UNIVERSIDAD DE MURCIA

DEPARTAMENTO DE INFORMÁTICA Y SISTEMAS

Sistema de Diálogo Basado en Mensajería Instantánea para el Control de Dispositivos en el Internet de las Cosas

D. José Ángel Noguera Arnaldos

2016

UNIVERSIDAD DE MURCIA
Departamento de Informática y Sistemas

TESIS DOCTORAL

SISTEMA DE DIÁLOGO BASADO EN MENSAJERÍA INSTANTÁNEA PARA EL CONTROL DE DISPOSITIVOS EN EL INTERNET DE LAS COSAS

José Ángel Noguera Arnaldos

ENERO 2016

Director:
Dr. Rafael Valencia García

Agradecimientos

A mi mujer Isabel y a mis hijos Pablo y Arturo, por su infinita generosidad

A mi madre, por su apoyo y cariño

A Rafa, por darme la oportunidad

A mis compañeros de Proasistech, por su inestimable ayuda

A todos, gracias

ÍNDICES

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE GRÁFICAS

ÍNDICE DE ALGORITMOS

LISTADO DE ACRÓNIMOS

Término	***Significado***
HTML	*Hiper Text Markup Language*
IoT	*Internet of Things*
XMPP	*Extensible Messaging and Presence Protocol*
App	*Aplicación de software que se instala en dispositivos móviles o tablets*
BOT	*Programa informático imitando el comportamiento humano*
AIML	*Artificial Intelligence Mark-up Language*
KNX	*Estadar abierto para el control de casas y edificios*
IPv4	*Internet Protocol version 4*
IPv6	*Internet Protocol version 6*
HMI	*Interface Hombre-máquina*
IEEE	*Instituto de Ingeniería Eléctrica y Electrónica*
ETSI	*European Telecommunications Standards Institute*
ISO	*International Organization for Standardization*
IBM	*International Business Machines Corp.*
M2M	*Machine to machine*
MQTT	*Message Queue Telemetry Transport*
HTTP	*Hypertext Transfer Protocol*
TCP-IP	*Conjunto de protocolos de red en los que se basa Internet*
TLS	*Transport Layer Security*
SSL	*Secure Sockets Layer*
VoIP	*Voz sobre Protocolo de Internet*
P2P	*Peer-to-peer*
CHAT	*Comunicación en tiempo real que se realiza entre varios usuarios conectados en red*
IRC	*Protocolo de comunicación en tiempo real basado en texto*
AIM	*Cliente de mensajería instantánea de America On Line (Instant Messenger)*
ALICE	*Artificial Intelligence Foundation*
AVC	*Agente Virtual Conversacional*

Resumen

Con el nacimiento de Internet nació la era de la comunicación global. Los adelantos en I+D+i en los campos del conocimiento de la Informática, Telecomunicaciones y Electrónica han posibilitado que Internet llegue de forma estable, rápida y segura a gran parte de la humanidad. La necesidad de comunicarnos entre los humanos ha llevado a inventar y mejorar estas tecnologías. Por ejemplo pasamos de tener un teléfono en casa al hecho de poder, hoy en día, llevarlo a todas partes gracias a la perseverancia, tenacidad, inteligencia y afán de superación de ingenieros como Martin Cooper (Motorola) cuando el 17 de octubre de 1973 dio a conocer al mundo el primer teléfono móvil. Se siguió investigando y desarrollando nuevos sistemas en el de la tecnología móvil y en poco más de 20 años nacía el primer teléfono móvil "inteligente" presentado por la compañía global Ericsson denominado GS88 "Penelope". Pasamos del reto tecnológico de hacer una llamada telefónica a intercambiar archivos fotográficos en tiempo real con el teléfono a través de Internet gracias a estos teléfonos inteligentes: más potentes, más pequeños, más inteligentes... conectados con todo y con todos.

La tecnología de la microelectrónica, entendida como la evolución natural indivisible de la tecnología de semiconductores de la década de los 50 y la del microprocesador de la década de los 70, ha supuesto en todo este escenario tecnológico una palanca fundamental y necesaria para hacer realidad la forma en que vivimos. Podemos afirmar que esta revolución tecnológica se desarrolló gracias a ingenieros y científicos como Werner Jacobi (circuito integrado), Marcian Edward "Ted" Hoff y Robert Noyce (microprocesador).

El ser humano no se impone límites tecnológicos y si tiene una necesidad busca una respuesta para su solución. Eso mismo debió pensar en la década de los 90 Tim Berners Lee cuando diseñó la primera versión del HTML (*Hiper Text Markup Language*) para la World Wide Web, un lenguaje para marcar textos que permitía incluir enlaces a otros documentos para el intercambio de documentos entre usuarios.

Al igual que los humanos estamos conectados a través de Internet, ¿se podrían conectar las cosas? Esta simple pregunta ha tenido una gran respuesta con el nacimiento de la Internet de las cosas o *Internet of things*. Hemos pasado en pocos años de hablar Internet a hablar del Internet de las cosas. Esas cosas que en la mayoría de los casos conviven en lo cotidiano con nosotros como, por ejemplo, los electrodomésticos. La distancia en la forma de interrelacionarnos con las cosas está estrechándose cada vez más. A día de hoy la forma de entendernos con un electrodoméstico, en la mayoría de los casos, es por medio de una interface tipo botonera. Dando un paso más en el desarrollo de la tecnología móvil, por medio de las aplicaciones o Apps, van apareciendo desarrollos en los que un usuario puede controlar distintas máquinas y dispositivos a través de la interfaz táctil de estos dispositivos móviles como, por ejemplo, pedir que una cafetera prepare un café.

Por otro lado, es cada vez más importante que las máquinas y los humanos se puedan comunicar y entender entre sí. Para lograr este entendimiento inexorablemente debemos de hablar de la Inteligencia Artificial. La lógica y algoritmos informáticos que encierra esta disciplina, tuvo un empuje sin igual en la década de los 50 por parte del matemático Alan Turing y sus famosa Máquina de Turing.

En la actualidad a pesar de los avances descritos anteriormente aún queda mucho por hacer en lo que a la comunicación entre hombre-máquina se refiere. El lenguaje natural que empleamos los humanos para comunicarnos entre nosotros debería ser el nexo de unión que nos acerque a la forma de comunicarnos con las máquinas y nuestro entorno. Esto supone avanzar en el campo de Protocolos de Comunicación, Inteligencia Artificial y Electrónica así como en el concepto de redes sociales colaborativas formadas por usuarios y dispositivos entre los que exista un nexo común de entendimiento: el lenguaje natural. Varios enfoques se han desarrollado a lo largo de los últimos años pero, debido a las carencias que algunos de estos enfoques presentan, aún no existe ninguna metodología estándar que permita realizar esta tesis doctoral ha sido, por tanto, proponer un enfoque dentro del ámbito de la comunicación con las máquinas en lenguaje natural, una nueva metodología para

el diseño de sistemas centrados en redes sociales tipo chat basados en lenguaje natural que cubra todas las necesidades de actuación y comunicación con distintos dispositivos como electrodomésticos, así como su extensión futura al resto de máquinas. Esta metodología está constituida por una serie de fases entre la que se pueden destacar las siguientes: el diseño e implementación de una red social en formato chat, el diseño e implementación de un servidor compartido en la nube así como los protocolos de comunicación y mensajería instantánea, el diseño e implementación de un módulo de dialogo basado en procesamiento del lenguaje natural y ontologías y, finalmente, el desarrollo de la electrónica capaz de interaccionar con el electrodoméstico y comunicarse con usuario a través de Internet.

La primera etapa tiene por objetivo afrontar un nuevo concepto de red social en formato chat a través de la cual el usuario y el dispositivo empleen el lenguaje natural escrito para comunicarse. Con esta solución se da un salto cuantitativo en este concepto de comunicación ya que hasta el momento estas redes sociales en formato chat solo las utilizan los usuarios por lo que su desarrollo supondrá dotar de inteligencia a los dispositivos, que como primer paso los centraremos en electrodomésticos.

La siguiente etapa se centra en afrontar el desarrollo de un servidor compartido en la nube donde gestionar los usuarios del chat. Los usuarios serán tanto humanos como dispositivos y la comunicación se establecerá de forma bidireccional desde humano-dispositivo y dispositivo-dispositivo. Para llevar a cabo esta fase es necesario modificar en algunos casos y crear en otros nuevos protocolos de comunicación y mensajería instantánea basados en estándar como XMPP.

La tercera etapa tiene por objetivo dotar de inteligencia a los dispositivos para que la comunicación con los usuarios sea en lenguaje natural. Para ello el uso de tecnologías de procesamiento del lenguaje natural ontológicas nos ha permitido crear una base de conocimiento con patrones de comunicación según el electrodoméstico a utilizar. Por otro lado el usuario podrá utilizar los dispositivos

sin necesidad de conocer comandos de control funcionales sino escribir en lenguaje natural los que quiere que el dispositivo haga. Además, los dispositivos se pueden comunicar con los usuarios de manera asíncrona mandando un mensaje en lenguaje natural cuando se active una alarma o cambie el estado de algún sensor del dispositivo o electrodoméstico.

La cuarta etapa afronta el uso de la electrónica capaz de realizar la tarea de interaccionar con el electrodoméstico y embeber los algoritmos desarrollado en la fase tercera que permita su integración como "usuario" dentro del chat. Para ello se han evaluado distintos desarrollos electrónicos comerciales basados en microprocesador para determinar su idoneidad en nuestra solución.

Con objeto evaluar el rendimiento de la metodología desarrollada y validación del sistema propuesto en esta tesis doctoral por usuarios finales interesados en los resultados del proyecto, se diseñó una estrategia de evaluación del sistema basado en la evaluación del rendimiento de los sistemas de control de comandos y los sistemas de diálogo. Para esta evaluación se utilizaron un conjunto de métricas de evaluación típicas de la disciplina de extracción de información como la precisión, cobertura y medida-F. Mediante estas métricas, que son de aplicación muy común en los procesos de evaluación de sistemas de procesamiento del lenguaje natural y de recuperación y extracción de información, se evaluó el rendimiento del sistema con uno resultados muy prometedores.

Capítulo I.- Introducción

Internet de la Cosas ha supuesto una extensión del propio concepto de Internet. Es difícil entender que a día de hoy la inteligencia de un sistema pueda residir en una parte concreta o centralizada cuando todo tiende a que esta inteligencia sea distribuida *(Atzori et. al, 2010).* El término de Internet de las Cosas abarca un modelo de estructurar los datos y sistemas de comunicación que hoy conocemos de forma que la inteligencia sea ubicua (*Xia et. al, 1012*), y la conectividad inteligente y escalable *(Guido et. al, 2010),* (*Troyano, 2011*).

La Inteligencia Artificial *(Jang et. al, 1997)*, ha encontrado en las ontologías el modelo de representación del conocimiento ideal para describir de manera formal el contenido de los recursos a utilizar en la comunicación hombre-máquina en lenguaje natural. Las ontologías son la tecnología más importante dentro de la Web Semántica y permiten representar el conocimiento de manera explícita y formal para que sea procesable por las máquinas. Más concretamente, una ontología puede definirse como una "*representación formal y explícita de una conceptualización compartida*" *(Studer et. al, 1998)*.

Internet y las redes sociales han revolucionado la forma en que nos comunicamos. Estamos informados en tiempo real de lo que sucede en cualquier parte del mundo gracias a aplicaciones para smartphones y tabletas (Apps). La aparición de dispositivos móviles como *smartphones* y tabletas ha propiciado la extensión y el uso de Internet. Tal es así que en 2014 se crearon alrededor de **un millón** de aplicaciones móviles[1].

Todos estos adelantos tecnológicos son los precursores de la denominada "revolución de la información": podemos comunicarnos a través de dispositivos electrónicos y sin embargo existe un vacío tecnológico aún por estandarizar en lo referente a la comunicación con los propios dispositivos en lenguaje natural.

[1] www.antevenio.com (accesible el 8 de septiembre de 2015)

Desarrollar y sentar las bases de un sistema de comunicación basado en redes sociales tipo chat para que los humanos nos podamos comunicar en lenguaje natural con las máquinas es el principal objetivo de esta tesis doctoral.

Las razones expuestas en los párrafos anteriores han sido las principales motivaciones para la realización de la investigación que se describe en esta tesis doctoral. La solución que se propone en este trabajo de investigación se sustenta en el desarrollo de una metodología para obtener un nuevo sistema de diálogo de comunicación hombre-máquina en lenguaje natural empleando redes sociales en formato chat basada en modelos ontológicos formales.

Para lograr este objetivo, se ha seguido la siguiente metodología:

- Análisis del estado del arte en Internet de las cosas, sistemas de diálogo y agentes conversacionales, inteligencia ambiental y domótica, bases de conocimiento y ontologías, hardware para IoT y tecnologías inalámbricas. Este estudio del estado del arte implicó el estudio y análisis de los antecedentes de las tecnologías a incorporar en el proyecto.

- Análisis detallado de las metodologías actuales en la evolución de la Internet de las Cosas y sus aplicaciones en el campo de la domótica y control de electrodomésticos.

- Definición y desarrollo de un sistema de diálogo basado en mensajería instantánea para el control de aparatos en el Internet de las Cosas (IoT). Esta tarea se dividió en distintas partes: (i) definición y desarrollo de un software que permita interpretar las instrucciones en lenguaje natural e interactuar con los dispositivos; (ii) definición y desarrollo de medios de comunicación de la aplicación móvil de chat; (iii) definición y desarrollo de medios de restricción de acceso y autenticación.; (iv) definición y desarrollo para configuración de Bot desde la aplicación cliente; (v) definición y desarrollo de la base del conocimiento; (vi) definición y desarrollo para la adaptación de los protocolos XMPP y AIML; (vii)

definición y desarrollo de los medios de seguridad y control de acciones a ejecutar por parte del usuario a través del chat.

- Definición y desarrollo de una arquitectura informatizada basada en Internet de las Cosas denominada **im4Things** formada por tres módulos principales: Im4Things app, Im4ThingsCloudService y el Dispositivo. Esta tarea se dividió en distintas partes: (i) definición y desarrollo de la aplicación de usuario: Im4Things app, (ii) definición y desarrollo aplicación servidor: Im4ThingsCloudService; (iii) definición y desarrollo del hardware y software para el Dispositivo.

- Validación de la metodología empleada desarrollada en dos direcciones bien diferenciadas: comprobación de resultados sobre prototipo real y comprobación de la aceptación de un grupo de 50 probadores del sistema (tester) por medio de análisis sociolingüístico.

Los objetivos que se establecieron en el desarrollo de la metodología descrita se han realizado con éxito, y los resultados que se han conseguido se presentan en esta memoria con la siguiente organización:

En este Capítulo I se realiza una introducción al contenido del presente proyecto y sitúa al lector en el foco tecnológico y de investigación que disco proyecto.

En el Capítulo II se realiza un detallado estudio del estado del arte de las tecnologías relacionadas con esta investigación que se encuentran integradas en las metodologías desarrolladas. El estudio comienza con un análisis profundo sobre la Internet de las Cosas como tecnología trasversal a otros campos de la técnica. Continúa con un estudio sobre los distintos protocolos de comunicación existentes y el paradigma de su desarrollo como protocolos propietarios o abiertos. El siguiente apartado centra la atención en el protocolo MQTT presentando su arquitectura y estructura de contenidos para pasar al siguiente apartado donde se define el protocolo XMPP como base para el desarrollo de todo el ecosistema presentada en esta tesis doctoral. Los siguientes apartados se

centran en presentar las diferencias entre dichos protocolos. A continuación se presentan diversos software de mensajería instantánea basada en el protocolo XMPP como clientes. El siguiente apartado se detalla los sistemas de diálogo conversacionales para llegar a la definición de los agentes conversacionales virtuales basados en el lenguaje de programación AIML. A continuación se detallan los conceptos de inteligencia ambiental y domótica y de cómo la tecnología esta tecnología acerca al usuario y dispositivos para la interrelación de funciones. En el siguiente apartado nos centramos en la base del conocimiento basada en ontologías para ser aplicadas en redes sociales en formato chat. Extendemos su uso a partir de sus nuevas características como la extensibilidad sintáctica, propiedades adicionales y constructores de cardinalidad calificados, metamodelos simples, cadenas de propiedades y anotaciones extendidas. Finalmente llegamos a la parte del hardware definido para Internet de las Cosas y su relación directa con los distintos tipos de comunicación: Wifi y Bluetooth. En este apartado presentamos las distintas opciones de hardware comercial usado en el ecosistema de la Internet de las Cosas y acabamos el capítulo describiendo los problemas a resolver en esta tesis doctoral.

En el Capítulo III definimos la solución propuesta para resolver los problemas descritos en el último apartado del anterior capítulo. Comenzamos definiendo en qué consiste el sistema de diálogo basado en mensajería instantánea para el control de aparatos en el Internet de las Cosas (IoT) propuesto en esta tesis doctoral así como la definición de los objetivos a cumplir. En el siguiente apartado presentamos la arquitectura del sistema im4Things donde quedan aclarados los distintos bloques que la forman para llegar a definir de forma pormenorizada cada uno de estos bloques en los siguientes apartados. Seguidamente se definen los distintos servicios que ofrece el sistema que van desde el sistema de diálogo hasta los servicios que ofrece la aplicación servidor Im4ThingsCloudService. A continuación se trata el aspecto de la gestión de colas en el servidor para tratar la información circulante, y se presentan los resultados de los cálculos matemáticos basadas en el modelo Engset. A continuación justificamos el uso de hardware basado en microprocesadores frente a la versión de microcontrolador por medio de los resultados de validación. Seguidamente se desarrolla cómo emplear

ontologías en el sistema im4Things y su idoneidad. A continuación se define un protocolo de seguridad utilizado para evitar fraudes y ataques contra el servidor para finalmente acabar detallando cómo se gestiona los bots de los dispositivos en el servicio Im4ThingsCloudService. El protocolo de seguridad es fundamental en este tipo de sistemas porque el usuario debe saber que la seguridad es el primer paso para su tranquilidad y le llevará a utilizarlo. Por otro lado, si el usuario percibe que el control de su propiedad está en peligro dejará de utilizar el sistema.

El Capítulo IV se centra en las medidas de validación basadas en la precisión y cobertura para pasar al siguiente apartado donde se detalla el experimento de validación realizado basado en un test sociolingüístico. El capítulo acaba detallando los resultados de la validación con usuarios finales.

En el Capítulo V se presentan las conclusiones y líneas futuras de trabajo que podrían continuarse partiendo de esta tesis doctoral.

Por último, en el Capítulo VI se muestran las contribuciones científicas derivadas de esta tesis doctoral.

Capítulo II.-Estado del Arte

II.1 INTRODUCCIÓN

En este capítulo se describe el estado actual de las diferentes tecnologías que componen el núcleo central de esta tesis doctoral.

En la primera y segunda sección se describe el concepto de Internet de las Cosas (IoT) desde el punto de vista de una tecnología que aporta valor a diferentes verticales de la técnica, de cómo IoT está moldeando nuevas formas de comunicarnos e integrar cada vez más sus raíces en nuestra vida diaria. En los diferentes subapartados de este primer punto apuntamos al escenario actual respecto a los protocolos vivos y que pueden contribuir cada uno a su forma, dentro del ecosistema IoT. Se pretende dejar claro cuáles de ellos son los idóneos para IoT desde el punto de vista funcional, escalabilidad y seguridad. Se introducen también los diferentes sistemas de mensajería instantánea presentes en el panorama actual en función de las plataformas.

En la tercera sección se introduce los diferentes lenguajes de programación para la IoT. Nos fijamos especialmente en el lenguaje de marcas AIML porque es el lenguaje que permite estructurar la información en formato tabla para ser embebida en los agentes conversacionales virtuales (Bots) para la comunicación hombre-máquina. En los subapartados de esta sección se presentan las diferentes técnicas aplicadas en los Bots y de su extensión a introducir estas técnicas en redes sociales del tipo chat. A partir de los resultados de estas técnicas se introduce el concepto de Inteligencia Ambiental, primer paso para la relación humano-cosa por medio de técnicas de inteligencia artificial.

En la sección cuarta se presenta el panorama actual de la domótica basada en el estándar KNX y cómo los sistemas cableados pueden ser integrados en una plataforma de control basada en chat. Se pretende con ello que la técnica haga más confortable la vida en la vivienda para personas dependientes.

En la sección quinta introducimos los conceptos de ontologías aplicadas a soluciones de redes sociales IoT chat y de cómo su empleo en la comunicación hombre-máquina mejoran considerablemente la operatividad y entendimiento de

los comandos a ejecutar por parte del dispositivo.

En la sección sexta, se presenta las distintas soluciones hardware embebidas para su integración en el ecosistema de la IoT. Estas soluciones de bajo coste, también conocidas como *low cost* aportan valor añadido a los diseños finales y consiguen en conjunto que el concepto de IoT se extienda más allá de las empresas fabricantes.

En las dos últimas secciones se introduce el panorama actual relativo a los distintos protocolos de comunicación y la tecnología inalámbrica al servicio de la IoT y de cómo cada uno aporta valor añadido a distintas soluciones.

En el último apartado de este capítulo, se expone brevemente cuál es el problema que se ha tratado de resolver en esta tesis doctoral.

II.2 INTERNET OF THINGS (IoT)

II.2.1 Internet transversal

Internet está revolucionado nuestra forma de comunicarnos y en cierta medida está modelando el día a día y nuestra forma de vida. Estamos en la era de la comunicación e interrelación digital entre personas a través de Internet (*Internet-People-Internet*) y cada vez con más frecuencia con las "cosas". A este último término se le conoce de forma más extendida como Internet de las Cosas (*Internet of Things o IoT*) y sus aplicaciones están presentes con más frecuencia en campos dispares de la ciencia y tecnología, ya sea para las ciudades inteligentes, también conocidas como *Smart Cities*, Eficiencia Energética, Hospitalario, Domótica, Dependencia y Discapacidad, Industria del automóvil, Agricultura, inteligencia distribuida y así un largo etcétera *(Atzori et. al, 2010).* Por otro lado, el factor de la integración de diversas tecnologías y soluciones de comunicaciones será determinante para hacer crecer este paradigma tecnológico (*Atzori et. al, 2010)*. Además, respecto a los protocolos de comunicaciones actuales, estos deben ser mejorados o incluso en su caso diseñar nuevos estándares que permitan una comunicación ilimitada vía Internet.

El término de Internet de las Cosas se refiere generalmente como la interconexión mediante redes de objetos cotidianos que contienen normalmente inteligencia ubicua (*Xia et. al, 2012)*. Esta tecnología incrementará la ubiquidad de Internet integrando cada objeto para su interacción mediante sistemas embebidos que permitirá la comunicación con los humanos y con otros dispositivos. Ejemplo de ello lo constituye las redes de sensores Mesh escalables e inteligentes *(Guido et. al, 2010) y (Troyano, 2011).*

Además, la aparición de dispositivos móviles como *smartphones* y tabletas ha propiciado la extensión y el uso de Internet. Según el portal especializado en tecnologías emergentes Gigaom Research[2], en 2014 se crearon alrededor de **un millón** de aplicaciones móviles[3]. En ese mismo año los sistemas operativos con más descargas fueron:

[2] www.research.gigaom.com (accesible el 13 de octubre de 2015)
[3] www.antevenio.com (accesible el 8 de septiembre de 2015)

- Dispositivos Android con más de 29.000 millones de descargas y 850.000 Apps en el *Google Play*[4].
- Dispositivos iOS con alrededor de 27.000 millones de descargas y alrededor de 905.000 aplicaciones en la *App Store*[5].

El 22 de julio de 2014, el portal de desarrolladores especializados en aplicaciones móviles Yeeply realizó el estudio "Economía App: Nuestros hábitos de uso y consumo de aplicaciones móviles[6]" informó que en España los usuarios del sistema operativo iOS (Apple) son los que más descargas de aplicaciones realizan, incluso siendo la plataforma donde hay más aplicaciones de pago. Con 88 Apps descargadas de media por dispositivo, iOS supera a Android que contabiliza 67 descargas de media por dispositivo, 57 para Windows y 49 para Blackberry.

En el mercado Domótico, en las últimas décadas se ha incrementado el uso y las necesidades de incorporar nuevas tecnologías en los hogares *(Harper, 2006)*. El desarrollo de Internet, el incremento de accesos a la red de redes en los hogares y la necesidad de dotar a la vivienda con las últimas tecnologías y comodidades son factores cada vez más a tener en cuenta en la construcción de nuevas viviendas.

Esto hace posible que nos veamos rodeados de multitud de electrodomésticos, sistemas de climatización, sistemas de seguridad, equipos multimedia y sobre todo smartphones y tabletas, además de una conexión a Internet.

Respecto a la tecnología aplicada a la vivienda, cada vez es mayor el número de sistemas con capacidad de interconexión a Internet. El problema actual es que toda esta tecnología está descentralizada teniendo distintos mandos o aplicaciones para cada dispositivo, distintas formas de funcionamiento y comunicación conllevando un gasto elevado por un uso no eficiente de la mayor parte de ellos.

II.2.2 Definición

Según Tessel Renzenbrink editor de TFF Elektor, Internet de las Cosas (*Internet of Things, IoT*), es "**la fusión del mundo físico y el mundo digital**". A finales del

[4] www.play.google.com (accesible el 8 de septiembre de 2015)
[5] www.app-store.es (accesible el 8 de septiembre de 2015)
[6] www.yeeply.com/blog/economia-app-habitos-y-uso-de-aplicaciones-moviles/ (accesible el 8 de septiembre de 2015)

año 2012, Tessel escribía en la revista técnica Elektor *(Dogan y Renzenbrink, 2012)* "*a los objetos comunes, que contendrán ordenadores diminutos, se les adjudicará una dirección IPv6*[7] *y tendrán una representación digital en Internet. Seremos capaces de tener acceso a la información de los objetos presentes en nuestras vidas cotidianas a través de Internet e incluso controlarlos remotamente*".

