo el pilar fundamental de la computación ubicua, se fundamenta el IoT, aunque en sus inicios, no era más que una apuesta de futuro. Es en la actualidad cuando comienza a ser una realidad que se consolidad día tras día. La primera vez que el término IoT emergió [12], fue por el investigador británico del MIT (Massachusetts Institute of Technology) Kevin Ashton, en 1999, cuando intentaba describir un sistema en el que los objetos se podían conectar a Internet por medio de sensores.

Además de los objetos más cotidianos que se han adaptado para poder recibir

una conexión a Internet, son muchos los dispositivos que han surgido en los últimos años. Plataformas como Arduino [13] se han impuesto en la sensorística de bajo coste. Por supuesto, son muchas las empresas que están invirtiendo ingentes cantidades de dinero y esfuerzos en el desarrollo de dispositivos y en la innovación en los mismos, siendo algunas de las más importantes: CISCO, IBM, Intel, Google, Microsoft, Oracle, Qualcomm o AWS.

De la misma manera que van surgiendo dispositivos a gran velocidad, se van adoptando también nuevos estándares de comunicación para realizar la transferencia de información entre ellos; protocolos como: Wi-Fi, GSM, 6LowPan, Bluetooth BLE, Zigbee o Z-Wave son algunos de los más utilizados en la actualidad.

A día de hoy, el IoT se basa en la interconexión de múltiples objetos a través de una plataforma con el fin de que los objetos puedan tomar decisiones adecuadas de forma automática, sin intervención humana, considerando el estado de otros objetos [12].

Cuando las decisiones de los objetos de manera automática en un entorno están especialmente orientadas a la mejora de la calidad de vida de las personas en base a sus necesidades, gustos y preferencias, se define dicho entorno en el área de inteligencia ambiental o ambientes inteligentes [14]. Dicho área de investigación se revisa en la siguiente sección.

2.3. Inteligencia Ambiental

Una de las citas más comunes de ambientes inteligentes la encontramos en la definición proporcionada por Alan Steventon y Steve Wright [14]: *Los ambientes inteligentes son sistemas en los que la computación es usada para mejorar las actividades comunes.* Desglosando dicha definición, se aprecia que el principal objetivo es incluir a la tecnología para realizar nuestras tareas diarias de forma inmersa.

Por dicho motivo es habitual que se relacione ambientes inteligentes con domótica [15], un concepto relacionado, aunque más primitivo y menos ambicioso. La domótica utiliza un conjunto de técnicas para automatizar una vivienda o entorno, pero si utilizamos estas técnicas junto con una lógica que aprenda y mejore la estancia del habitante, entonces se deriva en un *Ambiente Inteligente*. A lo largo de esta memoria también podemos encontrarnos esta definición como Entorno de inteligencia ambiental.

Recapitulando, nos encontramos por una parte, ante un entorno cambiante, y que por otra parte, se compone de dispositivos de diversa índole, cada uno con funcionalidades y requisitos específicos. Por tanto, es necesario buscar el paraguas que nos proporciona el paradigma del Internet de las Cosas para resolver los problemas que se plantean ante un escenario de tal complejidad y tener los recursos apropiados que permitan llevar a cabo diferentes propuestas. Es importante destacar que la red de sensores ubicada en el entorno de inteligencia ambiental que se pretende monitorizar, debe estar desplegada de manera transparente para el habitante [16].

Dentro de un contexto de inteligencia ambiental, en entornos inteligentes bajo el paradigma IoT, podemos encontrar un sinfín de áreas de interés y que suscitan un gran entusiasmo entre la comunidad de investigadores e investigadoras a nivel mundial. Entre ellas, podemos citar: seguridad [17], tele-asistencia [18], tele-medicina [18], turismo [19], agricultura [20], tráfico [21] y un largo etcétera.

En las siguientes tres subsecciones se describen los tres pilares sobre los que se fundamenta la Inteligencia Ambiental bajo el paradigma IoT: dispositivos inteligentes, protocolos y estándares de comunicación y, finalmente, plataformas IoT.

2.3.1. Dispositivos inteligentes

Por norma general, cuando hablamos de dispositivos inteligentes se hace mención a un dispositivo electrónico, a menudo conectado con otros dispositivos me-

diante algún tipo de red o protocolo de comunicación y que puede funcionar de manera interactiva y autónoma hasta cierto punto.

Dentro de una definición tan generalista también cabe destacar que el término está directamente relacionado con la computación ubicua, es decir, el mismo dispositivo puede exhibir propiedades de éste paradigma, incluyendo la inteligencia artificial.

En el contexto del presente trabajo, se van a clasificar los distintos tipos de dispositivos Inteligentes en base a la función que realizan dentro de un entorno inteligente. Así pues, hablaremos de sensores si la función del dispositivo es la recogida de datos y la medición de magnitudes; de actuadores si por el contrario el dispositivo lo que realiza es el envío de datos con el fin de realizar cambios o actuaciones sobre el entorno o, también, nos podemos encontrar con dispositivos híbridos ya que tienen las capacidades para realizar las veces tanto de sensor como de actuador.

A continuación, se van a mostrar algunos ejemplos de dispositivos inteligentes de acuerdo al criterio indicado anteriormente:

- **Sensores:** sensor de temperatura, humedad, presión atmosférica, etc.
- **Actuadores:** altavoces inteligentes, luces LED conectadas, cerradura de una puerta, persianas, alarma, etc.
- **Híbridos:** báscula inteligente, televisor inteligente, pulseras de actividad, etc.

Dentro de una industria tan grande y en continuo auge, no dejan de aparecer nuevos fabricantes, cada uno con su filosofía propia por lo que cada vez son más los dispositivos que, debido a que quien los fabrica ofrece otro tipo de soluciones compatibles, es imposible acceder a sus datos, o manipularlos sin pasar por la plataforma o el protocolo de comunicación que su fabricante haya diseñado para ellos. Así mismo, existe otra gran vertiente, la cual dota a sus dispositivos de las

tecnologías de comunicación más estandarizadas y actuales, por lo que la interacción con estos dispositivos y la integración de los mismos en otra plataforma se facilita de manera notable.

Por ello, es clave en el Smart Lab del CEATIC desplegar una plataforma que sea capaz de estandarizar el acceso a la información de gran parte de los dispositivos que se albergan en él, además de ofrecer otras funcionalidades y características.

2.3.2. Protocolos y estándares de comunicación

Tras el auge del Internet de las Cosas, muchos fabricantes que trabajan en este ámbito, como Samsung, LG, Bosch o Google, han decidido utilizar unos estándares de comunicación para que las comunicaciones entre dispositivos de distintas marcas sea posible [16].

A continuación, se indican los protocolos de comunicación más importantes:

- **ZigBee:** Es un protocolo reciente, fijándonos en la fecha en la que se aprobó, vemos que no ha pasado mucho tiempo. Esto nos da una pista del auge que esta teniendo esta tecnología. En cuanto a su comunicación, es inalámbrica, haciendo uso del estándar IEEE 802.15.4 (Radio-fusión de bajo consumo) [22].

Figura 2.1: Logo ZigBee

- **Thread:** Este estándar pertenece a la compañía Google, tras realizar en 2014 la compra de la empresa Nest a la que pertenecía. El objetivo principal de este estándar es conectar un gran número de dispositivos independientemente del la comunicación que venga establecida por sus proveedores (WiFi, Bluethooth o ZigBee). Para ello, Thread hace uso de 6LoWPAN [23] bajo el protocolo IEEE 802.15.4.

Figura 2.2: Logo Thread

- **Z-Wave:** Esta comunicación utiliza los mismos principios que la conocida ZigBee, siendo la principal ventaja que diferencia esta tecnología de ZigBee el rango de frecuencia con el que trabaja [24]. Haciendo uso de la banda 900Mhz frente a la 2.4 GHhz. Anteriormente, se ha tratado esta diferencia como una ventaja, ya que a más baja frecuencia obtenemos una mayor facilidad para que las ondas atraviesen paredes u otros obstáculos comunes de los ambientes inteligentes.

Figura 2.3: Logo Z-Wave

- **Bluetooth LE:** (Low Energy) Es un nuevo sistema, versión 4.0, de la conocida comunicación Bluetooth. Emite en la banda 2.4 GHz con un alcance teórico de 100 metros. La gran particularidad de esta tecnología es la poca energía que necesita para su funcionamiento [25].

Figura 2.4: Logo Bluetooth LE

En la Tabla 2.1, se comparan los protocolos revisados de acuerdo a diferentes criterios.

Indicador	Wi-Fi	Bluetooth	ZigBee	Z-Wave
Bandas de frecuencia	2.4Ghz	2.4Ghz	2.4Ghz, 868/915 MHz	868Mhz
Tamaño de pila	1Mb	1Mb	20Kb	20Kb
Tasa de transferencia	11Mbps	1Mbps	250Kbps (2.4Ghz), 40Kbps (915Mhz), 20Kbps (868Mhz)	9.6/ 40 Kbit/s
Número de canales	11-14	79	16 (2.4Ghz), 10 (915Mhz), 1 (868Mhz)	1
Tipos de Datos	Digital	Digital, Audio	Digital (Texto)	Digital (Texto)
Rango de Nodos Internos	100m	10m-100m	10m-100m	10m-100m
Número de Dispositivos	32	8	255/65535	232
Requisitos de Alimentación	Media- Horas de batería	Media- Días de batería	Muy baja- Años de batería	Muy baja- Años de batería
Introducción al Mercado	Alta	Media	Baja	Media

Continua en la siguiente página...

Tabla 2.1 – continua de la anterior página.

Indicador	Wi-Fi	Bluetooth	ZigBee	Z-Wave
Arquitecturas	Estrella	Estrella	Estrella, Árbol, Punto a Punto y Malla	Malla
Mejores Aplicaciones	Edificio con Internet	Ordenadores y Dispositivos Móviles	Control de Bajo Costo y Monitoreo	Domótica y seguridad
Consumo de potencia	400ma transmitiendo, 20ma en reposo	40ma transmitiendo, 0.2ma en reposo	30ma transmitiendo, 3ma en reposo	23ma transmitiendo, 3ma en reposo
Precio	Alto	Bajo	Bajo	Bajo

Tabla 2.1: Comparativa de protocolos de comunicación

2.3.3. Plataforma IoT

Dado que el objetivo principal de este trabajo es analizar las plataformas de IoT actuales y estudiar las necesidades de recolección y procesamiento de datos del Smart Lab del CEATIC, en esta sección se revisa el término plataforma IoT (también conocido como middleware) desde su orígenes. En el siguiente capítulo, se realizará una análisis en profundidad de las diferentes plataformas IoT en el contexto de este trabajo fin de grado.

El término de Middleware ha ido evolucionando desde su creación. Así, dicho

término fue utilizado por primera vez con los sistemas de computación distribuida con el fin de facilitar la interacción entre una gran colección de ordenadores que coordinan sus acciones mediante el paso de mensajes en una red [26]. Posteriormente, este término evolucionó, haciendo referencia a un software que su objetivo es el de facilitar la interacción entre un cliente y un servidor situándose en el centro de la arquitectura [27].

A día de hoy y a grandes rasgos, un Middleware como plataforma IoT, es el encargado de *conectar* la parte Front-End de una aplicación con la parte Back-End y su propósito es el de proporcionar las acciones de alto nivel dependientes de una capa a bajo nivel haciendo transparente este proceso.

Figura 2.5: Middleware

Una plataforma IoT en un entorno de Inteligencia Ambiental viene a cubrir la necesidad de conectar un gran número de actores entre los que se encuentran principalmente, los dispositivos inteligentes. Además, proporciona una capa de conectividad para los dispositivos y sus capas de aplicación, que a su vez proporcionan servicios que aseguran una comunicación efectiva entre los programas [28]. Por tanto, una plataforma IoT está destinada a llevar a cabo las conexiones de hardware mediante un software personalizado. Además, no solo conecta el hardware, sino que puede integrar otras redes, aplicaciones y distinta información de entrada. En resumen, se genera una piedra angular que supervisa la actividad, aumenta la sinergia entre los diferentes elementos para potenciar y mejorar los resultados.

Existen grandes diferencias entre las múltiples plataformas IoT. Sin embargo, se puede establecer un conjunto de características comunes entre ellas:

- Reflejar en tiempo real el estado de los dispositivos que hayan conectados a

la red.

- Base de datos escalable dependiendo de las necesidades y que además, esté conectada a la nube.
- Acceso a la información de los dispositivos de forma uniforme.

Como funcionalidad adicional y dependiente de la plataforma IoT, es también deseable que se ofrezca una interfaz de usuario agradable y satisfactoria, que permita que información que antes no estaba disponible de una manera visual, ahora sí lo esté, así como que los datos se transmitan y se presenten a través de gráficos.

2.4. Smart Labs académicos

En esta sección se describe el Smart Lab del CEATIC, el cual se centra este trabajo fin de grado. Posteriormente, se revisan otros smart labs académicos con los que tiene relación el Smart Lab del CEATIC, con la finalidad de compartir información entre ellos.

2.4.1. UJAmI Smart Lab

El Smart Lab de Inteligencia Ambiental de la Universidad de Jaén (UJAmI), es uno de los primeros laboratorios centrados en la atención de personas mayores con algún tipo de dependencia. Este laboratorio se encuentra dentro del Centro de Estudios Avanzados en Tecnologías de la Información y la Comunicación (CEATIC). En la dependencia 109 del edificio C6 del campus de esta Universidad podemos encontrar una estancia de 25 metros cuadrados divididos en cuatro áreas.

Las áreas del UJAmI Smart Lab se componen de una entrada, un salón con un espacio de trabajo, un dormitorio con aseo integrado y una cocina. En la Figura 2.7 se ilustran las diferentes áreas del laboratorio.

Figura 2.6: Logo UJAmI Smart Lab

El laboratorio se presenta como una vivienda estándar debidamente amueblada y con todos los instrumentos de los que se pueden disponer en una casa habitable. Está dotado con más de 130 sensores de más de 30 tipos diferentes entre los que cabe destacar: sensores de contacto, de presencia, de luminiscencia, de temperatura, de humedad, de calidad del aire, de detección de incendios e inundaciones. También, cuenta con los dispositivos IoT más actuales y que están siendo la referencia en el panorama internacional como pueden ser: Amazon Echo con Alexa, dispositivos wearables, entre otros; podemos encontrar hardware con tecnología para la localización como son las balizas así como cámaras de visión.

En el capítulo 3, se hará un revisión en detalle de todos los dispositivos que alberga el Smart Lab con el fin de analizar las múltiples plataformas IoT y escoger la óptima para desplegar en el Smart Lab del CEATIC. Entre ellas, será analizada la plataforma básica de IoT de desarrollo propio en la Sección 3.2.2.2.

2.4.2. Smart Labs académicos

En todo el mundo son muchos los Smart Labs académicos que podemos encontrar, y cada vez se van sumando más y más iniciativas innovadoras y cada una con unas características especiales y objetivos diferentes.

Es importante reseñar que cuando un entorno, como un smart lab, ofrece un espacio de experimentación donde los usuarios y los productores pueden co-crear innovaciones, se contextualiza en un Living Labs.

En este contexto, nace la red europea de Living Labs, European Network of Living Labs (ENoLL) [29], fundada en 2006 y que crece cada día más. Como podemos observar en la Figura 2.8, el mayor número de estos espacios, se encuentra

Figura 2.7: Smart Lab del CEATIC

en Europa y América.

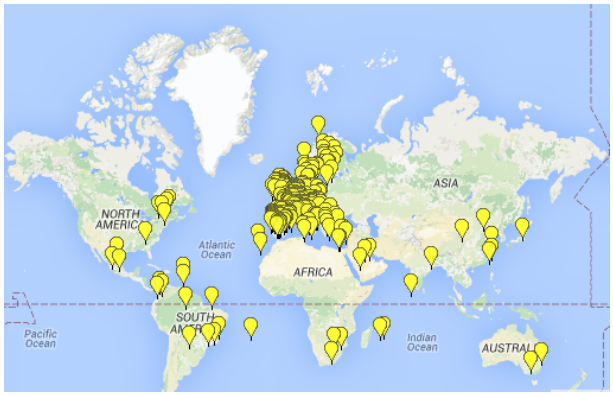

Figura 2.8: Mapa Smart Labs

Dicha red aglutina recursos conjuntos de interacción con el usuario, de prueba y de experimentación, orientadas a la innovación en muchos ámbitos diferentes, como energía, medios de comunicación, movilidad, salud, agroalimentación, etc.

Figura 2.9: Logo ENoLL

A continuación, se revisan los smart labs académicos con los que el Smart Lab del CEATIC interactúa con el objetivo de compartir información entre ellos en propuestas y aplicaciones de investigación y desarrollo.

- Smart Enviroment – Universidad de Ulster. Este laboratorio se encuentra situado dentro de la Facultad de Informática y Matemáticas de la Universidad de Ulster, en Belfast, Irlanda del Norte. Está compuesto de una serie de espacios de trabajo dedicados al estudio de los entornos inteligentes donde se trabajan las diferentes áreas de interés. El grupo de investigación SERG (Smart Enviroment Research Group) [30] tiene la visión de conseguir a través

de una investigación integrada y multidisciplinar el avance para apoyar y supervisar a las personas dentro de su casa y más allá. Este centro cuenta con 4 laboratorios inteligentes; cada uno tiene un tamaño de 17 metros cuadrados, y en la actualidad este grupo de investigación está enfocado en cuatro grandes áreas donde dirige su actividad investigadora:

Figura 2.10: Smart Lab Universidad de Ulster

- Informática wearable.
- Informática autónoma.
- Procesamiento y visualización de signos vitales.
- El movimiento y análisis del comportamiento.
- Modelización matemática.

Entre algunas de las aplicaciones que se desarrollan en este laboratorio, podemos destacar por ejemplo, la detección de caídas pode medio de un sensor de visión térmica 2.11.

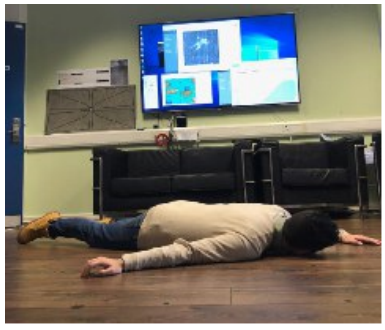

Figura 2.11: Detección de caídas con sensor de visión térmica

- **Halmstad Living Lab**.

  Este Smart Lab [31] está alojado por la Universidad de Halmstad [32], en Suecia. Su objetivo principal es el de mejorar los procesos de innovación para las empresas y proporcionar soluciones TI con gran valor añadido para el consumidor. Sus focos de trabajado actuales se centran en el campo de la tecnología para la salud.

  Figura 2.12: Logo Halmstad Living Lab

  Pertenecen a la ENoLL desde 2008 y llevan a cabo numerosos proyectos, algunos de gran impacto a nivel Europeo.

  Los modelos, métodos y herramientas que utilizan cubren la participación del usuario en cada paso del proceso de innovación y, por lo tanto, se refieren tanto a la creación como a la validación de nuevas innovaciones.

- **Botnia Living Lab** Ubicado en la Universidad Tecnológica de Luleå [33] en Suecia, este laboratorio [34] desarrolla una estrecha cooperación con usuarios finales y partes interesadas, así como con investigadores. Cuentan con 6000 usuarios finales que se encuentran en toda Suecia y participan de diversas maneras en el proceso total desde la búsqueda de las necesidades y la generación de ideas, pasando por el desarrollo de conceptos y las pruebas de prototipo/usabilidad, hasta el servicio piloto.

  Figura 2.13: Logo Botnia Living Lab

  Tienen desarrollada una metodología Living Lab propia llamada FormIT,

adoptado por muchos Living Labs en toda Europa y explotado por la industria. Esta metodología respalda la participación del usuario cuando desarrolla innovaciones digitales. También es una metodología para el empoderamiento del usuario, para capitalizar el poder de la multitud. Esta es una metodología iterativa e interactiva de procesos de innovación con participación del usuario en todas las fases del proceso de innovación, desde la búsqueda de las necesidades hasta el lanzamiento de prueba y premercado.

2.5. Aplicaciones en el Smart Lab del CEATIC

En esta sección, se revisan las tres principales propuestas de investigación y desarrollo del Smart Lab del CEATIC: el posicionamiento indoor, el reconocimiento de actividades y la ayuda en la toma de decisiones. Aunque sean tres áreas de aplicación diferentes, y cada una con sus particularidades, realmente las tres guardan una estrecha relación y se detallan en el orden en el que están relacionadas.

El posicionamiento es un factor importante a la hora de determinar la actividad que se está realizando, y a su vez, para ayudar en la toma de una decisión es de importancia conocer la actividad que se está llevando a cabo.

2.5.1. Posicionamiento Indoor

Las ubicaciones interiores brindan múltiples beneficios en el desarrollo de sistemas de atención domiciliaria. Dichas soluciones pueden localizar la ubicación de las personas en el interior y, por ejemplo, identificar una emergencia médica para solicitar ayuda inmediata [18]. Además, las ubicaciones interiores también se pueden considerar como un complemento para el reconocimiento de actividades que posteriormente se describirán.

A continuación, se realizará una pequeña reseña de algunas de las técnicas utilizadas para realizar posicionamiento indoor en entornos de inteligencia ambiental.

- **Fingerprinting:** la base de esta técnica es construir un mapa compuesto por un conjunto de puntos y en cada posición de ese mapa asocia una lista de los dispositivos emisores de señal y su intensidad, de manera que podemos posicionar a un sujeto calculando el valor que se obtiene del dispositivo asociado a la persona comparándolo contra el mapa de referencia. Aunque para realizar esta técnica la señal más utilizada es la Wi-Fi, se puede utilizar con otras señales como de campo magnético o GPS, incluso surgen nuevas tecnologías de radiofrecuencia muy prometedoras y con las que se podría resolver el problema como el UWB (Ultra Wide Band).

- **Balizas:** el funcionamiento de esta técnica está basado en la colocación de pequeñas balizas distribuidas por el entorno inteligente, estas son emisoras de una señal, generalmente Bluetooth LE, y mediante el dispositivo asociado a la persona que queremos posicionar se realiza el cálculo con la intensidad recibida de cada una de las balizas detectadas de manera que podemos detectar donde se encuentra el sujeto pero con un error de precisión grande por lo que, a lo sumo, servirá para dirimir si se encuentra en una estancia u otra del entorno.

- **Sensores de visión:** el funcionamiento de esta técnica se basa en algoritmos para la detección de marcadores visuales situados en el entorno que son identificados a través de un sensor de visión que porta el habitante. La ubicación del ocupante se determina utilizando técnicas de visión artificial que identificaron las marcas de referencia ubicadas en el entorno que luego se cruzaron con una base de conocimiento que conoce la ubicación conocida de los objetos.

2.5.2. Reconocimiento de Actividades

El objetivo del reconocimiento de actividad es identificar las actividades a medida que ocurren, en función de los datos recopilados de los objetos sensibilizados en un entorno [35]. Este es un problema desafiante y ha recibido una atención significativa en la literatura recientemente [36].

Partiendo de una colección de datos de los valores de los sensores con los que se interactúe en el ambiente inteligente, se hace uso de técnicas de machine learning [37] para predecir la actividad que se está realizando con la recolección de los datos de los sensores con los que se interactúe en el momento de realización de una actividad. Dependiente de la calidad de los datos con los que se entrene, pero si se realiza con cautela, puede arrojar resultados extremadamente buenos.

En caso de que no conste colecciones de datos sobre el ambiente, se puede hacer uso de aplicaciones basados en conocimiento experto para detectar la actividad.

2.5.3. Ayuda a la Toma de Decisión

Uno de los principales objetivos de la inteligencia ambiental, y concretamente de los entornos inteligentes bajo el paradigma IoT, es facilitar el día a día de las personas y ayudar a mejorar su calidad de vida. Por supuesto, también de personas que padezcan algún tipo de enfermedad.

De entre las enfermedades más comunes se encuentran las relacionadas con procesos cognitivos como la demencia, estas enfermedades actualmente son incurables. Por tanto, las necesidades que surgen se basan en retrasar su progreso y facilitar los procesos vitales mientras se desarrolla la enfermedad.

Con la ayuda en la toma de decisiones se puede determinar, y aquí entran en juego los dos puntos anteriores sabiendo que actividad está realizando, los pasos a seguir para completar esa actividad. De manera que si trazamos la realización de la actividad por parte de un usuario, podemos detectar anomalías en la traza

y ayudar a este usuario, interactuando con él, por medio de los actuadores del entorno inteligente a que realice la actividad en concreto de la manera adecuada.

Capítulo 3

Análisis y elección de la plataforma IoT

3.1. Introducción

Desplegar una plataforma IoT en el Smart Lab de inteligencia ambiental del CEATIC es el propósito principal de este trabajo fin de grado. El criterio fundamental para la elección de dicha plataforma IoT es que sea compatible con el mayor número de dispositivos inteligentes que alberga actualmente el Smart Lab y las ventajas que ofrece. En el mercado existen multitud de opciones, es por ello que es necesario realizar, por un lado, un análisis en profundidad de las diferentes plataformas IoT que se encuentran actualmente en el mercado con las ventajas e inconvenientes que brindan y, por otro lado, un análisis de los dispositivos inteligentes y de los protocolos que se encuentran presentes en el Smart Lab del CEATIC. Ambos análisis permitirán la selección la plataforma IoT a desplegar en el Smart Lab del CEATIC.

En este capítulo, en primer lugar, se realizará un análisis de las plataformas IoT actuales considerando un punto de vista genérico y un punto de vista orientado a investigación. Posteriormente, se analizarán el conjunto de dispositivos

inteligentes que alberga el Smart Lab del CEATIC, considerando sus protocolos de comunicación y su integración en las plataformas IoT analizados. Finalmente, en base a dichos análisis, se escogerá la plataforma a desplegar en Smart Lab del CEATIC.

3.2. Análisis de Plataformas IoT

En esta sección, se realiza un análisis de las plataformas IoT más populares existentes. Dicho análisis se agrupa en dos categorías, por un lado tenemos las plataformas IoT genéricas, cuyas funcionalidades, la mayoría van encaminadas a un bloque compacto de herramientas que nos permitan realizar las tareas que demandamos según nuestro tipo de problema. La otra categoría de este análisis son las plataformas IoT orientadas a la investigación, cabe destacar que, lo que proponen este tipo de soluciones son dotar de funcionalidades especiales a los entornos de investigación en Ambientes Inteligentes y que faciliten dicha tarea. Por tanto, es de vital importancia tener en cuenta los puntos en común que puedan tener las propuestas de una categoría con las de la otra puesto que teniendo una serie de requisitos específicos que cumplir, la solución ideal será aquella que sea capaz de interactuar de una manera más correcta con diversas plataformas.

3.2.1. Plataformas IoT genéricas

A continuación, se presenta un análisis de las plataformas IoT que podemos encontrar en el mercado, tanto software libre como privativo, centrándose en analizar las virtudes de cada una de ellas con el fin de poder arrojar luz en una toma de decisión sobre la implantación de alguna de estas soluciones.

3.2.1.1. Google Home

Una de las empresas más importantes del sector como es Google, presenta su solución orientada al IoT y a los Ambientes Inteligentes en formato aplicación móvil. Teniendo en cuenta este formato, ya podemos vislumbrar el enfoque que tiene, y que será una solución que aporte una serie de funcionalidades específicas con gran solvencia, pero que en un entorno inteligente complejo, puede no tener todas las funcionalidades que se le puedan exigir.

Figura 3.1: Logo Google Home

Esta aplicación sólo es compatible con los dispositivos de Google, por lo que ya perdemos la mayor parte de funcionalidades que podemos desear que nos ofrezca una solución de estas características implantada en un Ambiente Inteligente.

Por otra parte, la arquitectura de la aplicación basa su funcionamiento en tener integrado el altavoz de la misma compañía y que recibe su mismo nombre: Google Home. Sin este dispositivo integrado, el uso de la aplicación se limita a poder castear contenido multimedia a otros dispositivos de Google que podamos tener, por ejemplo, Google ChromeCast o Google ChromeCast Audio.

A continuación, se procede a describir las ventajas e inconvenientes de la solución:

- **Ventajas:** al ser una solución enfocada a los propios dispositivos de la compañía, hace que su funcionamiento sea muy bueno, con pocos errores y un gran desarrollo.

- **Inconvenientes:** el gran inconveniente de esta aplicación es el enfoque que

utiliza, ya que limita al máximo sus funcionalidades y uso, más si cabe, en un Ambiente Inteligente complejo o con gran variedad de dispositivos. Además, su diseño privativo hace de Google Home una aplicación muy cerrada y poco accesible.

3.2.1.2. Samsung SmartThings

Otra de las grandes empresas tecnológicas de relevancia mundial que también aporta una plataforma IoT. Desde que en 2014 Samsung comprara la empresa SmartThings [38], la compañía ha reforzado al máximo su división de Internet of Things y es por eso que empezaron a ofrecer diferentes productos orientados a este mercado, entre ellos, el que aquí se presenta.

Figura 3.2: Logo Samsung SmartThings

La solución propuesta por Samsung tiene dos partes importantes, una es la plataforma de gestión que es vía Web y otra es la plataforma de interacción cuyo formato es aplicación móvil.

Dicha plataforma es compatible con una gran cantidad de dispositivos y protocolos de comunicación pero es dependiente de que el Ambiente Inteligente posea un hardware controlador específico, denominado HUB SmartThings, que también está desarrollado por Samsung y sin el cual la plataforma no podría obtener los datos recogidos mediante los sensores ni comunicarse con otros posibles dispositivos integrados.

Desde la plataforma de gestión Web podemos desarrollar los componentes para la automatización de nuestro Ambiente Inteligente y dotar de funcionalidad al mismo, mientras que, desde la aplicación móvil podemos visualizar el entorno y los dispositivos que en él haya integrados.

Las principales ventajas e inconvenientes de esta plataforma IoT son las siguientes:

- **Ventajas:** tiene una gran compatibilidad con muchos dispositivos y además es compatible con prácticamente todos los protocolos de comunicación actuales y más usados.
- **Inconvenientes:** es dependiente de un HUB, que es el que se encarga de incrustar los dispositivos y que hace transparente todo el proceso, haciendo que el acceso a los datos sea muy reducido. Por tanto, se trata de una solución bastante cerrada tanto en software como en hardware.

3.2.1.3. Domoticz

Domoticz [39] surge de la necesidad de tener una plataforma IoT Open Source, además, antes tampoco se habían tenido en cuenta dispositivos con pocos recursos computacionales, por lo que Domoticz plantea resolver ambos problemas.

Figura 3.3: Logo Domoticz

Esta plataforma IoT está diseñada específicamente para dispositivos empotrados, tales como: Raspberry Pi, Orange Pi, Banana Pi. Es muy atractiva la idea de poder tener una plataforma de gestión de un ambiente Inteligente en un dispositivo propiamente IoT. Por lo que cuando Domoticz ve la luz obtiene mucho éxito. Su diseño para dispositivos con pocos recursos hace que tenga mucha facilidad de implantación.

Aquí vemos las ventajas e inconvenientes:

- **Ventajas:** está diseñado específicamente para dispositivos empotrados, por lo que su coste computacional es muy bajo y su facilidad de despliegue es alta.

- **Inconvenientes:** debido a su diseño, posee una interfaz muy poco amigable. Además, no es modular, por lo que presenta problemas ante dispositivos con características especiales o diferentes; por ejemplo, un sensor que por su diseño realice el envío de los datos recogidos mediante dos protocolos de comunicación diferentes.

3.2.1.4. Home Assistant

Tras surgir Domoticz, se abre un abanico importante de soluciones Open Source. Una de ellas es Home Assistant cuya aparición resolvió varios de los problemas que se planteaban con Domoticz.

Figura 3.4: Logo Home Assistant

Home Assistant [40] es una solución Open Source, que se ejecuta en Python 3 y es muy modular. Su funcionamiento se basa en una serie de módulos que podemos activar o desactivar según las funcionalidades de las que queramos dotar a la plataforma IoT.

Además de ofrecer una plataforma de gestión vía aplicación Web, también posee una aplicación móvil con la que podemos realizar cambios en el entorno además de ver los datos que recojan los sensores.

Entre sus ventajas e inconvenientes más importantes se encuentran:

- **Ventajas:** funcionamiento muy modular, permite aislar características además de poder configurarlas por separado. La aplicación móvil es muy versátil y funcional, ya que se pueden realizar casi las mismas tareas desde ella que desde la plataforma Web.

- **Inconvenientes:** al ser una solución Open Source, su desarrollo y mantenimiento dependen de la comunidad de usuarios que tenga y, si esta es reducida, existe un problema: hay muchos dispositivos que no cuentan con módulos desarrollados por lo que no son compatibles.

3.2.1.5. OpenHAB

Por último, entraremos a realizar el análisis de una de las soluciones más usadas en todo el panorama, OpenHAB (Open Home Automation Bus) [41].

Figura 3.5: Logo OpenHAB

Principalmente orientado a la automatización en el hogar, OpenHAB es un software para integrar diferentes sistemas y tecnologías de automatización en una sola solución. Ofrece interfaces de usuario uniformes a la vez que un motor de gestión que permite dominar las reglas de automatización en el hogar.

Está desarrollado en Java, por lo tanto, es compatible con cualquier plataforma capaz de ejecutar una Java Virtual Machine, por lo que es compatible con casi cualquier dispositivo que queramos utilizar de host para el despliegue.

Al igual que Domoticz y Home Assistant, OpenHAB es una solución Open Source. No obstante, la gran comunidad que tiene openHAB detrás, recibiendo

actualizaciones estables mensualmente y corrigiendo errores semanalmente, hacen que esta solución goce de una gran aceptación.

Su funcionamiento también es modular, y al tener tanto apoyo de la comunidad hace que sean las propias marcas de dispositivos las que se preocupen de que su dispositivo sea compatible aún teniendo que desarrollar ellos el módulo. A nivel de protocolos de comunicación también es compatible con prácticamente todos los estandarizados en la actualidad.

Además, posee una gran funcionalidad adicional que hacen de OpenHAB una solución única, como la transmisión de datos a aplicaciones de mensajería instantánea o redes sociales.

Las principales ventajas e inconvenientes de OpenHAB son los que a continuación se detallan:

- **Ventajas:** su principal ventaja es la posibilidad de despliegue en casi cualquier dispositivo, que sea una solución Open Source y, con una gran comunidad de soporte y mantenimiento detrás, funcionamiento modular de tal manera que prácticamente cada módulo es una aplicación independiente, es compatible con todos los estándares de comunicación y posee grandes funcionalidades adicionales que no podemos encontrar en otras plataformas IoT.

- **Inconvenientes:** el principal inconveniente que podemos encontrar es que al ser una solución tan completa, si está desplegada en un ambiente inteligente complejo se puede hacer algo difícil su mantenimiento, además de la introducción de cambios.

3.2.2. Plataformas IoT orientados a investigación

Dentro de la gran oferta de plataformas para el mundo IoT podemos encontrar, como hemos visto en la sección anterior, soluciones que cumplen casi con todos los

requisitos funcionales necesarios para la gestión de un Ambiente Inteligente, desde algo muy simple y que podamos manejar con una aplicación móvil hasta entornos más complejos para los que sean necesarios un gran compendio de módulos que nos den soporte ante el abanico de dispositivos y funcionalidades que necesitemos implementar.

En un contexto investigativo y en el contexto del Smart Lab del CEATIC, surgen necesidades especiales que cubrir y que no se le pueden pedir a soluciones comerciales o puramente orientadas a la gestión de ambientes. Es por esto que surgen plataformas desarrolladas en entornos de investigación y que proponen satisfacer las necesidades especiales que surgen a los investigadores facilitando herramientas que, en combinación con otras plataformas IoT, eleven las tareas de investigación en Ambientes Inteligentes a un nivel superior.

A continuación, se presenta un análisis detallado de dos de éstas alternativas. La primera de ella es "Sensor Central" desarrollada en el Smart Lab de la Universidad del Ulster, el cual fue revisado en la Sección 2.4.2. La segunda de ella es "Web System for Monitoring Smart Environments" desarrollada por la Universidad de Jaén en la creación del Smart Lab del CEATIC.

3.2.2.1. Sensor Central

La plataforma Sensor Central [42], desarrollada por el grupo de investigación de Ambientes Inteligentes de la Universidad de Ulster en Belfast, Irlanda del Norte [30], nace a raíz de la inexistencia de una plataforma que facilite el almacenamiento y recuperación de datos de sensores para permitir resultados tales como el análisis predictivo y el reconocimiento de actividades.

El enfoque que tiene Sensor Central, desde su arquitectura hasta la forma en la que ofrece los datos, es el de una plataforma Big Data [43]. En la actualidad hay muchas soluciones de este tipo, sin embargo, todas presentan una serie de deficiencias reseñables, como son la falta de compatibilidad e interoperabilidad

genérica con sensores y la ausencia de características que respalden la investigación en el ámbito académico.

Este es el problema que pretende resolver Sensor Central, centrándose en aportar un mecanismo de acceso rápido a datos, con gran compatibilidad ante el gran número de dispositivos que podemos encontrar en el mercado y con un alto rendimiento. La plataforma hasta el momento almacena más de 850 millones de registros y una de las líneas que pretenden seguir en trabajos futuros es la inclusión de la misma en la Open Data Iniciative, lo que permitirá la colaboración con la comunidad internacional de investigadores.

Dentro de los grandes problemas que pretende resolver el Big Data, cabe reseñar las claves en las que se centra para empezar a entender las funcionalidades que debe tener una plataforma Big Data. Estos tres ítem claveS están denominados las tres V y son: Variedad, Volumen y Velocidad.

- **Variedad:** la heterogeneidad de los datos que se almacenan y su constante cambio.
- **Volumen:** los datos almacenados serán de un gran volumen y crecerán con gran rapidez.
- **Velocidad:** los registros de estos datos se realizarán con alta frecuencia.

Como hemos visto, Sensor Central está enfocado al Big Data y a las grandes colecciones de datos pero con una orientación al IoT. Es por ello que ofrece una gran compatibilidad con sensores, tales como: acelerómetro, calidad del aire, Beacons, contacto, GPS, humedad, luminiscencia, magnetómetro, etiquetas NFC, suelo inteligente, temperatura y visión térmica, entre otros.

Las principales características a nivel de investigación que ofrece Sensor Central son las siguientes:

- La compatibilidad y posibilidad de integración de sensores.

- Gestión experimental: definición de experimentos con su información relacionada e investigadores, capacidad de instanciación y almacenamiento de los mismos, interfaz de anotación, posibilidad de importar y exportar los experimentos.
- Funcionalidades de aprendizaje automático.
- Puesta en común de los datos de los sensores con otros procesos independientes.
- Plataforma flexible y modular.

La arquitectura de Sensor Central se establece en tres capas de operación. Una primera capa donde se encuentra el núcleo de la aplicación y que contiene toda la lógica de la misma, además de los sistemas de almacenamiento y las conexiones con otros sistemas externos para, entre otras tareas, realizar las exportaciones de los datos. Existe una capa intermedia que es la encargada de dar soporte a todos los protocolos de comunicación con los que Sensor Central es compatible, así pues en esta capa encontraremos los End Points de los servicios REST, MQTT, WebServices y Firebase. Por último, encontramos una tercera capa que gestiona las conexiones con los sensores y sus lecturas, así como las aplicaciones de consumo.

El funcionamiento y el intercambio de datos entre los sensores y la plataforma se basa en "listeners" que están configurados con cada dispositivo y que, mediante servicios REST y MQTT, publican en el servidor los datos del sensor en formato JSON.

Sensor Central ofrece una interfaz muy completa al investigador donde puede gestionar sus experimentos, tener en cuenta infinidad de variables y aplicar técnicas de aprendizaje automático con el fin de mejorar la tarea investigativa. Además, con esta interfaz se pueden exportar los experimentos para almacenarlos o poderlos tratar con otras plataformas.

3.2.2.2. Sistema Web para la Monitorización de Ambientes Inteligentes

Esta plataforma [7] IoT fue desarrollada por el grupo de investigación de Sistemas Inteligentes basados en el Análisis de Decisión Difuso de la Universidad de Jaén [44]. Esta plataforma surge a raíz de realizar un exhaustivo estudio del arte y llegar a la conclusión de que, aunque existían plataformas de monitorización de ambientes inteligentes, ninguna de ellas era a través de un sistema web, por tanto, eran dependientes de un software de escritorio instalado en un ordenador y no permite gestionar ambientes, así como ninguna estaba orientada al reconocimiento de actividades.

Las principales características de esta plataforma son las siguiente:

- Sistema Web, accesible desde cualquier dispositivo y lo suficientemente flexible para su adaptación a entornos cambiantes.
- Interfaces, tanto de administración como de gestión de entornos inteligentes.
- Enfocado al reconocimiento de actividades.
- Compatibilidad con sensores y actuadores.
- Posibilidad de localización de sensores.

La forma en la que esta plataforma IoT gestiona los datos de los sensores y dispositivos inteligentes en general, es mediante un esquema objeto-propiedad-valor. Con dicho esquema y utilizando como base de comunicación los servicios REST con notación en JSON es posible describir cualquier sensor o dispositivo inteligente.

En cuanto a la arquitectura que sigue esta plataforma se trata de un modelo cliente-servidor [45]. En la parte del servidor se desarrollan los servicios básicos que proporcionan una capa de abstracción mediante la cual los clientes pueden

comunicarse con el sistema. Es el servidor el encargado de comunicarse con los sensores y esta tarea, la realiza mediante servicios REST, con los cuales la plataforma es capaz de acceder a los datos de los sensores, así como de persistirlos para poder mostrarlos y ofrecerlos a los clientes.

Según el enfoque que tiene esta plataforma, cada sensor tiene asociado un objeto que es el que representa sus valores. Cada objeto representa un sensor, de manera que la inclusión del sensor a la plataforma no sirve de nada si no tiene un objeto que represente sus valores.

Al estar desarrollado en base a los servicios REST, cada objeto incluido en la plataforma tiene una URL única que lo identifica y mediante la cual se pueden acceder a los datos, así como almacenar nuevos.

La parte cliente, es accesible mediante cualquier navegador Web y está desarrollada combinando HTML5, CSS3, Bootstrap y AngularJS.

En cuanto a la funcionalidad, es necesario diferenciar los dos perfiles con los que se puede acceder a la aplicación, como hemos comentado antes. Uno es el perfil de Administrador y otro es el perfil público, las funcionalidades que ofrece cada uno serán repasadas a continuación.

Entre las funcionalidades del perfil Administrador, podemos destacar las siguientes:

- **Gestión de Ambientes:** permite al usuario crear, editar o eliminar ambientes, así como permite también la edición de sus propiedades, nombre y descripción.
- **Gestión de Sensores:** permite añadir y eliminar sensores.
- **Gestión de Objetos:** permite crear, eliminar o editar objetos asociados a los sensores.
- **Gestión de Localizaciones de objetos:** como una de las características de esta plataforma es la localización de sensores, en este apartado se puede

gestionar la localización de los objetos asociados a los sensores.

A continuación vamos a repasar las funcionalidades que se podrán desarrollar con un perfil público. Este perfil está mas encaminado a la visualización de datos, como podemos observar:

- **Visualización de estados actuales:** en tiempo real, la plataforma muestra el estado de cada sensor.
- **Visualización de estados pasados:** podemos consultar un histórico de los estados de los sensores.
- **Consulta de eventos pasados:** también, a modo de histórico, podemos obtener información de eventos, instanciados mediante un timestamp.

A nivel de investigación, la plataforma destaca en las siguientes cualidades, principalmente orientadas al reconocimiento de actividades:

- **Gestión de actividades**. Se puede gestionar tipos de actividades a realizar en la plataforma IoT. A través de una aplicación móvil, se puede indicar el inicio y el fin de cada una de esas actividades.
- **Segmentación de actividades**. La plataforma permite la segmentación de cambios de estado del ambiente a partir del etiquetado del inicio y el fin de actividades.
- **Exportar datos**. La plataforma permite 3 tipos de exportación de datos del ambiente a partir de un periodo de tiempo. El primer tipo es binario, indicando si un sensor cambia o no de estado en un periodo de tiempo. El segundo tipo es numérico, indicando el número de veces que cambió de estado y, finalmente, flujo de eventos, indicando para cada cambio de estado (evento) el tipo y la fecha en la que se produjo.

3.3. Análisis de dispositivos del Smart Lab

Como fue indicado en la Sección 2.3.1, los dispositivos inteligentes son dispositivos electrónicos con la capacidad de comunicarse con otros elementos. Este trabajo fin de grado se centra en desplegar una plataforma IoT para el Smart Lab del CEATIC que unifique la comunicación entre tales dispositivos, entre otros objetivos. Por dicho motivo, es esencial revisar cada uno de los dispositivos inteligentes del laboratorio para poder seleccionar posteriormente la plataforma IoT compatible con el mayor número de dispositivos.

A continuación, se analizan los diferentes dispositivos inteligentes del CEATIC de acuerdo a su naturaleza, considerando los protocolos de comunicación. Además, se tiene en cuenta si los dispositivos son o necesitan obligatoriamente para funcionar de algún software **privativo**. Se conoce como privativos a aquellos dispositivos o software donde el usuario tiene limitadas las posibilidades de uso, modificación o redistribución y que a menudo tienen licencias con algún coste.

3.3.1. Dispositivos con sensores ambientales

En este apartado, se van a analizar los dispositivos inteligentes con sensores ambientes que existen en el Smart Lab del CEATIC.

- **Contacto.** Empleados para detectar el final del recorrido o la posición límite de algún componente, generalmente mecánico. El sensor consta de dos partes: una pequeña pieza móvil (NA - Normalmente Abierto) y una pieza fija (NC - Normalmente cerrado). Cuando ambas partes están separadas, el estado del sensor será 1 (Abierto) y por el contrario, cuando ambas piezas se juntan, el sensor devolverá el estado 0 (Cerrado). Su principal uso es saber si una puerta o ventana está abierta o cerrada. El Smart Lab del CEATIC posee dos tipos de sensores de contacto:

- **Everspring Door/Window Sensor.** Muy robusto y fiable, utiliza Z-Wave como protocolo de comunicación. (Figura 3.6).

Figura 3.6: Sensor de contacto Everspring

- **Fibaro Door/Window Sensor.** Tiene un tamaño más reducido, aunque algunas desventajas importantes como la duración de las baterías. También funciona mediante el protocolo de comunicación Z-Wave. (Figura 3.7).