Esta premisa nos lleva indudablemente a pensar que se puedan realizar procesos sencillos como sobre distintos aparatos a través de Internet como que los usuarios tengan acceso a la información de estado de cualquier dispositivo. Por ejemplo, existen termostatos[8] conectados a Internet que permiten el control de temperatura de una máquina de aire acondicionado que regula el nivel de confort de una estancia.

El propósito de la IoT pasa por preguntarse si la sociedad necesita esta tecnología. En realidad se podría decir que es una evolución natural de la tecnología actual adaptada a necesidades reales. Podemos poner ejemplos de muchos tipos de aplicaciones de esta tecnología, pero una aplicación directa de IoT está en la domótica o automatización de viviendas. Por ejemplo, un sensor que detecta la presencia de agua en el suelo del baño como consecuencia de la rotura de la tubería, podría indicarle a la válvula del agua que se cerrase para impedir que sigua saliendo más agua, y todo ello de forma totalmente autónoma, distribuida y ubicua. Por otro lado, estos adelantos tecnológicos abren las puertas a la innovación en la forma de que los humanos pueden "relacionarse" con las máquinas y la misma relación entre las máquinas. La propuesta de esta tesis doctoral gira entorno a esta novedosa forma de mejorar la comunicación Interfaz Hombre Máquina (HMI) aplicando tecnologías de IoT.

II.2.3 Antecedentes

Desde julio de 2014, el *Institute of Electrical and Electronics Engineers* más conocido como IEEE, está diseñando una arquitectura global para la IoT de interés

[7] IPv6 o *Internet Protocol Version 6* es la nueva generación de protocolos para Internet

[8] www.nest.com (accesible el 8 de septiembre de 2015)

para múltiples sectores y tecnologías[9]. El objetivo final es crear una plataforma unificada y escalable. Más concretamente, el grupo de trabajo IEEE P2413[10], que comenzó sus actividad en julio de 2015, es el encargado de crear el marco de interoperabilidad entre dispositivos conectados y aplicaciones relacionadas en áreas como automatización del hogar, sistemas industriales y telemáticos, entre otras muchas en las que la IoT será una realidad en los próximos años. Aunque tales áreas presentan múltiples diferencias, la creación de un estándar permitiría compartir datos entre los sistemas IoT interconectados.

Otras iniciativas de estandarización en IoT incluyen los trabajos que se están realizando en el seno del *Industrial Internet Consortium*[11] en el lado de las empresas, y *AllJoyn*[12], más centrados en el consumidor. La intención del IEEE P2413 no es eclipsar la labor de estas organizaciones sino crear una arquitectura estándar que haga posible que los sistemas IoT de todos los sectores puedan trabajar juntos sobre una plataforma unificada y escalable. Tal plataforma común hará posible introducir economías de escala, reducir los precios del hardware e impulsar el desarrollo de todo un universo de aplicaciones futuras.

IEEE P2413 se centra en el diseño de los componentes básicos de los sistemas IoT comunes a todos los sectores. Entre otros objetivos, el grupo pretende convertir la información procedente de diferentes plataformas en objetos de datos comunes. Se espera que el estándar esté disponible en 2016[13].

En el grupo colaboran 23 fabricantes y organizaciones, entre los que se encuentran Cisco Systems, Huawei Technologies, General Electric, Oracle, Qualcomm y la ZigBee Alliance. Sus actividades se coordinan con grupos como ETSI (*European Telecommunications Standards Institute*[14]), ISO (*International Organization for Standardization*[15]) y *oneM2M*[16].

[9] www.standards.ieee.org/develop/corpchan/sfocus/index.html (accesible el 13 de octubre de 2015)
[10] www.standards.ieee.org/develop/project/2413.html (accesible el 8 de septiembre de 2015)
[11] www.industrialinternetconsortium.org (accesible el 8 de septiembre de 2015)
[12] www.developer.qualcomm.com (accesible el 8 de septiembre de 2015)
[13] www.grouper.ieee.org/groups/2413/Intro-to-IEEE-P2413.pdf (accesible el 8 de septiembre de 2015)
[14] www.etsi.org (accesible el 8 de septiembre de 2015)
[15] www.iso.org (accesible el 8 de septiembre de 2015)
[16] www.onem2m.org (accesible el 8 de septiembre de 2015)

II.2.3.1 Estándar abierto frente a protocolo cerrado (propietario)

En un reciente artículo[17], el periodista y escritor de tecnología Glyn Moody explicaba cómo el futuro del IoT pasa por los estándares abiertos, dando varias razones para ello. La primera de ellas es que habrá tal cantidad de empresas involucradas en esa red de billones de objetos interconectados que será imposible que ninguna de ellas consiga imponer un estándar propietario, es decir, una aplicación cerrada, de uso exclusivo de su creador y para uso exclusivo de sus clientes o productos. Y seguramente será no sólo un estándar abierto, sino que será de código abierto, ya que será igualmente imposible que ninguna compañía o grupo de ellas sea capaz de escribir todo el código necesario para los millones de diferentes sistemas que se irán uniendo en la Internet de las Cosas. Según el divulgador científico **Mike Simons[18]**, la única solución posible es que fabricantes, integradores, y usuarios puedan simplemente adaptar el código abierto a cada uno de sus dispositivos, como ya está pasando en el campo de los teléfonos y tabletas.

Otra razón de peso a favor del código abierto es la gran cantidad de dispositivos que ya actualmente llevan Linux embebido, por razones conocidas como el coste, la adaptabilidad o las herramientas disponibles. Sistemas embebidos y aplicaciones de código abierto serán obviamente más fácilmente ensamblables e integrables entre ellas. Empresas como IBM ya están en la misma dirección del código abierto. IBM junto con Eurotech, Sierra Wireless y la Fundación Eclipse han anunciado recientemente la formación de un nuevo grupo de trabajo para la definición e implementación de una plataforma de estándar abierto para las herramientas de desarrollo de las aplicaciones *Machine to Machine* (M2M), cuyo primer exponente es el proyecto ***Koneki*[19]** de la Fundación Eclipse.

Además, IBM y Eurotech han anunciado también que contribuirán al proyecto de código abierto de la Fundación Eclipse aportando software para acelerar y dar soporte al desarrollo de una nueva generación de dispositivos móviles inalámbricos, y que podría llegar a ser la base de un nuevo estándar abierto de conectividad e interoperabilidad móvil. Todo ello con garantías de ser

[17] www.tecnologiayterritorio.wordpress.com/2013/01/03/razones-por-las-que-internet-de-las-cosas-sera-un-estandar-abierto/ (accesible el 8 de septiembre de 2015) (accesible el 8 de septiembre de 2015)
[18] www.computerworlduk.com/author/mike-simons/ (accesible el 8 de septiembre de 2015)
[19] www.projects.eclipse.org/projects/technology.koneki (accesible el 8 de septiembre de 2015)

realmente abierto -"*restriction free*"- en palabras del Director ejecutivo de la Fundación Eclipse[20]: *"...la contribución con código abierto a las herramientas y protocolos de M2M ayudarán a reducir los tiempos de desarrollo y los costes, así como asegurará la interoperabilidad entre los sistemas a pesar de la rápida evolución de los sistemas".*

En palabras de este doctorando, esta última afirmación por parte del CEO de la Fundación Eclipse, podría estar en contraposición con la afirmación realizada por Tessel Renzenbrink por el que se afirma que *"...la IoT daría una dirección IPv6 a cada dispositivo para su conexión a Internet" (Dogan y Renzenbrink, 2012)*.

Para entender la trascendencia del anuncio es necesario conocer, al menos, un poco el protocolo MQTT (*Message Queue Telemetry Transport*) abierto y transferido por sus creadores a la Fundación Eclipse. En la siguiente sección se describe este protocolo.

II.2.4 Protocolo MQTT

II.2.4.1 Introducción

El protocolo **MQTT**[21] fue creado en 1999 por el Dr. Stanford-Clark y sus colegas en IBM y Eurotech. Originalmente este protocolo se diseñó con el objetivo de comunicar dispositivos con muy pocos recursos, con un consumo de ancho de banda reducido y para comunicaciones por líneas con una alta latencia, incluso con pérdida de señal. Desde entonces ha sido utilizado por empresas e industrias para las comunicaciones entre máquinas o sensores. Pero MQTT podía hacer mucho más: tiene el potencial de ser para Internet de las Cosas lo que el Hypertext Transfer Protocolo (HTTP) es para el Internet que conecta gente, haciendo potencialmente posible, que cada dispositivo en la red se comunique y comparta información con cualquier otro. En 2011 se donó el código a la fundación Eclipse y

[20] www.eclipse.org/users (accesible el 8 de septiembre de 2015)
[21] www.mqtt.org (accesible el 8 de septiembre de 2015)

en el año 2013 se presentó para ser estandarizado por la sociedad OASIS[22] (*Open Standards for the Information Society)*.

Según la investigación de la técnica realizada para esta tesis doctoral, la tendencia actual de los fabricantes es convertir el protocolo MQTT en un estándar que acelere la transmisión de información no sólo entre máquinas sino entre empresas, usuarios y consumidores, conectando toda clase de dispositivos y sistemas, de manera que sea posible compartir la información más fácilmente. En la actualidad MQTT es el protocolo de comunicación de los mensajes de Facebook, entornos móviles y de muchos otros sistemas como por ejemplo para aplicaciones donde los dispositivos de una casa o edificio se conectan a un servidor concentrador de datos. Es el caso de los productos de IBM MessageSight[23].

Estos movimientos tácticos en favor del código abierto de gigantes como IBM confirman claramente los argumentos señalados en el apartado anterior, y son desde mi punto de vista una muy buena noticia, pues permitirá escenarios de cooperación entre pequeñas empresas de desarrolladores y grandes marcas del sector. Pero sobre todo porque marca un camino claro de desarrollo corporativo para el Internet de las Cosas, que coinciden con los esfuerzos de los fabricantes de hardware de generar estándares abiertos de telecomunicaciones como el WiFi o el ZigBee, marcando también un horizonte para los proyectos destinados a las ciudades inteligentes o **smart cities**.

A pesar de todo lo anterior, aún hoy hay muchas compañías que tienen miedo escénico a la transparencia, temiendo que otros se apropien de su *Know How*, o a perder el control sobre sus productos o sus clientes. Por estas razones algunas empresas siguen trabajando con soluciones propietarias, siguiendo el anterior modelo de la propia IBM.

II.2.4.2 Arquitectura del protocolo

El protocolo MQTT protocolo cliente-servidor sobre el que se publican/suscriben mensajes por parte de los dispositivos interconectados. Está especialmente desarrollado para la comunicación *Machine to Machine* (*M2M) e IoT*,

[22] www.oasis-open.org (accesible el 8 de septiembre de 2015)
[23] www-03.ibm.com/software/products/es/messagesight (accesible el 13 de octubre 2015)

y está caracterizado principalmente por un código sencillo. Eso lo hace atractivo para este tipo de aplicaciones, las cuales por ejemplo en el caso de M2M son la telemetría y telecontrol y en el caso de IoT hablamos principalmente para sensorización en general. Los mensajes se ejecutan y transmiten sobre TCP-IP.

Una aplicación o dispositivo que utiliza MQTT como **cliente** es el encargado de establecer la conexión con el servidor en red. A su vez, el cliente puede "publicar" mensajes de aplicación que otros dispositivos clientes pueden estar interesados en recibir. Por otro lado, el cliente podría dar de baja o eliminar una solicitud de mensaje así como la desconexión del servidor.

El **servidor (bróker)** es un programa o dispositivo que actúa como intermediario entre los clientes que publican mensajes de aplicación y los clientes que han hecho de suscriptores. Un servidor acepta conexiones de red de los clientes, acepta mensajes de aplicación publicados por los clientes y tiene la capacidad para dar de baja a solicitudes de los clientes entre otras funciones.

Formato de control de paquetes

El control de paquetes sigue la siguiente estructura (tabla 1):

Encabezado fijo (*Fixed header*), presente en todos los paquetes de control MQTT
Encabezado variable (*Variable header*), presente en algunos paquetes de control MQTT
Información (*PayLoad*), presente en algunos paquetes de control MQTT

Tabla 1: Formato de control de paquetes del protocolo MQTT

Bit	7	6	5	4	3	2	1	0
Byte 1	*MQTT Control Packet type*				*Flags specific to each MQTT Control Packet type*			
Byte 2	*Remaining Length*							

Tabla 2: Formato del campo Fixer header dentro del protocolo MQTT

En la posición Byte 1 y bits 7-4, se define el **Tipo de Paquetes de Control** *(Control Packet type)* del protocolo MQTT. Estos tipos se seleccionan a partir de los valores que aparecen en la siguiente tabla 3:

Nombre	Valor	Dirección (flujo de información)	Descripción
Reserved	0	Prohibida	Reservado
Connect	1	Client to Server	Client request to connect to Server
Connack	2	Server to Client	Connect acknowledgment
Publish	3	Client to Server or Server to Client	Publish message
Puback	4	Client to Server or Server to Client	Publish acknowledgment
Pubrec	5	Client to Server or Server to Client	Publish received (assured delivery part 1)
Pubrel	6	Client to Server or Server to Client	Publish release (assured delivery part 2)
PubComp	7	Client to Server or Server to Client	Publish complete (assured delivery part 3)
Subscribe	8	Client to Server	Client subscribe request
Suback	9	Server to Client	Subscribe acknowledgment

Unsubscribe	10	Client to Server	Unsubscribe request
Unsuback	11	Server to Client	Unsubscribe acknowledgment
Pingreq	12	Client to Server	PING request
Pingresp	13	Server to Client	PING response
Disconnect	14	Client to Server	Client is disconnecting
Reserved	15	Forbidden	Reserved

Tabla 3: Tipos de paquetes de control del protocolo MQTT

En la posición Byte 1 y bits 3-0 de la cabecera fija, están contenidos los indicadores o ***Flags Bits*** específicos para cada tipo de control de paquetes MQTT. Cuando un bit de flag se marca como "Reservado", está reservado para uso futuro y deberá ser fijado al valor que aparece en la tabla 4. Si se reciben flags válidas, el receptor debe cerrar la conexión de red.

Control Packet	**Fixed header flags**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Connet	Reserved	0	0	0	0
Connack	Reserved	0	0	0	0
Publish	Used in MQTT	DUP1	QoS2	QoS2	RETAIN3
Puback	Reserved	0	0	0	0
Pubrec	Reserved	0	0	0	0
Pubrel	Reserved	0	0	1	0
Pubcomp	Reserved	0	0	0	0
Subscribe	Reserved	0	0	1	0
Suback	Reserved	0	0	0	0
Unsubscribe	Reserved	0	0	1	0

Unsuback	Reserved	0	0	0	0
Pingreq	Reserved	0	0	0	0
Pingresp	Reserved	0	0	0	0
Disconnect	Reserved	0	0	0	0

Tabla 4: Descripción de las flags del protocolo MQTT

DUP1 = Duplicate delivery of a PUBLISH Control Packet

QoS2= PUBLISH Quality of Service

RETAIN3 = PUBLISH Retain flag

Remaining Length es el número de bytes restantes en el paquete actual, incluyendo los datos de la cabecera variable y la carga útil. *Remaining Length* no incluye los bytes utilizados para codificar el restante Longitud. Para su codificación se emplea un esquema de codificación de longitud variable que utiliza un solo byte para valores de hasta 127.

Como ejemplo se expone parte de código que hemos implementado en una plataforma Arduino al objeto de comprobar la viabilidad de éste para gestionar el tráfico de mensajes suscritos en MQTT y un PC. Básicamente el sistema consiste en un cliente[24] en Java para la emisión y recepción de mensajes empleando el protocolo de mensajería MQTT. La idea en enviar la información recogida de un sensor de intensidad luminosa y encender o apagar un dispositivo lumínico:

```
public void setup() {
// Definimos la etiqueta del cliente MQTT= im4Things
MQTT im4Things = new MQTT();
mqtt.setHost( "tcp://192.0.0.12:8080" );
}
public void testJustPublish( String intensidad_luminica ) throws Exception {
```

[24] www.oracle.com/technetwork/articles/java/javaclient-484666.html (accesible el 8 de septiembre de 2015)

```
        // broker del PC
        BlockingConnection connection = mqtt.blockingConnection();
        connection.connect();
        // publicamos el ON/OFF en función de la intensidad lumínica del sensor
        if ( Integer.parseInt( intensidad_luminica ) < 100 ) ) {
    connection.publish( "ArduinoReceiveTopic",
    "ON".getBytes(), QoS.AT_LEAST_ONCE, false );
        }
        else {
    connection.publish( "ArduinoReceiveTopic",
    "OFF".getBytes(), QoS.AT_LEAST_ONCE, false );
        }
        connection.disconnect();
    }
```

Algoritmo 1. Algoritmo para el intercambio de mensajes bajo el protocolo MQTT entre Arduino y un PC

II.2.4.3 Anotación final

Después de lo visto hasta el momento, el protocolo MQTT ofrece tres ventajas primordiales. La primera es que es un protocolo simple que está especialmente diseñado para dispositivos con pocos recursos de gestión de datos, bajo ancho de banda y redes de alta latencia lo que lo hace ideal para su uso en el ecosistema de la IoT. Por otro lado está su arquitectura de trabajo pensado en la generación de topics (ver figura 1) y mensajería de publicación-suscripción de mensajes lo que lo hace ideal para el intercambio de notificaciones entre el servidor de mensajería (acciones de control) y el sensor que se suscribe a un topic. Por último, una ventaja también destacada de este protocolo es que ofrece tres calidades de servicio en la entrega del mensaje:

- Nivel *"fire-and-forget"*, a lo sumo el mensaje publicado se recibe 1 vez. Ejemplo, mensaje emitido por un sensor ambiente en el que no importa si una lectura individual se pierde ya que instantes después de publicará un nuevo valor.
- Nivel "*no confinable*", donde los mensajes se aseguran que van a llegar, pero puede ocurrir alguno de ello esté duplicado.
- Nivel "*at least once*", donde el mensaje se asegura que llegará exactamente una vez. Este nivel de utilización es necesario en sistemas de facturación de billetes, contaje, etc.

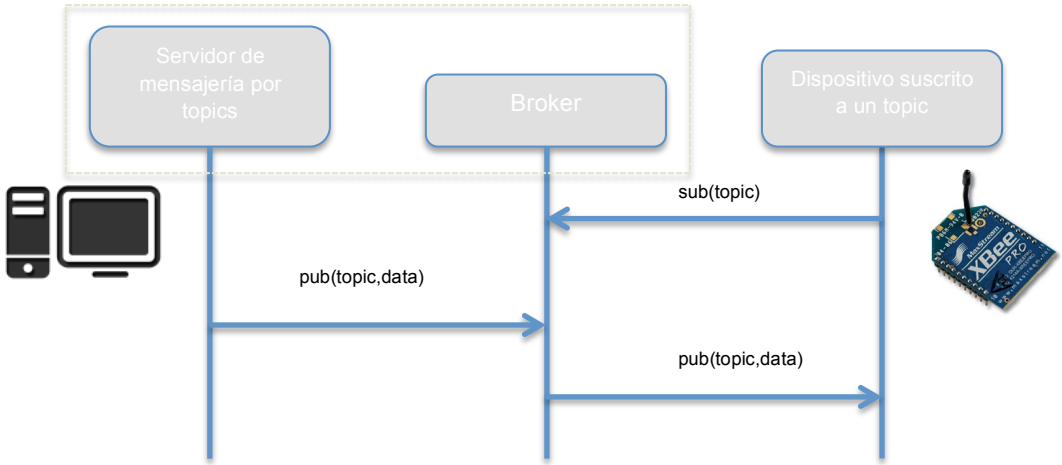


Figura 1. Secuencia MQTT de intercambio de mensajería

Como importante desventaja para su uso en la IoT, es que actualmente necesita de un sistema conversor al estilo de "centralita" para llegar hasta Internet. Por ejemplo, los datos MQTT de un sensor de temperatura de una vivienda necesita un dispositivo que transforme los datos inalámbricos (radiofrecuencia) al formato Ethernet y después conectarse al roúter de la vivienda para que los datos sean monitorización a través de Internet. Y por otro lado, desde el punto de vista de la seguridad, este protocolo no es verdaderamente seguro para aplicaciones de IoT, tanto como para no confiar operaciones internas domóticas, ya que como se ha comentado anteriormente está diseñado para la comunicación máquina-máquina y el tráfico de datos se limita al estado de ésta o valor leído por un sensor, donde lo importante aquí es la información del valor con la información justa del mensaje.

Todos estos problemas han supuesto un punto de inflexión a la hora de seleccionar la tecnología para la realización de esta tesis doctoral. La solución propuesta en esta tesis es la utilización de un servicio de mensajería para poder comunicar a todos los dispositivos. Más concretamente, se pretende desarrollar un sistema que permita dotar de inteligencia a los dispositivos para que puedan comunicarse con los usuarios y entre sí a través de diálogo en lenguaje natural. Dentro de la mensajería instantánea existe el protocolo XMPP (*Extensible Messaging and Presence Protocol*) que puede dar respuesta a algunos de los interrogantes antes descritos como por ejemplo los relacionados con el formato del mensaje. XMPP es un protocolo de mensajería instantánea que define cuidadosamente todos los formatos de mensaje. En el caso de MQTT, es un protocolo de transporte básicamente en el que el contenido del mensaje está formado por los topics a cual dirigir el mensaje.

Por el lado de la seguridad, el protocolo XMPP va un paso más allá. La fundación de software americana Jabber [25] actualmente denominada *XMPP Standards Foundation* [26] publicó en el año 2004 el informe *"Extensible Messaging and Presence Protocol (XMPP): Core*[27]*"*, en el que se detalla la seguridad del protocolo. De este documento se extrae que, respecto a la identidad de la fuente, los usuarios de XMPP deben autenticarse en el servidor host, lo que aumenta la seguridad de suplantación. Por otro lado, la información que sigue el camino cliente a servidor está encriptada (cifrada) de dos forma. El primer cifrado se realiza durante el establecimiento de la conexión y la autenticación. Esta técnica se conoce con las siglas SASL[28] que es un estándar con amplia difusión para la seguridad del tráfico de mensajes de correo electrónico.

Una vez establecida la conexión, a todas las transmisiones entre el cliente y el servidor se le añade seguridad en la capa de transporte por medio del cifrado TLS *(Transport Layer Security)* sucesor del cifrado SSL *(Secure Sockets Layer).* Estas propiedades hacen que XMPP sea seguro tanto en la fase de establecimiento de la conexión y la fase de comunicación. De igual forma los principios de seguridad

[25] www.xmpp.org/xsf/press/2007-01-16.shtml (accesible el 8 de septiembre de 2015)
[26] www.xmpp.org (accesible el 8 de septiembre de 2015)
[27] www.xmpp.org/rfcs/rfc3920.html - tls (accesible el 8 de septiembre de 2015)
[28] https://www.ietf.org/rfc/rfc2554.txt (accesible el 8 de septiembre de 2015)

utilizados para asegurar las conexiones cliente a servidor se extienden para las conexiones de servidor a servidor relacionados bajo un dominio también conocido como federación de servidores[29].

II.2.5 Protocolo XMPP

II.2.5.1 Introducción

El protocolo XMPP es un protocolo abierto y extensible basado en el lenguaje de programación XML[30]. El invento de este protocolo se le asigna a Jeremie Miller, estudiante de ingeniería eléctrica y computadores en la Iowa State University of Science and Technology el cual creo el primer servidor XMPP en el año 1988 para el envío de mensajería instantánea. Entre los años 1999 y 2000 este protocolo fue mejorado y adaptado para el envío de una combinación de VoIP y video bajo el denominado proyecto Jabberd[31]. Con esta mejora del protocolo XMPP se consigue, además de la mensajería instantánea, ser un protocolo de señalización de transferencia *(Moreno et. al, 2001)*. Los protocolos de señalización son utilizados para el establecimiento de sesiones en una red IP.

La popularidad del XMPP entre la comunidad informática se hizo patente en el año 2006, fecha en la cual se desarrollaron nuevas extensiones del protocolo para funcionalidades personalizadas incluyendo la comunicación entre servidores por medio de éste. En general el protocolo está implementado para una arquitectura cliente-servidor descentralizada pero con la extensión comentada XMPP para establecer una comunicación directa, de extremo a extremo (también conocida como peer-to-peer (P2P)) y entre los clientes. En la figura 2 se muestra la arquitectura XMPP para comunicación entre servidores y clientes.

[29] www.technet.microsoft.com/es-es/library/cc731388(v=ws.10).aspx (accesible el 8 de septiembre de 2015)
[30] www.xml.com (accesible el 8 de septiembre de 2015)
[31] www.jabberd.org (accesible el 8 de septiembre de 2015)

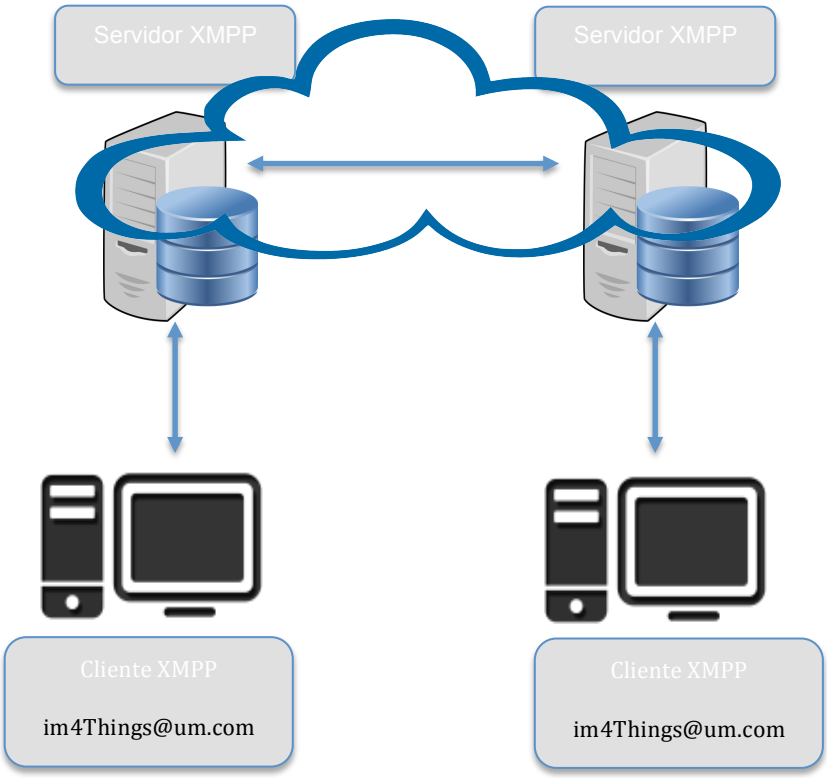


Figura 2. Arquitectura XMPP para comunicación entre servidores y clientes

En la arquitectura que se presenta en la anterior figura cada "entidad" bajo el protocolo XMPP se identifica por medio de un identificador JabberID[32] (JID). En XMPP, este identificador tiene el formato de correo electrónico. En nuestro proyecto hemos empleado el JID im4Things@um.com. Esto supone una gran ventaja se emplea estos identificadores en vez de la dirección IP, más fácil de recordar.