Figura 3.7: Sensor de contacto Fibaro

- **Multipropósito.** Los sensores multipropósito contienen varios sensores y su utilización varía según la combinación de ellos que el usuario requiera.

  - **Samsung Multipropose Sensor.** El sensor multipropósito de Samsung ofrece datos de temperatura y luminosidad. Además también se puede utilizar como sensor de contacto. Utiliza Zigbee como protocolo de comunicación. (Figura 3.8).

Figura 3.8: Sensor multipropósito Samsung

- **Inundación.** Alertan al usuario con fuertes señales sonoras cuando detectan líquidos en su rango de acción.

  - **Fibaro Flood Sensor.** Este sensor ofrece datos sobre fuertes variaciones de temperatura, además de alertar sobre posibles inundaciones. Funciona bajo el protocolo Z-Wave. (Figura 3.9).

Figura 3.9: Sensor de inundación Fibaro

- **Presencia.** Estos sensores poseen dos estados: P (Presencia) y NP (No Presencia). Se dividen en dos partes; una parte fija que detecta cuando otra parte móvil está dentro del rango de acción y, en función de esto, cambia el estado del sensor.

  - **Samsung Arrival Sensor.** En combinación con el HUB Samsung, posee un rango de acción bastante amplio. Se comunica mediante ZigBee. (Figura 3.10).

Figura 3.10: Sensor de presencia Samsung

- **Movimiento.** Están basados en el principio PIR (Passive Infrared), funcionan detectando el calor corporal, poseen un estado "Alarma" que puede contener dos valores diferentes (Activo o No Activo). Una vez que el sensor está activo, crea una "rejilla de cuadrículas protectora" de manera que si un objeto en movimiento bloquea demasiadas zonas de la cuadrícula cambiando los niveles de energía infrarroja, el estado cambia de No Activo a Activo.

  - **Samsung Motion Sensor.** Además de movimiento, este sensor es capar de detectar la temperatura del ambiente. El protocolo de comunicación que utiliza es ZigBee. (Figura 3.11).

Figura 3.11: Sensor de movimiento Samsung

  - **Fibaro Motion Sensor.** No ofrece tan buenos resultados en la detección de movimiento aunque también posee la capacidad de detectar la luminosidad del ambiente. Se comunica mediante Z-Wave y ZigBee. (Figura 3.12).

Figura 3.12: Sensor de movimiento Fibaro

La Tabla 3.1 resume los dispositivos analizados, indicando su fabricante, protocolo de comunicación y si posee un software asociado privativo.

Dispositivo con sensores ambientales	Fabricante	Protocolo	Privativo
Contacto	Everspring	Z-Wave	No
Contacto	Fibaro	Z-Wave	No
Multipropósito	Samsung	ZigBee	Sí
Inundación	Fibaro	Z-Wave	No
Presencia	Samsung	ZigBee	Sí
Movimiento	Samsung	ZigBee	Sí
Movimiento	Fibaro	Z-Wave	No

Tabla 3.1: Comparativa de dispositivos con sensores ambientales

3.3.2. Dispositivos actuadores

En esta sección, se van a analizar los dispositivos actuadores en el Smart Lab del CEATIC.

- **Philips HUE.** El sistema de iluminación completo consta de diferentes dispositivos; por un lado tenemos las bombillas y por otro lado tenemos un Bridge que centraliza y facilita la gestión de las diferentes bombillas. Ofrece posibilidades como controlar el color y el brillo de las bombillas mediante

software, acceso remoto a las mismas, y actualizaciones remotas. El protocolo de comunicación que sigue este sistema de iluminación es ZigBee. (Figura 3.13).

Figura 3.13: Sistema de iluminación Philips HUE (Bridge + Bombillas)

- **Altavoces SONOS Play:1.** Además de ofrecer las posibilidades de un altavoz normal, al funcionar mediante Wi-Fi, podemos combinar varios altavoces y tratarlos como uno solo. Son resistentes a la humedad. (Figura 3.14).

Figura 3.14: Sistema de sonido SONOS Play:1

- **Harmony SmartHome.** Con el Hub Harmony, podemos centralizar la gestión de todos los dispositivos que funcionen por Wi-Fi, Bluetooth o IR y controlarlos desde la aplicación que proporciona para ello. Tiene capacidad de control de hasta 15 dispositivos. (Figura 3.15).

- **Termostato NEST.** Este termostato tiene la capacidad de memorizar la temperatura a la que se encuentra el ambiente, de manera que puede interactuar sobre ella según las configuraciones que se realicen. También, es

Figura 3.15: Hub Harmony SmartHome

capaz de aprender sobre la forma en la que se calienta el ambiente o si se crean corrientes de aire dentro de él, actuando en consecuencia sobre la temperatura. Su protocolo de comunicación es Wi-Fi. (Figura 3.16).

Figura 3.16: Termostato NEST

- **Detector de humo NEST.** El detector de humo NEST Protect cuenta con un sensor de espectro disperso para detectar el humo, comprueba su propio funcionamiento automáticamente. Además, ofrece amplias posibilidades de interacción mediante dispositivos móviles como silenciarlo o envío de alertas. Se comunica mediante Wi-Fi. (Figura 3.17).

Figura 3.17: Detector de humo NEST Protect

- **Sense Mother.** Este dispositivo es un monitor de información construido para ofrecer, sobre todo, flexibilidad. Realmente funciona como un compendio de sensores, denominados "cookies", con los cuales podemos obtener información sobre movimiento, temperatura y proximidad. Las "cookies" se comunican con el Sense Mother que monitoriza los datos y nos ofrece la posibilidad de programar reacciones. Funciona mediante el protocolo Z-Wave. (Figura 3.18).

Figura 3.18: Sense Mother + Cookies

- **Etiquetas NFC.** Una de las tecnologías en auge en los últimos años es el NFC. Uno de sus usos son las etiquetas inteligentes o SmartTags. Podemos encontrarlas tanto en forma de llaveros como en formato pegatina, permitendo realizar diversas acciones preconfiguradas. Generalmente se adjuntan a algún objeto y mediante ellas podemos etiquetar el uso de ese objeto. (Figura 3.19).

Figura 3.19: Etiquetas NFC en sus diferentes formatos

La Tabla 3.2 resume los dispositivos actuadores analizados, indicando su fabricante, protocolo de comunicación y si posee un software asociado privativo.

Dispositivo actuador	Fabricante	Protocolo	Privativo
Bombillas HUE	Philips	ZigBee	Sí
Altavoces Play:1	SONOS	Wi-Fi	Sí
Switch SmartHome	Harmony	Wi-Fi, Bluetooth, IR	Sí
Termostato	NEST	Wi-Fi	Sí
Detector de humo	NEST	Wi-Fi	Sí
Mother	Sense	Z-Wave	Sí
Etiquetas NFC	-	-	No

Tabla 3.2: Comparativa de dispositivo actuador

3.3.3. Dispositivos para posicionamiento indoor

En esta sección se van a analizar los dispositivos para posicionamiento de interiores en el Smart Lab del CEATIC.

- **Suelo inteligente SensFloor.** Es un gran panel de sensores capacitivos instalable bajo cualquier tipo de superficie. Cuando una persona camina por encima se activan señales en los sensores capacitivos que se envían a un transceptor. El sistema puede calcular: número de personas, su dirección y velocidad, así como detectar caídas. El protocolo de comunicación que utiliza es propio. (Figura 3.20, Figura 3.21).

- **Balizas.** Dispositivos de bajo consumo que emiten una señal broadcast, de tamaño reducido, generalmente para poderlos asociar a objetos o paredes. El protocolo que siguen para su comunicación es Bluetooth de bajo consumo (BLE). Existen diferentes dispositivos baliza y aunque la premisa es la misma para todos, podemos obtener funcionalidades diferentes dependiendo de la

Figura 3.20: Instalación de SensFloor

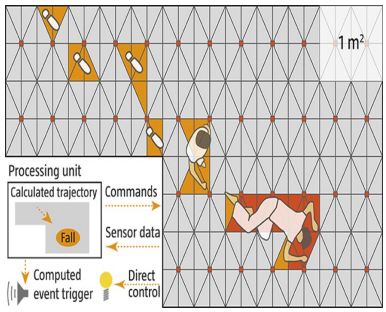

Figura 3.21: Aplicación de SensFloor

baliza. Existen dos formatos de balizas: **iBeacons**, desarrollada por Apple y **Eddystone**, propiedad de Google; la principal diferencia entre las dos reside en la gestión de paquetes y su identificación, ya que iBeacon utiliza un identificador denominado UUID mientras que Eddystone utiliza URLs de manera que podemos obtener la información incluso desde un navegador Web.

- **Estimote Beacons.** Diseñadas para el posicionamiento y la proximidad. Están optimizadas para durar hasta 3 años, dependiendo de las opciones de configuración. Siguen el formato iBeacon. A nivel de aplicación, con este dispositivo podemos posicionar a un sujeto en las diferentes estancias del Ambiente Inteligente. (Figura 3.22).
- **Estimote Stickers.** Bajo el paraguas de la tecnología "Nearable", estos dispositivos envían paquetes que no solo contienen una señal BLE

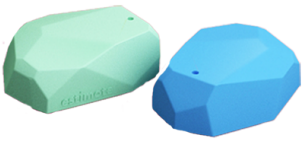

Figura 3.22: Estimote Beacons

sino que también envían un identificador, su estado de movimiento y la duración del mismo. Su duración aproximada, en cambio, es de un año. Los paquetes se identifican mediante el formato iBeacon. Su utilización se basa en adherirlas a objetos cotidianos con el fin de obtener datos referentes a la utilización de los mismos por parte de un usuario. (Figura 3.23).

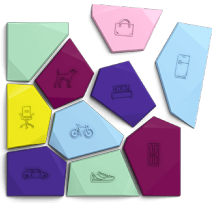

Figura 3.23: Estimote Stickers

- **Estimote Mirror.** La idea de este dispositivo es la de obtener una "Video Beacon". Para ello, transmite la presencia del usuario a los dispositivos cercanos, como smartphones o SmartTVs, de manera que las aplicaciones en ejecución en estos dispositivos sepan cuando hay un usuario dentro de su rango de acción e interactuar con él. El formato para la transmisión de información es iBeacon. A nivel de aplicación se utilizan para mostrar por la televisión, cuando se detecta que el usuario está enfrente, datos referentes al entorno. (Figura 3.24).
- **iBKS Beacons.** Desarrolladas en España, son compatibles con la tecnología iBeacon como con Eddystone, debido a ello, son muy configura-

Figura 3.24: Estimote Mirror

bles. Su duración es de 3 años. Sus aplicaciones van desde el posicionamiento hasta la detección de proximidades de objetos. (Figura 3.25).

Figura 3.25: iBKS Beacons

La Tabla 3.3 resume los dispositivos de localización de interiores analizados, indicando su fabricante, protocolo de comunicación y si posee un software asociado privativo.

Dispositivo de localización	Fabricante	Protocolo	Privativo
SensFloor	Future Shape	Propio	Sí
Beacons	Estimote	BLE	Sí
Beacons	iBKS	BLE	Sí
Stickers	Estimote	BLE	Sí
Mirror	Estimote	BLE	Sí

Tabla 3.3: Comparativa de dispositivos de posicionamiento de interiores

3.3.4. Dispositivos con sensor de visión

En esta sección se van a analizar los dispositivos con sensor de visión en el Smart Lab del CEATIC.

- **Cámara IP D-LINK 5020L.** Captura de vídeo, accesible mediante la red. Utilizadas a nivel de aplicación para "tracking" de personas y detección de caídas. (Figura 3.26).

Figura 3.26: D-LINK 5020L

- **Cámara Raspberry Pi.** Captura de vídeo. Entre sus aplicaciones destacadas encontramos la detección de presencia. (Figura 3.27).

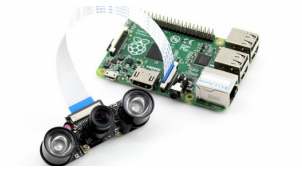

Figura 3.27: Cámara Raspberry Pi

- **Xbox Kinect.** Es otro sistema de control gestual, desarrollado por Microsoft y asociado a una Consola Xbox como dispositivo Host. (Figura 3.28).

Figura 3.28: Kinect

La Tabla 3.4 resume los dispositivos con sensores de visión, indicando su fabricante, protocolo de comunicación y si posee un software asociado privativo.

Dispositivo con sensores de visión	Fabricante	Protocolo	Privativo
5020L	D-LINK	Wi-Fi	Sí
Cámara Rasbberry Pi	Adafruit	Propio	No
Xbox Kinect	Microsoft	Wi-Fi, Propio	Sí

Tabla 3.4: Comparativa de dispositivos con sensores de visión de interiores

3.3.5. Dispositivos multimedia

En esta sección se van a analizar los dispositivos multimedia en el Smart Lab del CEATIC.

- **Samsung SmartTV 6400.** Televisión inteligente para mostrar información al usuario en tiempo real, realizar rehabilitaciones con simuladores virtuales o información detallada de lo que ocurre en el entorno.

Figura 3.29: Samsung SmartTV 6400

- **Consola Xbox ONE.** Videoconsola desarrollada por Microsoft, capaz de conectar periféricos de realidad aumentada como Kinect.

La Tabla 3.5 resume los dispositivos multimedia, indicando su fabricante, protocolo de comunicación y si posee un software asociado privativo.

Figura 3.30: Xbox ONE

Dispositivo multimedia	Fabricante	Protocolo	Privativo
SmartTV 6400	Samsung	Wi-Fi	Sí
Consola Xbox ONE	Microsoft	Wi-Fi	Sí

Tabla 3.5: Comparativa de dispositivos multimedia

3.3.6. Dispositivos de salud

En esta sección se van a analizar los dispositivos de salud en el Smart Lab del CEATIC.

- **Báscula inteligente Withings Smart Body Analyzer.** Con ella podemos medir de manera precisa tanto el peso como la masa corporal, además de la frecuencia cardíaca. También analiza la calidad del aire ambiental. Su protocolo de comunicación es tanto Wi-Fi como Bluetooth. (Figura 3.31).

Figura 3.31: Withings Smart Body Analyzer

- **Pulsera Withings Pulse 0x.** Con ella obtenemos datos como la frecuencia cardíaca, oxígeno en sangre, actividad física, calidad del sueño y temperatura corporal. Funciona bajo Bluetooth LE. (Figura 3.32).

Figura 3.32: Withings Pulse 0x

- **Xiaomi Mi Band 2.** Ofrece datos como la frecuencia cardíaca, calidad del sueño y actividad física. El protocolo de comunicación que utiliza es Bluetooth LE. (Figura 3.33).

Figura 3.33: Xiaomi Mi Band 2

- **Smartwatch LG Urbane.** Además de las funciones propias de un smartwatch, se aprovecha los sensores que posee, tales como acelerómetro, giroscopio o frecuencia cardíaca para su uso en aplicaciones de monitoreo de actividad humana en las tareas cotidianas. Se comunica mediante Wi-Fi y Bluetooth LE. (Figura 3.34).

Figura 3.34: LG Watch Urbane

- **Smartwatch Polar M600.** De la misma manera, se utilizan la gran cantidad de sensores que lleva el dispositivo para hacer mediciones precisas y recoger datos usados en aplicaciones como la detección de posiciones en la cama de pacientes sensibles a distintas enfermedades; por ejemplo, úlceras por presión. Su protocolo de comunicación es Bluetooth LE(Figura 3.35).

Figura 3.35: Polar M600

La Tabla 3.6 resume los dispositivos de salud, indicando su fabricante, protocolo de comunicación y si posee un software asociado privativo.

Dispositivo de salud	Fabricante	Protocolo	Privativo
Báscula Smart Body Analyzer	Withings	Wi-Fi, Bluetooth	Sí
SmartBand Pulse 0x	Withings	BLE	Sí
SmartBand Mi Band 2	Xiaomi	BLE	Sí
Smartwatch Urbane	LG	Wi-Fi, BLE	Sí
Smartwatch M600	Polar	Wi-Fi, BLE	Sí

Tabla 3.6: Comparativa de dispositivos de salud

3.3.7. Dispositivos de interfaces cerebrales

En esta sección se van a analizar los dispositivos de interfaces cerebrales en el Smart Lab del CEATIC.

- **BrainLink Macrotellect.** Se trata de un sensor de ondas cerebrales que utiliza Bluetooth como protocolo de comunicación. (Figura 3.36).

Figura 3.36: BrainLink Macrotellect

- **Emotiv Insight.** Proporciona datos referentes a las ondas cerebrales para su posterior análisis. También, utiliza Bluetooth como protocolo de comunicación. (Figura 3.37).

Figura 3.37: Emotiv Insight

- **Emotiv EPOC+.** Es un sensor encefalográfico de alta resolución. Además de Bluetooth, proporciona un protocolo de comunicación propio. (Figura 3.38).

Figura 3.38: Emotiv EPOC+

La Tabla 3.7 resume los dispositivos de interfaces cerebrales, indicando su fabricante, protocolo de comunicación y si posee un software asociado privativo.

Dispositivo de interfaz cerebral	Fabricante	Protocolo	Privativo
Macrotellect	BrainLink	Bluetooth	Sí
Insight	Emotiv	Bluetooth	Sí
EPOC+	Emotiv	Bluetooth, BLE	Sí

Tabla 3.7: Comparativa de dispositivos de interfaces cerebrales

3.3.8. Dispositivos con interfaz persona - ordenador

En esta sección se van a analizar los dispositivos con interfaz persona - ordenador en el Smart Lab del CEATIC.

- **Amazon Echo con Alexa.** Es un altavoz inteligente que, combinado con el asistente virtual Alexa ofrece variedad de posibilidades a nivel de aplicación como: interacción con los actuadores del Smart Lab del CEATIC, posibilidad de obtener información climatológica o programar alarmas. Utiliza Wi-Fi como protocolo de comunicación. (Figura 3.39).

Figura 3.39: Amazon Echo

- **Leap Motion.** Se trata de un sistema de control gestual capaz de reconocer con gran precisión gestos realizados con las mano, dedos u objetos. Se comunica mediante una conexión USB a un ordenador. (Figura 3.40).

- **Apple iPad Air 2.** Nos permite el acceso a aplicaciones desarrolladas para la interacción con usuarios en el Ambiente Inteligente. (Figura 3.41).

Figura 3.40: Leap Motion

Figura 3.41: Apple iPad Air 2

- **Samsung Galaxy Tab.** Debido al gran potencial del desarrollo Android, con ellas podemos realizar el etiquetado NFC para la detección de actividades, entre otras cosas. (Figura 3.42).

Figura 3.42: Samsung Galaxy Tab

La Tabla 3.8 resume los dispositivos con interfaz persona-ordenador multimedia, indicando su fabricante, protocolo de comunicación y si posee un software asociado privativo.

Dispositivos con interfaz persona - ordenador	Fabricante	Protocolo	Privativo
Echo con Alexa	Amazon	Wi-Fi	Sí
Leap Motion	Leap	Propio	Sí
iPad Air 2	Apple	Wi-Fi, BLE	Sí
Galaxy Tab	Samsung	Wi-Fi, BLE	Sí

Tabla 3.8: Comparativa de dispositivos con interfaz persona - ordenador

3.3.9. Dispositivos robots

En esta sección se van a analizar los dispositivos robots en el Smart Lab del CEATIC.

- **BQ Zowi.** Un robot inteligente y educativo, enfocado con la idea de hacer fácil la docencia de técnicas básicas de programación a personas de corta edad. (Figura 3.43).

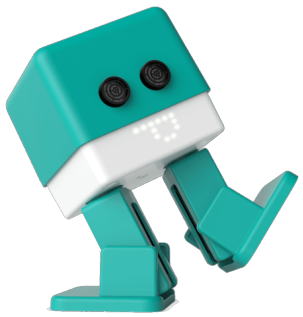

Figura 3.43: BQ Zowi

- **Bioloid Robotics.** Se trata de un robot humanoide, tecnológicamente muy avanzado, que posee, entre otras cosas: 18 servos para articular sus movimientos, 2 sensores IR, un sensor de distancia de alta precisión y giroscopio. Se pueden comunicar a través del protocolo ZigBee, ya que también poseen un módulo para ello. (Figura 3.44).

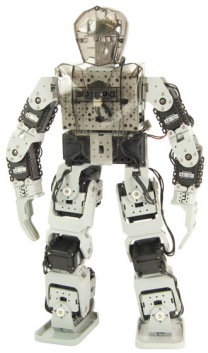

Figura 3.44: Bioloid Robotics

La Tabla 3.9 resume los dispositivos robots, indicando su fabricante, protocolo de comunicación y si posee un software asociado privativo.

Dispositivos robots	Fabricante	Protocolo	Privativo
Zowi	BQ	-	Sí
Bioloid Robotics	Bioloid	ZigBee, Propio	No

Tabla 3.9: Comparativa de dispositivos robots

3.4. Elección de la plataforma IoT a desplegar

El criterio fundamental para la elección de la plataforma IoT para desplegar en el Smart Lab del CEATIC es que sea compatible con el mayor número de dispositivos inteligentes que alberga el centro actualmente.

Fruto del análisis exhaustivo de plataformas IoT realizado en la Sección 3.1 junto al análisis en detalle del conjunto de dispositivos del Smart Lab del CEATIC realizado en la Sección 3.2, se deriva la Tabla 3.10 que muestra la compatibilidad de cada dispositivos del Smart Lab del CEATIC con cada una de las plataformas IoT revisadas durante este capítulo. Para realizar la tabla y para mejor comprensión del lector, se han utilizado una serie de acrónimos, siendo GH Google Home, SS Samsung SmartThings, D Domoticz, HA Home Assistant, OH OpenHAB,

SC Sensor Central y, finalmente, SW Sistema Web para la Monitorización de Ambientes Inteligentes.

Dispositivos	Fabricante	GH	SS	D	HA	OH	SC	Mv2.0
Contacto	Everspring	-	-	X	X	X	X	X
Contacto	Fibaro	-	-	X	X	X	X	X
Multipropósito	Samsung	-	X	-	-	X	-	X
Inundación	Fibaro	-	-	X	X	X	-	X
Presencia	Samsung	-	X	-	-	X	-	X
Movimiento	Samsung	-	X	-	-	X	-	X
Movimiento	Fibaro	-	-	-	-	X	-	X
Bombillas HUE	Philips	X	X	-	X	X	-	X
Altavoces Play:1	SONOS	-	-	-	-	X	-	-
Switch SmartHome	Harmony	-	-	-	-	X	-	-
Termostato	NEST	X	-	X	X	X	-	-
Detector de humo	NEST	X	-	X	X	X	-	-
Mother	Sense	-	-	-	-	-	-	-
Etiquetas NFC	-	-	-	-	-	X	X	X
SensFloor	Future Shape	-	-	-	-	X	X	-
Beacons	Estimote	-	-	-	-	-	X	X
Beacons	iBKS	-	-	-	-	-	-	X
Stickers	Estimote	-	-	-	-	-	-	X
Mirror	Estimote	-	-	-	-	-	-	-
Báscula Smart Body Analyzer	Withings	-	-	-	-	-	-	-
SmartBand Pulse 0x	Withings	-	-	-	-	-	-	-
SmartBand MiBand 2	Xiaomi	-	-	-	-	-	-	-
Macrotellect	BrainLink	-	-	-	-	-	-	-
Insight	Emotiv	-	-	-	-	-	-	-
EPOC+	Emotiv	-	-	-	-	-	-	-

Tabla 3.10: Dispositivos (Smart Lab) compatibles en plataformas IoT revisadas

Por tanto, la plataforma IoT que mayor compatibilidad de dispositivos del Smart Lab tiene es OpenHAB. Dicha plataforma IoT será desplegada en el Smart Lab del CEATIC.

Capítulo 4

Despliegue de la plataforma IoT

4.1. Introducción

Después de haber seleccionado la plataforma IoT para desplegar en el Smart Lab de acuerdo al mayor número de dispositivos compatibles, el siguiente objetivo a culminar en este trabajo fin de grado es el despliegue de dicha plataforma cubriendo las necesidades específicas del CEATIC. Para ello, es necesario estudiar las necesidades específicas del Smart Lab y analizar en profundidad la plataforma IoT que ha sido seleccionada para su despliegue, en nuestro caso OpenHAB. Fruto de este análisis, se desarrollará el despliegue inicial de la plataforma IoT, su configuración nativa de acuerdo a las necesidades y, finalmente, la funcionalidad adicional que es necesario implementar.

Por tanto, este capítulo queda organizado del siguiente modo. En primer lugar, se indicarán las necesidades específicas del laboratorio. Posteriormente, se describirá en detalle la plataforma IoT seleccionada, OpenHAB. A continuación, se detallará el despliegue que ha sido llevado a cabo para finalmente indicar la funcionalidad que ha sido llevada a cabo a medida para cubrir las necesidades.

4.2. Necesidades específicas del Smart Lab

En esta sección, se van a exponer las necesidades específicas a cubrir del Smart Lab bajo estudio. Dichas necesidades han sido indicadas por la junta directiva del CEATIC de acuerdo a las estrategias propias del centro.

A continuación, se indican el listado de necesidades específicas del Smart Lab:

1. Conexión con las plataformas IoT de investigación IoT de "Sensor Central" y "Sistema Web para la Monitorización de Ambientes Inteligentes". La plataforma IoT de OpenHAB brinda la funcionalidad óptima para la integración de la mayoría de dispositivos IoT del Smart Lab. Sin embargo, como se indicó en el capítulo anterior, carece de características propias de investigación. Por tanto, es necesario comunicar a la plataforma de OpenHAB con ambas plataformas y así adquirir la funcionalidad de investigación.

2. Conexión con Redes Sociales. Se desea representar los datos obtenidos en el Smart Lab en la red social Twitter [46], con el fin de darle difusión en esta red social.

3. Integración completa con los dispositivos que ya se encuentran instalados y posibilidad de incluir nuevos. Como hemos visto en el capítulo anterior, OpenHAB ofrece la posibilidad de incluir nuevos dispositivos con un índice de compatibilidad bastante alto, pero también es necesario que los que hay desplegados en el Smart Lab se puedan gestionar por la plataforma sin generar problemas de integración o incompatibilidad.

4. Visualización del estado de los sensores del Smart Lab a través de una interfaz amigable y sencilla que pueda ser accesible desde el exterior. Se necesita obtener una interfaz mediante la cual se pueda observar el estado del Smart Lab, definido por los sensores que se encuentran funcionando en dicho Smart Lab.

5. Interfaz que ofrezca control general sobre los dispositivos del Smart Lab, además se debe tener en cuenta la interacción con los dispositivos actuadores. Ante una plataforma que permite la inclusión de tantos dispositivos heterogéneos, es importante tener una interfaz con la que poder gestionar esos dispositivos, así como para tener el control general de la plataforma. Además, es necesario tener algún mecanismo con el que poder interactuar con los diferentes actuadores instalados en el Smart Lab. Esta interfaz, debe presentar la información de una manera categorizada y debidamente organizada.

6. Visualización de las imágenes captadas por las cámaras a través de una interfaz dedicada. Debido a que cada una de las cámaras tiene su propio software con el que poder visualizar las imágenes captadas por el dispositivo, estas imágenes se obtienen de fuentes diferentes, es decir, de manera separada. Es necesario obtener una representación de las imágenes que lo haga de manera conjunta para todas las cámaras y se pueda realizar un repaso de las mismas conjuntamente y de este modo tener una visión general.

7. Almacenamiento de los datos recogidos y generados por el Smart Lab. Además de poder compartir los datos con otras plataformas, estos datos deben de almacenarse de una manera estructurada, segura y organizada en la propia infraestructura de soporte del centro.

8. Mecanismo de automatización de procesos. Dentro del contexto al que pertenece el Smart Lab del CEATIC, el de un Smart Lab académico, es importante que la plataforma que se despliegue tenga un mecanismo de automatización de procesos y además, que el mecanismo no presente incompatibilidades con los dispositivos instalados en este Smart Lab según la categoría a la que pertenezcan (dispositivos con sensores ambientales, dispositivos actuadores, dispositivos multimedia), es decir, que sea un mecanismo que sirva para au-

tomatizar procesos, pero que brinde la posibilidad de extenderlo a todos los dispositivos.

9. Comunicación con aplicaciones propias del CEATIC, desarrolladas para dispositivos móviles así como wearables. Es importante que la plataforma que se va a desplegar se pueda comunicar con las distintas aplicaciones móviles que el CEATIC tiene en uso y que son de desarrollo propio. El objetivo es poder obtener los datos del Smart Lab en los diferentes métodos de acceso a la información con los que trabajan los investigadores, así como el personal del CEATIC.

Las necesidades que se han enumerado previamente deberán ser cubiertas por la plataforma IoT de OpenHAB de manera nativa mediante componentes o con implementación adicional.

4.3. OpenHAB en detalle

En esta sección, se va a describir la arquitectura de la plataforma IoT de OpenHAB con el fin de comprender su estructura y su lógica de funcionamiento. Para así poder hacer una explotación correcta de la misma y poder alinear las posibilidades que ofrece con las necesidades básicas del Smart Lab. Posteriormente, se indicarán los componentes que han sido desplegados para cubrir las necesidades específicas del laboratorio.

4.3.1. Arquitectura de OpenHAB

A continuación, en este apartado se van a revisar los detalles de la arquitectura de OpenHAB así como su estructura, funcionamiento y sus componentes más relevantes.

Como podemos observar en la Figura 4.1, OpenHAB está basado en el framework Eclipse SmartHome, Apache Karaf, Eclipse Equinox y rutinas OSGi (Open Services Gateway initiative). El objetivo de esta sección es conocer el funcionamiento de OpenHAB y entender su arquitectura, pero enfocado a cubrir una serie de necesidades, por lo que revisaremos más en profundidad el Framework "Eclipse Smart Home" que es la base sobre la que se asienta la plataforma y lo que nos brinda en una capa inferior, cubriendo las necesidades que se han especificado tanto en el capítulo anterior como en la primera sección de este capítulo.

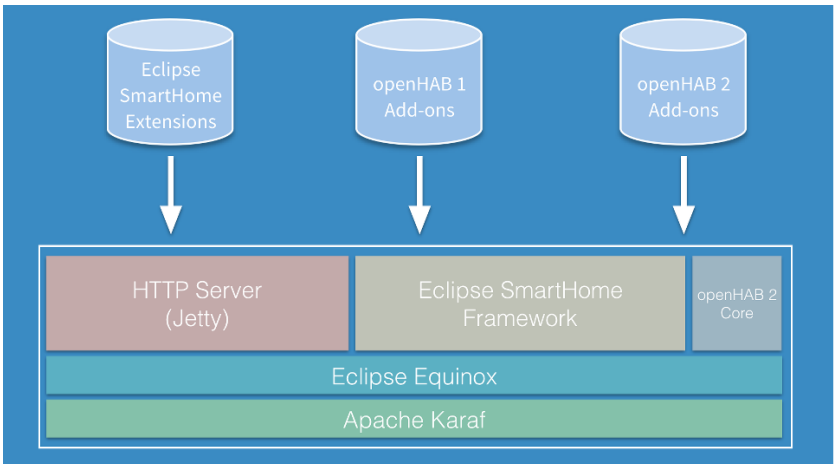

Figura 4.1: Arquitectura de OpenHAB

Eclipse SmartHome

Eclipse SmartHome es un framework para construir soluciones enfocadas al hogar. Con una arquitectura muy flexible, fomenta la modularidad proporcionada por las aplicaciones OSGi para Java. Como tal, Eclipse SmartHome consiste en un

Figura 4.2: Logo Eclipse SmartHome

amplio conjunto de paquetes OSGi que sirven para diferentes propósitos. No todas las soluciones que se basan en Eclipse SmartHome requerirán todos los paquetes, sino que se pueden elegir qué partes son interesantes para ellos, aunque OpenHAB los utiliza todos.

Los paquetes que ofrece este framework están divididos en las siguientes categorías:

- **config:** todo lo que se refiere a la configuración general del sistema como archivos de configuración, análisis XML, descubrimiento, etc.
- **core:** son los principales paquetes para el funcionamiento lógico del sistema, basado en el elemento abstracto y los conceptos de evento.
- **io:** este paquete ofrece todo tipo de funcionalidad opcional que tiene que ver con entrada y/o salida como comandos de consola, soporte de audio o comunicación HTTP/REST.
- **model:** aporta el soporte necesario para que se puedan utilizar lenguajes específicos de dominio (DSL).
- **designer:** proporciona soporte "Eclipse RCP" para DSL y otros archivos de configuración.
- **ui:** paquetes relacionados con la interfaz de usuario que proporcionan servicios que pueden ser utilizados por diferentes UI, como gráficos o iconos.

Al tratarse de un framework, Eclipse SmartHome define muchos tipos de extensiones, las cuales permiten construir soluciones modulares con componentes conectables (extensiones), lo que dotará a la plataforma de un nivel alto de compatibilidad y modularidad. Por tanto, se podrán separar las diferentes características que sean necesarias y obtener la funcionalidad requerida en el Smart Lab.

Funcionamiento

Como hemos apuntado en la introducción de este capítulo, es importante conocer la plataforma que se va a desplegar, ya que depende de ello que las necesidades planteadas se puedan afrontar con garantías.

Según podemos ver en la Figura 4.3, el funcionamiento de OpenHAB se basa en la interacción de diferentes elementos, cada uno con una función específica dentro de la lógica de la aplicación.

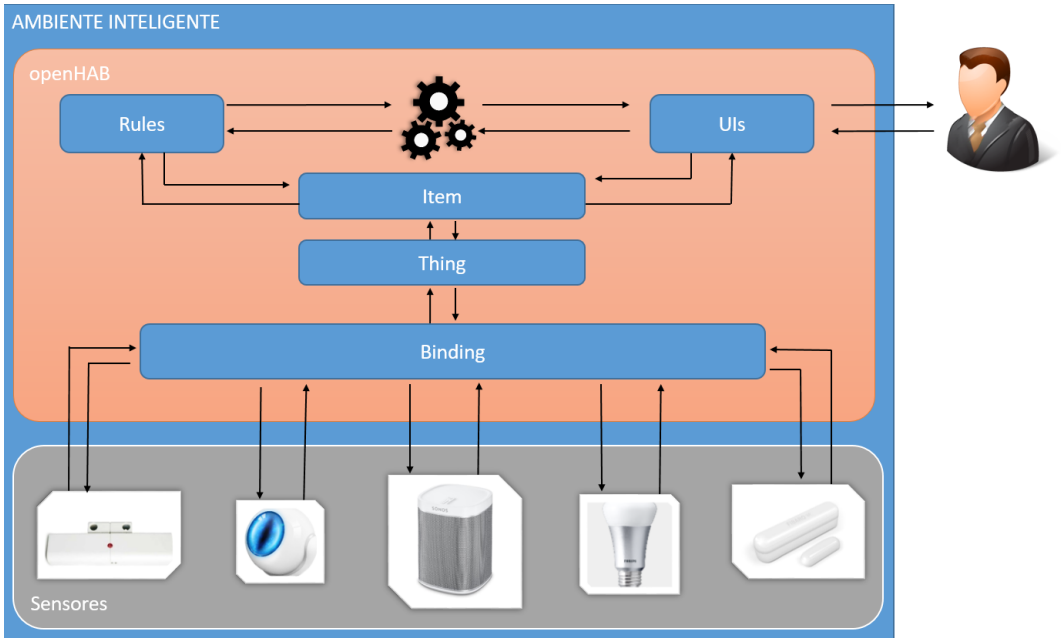

Figura 4.3: Esquema de funcionamiento de OpenHAB

A continuación, se hace referencia a los dos conceptos más importantes en lo que a la gestión de dispositivos dentro de la aplicación se refiere. Son, por tanto, dos conceptos importantes a reseñar, ya que el entendimiento de la lógica del funcionamiento de esta plataforma se basa, en gran medida, en estos conceptos.

- **Things**. Son entidades que se pueden agregar físicamente a un sistema. Las things pueden proporcionar más de una función (por ejemplo, un multisensor Z-Wave puede proporcionar un detector de movimiento y también,

medir la temperatura ambiente). Las things ofrecen la información a través de canales. No es necesaria la utilización de todos los canales que presente un thing puesto que cada uno de ellos ofrecerá un tipo de información. Los que utilicemos variarán en función de la información que queramos obtener de ese thing.

Mediante things obtenemos la representaciones de objetos físicos, por tanto, un sensor estará asociado a un thing, como este sensor nos ofrece varios canales de información se puede representar la información que ofrece cada uno de esos canales mediante items 4.3.1.

- **Items**. Un item representa las propiedades y capacidades de un thing. Un thing posee varios canales de comunicación y cada canal obtiene su representación con un item, es decir, si un thing proporciona tres canales pero solo se enlaza uno a un item, ese thing sólo podrá obtener la información que proporciona ese canal. Por ejemplo: Sensor de contacto Everspring que posee tres canales (contacto, switch, nivel de batería), en caso de interesar sólo la parte de contacto, se asociaría el canal "contact" de ese sensor al objeto donde esté ubicado ese sensor. Para asociar un item a un canal de un thing, el item debe estar creado previamente.

  Como hemos dicho anteriormente, los items representan las capacidades que pueden usar las aplicaciones, ya sea en las interfaces de usuario o en la lógica de automatización. Los elementos tienen un estado y pueden recibir comandos.

  La conexión entre things e items son los bindings 4.3.2. Un binding, como se ha explicado, es una asociación entre un canal y un item. Si un canal está vinculado a un elemento, está "habilitado", lo que significa que la capacidad que el elemento representa es accesible a través de ese canal. Los canales pueden estar vinculados a varios elementos y los elementos pueden estar

vinculados a varios canales.

Entonces, la diferencia entre un thing y un item es que un thing sólo está asociado a una entidad, generalmente física, mientras que el item obtiene la información de los canales de este dispositivo. Esta diferencia se ilustra en la Figura 4.4, donde podemos ver que un solo thing obtiene información con dos canales, por tanto, se tendrá que representar mediante dos items diferentes.

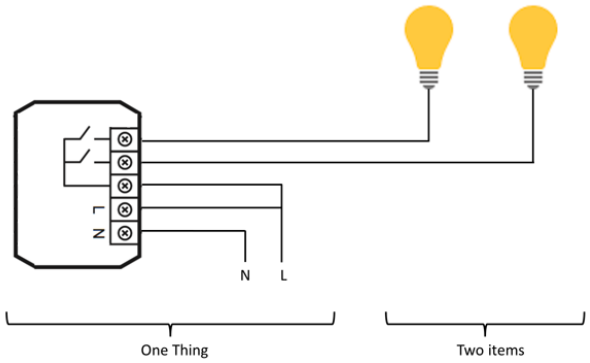

Figura 4.4: Ejemplo de actuador

El actuador es un thing que podría instalarse en un armario eléctrico. Tiene una dirección física y debe configurarse para poder ser utilizada. Para que el usuario controle las dos luces, accede a la capacidad de thing del actuador (encender y apagar dos luces separadas) a través de dos canales, cada uno de ellos representados por un item diferente. También, que están vinculados a dos elementos de interruptor presentados al usuario a través de una interfaz de usuario.

Una vez revisados los detalles importantes del funcionamiento y la estructura de OpenHAB, en la siguiente sección haremos referencia a los componentes de OpenHAB, ofrecidas de manera nativa por la plataforma con las que abordaremos diversas necesidades requeridas por el CEATIC.

4.3.2. Componentes básicos a desplegar

En este aparatado, se van a describir los componentes básicos de OpenHAB que han sido analizados para la funcionalidad requerida en el Smart Lab.

Bindings

Los Bindings son la parte software que sirve como enlace para integrar hardware físico a OpenHAB. Cuando obtenemos un dispositivo nuevo, que antes no teníamos configurado en el sistema, lo primero que debemos hacer es buscar su Binding e instalarlo en caso de que no estuviera instalado ya.

Los Bindings pueden incluir opcionalmente servicios de descubrimiento, que permiten que el sistema encuentre automáticamente dispositivos accesibles.

Esta es la parte de la plataforma que nos permite satisfacer la necesidad específica número 3 de la Sección 4.2, ya que mediante los distintos Bindings podremos incluir en la plataforma IoT los dispositivos existentes en el Smart Lab del CEATIC, además de brindarnos un mecanismo de inclusión para los posibles nuevos dispositivos.

Los bindings que ha sido necesario instalar son los siguientes:

- **Sonos:** nos permite comunicarnos con los altavoces e interactuar con ellos.
- **HUE:** reconoce el hub Philips y de manera transparente el sistema se comunica con él para poder interactuar con todas las bombillas Philips HUE.
- **Z-Wave:** mediante este binding se obtiene la implementación del protocolo Z-Wave. Puesto que la mayoría de sensores que se encuentran instalados en el Smart Lab utilizan Z-Wave como protocolo de comunicación como se ha visto en la Sección 3.3, este Binding es uno de los más importantes.
- **Nest:** permite reconocer hardware de la marca NEST, de manera que los dispositivos de detección de humo y el termostato son accesibles desde la

plataforma.

- **Samsung TV:** permite la interacción con televisiones Samsung, por lo que mediante él podemos interactuar con la Smart TV del Smart Lab.

User Interfaces

Las interfaces de usuario son aplicaciones nativas de smartphone o web que acceden al servidor OpenHAB a través de la API REST como veremos en la Sección 4.3.2. Mediante estas interfaces, OpenHAB nos ofrece la posibilidad de crear visualizaciones propias de los datos, así como de gestión de los dispositivos. Con lo cual, podremos cumplir con varias necesidades específicas del Smart Lab.

A continuación, tenemos las interfaces de usuario que se han instalado y configurado en el Smart Lab del CEATIC:

Basic UI

Esta interfaz nos da acceso a los sitemap, en una instalación de OpenHAB podemos tener varios sitemaps aunque lo recomendable es tener uno por ambiente inteligente.

Un sitemap es una representación compuesta por elementos físicos o lógicos orientada al usuario con el fin de facilitar una herramienta de gestión y explotación del ambiente inteligente. En definitiva, el sitemap se construye anidando frames con elementos dentro de este; algunos detalles de la construcción de un sitemap, así como sus elementos, los encontramos en la Figura 4.5 y en la Figura 4.6.

```
Frame label="Control General" {
        Switch item=Puerta          mappings=[ON="Abrir",OFF="Cerrar"]
        Switch item=General_Iluminacion         mappings=[ON="Encender",OFF="Apagar"]
        Switch item=ScenesButtom icon="video" mappings=[0="Relajarse", 1="Lectura", 2="Fiesta", 3="Bienvenida"] label="Escenas"
        Colorpicker    item=General_Color icon="colorlight" label="Color"
        Slider item=General_Intensidad label="Brillo"
        Switch item=SwGotoSleep  mappings=[ON="Dormir",OFF="Despertarse"] icon="bedroom"
        Default item=Sonos_Control_General
        //Text item=Presencia_Dani label="Presencia Dani" icon="boy_5"
        //Text item=Presencia_Macarena  label="Presencia Macarena" icon="girl_5"
}
```

Figura 4.5: Elementos del sitemap 1

Figura 4.6: Elementos del sitemap 2

Atendiendo a las necesidades específicas número 4 y 5, se ha desarrollado un sitemap que facilite la visualización de los datos recogidos por los sensores, así como al mismo tiempo, ofrezca control sobre los actuadores del Smart Lab. Este sitemap esta estructurado por zonas. Existen 4 zonas en el Smart Lab, por tanto, la interfaz se divide en 4 categorías y en cada una de ellas se puede interactuar con los dispositivos que se encuentran instalados en ella. Adicionalmente, en este sitemap se pueden visualizar las imágenes de las cámaras de visión, pero debido a que esta dividido por zonas, no cumple con la necesidad específica número 6, ya que la interfaz a la que hace referencia esta necesidad debe ser dedicada. Por tanto, esta es una necesidad que se tratará en la Sección 4.3.2.

El desarrollo se realiza a través de un script que carga la plataforma una vez completo, cada elemento se asocia a un item y a una función. Los elementos que se utilizan para su desarrollo los proporciona la propia plataforma y son los siguientes:

- **Switch:** se trata de un botón, que mapea dos estados y se usa para activar o desactivar actuadores, como por ejemplo, las bombillas.
- **Colorpicker:** es un selector de color, también está asociado a las luces.
- **Slider:** se refiere a una barra de desplazamiento que sirve de selector, es utilizada tanto para el brillo de las luces como para, el volumen de los altavoces.
- **Text:** este elemento muestra una cadena de texto especificada. Es utilizada para mostrar los estados de los sensores, así como los datos que recogen.

- **Image:** proporciona un visor de imágenes que se utilizan para enlazar la reproducción de vídeo de las cámaras al sitemap.
- **Selection:** este elemento proporciona un selector de contenido. este puede estar preestablecido, pudiendo ser utilizado para realizar cambios en actuadores que tienen más de dos estados y pueden estar predefinidos, como la reproducción de sonido por los altavoces.
- **Webview:** con este elemento podemos integrar frames proporcionados por otros sistemas para visualizar contenido en el sitemap. Se utiliza para integrar un mapa de localización de sensores que ofrece el Sistema para Monitorización de Ambientes Inteligentes visto en 3.2.2.2. La introducción de este elemento se encuadra dentro de la necesidad específica número 1.

Paper UI

OpenHAB es una plataforma IoT compatible con infinidad de dispositivos, como hemos visto en la Sección 2.3.1 del Capítulo 2. Ante la aparición constante de nuevas necesidades basadas en la inclusión de nuevos dispositivos, es necesario tener un mecanismo de gestión general de la plataforma IoT que nos permita controlar los aspectos generales más importantes. Como se ha indicado en la necesidad específica número 5, es importante implantar en este despliegue una interfaz que permita controlar, de manera general, la plataforma IoT y todos los componentes que la forman. Refiriéndonos a la necesidad específica número 5,

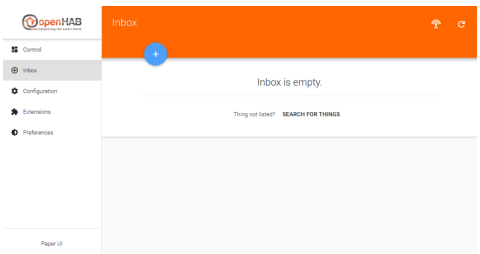

Figura 4.7: Interfaz PaperUI

mediante el punto anterior, se resuelve la parte que tiene que ver con los dispositivos actuadores, ya que mediante el sitemap, se pueden controlar los dispositivos actuadores. Pero es en este punto donde la necesidad se resuelve por completo, ya que la interfaz que podemos observar en la Figura 4.7 nos proporciona, de manera general: gestión de bindings, servicios, things e items. Nos proporciona gestor de módulos, además de un motor de reglas que en secciones próximas se detallará. También, una pequeña sección de "control" donde podemos interactuar con los diferentes items, además están categorizados por su ubicación dentro del ambiente (dormitorio, cocina, entrada, salón).