II.2.5.2 Principales características del protocolo

El protocolo XMPP es **abierto, gratuito y público**. Su configuración lo hace de fácil comprensión, hecho que permite contar con múltiples implementaciones entre clientes, servidores, componentes de servidores y librerías de código. Estas características fomentan el desarrollo del IoT porque los desarrolladores de código abierto pueden seguir refinando y realizando nuevas extensiones que lo mejoren y se adapte de forma escalonada a la demanda mundial.

[32] www.xmpp.org/search/JabberID (accesible el 8 de septiembre de 2015)

El protocolo XMPP es **estándar**. Así lo reconoció la asociación estadounidense *Internet Engineering Task Force* (IETF[33]) que desarrolla y promueve estándares de Internet (RFC 6120 de marzo de 2011[34]). Por lo tanto, XMPP es un protocolo estándar como una tecnología de mensajería instantánea e información entre dispositivos informáticos.

El protocolo XMPP está **testado a nivel mundial**. Es una tecnología muy estable para aplicaciones en Internet. En la actualidad existen miles de servidores utilizando este protocolo en Internet, y millones de personas utilizándolo para mensajería instantánea para servicios públicos como Google Talk [35] e implementaciones en organizaciones.

El protocolo XMPP es **descentralizado.** Emplea una arquitectura similar a la empleada en el correo electrónico por lo que su uso queda garantizado en servidores propios.

El protocolo XMPP es **seguro.** Para el campo de actuación de esta tesis doctoral esta característica es primordial. Cualquier servidor XMPP puede ser aislado de la red pública (como en una Intranet) y utilizar seguridad adicional en el formato SASL y TLS.

El protocolo XMPP es **extensible**. Esta cualidad ha sido fundamental a la hora de elegir este protocolo para la realización de esta tesis doctoral. Esto es así porque como se ha indicado en el punto anterior, el protocolo fue creado en 1988 y le han seguido diferentes extensiones de uso. Parte del trabajo de este doctorando ha consistido en realizar nuevas funcionalidades personalizadas sobre el núcleo del protocolo, es decir, nuevas extensión del protocolo XMPP que permitan el envío de mensajería instantánea a dispositivos con base de inteligencia artificial (Bots) y que se definen en profundidad en el capítulo III.

El protocolo XMPP es **flexible**. Se ha partido de las aplicaciones originales de XMPP (de mensajería y presencia) hasta llegar a administración de redes, sindicalización de contenidos, herramientas de colaboración, compartimiento de

[33] www.ietf.org (accesible el 8 de septiembre de 2015)
[34] www.tools.ietf.org/html/rfc6120 (accesible el 8 de septiembre de 2015)
[35] www.google-talk.uptodown.com (accesible el 8 de septiembre de 2015)

archivos, juegos, monitorización de sistemas remotos, servicios web, computación en la nube, etc.

En definitiva, el protocolo XMPP está siendo utilizado por un amplio rango de empresas en diferentes proyectos. Se puede entonces afirmar que este protocolo es **diverso**. Un aspecto importante que recientemente están optando los usuarios es la construcción y despliegue de aplicaciones en tiempo real y servicios[36].

Como ventaja a todo esto cabe destacar la **cooperación** entre desarrolladores y empresas. Esto ha hecho que se pase de los sistemas de mensajería a ofrecer funciones adicionales al intercambio de mensajes, como transferencia de archivos, listas de contactos, conversaciones simultáneas. Desde el punto de vista comercial, todas estas funciones pueden ser requeridas por pequeñas empresas y corporaciones.

Por otro lado está la mensajería móvil, a través de la cual podemos transferir la experiencia de mensajería de escritorio a dispositivos móviles con acceso a Internet. Posibilita las redes de contactos para la comunicación entre usuarios a través de salas de conversación para mensajería instantánea entre todos los miembros de una red.

Pero el protocolo XMPP tiene algunas desventajas entre las que cabe destacar las siguientes:

- **Sobrecarga de datos** de presencia en servidores debido al tráfico entre ellos (70%) y en torno al 60% en transmisiones redundantes.
- **Escalabilidad.** El protocolo tiene este problema, quizás asociado al tráfico referido en el punto anterior, y de igual forma el problema se deriva a los servicios de Chatroom[37] en tiempo real y de suscripción.
- **Ausencia de datos binarios**. Cada mensaje en este protocolo está formado por 64 bits. Esto obliga a XMPP a codificarse como un único y largo documento XML, lo que hace imposible entregar datos binarios sin modificar. Como solución a esta desventaja, las transferencias de archivos se han solucionado usando otros protocolos como HTTP.

[36] www.hipertextual.com/archivo/2014/07/protocolo-xmpp/ (accesible el 8 de septiembre de 2015)
[37] www.en.wikipedia.org/wiki/Chat_room (accesible el 8 de septiembre de 2015)

II.2.5.3 Clientes XMPP

Salvatore Gaglio y **Giuseppe Lo. Re** de la Universidad de Palermo (Italia) hacen referencia en su libro "*Advances onto the Internet of Things. How Ontologies Make the Internet of Things Meaningful*"*(Gaglio y Lo Re, 2014)*, al empleo del protocolo XMPP como mecanismo de **seguridad** en el tráfico de información del sistema ERMI (*External Remote Method Invoker*[38]) en la arquitectura CLEVER para permitir la interacción de componentes en la nube. Este tráfico se lleva a cabo entre *clusters*[39] y *hostging*[40]. En palabras de los autores "…*data confidentiality, integrity, and non-repudiation*".

Un **cliente** XMPP es una aplicación que permite conectarse a un servidor XMPP utilizado para ello la mensajería instantánea. Existen muchos clientes para diferentes sistemas operativos e incluso para dispositivos. Aquí se van a describir el conjunto de programas que bajo el concepto **CHAT** emplean el protocolo XMPP para el tráfico de mensajería instantánea. Cabe destacar los siguientes:

II.2.5.3.1 Clientes XMPP para Windows

Psi [41] . Es un servicio de mensajería instantánea orientada al trabajo colaborativo. Ofrece una serie de ventajas respecto a otras herramientas de comunicación como el e-mail y el Teléfono como son la inmediatez de uso ya que ofrece una interface de usuario muy sencilla. Por otro lado se caracteriza por la alta velocidad de envío-respuesta del mensaje e indica el estado del usuario (ausente, reunido, ocupado…). Soporta la codificación internacional de caracteres Unicode[42].

Gajim [43] . Servicio de mensajería instantánea multiplataforma para Jabber/XMPP.

---

[38] www.dsnet.tu-plovdiv.bg (accesible el 8 de septiembre de 2015)
[39] Los clústeres de servidores conectan varios servidores dedicados a través de una LAN, por lo que ofrecen métodos rentables para alcanzar niveles más altos de rendimiento y disponibilidad.
[40] Éstos servicios funcionan ofreciendo soluciones en la nube por medio de servidores web de altas prestaciones utilizando conexiones de alta velocidad, seguridad, mantenimiento ante fallos, etc. Ejemplo: Amazon, Google, etc.
[41] www.psi-im.org (accesible el 8 de septiembre de 2015)
[42] www.unicode.org/consortium/consort.html (accesible el 8 de septiembre de 2015)
[43] www.gajim.org (accesible el 8 de septiembre de 2015)

Jabbim[44]. Servicio gratuito de mensajería instantánea. No consigue llegar al "tiempo real".

Miranda IM[45]**.** Servicio cliente de mensajería instantánea multiprotocolo. Se caracteriza por su reducido uso de recursos del sistema frente a su extremada velocidad de respuesta. A diferencia de otros servicios, Miranda IM no requiere instalación ya que está basado en plugin[46]. En concreto los desarrolladores de Miranda has puesto al servicio de los usuarios 350 plugin distintitos para su uso y personalización. Es flexible y adaptable por parte del usuario.

II.2.5.3.2 Clientes XMPP para GNU/Linux

Los clientes XMPP Psi, Gajim y Jabbim están disponibles también para Linux.

Jabber.el[47]**.** Servicio para el intercambio en tiempo real de mensajes y presencia entre dos puntos en Internet. Ofrece un valor añadido de gran utilidad y es que en Jabber puedes estar varias veces en la misma cuenta, simultáneamente, incluso con el mismo cliente y en el mismo ordenador. Por ejemplo permite entrar a un ordenador desde distintas cuentas en lugares distintos.

Ayttm[48]**.** Servicio de mensajería instantánea caracterizado por una fácil configuración y uso gracias a una interface intuitiva. Por otro lado soporta múltiple protocolos utilizando para ello una interface común lo cual favorece su utilización por parte del usuario. Eso le permite trabajar con los protocolos propios de Yahoo!, MSN, Jabber, IRC, AIM e ICQ.

Mcabber[49]**.** Quizás unos de los primeros chats multiusuario para el envío de mensajería instantánea muy similares a los famosos canales IRC[50]. El cliente XMPP Mcabber es una instancia instalada en el programa de mensajería utilizado por el usuario. Ello permite la ejecución de comandos en el ordenador remoto desde el menú del programa de mensajería utilizado, es decir, un comando en el ordenador

[44] www.jabbim.com (accesible el 8 de septiembre de 2015)
[45] www.miranda-im.org (accesible el 8 de septiembre de 2015)
[46] Software de complemento que se instala en un programa para realizar funciones adicionales.

[47] www.emacs-jabber.sourceforge.net (accesible el 8 de septiembre de 2015)
[48] www.ayttm.sourceforge.net (accesible el 8 de septiembre de 2015)
[49] www.mcabber.com (accesible el 8 de septiembre de 2015)
[50] Canal o red de comunicaciones en tiempo real multiusuario para el intercambio de mensajes

remoto puede ser ejecutado solo enviar un mensaje desde el ordenador fuente. Esto entraña el peligro inherente de la mal intención de ejecutar un programa sin el consentimiento del usuario del ordenador remoto.

Gossip[51]**.** Servicio de mensajería instantánea utilizado por usuarios de GNOME[52] para contactos entre usuarios. Gnome es un paquete de productos software realizados por una comunidad internacional de desarrolladores sin ánimo de lucro. Entre los servicios que ofrece son el de documentación, diseño de interfaces, traductores, paquetes de código de aplicaciones diversas y otros productos que pueden ser consultados y descargados desde la web referenciada. El software de mensajería instantánea Gossip no es desarrollada directamente por la comunidad Gnome pero la aplicación está desarrollada para coger los contactos del servicios agenda de contactos de Gnome y permitir en envío de mensajería instantánea entre dichos contactos.

Bitlbee[53]**.** Servicio de mensajería instantánea para clientes de un canal IRC. Lo que ofrece este servicio es reunir a los usuarios de un canal IRC y formar grupos de usuarios para el intercambio de mensajería instantánea.

GNU Freetalk[54]**.** Freetalk es una consola basada en cliente de chat de Google Hangout[55] y otros servidores XMPP. Con esta consola los clientes de chat pueden intercambiar además de mensajería instantánea archivos multimedia. Para mejorar la experiencia de uso, la consola dispone de un motor contextual de auto-completado de nombres de usuarios, comandos y palabras en inglés. Es personalizable.

II.2.5.3.3 Clientes XMPP para Mac OS

Los clientes XMPP Psi y Ayttm están disponibles también para Mac OS.

Tkabber[56]**.** Cliente XMPP de código abierto que permite establecer llamadas directas entre PCs a través del servicio voz IP (VoIP). Para la compresión del flujo de la conversación, es decir, para la compresión de la voz, este cliente emplea el

[51] www.launchpad.net/gossip (accesible el 8 de septiembre de 2015)
[52] www.gnome.org (accesible el 8 de septiembre de 2015)
[53] www.bitlbee.org/main.php/news.r.html (accesible el 8 de septiembre de 2015)
[54] www.gnufreetalk.github.io (accesible el 8 de septiembre de 2015)
[55] www.hangouts.google.com (accesible el 8 de septiembre de 2015)
[56] www.tkabber.jabber.ru (accesible el 8 de septiembre de 2015)

códec speex, que es una tipo de compresión de audio de fuente abierta y libre de patentes, y creado específicamente para la compresión de la voz. Además, los usuarios de este cliente pueden intercambie mensajería instantánea.

Coccinella[57]. Plataforma de comunicación libre basado en pizarra que permite la comunicación gráfica colaborativa entre dos usuarios. Además, permite la comunicación por voz (simultáneamente que la gráfica) lo que potencia aún más esta útil herramienta de comunicación. Además los usuarios pueden intercambiar archivos y documentos todo ello multiusuario.

Pidgin[58]. Plataforma chat que permite conectarse a las cuentas de usuario en múltiples redes de chat simultáneamente. Esto significa que permite estar chateando con usuarios en MSN, hablando con otro usuario en Google Talk, e iniciar una sesión de chat Yahoo, todo al mismo tiempo.

iChat[59]. Cliente de mensajería instantánea anteriormente conocido como iChat. Puede conectarse a las redes AIM y XMPP, y establecer comunicaciones entre los usuarios participantes. De igual forma puede establecer la comunicación entre usuarios de Mac dentro de la misma red local.

CenterIM[60]. Servicio de mensajería instantánea multiprotocolo para Linux. Se caracteriza por ser compatible con la mayoría de protocolos de mensajería instantánea ampliamente utilizados, incluyendo AIM, ICQ, IRC, MSN. El funcionamiento de este cliente se basa en crear una lista de usuarios tipo correo electrónico para establecer el intercambio de mensajería instantánea, direcciones URL, mensajes SMS y correo electrónico.

II.2.5.3.4 Clientes XMPP para teléfonos móviles

Jabber Mix Client (JME)[61]. Cliente de mensajería instantánea para el protocolo XMPP para terminales móviles que ofrece al usuario, además del servicio de mensajería instantánea, otros servicios como es el envío de información entre usuarios relativa a la ubicación del terminal. Trabaja con agenda de contactos y la

[57] www.coccinella.im (accesible el 8 de septiembre de 2015)
[58] https://developer.pidgin.im/wiki/Using%20Pidgin#UsingPidgin (accesible el 8 de septiembre de 2015)
[59] www.macupdate.com (accesible el 8 de septiembre de 2015)
[60] http://www.centerim.org/index.php/Main_Page (accesible el 8 de septiembre de 2015)
[61] http://jabbermixclient.sourceforge.net (accesible el 8 de septiembre de 2015)

comunicación entre ellos se lleva a cabo por medio de una interface fácil de usar e intuitiva. El código de programación está implementado a partir de las librerías disponibles en Java lo cual permite a los desarrolladores realizar r

Talkonaut[62]. Este cliente ofrece a los usuarios la posibilidad de intercambiar mensajería instantánea para terminales móviles y otros servicios como son el intercambio de mensajes de audio pregrabados denominados Jingle Audio. Para ello emplea la tecnología denominada "*peer-to-peer*" que consiste en el envío de mensajes de voz y otros sonidos pregrabados. Para ello este cliente ofrece al usuario dirigirse a sitios webs donde poder descargarse algunos de los estos sonidos pregrabados. Sirva de ejemplo el siguiente link: http://audiojungle.net/?osr=tn

Por último, este cliente es multiusuario para realizar grupos de usuarios e intercambiar mensajes entre ellos y de igual forma compartir archivos, imágenes y avatares basados vCard. Este cliente permite realizar llamadas de voz a teléfonos fijos o teléfonos móviles a través de la infraestructura GTalk2VoIP[63].

Hasta aquí se ha hecho una exposición de los servicios de mensajería instantánea destinados a la comunicación entre usuarios por medio del empleo de consolas en formato interface y el protocolo de comunicaciones XMPP. Estos sistemas están muy probados y testados y son comerciales desde hace mucho tiempo por lo que se da por sentada su viabilidad. Por otro lado, la "masa crítica" de usuarios es enorme por lo que el público objetivo que utiliza estos sistemas de comunicación es muy grande y familiarizado con el entorno.

Como se ha comentado anteriormente, este protocolo se diseñó para sistemas de mensajería instantánea que usuarios humanos pueden utilizar para comunicarse entre sí. Para poder aplicar este protocolo en el mundo de la IoT será necesario dotar a los dispositivos de inteligencia y de un sistema de diálogo que permitan su interacción con humanos.

En el siguiente apartado se explica el concepto de sistema de diálogo y agente conversacional que sirve como base al desarrollo de esta tesis doctoral.

[62] www.talkonaut.com (accesible el 8 de septiembre de 2015)
[63] www.gtalk2voip.com (accesible el 8 de septiembre de 2015)

II.3 SISTEMAS DE DIÁLOGO

II.3.1 Introducción

Según la RAE, podemos definir diálogo como "*Plática entre dos o más personas, que alternativamente manifiestan sus ideas o afectos*". De esta definición podemos deducir que el diálogo en lenguaje natural es la forma que tienen los humanos de comunicarse y relacionarse todos los días.

Los sistemas de diálogo, también llamados sistemas conversacionales, tratan de imitar el comportamiento inteligente de comunicación de los seres humanos para desarrollar programas de ordenador que permitan interactuar con usuarios humanos por medio de lenguaje natural.

II.3.2 Sistemas de diálogo y agentes conversacionales

Desde la década de los 90, se han desarrollado distintos sistemas de diálogo comerciales para apoyar el acceso a bases de datos para que un usuario pueda buscar información utilizando expresiones en lenguaje natural (*Jokinen y McTear, 2009)*. En los últimos años, diversas empresas e instituciones públicas y privadas usan estos sistemas con la finalidad de proporcionar información y otros servicios de forma automática, principalmente de forma oral a través del teléfono *(López-Cózar et. al, 2005).*

Tradicionalmente, Los sistemas de diálogo han sido orales (*Lison y Meena, 2014)*. En estos sistemas de diálogo oral (Spoken Dialogue Systems, SDSs) el canal de comunicación es mediante habla en lenguaje natural.

En la literatura pueden encontrarse numerosas referencias sobre sistemas de diálogo hablado para permitir realizar tareas muy definidas como consultas en sistemas de información *(Aust et. al, 1994), (Bonafonte et. al, 2000)* o *(Zue et. al, 2000).*

Con la irrupción de IoT, los sistemas de diálogo están cobrando importancia para la interacción con dispositivos *(Turunen et. al, 2015)*. Más concretamente, existen multitud de trabajos que pretenden monitorizar y controlar una vivienda mediante

órdenes en lenguaje natural. Por ejemplo, el sistema INVOX *(Vivancos et. al, 2009)* permite el control centralizado de dispositivos domóticos mediante el uso de comandos hablados en lenguaje natural. El sistema principal de INVOX es un pequeño ordenador que permite transformar las órdenes habladas en comandos KNX. El sistema G.H.O.S.T *(Vanus et. al, 2015)* también hace uso del estándar KNX para poder comunicarse con todos los dispositivos domóticos. En este sistema se pueden configurar las órdenes predefinidas, pero no tiene un control de órdenes en lenguaje natural como el que proporciona el sistema INVOX.

Recientemente el trabajo presentado en *(Katsamanis et. al, 2014)* permite mejorar la precisión del sistema utilizando filtros de ruido acústico y reconociendo comandos a partir de palabras claves en micrófonos de ambiente. En este trabajo, la comprensión de las órdenes se realiza de manera centralizada por un dispositivo que es el que traduce las órdenes en lenguaje natural y se comunica con todos los aparatos de la casa.

Por otro lado, el sistema *Mayordomo (Espejo et. al, 2010)* hace uso de tecnologías de diálogo y no sólo de reconocimiento de comandos que permita además de controlar los dispositivos, poder consultar su estado. Para ello, implementa un módulo de gestión del diálogo que detecta si se está ejecutando una orden, o bien si por el contrario, se está consultando el estado de algún dispositivo en la instalación domótica. El control de estas instalaciones también se realiza de manera centralizada en este sistema.

Por otro lado, los sistemas de diálogo se han aplicado ampliamente en el desarrollo de agentes conversacionales o asistentes virtuales *(Kopp et. al, 2003).* En los últimos años ha crecido el interés de integrar estos asistentes virtuales con capacidad de comunicarse en lenguaje natural en web corporativas como es el caso de RENFE o IKEA. Las principales facilidades que pretenden proveer estos asistentes son facilitar la navegación por páginas de productos o recuperar páginas con determinados productos. Muchos de estos asistentes virtuales se implementan mediante aplicaciones del tipo "chatter bots", como ocurre en los casos de ALICE[64], Hal, Max y NICOLE que no realizan ningún tipo de procesamiento del lenguaje y

[64] www.alicebot.org (accesible el 8 de septiembre de 2015)

para generar las respuestas disponen de un gran conjunto de patrones con respuestas predefinidas asociadas *(Aguado et. al, 2002)*. Más concretamente, para ALICE se utiliza el lenguaje AIML que se explica en más detalle en el siguiente apartado.

II.3.3 Lenguaje AIML

El lenguaje AIML es un lenguaje de marcas basado en XML desarrollado por el **Dr. Richard Wallace** junto con la comunidad de código abierto Alicebot. Dicho desarrollo se llevó a cabo entre los años 1995 y 2000 y estuvo dedicado a la creación de la primera entidad **ChatBot** informática de lenguaje artificial online también conocido como proyecto A.L.I.C.E., sus siglas en inglés de *Artificial Linguistic Internet Computer Entity Chatterbot*.

AIML es un lenguaje de programación de código abierto en el que los desarrolladores han podido aportar al mundo numerosos Bots basados en la implementación original del programa y de la base de conocimiento AIML. El desarrollo de este lenguaje continúa, tal es así que en el otoño de 2004 se creó una nueva versión del conjunto ALICE AIML. A día de hoy existen hosting de bots online para el alojamiento de bots implementados mediante este lenguaje. Ejemplos de hosting especializados en alojar bots pueden ser Budabot[65], Pandorabots[66], Botlibre[67].

II.3.3.1 Elementos estructurales de AIML

Los elementos estructurales más importantes de AIML son:

- **Categorías:** unidades fundamentales del conocimiento basabas en al menos dos o más elementos que son los componentes patrón o *pattern* y plantilla o *template*, que normalmente se codifican en ese orden.
- **Patrones:** basados en comparación de patrones o *pattern-matching*. Esta técnica reduce o elimina en algún caso la necesidad del empleo de redes

[65] www.budabot.com (accesible el 8 de septiembre de 2015)
[66] www.pandorabots.com/static/html/ (accesible el 8 de septiembre de 2015)
[67] www.botlibre.com (accesible el 8 de septiembre de 2015)

neuronales para que la máquina dé respuesta a las órdenes recibidas del usuario en lenguaje natural. AIML permite al programador realizar la búsqueda de **patrones** o palabras clave, que el usuario haya introducido para que la máquina responda de acuerdo a lo que se le haya preguntado.

- **Plantillas (*Templates*):** cuando un patrón coincide (o el que más concuerda) entonces la respuesta de la máquina añade una plantilla específica de la categoría. Esta plantilla puede contener otros componentes de AIML, que permiten personalizar dicha respuesta.

II.3.3.2 Agentes Conversacionales Virtuales en AIML

En el año 2013 los investigadores **Tetsuo Shinozaki, Yukiko Yamamoto** y **Setsuo Tsuruta** de la Universidad de Denki (Tokyo), publican el artículo ***"Context-based counselor agent for software development ecosystem"*** *(Shinozaki et. al, 2013)* donde avanzan en el desarrollo de software para la creación de un Agente Virtual Conversacional capaz de desplazar la intervención humana en futuros ecosistemas sociales.

En este trabajo se pone de manifiesto que el desarrollo de software para futuros ecosistemas sociales es una apuesta segura y a la vez un reto tecnológico que debe afrontarse para obtener sistemas sociales e inteligentes diseñados para máquinas y objetos los cuales en un futuro se integran en la sociedad de las comunicaciones como un consumidor de recursos más.

El Agente Virtual Conversacional (AVC) propuesto por estos investigadores es un chatterbot formado por un módulo analizador sintáctico que analiza la entrada del usuario, un transformador para generar la salida del chatterbot basado en la entrada del usuario anterior(s) y una base de conocimiento que permite guiar el diálogo. También se apunta de la existencia de chatterbots, para propósitos específicos y otros que pueden conversar con los usuarios (humanos) en una amplia gama de temas.

En general las técnicas de procesamiento empleadas más eficaces se pueden resumir en dos: incluir tecnologías avanzadas de procesamiento de lenguaje

natural y las técnicas "aprender-por-enseñanza" propuesta por de la comunidad *e-learning (Arena y Adell, 2009).*

En este método los usuarios en primer lugar enseñan a un agente (*Brain*) sobre cómo resolver algún problema. Seguidamente el agente realiza preguntas con inicio del tipo "qué", "cómo", etc., sobre la solución del problema. De esta forma permite identificar los conocimientos necesarios para manejar el problema de manera más efectiva. Este método es eficaz en su estricta forma de funcionamiento con la salvedad “emocional” por parte de los usuarios.

Es en el año 2002 cuando la comunidad e-learning introduce por primera vez el concepto de *scaffolding (Wu et. al, 2002) o (Brian, 2002).*La idea de este método se basa en que los usuarios están expuestos inicialmente a sugerencias generados de forma automática, independientemente de su relevancia para el problema en cuestión y de forma recurrente. A medida que el grado de coincidencia aumenta se van reduciendo gradualmente el número *scaffolding* y como resultado los usuarios dejan de considerar de forma autónoma las soluciones dadas por el método.

Según estos investigadores, la técnica de *scaffolding* funciona bien cuando los problemas son claros y están bien planteados.

El usuario escribe una frase la cual se convierte en el lenguaje abstracto AIML. Esta traducción se utiliza para analizar el contenido de la sentencia y hacer peticiones a través de un motor de búsqueda en una base de conocimiento. La respuesta en lenguaje natural se genera a través de un lenguaje abstracto, también AIML, que será presentado al usuario.

El sistema está formado por una aplicación móvil App para Android, un servicio de Agente Virtual (AV) basado en el lenguaje AIML para la conversación AV-User, un sistema de reconocimiento automático del habla (ASR), servicio de traducción y el servicio para sintetizar la voz (TTS).

En esta ocasión, el usuario puede interaccionar con el agente por medio del teclado del computador o por medio del habla. Para el caso del empleo del habla, el sistema incorpora un ASR que convierte la expresión sonora en un texto en

lenguaje natural. Una vez codificada esta información, se traslada al módulo de traducción para que finalmente pase al motor de búsqueda de patrones de AIML.

Estos investigadores dejan recaer esta tarea en un servicio Web de forma que el código AIML y la base del conocimiento están alojados en los servidores del prestador de estos servicios. Es decir, las categorías, patrones y plantillas AIML están alojados en el servidor Web (Cloud).Este servicio tratará de encontrar una respuesta en base a lo definido por la base de conocimiento y, entonces, el servidor Web devuelve dicha respuesta al *smartphone* del usuario que la transformará utilizando un sistema TTS.

Los investigadores realizaron un test para conocer la satisfacción de uso por parte del usuario tomando como muestra representativa a 18 personas entre las edades de 20 y 50 y paridad entre sexo. La mayoría de los participantes tenían experiencia en el uso de teléfonos móviles. Las conclusiones fueron las siguientes:

- En general la mayoría de los participantes (83%) pensaba que el agente virtual es fácil de usar y el 76% de los participantes se mostraron satisfechos con las respuestas proporcionadas por el agente. La entrada de voz fue utilizada por el 83% de los participantes.
- Respecto a la satisfacción de uso de la traducción (interacción y la exactitud de la traducción) el 83% de los participantes creían que el servicio era bueno o muy bueno. No obstante, el 67% de los usuarios no valoraron bien la calidad del reconocimiento de voz.

En ese mismo año los investigadores **Ioannis Doumanis** y **Serengul Smith** de la Universidad de Middlesex University (Reino Unido), publican un interesante artículo titulado ***"Beyond Artificial Intelligence Markup Language (AIML) Rapid prototyping of a Q&A system"*** *(Doumanis y Smith, 2013)*. En este artículo se presenta un sistema pregunta-respuesta (Q&A) que es capaz de aceptar preguntas del usuario, en lenguaje natural, sobre algún tema en cuestión. El sistema que se explica en este trabajo se basa en una herramienta *on line* para que usuarios puedan interaccionar con el sistema en tiempo real y con la complejidad añadida de mantener varios diálogos en paralelo denominado *Virtual People Factory* (VPF). Desarrollaron la idea llevándola a la práctica para aplicaciones en educación de

medicina y farmacia donde los estudiantes establecen comunicaciones paciente-médico basada en reglas heurísticas.

Con este sistema, se permite el desarrollo de modelos de conversación en línea por desarrolladores con conocimientos de programación mínimos mediante la definición d patrones utilizando el lenguaje AIML.

Según lo comentado, el lenguaje AIML está siendo utilizado actualmente como una alternativa plausible al desarrollo de Agentes Conversacionales.

II.4 INTELIGENCIA AMBIENTAL Y DOMÓTICA

II.4.1 Introducción

El concepto de IAm apareció por primera vez en el año 1998 cuando el Consejo de Dirección de la empresa Philips encargó **Eli Zelkha y Brian Epstein** y **Simon Birrell** investigar sobre diferentes escenarios futuros para transformar la industria de electrónica de alto consumo en segmentos bien definidos a partir de sus principales características de uso, según los autores *"…todo con el objetivo de que para el año 2020 los dispositivos fuesen fáciles de usar apoyándose en la información ubicua, la comunicación y el entretenimiento"*.

Sin duda alguna esta visión de futuro es crucial para el desarrollo de esta tecnología que converge hacia la unión ideal **Internet-IAm**. Dada la importancia de esta tecnología y los posibles ingresos económicos para las empresas que estén involucradas en estos proyectos, aparece en el año 1999 el *Oxygen Project*[68] en el que también colabora el *Massachusetts Institute of Technology (MIT),* con objetivo de investigar y promover el desarrollo de la tecnología IAm. En el año 2.002 se crea el centro de viabilidad y utilidad dedicada a la Inteligencia Ambiental llamado *HomeLab*[69].

Podemos decir que el concepto IAm es la unión tecnológica entre dispositivos inteligentes y conectivos para el hogar junto al software que hace que estos dispositivos sean capaces de interaccionar con el usuario de un hogar ofreciendo una mejora en el confort como resultado.

En el año 2008 el investigador **José Carlos Díaz García** de la Universidad Politécnica de Madrid publica un estudio sobre el ***"Protocolo XMPP y sus aplicaciones en el campo civil y militar"***, y describe el alcance de aplicación de la IAm como "…*La Inteligencia Ambiental (AmI) se alcanza cuando un entorno electrónico, compuesto por toda clase de dispositivos electrónicos, software y comunicaciones, es sensible y responde a la presencia de las personas, colaborando de manera transparente para el usuario".*

[68] www.oxygen.lcs.mit.edu (accesible el 8 de septiembre de 2015)
[69] www.homelab.com (accesible el 8 de septiembre de 2015)

Cuando hablamos de **entorno**, de forma indirecta se está haciendo una alegación a la dotación de inteligencia a los dispositivos que rodea al usuario dentro de ese entorno. Si imaginamos una vivienda podremos afirmar que este entorno estará formado por todos los dispositivos que pueden interaccionar con el usuario: electrodomésticos, sistemas de seguridad, sistemas de alumbrado, sensores, actuadores, etc., en definitiva lo que se conoce como **Sistemas Integrados en el Entorno (SIE)**. Un ejemplo de **SIE** son las redes de sensores médicos: su aplicación en la monitorización remota de bioseñales de pacientes, deportistas, soldados o trabajadores en entornos de riesgo o críticos, es más que evidente.

II.4.2 KNX

La automatización de los hogares (conocida también como domótica) tiene sus comienzos en el año 1975 de la mano del estándar **X10**[70]. Esta tecnología se basa en el telecontrol de los dispositivos por medio del envío de datos a través de la propia línea eléctrica de la vivienda. Esta tecnología también se conoce como **corrientes portadoras** o **power line** (PL) y su arquitectura se muestra en la figura 3. El sincronimos de los distintos elementos de la red: emisor y receptor se lleva a cabo por medio del paso por 0 de la línea eléctrica. Eso significa que se sincronizarán 50 veces al segundo si la frecuencia de la red eléctrica es en España, ya que es la frecuencia normalizada en nuestro país.

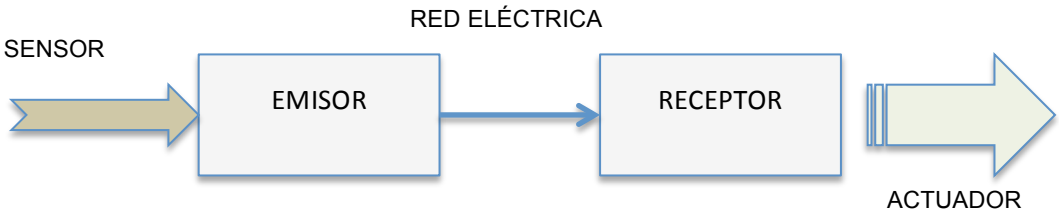


Figura 3. Arquitectura X10 para el envío de la información a través de la red eléctrica

X10 es un protocolo estándar que se extendió mucho por Estados Unidos y en Europa (sobretodo Reino Unido y España). Una de las características más importantes de este estándar es la sencillez y sobretodo la accesibilidad al protocolo el cual derivó en multitud de aplicaciones (software y hardware).

[70] http://www.lcc.uma.es/~pedro/publications/566_art.pdf (accesible el 8 de septiembre de 2015)

La lista de comandos permitidos por el protocolo X10 se reduce a 16 tal y como se muestra en la siguiente tabla 5:

Código	Función	Unidireccional	Bidireccional
0 0 0 0	All units off	X	
0 0 0 1	All lights on	X	
0 1 1 0	All lights off	X	
0 0 1 0	On	X	
0 0 1 1	Off	X	
0 1 0 0	Dim	X	
0 1 0 1	Bright	X	
0 1 1 1	Start Code		X
1 0 0 0	Hail request		X
1 0 0 1	Hail acknowledge		X
1 0 1 0	Pre-set dim		X
1 1 0 1	Status is on		X
1 1 1 0	Status is off		X
1 1 1 1	Status request		X

Tabla 5: Direccionamiento del protocolo X10

Cada comando del mensaje X10 está formado por 4 bits y representa una función concreta a realizar. El mensaje completo de un sistema X10 está formado por 11 bits: cuatro bits de inicio de la transmisión correspondiente al código reservado Start Code (0111) seguido de un código de identificación del actuador (3 bits) seguido del código de función. Por ejemplo, para el caso de encender una lámpara el código a transmitir sería 0111+identificador de la lámpara+0010.

De forma similar podríamos ejecutar los distintos códigos de función definidos en la tabla 5. Por ejemplo la función All lights On es el código encargado de encender todas las luces. La función Dim es el encargado de regular fijar los niveles de iluminación. La función de Estatus es el encargado de marcar el estado del actuador (On/Off).

El estándar X10 presenta un problema importante y es el uso de las líneas de transmisión para el tráfico de la información. El uso de las corrientes portadoras para transmitir la señal, depende directamente de la calidad de la energía con la que se suministre a la vivienda y por tanto es muy vulnerable a las frecuentes alteraciones de la misma. Por ese motivo este protocolo se ha quedado para uso marginal de operaciones sencillas del tipo Encendido/Apagado de dispositivos discretos descartándolo para regulaciones con funciones lógicas más complejas como por ejemplo la climatización.

En la década de los 90 aparecieron distintas tecnologías digitales para el control domótico de viviendas y edificios de la mano de Batibus[71], EIB[72] y EHS[73] (*European Home Systems Protocol*). En Europa estas tres soluciones para el control de viviendas y edificios intentaron al principio desarrollar sus mercados separadamente, tratando de hacerse un lugar en la normalización europea. Batibus lo hizo especialmente bien en Francia, Italia y España, mientras que EIB lo hizo en Alemania y norte de Europa. Por su parte, EHS fue la solución preferida para fabricantes de productos de línea blanca y marrón.

En el año 1997 estos tres consorcios decidieron unir sus fuerzas con el declarado objetivo de desarrollar conjuntamente el mercado del hogar inteligente, acordando crear una norma industrial común que también podría ser propuesta como norma internacional. La especificación KNX fue publicada en primavera de 2002 por la recién establecida ***KNX Association***[74] ***(KNX)***, logrando penetrar lentamente en un mercado reticente como es el retail y vivienda a pesar de que es un sistema muy robusto y fiable.

[71] www.es.wikipedia.org/wiki/BatiBUS (accesible el 8 de septiembre de 2015)
[72] www.wiki.linuxmce.org/index.php/EIB/KNX (accesible el 8 de septiembre de 2015)
[73] www.en.wikipedia.org/wiki/European_Home_Systems_Protocol (accesible el 8 de septiembre de 2015)
[74] www.knx.org (accesible el 8 de septiembre de 2015)

En diciembre de 2003 el protocolo KNX así como los dos medios de transmisión TP (par trenzado) y PL (línea eléctrica) fueron aprobados por los comités nacionales europeos y ratificados por el ***CENELEC Bureau Technique***[75], como el estándar europeo EN 50090. KNX en radio frecuencia fue aprobado en mayo de 2006.

En el "mundo" de la domótica de edificios y viviendas la mayoría de los fabricantes de dispositivos se acogen a fabricarlos bajo unos estándares de comunicaciones. En Europa es la **KNX Association** (385 fabricantes KNX en 38 países) quien determina estas características. Esta asociación es la creadora y propietaria de la tecnología KNX presente en todas las aplicaciones de control de la vivienda y edificios, abarcando desde control de la iluminación y las persianas, como variados sistemas de seguridad, calefacción, ventilación, aire acondicionado, monitorización, alarma, control de agua, gestión de energía, contador, así como electrodomésticos del hogar, audio/video y mucho más.

II.4.3 Arquitectura KNX y nivel físico

La arquitectura de comunicación e instalación de dispositivos se presenta en la siguiente figura 4:

75 http://www.cenelec.eu (accesible el 8 de septiembre de 2015)

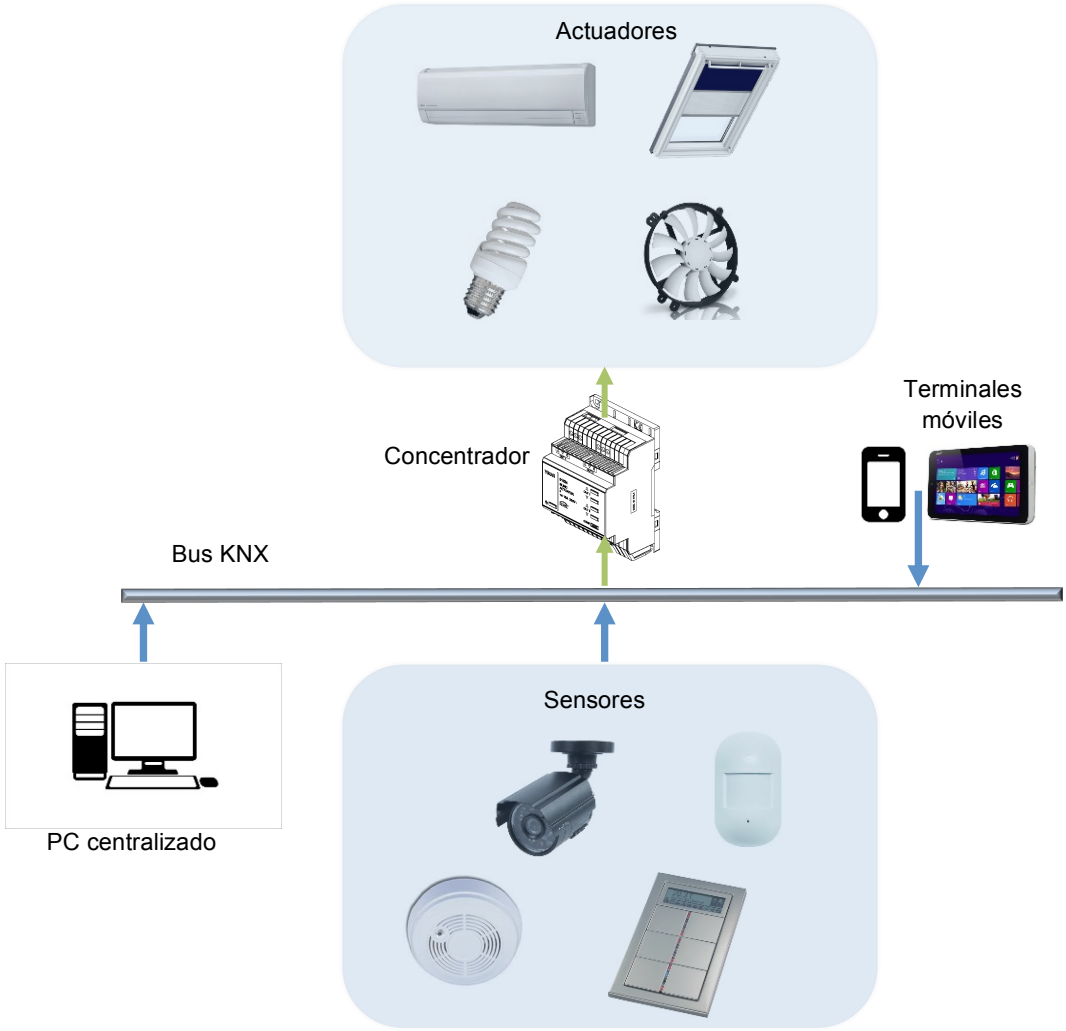


Figura 4. Arquitectura de KNX para domótica

La arquitectura tipo de una instalación domótica está formada por los elementos actuadores, sensores, concentrador lógico y terminales para acceder a las funciones programadas *(Stefan et. al, 2004)*. El programa de funcionamiento de la instalación se instala en el concentrador lógico a través de un terminal de programación. Este terminal de programación puede ser un PC centralizado o terminal remoto. El programa básicamente relaciona las señales recibidas de los sensores con el correspondiente actuador a través de unas funciones de estado. La asociación KNX publicó en el año 2004 el documento técnico denominados

Interworking[76] donde se establecen las bases técnicas para programar y definir variables para el control KNX de una instalación domótica.

KNX ofrece la ventaja de ser una instalación flexible respecto a su funcionalidad. Esto significa que los elementos actuadores y detectores (sensores) de una instalación pueden modificarse por software en cualquier momento.

Respecto a la instalación, la conexión entre dispositivos KNX están interconectados por medio de un bus que tiene la doble función de servir para la transmisión de datos así como de alimentación eléctrica, es decir, conecta y transmite todas las señales de control. Una de las características importantes de este sistema es que para la transmisión de datos no es necesario una gran velocidad de procesado porque el intercambio de información se realiza a través de "telegramas". Con esto se consigue una velocidad de transmisión rápida con un alto nivel de inmunidad al ruido.

Esta arquitectura es un bus de datos serie que recorre toda la instalación e interconectas entre sí los dispositivos actuadores, sensores y las centrales lógicas. KNX tiene una tecnología flexible y distintos medios de transmisión:

- TP1 (Par trenzado)
- PL110 (Powerline, esto es la red eléctrica)
- RF (Radio frecuencia)
- Ethernet (IP)

Los medios de transmisión anteriores se pueden unir mediante los acopladores de medios correspondientes. De entre todos ellos, el medio de transmisión por excelencia de KNX es TP1 (Par trenzado), es decir, mediante un bus de control independiente (un bus consiste básicamente en un par de cables), normalmente de color verde, al cual se conectan los dispositivos.

Los componentes de un sistema KNX son los **sensores** y **actuadores**. Los sensores recopilan la información del medio (temperatura, velocidad del viento,

[76] www.knx.org/fileadmin/template/documents/downloads_support_menu/KNX_tutor_seminar_page/Advanced_documentation/05_Interworking_E1209.pdf (accesible el 20 de octubre de 2015)

movimientos, instrucciones por parte del usuario, etc.) y los envían en el bus como telegramas de datos. Los actuadores reciben datos que posteriormente se convierten en acciones. Esto puede incluir incluso el control de persianas, la atenuación de luces o el control de sistemas de calefacción y aire acondicionado (figura 5).

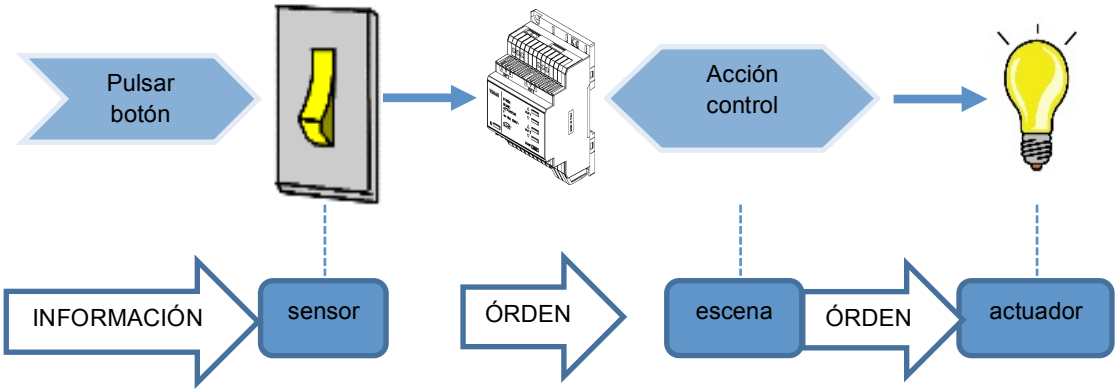


Figura 5. Principio funcional y estructura de la propagación de la información en KNX

II.4.4 Ventajas del uso de KNX

El uso de KNX presenta una serie de ventajas frente a la solución tradicional destacando:

- Arquitectura de control distribuida; esto quiere decir que no es necesario disponer de un sistema de control centralizado del tipo ordenador/autómata para controlar la instalación. Para ello cada elemento del sistema dispone de su propia **inteligencia** y se comunican entre ellos, lo que permite, además, un rápida modificación de la instalación.
- Configuración de los dispositivos por medio de un software único independientemente del modelo y fabricante.
- Programación de escenas. Estos significa que se pueden programar la acción paralela de varios actuadores accionadas por medio de una pulsación de un elemento sensor, por ejemplo un pulsador.
- Ahorro energético. Los sistemas domóticos permiten tener ahorros energéticos por desconexión de aquellos elementos que no están siendo

utilizados, por ejemplo, desconexión automática de la iluminación de una sala vacía.

Por último cabe destacar que la domótica debe estar sustentada por 5 importantes pilares *(Sáez, 2004)* que tendrán que tener en cuenta el sistema desarrollado en esta tesis doctoral;

- **Seguridad:** Una seguridad tanto para proteger de las agresiones externas (robos e intrusos) como de las internas (fugas de gas, agua, incendios, emergencias de salud).

- **Confort:** Una vivienda domótica debe hacer más agradable la vida a sus ocupantes, librándoles de tareas repetitivas y ofreciéndoles nuevos servicios dirigidos hacia su comodidad.

- **Ahorro:** La vivienda domótica ayuda a ahorrar recursos a sus habitantes optimizando el rendimiento, y por tanto el gasto, de aspectos como la climatización, consumo eléctrico en grandes electrodomésticos, iluminación, etc.

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

- **Ocio:** El aprovechamiento del tiempo libre se facilita con el soporte de sistemas domóticos tales como los sistemas audiovisuales, equipos de entretenimiento y complementos tecnológicos para el hogar.

II.4.5 Control de dispositivos mediante lenguaje natural

Como se ha comentado anteriormente, en el contexto las aplicaciones para el control de dispositivos a través de Internet se ha incrementado de manera drástica. Gracias a estas aplicaciones, los usuarios pueden manipular los sistemas

domóticos por medio de dispositivos móviles como smartphones o tabletas. Por otro lado, existen muchos esfuerzos para permitir el control de estos dispositivos mediante lenguaje natural como se ha visto en la sección II.3.2.

En la tabla 6 seleccionamos algunos de los trabajos más representativos con respecto al control de los dispositivos mediante lenguaje natural.

La heterogeneidad de este campo queda patente también en la tabla donde se han destacado distintos parámetros que pueden utilizarse para clasificar este tipo de sistema y se explican a continuación:

- Lenguaje natural / comandos predefinidos: Muchos sistemas únicamente permiten interactuar con ellos mediante comandos predefinidos, mientras que otros permiten el uso de cualquier expresión en lenguaje natural que tenga el mismo significado de la orden a realizar.
- Oral / escrito: Si utilizan solamente interacción oral, escrita o las dos.
- Consulta de estado: Si permiten además de la ejecución de comandos, la consulta del estado de los dispositivos.
- Gestión del diálogo: Si contempla una gestión del diálogo guardando el estado del diálogo y la sesión.
- Centralizado / distribuido: Si el control de los dispositivos es centralizado en un sistema o bien es distribuido y cada sistema es independiente.
- Estándares: En qué estándares domóticos o de IoT puede funcionar.
- Alertas: Si los dispositivos pueden comunicarse de manera asíncrona con el usuario cuando se generan alertas.

II.4.5.1 Lenguaje natural / comandos predefinidos (LN / CP)

Algunos sistemas sólo permiten el control mediante comandos predefinidos que pueden ser configurados por los usuarios como es el caso de G.H.O.S.T *(Santos et. al, 2011)*, SWEET-HOME *(Portet et. al, 2013)*, AVATAR *(Kumar, 2014)* y el trabajo presentado por Kumar en el año 2014. Estos sistemas tienen bastantes limitaciones ya que es necesario memorizar y recordar distintos comandos para poder utilizarlos. Para dar más flexibilidad a estos sistemas aparecen los basados

en reconocimiento del lenguaje natural que permiten interpretar y extraer el significado de cada orden para poder así manejar estos dispositivos de manera inteligente y sin necesidad de memorizar comandos predefinidos. Este es el caso de los sistemas INVOX *(Kumar, 2014)*, Mayordomo *(Soda et. al, 2013)* y im4Things *(Noguera et. al, 2015)* analizan el lenguaje natural del comando, orden o consulta para poder así transformarlo en una posible acción en base a su significado.

II.4.5.2 Oral / escrito

La mayoría de sistemas de control de dispositivos son orales, es decir, se utilizan a través de la voz *(G.H.O.S.T (Santos et. al, 2011), SWEET-HOME (Portet et. al, 2013), AVATAR (Kumar, 2014), INVOX (Kumar, 2014) y Mayordomo (Soda et. al, 2013))* e incorporan sistemas ASR en sus plataformas. El sistema im4Things propuesto en esta tesis doctoral se basa en el envío de órdenes en lenguaje natural escrito mediante una plataforma de mensajería instantánea.