Particularmente, con la interfaz Paper UI se pueden realizar todas las tareas de mantenimiento y soporte requeridas por la plataforma IoT, a parte de todas las tareas que tienen que ver con la inclusión de nuevos dispositivos.

HABPanel

Esta interfaz proporciona, de una forma muy visual, y con diseño responsive que la hace una opción muy interesante para dispositivos móviles, la información deseada.

Para cumplir con la necesidad específica número 6, se ha optado por esta interfaz, puesto que además de presentar un diseño responsive que la hace compatible con todos los dispositivos, presenta una serie de características, como la visualización de vídeo, que hacen de esta interfaz la mejor opción para montar un sistema de visualización dedicado para las cámaras del Smart Lab del CEATIC.

Figura 4.8: Visualización responsive

API REST

A través de la API REST, se obtiene el mecanismo de comunicación necesario para que OpenHAB puede comunicarse con otras plataformas y pueden acceder fácilmente a todos los datos relacionados con items, así como disparar acciones que pueden cambiar el estado de los items. Las interacciones con la API REST se basan en el protocolo http [47].

Como hemos dicho, la API REST permite el acceso de lectura a estados, actualizaciones de los mismo, pero también permite el envío de comandos a otras plataformas con lo que, de esta manera, se pueden compartir los datos generados por los dispositivos del Smart Lab a otras plataforma. Con lo cual, obtenemos el mecanismo necesario para cumplir con la necesidad específica número 1.

Reglas

La plataforma OpenHAB, permite de modo nativo, la creación de una serie de reglas de automatización que se pueden utilizar con el fin de automatizar procesos. OpenHAB es una plataforma con un enfoque "device-agnostic" que es capacidad de un componente de computación para trabajar con varios sistemas sin requerir adaptaciones especiales. Al ser una plataforma "device-agnostic" nos permite interactuar con los dispositivos sin la necesidad de que estos deban tener requisitos o funcionalidades especiales, por tanto, mediante este mecanismo podemos satisfacer la necesidad especifica número 8, ya que, no solo nos permite automatizar los procesos, sino también realizar esta automatización haciendo participes de ella a todos los dispositivos instalados en el Smart Lab del CEATIC.

El funcionamiento de este mecanismo está basado en un esquema disparador-acción y consiste en que la regla que determinemos espera un evento que será el que dispare la acción, previamente definida en la regla y una vez producido ese evento, ejecuta la acción correspondiente, también definida en la regla en cuestión. Cuando se activa una regla puede realizar tareas como: ejecutar un script, hacer

uso de la API REST, hacer cálculos matemáticos, etc.

La plataforma OpenHAB nos permite realizar la gestión e implementación de estas reglas mediante dos formas diferentes. Por un lado, proporciona un motor de reglas, con el cual podemos realizar los procesos de automatización mas básicos; por otra parte, brinda la posibilidad de programar estas reglas a través de scripts, y es aquí donde reside la potencia de este mecanismo ya que nos permite definir los procesos de automatización deseados y programarlos teniendo en cuenta aspectos como: disparador, tareas programadas en el tiempo, acciones definidas, por ejemplo, envío de datos a otras plataformas.

Debido a la importancia de este componente, es necesario realizar una pequeña reseña a las diferentes formas de implementación que nos brinda la plataforma. A continuación, se van a mencionar los detalles de cada una de ellas.

- **Rule Engine.** El motor de reglas nos permite crearlas reglas visualmente. Aunque su uso se reduce, es una manera muy accesible de automatizar procesos sin tener nociones de programación. En la Figura 4.9 podemos observar de que manera se presenta este motor.

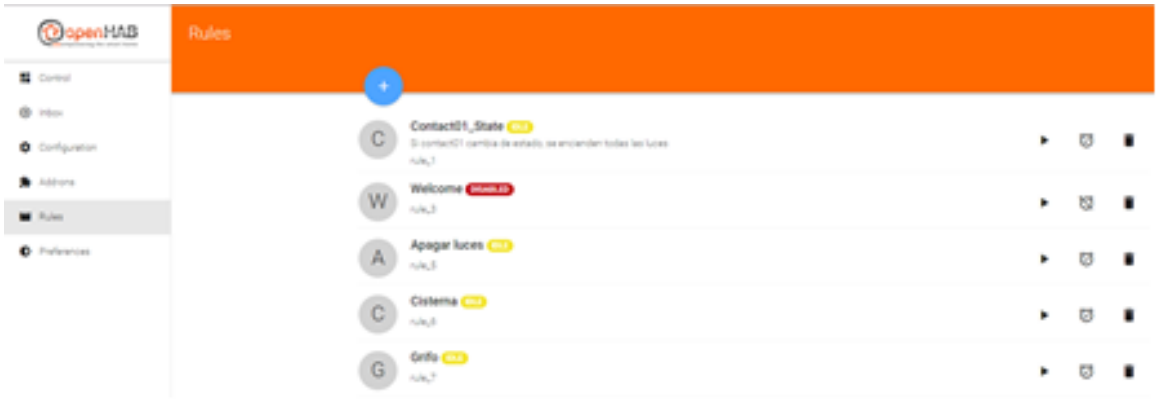

Figura 4.9: Motor de Reglas

- **Programación de Reglas.** Para explotar verdaderamente todo el potencial de las reglas, tenemos un fichero en el que las podemos programar. La sintaxis de las reglas en este fichero es la siguiente:

rule rule name"when <TRIGGER CONDITION1> or
<TRIGGER_CONDITION2> or <TRIGGER_CONDITION3> ...then
<SCRIPT_BLOCK>end

Existen a su vez, diferentes tipos de disparadores con los que poder realizar una mejor gestión de los procesos. Los tipos de disparadores que se pueden programar son los que se mencionan a continuación.

- Basados en items: Item <item> received command [<command>], Item <item> received update [<state>], Item <item> changed [from <state>] [to <state>]
- Basados en tiempo: Time is midnight, Time is noon, Time cron "<cron expression>".
- Basados en el sistema: system started, system shutdown.

Una vez revisados los componentes de la plataforma necesarios para satisfacer algunas de las necesidades planteadas en la Sección 4.2, vamos a entrar en profundidad en las cuestiones de despliegue, así como en la implementación de las funcionalidades adicionales requeridas en el Smart Lab del CEATIC.

4.4. Despliegue de OpenHab en el Smart Lab

En esta sección se van a detallar todos los aspectos necesarios para el despliegue de la plataforma en el Smart Lab del CEATIC, atendiendo a las necesidades específicas que se han marcado en la Sección 4.2. La sección comienza realizando una reseña de toda la plataforma IoT del Smart Lab, también se hace alusión a los detalles de la implementación inicial y para terminar, se detallan dos funcionalidades realizadas, como son la conexión a otras plataformas externas y la gestión de persistencia.

4.4.1. Esquema operativo de la plataforma IoT

Para llevar a cabo el despliegue de la plataforma IoT que se va a realizar en el Smart Lab del CEATIC, es preciso, en primer lugar, definir un esquema que será el que se tenga como referencia, el cual se seguirá para llevar a cabo con éxito el despliegue. Este esquema se ha diseñado con el objetivo de satisfacer todas las necesidades demandadas por la junta de dirección del CEATIC y poder cumplir con todos los objetivos y necesidades específicas arrojadas por la propuesta del presente trabajo.

El esquema que se ha diseñado para el despliegue de la plataforma IoT OpenHAB es el que se puede observar en la Figura 4.10 y que se detalla a continuación.

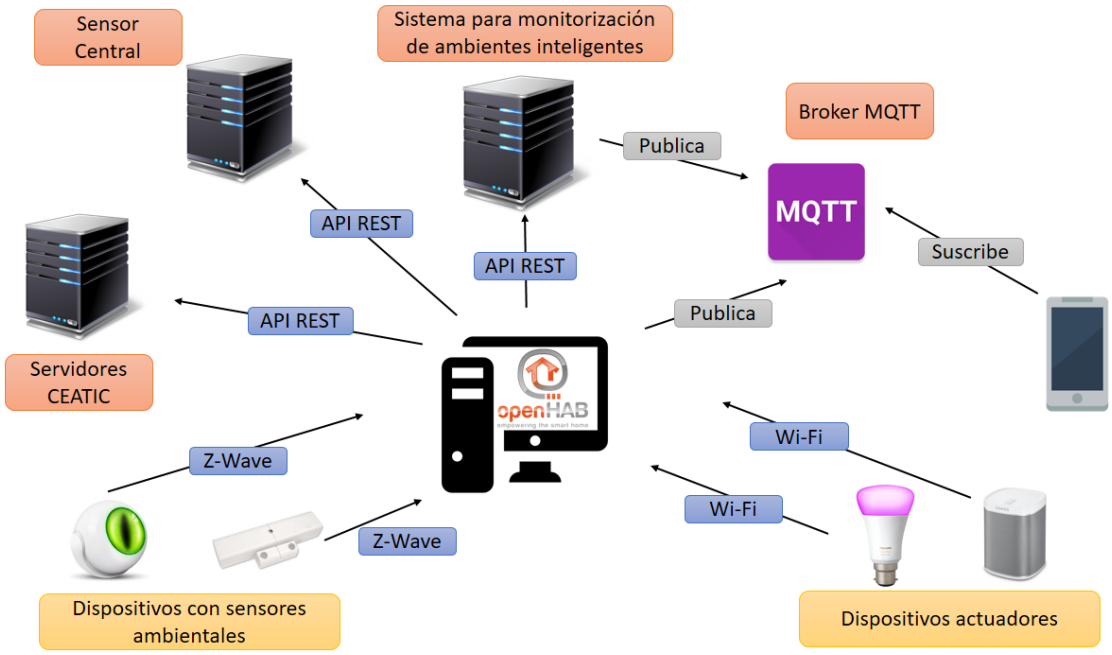

Figura 4.10: Esquema operativo de la plataforma IoT del Smart Lab

Como se ha explicado antes, el objetivo es cumplir las necesidades establecidas en el análisis de este trabajo, para ello, nuestra plataforma debe aportar los mecanismos de comunicación necesarios para que se puedan realizar con garantías las conexiones con las plataformas y sistemas con las que se va a interactuar.

Tras el estudio realizado en la Sección 4.3.2 de los componentes básicos de OpenHAB que se necesitan para satisfacer todas las necesidades, se sabe que el principal método de comunicación con las plataformas externas será el brindado por la API REST. Debido a que una de las necesidades específicas reseñadas es la posibilidad de conexión con otras plataformas orientadas a la investigación, y más concretamente, las especificadas que son Sensor Central, revisado en la Sección 3.2.2.1, y el Sistema Web para la Monitorización de Ambientes Inteligentes, revisado en la Sección 3.2.2.2. La parte más importante de este esquema es resolver ese enlace que tendrá la plataforma IoT con las otras plataformas. El protocolo de comunicación mediante el cual se realizará la conexión con las dos plataformas IoT orientadas a la investigación será el proporcionado por la API REST.

Además de con las dos plataformas IoT orientadas a la investigación, las necesidades específicas también detallan que se debe contemplar un mecanismo de persistencia que garantice la seguridad de los datos. Este mecanismo de persistencia se llevará a cabo mediante la infraestructura propia del centro, por lo que la conexión con los servidores del CEATIC también es importante. Esta conexión se realizará, igual que para las plataformas IoT orientadas a la investigación, mediante los servicios proporcionados por la API REST.

Según la necesidad específica número 10, también se debe contemplar la opción de que la plataforma desplegada tenga los mecanismos de comunicación necesarios para poder realizar un intercambio de información con las aplicaciones propias desarrolladas por el centro. Como podemos observar en la parte derecha del esquema de la Figura 4.10, este proceso se lleva a cabo mediante un servidor de MQTT [48].

MQTT es un protocolo de mensajería de comunicación Machine-to-Machine (M2M) basado publicación/suscripción. Se basa en minimizar el ancho de banda de la red y los requerimientos de uso de los dispositivos. Es extremadamente simple y ligero y por tanto, ideal para IoT y aplicaciones móviles donde es crítico tanto

el ancho de banda como el consumo de energía.

El servidor que gestiona todo el tráfico MQTT se denomina "Broker" y forma parte de la infraestructura del CEATIC. Teniendo el servidor MQTT disponible, la comunicación de la plataforma IoT con las aplicaciones móviles del centro es sencilla puesto que la plataforma publicará en el servidor MQTT los datos relativos a los dispositivos, mientras que las aplicaciones, siguiendo la lógica del protocolo MQTT, estarán suscritas al canal donde la plataforma IoT a desplegar realice la publicación de la información. Como se puede apreciar en el esquema, la plataforma "Sistema para la monitorización de Ambientes Inteligentes" también realiza publicaciones de datos en el servidor MQTT, con el fin de homogeneizar los datos a los que pueden acceder las aplicaciones, incluyendo los que gestiona esta plataforma que son orientados a la investigación.

La última parte del esquema general de la plataforma IoT del Smart Lab, es la capa física. Todos los dispositivos actuadores y dispositivos con sensores ambientales se comunican con la plataforma IoT mediante los protocolos de comunicación vistos en la Sección 2.3.1. Como podemos observar en el esquema, no aparecen todos, se instancia con los ejemplos más usuales como son: los dispositivos con sensores ambientales que realizan la conexión con la plataforma mediante el protocolo de comunicación Z-Wave y los dispositivos actuadores que la realizan por red ya sea Wi-Fi o cableada.

Una vez repasados los detalles del esquema general de la plataforma IoT del Smart Lab del CEATIC, se revisará en los apartados siguientes los aspectos más relevantes sobre la implementación de la misma.

4.4.2. Implementación inicial

A continuación, se describirá cada uno de los medios físicos por los que está compuesto todo el sistema OpenHAB, sistema operativo, entre otros.

Para su despliegue ha sido utilizada una máquina virtual, de este modo tendre-

mos un mayor control sobre las versiones del sistema y se podrán realizar copias de seguridad periódicamente.

Para la instalación de OpenHAB en este trabajo fin de grado se utilizará una máquina virtual con el sistema operativo Linux. Todas las pruebas que se muestran en las ilustraciones inferiores se han llevado a cabo con la distribución Debian 8.

Finalmente, la máquina virtual estará albergada dentro del Smart Lab con la siguiente información:

- Nombre de la máquina virtual: OpenHAB
- Usuario: openhab
- IP: 192.168.83.217

Tras realizar la instalación del sistema operativo, podremos acceder a la máquina virtual mediante:

- Físicamente con usuario y contraseña.
- Mediante SSH: generalmente se utilizan sesiones SSH desde la línea de comandos de Linux o desde Windows, utilizando como cliente PuTTy o WinSCP.
- Mediante TeamViewer: con las sesiones SSH no tenemos entorno gráfico, por tanto, y aunque no es lo habitual, se facilita acceso mediante escritorio remoto con TeamViewer.

Como se ha comentado anteriormente, una de las facilidades que nos brinda la máquina virtual es realizar copias de seguridad de una forma segura y rápida, para ello, tras realizar la copia de seguridad, esta será albergada en una cuenta cooperativa de Google Drive.

Una vez revisados los detalles relativos a los medios físicos utilizados y al acceso a ellos, cabe reseñar algunos aspectos importantes sobre la plataforma OpenHAB

que serán cruciales a la hora de realizar este despliegue en el Smart Lab del CEATIC.

Lo primero que tenemos que tener en cuenta es que la plataforma OpenHAB está diseñada para que su funcionamiento sea como el de un servicio del sistema. El nombre del servicio es "openhab2" y las operaciones más usuales que se realizan con él son: iniciarlo, pararlo o reiniciarlo. Para esta tarea, usaremos la herramienta systemctl, la cual esta instalada en el sistema. A continuación, se detallan las órdenes de la línea de comandos utilizadas.

- Start: sudo systemctl start openhab2
- Stop: systemctl stop openhab2
- Restart: systemctl restart openhab2

El otro punto importante a revisar para realizar el despliegue de la plataforma y las posteriores implementaciones que se realizarán en ella es el árbol de directorios que presenta OpenHAB, para ello, a continuación se describen los detalles más importantes de este árbol de directorios.

El directorio raíz de OpenHAB es /etc/openhab2. Dentro del directorio raíz tenemos, separados por directorios, todos los puntos de configuración de OpenHAB.

- **html:** Destinado a guardar ficheros de código html, para mostrar vía web. Aquí se encuentra, por ejemplo, la página por defecto que muestra el servicio web de OpenHAB nada más ser instalado. No es de mucha utilidad una vez OpenHAB está configurado.
- **icons:** si se desearan incluir iconos personalizados habría que guardarlos en este directorio para poder usarlos.
- **items:** En este directorio se encuentra el fichero smartlab.items. Es el fichero donde se configuran los items.

- **persistence** Todo lo que tiene que ver con el servicio de persistencia se encuentra en este directorio.
- **rules:** En este directorio se encuentra el fichero donde se implementan las reglas (exceptuando las que se implementen con Rule Engine), el fichero es smartlab.rules. Además de este, se generan varios ficheros automáticamente con el nombre smartlab.rules.save.x. La finalidad de generar estos ficheros es obtener copias de seguridad locales cuando se realiza alguna modificación importante en el fichero smartlab.rules.
- **scripts:** scripts necesarios que haya que implementar para realizar alguna tarea más específica se encuentran aquí.
- **services:** Todos los ficheros de configuración de los servicios que instalemos en OpenHAB, por medio de acciones, se encuentran en este directorio.
- **sitemaps:** En este directorio tenemos el fichero smartlab.sitemap. En este fichero se programan los sitemaps.
- **sounds:** Los sonidos que sean necesarios para reproducirse se encuentran en este directorio con el formato MP3, obligatoriamente.
- **things:** En este directorio se encuentra el fichero smartlab.things, con el que se pueden declarar things.
- **transform:** Aquí se encuentran las traducciones de todo el servicio web de OpenHAB.

Una vez se han conocido los aspectos mas importantes relacionados con la implementación inicial y los que serán importantes para realizar las implementaciones correspondientes a las funcionalidades que se pretenden desarrollar, continuamos viendo cada una de esas implementaciones y la necesidad que cubren.

4.4.3. Conexiones a otras plataformas externas

Uno de los pilares en los que se fundamenta el Smart Lab del CEATIC es la colaboración con otras entidades o instituciones, así como la participación en proyectos donde aspectos como el poder compartir información o la posibilidad de generar datos conjuntos con otros Smart Labs, cobran gran importancia. Es por ello que con la plataforma a desplegar se tenga en cuenta que habrá que interactuar con otras plataformas, desarrolladas y gestionadas por terceros. Con lo cual, una necesidad específica enmarcada dentro del contexto del Smart Lab en el que se va a realizar el despliegue es que los datos tienen que poder ser expuestos al exterior, de manera que, síncronamente a la producción de un evento nuestra plataforma sea capaz de replicar ese cambio a otras plataformas. Estas plataformas, concretamente son Sensor Central, revisada en la Sección 3.2.2.1, y el Sistema Web para la Monitorización de Ambientes Inteligentes, revisado en la Sección 3.2.2.2.

Como se pudo certificar en dicha revisión, estas plataformas utilizan servicios REST para su comunicación por lo que una vez identificada esta necesidad, implícitamente observamos que la configuración de la plataforma IoT que se va a desplegar debe estar contemplada la comunicación vía servicios REST.

Esta conexión con otras plataformas, se realiza a través de una regla programada mediante un script que podemos observar en la Figura 4.11. Los detalles de la implementación de esta regla son los que se van a detallar a continuación.

Esta regla, siguiendo las especificaciones de la junta de dirección del CEATIC, se ha de diseñar con el objeto de compartir los estados de los dispositivos con sensores ambientales, por tanto, la regla se dispara cuando algún miembro del grupo de dispositivos con sensores ambientales que queremos almacenar cambia de estado, como podemos ver en la Figura 4.12.

El funcionamiento de esta implementación, se basa en ordenar por último cambio los miembros del grupo deseado y recupera el último que ha cambiado,

Figura 4.11: Conexión con plataformas externas

```
var lastItem = SensorContact?.members.sortBy[lastUpdate].last
```

Figura 4.12: Cambio de estado en dispositivo

que es el que se guarda en la variable.

Una vez se tiene instanciado el dispositivo con sensores ambientales que ha disparado la regla, se almacenan sus datos en variables. Además, se controla la posibilidad de realizar o no el envío mediante otra variable, booleana, en el inicio instanciada a “false”, y con la que controlaremos que el servicio donde queremos compartir los datos del dispositivo con sensores ambientales que ha disparado la regla. Estas variables se pueden observar en la Figura 4.13.

```
var itemState
var send = false
```

Figura 4.13: Envío de datos

Debido a que el estado que devuelven los dispositivos con sensores ambientales, concretamente los de tipo contacto, vistos en la Sección 3.3.1, es una cadena de caracteres que puede contener el valor “open” o “closed”; por tanto, a continuación

debemos preparar el cambio para las plataformas que tengan instanciados los datos binarios a “0” y “1”, en este caso. Para ello, se realizan las dos condiciones que podemos observar en la Figura 4.14.

```
if (lastItem.state.toString == "OPEN"){
        itemState = '1'
        send = true

        if (lastItem.name.toString == "contact14" || lastItem.name.toString == "contact18"){
                itemState = '0'
        }
}
if (lastItem.state.toString == "CLOSED"){
        itemState = '0'
        send = true

        if (lastItem.name.toString == "contact14" || lastItem.name.toString == "contact18"){
                itemState = '1'
        }
}
```

Figura 4.14: Persistencia con condicionales

El mecanismo para subir el evento a la plataforma indicada es mediante una petición “POST”. Para ello, se guarda la “URL” de destino en una variable. Como las plataformas a las que se envía esta información son diferentes, tenemos una instancia de la “URL” de cada una de ellas para poder realizar el envío a todas. Además, si en el futuro la junta de dirección del Smart Lab del CEATIC decide compartir información con alguna otra plataforma externa, solo habrá que instanciarla con su “URL” en esta regla.

Una vez se tienen los datos necesarios para realizar el envío y su destino, se pasa a construir la petición POST con la que se realizará el mismo. Esta petición se construye con los siguientes datos:

- idEvent: contiene el id del evento que se ha producido.
- idSensor: contiene el id del dispositivo con sensores ambientales que ha disparado el evento.
- property: la propiedad es el tipo de información que se va a almacenar.

- value: contiene el valor que se ha recibido.
- timestamp: se compone de una marca de tiempo, en la que se generó el evento.

Por último, como se puede observar en la Figura 4.15, con la función sendHttpPostRequest() se realiza el envío de la petición.

```
if( send ){
 var String MY_URL = "http://sinbad2.ujaen.es:8069/SmartLab-1.0/services/events/environment/3"
 var String myData =
       '{"idEvent":"0","idSensor":"'+lastItem.name.toString+'","property":"Open","value":"'+itemState+'","timestamp":"0"}'
 logInfo("POST",sendHttpPostRequest(MY_URL, "application/json", myData))
 publish("brokerExterno", "contact/"+lastItem.name.toString, itemState)
}
```

Figura 4.15: Envío de la petición POST

Mediante el mecanismo que se ha implementado, como hemos dicho, se pueden guardar los datos de los dispositivos en cualquier plataforma que permita como conexión los servicios REST. En el Smart Lab se ha implementado para que realice envíos a la plataforma Sensor Central y a la plataforma Sistema Web de Monitorización de Ambientes Inteligentes así como a los servidores de base de datos propios del centro. Es por eso que con la implementación de esta funcionalidad se resuelven las necesidades específicas número 1 y 7.