II.4.5.3 Consulta de estado

Además de poder controlar los dispositivos mediante órdenes en lenguaje natural, también es importante poder conocer su estado. Por ejemplo, en caso de avería o detección de un mal funcionamiento en el dispositivo, sería deseable poder preguntarle al dispositivo cuál es el problema. Por ejemplo, en una instalación domótica, además de poder ordenar la apertura y cierre de ventanas y persianas, sería adecuado poder preguntar si esas ventanas están abiertas o cerradas o si una llave de paso está abierta o no.

La mayoría de sistemas no permite la consulta de estado. De los sistemas estudiados, solamente los sistemas Mayordomo *(Soda et. al, 2013)* y el sistema im4Things propuesto en esta tesis doctoral permiten esta funcionalidad.

II.4.5.4 Gestión del diálogo

Dentro de la interacción con el dispositivo también es interesante saber si el sistema permite el diálogo inteligente con el mismo. Es decir, si guarda

información sobre el estado del diálogo y en base al diálogo anterior eso puede realizar unas acciones u otras.

Muchos sistemas no guardan información de contexto del diálogo mantenido y cada interacción con los dispositivos es como si fuera nueva. Este es el caso de los sistemas INVOX *(Kumar, 2014)*, G.H.O.S.*T (Santos et. al, 2011),* SWEET-HOME *(Portet et. al, 2013)*. Por otro lado, los sistemas Mayordomo *(Soda et. al, 2012)*, AVATAR *(Kumar, 2014)* y el sistema Im4Things *(Noguera et. al, 2015)* permiten seguir una conversación con el dispositivo.

II.4.5.5 Centralizado / distribuido

Según el estudio realizado y los artículos seleccionados, todas las aproximaciones suelen ser centralizadas. Esto es, utilizan algún ordenador o dispositivo que sea el que gestione toda la comunicación y las órdenes con respecto a todos los dispositivos que se quieren controlar.

El sistema propuesto en esta tesis doctoral permite que cada dispositivo sea autónomo y contenga la inteligencia para que pueda manejarse independientemente por lo que se propone un control distribuido.

II.4.5.6 Estándares

Debido a que los dispositivos a controlar son dispositivos en instalaciones domóticas es común que muchas de los métodos propuestos utilicen el estándar KNX (Soda et. al, 2012), INVOX *(Kumar, 2014)*, G.H.O.S.T *(Santos et. al, 2011)* y SWEET-HOME *(Portet et. al, 2013)* o X10 (Mayordomo *(Soda et. al, 2012)*). Por otro lado, otros sistemas no implementan ningún protocolo o tienen un protocolo propietario como es el caso de Mayordomo *(Soda et. al, 2012)*, o AVATAR *(Kumar, 2014)*. El sistema Im4Things propuesto en esta tesis doctoral permite el control distribuido y autónomo de cada dispositivo mediante un diálogo en lenguaje natural mediante servicios de mensajería instantánea, por lo que todos los dispositivos y usuarios pueden comunicarse entre sí a través de estos servicios que se explican en el capítulo III.

II.4.5.7 Alertas

Por último, en base al estudio realizado para esta tesis doctoral, no existe ningún sistema que identifique alertas y las transmita a los usuarios o a otros dispositivos en lenguaje natural. Uno de los objetivos de esta tesis doctoral es poder desarrollar un sistema que permita avisar al usuario en lenguaje natural si se produce alguna alerta en el dispositivo como, por ejemplo, que se ha terminado un programa de la lavadora o se detecta algún fallo en el funcionamiento.

Sistema	**LN / CP**	**Oral / escrito**	**Consulta de estado**	**Gestión diálogo**	**Centralizado / distribuido**	**Estándares**	**Alertas**
INVOX *(Kumar, 2014)*	LN	Oral	No	No	Centralizado	KNX	No
G.H.O.S.T *(Santos et. al, 2011)*	CP	Oral	No	No	Centralizado	KNX	No
Mayordomo *(Soda et. al, 2012)*	LN	Oral	Sí	Sí	Centralizado	X10	No
SWEET-HOME (Portet et. al, 2013)	CP	Oral	No	No	Centralizado	KNX	No
KNX (Soda et. al, 2012)	LN	Oral	No	Sí	Centralizado	-	No
AVATAR (Kumar, 2014)	CP	Oral	No	No	Centralizado	- ARDUINO	No

AVATAR (Santos et. al, 2011)	CP	Oral	No	Sí	Centralizado	-	No
Im4Things *(Noguera et. al, 2015)*	LN	Escrito	Sí	Sí	Distribuido	-	Sí

Tabla 6. Comparación entre distintos sistemas de diálogo.

II.4.5.8 Conclusiones

Como se ha podido observar el sistema Mayordomo *(Soda et. al, 2012)* es uno de los sistemas más completos ya que permite la consulta del estado de los dispositivos, gestiona el diálogo y permite órdenes en lenguaje natural. Sin embargo, el control del mismo es centralizado y no permite el aviso al usuario por alertas.

El sistema Mayordomo supone un paso previo para el desarrollo del sistema im4Things que se presenta en esta tesis doctoral. El desarrollo de esta tecnología supone un salto sustancial en el campo de la **IAm** porque nuestro sistema permite la manipulación (control) de los electrodomésticos instalados en la vivienda de manera escrita en lenguaje natural así como a través de una interfaz gráfica tipo Scada.

El sistema planteado en esta tesis doctoral permitirá el desarrollo de un hardware específico que será integrable en cualquier dispositivo y permita a cada dispositivo de manera distribuida interactuar en el sistema global o bien recibiendo mensajes y órdenes, o bien generando alertas y alarmas. Esta comunicación se realizará en base a mensajería instantánea y no dependerá de ningún protocolo domótico. Además, el sistema de diálogo planteado se basará en una aplicación de mensajería instantánea tipo Chat que permita de manera inteligente interpretar las órdenes o las consultas y mantener un diálogo con los distintos usuarios.

Es importante también destacar que para que los dispositivos sean fácilmente configurables, es necesario que dispongan de una base de conocimiento que permita definir sus funciones y comportamiento entre otras cosas.

II.5 BASES DE CONOCIMIENTO. ONTOLOGÍAS

II.5.1 Introducción

La ingeniería del conocimiento surgió en los años 70 y trataba de diseñar y construir sistemas software basados en conocimiento. Las principales diferencias con la ingeniería del software era que esta disciplina se basaba en proporcionar un enfoque sistemático para el desarrollo de sistemas inteligentes que permitiesen modelar conocimiento humano de expertos con posibilidad de aprendizaje automático y la aplicación de razonamiento.

En el principio de los años 80 el desarrollo de sistemas basados en conocimiento se veía como un proceso de transferencia del conocimiento humano a una base de conocimiento implementada. Esta transferencia se basaba en la suposición de que el conocimiento necesario para los sistemas basados en conocimiento existe y solamente tiene que ser recolectado e implementado en un ordenador *(Studer et. al, 1998)*.

Debido a la dificultad de modelar esas bases de conocimiento, durante los años 90, aparecieron distintos frameworks para el modelado de sistemas basado en conocimiento delos que se pueden destacar los siguientes: CommonKADS *(Schreiber, 2000)* que permitía definir la estructura de la base de conocimiento, MIKE *(Ángele et. al 1998)* que ponía énfasis en una especificación formal y ejecutable de la base de conocimiento y PROTÉGÉ II (*Eriksson et. al 1995)* que explotaba el concepto de ontología que se describe en la siguiente sección.

Las formas de representación de estas bases de conocimiento han sido diversas durante los años utilizándose principalmente reglas lógicas, marcos, lógica fuzzy *(Hájek, 2001)* y ontologías *(Sowa, 1999)*.

Para esta tesis doctoral se ha optado por representar el conocimiento con ontologías debido al auge actual que están teniendo como tecnología esencial para el desarrollo de la Web Semántica *(Berners-Lee, 2001)*.

En el siguiente apartado se explica qué son las ontologías y qué componentes las forman.

II.5.2 Ontologías

Las ontologías son la tecnología más importante dentro de la Web Semántica y permiten representar el conocimiento de manera explícita y formal para que sea procesable por las máquinas. Más concretamente, una ontología puede definirse como una "*representación formal y explícita de una conceptualización compartida*" *(Studer, 1998)*. Con conceptualización se refiere a que se crea un modelo abstracto de una realidad mediante el uso de conceptos. Explícito quiere decir que sus conceptos, tipos y restricciones se definen de manera explícita. Formal se refiere a que puede ser procesable por una máquina y compartido que el conocimiento incluido en la ontología es consensuado y aceptado por una comunidad.

Las ontologías se están empleando en todo tipo de sistemas basados en conocimiento en las que sea necesario definir concretamente el conjunto de entidades relevantes en el campo de aplicación determinado, así como las interacciones entre las mismas. Como ejemplos de dominios donde se están aplicando estas tecnologías podemos destacar el turismo *(Ruiz-Martınez, 2011)*, e-learning *(Fernández-Breis, 2009)*, la biomedicina *(García-Sánchez, 2008)* y las finanzas *(Esteban-Gil, 2012)*.

Además de estas aplicaciones, el uso de ontologías puede mejorar las limitaciones de los métodos tradicionales de procesamiento del lenguaje natural. También son relevantes en el ámbito de algunos métodos relacionados como la búsqueda semántica *(Lupiani-Ruiz, 2011)*, descubrimiento de servicios *(García-Sánchez, 2009)*, extracción de información *(Valencia-García, 2008)* y sistemas de recomendación *(Carrer-Neto, 2012)*, *(Colombo-Mendoza, 2015)*, entre otros.

En las aplicaciones anteriormente destacadas, estas ontologías se crean con el mero objetivo de alcanzar una comprensión del dominio pertinente, ya que su creación impone una especificación muy detallada. Otras ontologías han sido creadas con un propósito general, como por ejemplo el proyecto Cyc *(Lenat, 1995)*, que está orientado a la construcción de una base de conocimiento que contenga el conocimiento humano necesario para hacer inferencias.

Según *(Contreras y Martínez)*[77], podemos clasificar las ontologías, según el dominio y propósito que cubren, en los siguientes tipos:

- Ontologías de nivel superior: Permiten modelar los niveles altos de una realidad, ofreciendo conceptos genéricos para la clasificación de términos. Ejemplos de ontologías de estas características son CyC *(Lenat, 1995)*, WordNet *(Miller, 1995)* o SUMO[78].
- Ontologías generales: Algunos conceptos como el tiempo, el espacio, eventos, etc., pueden reutilizarse a través de diferentes dominios.
- Ontologías de dominio: Ontologías de dominios específicos o para aplicaciones concretas modelan las particularidades de las realidades de acuerdo a los propósitos de explotación impuestos.

En esta tesis doctoral interesan las ontologías del dominio que dan cuenta del conocimiento del dominio en base a su representación mediante principalmente 5 elementos: clases, atributos, relaciones, instancias y axiomas.

- Clases: Representan los tipos de objetos del mundo real. Se suele usar tanto el término clases como conceptos. Un concepto puede ser algo sobre lo que se dice algo y, por lo tanto, también podría ser la descripción de una tarea, función, acción, estrategia, sensor, dispositivo, alerta, etc.
- Atributos: Los atributos son características que permite describir más detalladamente la clase y sus instancias. Establece que la clase o concepto posee una propiedad que se concretará mediante un valor. Los valores de las propiedades o atributos pueden ser tipos básicos como cadenas de caracteres o números.
- Relaciones: Las relaciones representan un tipo de interacción entre los conceptos del dominio. Se definen formalmente como cualquier subconjunto de un producto de n conjuntos, esto es, R: C1 × C2 × ... × Cn. Como ejemplos de relaciones binarias incluimos: "subclase de" y "conectado a".
- Axiomas: Los axiomas son expresiones y reglas del dominio que siempre son verdad. Pueden ser incluidas en una ontología con muchos

[77] www.sedic.es/gt_normalizacion_tutorial_ontologias.pdf (accesible el 27 de octubre de 2015)
[78] www.adampease.org/OP/ Suggested Upper Merged Ontology (SUMO) (accesible el 27 de octubre de 2015)

propósitos, tales como definir el significado de los componentes ontológicos, definir restricciones complejas sobre los valores de los atributos, argumentos de relaciones, etc. verificando la corrección de la información especificada en la ontología o deduciendo nueva información. Tales ontologías son llamadas ontologías pesadas, en contraste con las ontologías ligeras, que no incluyen axiomas.

- Instancias: Las instancias se usan para representar elementos u objetos específicos del mundo real.

Para representar las ontologías existen distintos lenguajes ontológicos. Actualmente el estándar es la segunda versión del Web Ontology Language (OWL 2[79]) que es una recomendación del W3C desde diciembre de 2012. OWL 2 es una ampliación de OWL en el cual se mejoraron algunas limitaciones identificadas en esta primera versión como la mejora de los tipos de datos y de la definición de algunos axiomas que con la primera versión eran mucho más complejos de lo que son conjuntos de las características, para las cuales se han incorporado algoritmos eficaces de razonamiento. Además, incluye nuevas características como la extensibilidad sintáctica, propiedades adicionales y constructores de cardinalidad calificados, metamodelos simples, cadenas de propiedades y anotaciones extendidas.

En esta tesis doctoral se ha hecho uso de OWL 2 debido a que es el estándar actual para el desarrollo de ontologías y tiene todas las características necesarias para poder modelar el conocimiento de los dispositivos.

[79] http://www.w3.org/TR/owl2-overview/ (accesible el 27 de octubre de 2015)

II.6 HARDWARE PARA IoT

II.6.1 Introducción

En el ecosistema de la IoT, la piedra angular sobre la que se basa la estructura es sin duda el hardware. Hasta hace unos años, el acceso al diseño y fabricación de hardware se limitaba fundamentalmente a Universidades, Centros tecnológicos y Empresas especializadas porque durante el proceso de fabricación del circuito impreso se hacía imprescindible contar con equipamiento de revelado de la placa PCB (**P**lace **C**ircuit **B**oard) entre otros elementos los cuales eran costosos y voluminosos.

Con la llegada de la microelectrónica en el formato microprocesador primero y el microcontrolador después, el diseño electrónico dio paso a una nueva revolución tecnológica porque se abría paso a la posibilidad de realizar sistemas Hardware y Software programables para tareas complejas y con ello la computación de alto nivel. Gracias a esta tecnología contamos a día de hoy con los dispositivos que han revolucionado la forma de comunicarnos entre humanos y con máquinas: los dispositivos móviles como smartphones y tabletas.

Ya recientemente, con la llegada de la tecnología embebida tipo Arduino[80], Raspberry Pi[81] y similares, se ha roto la barrera de entrada antes comentada y es posible que tanto profesionales como aficionados a la electrónica puedan realizar sus propios diseños incluido la programación software de sus aplicaciones para desempeñar una función concreta, al menos en fase de prototipado.

En esta línea de innovación en el ecosistema IoT de los miniordenadores embebidos, se recoge en ***Advances onto the Internet of Things*** *(Advances in Intelligent Systems and Computing)(Gaglio y Lo Re 2014)* cómo a partir de una placa base formada por un microcontrolador y diversos puertos de entradas y salidas de señal así como puertos de comunicación serie[82], se puede construir una plataforma completa relativamente económica para sumergirse en este entorno, con aspectos de investigación y desarrollo de aplicaciones IoT. De hecho a partir del año 2010 empiezan aparecer diferentes grupos de desarrolladores sin ánimo de lucro que

[80] www.arduino.cc (accesible el 8 de septiembre de 2015)
[81] www.raspberrypi.org (accesible el 8 de septiembre de 2015)
[82] www.arduino.cc/en/Main/ArduinoShields (accesible el 8 de septiembre de 2015)

comparten sus programas aplicativos en la nube con la intención de ser utilizados por aquellas personas que deseen descargarla en su plataforma embebida. Estamos entonces sin duda alguna ante el nacimiento de la IoT facilitado en parte a la rotura de las barreras antes comentadas.

II.6.2 Introducción

Sería muy difícil y extenso arrancar este punto de la tesis doctoral desde una fecha objetiva para describir el verdadero punto de inicio de la electrónica orientada a sistemas relacionales con la IoT. Podemos empezar desde el año en que se descubrió la estructura atómica y el conjunto electrón-protón y desde ahí desarrollar toda la temática. Por el método de inducción podríamos partir desde el año 1883 momento en que Thomas Alva Edison, a fin de alargar la vida de sus lámparas incandescentes, observó que después de un tiempo de funcionamiento el interior de la lámpara se ennegrecía y gracias a esa observación descubrió que poniendo una placa sobre el filamento la lámpara desaparecía este efecto. Observó que la placa colocada se cargaba eléctricamente. A día de hoy este efecto se le conoce con el nombre de Efecto Édison. Más tarde, J.A. Fleming estudió este efecto y acuñó el término de **Emisión Termoiónica** y con ello al nacimiento del **Diodo de vacío** (predecesor por muchos años al diodo semiconductor).

En el año 1948 el equipo de físicos formado por John Bardeen, William Shockley y Walter Brattain, dieron a conocer al mundo su invención conocida como **Transistor de unión bipolar** como unión de las partes semiconductoras del **Diodo NPN y PNP**. Este descubrimiento fue de tal calado e importancia para el mundo de la electrónica física que fueron merecedores del premio Nobel en el año 1956[83].

Dando un salto temporal cuantitativo, llegamos a la década de los años 50 con el nacimiento de la **microelectrónica**. Hablar de microelectrónica lleva asociada sin duda la figura del científico y empresario Robert Noyce que fue co-fundador de la empresa fabricante de componentes electrónicos **Fairchild Semiconductor**[84]

[83] www.bell-labs.com/our-people/recognition/1956-transistor/# (accesible el 19 de septiembre de 2015)
[84] www.fairchildsemi.com (accesible el 19 de septiembre de 2015)

(1957) y posteriormente fundador de la empresa **Intel Corporation**[85] (1968). A Robert Noyce se le reconoce junto a Jack Kilby como el inventor del **Circuito Integrado**[86] **(CI)**, que es una pieza calve e indispensable para el desarrollo de la microelectrónica y que a día de hoy está instalada en cualquier dispositivo electrónico digital y verdadero motor de la revolución informática. Por otro lado Robert Noyce sentó las bases prácticas para la producción en masa de los microprocesadores. Se puede afirmar que la ternar **E**lectrónica-**I**nformática-**T**elecomunicaciones o también conocida como tecnologías **TIC** han posibilitado el nacimiento de la hoy comentada Internet de las Cosas como tecnología trasversal a otras tecnologías verticales como por ejemplo la conocida Smart City, Smart Tourism y similares.

Se abre aquí una cuestión importante desde el punto de vista técnico y que puede suponer no potenciar la tecnología IoT en el futuro debido a que el protocolo IPv4 no tiene capacidad para direccionar todas las posibles direcciones de los dispositivos electrónicos destinados a la IoT. Para solucionar este problema se ha desarrollado el protocolo IPv6[87].

En la década de los 90, Internet era utilizado por universidades, gobiernos (defensa) y empresas tecnológicas de todo el mundo. El interés masivo por el uso de Internet y las posibilidades que este ofrece propició que desde finales de la década de los 90 y hasta la fecha proliferase que otras tecnologías, además del PC, se conectasen a la capa de red de Internet. Me refiero a los equipos de telefonía móvil, tabletas, televisores inteligentes y ahora los dispositivos de la *Internet of Things*. Este escenario ha obligado que el IP evolucionase y se hiciese más flexible.

La organización ***Internet Engineering Task Force***[88] **(IETF)**, cuyo cometido es hacer que Internet funcione mejor garantizando su uso y gestión eficiente y de calidad, comenzó a trabajar en el año 1990 en una nueva versión del IP con la premisa de que nunca se quedase sin direcciones además de ser flexible y eficiente. Según apunta el **Dr. Andrew Stuart Tanenbaum**, las principales metas que se fijaron para el nuevo IPv6 fueron:

[85] www.intel.es/ (accesible el 19 de septiembre de 2015)
[86] www.nobelprize.org/educational/physics/integrated_circuit/history/ (accesible el 19 de septiembre de 2015)
[87] www.ipv6.com (accesible el 19 de septiembre de 2015)
[88] www.ietf.org (accesible el 19 de septiembre de 2015)

1. Manejar miles de millones de hosts.
2. Reducir el tamaño de las tablas de enrutamiento.
3. Simplificar el protocolo, para permitir a los enrutadores el procesamiento más rápido de los paquetes.
4. Proporcionar mayor seguridad (verificación de autenticidad y confidencialidad) que el IP actual.
5. Prestar mayor atención al tipo de servicio, especialmente con datos en tiempo real.
6. Ayudar a la multidifusión permitiendo las especificaciones de alcances.
7. Posibilitar que in host sea móvil sin cambiar su dirección.
8. Permitir que el protocolo evolucione.
9. Permitir que el protocolo viejo y el nuevo coexistan por años.

Tras una selección de 21 propuestas recibidas por el IETF para mejorar la versión de la IPv6, la IEEE Network publicó las 3 mejores propuestas y finalmente seleccionó como mejor propuesta se selección la propuesta SIPP, combinación de las propuestas por Deering y Francis (1993). Además el Dr. Andrew Stuart Tanenbaum apunta que *"...con esta elección se cumple los objetivos bastante bien: mantiene las buenas características del IP, descarta y reduce las malas y agrega nuevas donde se necesitan pero el IPv6 no es compatible con el IPv4".*

En la tabla 6 se detallan las principales diferencias entre los protocolos IPv4 e IPv6:

Característica	IPv4	IPv6	Ventaja competitiva
Direccionamiento	32 bits	128 bits	Mayor número de direccionamientos
Encabezado	13 bits	7 bits	Aumento de enrutamiento y por lo tanto menor tiempo de respuesta
Seguridad	Empleo del protocolo IPsec[89]	Posterior a establecerse en el IPv6	Mejora la seguridad durante el tráfico de paquetes bajo TCP
Multicast	Posibilidad de enviar un paquete único a destinos múltiples	Recientemente implementado	Calidad de servicio
Autoconfiguración	Implementada	No	Durante el proceso de instalación

Tabla 7. Principales diferencias entre los protocolos IPv4 e IPv6

Pero sin duda alguna la ventaja más importante que traerá la versión IPv6 es la desaparición de los NAT (direccionamientos privados) que apuntan a una **única** dirección IP por no disponer de suficientes direcciones IP. Este hecho constituye de por sí el gran avance tecnológico y con ello la posibilidad de que la IoT sea una realidad.

Hasta este punto ya disponemos de suficientes conexiones IP a Internet por medio del protocolos TCP/IP, es decir, podríamos conectar sensores bajo una IP para que la información se envíe a servidores en cualquier parte del mundo. Ahora es necesario que esta conexión se realice por medio de la tecnología inalámbrica para romper la barrera de necesitar cables. Es el momento de integrar las tecnologías inalámbricas **Bluetooth**[90]**, ZegBee**[91] y **WiFi**[92].

[89] Internet Protocol security. Asegura el tráfico bajo el Protocolo de Internet (IP), por medio de la autenticación y cifrado de cada paquete transmitido.

[90] www.bluetooth.com (accesible el 19 de septiembre de 2015)
[91] www.zigbee.org (accesible el 19 de septiembre de 2015)
[92] www.wi-fi.org (accesible el 19 de septiembre de 2015)

II.7 TECNOLOGÍAS INALÁMBRICAS: Bluetooth, ZegBee y WiFi

II.7.1 Introducción

De las tecnologías inalámbricas más extendidas en el mercado encontramos dos clasificaciones importantes: **redes de área personal** (*wireless personal area network* -WPAN-Bluetooth y ZegBee) y **redes de área local** (Wifi). Estas dos tecnologías conforman la arquitectura actual más utilizada en el ecosistema IoT y dada su importancia se van a describir en la presente tesis doctoral. Desde el punto de vista de este doctorando la tecnología inalámbrica más importante para IoT es la tecnología Wifi mientras que las otras dos tecnologías inalámbricas conforman una arquitectura auxiliar de integración de sistemas necesaria para la conformación global de la plataforma Smart por sus cualidades funcionales y energéticas.

A continuación se describen en mayor detalle estas tecnologías.

II.7.2 Bluetooth

La necesidad de conectar dispositivos móviles de forma inalámbrica llevó en el año 1994 a las empresas L.M Ericsson y después a la creación de un consorcio empresarial compuesto por las empresas IBM, Intel, Nokia y Toshiba a desarrollar un estándar inalámbrico para la comunicación de PCs, terminales móviles y accesorios. El proyecto desarrollado por este consorcio se denominó "Bluetooth" en honor a **Harald Blaatand**[93]. Nació así la tecnología para conectar dispositivos dentro de un área local útil para la conexión de dispositivos *wearables*.

La tecnología inalámbrica Bluetooth ha tenido desde su nacimiento diferentes mejoras en lo que a especificaciones se refiere así como versiones "competencia" de organizaciones como el IEEE con el estándar 802.15 incompatible desde el punto de vista eléctrico con la tecnología Bluetooth.

[93] Rey vikingo que conquisto y unificó Dinamarca (958 dC) y Noruega (970 dC)

¿Cómo está integrada esta tecnología en el ecosistema IoT? El portal de ***Application Developers Alliance***[94] informa que a finales del año 2014 más de 35 millones de dispositivos wearables fueron vendidos y conectados de algún modo a Internet. Estos dispositivos están focalizados principalmente a relojes inteligentes y prendas de vestir. El año 2014 fue el punto de "no retorno" en lo que a esta tecnología se refiere. Aparece otro concepto familiar que es una prolongación de *Internet of Things* y es el de *"Internet of Me*[95]*"*.

La unidad básica de un sistema Bluetooth es una **piconet**[96]. La arquitectura piconet consta de un nodo maestro (máster) y hasta siete nodos esclavos (slaves) activos. El radio de cobertura desde el nodo máster hasta el slave es de 10 metros (Ver figura 6). A su vez, se pueden formar varias redes piconets a través nodos puente y con ello obtener una red mallada. En total podemos tener interconectados hasta 255 nodos en la misma red mallada.