4.4.4. Gestión de persistencia

La persistencia que se ha implementado con la plataforma OpenHAB está basada en el sistema gestor de base de datos MySQL. Por su facilidad de integración, seguridad y posibilidad de adaptarse a cambios o interactuar con otros servicios se ha utilizado MySQ,L ya que es una de las base de datos que más se utilizan actualmente.

Por motivos de seguridad, la junta de dirección del Smart Lab decidió separar la base de datos de la propia máquina donde se ha implantado OpenHAB. De este modo, la integridad total de los datos está asegurada. Para ello, el servicio de

persistencia ha sido desplegado en una máquina virtual ubicada en los servidores que posee el CEATIC utilizando MySQL Server 5.5.55.

Como podemos observar en la Figura 4.16, para configurar el servicio de persistencia hay que hacerlo a través de un script. La configuración de este fichero se nutre de los datos proporcionados en el acceso.

```
# the database url like 'jdbc:mysql://<host>:<port>/<database>' (without quotes)
url=jdbc:mysql://150.214.174.25:8062/openhab
# the database user
user=openhab

# the database password
password=openhab109

# the reconnection counter
#reconnectCnt=

# the connection timeout (in seconds)
#waitTimeout=

# Use MySQL Server time to store item values (=false) or use openHAB Server time (=true).
# For new installations, its recommend to set "localtime=true".
# (optional, defaults to false)
localtime=true
```

Figura 4.16: Configuración MySQL

OpenHAB sigue una serie de estrategias a la hora de persistir cada uno de los datos. Estas estrategias definen la frecuencia con la que se persisten los datos, como se puede observar en la Figura 4.17. Se puede establecer el fichero que tiene habilitado para ello.

```
Strategies {
        everyMinute     : "0 * * * * ?"
        every5Minutes : "0 */5 * * * ?"
        everyHour    : "0 0 * * * ?"
        everyDay     : "0 0 0 * * ?"
        default = everyChange
}
```

Figura 4.17: Estrategias de persistencia

Los tipos de estrategias para persistir datos que se presentan son las siguientes:

- **Basada en tiempo:** las estrategias basadas en tiempo están establecidas mediante una expresión Cron, compuesta por cinco opciones, las cuales re-

presentan la fecha en la que se quiere hacer la estrategia. En la Figura 4.18 se puede observar con detalle cada una de las opciones.

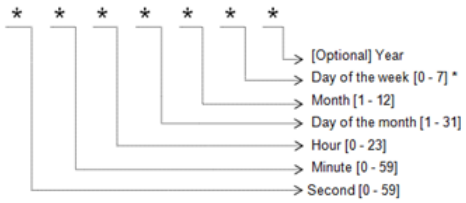

Figura 4.18: Configuración horaria Cron

- **Basada en cambios de estado:** la estrategia basada en cambios de estado es la que se ha escogido por defecto, como podemos observar en la Figura 4.17. Su funcionamiento es simple, cada vez que un dispositivo cambia de estado, este se persiste.

La forma en la que se aplican estas estrategias es especificando qué estrategia sigue cada item o grupo de items, como podemos observar en la Figura 4.19.

```
Items {
    * : strategy = everyChange, everyDay, restoreOnStartup
    Puerta : strategy = everyChange, restoreOnStartup
    IluminacionI : strategy = everyChange, restoreOnStartup
    Entrada* : strategy = everyChange, restoreOnStartup
    SensorContact* : strategy = everyChange, restoreOnStartup
}
```

Figura 4.19: Estrategia en items

La persistencia utilizada tiene una estructura separada en tablas por cada uno de los items que existen en la plataforma. Se trata de una base de datos indexada, que contiene una tabla de índices con los que se identifican los items. Para facilitar futuras consultas, se inserta en una tabla la descripción de cada uno de los items, obteniendo la siguiente estructura:

- **Tabla Items:** Contiene la descripción de cada uno de los items, se trata de la tabla de índices que anteriormente se ha comentado como se puede observar en la Figura 4.20.

ItemId	ItemName
1	Movimiento Bano
2	Contacto Telefono
3	Movimiento Dormitorio
4	Movimiento Cama
5	Movimiento Cocina

Figura 4.20: Estructura en base de datos de items

- **Tabla Items[N]:** Contiene los valores guardados de un item concreto. "N" será un valor incremental correspondiente al momento de guardar los datos, como podemos observar en la Figura 4.21.

Time	Value
2017-10-24 12:16:22	closed
2017-10-24 12:20:02	open
2017-10-24 12:20:10	closed
2017-10-24 12:20:21	open
2017-10-24 12:32:02	closed

Figura 4.21: Ejemplo de persistencia en tabla Items[N]

Una vez conocida la estructura, a continuación se describen los tres procesos de consulta de los datos que han sido guardados. Cada una de estas tres opciones está enfocada a un uso y su implementación ha sido necesaria para resolver cuestiones de uso. En cada una de las opciones que se describen a continuación se especifica también la necesidad de su implementación.

- Cliente Externo. Con fines de mantenimiento de los datos y de detección de inconsistencias o errores en los mismos, ya que una de las necesidades específicas que se han marcado es que los datos deben estar seguros y en consistencia. Se consultan los datos a través del cliente propio de MySQL, denominado MySQL Workbench.

- Reglas de OpenHAB. Desde las reglas OpenHAB se pueden realizar consultas simples a la base de datos. Esta opción se utiliza para consultas simples y que su resultado sea claro. Un mecanismo de consulta intuitivo y de fácil

acceso pero que presenta problemas de funcionalidad ante consultas complejas.

- Script desarrollado. Como se comenta en el punto anterior, una de las limitaciones de las consultas predefinidas en las reglas de la plataforma es la gran limitación a la hora de realizar consultas más complejas. Por este motivo, se ha creado un script en Unix Bash, el cual es llamado con una cadena de texto correspondiente a cualquier tipo de consulta. Como se puede observar en la Figura 4.22, al editarlo solo tenemos la configuración de la base de datos donde se realizaría la conexión y la consulta, la cual recibe como parámetro.

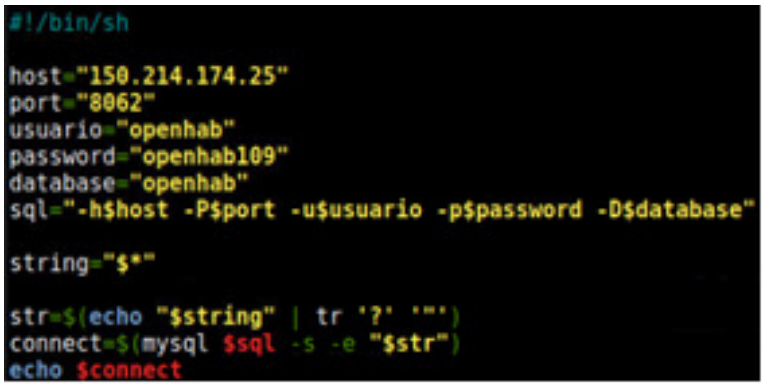

Figura 4.22: Script de consultas SQL

Recapitulando, en esta sección hemos podido obtener una visión en profundidad de los puntos importantes en el despliegue que se ha realizado de la plataforma OpenHAB en el Smart Lab del CEATIC, atendiendo a las necesidades que se han marcado. Pero estas necesidades, como hemos podido comprobar, no están del todo cubiertas, faltan algunos puntos importantes a desarrollar para cubrir el abanico que se desea. Será en la siguiente sección donde se abordará el desarrollo de las características necesarias en la plataforma para cumplir con todas las necesidades que se han propuesto.

4.5. Funcionalidad adicional

Debido a que, como hemos visto en la sección anterior, la instalación de componentes de la plataforma no basta para cubrir las necesidades requeridas. En esta sección se van a buscar las soluciones óptimas para poder llevar a cabo todas las necesidades y cumplir con los objetivos del presente trabajo fin de grado.

En las siguientes secciones se detallan los aspectos mas interesantes de cada una de las funcionalidades adicionales que se han implementado, así como detalles relevantes a su desarrollo.

4.5.1. Recordatorio de medicación

Como se ha comentado en la Sección 2.4.1 el Smart Lab del CEATIC está enfocado a la propuesta de soluciones para mejorar la calidad de vida de las personas. En este contexto, nacen ideas como la que aquí se implementa. Esta funcionalidad se basa en que la plataforma sea capaz de recordar, mediante los dispositivos actuadores, al habitante del Smart Lab que tiene que tomar su medicación porque se le ha olvidado.

Los dispositivos que intervienen en esta regla son los siguientes:

- Pastillero: es el dispositivo hardware clave en el desarrollo de esta funcionalidad. Se trata de un dispositivo capaz de realizar descargas de pastillas en su bandeja automáticamente.

- Dispositivos actuadores: los dispositivos actuadores que intervienen aquí son los altavoces, mediante los cuales, se realiza el recordatorio de la toma olvidada.

La implementación de esta funcionalidad en la plataforma se realiza mediante dos reglas que podemos observar en la Figura 4.23, que se encargan de controlar la medicación. Su funcionamiento es el que se describe a continuación.

La primera regla controla la ventana de tiempo definida mientras que la segunda se encarga de controlar los dispositivos actuadores. Una vez que se entre en la ventana de tiempo establecida, si la pastilla no se ha tomado, se envía la señal de recordatorio por medio de los dispositivos actuadores.

```
rule "Tomar Medicacion"
when
        Item contact08 received update OPEN
then
        if(now.getHourOfDay==10){
                pasti=true
        }
end

rule "Tomar Medicacion Recordatorio"
when
        Time cron "0 5 11 1/1 * ? *"
then
        if(!pasti){
                say("Recordatorio Medicación")
        }
end
```

Figura 4.23: Recordatorio medicación

Con estas reglas, obtenemos la funcionalidad requerida y así poder facilitar el acceso a la medicación, controlando, que esta se tome en base a las indicaciones óptimas.

4.5.2. Definición de escenas en el Smart Lab

En concordancia con los requerimientos demandados en la necesidad específica número 5, se ha decidido introducir en la interfaz de visualización del estado de dispositivos y el control de ellos, algún mecanismo de adaptación del ambiente en función de las necesidades de sus habitantes. Todo de una manera muy visual y cómodamente manejable para cualquiera.

Por este motivo, se han decidido definir cuatro escenas con las que se pueda adaptar el ambiente, de manera que en esta adaptación sólo intervengan disposi-

tivos actuadores, que se explican a continuación.

- **Bienvenida:** Es una de las escenas más simple, consiste en encender la iluminación del la entrada de la vivienda y reproducir un mensaje de bienvenida. Para la iluminación se ha utilizado colores claros y un mensaje estático "You are welcome", el cual se encuentra en dos idiomas (inglés y español), los cuales pueden ser configurados manualmente por el administrador de OpenHAB al editar la definición de la regla.

  Una de las grandes ventajas que nos proporciona esta escena es que puede ser disparada automáticamente y, de una manera muy simple, cuando un habitante entre a la casa.

- **Relajarse:** Al establecer esta escena, la iluminación general será atenuada casi a un nivel mínimo, tras llevar a cabo esta atenuación, OpenHAB reproducirá una canción tranquila y efectos de sonidos tranquilizantes como agua, brisa, entre otros. Los colores de esta iluminación han sido establecidos en turquesa y blanco.

- **Lectura:** Esta escena establece una iluminación general atenuada, por lo que en el sitio donde el habitante desee llevar a cabo la actividad deberá de establecer manualmente unos valores más elevados. En cuanto al actuador de sonido, esta escena reproduce una melodía continua, la cual brinda al habitante un alto nivel de concentración y relajación.

- **Fiesta:** Para llevar a cabo esta escena, se han utilizado tanto los actuadores de iluminación como sonoros, para ello, la iluminación irá alternando su color aleatoriamente cada 5 segundos. En cuanto al audio, se eleva por encima de su capacidad y se reproduce una canción motivadora para llevar a cabo la actividad.

Para llevar a cabo esta tarea, se ha implementado una regla, la cual combina

diferentes configuraciones en los principales actuadores de la vivienda (iluminación y sonido), por lo que como se menciona en el inicio de esta definición, la principal funcionalidad es adaptar el laboratorio al estado más adecuado para la actividad que desee realizar un habitante.

Para este fin, se ha añadido un nuevo botón en el sitemap, es decir, hay un item llamado SceneButton y que está reflejado en el sitemap:

Figura 4.24: SceneButton en el sitemap

Este botón otorga un valor numérico a cada opción (0-Relajarse, 1-Lectura, 2-Fiesta, 4-Bienvenida). La lógica de la regla es la que se describe a continuación. El disparador de la regla es el cambio de estado del botón SceneButton. Una vez se activa el disparador, ya en la parte "THEN" se recupera el valor del botón. Posteriormente, se envían dos comandos, uno a los altavoces con una canción y otro a las luces con un color. Y por ultimo, se hace el cambio en base al valor recuperado del SceneButton, como se puede observar en la Figura 4.25.

```
if (stationIndex == 0) {
        //ESCENA RELAJARSE
        fiesta = false
        sendCommand(General_Color, "60,7,60")
        sendCommand(Sonos_Dormitorio, "http://sinbad2.ujaen.es:8068/avisos/musica/relajarse.mp3")
}
```

Figura 4.25: SceneButton como disparador

4.5.3. Smart Lab Social. Twitter

En esta sección se van a mostrar las reglas que publican información sobre el Smart Lab en la cuenta de Twitter: "@UJAmI_CEATIC".

A continuación, se indican los tuits que han sido programados.

- **Tomas de Temperatura:** Uno de los aspectos que se quieren publicar, bajo el contexto de Smart Lab Social, es la temperatura a la que se encuentran los habitantes de este ambiente inteligente. Con esa idea, se pretende implementar una regla que publique, de una forma automatizada y cada cierto periodo de tiempo, la temperatura recogida por los sensores en la cuenta de Twitter.

```
rule "Twitter_Temp"
when
        Time cron "0 0 0/4 ? * * *"
        //Time cron "0 0/1 * 1/1 * ? *"
then
        var tmp=Temperatura_Dormitorio.state.toString
        logInfo("Upload",tmp)
        sendTweet("The temperature in the UJAmI Lab is "+tmp+" degrees ↓")
end
```

Figura 4.26: Tomas de temperatura

Como podemos observar en la Figura 4.26, la lógica de esta regla es simple, se define como disparador una tarea Cron de cuatro horas. Una vez salta, se toma la temperatura en el dispositivo con sensores ambientales ubicado en el dormitorio del Smart Lab y se guarda en una variable para posteriormente pasarla en la función sendTweet() para publicar la información.

- **Resumen medias temperatura:** Otro de los intereses con respecto a la publicación de información en redes sociales que tiene que ver con la temperatura. Se trata de ofrecer diariamente un informe que contenga la temperatura máxima, la mínima, así como la media de temperatura que ha habido durante ese día en el Smart Lab.

  Para hacerlo posible, se ha implementado la regla que podemos observar en la Figura 4.27 y que a continuación se detalla.

  El disparador de esta regla es de nuevo el tiempo, por lo que se gestiona mediante una tarea Cron definida a la hora establecida según el criterio

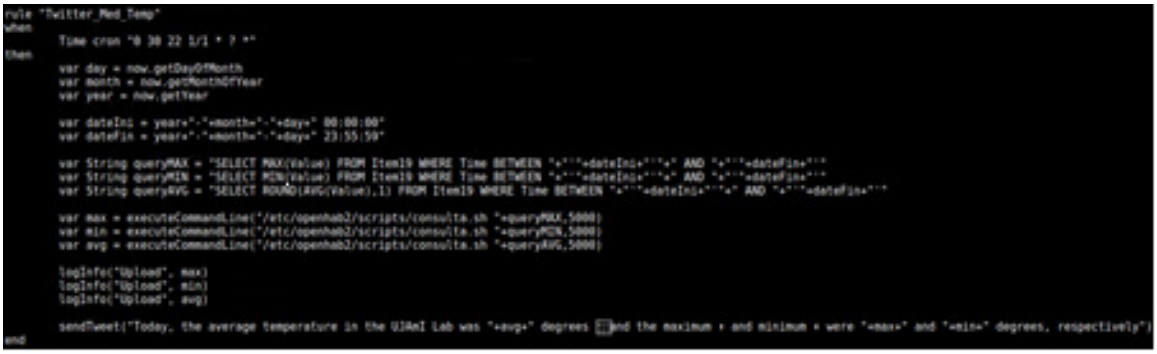

Figura 4.27: Medias de temperatura

de la junta de dirección del CEATIC. Lo primero que se realiza en esta implementación es obtener una fecha inicio y fecha fin. Ambas serán el día actual, con la hora de inicio a las 00:00 y la hora fin 23:55:59. Una vez definida fecha inicio y fecha fin se hacen tres consultas, las tres devuelven el "Value" de la tabla del item correspondiente que se encuentra entre fecha inicio y fecha fin. Las consultas se detallan a continuación.

El máximo se obtiene incluyendo en la consulta la función "MAX(Value)", que nos devolverá el máximo valor almacenado en la tabla de ese item comprendido entre la fecha inicio y la de fecha fin. El mínimo, se realiza incluyendo en la consulta la función "MIN(Value)" que retornará una entrada comprendida entre el intervalo definido con las fechas con el mínimo valor. Para realizar la media de las temperaturas, se sigue el mismo funcionamiento, utilizar las funciones de consulta SQL que brinda MySQL para realizar operaciones. En este caso, la función es "AVG(Value)" y devuelve la media aritmética con un decimal obtenido por truncamiento de todos los valores almacenados comprendidos entre el inicio y el fin preestablecidos.

Una vez obtenidos y almacenados en tres variables (max, min y avg), los tres valores deseados, los publicamos por medio de la función sendTweet(), como se detalla en la Figura 4.27.

- **Resumen interacciones puerta:** La información que se quiere publicar en este punto es un resumen de las veces que se ha abierto la puerta del Smart Lab en el día.

  Para ello, se ha implementado la regla que podemos observar en la Figura 4.28 y que seguidamente se describe.

```
rule "Twitter_nPuerta"
when
        Time cron "0 0 22 ? * MON-FRI *"
        //Time cron "0 0 1 ? * MON-FRI *"
        //Time cron "0 0/1 * 1/1 * ? *"
then
        var day = now.getDayOfMonth
        var month = now.getMonthOfYear
        var year = now.getYear

        var dateIni = year+"-"+month+"-"+day+" 00:00:00"
        var dateFin = year+"-"+month+"-"+day+" 23:55:59"

        var String query = "SELECT COUNT(Value) FROM Item32 WHERE Time BETWEEN "+"'"+dateIni+"'"+" AND "+"'"+dateFin+"'"

        var nPuerta = executeCommandLine("/etc/openhab2/scripts/consulta.sh "+query,5000)
        var res = Integer::parseInt(nPuerta)/2
        logInfo("Upload",res.toString)

        if(res != 0){
                sendTweet("Today, the door of UJAmI Lab was opened "+res.toString+" times.#")
        }
end
```

Figura 4.28: Resumen puerta

Como el resumen que se quiere presentar es diario y a una hora preestablecida, el disparador de la regla será una tarea Cron, como en el punto anterior, los datos que se van a representar se obtienen realizando una consulta a la base de datos, por tanto, también necesitaremos una ventana de tiempo que indique el día seleccionado. Para ello, se crea una fecha inicio y una fecha fin. Realizada la consulta y una vez tenemos los valores, estos se dividen entre dos, porque recordemos que los valores que retorna la base de datos son de interacciones, es decir, tanto del estado abierto como del estado cerrado, para obtener el número de veces que se ha abierto la puerta al día.

- **Saludo Primer Visitante:** con esta publicación, lo que se realiza es un saludo diario por parte de la cuenta de Twitter del Smart Lab del CEATIC, al primer visitante que entre. Con el fin de ofrecer una imagen simpática y divertida de la cuenta de Twitter, se ponen en práctica este tipo de acciones,

ya que con esta implementación, el Smart Lab, saluda a su primer visitante todos los días.

Para implementar esta publicación se ha creado la regla que podemos apreciar en la Figura 4.29 y que se detalla en las siguientes líneas.

```
rule "Twitter_primPuerta"
when
        Item Contacto_Puerta_Principal changed to open
then
        //var String cosa = "ENTRO"
        //logInfo("Upload",cosa)

        var day = now.getDayOfMonth
        var month = now.getMonthOfYear
        var year = now.getYear

        var dateIni = year+"-"+month+"-"+day+" 00:00:00"
        var dateFin = year+"-"+month+"-"+day+" 23:55:59"

        var String query = "SELECT COUNT(Value) FROM Item32 WHERE Time BETWEEN "+"'"+dateIni+"'"+" AND "+"'"+dateFin+"'"

        var nPuerta = executeCommandLine("/etc/openhab2/scripts/consulta.sh "+query,5000)
        var res = Integer::parseInt(nPuerta)

        var String str = null

        if(res == 1){
                if(now.getHourOfDay >= 5 && now.getHourOfDay < 12){
                        str = "Morning"
                }
                if(now.getHourOfDay >= 12 && now.getHourOfDay < 18){
                        str = "Afternoon"
                }
                if(now.getHourOfDay >= 18 && now.getHourOfDay < 5){
                        str = "Evening"
                }

                sendTweet("Good "+str+", welcome to our first visitor today in the UJAmI Lab. We hope you enjoy! [illegible]")
        }
end
```

Figura 4.29: Saludo primer visitante

El disparador que genera la activación de esta regla es el dispositivo con sensores ambientales de contacto instalado en la puerta del Smart Lab. Cada vez que esta se abre, guarda la ventana inicio/fin del día actual como en las implementaciones anteriores, consultando en la base de datos el número de valores que hay en la tabla de la puerta y que estén dentro de la ventana inicio/fin.

Posteriormente se comprueba si el resultado es 1 (si es 1 significa que es la primera vez que se abre la puerta en ese día, por tanto, habrá que realizar el saludo). Como particularidad, tenemos que el saludo se realiza de formas diferentes según la hora del día en la que se produzca, en la publi-

cación aparecerá buenos días/tardes/noches, en función de la hora que se reciba al primer visitante. El ajuste de esta palabra se realiza al final de la implementación, como podemos observar en la Figura 4.29.

- **Detección de Actividad Sofá:** en este apartado, lo que se quiere conseguir es que la plataforma notifique en Twitter cada vez que alguien utiliza el sofá. Como en los apartados anteriores, para desarrollar esto, nos servimos del mecanismo que la plataforma nos ofrece para automatizar procesos, las reglas. La regla que se ha programado para llevar a cabo esta publicación en Twitter tiene un funcionamiento muy básico, como podemos observar en la Figura 4.30; su disparador es el dispositivo con sensores ambientales instalado en el sofá del Smart Lab.

```
rule "Twitter_Sofa"
when
        Item PresionSofa_DoorWindowStatus changed to OPEN
then
        sendTweet("An occupant is relaxing on the sofa. Relax!
end
```

Figura 4.30: Actividad en sofá

Como hemos comentado anteriormente, el disparador se encuentra instalado en el sofá; la regla se dispara cuando este dispositivo presenta un cambio de estado de "cerrado" a "abierto". Una vez se detecta el cambio de estado y la regla se ha disparado, lo que hace es publicar en la cuenta de Twitter mediante la función sendTweet() el mensaje correspondiente a que hay alguien relajado en el sofá.

- **Detección de Actividad Basura:** en esta implementación lo que se ha requerido es que el Smart Lab notifique a través de Twitter que alguien esta sacando la basura. Para realizarla se desarrolla la regla que podemos observar en la Figura 4.31. En esta regla entran en juego varios dispositivos,

no solo el que la dispara, puesto que la forma de detectar que alguien está tirando la basura es que la puerta del Smart Lab se abra como mucho tres minutos después de que el estado del dispositivo instalado en el cubo de basura reciba una actualización de estado a "abierto".

```
rule "Twitter_Basura"
when
        Item contact08 received update OPEN
then
        var day = now.getDayOfMonth
        var month = now.getMonthOfYear
        var year = now.getYear

        var dateIni = year+"-"+month+"-"+day+" 00:00:00"
        var dateFin = year+"-"+month+"-"+day+" 23:55:59"

        var String query = "SELECT COUNT(Value) FROM Item88 WHERE Time BETWEEN "+"'"+dateIni+"'"+" AND "+"'"+dateFin+"'"

        var nCubo = executeCommandLine("/etc/openhab2/scripts/consulta.sh "+query,5000)
        var res = Integer::parseInt(nCubo)

        Thread::sleep(3 * 60 * 1000)

        var String query2 = "SELECT COUNT(Value) FROM Item88 WHERE Time BETWEEN "+"'"+dateIni+"'"+" AND "+"'"+dateFin+"'"

        var nCubo2 = executeCommandLine("/etc/openhab2/scripts/consulta.sh "+query2,5000)
        var res2 = Integer::parseInt(nCubo2)

        if(res!=res2){
                sendTweet("An occupant has gone out to throw the garbage 🚮")
        }
end
```

Figura 4.31: Actividad basura

El disparador de esta regla es el dispositivo con sensores ambientales de contacto instalado en el cubo de la basura, recibiendo una actualización de estado de "cerrado" a "abierto". La lógica de la regla sigue los pasos que a continuación se describen. Primero, se crea una ventana horaria, con fecha inicio y fecha fin, ambas son el día actual, el inicio a las 00:00 y el fin a las 23:55:59. Una vez tenemos la ventana definida, realizamos la consulta, con la que obtenemos el número de veces que se ha abierto la puerta. Después de realizar la consulta, por medio de la hebra de ejecución del proceso, se detiene la misma y se dejan pasar tres minutos; una vez transcurrido el tiempo definido para tomar que la actividad se está llevando a cabo, se vuelve a realizar la consulta. Si los resultados obtenidos antes y después de la detección son diferentes, sabemos que la puerta se ha abierto en esa ventana de tres minutos, con lo que podemos publicar que alguien está tirando la

basura.

- **Detección de Actividad Tomar Medicación:** siendo la detección de actividades una de las áreas de conocimiento donde se centra el Smart Lab y, además, con un enfoque orientado a facilitar el uso de la vivienda a personas mayores; este apartado cobra gran importancia dentro del contexto de Smart Lab Social, puesto que la plataforma será capaz de notificar las tomas de medicación.

  La implementación de esta tarea se lleva a cabo mediante el desarrollo de una regla que permita detectar el uso de los dispositivos que tienen protagonismo en esta actividad y que a continuación se reseñan.

  Para la actividad de "tomar la medicación" entran en juego tres dispositivos con sensores ambientales como son: el armario de tazas, sensor en pastillero y por último, el sensor de contacto en la botella de agua. Además de que en la tarea intervengan tres dispositivos diferentes, tienen que hacerlo en una ventana de tiempo predefinida, por lo que será otro aspecto a tener en cuenta en el desarrollo de la regla.

  El disparador de esta regla, cuya implementación podemos ver en la Figura 4.32 puede ser cualquiera de los anteriormente mencionados, ya que la actividad no tiene por qué seguir un orden. Una vez definidos los disparadores se guarda la fecha inicio, que será la fecha y hora actuales, y tres minutos anterior a que se dispare la regla. La fecha fin la ajustamos como la actual, es decir, el momento en el que se dispara.

  Una vez tenemos la fecha inicio y fecha fin definidas, realizamos tres consultas, una para cada uno de los items que intervienen en la regla. El resultado de estas será el valor del item comprendido entre fecha inicio y fecha fin, es decir, vemos si en la ventana de tres minutos los tres sensores han cambiado de estado.

```
rule "Twitter_Medicacion"
when
        //Time cron "0 0/1 * 1/1 * ? *"
        Item contact05 received update OPEN or
        Item contact01 received update OPEN or
        Item Contacto_Armario_Tazas received update open
then
        var day = now.getDayOfMonth
        var month = now.getMonthOfYear
        var year = now.getYear
        var hour = now.getHourOfDay
        var minute = now.getMinuteOfHour - 3
        var second = now.getSecondOfMinute

        var dateIni = year+"-"+month+"-"+day+" "+hour+":0"+minute+":00"

        minute = minute + 3

        var dateFin = year+"-"+month+"-"+day+" "+hour+":0"+minute+":0"+second

        var String query1 = "SELECT Value FROM Item107 WHERE Time BETWEEN "+dateIni+" AND "+dateFin
        var String query2 = "SELECT Value FROM Item92 WHERE Time BETWEEN "+dateIni+" AND "+dateFin
        var String query3 = "SELECT Value FROM Item40 WHERE Time BETWEEN "+dateIni+" AND "+dateFin

        var res1 = executeCommandLine("/etc/openhab2/scripts/consulta.sh "+query1,5000)
        var res2 = executeCommandLine("/etc/openhab2/script$/consulta.sh "+query2,5000)
        var res3 = executeCommandLine("/etc/openhab2/scripts/consulta.sh "+query3,5000)

        if(res1 != "" && res2 != "" && res3 != ""){
                sendTweet("An occupant is taking a medication in the UJAmI Lab [illegible]")
        }
```

Figura 4.32: Actividad medicación

Por último la condición para publicar es que ninguno de los tres resultados de las consultas contenga un valor nulo, ya que de ser así significará que no se ha interactuado con ese sensor, por tanto, no se puede dar que se esté realizando la actividad.

- **Temporizar Tiempo Luces Encendidas:** el objetivo de este apartado es que se pueda obtener de manera visual y a través de la cuenta de Twitter el tiempo que han estado activas las distintas luces distribuidas por las estancias del Smart Lab.

  El procedimiento se realiza mediante la implementación de la regla que podemos observar en la Figura 4.33 y en la Figura 4.34.

  En esta regla, el disparador es el cambio de estado de "ON" a "OFF" de cualquiera de los items enlazados a los dispositivos actuadores de iluminación. La lógica reside en que si el estado ha cambiado a "OFF" es porque

```
rule "Twitter_Luz_Salon"
when
        Item Salon_Iluminacion received update OFF
then
        var String query1 = "SELECT Time FROM Item62 WHERE Value = ?on? ORDER BY Time DESC LIMIT 0,1"
        var String query2 = "SELECT Time FROM Item62 WHERE Value = ?off? ORDER BY Time DESC LIMIT 0,1"

        var res1 = executeCommandLine("/etc/openhab2/scripts/consulta.sh "+query1,5000)
        var res2 = executeCommandLine("/etc/openhab2/scripts/consulta.sh "+query2,5000)

        logInfo("Upload",res1)
        logInfo("Upload",res2)

        val time1 = res1.split(" ")
        val time2 = res2.split(" ")

        logInfo("Upload",time1.get(1))
        logInfo("Upload",time2.get(1))

        val hhmmss1 = time1.get(1).split(":")
        val hhmmss2 = time2.get(1).split(":")

        val min1str = hhmmss1.get(1).toString()
        val min2str = hhmmss2.get(1).toString()
```

Figura 4.33: Tiempo luces 1

```
val seg1str = hhmmss1.get(2).toString()
val seg2str = hhmmss2.get(2).toString()

val min1 = Integer::parseInt(min1str)
val min2 = Integer::parseInt(min2str)

val seg1 = Integer::parseInt(seg1str)
val seg2 = Integer::parseInt(seg2str)

logInfo("Upload",min1.toString())
logInfo("Upload",min2.toString())

logInfo("Upload",seg1.toString())
logInfo("Upload",seg2.toString())

if(hhmmss1.get(0) == hhmmss2.get(0)){
        var minutes = min2 - min1
        var seconds = seg2 - seg1
        sendTweet("The living room light has been on for "+minutes+":"+seconds+" minutes. ")
}
```

Figura 4.34: Tiempo luces 2

ha estado encendida y por tanto, hay que contabilizar el tiempo que lo ha estado para poder publicarla.

Lo primero que se hace en la regla es recuperar de la base de datos el último valor “OFF” y el último valor “ON” del dispositivo que haya actuado como disparador. Mediante los métodos split() y parseInt() se obtienen los valores necesarios del campo “timestamp” para realizar el cálculo del tiempo que ha

estado el dispositivo activo. Los cálculos que se realizan son los siguientes:

minutosTimestampOFF - minutosTimestampON = minutos de dispositivo de iluminación activo segundosTimestampOFF - segundosTimestampON = segundos de dispositivo de iluminación activo

Una vez obtenidos los valores deseados, se realiza la publicación de los resultados con su mensaje correspondiente en Twitter mediante la función sendTweet().

Figura 4.35: Tuits

De este modo, se han creado todas las reglas asociados a los tuits que habían sido predefinidos en la cuenta de Twitter: "@UJAmI_CEATIC". En la Figura 4.35 se muestra una imagen de la cuenta de Twitter donde se ilustran algunos de los tuits que han sido programados.

Capítulo 5

Conclusiones y línea de trabajo futuras

El Smart Lab del CEATIC es un Smart Lab académico enfocado a la inteligencia ambiental que nació en 2015 con la especial vocación de investigar y desarrollar propuestas y aplicaciones que mejoren la calidad de vida de las personas.

Desde la creación del Smart Lab, se desarrolló un sistema web propio de monitorización de ambientes inteligentes donde se fueron integrando algunos de los principales dispositivos del Smart Lab con el objetivo de almacenar y gestionar los datos de los sensores en tiempo real, así como ofrecer funcionalidades relacionadas con el área de investigación de reconocimiento de actividades.

Debido a la proliferación de dispositivos inteligentes y las grandes oportunidades que brindan en el área de inteligencia ambiental, numerosas adquisiciones en términos de dispositivos han sido realizadas en el Smart Lab del CEATIC. Por ello, surge la necesidad de desplegar una plataforma IoT consolidada que permita gestionar los múltiples dispositivos adquiridos y explotar la funcionalidad que esta brinda.

Este trabajo fin de grado se ha elaborado con el fin de satisfacer dicha necesidad, desplegando una plataforma IoT que permite la integración de nuevos

dispositivos heterogéneos de una forma escalable, sencilla y con un coste reducido, así como permitir su evolución ante el nuevo mosaico de tecnologías que seguirán surgiendo sobre dispositivos inteligentes.

Para ello, en primer lugar, en este trabajo fin de grado, se ha realizado una breve descripción de Internet de las Cosas e Inteligencia Ambiental, ya que son los pilares sobre los que se fundamenta el Smart Lab del CEATIC. Además, se han revisado los Smart Lab académicos con los que interacciona el Smart Lab del CEATIC y las principales propuestas y aplicaciones que se desarrollan en él.

A continuación, se realizó un profundo análisis para escoger la plataforma IoT a desplegar en el Smart Lab. El criterio fundamental para escoger la plataforma IoT ha sido seleccionar la plataforma con una mayor compatibilidad de dispositivos. Para ello, ha sido necesario, por un lado, realizar un análisis de las plataformas IoT consolidadas actuales y, por otro lado, una análisis de todos los dispositivos que alberga el Smart Lab del CEATIC. Fruto de este análisis, se elaboró una tabla de compatibilidad donde la plataforma de IoT denominada OpenHAB se posicionaba como la mejor en base al criterio mencionado.

Una vez seleccionada la plataforma IoT de OpenHAB como la plataforma a desplegar, las necesidades específicas del Smart Lab en relación a la plataforma IoT fueron indicadas, y son las que han guiado el despliegue de la plataforma IoT y sus componentes. Además, ha sido propuesto un esquema operativo de la plataforma IoT, los recursos físicos y lógicos que han sido desplegados junto a la gestión de la persistencia que ha sido desarrollada.

En este punto, cabe señalar que la plataforma IoT OpenHAB ofrece numerosas ventajas. La principal, y por la que fue escogida, es que ofrece la posibilidad de despliegue de casi cualquier dispositivo, siendo una solución Open Source y con una gran comunidad de soporte y mantenimiento detrás. Además, el funcionamiento de OpenHAB es modular de tal manera que, prácticamente, cada módulo es una aplicación independiente; es compatible con todos los estándares de comu-

nicación y posee grandes funcionalidades adicionales que no se han encontrado en otras plataformas IoT.

El principal punto débil de OpenHAB, en el contexto del Smart Lab del CEATIC, es que dicha plataforma IoT carece de características y funcionalidades propias de una plataforma IoT centrada en investigación. Para solventar dicha limitación, en el esquema operativo de funcionamiento de la plataforma IoT del Smart Lab del CEATIC, se han definido conexiones a dos plataformas IoT orientadas a investigación. De este modo, la información capturada por el Smart Lab del CEATIC es enviada de forma transparente a dos plataformas de IoT: Sensor Central de la Universidad del Ulster y la plataforma IoT de desarrollo propio de la Universidad de Jaén. De esta forma, las plataformas IoT orientadas a investigación se nutren de los datos de la plataforma IoT basada en OpenHAB, ofreciendo todo su potencial a nivel de investigación.

Las funcionalidades específicas del Smart Lab, las cuales no abarcaban los módulos del OpenHAB, han sido incluidas. Entre ellas, destaca la elaboración de configuraciones de actuadores del Smart Lab, una aplicación para recordar la toma de medicación en el Smart Lab y, finalmente, la creación de reglas para publicar en una cuenta de Twitter el estado del Smart Lab.

Además, el manual de instalación de la plataforma IoT basado en OpenHAB ha sido elaborado junto con el manual de mantenimiento de dicha plataforma.

Como se puede comprobar, la propuesta principal de este trabajo fin de grado ha sido alcanzada con éxito, cubriendo cada uno de los objetivos específicos que se indicaban en el Capítulo 1 de la presente memoria.

Derivado de este trabajo fin de grado, se podrían analizar, diseñar y programar módulos específicos propios para integrar aquellos dispositivos que actualmente no soporta la plataforma OpenHAB, como por ejemplo, el suelo inteligente.

Anexo I. Manual de instalación y mantenimiento

El objetivo de este manual de instalación y mantenimiento es detallar los pasos a seguir en la instalación de la plataforma IoT OpenHAB y sus componentes, así como las cuestiones referentes a su mantenimiento.

El documento se compone de las siguientes secciones:

- **Hardware de conexión. Z-Wave:** esta sección abarca los detalles referidos al hardware necesario para utilizar el protocolo de comunicación Z-Wave en la plataforma IoT.
- **Instalación de OpenHAB:** aquí se muestra el proceso de instalación de OpenHAB.
- **Instalación de Componentes:** se detallan los pasos a seguir para instalar componentes así como para el acceso a ellos.
- **Creación de entidades lógicas. Things, items, grupos:** en esta sección se detallan los pasos a seguir para la creación de todas las entidades lógicas con las que trabaja la plataforma IoT.
- **Acciones. MQTT, Twitter, Persistencia, Voz:** contiene el modo de configuración de las tres acciones que se mencionan.

- **Mantenimiento. Consola de OpenHAB:** debido a la importancia que tiene la consola en la plataforma IoT, en esta sección se detallan los aspectos relevantes de su uso.

Hardware de Conexión

Z-Wave

La parte hardware de la implementaciónón del protocolo Z-Wave en el Smart Lab se realiza mediante un stick USB que podemos observar en la Figura 5.1. Este stick es el que realiza la comunicación con los dispositivos físicos para que desde la plataforma se pueda tener acceso a ellos. El stick, en el Smart Lab del CEATIC, está configurado con la máquina virtual en el puerto situado más a la derecha en el PC, que será donde habrá que conectarlo siempre. En la Figura 5.2 podemos ver como queda el dispositivo una vez se conecta por USB.

Figura 5.1: Stick Z-Wave

Para configurarlo en otro puerto, se seguirán los siguientes pasos:

1. Identificamos el puerto con el comando:

   ls /dev/tty*

2. Una vez tenemos identificado el puerto, vamos a PaperUI → Configuration → Things → Z-Wave Serial Controller, y pinchamos en el lápiz de arriba para editar su configuración.

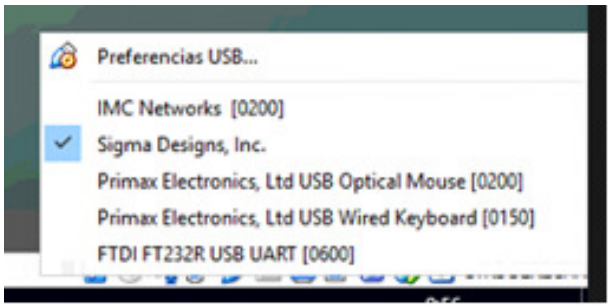

Figura 5.2: Listado de USB

3. Dentro de la configuración tenemos el apartado "Port Configuration" y ahí, en, el apartado "Serial Port", introducimos el nuevo puerto.

Para realizar los emparejamientos con los diferentes dispositivos que utilicen Z-Wave se seguirán los siguientes pasos:

1. Para poner el stick en modo inclusión tendrá que estar desconectado del PC. Una vez desconectado bastará con pulsar el botón de acción hasta que el LED parpadee despacio y de color azul.

2. El segundo paso es con el dispositivo con sensores ambientales o actuador, cada sensor se empareja de un modo diferente y su mecanismo de inclusión estará indicado en el manual de uso del mismo.

A continuación, se indica los pasos para realizar el emparejamiento de un dispositivo con sensores ambientales de contacto Everspring:

- El modo inclusión en estos disapositivos se inicia presionando el botón de la parte de atrás del dispositivo tres veces seguidas, en una ventana de tiempo no mayor de 1'5 segundos.

- Una vez realizado el emparejamiento, el stick lo confirmará con un parpadeo rápido del LED.

- El stick continuará parpadeando en modo inclusión, esperando nuevos emparejamientos.

- Es posible realizar un reset del sensor para borrar el emparejamiento con el stick. Para ello, debemos pulsar el botón del sensor cuatro veces seguidas, manteniendo el botón presionado la última vez hasta que el led se apague.

Instalación de OpenHAB

Para realizar la instalación de OpenHAB, lo primero que se debe hacer es instalar los respositorios necesarios con los comandos que se muestran a continuación. Para realizar la instalacion por medio del protocolo https, debemos generar una clave "bintray" que será introducida en el comando.

wget -qO -
'https://bintray.com/user/downloadSubjectPublicKey?username=openhab' —
sudo apt-key add -
sudo apt-get install apt-transport-https
echo 'deb https://dl.bintray.com/openhab/apt-repo2 stable main' — sudo tee
/etc/apt/sources.list.d/openhab2.list

Una vez añadidos los repositorios, realizamos una actualización del índice de paquetes para que se carguen. Para ello utilizamos el comando:

sudo apt-get update

Para instalar OpenHAB, utilizamos la siguiente orden:

sudo apt-get install openhab2

Y posteriormente instalamos todos los paquetes de OpenHAB adicionales necesarios, utilizando el comando:

sudo apt-get install openhab2

Una vez instalados, se pondrá en marcha el servicio, además se comprobará que su estado es "active (online)". Para ello se utilizan las órdenes:

sudo systemctl start openhab2.service sudo systemctl status openhab2.service

El primer inicio se demora unos 15 minutos. Cuando esté en marcha, podremos acceder a la plataforma IoT vía web con el navegador. Para ello, tenemos que poner en el navegador la dirección IP de la máquina en la que está instalado OpenHAB y acceder por el puerto 8080.

http://<direcciónIP>:8080

http://192.168.83.217:8080

Si la instalación se ha realizado correctamente, al acceder a la plataforma se debe obtener el resultado que se muestra en la Figura 5.5

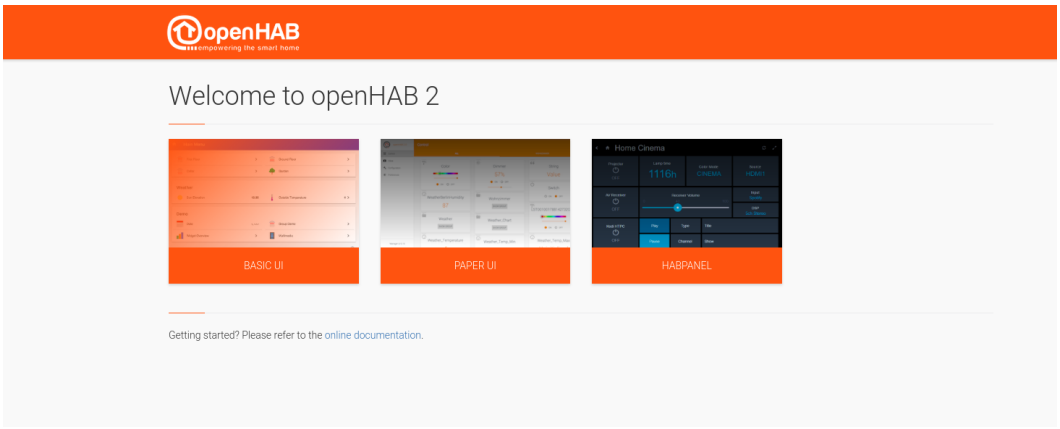

Figura 5.3: Instalación de OpenHAB

Cuando se realiza una instalación de OpenHAB es conveniente, por cuestiones de mejora en el funcionamiento, sobre todo de características tan importantes como la utilización del protocolo Z-Wave, crear los usuarios que serán los que tengan asociados los procesos. Estos usuarios se pueden crear con los siguientes comandos:

sudo adduser openhab dialout

sudo adduser openhab tty

sudo adduser openhab audio

Una cuestión importante, es poder definir OpenHAB como servicio de sistema para tenerlo siempre en ejecución y que se inicie cuando se inicia el sistema, para ello primero editamos el fichero "/lib/systemd/system/openhab2.service" con las siguientes líneas:

```
[Unit]
Description=The openHAB 2
Documentation=http://docs.openhab.org
Wants=network-online.target
After=network-online.target

[Service]
Type=simple
User=openhab
Group=openhab
GuessMainPID=yes
WorkingDirectory=/opt/openhab2
#EnvironmentFile=/etc/default/openhab2
ExecStart=/opt/openhab2/start.sh server
ExecStop=/bin/kill -SIGINT $MAINPID
Restart=on-failure

[Install]
WantedBy=multi-user.target
```

Adicionalmente, hay que dar los permisos para que la plataforma pueda actuar como servicio del sistema. Se hace con las siguientes ordenes:

```
sudo systemctl daemon-reload
sudo systemctl enable openhab2.service
```

Una vez realizado, se debe reiniciar la plataforma y ver su estado. Para ello, ejecutamos los siguientes comandos:

sudo systemctl start openhab2.service

sudo systemctl status openhab2.service

Si todo es correcto, la salida que debemos obtener por pantalla tiene que ser similar a la que podemos observar en la Figura 5.4.