La comunicación se realiza en el sentido máster-slave no entre scales. Este hecho conlleva una ventaja sustancial y es que esta tecnología está pensada para trabajar con consumos muy reducidos. De hecho, los nodos slave habitualmente están en estado "sleeping" y solo se activan cuando el nodo máster les envía un mensaje de activación.

[94] www.appdevelopersalliance.org (accesible el 19 de septiembre de 2015)
[95] www.cmo.com (accesible el 19 de septiembre de 2015)
[96] www.ieeexplore.ieee.org (accesible el 19 de septiembre de 2015)

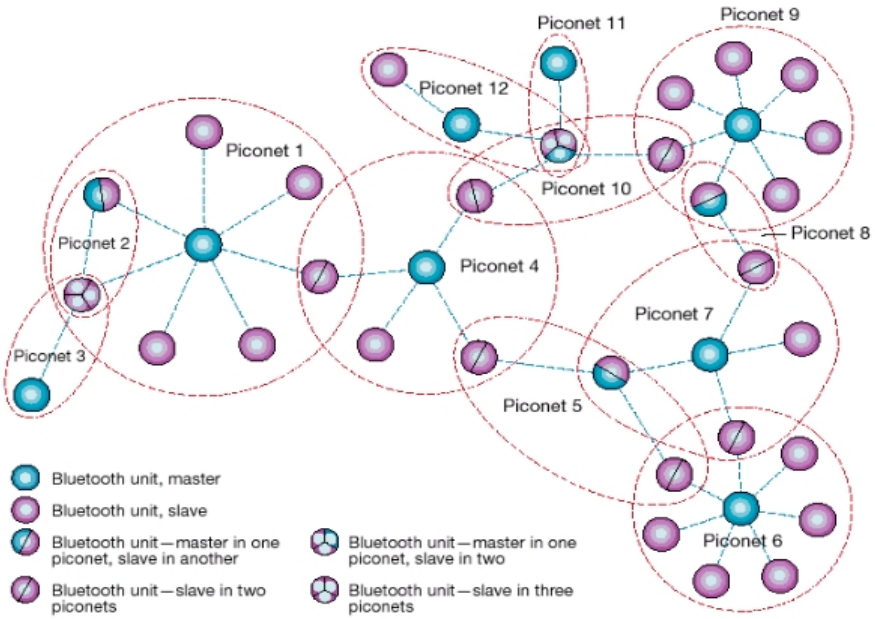


Figura 6: Arquitectura Piconet para Bluetooth (*Fuente www.infotechplus.free.fr)*

Las aplicaciones de esta tecnología son enormes. Un ejemplo de ello es la monitorización de temperatura, en el que el proceso repetitivo es el siguiente: lectura de temperatura, envío de información al nodo máster, entrada en sleeping y así repetidamente. Por otro lado está las ya conocidas a día de hoy como son la conexión a auriculares, sincronización de telefonía móvil en vehículos, etc.

Después de la exposición anterior y bajo la influencia de IoT, podríamos decir que esta tecnología **no** constituye la base purista del concepto de IoT porque los equipos no están conectados directamente a Internet y si no que están conectados básicamente a los teléfono móviles y estos a su vez envían la información a un servidor bajo una única dirección IP.

II.7.3 ZegBee

ZigBee es un estándar inalámbrico abierto con la capacidad de interaccionar con objetos de forma conjunta. Sus desarrolladores se han basado en el diseño abierto y global que sirva para proporcionar la base del ecosistema de IoT. Esta tecnología está actualmente en una posición dominante frente a otras para ser el estándar mundial de red, detección y control para su uso en aplicaciones domésticas, comerciales e industriales.

Esta tecnología inalámbrica irrumpió con fuerza en el año 2003 bajo el estándar IEEE 802.15[97] quedando totalmente definidas las especificaciones de nivel físico y control de acceso al medio. Esta tecnología avanzó en el concepto de redes de área personal con las ventajas de reducido consumo energético, topología mallada (lo que permite acceder a dispositivos interconectados por medio de red mesh[98]) y facilidad de integración.

En el año 2004 se publicó las especificaciones de la versión 1.0 de ZigBee, disponibles para el grupo de desarrolladores de la ZigBee Alliance[99]. A partir de ese año y hasta el año 2006 se crearon los productos que iban a ser comercializados. Desde ese año se ha producido la comercialización masiva de los dispositivos de comunicación inalámbrica con las especificaciones ZegBee alcanzando en los años 2014 y 2015 su nivel más alto de cuota de mercado a nivel mundial, impulsado por las distintas aplicaciones que los desarrolladores han hecho de esta tecnología. La arquitectura de ZegBee se basa en diferentes capas formadas definidas por el estándar como son la capa física y la capa del medio de control de acceso (MAC) y las capas definidas por ZigBee Alliance como son la capa de red y la capa de aplicación *(Baronti et. al, 2007), ver figura 7:*

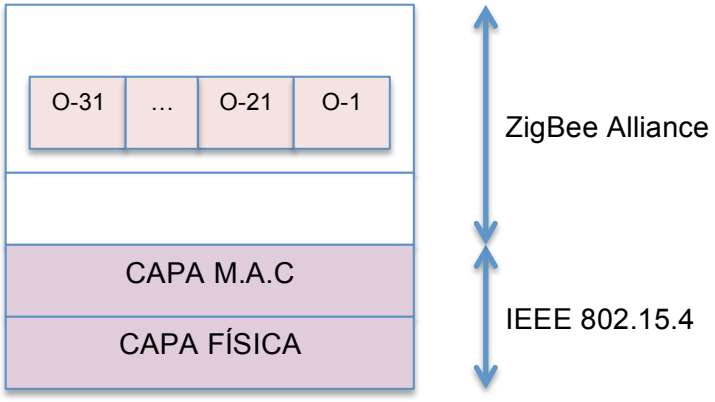


Figura 7. Capas que definen el protocolo ZegBee

[97] www.cse.wustl.edu/~jain/cse574-06/ftp/wpans/index.html (accesible el 19 de septiembre de 2015)
[98] www.meshdynamics.com (accesible el 19 de septiembre de 2015)
[99] www.zigbee.org (accesible el 19 de septiembre de 2015)

Estos protocolos están sustentados sobre algoritmos basados en red ad-hoc de baja velocidad de transferencia de datos (300 kBs). En la práctica, las topologías de red más extendidas son en estrella, árbol y mallada. La arquitectura de red mallada (mesh) es la más utilizada por la tecnología ZegBee porque permite una configuración automática de caminos de comunicación entre dispositivos de la red, tal y como se presenta en la figura 8.

Aparecen aquí la figura de nodo coordinador y nodo satélite (baliza) propio de esta arquitectura. Ésta permite que si un nodo, por donde se está realizando la comunicación dentro de la red mallada falla, la comunicación puede restablecerse de forma automática por otros caminos dentro de la red. Esta gestión de "caminos" la efectúa el nodo coordinador.

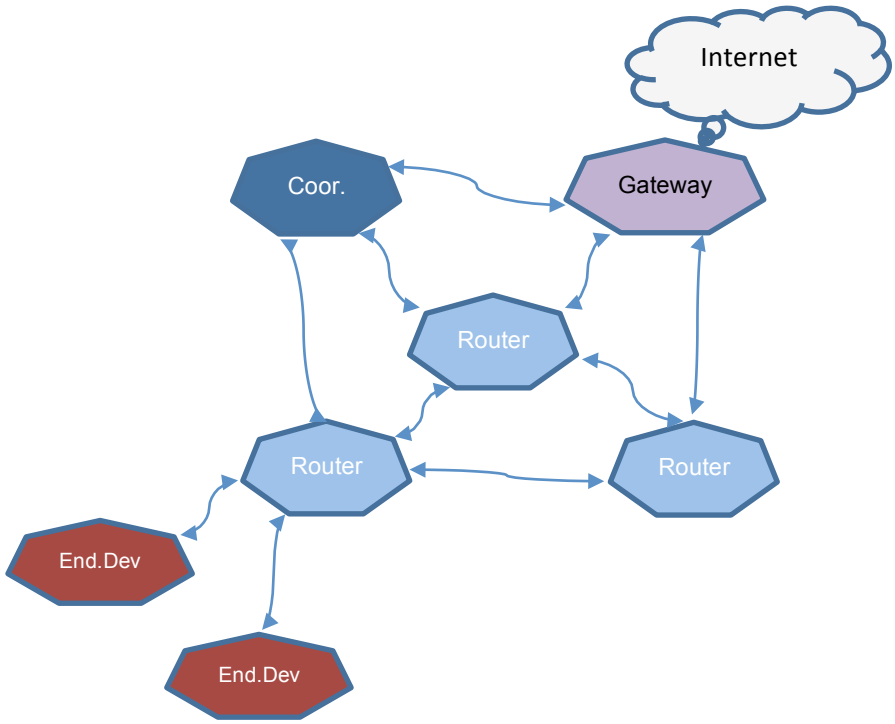


Figura 8. Topología de la red ZegBee. Comunicación entre dispositivos de la red

Tal y como se observa en la figura 8, los elementos principales de la red son el nodo coordinador sin puerto Ethernet, el nodo router, el nodo *End Device* y el nodo coordinado Gateway para salida a Internet. En una red, debe existir al menos un

nodo coordinador el cual es el encargado del control de la red y conexión entre dispositivos. El nodo router interconecta nodos aislados y los enruta dentro de la red mallada. Por otro lado, transfiere la información de usuario a la red. Los nodos End Device sólo tienen la capacidad para la comunicación con un nodo router o coordinador.

A nivel de capa OSI (figura 9), la ZigBee Alliance define más de 130 tipos de dispositivos y comandos relacionados para asegurar así que los dispositivos de diferentes fabricantes pueden trabajar juntos sin problemas y activar redes donde la más amplia variedad de dispositivos puede trabajar juntos en formas innovadoras para mejorar la comodidad y la eficiencia en los hogares y negocios.

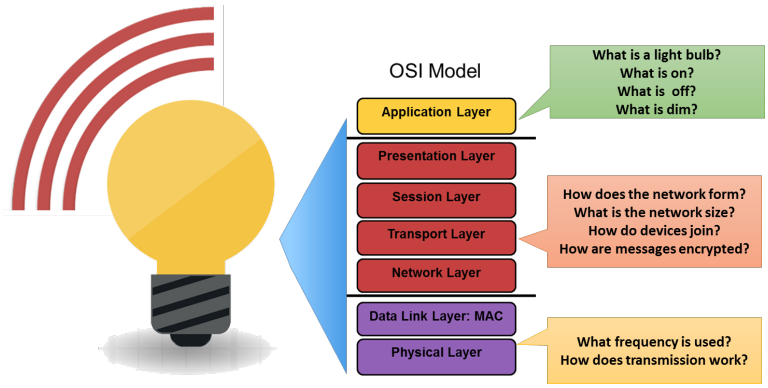


Figura 9. Nivel de capa OSI del protocolo ZigBee (fuente: www.zegbee.org)

II.7.4 ZigBee vs Bluetooth

Para aplicaciones de IoT, la tecnología ZigBee y Bluetooth proporcionan soluciones similares. Nos obstante presentan algunas ventajas e inconvenientes que se recogen en la tabla 8:

ZegBee	Bluetooth
Red de hasta 65535 direccionamiento de nodos en subredes de 255 nodos	Red de hasta 8 direccionamiento de nodos en una subredes piconet
Implementa la función "sleeping"	No implementa la función "sleeping"
Consumo medio en transmisión: 30mA Consumo medio en "sleeping": 3uA	Consumo medio en transmisión: 40mA Consumo medio en stand-by: 0,2 mA
Velocidad de transmisión: 250 kBit/s	Velocidad de transmisión: 3000 kBit/s
Uso en telefonía: No	Uso en telefonía: Si

Tabla 8. Principales diferencias entre ZigBee vs Bluetooth

Como resumen, se puede afirmar que una tecnología es más apropiada que la otra según la aplicación concreta. Por ejemplo, debido a la velocidad de transmisión, la tecnología ZegBee no puede ser utilizada para tráfico de audio y vídeo, más apropiado para la tecnología Bluetooth. En cambio, la tecnología ZegBee es ideal para productos dependientes de la batería dígase sensores.

II.7.5 Estándar 802.11: WiFi

Una de las tecnologías de comunicación inalámbrica más extendida, conocida y utilizada en los últimos años es la tecnología *Wireless Lan*[100] (WLan) o también conocida más técnicamente como estándar 802.11, aunque para el gran público se conoce como WiFi (*Wi-Fi Alliance*[101]). La tecnología WiFi nos permite conectar de forma inalámbrica multitud de dispositivos electrónicos como PC's, Smart TV's, Smartphones, video consolas, equipos de radio, etc., también tiene su utilidad en el ecosistema de la IoT. Todos estos dispositivos interconectados tienen en común que pueden conectarse a Internet a través de un punto de acceso tanto en interior, a través de un roúter o exterior como es el caso de la tecnología móvil o tarjetas M2M.

[100] www.standards.ieee.org/about/get/ (accesible el 19 de septiembre de 2015)
[101] www.wi-fi.org (accesible el 19 de septiembre de 2015)

En relación con la normativa, WiFi cumple con la norma estándar 802.11 en aspectos de redes inalámbricas de área local. Esta norma se subdivida a la vez en la 802.11b para velocidad de tráfico de datos hasta los 11 Mb/seg y la 802.11g que puede llegar hasta los 54 MB/seg.

La pregunta clave es ¿cómo puede esta tecnología integrarse en el ecosistema IoT para dar sentido a la solución establecida? Esta pregunta se responderá a través de los siguientes apartados.

II.7.6 Integración de la tecnología WiFi en el ecosistema IoT

En palabras de este doctorando, WiFi supone el tipo de comunicación por excelencia para ser integrada como parte del ecosistema IoT. Pensemos que tal es el grado de desarrollo tecnológico de este tipo de comunicación WiFi que está presente en la mayoría de las viviendas que disponen de Internet. Además es una tecnología con un alcance de cobertura que va desde los 20 metros en espacios cerrados a los 150 metros en espacios abiertos.

Podemos entonces considerar que la tecnología Wifi es trasversal a muchas verticales dentro del ecosistema IoT. Un ejemplo de ello lo encontramos en la publicación ***Trends in practical applications of agents and multiagents systems*** publicado en el 10th *International Conference on Practical Applications of Agents and Multi-Agent Systems*donde los autores Yves Demazeau, Jörg P. Müller, Juan M. Corchado Rodríguez y Javier Bajo Pérez, describen la tecnología Wifi como *"…una de las tecnología de comunicación inalámbrica para el uso de agentes virtuales (Virtual Agents in Next Generation Interactive Homes)".*

Respecto a la seguridad implantada en red, cada vez más se están haciendo esfuerzos en reforzar este punto que sin duda vulnera el desarrollo futuro de esta tecnología. Actualmente contamos con seguridad de la red basada en contraseñas, protocolos de cifrado (WEP, WPA y WPA2). Además, es útil para ciertas comunicaciones utilizar los túneles IP (IPSec).

Por otro lado encontramos los denominados Dispositivos de Distribución de Red (enrutadores, repetidores y puntos de acceso) y los **Dispositivos Terminales (Wifi-DT)** (tarjetas PCMCIA y USB). Dentro de los Wifi-DT y extendiendo su significado podemos englobar a aquellos dispositivos con capacidad de ser integrados en máquinas y conectarse por ejemplo a un punto de acceso. De esta forma se tendría la comunicación establecida con Internet del electrodoméstico a través del router y con una dirección IP a la cual apuntar desde el server en la nube.

II.7.7 Internet of Things presente en la industria

Los fabricantes de electrodomésticos están potenciando a su manera, el uso de máquinas cada vez con más funciones Smart, en definitiva más inteligentes. Hubo algunos fabricantes de electrodomésticos que en la década de los 90 introdujeron el concepto de **Control Fuzzy** *(Hájek, 2001)* en lavadoras. Hoy en día este concepto está fuera del ámbito de fabricación pero sin duda alguna se sigue utilizando en la industria de proceso.

En la actualidad existen diversas iniciativas de grandes fabricantes de electrodomésticos para dotar a sus productos de funcionalidad domótica en lo relativo al confort, el ahorro y las comunicaciones.

II.7.7.1 Electrodomésticos

- **Panasonic**[102]**,** ofrece soluciones con SmartCloud sistema propietario basado en modelo de "centralita". Este sistema está orientado al sector *retail* y comercio por medio de la tecnología Cloud en el que los sistemas conectados a Internet actualizan datos periódicamente de forma síncrona y asíncrona a través de peticiones de usuario bien a través de dispositivos móviles o PC. El ecosistema que integra esta solución es a través de interconexión de equipos a través del protocolo propietario de la marca y puerto de comunicaciones serie RS485. Una vez centralizada la información

[102] www.aircon.panasonic.eu (accesible el 8 de septiembre de 2015)

se envía al servicio en la nube para la gestión de los datos por parte del usuario.

- **Nest[103],** ofrece un termostato que puede funcionar como centralita y con los que se han hecho compatibles los electrodomésticos de Whirlpool. Este sistema se instala directamente en el interior de la vivienda y conecta de forma inalámbrica con aquellos electrodomésticos que dispongan de las funciones *Smart Assistant* y *Smart Stats*. A través de una aplicación móvil el usuario puede gestionar el funcionamiento del dispositivo a través de un panel de control. Las funciones están programadas lo que permite al usuario poder seleccionar aquella que tiene que realizar el electrodoméstico. Se puede considerar que el sistema que propone esta marca no está fundamentada en el concepto de IoT ya que el sistema Nest actúa de centralita la cual se conecta al roúter de la vivienda.

- **Samsung[104],** ha desarrollado una lavadora inteligente. La propuesta de esta marca sí que penetra directamente en la IoT. La lavadora tiene conexión a internet y el usuario puede controlar e informarse del estado del equipo a través de una aplicación propia del fabricante.

Según la investigación de la tecnología actual realizada por este doctorando, todos los esfuerzo están orientados a aplicaciones propietarias, tantas como fabricantes haya, lo cual **obliga** al usuario del electrodoméstico a tener la App del fabricante y tantas como electrodomésticos o familias de electrodomésticos tengan. En el fondo y como resultado esto no es práctico y por otro lado, podría darse el caso de que a la larga el usuario se canse de esta tecnología.

Para ello la presente tesis recoge los trabajos y avances técnicos conseguidos para que el control multimarca de cualquier electrodoméstico permita al usuario, con una sola aplicación, tener el control de los citados electrodomésticos basada en una la plataforma tipo chat denominada im4Things.

[103] www.whirlpool.com (accesible el 8 de septiembre de 2015)
[104] www.downloadcenter.samsung.com (accesible el 8 de septiembre de 2015)

II.7.7.2 Electrodomésticos

Según el Informe de Eficiencia Energética en el Hogar de Gas Natural Fenosa[105] de 2014, sólo el 1,4% de los hogares españoles dispone de sistemas de gestión energética, siendo los sistemas de control de calefacción, toldos y persianas e iluminación los más frecuentes.

Este informe proporciona una orientación de la capacidad del mercado español para recibir propuestas relacionadas con la domótica y la eficiencia energética ligada a los electrodomésticos. Esta situación corresponde a la media de los países de la Unión Europea y es extrapolable a otros países desarrollados.

Un concepto que está tomando fuerza es el de **Tecnodoméstico**, entendido como **Electrodoméstico Inteligente que puede ser controlado a distancia desde dispositivos móviles con sistemas operativos avanzados, como Android e iOS, conectado mediante redes inalámbricas**. A día de hoy, estos dispositivos ofrecen aplicaciones todavía básicas y de limitada utilidad pero de cara al futuro el control de estos tecnodomésticos a través del móvil será algo cotidiano y cambiará sustancialmente la forma de vida en el hogar.

Por medio de los avances tecnológicos recogidos en este trabajo de tesis doctoral, los fabricantes pueden tener en tiempo real información sobre su producto funcionando en condiciones reales, es decir, mientras es utilizado por el usuario. Además, una de las potencialidades de este desarrollo tecnológico que más interés puede suscitar, se enmarca en el mundo empresarial, máximamente en el ámbito del **marketing** y la **publicidad**. Esto es debido a que los datos se entienden como una oportunidad de cara a la creación de perfiles específicos de usuarios a los que dirigir la comunicación relacionada con la venta de productos o servicios. Un ejemplo de ello es el motor de búsqueda contenido en im4Things, por el cual se informa al usuario de la necesidad de comprar un determinado alimento y se le facilitará precio y ubicación de la mejor oferta, todo presentado en formato "carrito de la compra".

[105] www.gasnaturalfenosa.com/es/actividades/eficiencia+y+sostenibilidad/estudios+de+eficiencia+energetica/1297173567104/estudio+hogar+2014.html (accesible el 8 de septiembre de 2015)

Algo en común a estos avances tecnológicos incluyendo el presentado en esta tesis doctoral pasan por la creación de nuevas líneas de investigación, cuya temática gira en torno al transporte de datos en general y, en particular, a su aplicación de forma bidireccional. Es decir, el procedimiento óptimo para el transporte de la información basado en conversaciones entre usuarios-máquinas-usuarios en lenguaje natural así como notificación de eventos.

II.8 PROBLEMAS A RESOLVER EN ESTA TESIS DOCTORAL

Las tecnologías descritas hasta el momento están siendo desarrolladas a nivel mundial, viviendo una época en la que se vislumbran pequeños éxitos, que nos dan pié a pensar que serán tecnologías de obligado cumplimiento en un futuro no muy lejano. Hasta la fecha el desarrollo de la computación en la nube y el almacenamiento distribuido tienen una muy limitada aplicación en España, donde son las grandes compañías las que realmente están haciendo un extensivo uso y obteniendo los beneficios que se prometían. Tal es el caso de aplicación de los últimos avances en *Cloud Computing* que Telefónica en el año 2012 consiguió una reducción del 50% de los errores en las llamadas al servicio de atención al clientes (1004) gracias a la aplicación de un **análisis predictivo** sobre un volumen muy grande de datos[106].

Es probable que en un futuro cercano el nivel de desarrollo tecnológico de un país se mida a por medio de indicadores cuantitativos relacionados con el volumen de datos/anual que computa a través de las tecnologías *Cloud*. No será de extrañar que este volumen sólo se reduzca a computación de grandes datos sino que su relación sea directamente proporcional a pequeñas computaciones descentralizadas en viviendas y que los resultados de esta computación sean finalmente centralizado en un supercomputador. Es España tenemos aún mucho camino por recorrer. A nivel internacional IBM dispone de una nueva plataforma de servicios integrados denominada *SmartCloud Applications Services*[107] para servicios empresariales en la nube aportando un alto nivel de seguridad y control en la implantación y en el acceso a la información según afirma Antonio Raposo responsable del negocio Cloud de IBM[108]

Internet está posibilitando que usuarios y máquinas tengan acceso a la red global por medio del protocolo TCP/IP. Internet de las Cosas, trasversal a diferentes verticales de la técnica, amplia el concepto de conexión a máquinas, sensores y resto de cosas ya sea por la implementación del mismo protocolo

106 www.sorayapaniagua.com/2012/03/26/big-data-y-analisis-predictivos-en-el-1004-de-telefonica/ (accesible el 8 de septiembre de 2015)
107 www.ibm.com/cloud-computing/us/en/paas.html (accesible el 8 de septiembre de 2015)
108 www.revistacloudcomputing.com/2011/10/ibm-ayuda-a-las-empresas-a-aprovechar-la-oportunidad-de-negocio-de-cloud-computing/ (accesible el 8 de septiembre de 2015)

TCP/IP o por medio de los protocolos MQTT o XMPP.

Por otro lado, la comunicación entre usuarios y electrodomésticos está limitada al control por periféricos del tipo botonera, donde el usuario introduce la combinación de botones que en conjunto ejecutarán la función deseada. Algunos electrodomésticos denominados inteligentes, son capaces de comunicarse con el usuario a través de un Smartphone por medio de una App con comportamiento tipo Scada.

A medida que el número de electrodomésticos inteligentes comercializados aumenta, también lo hace el número de aplicaciones App para el control de éstos, por lo cual el usuario se encontrará con tantas aplicaciones de control Apps en proporción al número de electrodomésticos.

Respecto al desarrollo de los protocolos de comunicaciones actuales plenamente operativos para la IoT, aún no existe claramente un estándar a utilizar. Por su parte la organización IEEE ratificará en el año 2016 el estándar IEEE 802.11ah[109] para redes de sensores Wifi.

A nivel mundial el empleo de aplicaciones Apps como red social en formato Chat se están multiplicando cada año de forma exponencial. Tal es el caso de que el formato de comunicación más extendido es el empleo de aplicaciones como WhatsApp, Telegram, Line y similares.

Actualmente, la comunicación entre humanos y electrodomésticos es por medio de un interface basada, en la inmensa mayoría de casos, en una botonera. Como es sabido la botonera permite al usuario realizar las funciones de encender/programar/desconectar dicho electrodoméstico. Esta funcionalidad guarda una gran carencia sobre la información que dicho electrodoméstico puede ofrecer a un usuario como por ejemplo, indicaciones fallos de funcionamiento o alarmas.

El objetivo de la tesis doctoral es el desarrollo de un sistema de comunicación basado en el lenguaje natural tipo Chat en el que convivan tanto usuarios humanos como máquinas (más concretamente, electrodomésticos), intercambiando

[109]www.ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6364903&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber=6364903 (accesible el 8 de septiembre de 2015)

mensajería instantánea entre ellos, dar pie a que el electrodoméstico realice su rutina de funcionamiento e informe al usuario del Chat de los distintos estados que recorre en el su proceso normal de funcionamiento.

El sistema aquí propuesto en esta tesis doctoral deberá permitir:

i) Que diferentes usuarios y electrodomésticos interaccionen entre sí bajo la misma plataforma, el servicio de mensajería instantánea tipo Chat, de modo que un mensaje emitido por diferentes usuarios, en lenguaje natural, sea interpretado y ejecutado como una orden o consulta hacia el electrodoméstico.
ii) Que un electrodoméstico envíe información sobre eventos y estados a un usuario a través del Chat y en lenguaje natural.
iii) Que distintos electrodomésticos interaccionen entre sí intercambiando órdenes de control para la realización de una determinada tarea.
iv) Que sea seguro, robusto y sencillo de utilizar.
v) Que sea escalable.

Para cumplir los objetivos anteriormente descritos se plantearon los siguientes desarrollos:

- Desarrollo de un sistema electrónico con comunicación WiFi para embeber en él la base de inteligencia artificial. El sistema electrónico debe contar de una arquitectura con microprocesador modular y escalable, dotado de periféricos como son bloques de entradas y salidas dedicadas. Este dispositivo irá integrado en el electrodoméstico a controlar.
- Desarrollo de una aplicación de mensajería instantánea tipo Chat basado basada en las plataformas Android e iOS que sirva de interface entre el hardware a controlar y el usuario. La aplicación tendría una interface de escritura y reconocimiento de voz de forma que el usuario podrá escribir en lenguaje natural aquellas órdenes de control que desee dar a la máquina. De igual forma, la aplicación representará en lenguaje natural la respuesta a diferentes estados de la máquina, como por ejemplo eventos (finales de proceso) y alarmas.

- Desarrollo de sistema de diálogo o agente conversacional (también llamado Bot) basado en ontologías y el lenguaje AIML.
- Desarrollo de distintas bases de conocimiento y configuraciones de los sistemas de diálogo para distintos electrodomésticos que permita validar el sistema propuesto.
- Desarrollo del servicio de mensajería de alto rendimiento que proporcione el enrutamiento de los mensajes entre los distintos usuarios incluyendo el hardware como usuario. Para este servicio empleamos el protocolo XMPP de MongooseIM[110]. Por medio de este protocolo nos aseguramos que la comunicación es efectiva para millones de usuarios simultáneos.

Así pues, los distintos bloques temáticos de investigación que se han planteado en la tesis doctoral están basados en:

- **Arquitectura orientada a eventos.** Desde el momento que la prioridad es la transmisión de datos, de forma que se puedan definir el nivel de seguridad en la entrega, con esta línea se ha investigado y definido una arquitectura global diseñada para el tratamiento de eventos.
- **Segregación de responsabilidad comando-consulta** (*Command Query Resposability Segregation - CQRS).* Con esta línea se ha investigado en el modelo de separación de responsabilidades por el cual se mantienen repositorios distintos para los comandos y para las consultas. Este modelo ha sido ya impulsado por el nuevo advenimiento de los sistemas de Big-Data y de servicios de computación en la nube, en los que los modelos relacionales de almacenamiento se quedan insuficientes. La aplicación de esta arquitectura promete un verdadero avance en el aprovechamiento de los recursos computacionales, y en la facilidad de ampliación del sistema.
- **Desarrollo de micro-servicios.** En esta línea de investigación se ha buscado la manera de programar micro-servicios en un entorno distribuido.
- **Concentrador de mensajes** (*Message Broker*). Para facilitar el trasporte de datos en condiciones de calidad del servicio, lo más conveniente es aplicar

[110] www.mongooseim.readthedocs.org/en/latest/ (accesible el 8 de septiembre de 2015)

la tecnología de concentrador de mensajes. Esta tecnología permite un gran control de la comunicación, permitiendo desacoplar los productores de los consumidores, manteniendo la seguridad y proporcionando escalabilidad.

- **Transporte de encolado de mensajes XMPP**. Es un protocolo ligero de mensajería que implementa el patrón de integración publish/subscribe para sistemas con recursos limitados. Actualmente, esta tecnología se está convirtiendo en el estándar de facto para la Internet de las Cosas (IoT), por su facilidad y sencillez de implementación.

En España estamos muy por detrás en la aplicación de estas tecnologías, aunque se espera una respuesta muy amplia gracias a los éxitos cosechados a nivel internacional. Junto con las tecnologías que hemos destacado, se prevé el uso de un conjunto de herramientas que nos ayuden a la implementación de las técnicas descritas y poder dar respuesta a los objetivos planteados en esta tesis.

La presente tesis doctoral ha supuesto la oportunidad de investigar e innovar en nuevas tecnologías que están apareciendo de forma tímida en forma de estándares, al igual que otras que ya podríamos considerar estables y maduras.

Capítulo III. Sistema de diálogo basado en mensajería instantánea para el control de aparatos en el Internet de las Cosas (IoT)

III.1 INTRODUCCIÓN

Im4Things es un sencillo y a la vez novedoso sistema que mezcla las disciplinas de Informática y Electrónica para permitir la comunicación bidireccional entre usuarios y aparatos en lenguaje natural. El medio elegido para esta comunicación es una red social sustentada en formato App para plataformas móviles: smartphones y tabletas. El usuario podrá descargar de los distintos markets actuales dicha aplicación. Un usuario puede enviar órdenes de control a un electrodoméstico, por ejemplo una cafetera, para que se encienda o haga café simplemente enviándole el mensaje: "cafetera quiero café" a través de una aplicación de mensajería móvil. Como se ha comentado anteriormente, la finalidad de esta tesis doctoral es el desarrollo de una herramienta que proporcione una interfaz de comunicación entre humanos y dispositivos en la Internet de las cosas mediante diálogo en lenguaje natural escrito a través de servicios de mensajería instantánea. Esta comunicación puede ser de distintos tipos tales como el envío de órdenes, la consulta del estado e incluso se permite que sean los mismos dispositivos los encargados de alertar al usuario, si se ha producido un cambio del estado en los sensores de los dispositivos. Para alcanzar este objetivo, se han utilizado distintos protocolos entre los que podemos destacar: XMPP y AIML que han extendido con nuevas funcionalidades para ser adaptados a la IoT. En concreto, las tecnologías destacadas que se han utilizado para el desarrollo del sistema propuesto son:

- XMPP: Protocolo base para la comunicación de las distintas entidades del sistema. Sobre este protocolo se han desarrollado extensiones para permitir la interacción de los usuarios con las máquinas, así como las máquinas entre sí.

- AIML: Lenguaje utilizado para implementar la inteligencia artificial que interactúa con el usuario en lenguaje natural.

Este capítulo representa la parte central de esta tesis doctoral. El objetivo del mismo es describir una nueva metodología para la integración de dispositivos hardware en el ecosistema de la Internet de las Cosas por medio de un modelo ontológico predefinido. El capítulo explica en profundidad la arquitectura del sistema tanto hardware como software desde el punto de vista de la ingeniería de protocolos expresados aquí, así como las técnicas para el registro y explotación de aparatos por parte de un usuario a través de la tecnología móvil y una aplicación app empleando comandos en lenguaje natural mediante servicios de mensajería instantánea.

III.2 DESCRIPCIÓN DEL PROBLEMA Y OBJETIVOS

Durante la redacción de esta tesis doctoral, cabe destacar que los protocolos destinados al ecosistema de Internet de las Cosas, más concretamente XMPP y AIML, no ofrecen soluciones completas para resolver de forma robusta y segura el control de dispositivos y sistemas electrónicos a través de Internet y más concretamente a través del protocolo HTTP. Por otro lado, en relación a la tecnología inalámbrica, principalmente Bluetooth y Wifi, actualmente existe una formalidad robusta del protocolo de comunicación pero no está desarrollado el estándar para la gestión de medios, y más concretamente en la parte del envío y registro de credenciales desde un servidor en la nube.

Del capítulo anterior del estado del arte podemos concluir que, según nuestro conocimiento no existe ninguna red social en formato de chat de mensajería instantánea que permita la comunicación en lenguaje natural entre usuarios y máquinas.

La presente tesis doctoral tiene como objetivo el desarrollo de los medios y protocolos necesarios para el control, por medio del lenguaje natural, de dispositivos equipados con tecnología inalámbrica Bluetooth y Wifi destinados a su uso en electrodomésticos, por medio de una red social en formato de chat. Para lo cual resolveremos los siguientes problemas:

- Desarrollo de un software que permita interpretar las instrucciones en lenguaje natural e interactuar con los dispositivos. Este software lo denominamos Bot ya que la definición de Bot está extendida y se refiere a un programa informático que imita el comportamiento de un humano. En el caso de esta tesis doctoral este Bot interpreta los mensajes y órdenes en lenguaje natural e interactúa con el dispositivo.
- Desarrollo de medios de comunicación de la aplicación móvil de chat para la interacción usuario-usuario, usuario-Bot y Bot-Bot.
- Desarrollo de medios de restricción de acceso y autenticación desde un servidor en la nube, hacia y desde el Bot.

- Desarrollo de medios para configuración del Bot desde la aplicación cliente. Esta configuración debe permitir a los usuarios añadir dispositivos al sistema y establecerse como administrador, así como establecer su configuración inicial.
- Creación de las bases del conocimiento necesarias para la interacción User-Bot.
- Adaptación de los protocolos XMPP y AIML para su uso en el ecosistema de Internet de las Cosas.

La resolución de los problemas expuestos arriba permitirá una innovación y avance en los siguientes campos:

- Medios en el servidor que impidan el registro fraudulento de Bots en el sistema.
- Medios en los Bots que permitan ejecutar acciones y generar respuestas a partir de mensajes en lenguaje natural recibidos desde el dispositivo móvil del usuario.
- Medios que permitan a los usuarios obtener el estado actual de un Bot, así como los comandos que puede realizar.
- Medios que permitan a un usuario modificar la lista de control acceso de un Bot, permitiendo a otros usuarios o Bots interactuar con él.
- Medios que permitan a un usuario actualizar el perfil de un Bot.
- Medios que permitan a un Bot descubrir si otro Bot implementa una interfaz concreta, estableciendo una relación entre ambos para una posterior comunicación.
- Medios que permita a un usuario la actualización de la base de conocimiento de un Bot.
- Medios que permita a un usuario resetear un Bot a su configuración inicial.

III.3 ARQUITECTURA DEL SISTEMA Im4Things

En este apartado se describe la arquitectura del sistema Im4Things propuesto en esta tesis doctoral. La figura 3.1 muestra la arquitectura general de la plataforma Im4Things.

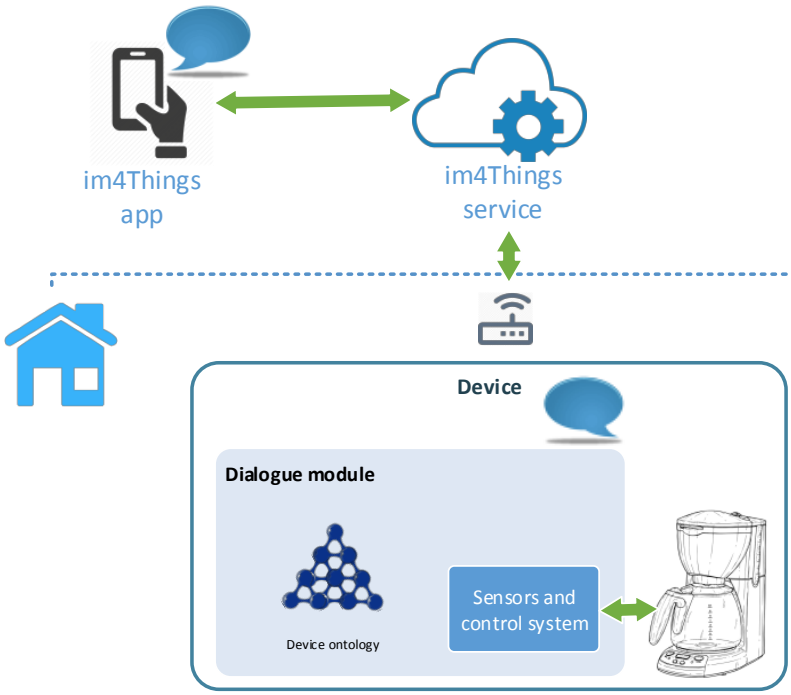


Figura 10. Arquitectura general de la plataforma Im4Things

La solución propuesta en esta tesis doctoral se compone de tres módulos principales: Im4Things app, Im4ThingsCloudService y el Dispositivo (ver figura 3.1). La Im4Things app implementa un chat de mensajería instantánea a través del cual los usuarios pueden enviar y recibir mensajes, archivos multimedia y formar grupos (usuarios-usuarios, usuarios-Bots y Bots-Bots) desde el servicio en la nube Im4ThingsCloudService. El servicio en la nube Im4ThingsCloudService es el encargado de mantener la seguridad, gestionar la comunicación de mensajería instantánea y resto de servicios multimedia entre las Im4Things apps y los dispositivos. Dentro de cada dispositivo se encuentra el Bot que implementa un módulo de diálogo inteligente que permite ejecutar comandos y consultar el estado actual del mismo. Este módulo también permite detectar alertas del dispositivo, transformarlas a lenguaje natural y enviarlas al servicio Im4ThingsCloudService.

Un ejemplo de esta alerta puede ser que la ejecución de un comando ha finalizado. A continuación, se describe cada módulo en detalle.

III.3.1 Aplicación de usuario: Im4Things app

Esta aplicación móvil permite una comunicación mediante mensajería instantánea entre usuarios y dispositivos en tiempo real. Esta comunicación se realiza utilizando el protocolo XMPP que permite gestionar sesiones seguras de mensajería instantánea

En la figura 11 se ilustran los distintos módulos que conforman la **aplicación de usuario Im4Things app** que se ejecuta en un dispositivo móviles (smartphone/tableta) de los usuarios.

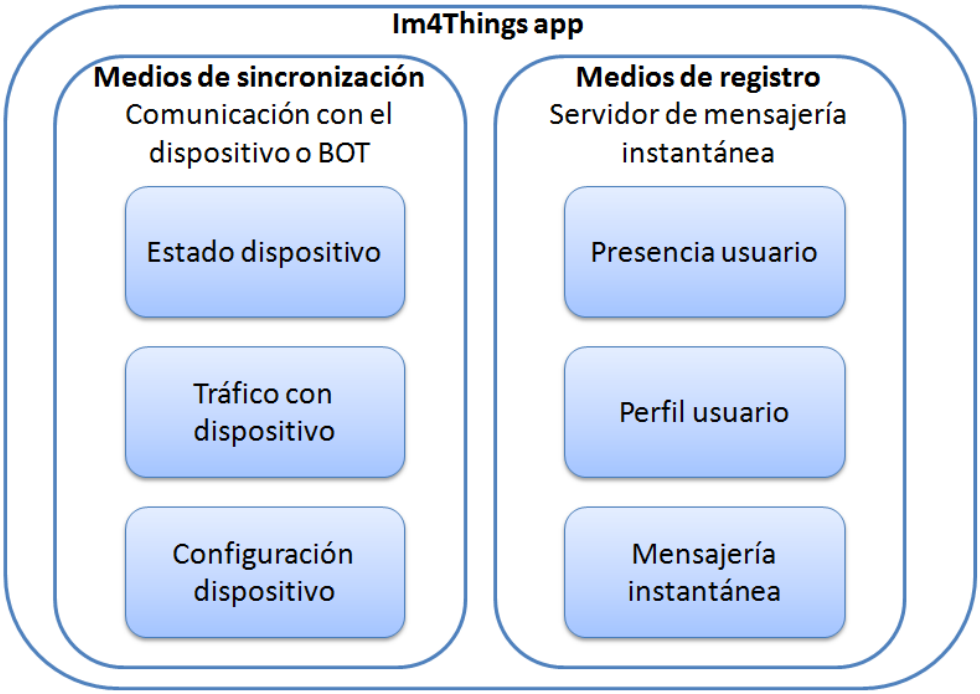


Figura 11. Arquitectura funcional de la aplicación im4Things app

Como muestra la figura 11 podemos distinguir las funcionalidades de esta app en dos grandes bloques: los medios de sincronización (más relacionados con la comunicación con el dispositivo o Bot) y los medios de registro (más relacionados con la aplicación de mensajería instantánea).

Para los medios de registro se han aprovechado otras aplicaciones como *Telegram*[111], que contempla las funcionalidades siguientes**:**

- Presencia usuario. Esto significa que todas las entidades de Im4Things (usuarios y bots) pueden proteger su privacidad a través del concepto de presencia. Esto significa que una entidad puede otorgar privilegios a otras entidades para que puedan subscribirse a su presencia y de esta manera acceder a distintas funcionalidades como saber si una entidad está actualmente conectada al servidor, saber cuándo fue la última vez que se conectó, o poder enviarle una petición para que sea procesada en la otra entidad. Esto es necesario para los grupos de chat de forma que un usuario autorice a un bot a participar como miembro de un grupo determinado, por ejemplo el compartido con familiares o amigos. De esa forma cada usuario que comparte el grupo puede comunicarse con el electrodoméstico que ha sido invitado a dicho grupo.
- Perfil usuario. Esto significa que todas las entidades de im4HiThings pueden configurar un perfil personal en el que establecer un nickname y un avatar, para posteriormente compartirlo con otra entidad (generalmente un usuario), y que le resulte más amigable a la hora de establecer una conversación con esa otra entidad. Por ejemplo un perfil de usuario podrá constituirlo un nombre de usuario y una fotografía, similar a como lo hacen otros chat como Telegram o WhatsApp.
- Mensajería instantánea. Una de las funcionalidades que ofrece im4HiThings es la mensajería instantánea, permitiendo a las distintas entidades del sistema enviar tanto mensajes individuales como mensajes a grupos. Esto permite por un lado la comunicación entre usuarios de im4HiThings, así como el envío de comandos en lenguaje natural a los bots que haya registrado un usuario en el sistema. El tráfico de información es bidireccional entre usuarios y bots. Por un lado el usuario envía una orden al electrodoméstico para hacer alguna tarea y el electrodoméstico envía al usuario información sobre el estado de esa tarea. Esto ultimo es lo que denominamos “estado del dispositivo”.

[111] https://telegram.org (accesible el 3 de octubre de 2015)

Por otro lado, el bloque de medios de sincronización se ha desarrollado desde cero, diseñando un nuevo protocolo propio que implementa las siguientes funciones:

- Estado del dispositivo. El bot facilita a sus usuarios autorizados información acerca del sistema, como por ejemplo si está actualmente conectado, si está en proceso de realizar alguna tarea, los parámetros que tiene programados para realizar dicha tarea, etc. Esta información es proporcionada al usuario tanto por lenguaje natural a través de la inteligencia artificial como por la interfaz de la aplicación móvil. Esta interface es básicamente un panel de control amigable donde están definidas las distintas tareas que puede realizar un electrodoméstico a través de la cual la interacción con la máquina es a través de botones.
- Tráfico con dispositivo. Este bloque permite al usuario consultar las distintas tareas que es capaz de realizar un bot concreto, por ejemplo una cafetera, así como interactuar con él para darle órdenes al electrodoméstico para que ejecute una tarea determinada. De nuevo, esto puede realizarse tanto por lenguaje natural como por la interfaz de la aplicación móvil.
- Configuración del dispositivo. En su estado inicial el bot debe ser configurado con los datos necesarios para poder registrarse en el sistema, como pueden ser el medio de conexión a la red, la identidad de su usuario administrador, o su apodo y foto en im4HiThings. Este módulo hace posible dicha configuración desde la aplicación móvil. También se encarga de actualizar el bot a una nueva versión o reiniciarlo a su estado inicial.

La aplicación "Im4Things" app se ha desarrollado para los sistemas operativos Android e iOS. En 2014, estos dos sistemas operativos aglutinaban el 96,4% del mercado de los *smartphones* según un estudio de mercado publicado por el portal tecnológico Xataka. Como se ha comentado anteriormente, la aplicación Im4Things es una red social en formato de chat de mensajería instantánea. Desde este chat el usuario puede realizar distintas funciones que se listan a continuación:

Respecto al chat de mensajería instantánea se pueden realizar distintas operaciones como las siguientes:

- Alta de usuario. Im4Things app se ha diseñado para que los usuarios se puedan dar de alta en el chat a partir de los datos de las cuentas Facebook y Google+. De esta forma cada usuario puede heredar dichos datos a Im4Things app facilitando así el paso previo de alta. Esta forma de alta en nuevas aplicaciones está muy extendida por ejemplo para la fase de registro de suscripciones webs.
- Envío de mensajería instantánea entre usuarios del chat. La aplicación Im4Things permite el envío de mensajería instantánea como núcleo de comunicación entre ellos. Además de mensajería de texto, la aplicación permite el envío de emoticonos muy extendido en otras aplicaciones.
- Formar grupos de usuarios. A través de la función crear grupos, la aplicación Im4Things permite la multidifusión de mensajes instantáneos, emoticonos y datos multimedia entre los usuarios de un grupo.
- Envío y gestión de datos multimedia. Un usuario puede enviar datos multimedia a otro usuario de forma privada o a través de un grupo. Estos datos multimedia son archivos de voz, imágenes y video previamente archivados en el terminal móvil o grabar y enviar dichos archivos en tiempo real.

Respecto a las funciones que permiten interactuar con el dispositivo o Bot podemos destacar las siguientes:

- Configuración y alta de dispositivos. Esta parte de la tesis resuelve por primera vez el problema de cómo dar de alta un dispositivo (no usuario) en una red social en formato chat. La configuración de los dispositivos se realiza en dos pasos: el primero es identificar el dispositivo y el segundo es introducir los parámetros de configuración. Cuando se enciende el dispositivo por primera vez, este transmite de forma automática y por bluetooth, un identificador único programado previamente para que el usuario sincronice su teléfono móvil al dispositivo. Una vez conectado al electrodoméstico por bluetooth, el usuario accederá a un menú definido tipo formulario para introducir los datos de configuración que son fundamentalmente la dirección IP a la que conectarse, seguridad de red y el nombre que del electrodoméstico. Cuando estos datos están introducidos, el

electrodoméstico se da de alta automáticamente en la base de datos Im4ThingsCloudService.

- Envío de mensajería instantánea entre usuarios y/o dispositivos. Una vez el dispositivo está dado de alta en la base de datos, es cuando el usuario puede invitar a este dispositivo a intercambiar mensajes instantáneos entre usuario-dispositivo. El tráfico de estos mensajes son el lenguaje natural tanto del usuario hacia el dispositivo y viceversa. En el caso de tráfico de información entre dispositivos, estos se realizan de forma automática por medio de eventos, es decir, estados que cambian en cada uno de ellos y que lleva asociado una actuación. Por ejemplo, un evento entre dispositivos podría ser el escenario de alarma por inundación, el cual el sensor envía un mensaje a una válvula de corte con la orden de cerrar. Este tráfico de mensajes entre dispositivos queda reflejado en el chat del usuario por medio de mensaje instantáneo en el chat.
- Formar grupos de dispositivos. De igual forma que en el punto anterior, im4Things puede formar chat entre dispositivos. Estos grupos de dispositivos los crea el usuario y también él decide qué dispositivos pertenecen a ese grupo. En ese chat estará de forma permanente el administrador del grupo de forma que toda la información intercambiada por los dispositivos se le notifica al administrador en forma de mensaje.

Los medios de sincronización empleados se basan en el tráfico de información **<send contacts>+<get contacts from DB>** tal y como lo implementa la API de Google para la gestión de contacto[112]**.** En este caso se ha desarrollado una nueva API denominada HiThing API para la sincronización de contactos. La sincronización de los contactos es necesaria para que el usuario conozca qué contactos propios están registrados en el servicio Im4ThingsCloudService, a través de los medios de registro, y pueda intercambiar mensajería con ellos. El procedimiento es como sigue: cuando un usuario descarga la aplicación im4Things app se le solicita los datos de contacto que son el número de teléfono, nombre (username), nick y foto (opcional). Cuando los datos de contacto han sido

[112] https://developers.google.com/google-apps/contacts/v3/reference (accesible el 7 de octubre de 2015)

registrados, el servicio Im4ThingsCloudService le solicita también que sincronice la agenda de contactos clasificados por email o teléfono. De esta forma el usuario obtiene del servicio (base de datos) cuáles de esos contactos están registrados en nuestro servicio Im4ThingsCloudService. De esta forma el usuario recibe el *username* de cada usuario que tiene en la agenda para que sean invitados al chat y poder comunicarse con ellos. En la figura 12 se detalla la secuencia del servicio de sincronización

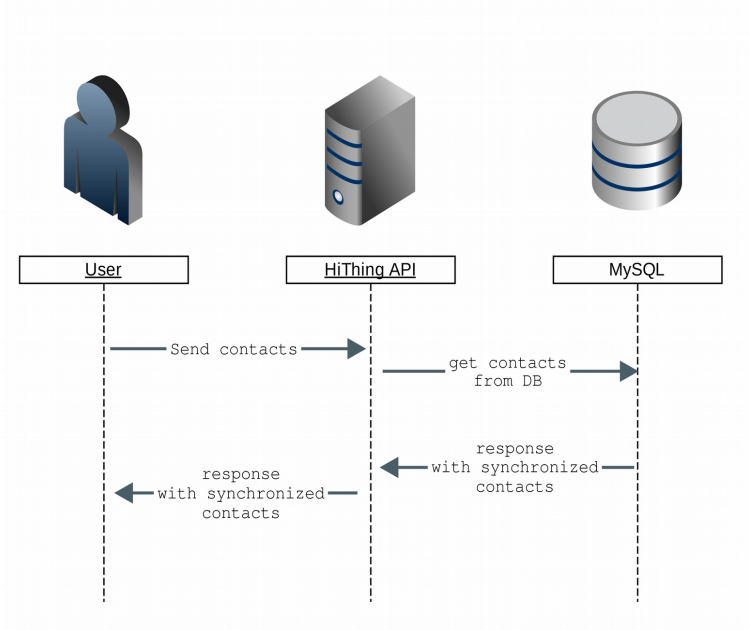


Figura 12. Diagrama de secuencia del servicio de sincronización

III.3.2 Aplicación servidor: Im4ThingsCloudService

El objetivo de este servicio es la gestión y el mantenimiento de la comunicación de mensajería instantánea entre usuarios y dispositivos utilizando un servicio seguro MongooseIM[113]. El servicio MongooseIM es una plataforma de servicios integrados para la construcción de sistemas de mensajería instantánea aprovechando las cualidades del protocolo XMPP (utilizado por Facebook y Google Talk). Esta plataforma está gestionada en el servidor del sistema. Este servicio está especialmente pensado para dar soporte de comunicaciones a segmentos creciente como son la social media, juegos y servicios de telecomunicación. Es escalable y

[113] www.erlang-solutions.com/products/mongooseim-massively-scalable-ejabberd-platform (accesible el 9 de octubre de 2015)

por ello es capaz de proporcionar una comunicación a millones de usuario. Por otro lado, este servicio es de código abierto lo que ha supuesto para este proyecto una ventaja al no tener que comprar licencias adicionales para el desarrollo de la mensajería instantánea.