Figura 5.4: Servicio de OpenHAB

Instalación de componentes

La instalación de componentes se realiza mediante la interfaz Paper UI como podemos observar en la Figura 5.3

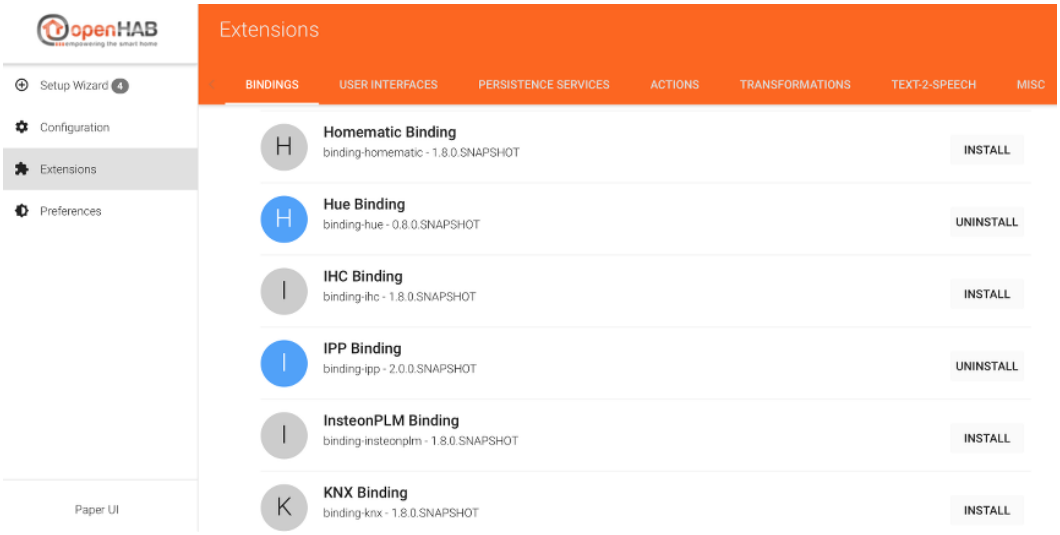

Figura 5.5: Instalación de componentes

Se pueden instalar componentes según su categoría, tenemos:

- Bindings.
- Acciones.
- Interfaces de usuario.
- Servicios de persistencia.
- Acciones.

Los pasos a seguir para realizar la instalación son los siguientes:

- En la columna de la izquierda de la interfaz Paper UI, entramos en "Add-Ons".
- Una vez dentro, navegamos por las categorías situadas en la parte superior de la página y seleccionamos la deseada.
- Cuando la seleccionamos, se navega por la misma hasta encontrar el componente a instalar.
- Para instalarlo, habrá que pulsar sobre el botón "instalar" que aparece a la derecha de cada componente.

Creación de entidades lógicas

Things

La vinculación de un Thing con OpenHAB se realiza mediante la interfaz de usuario Paper UI de la manera que se explica a continuación.

1. Desde PaperUI iniciamos la fase de "descubrimiento", para ello seleccionamos "Inbox" en el panel lateral y posteriormente seleccionamos el símbolo "+" situado en la parte superior.

2. Seleccionamos el binding que controlará el Thing a añadir. Por ejemplo, si queremos añadir un dispositivo con sensores ambientales de contacto que funciona por Z-Wave, tendremos que seleccionar el binding Z-Wave, con la restricción de que antes de esto el sensor debe estar emparejado con el controlador Z-Wave.

3. Nos aparecerá la información pantalla del nodo nuevo que hay emparejado en el controlador. Para añadirlo solo tendremos que marcar el tick en la parte izquierda del nodo.

4. Por último, se le introducen los datos al nuevo nodo (nombre, descripción).

Items

Tenemos varias formas de creación de items, a continuación se explican.
Desde Paper UI:

- Abrimos el desplegable "Configuración" de la parte lateral y seleccionamos Items.

- En la parte superior aparece un símbolo "+" con el que podremos añadir items.

- Debemos proporcionarle mínimo un nombre y un tipo, aunque también podemos incluirlo en un grupo.

La otra posible forma es desde el fichero "smartlab.items", que podremos encontrar en la siguiente ruta:

/etc/openhab2/items/smartlab.items

Para añadir un item desde el fichero basta con añadir una línea con la información que se detalla a continuación.

Tipo “NombreItem” “EtiquetaItem” <IconoEnSitemap> (Grupo) {canal}

Donde:

- **Tipo:** Player, Dimmer, Switch, String, Number, Color, Contact; según la representación indicada.
- **NombreItem:** Nombre con el que nos referimos al item.
- **EtiquetaItem:** Es el “label” que se mostrará en el sitemap. Si además del “label” en el sitemap queremos mostrar algún valor como puede ser un porcentaje, debemos especificarlo en esta etiqueta separado por un espacio del nombre y entre corchetes. Por ejemplo, para el volumen sería: “Volumen [%.1f%%]” y el resultado sería el que podemos observar en la Figura 5.6

Figura 5.6: Nivel de volumen

- **Grupo (opcional):** con esta etiqueta indicamos si el item pertenece a algún grupo y en caso afirmativo de qué grupo se trata.
- **Canal:** el canal es el enlace que comunica el item con el thing, este enlace se puede obtener desde la interfaz PaperUI en el apartado Channel de cualquier Thing. Los diferentes tipos que tenemos son:
  - **Luces:** {channel=”hue:0210:00178812e2fe:4:color”}
  - **Altavoces:** {channel=”sonos:PLAY1:RINCON_B8E937B5868C01400:volume”}
  - **Dispositivos Z-Wave:** {channel=”zwave:device:b2f947e6:node30:sensor_door”}
  - **MQTT:** {mqtt=”<[broker:smartthings/Motion Baño/motion:state:default]”}

Particularmente, el de MQTT no lo podemos obtener desde ninguna interfaz, se tiene que construir de forma manual teniendo en cuenta el topic al que queremos que el item se suscriba.

Grupos

Un Grupo es un tipo especial de item que se usa para definir una categoría para agrupar otros items. Un item puede estar en uno o más grupos. También se definen en el fichero items y la sintaxis es la siguiente:

Group groupname ["labeltext"] [<iconname>] [(group1, group2,...)]

Acciones

MQTT

Una vez instalada la acción, primero debemos configurarlo. Para ello en el fichero /etc/openhab2/services/mqtt.cfg configuramos la dirección del broker como se observa en la Figura 5.7.

```
broker.url=tcp://localhost:1883
broker.clientId=openhab

brokerExterno.url=tcp://150.214.174.25:8060
```

Figura 5.7: Configuración broker

Para configurarlo se referencia el broker con la variable brokerExterno y con el método ".url" se introduce la dirección física de donde se encuentra el broker.

Twitter

La configuración de la acción Twitter se lleva a cabo de la siguiente manera:

- Una vez instalada la acción, veremos por el log un mensaje con las instrucciones a seguir para su correcta configuración, como podemos observar en la Figura 5.8

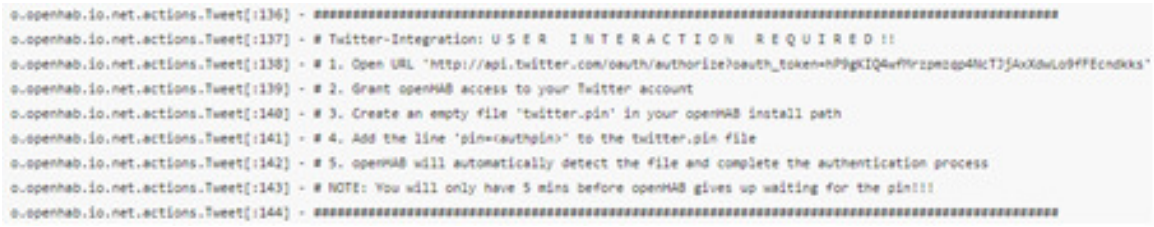

Figura 5.8: Configuración twitter

- El primer paso es abrir la url con la que daremos permiso desde Twitter a que OpenHAB publique en la cuenta.

- Seguidamente, creamos un fichero llamado twitter.pin, que se puede observar en la Figura 5.9 en el directorio de datos "/var/lib/openhab2" y en él se introduce el pin que genera la URL.

Figura 5.9: Configuración twitter (PIN)

- Una vez introducido el pin, automáticamente se genera el token para autenticar contra la API de Twitter y se habrá terminado la configuración, por ello en el log se mostrará un mensaje de confirmación, como observamos en la Figura 5.10.

Figura 5.10: Confirmación Twitter

Persistencia

Una vez instalado el módulo, el acceso a los datos se realiza mediante los siguientes métodos:

- <item>.persist Persiste el estado actual.
- <item>.lastUpdate Consulta de la última fecha y hora de actualización de un elemento determinado.
- <item>.historicState (AbstractInstant) Recupera el estado en un cierto punto en el tiempo.
- <item>.changedSince (AbstractInstant) Comprueba si el estado del elemento ha cambiado (alguna vez) desde cierto punto en el tiempo.
- <item>.updatedSince (AbstractInstant) Comprueba si el estado del elemento se ha actualizado desde cierto punto en el tiempo.
- <item>.maximumSince (AbstractInstant) Obtiene el elemento con el valor máximo (estado) desde cierto punto en el tiempo.
- <item>.minimumSince (AbstractInstant) Obtiene el elemento con el valor mínimo (estado) desde cierto punto en el tiempo.
- <item>.averageSince (AbstractInstant) Obtiene el valor promedio del estado de un elemento dado desde cierto punto en el tiempo.
- <item>.deltaSince (AbstractInstant) Obtiene el valor de diferencia del estado de un elemento dado desde un cierto punto en el tiempo.
- <item>.previousState () Recupera el elemento anterior.
- <item>.previousState (true) Recupera el elemento anterior, salta elementos con valores de estado iguales y busca el primer elemento con estado no igual al estado actual.

- <item>.sumSince (AbstractInstant) Recupera la suma de los estados previos desde cierto punto en el tiempo.

Es importante tener en cuenta que AbstractInstant viene representado por una marca temporal, la cual debe ser parseada con la siguiente estructura de fecha: "AÑO-MES-DIA".

A continuacion, se indican una serie de ejemplos:

Lights.changedSince(now.minusMinutes(2).minusSeconds(30))

Comprueba si las luces han cambiado de valor hace 2 minutos y 30 segundos atrás.

Temperature.maximumSince(now.toDateMidnight)

Devuelve el valor máximo de temperatura del último día

Temperature.minimumSince(parse("2012-01-01"))

Devuelve el valor mínimo de temperatura desde la fecha introducida.

Voice

Una vez instalada, el software detecta la instalación automáticamente y con ella puedes utilizar la acción "say something" en el motor de reglas, que acompañada de una cadena de texto reproduce ese texto en el altavoz.

Mantenimiento

OpenHAB tiene un mantenimiento sencillo que consta de los puntos que se detallan a continuación y en los siguientes apartados se revisan:

- Sistema host.
- Actualización.

- Desinstalación.
- Copias de seguridad.
- Log. Consola.
- Dispositivos.

Sistema host

El sistema que albergue la plataforma, debe estar completamente actualizado. Hay que tener en cuenta que OpenHAB es una plataforma desplegada sobre un sistema, por lo que los fallos de seguridad y errores en este sistema afectarán al correcto desempeño de la plataforma IoT. Es importante mantenerlo actualizado. Además, al funcionar OpenHAB sobre la máquina virtual de Java (JVM), es de gran importancia tener en cuenta este software en el sistema sobre el que se despliegue OpenHAB. La versión recomendada y, por tanto, la más indicada para que el funcionamiento de la plataforma IoT sea correcto es:

java version "1.8.0_121"
Java(TM) SE Runtime Environment (build 1.8.0_121-b13)
Java HotSpot(TM) Client VM (build 25.121-b13, mixed mode)

Actualización

Para actualizar la versión de la plataforma, se deben de utilizar los comandos que proporciona el sistema para la actualización de sus paquetes como si de otro software se tratara.

```
sudo apt-get update
sudo apt-get upgrade
```

De no saber que versión de la plataforma es la que hay disponible para su actualización en caso de haber varias, por ejemplo, por el paso del tiempo, se pueden listar las versiones antes de actualizar para seleccionar la que queremos instalar.

```
sudo apt-get update
apt-cache showpkg openhab2
```

En caso de realizar algún cambio sensible en la plataforma, como una actualización, podemos guardar su estado antes para luego poder recuperarlo con la orden:

```
sudo runtime/bin/update 2.2.0-SNAPSHOT
```

En el caso contrario, tener perfectamente identificada la versión que se va instalar, se puede hacer directamente con el siguiente comando:

```
sudo apt-get install openhab2=2.1.0-1
```

Desinstalación

Para desinstalar la plataforma IoT del sistema, habrá que ejecutar las siguientes órdenes:

```
sudo apt-get purge openhab2*
sudo rm /etc/apt/sources.list.d/openhab2.list
```

Copias de seguridad

Para realizar backups de la plataforma, tenemos disponible la siguiente orden:

```
sudo $OPENHAB_RUNTIME/bin/backup
```

Dicha orden genera un archivo ".zip" que contiene el backup de la plataforma.

Una vez tenemos copias de seguridad realizadas, la forma en la que se restauran es la siguiente:

sudo $OPENHAB_RUNTIME/bin/restore
$OPENHAB_BACKUPS/myBackup.zip

Consola de OpenHAB

OpenHAB posee una consola para realizar tareas dentro del mismo software. El acceso se realiza mediante ssh desde la misma máquina en la que esté instalada la plataforma:

ssh localhost -p 8101
Password: ———

La principal utilidad de la consola de OpenHAB en base a su mantenimiento, es el log. Para mostrarlo tendremos que escribir una vez dentro:

log:tail

Los ficheros de logs se encuentran en el directorio "/var/log/openhab2". El fichero que almacena los logs sobre eventos se llama events.log y el que almacena lo relacionado con el sistema es openhab.log. A continuación, se muestran algunos ejemplos sobre la utilidad de esta herramienta enfocada al mantenimiento.

grep -i "2018-01-23" events.log

Dicha orden devolverá todo el log relativo a eventos de ese día en concreto.

grep -i "10:41:00" events.log

En este caso, obtendremos las entradas que se generaron a esa hora exacta. Además, podremos obtener una hora sin necesidad de especificar minutos o segundos usando comodines en el string:

grep -i “10:*” events.log

También podemos buscar, por ejemplo, lo ocurrido el día x a una hora y:

grep -i “2018-01-23 10:*” events.log

Nos devolverá todo lo que ocurrió el día 23/01/2018 en el intervalo de 10:00 a 10:59 horas. Adicionalmente, podremos filtrar por el tipo de mensaje: información, warmings. Esto lo podremos hacer en el fichero openhab.log debido a que el de eventos no proporciona mensajes de información o problemas. Por ejemplo:

grep -i “INFO” openhab.log
grep -i “WARN” openhab.log

Dispositivos

En cuanto a todos los dispositivos que se han integrado, hay que realizar su mantenimiento también. Las cuestiones relevantes en este aspecto, las podremos encontrar en el manual de cada uno de los dispositivos. En general, el trabajo común con todos es estar al tanto del nivel de batería que poseen cada uno para cuando este nivel sea bajo, o se agote, se pueda actuar sin perder ningún dato ofrecido por ese dispositivo. La manera de actuación es sustituir las baterías del dispositivo por otras nuevas.

Bibliografía

[1] "UJAmi - University of Jaén and Ambient Intelligent." url: http://ceatic.ujaen.es/ujami/.

[2] "Smartlab CEATIC [online]." url: http://ceatic.ujaen.es/es/smartlabweb.

[3] D. Evans, "Internet de las cosas," *Cómo la próxima evolución de Internet lo cambia todo. Cisco Internet Bussiness Solutions Group-IBSG*, vol. 11, no. 1, pp. 4–11, 2011.

[4] A. Steventon and S. Wright, *Intelligent spaces: The application of pervasive ICT*. Springer Science & Business Media, 2010.

[5] M. Espinilla, L. Martinez, J. Medina, and C. Nugent, "The experience of developing the UJAmi Smart Lab," vol. PP, pp. 1–1, 06 2018.

[6] P. Sanmartín Mendoza, K. Ávila Hernández, C. Vilora Núñez, and D. Jabba Molinares, "Internet de las cosas y la salud centrada en el hogar," *Revista Salud Uninorte*, vol. 32, no. 2, pp. 337–351, 2016.

[7] D. Zafra, J. Medina, L. Martinez, C. Nugent, and M. Espinilla, "A web system for managing and monitoring smart environments," in *Bioinformatics and Biomedical Engineering* (F. Ortuño and I. Rojas, eds.), (Cham), pp. 677–688, Springer International Publishing, 2016.

[8] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, and Z. Yu, "Sensor-based activity recognition," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, pp. 790–808, Nov 2012.

[9] F. Rivero, "Informe ditrendia 2016: Mobile en España y en el mundo," 2016.

[10] D. C. Dominguez, "Las redes sociales. tipología, uso y consumo de las redes 2.0 en la sociedad digital actual," *Documentación de las Ciencias de la Información*, vol. 33, pp. 45–68, 2010.

[11] M. Weiser, "The Computer for the 21st Century," *Scientific American*, vol. 265, p. 94, 09 1991.

[12] K. Ashton *et al.*, "That 'internet of things' thing," *RFID journal*, vol. 22, no. 7, pp. 97–114, 2009.

[13] S. Duggleby, "El desarrollo del IoT industrial toma impulso con la plataforma arduino," *Revista española de electrónica*, no. 744, pp. 92–93, 2016.

[14] A. Steventon and S. Wright, *Intelligent Spaces: The Application of Pervasive ICT*. Springer-Verlag, 2006.

[15] M. F. de la Colina, "Hacia una definición de la domótica," *Informes de la Construcción*, vol. 56, no. 494, pp. 11–17, 2004.

[16] "Sistema para la monitorización de ambientes inteligentes a través de la gestión y acceso a servicios [online]." url: http://sinbad2.ujaen.es/?q=es/work/2652-sistema-para-la-monitorización-de-ambientes-inteligentes-través-de-la-gestión-y-acceso-servicios.

[17] R. K. Kodali, V. Jain, S. Bose, and L. Boppana, "IoT based smart security and home automation system," in *Computing, Communication and Automation (ICCCA), 2016 International Conference on*, pp. 1286–1289, IEEE, 2016.

[18] J. Santos, J. J. Rodrigues, B. M. Silva, J. Casal, K. Saleem, and V. Denisov, "An IoT-based mobile gateway for intelligent personal assistants on mobile health environments," *Journal of Network and Computer Applications*, vol. 71, pp. 194–204, 2016.

[19] E. Balandina, S. Balandin, Y. Koucheryavy, and D. Mouromtsev, "IoT use cases in healthcare and tourism," in *Business Informatics (CBI), 2015 IEEE 17th Conference on*, vol. 2, pp. 37–44, IEEE, 2015.

[20] F. TongKe, "Smart agriculture based on cloud computing and IoT," *Journal of Convergence Information Technology*, vol. 8, no. 2, 2013.

[21] A. Volkov, A. Khakimov, A. Muthanna, R. Kirichek, A. Vladyko, and A. Koucheryavy, "Interaction of the IoT traffic generated by a smart city segment with SDN core network," in *International Conference on Wired/Wireless Internet Communication*, pp. 115–126, Springer, 2017.

[22] P. Kinney *et al.*, "Zigbee technology: Wireless control that simply works," in *Communications design conference*, vol. 2, pp. 1–7, 2003.

[23] G. Mulligan, "The 6LoWPAN architecture," in *Proceedings of the 4th workshop on Embedded networked sensors*, pp. 78–82, ACM, 2007.

[24] C. Withanage, R. Ashok, C. Yuen, and K. Otto, "A comparison of the popular home automation technologies," in *Innovative Smart Grid Technologies-Asia (ISGT Asia), 2014 IEEE*, pp. 600–605, IEEE, 2014.

[25] E. Georgakakis, S. A. Nikolidakis, D. D. Vergados, and C. Douligeris, "An analysis of bluetooth, zigbee and bluetooth low energy and their use in wbans," in *Wireless Mobile Communication and Healthcare*, pp. 168–175, Springer, 2011.

[26] N. Gall, "The Origin (Coining) of the Term 'Middleware'," 2003.

[27] J. Aguilar, M. Jerez, E. Exposito, and T. Villemur, "Carmicloc: context awareness middleware in cloud computing," in *Computing Conference (CLEI), 2015 Latin American*, pp. 1–10, IEEE, 2015.

[28] C. A. E. Galicia, W. G. Lopez, and R. F. G. Mallette, "Implementación de plataforma para el internet de las cosas en un ambiente de nube pública IoT platform development in a public cloud environment,"

[29] "European Network of Living Labs (ENoLL)." url: https://enoll.org/.

[30] "Smart Environments Reserch Group." url: https://www.ulster.ac.uk/research/institutes/computer-science/groups/smart-environments.

[31] "Halmstad Living Lab." url: http://www.halmstadlivinglab.se/index.php?page=hll.

[32] "Universidad de Halmstad." url: http://www.hh.se/en-US/5.html.

[33] "Universidad Tecnológica de Luleå." url: https://www.ltu.se/.

[34] "Botnia Living Lab." url: https://www.ltu.se/research/subjects/information-systems/Botnia-Living-Lab?l=en.

[35] N. Irvine, C. Nugent, S. Zhang, H. Wang, W. W. Ng, I. Cleland, and M. Espinilla, "The impact of dataset quality on the performance of data-driven approaches for human activity recognition,"

[36] H. Alemdar and C. Ersoy, "Multi-resident activity tracking and recognition in smart environments. journal of ambient intelligence and humanized computing (4): 513–529," 2017.

[37] C. Li, M. Lin, L. T. Yang, and C. Ding, "Integrating the enriched feature with machine learning algorithms for human movement and fall detection," *The Journal of Supercomputing*, vol. 67, no. 3, pp. 854–865, 2014.

[38] "Samsung smartthings." url: https://www.samsung.com/us/smart-home/smartthings/.

[39] "Domoticz." url: http://www.domoticz.com/.

[40] "Home assistant." url: https://home-assistant.io.

[41] "Openhab." url: https://www.openhab.org/.

[42] J. Rafferty, J. Synnott, A. Ennis, C. Nugent, I. McChesney, and I. Cleland, "Sensorcentral: A research oriented, device agnostic, sensor data platform," pp. 97–108, 11 2017.

[43] M. d. Buenaga, M. J. Maña López, L. Borrajo, D. Gachet, J. Mata Vázquez, and E. L. Lorenzo, "Iphealth: Plataforma inteligente basada en open, linked y big data para la toma de decisiones y aprendizaje en el ámbito de la salud," 2015.

[44] "Sistemas inteligentes basados en análisis de decisión difuso." url: https://sinbad2.ujaen.es/.

[45] A. Berson, *Client/Server Architecture*. New York, NY, USA: McGraw-Hill, Inc., 1992.

[46] "Twitter." url: https://twitter.com/?lang=es.

[47] L. Richardson and S. Ruby, *Restful Web Services*. O'Reilly, first ed., 2007.

[48] U. Hunkeler, H. L. Truong, and A. J. Stanford Clark, "MqTT-S - A publish/subscribe protocol for Wireless Sensor Networks.," in *COMSWARE*, pp. 791–798, IEEE, 2008.