Un aspecto muy importante para el proyecto que ha sido determinante ha sido que el servicio MongooseIM dispone de diversos módulos o paquetes de software en formato API que está a disposición de los desarrolladores. Entre ellos cuenta con los módulos chat multiusuario, Websockets y de privacidad. Además, la compatibilidad con bases de datos en concreto con MySQL ha sido muy ventajoso porque con los módulos antes comentados y la comunicación con la base de datos MySQL se ha conformado todo el sistema server del proyecto.

Por tanto los dispositivos como los usuarios (clientes) utilizan los módulos de la plataforma MongooseIM chat multiusuario, Websockets y privacidad. Con ello se ha confeccionado la HiThingAPI que comunica la im4Things app con la base de datos MySQL para la gestión de los datos de usuario.

Este servicio se adaptó para ser integrado con otras funcionalidades como el registro de usuarios, sincronización, registro de dispositivo, etc.

En la figura 13 se ilustran las distintas funcionalidades que conforman la **Aplicación de servidor** llamada Im4ThingsCloudService.

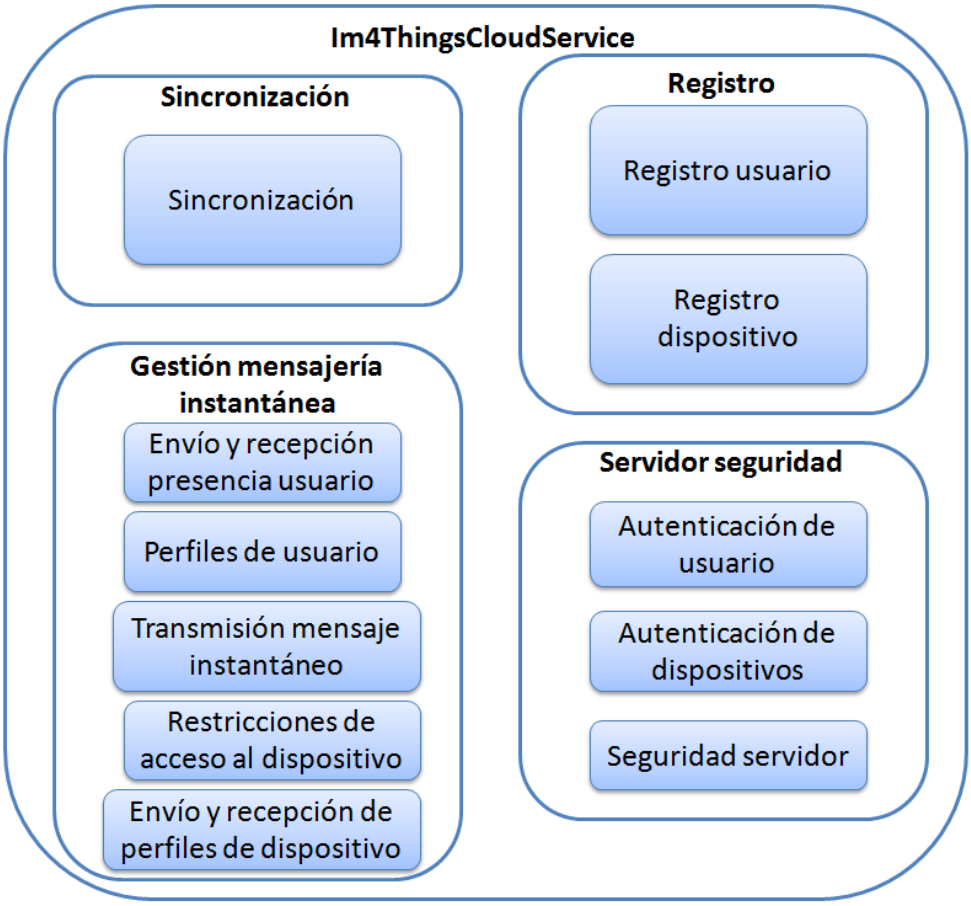


Figura 13. Arquitectura funcional del servicio Im4ThingsCloudService.

Para implementar la mensajería instantánea se ha utilizado el protocolo **XMPP**. El servicio principal que maneja este protocolo es **MongooseIM**, que es un ***fork***[114] (una versión refactorizada) del servicio **Ejabberd**. Los servicios que ofrece la plataforma Ejabberd, entre ellos está **MongooseIM** son idóneos para los servicios de mensajería masiva probada para sistemas críticos y escalables. Estas dos cualidades fueron determinantes para elegir esta plataforma para la realización de este proyecto porque como trabajos futuros es la explotación comercial del producto final y ello presupones ofrecer servicio de mensajería a miles de usuarios y dispositivos.

[114] Creación de un proyecto en una dirección distinta de la principal u oficial tomando el código fuente del proyecto ya existente.

En nuestro caso hemos desarrollado una extensión del protocolo XMPP para la elaboración de los formularios de datos necesario para la configuración de cada servicio a petición de los clientes (usuarios y dispositivos). El conjunto de peticiones las clasificamos en función de si procede de un usuario o de un dispositivo. En el caso de que la petición proceda de un usuario se interpreta como mensaje instantáneo y se transmite como tal por el servicio MongooseIM. En caso de que la petición procesa de un dispositivo, se interpreta como un mensaje para la ejecución de una tarea hacia otro dispositivo o como mensaje informativo hacia el usuario. En todo caso, esta tarea recae sobre los servicios de MongoosIM chat multiusuario y Websockets[115].

Una de las principales dificultades que se ha tenido que resolver en nuestro proyecto ha sido el procedimiento por el cual un dispositivo recibe un mensaje de un usuario fuera de la red Ethernet de la vivienda, es decir, cuando el mensaje procede directamente del servidor Im4ThingsCloudService. El problema lo hemos resuelto de la siguiente forma. Cuando el dispositivo se configura, tiene programado por defecto una IP a la cual enviar la información de registro. La información que envía al servidor Im4ThingsCloudService contiene la información de los sockets. Nuestra arquitectura está basada en cliente-servidor siendo el cliente el dispositivo y el servidor claramente nuestro servicio Cloud Im4ThingsCloudService.

La solución anterior es parcial, sólo válida para roúters con IPs fijas. La realidad es que las IPs de los roúters domésticos pueden ser variables. Para que un dispositivo pueda comunicarse con el Im4ThingsCloudService en todos los casos se ha implementado en el dispositivo un Websocket. Con esto garantizamos un canal de comunicación bidireccional cliente-servidor.

Este servicio funciona con módulos que se pueden crear, añadir y eliminar dependiendo de los servicios que se quieran proporcionar con este protocolo. Además de este servicio se ha implementado una arquitectura especialmente adaptada para llevar el control de los usuarios de la aplicación (ver figura 14). El lenguaje de programación que se utilizó para los servicios es **Erlang**.

[115] www.datatracker.ietf.org/doc/rfc6455/?include_text=1 (accesible el 10 de octubre de 2015)

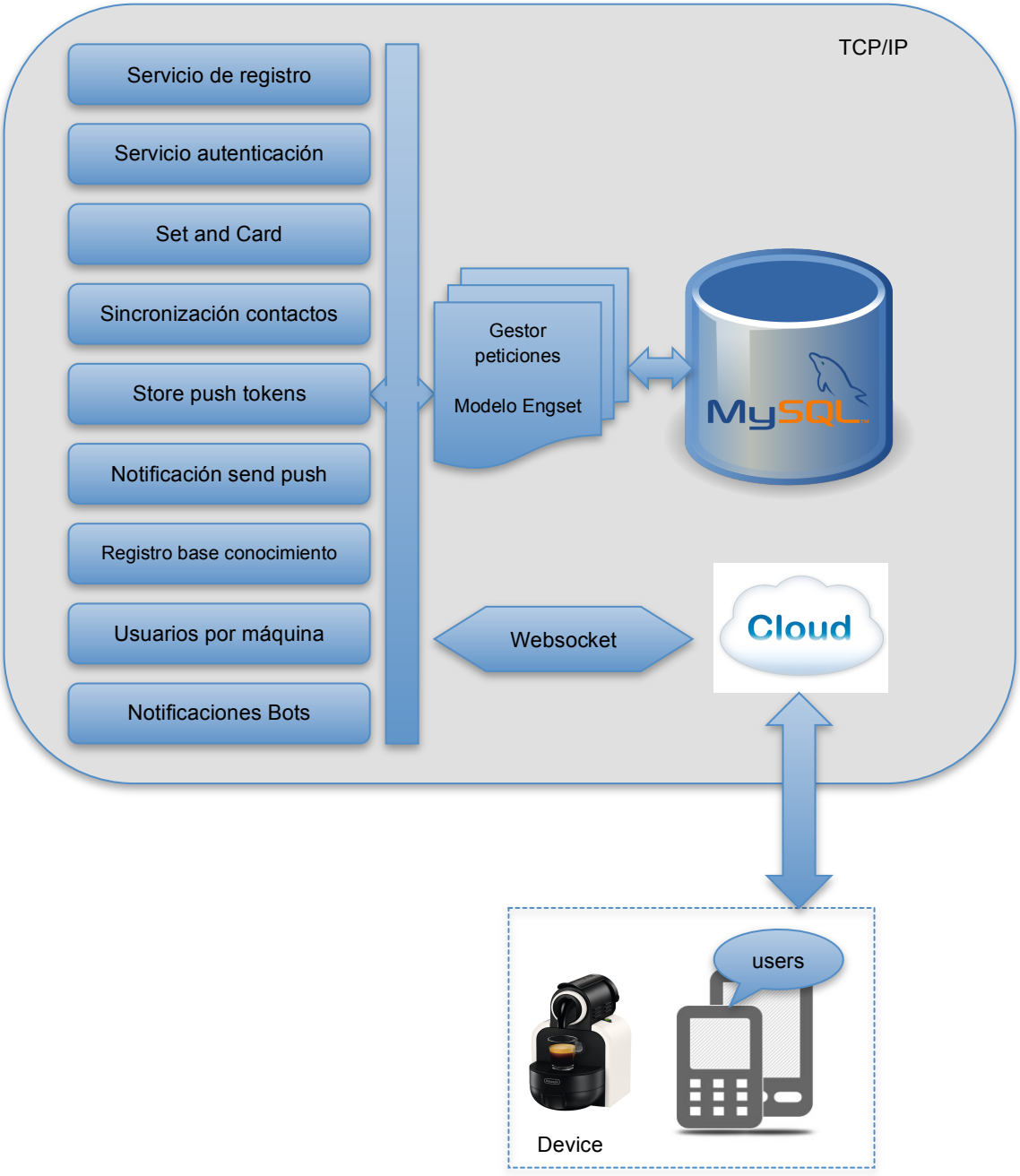


Figura 14. Servicios desarrollados para el control de los usuarios de la im4Things app

La decisión de utilizar el lenguaje de programación Erlang para nuestro proyecto es que es código abierto y por lo tanto nos permitió realizar las adaptaciones pertinentes dentro del código base. Además Erlang está construido como un lenguaje de programación concurrente, es decir óptimo para la ejecución de varios procesos en paralelo como es nuestro caso ya que la demanda de servicios está prevista que sea de miles de usuarios simultáneos. Por otro lado, Erlang ha sido y es utilizado fuertemente por la comunicada de desarrolladores lo

cual ha supuesto una gran tolerancia a fallos de ahí que Erlang esté presente en servicios informáticos para banca, comercio electrónico y telefonía. En nuestro proyecto Erlang ha resuelto con éxito la demanda en tiempo real de una gran demanda de peticiones y servicios entrono a la mensajería instantánea, todo en tiempo real.

En general han sido de gran utilidad las librerías de Erlang, denominadas OTP[116], y en particular las relacionadas con la bases de datos, interfaces, comunicaciones y test.

A continuación se detallan cómo se realizan cada una de las funcionalidades mostradas en la figura 14.

III.3.3 Servicio de autenticación

Una de las funcionalidades más importantes es poder identificarse en el servicio para poder empezar a utilizar la mensajería instantánea. Para el servicio de autenticación podríamos emplear dos técnicas: autenticación en el servidor almacenando la sesión del servicio o autenticación sin estado con tokens[117]. Decidimos finalmente emplear la técnica de la autenticación por medio de tokens y más concretamente el que emplea Google denominado token de Google. Un token es una firma digital cifrada que permitirá a nuestro servidor identificar al cliente. En el primer caso, la autentificación por sesión no valdría para nuestra aplicación porque un usuario de im4Things app puede abrir y cerrar un número indefinido de veces la aplicación de modo que cada vez que abra una nueva sesión la plataforma le pedirá que se autentique. Para un chat tipo red social no es práctico además, no lo emplea ninguna de las redes social en formato chat que conocemos a día de hoy. Lo que nosotros hemos hecho es que la información que se envía a través del token es el identificador único e inconfundible del usuario, más concretamente, al que identifica a la API instalada en el terminal móvil del usuario o del dispositivo.

El funcionamiento es como se detalla a continuación. El usuario o dispositivo se registra en nuestra plataforma empleando para ello el nombre de usuario

[116] www.erlang.org/doc/applications.html (accesible el 9 de octubre de 2015)
[117] www.en.wikipedia.org/wiki/Google_Authenticator (accesible el 9 de octubre de 2015)

(*username*) y contraseña, o a través de un proveedor de servicios que en nuestro caso hemos seleccionado Google+ (tal y como se detalla en el punto III.5.2.). A partir de entonces, cada petición de sesión que haga el usuario/dispositivo va acompañada de un la firma digital (token) en la cabecera del protocolo. El token no se almacena en el servidor, si no en el lado del cliente y es el API Rest de im4Things app el que se encarga de descifrar ese token y finalizar el procedimiento de identificación e iniciar la sesión para el envío de mensajería u otros datos en un sentido u otro. Este procedimiento también aporta seguridad a nuestro sistema.

Como se muestra en la figura 15 el usuario primero envía su *nombre de usuario* y *contraseña*, después el servicio comprueba si los datos son correctos y si es así, responde con un token de sesión que se almacena en el API Rest im4Things app.

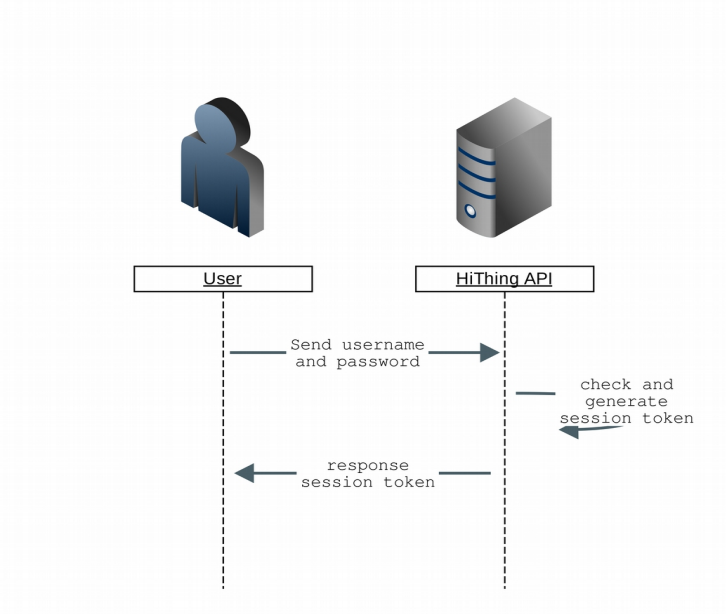


Figura 15. Diagrama de secuencia de negociación del token para la autenticación

Una API Rest[118] (*REpresentational State Transfer*) es un tipo de arquitectura de desarrollo web que se apoya totalmente en el estándar HTTP. REST nos ha permitido crear los servicios y aplicaciones usadas por los dispositivos como clientes basados en http.

[118] www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm (accesible el 9 de octubre de 2015)

III.3.4 Servicio de registro

Los usuarios pueden registrarse haciendo uso de esta funcionalidad. Cuando el usuario descarga del market la aplicación im4Thing app, el siguiente paso es su instalación en el dispositivo móvil. Una vez instalada el usuario dispone de dos formas de registrar la aplicación en el servidor. Una es a partir de redes sociales en concreto Google+, es decir es Google+ el proveedor de los datos de registro y la otra posibilidad es el registro por correo electrónico.

Los medios de registro se hacen a través del servicio Im4ThingsCloudService en el cual se envía información del usuario a registrar mediante varias formas que se detallan a continuación:

i) Mediante email:

En la figura 16 se muestra un diagrama de secuencia con el funcionamiento del registro mediante email. Como se puede observar, el usuario primero envía la información de su email, entonces el servicio envía un correo electrónico a dicho email para verificar que la cuenta es válida. Una vez que el usuario recibe el mensaje, entonces accede a un enlace para poder habilitar ese usuario con un código de verificación. Se chequean los datos y si son correctos se procede a registrar al usuario.

En relación a la base de datos, ésta se ha estructurado por medio de arrays de enlaces simbólico[119] siguiendo el modelo:

[website root)/components/com_emails => (code root)/com_emails]

A continuación se presenta un extracto de código el servicio de registro y organización de la base de datos:

```
$data = array(
   'fields' => array(
      'id' => 'user',
      'nickname'  => 'username'
   ),
   'component'  => 'com_foo', // identificador de componente, hace referencia a la
```

[119] www.man7.org/linux/man-pages/man2/symlink.2.html (accesible el 10 de octubre de 2015)

```
columna de los componentes de la table de plantilla utilizada
   'type'      => 'bar',    // identificador de tipo, hace referencia a la columna de tipo en el
cuadro plantilla
   'recipients' => array('user@im4Thinghs.com')
);

// Obtener controlador de correo electrónico y enviar el correo electrónico
$this->getService('com://site/emails.controller.email')->send($data);

// Tablas de la base de datos

jos_emails_templates

jos_emails_layouts

jos_emails_emails

preg_match_all( '/\{\{([^\}]*)\}\}/', $body, $matches );

                foreach( $matches[0] as $i => $match )
                {
                        $body = preg_replace( '/' . preg_quote( $match ) . '/', $this->fields-
>get($matches[1][$i].'_'.$this->format, $this->fields->get($matches[1][$i])), $body );
                        $key       = explode('|',$matches[1][$i], 2);
                        $key       = $key[0];
                        $default    = isset($key[1]) ? $key[1] : null;
                        $value      = $this->fields->get($matches[1][$i].'_'.$this->format, $this-
>fields->get($matches[1][$i]));
                        $value      = $value ?: $default;
                        $body = preg_replace( '/' . preg_quote( $match ) . '/', $value, $body );
                }
                return $body;
```

Algoritmo 2. Algoritmo para la gestión del servicio de registro y organización de la base de datos

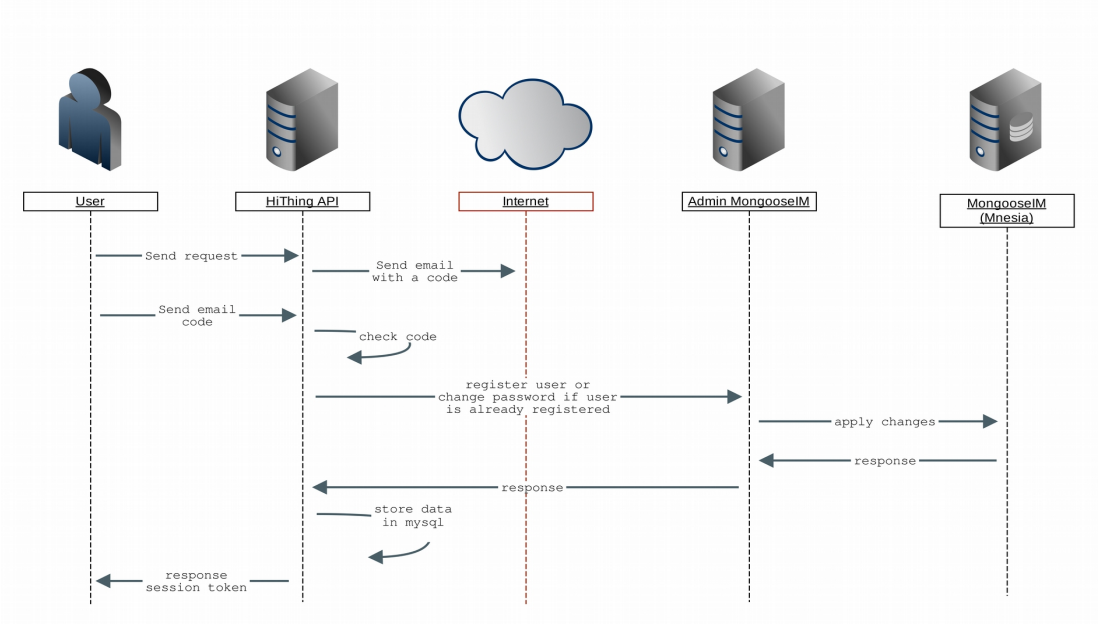


Figura 16. Diagrama de secuencia del funcionamiento del servicio de registro mediante E-mail

ii) Mediante Google+: *token* de Google:

Como se muestra en la figura 17 inicialmente el usuario envía un token al servidor Im4ThingsCloudService para su verificación.

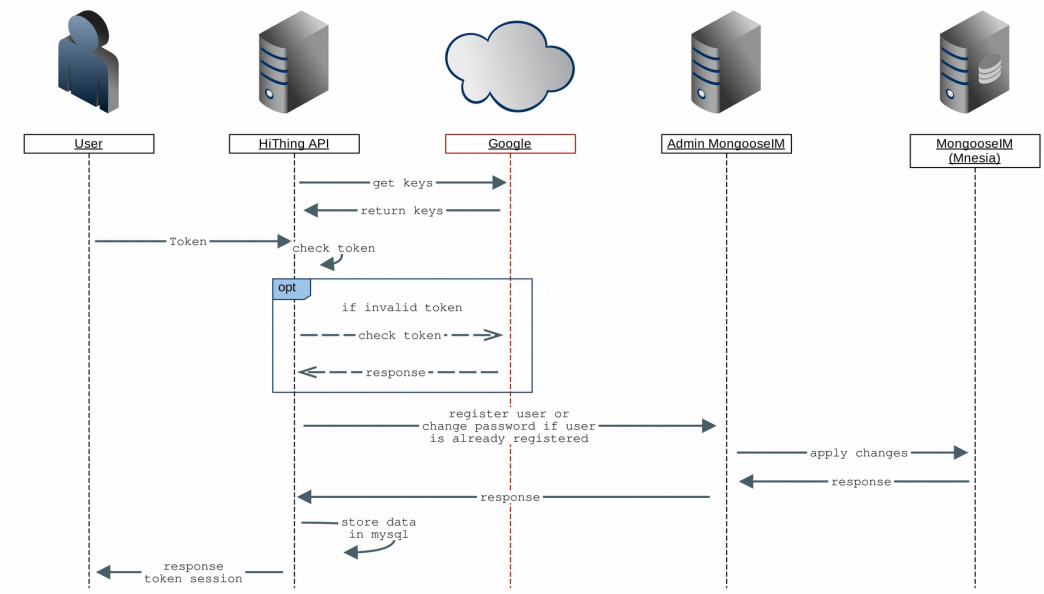


Figura 17. Diagrama de secuencia del servicio de registro mediante token de Google

Con esta firma digital extraída de la red social Google+, una vez que el usuario esté verificado, se creará un identificador ID y un PASSWORD único para ese usuario, que se almacenará en la base de datos MySQL del servicio Im4ThingsCloudService (ubicada físicamente en el servidor). Además, se identificará a este usuario en servicio MongooseIM. El registro de un usuario en MongooseIM se hace en la red local por motivos de seguridad, para que ningún usuario que no esté verificado pueda enviar mensajes.

El servicio responde con un *token* de sesión para que pueda contactar con los servidores en las próximas consultas, identificándose, y proporcionando su *nombre de usuario* y *contraseña* para iniciar las conversaciones dentro de la aplicación de mensajería instantánea. Para garantizar la privacidad, los datos se envían cifrados usando, usando 2048 bits y SHA-2 bajo el protocolo de seguridad SSL *(Security Socket Layer).* Para este servicio fue necesario subcribirse a un proveedor de certificados de seguridad. En concreto contratamos con la empresa certificadora DigiCert[120].

III.3.5 SetvCard

Este servicio es de gran utilidad para el usuario ya que le permite localizar a un contacto por medio de su fotografía. Este servicio forma parte del registro del usuario y para ello la aplicación im4Things app permite adjuntar una fotografía cuando el usuario se da de alta junto al *username* y *password*. La terna de datos lo englobamos bajo la denominación SetvCard, que puede contener una imagen del perfil de usuario además del resto de información del usuario. Esta información se guardará en el servicio para obtener toda la información de manera mucho más rápida y eficiente.

La aplicación im4Things permite que el usuario realice una fotografía en ese momento o bien la adjunte desde el repositorio de fotografías del terminal móvil. En la figura 18 se detalla la secuencia de este servicio. Una vez que se ha cargado la fotografía, se envía la información al servicio encargado del almacenamiento de ficheros del servidor (file_storage), para su almacenamiento en el disco duro del

[120] www.digicert.com (accesible el 11 de octubre de 2015)

servidor. Una vez almacenada, el servicio devuelve al usuario una url que será la encargada de obtener la imagen. En ese momento, el usuario envía los datos de registro formado por la terna username, password y fotografía. Una vez que la SetvCard ha sido actualizada, el servicio envía esta información a MongooseIM, para que los demás usuarios puedan actualizar la información del perfil.

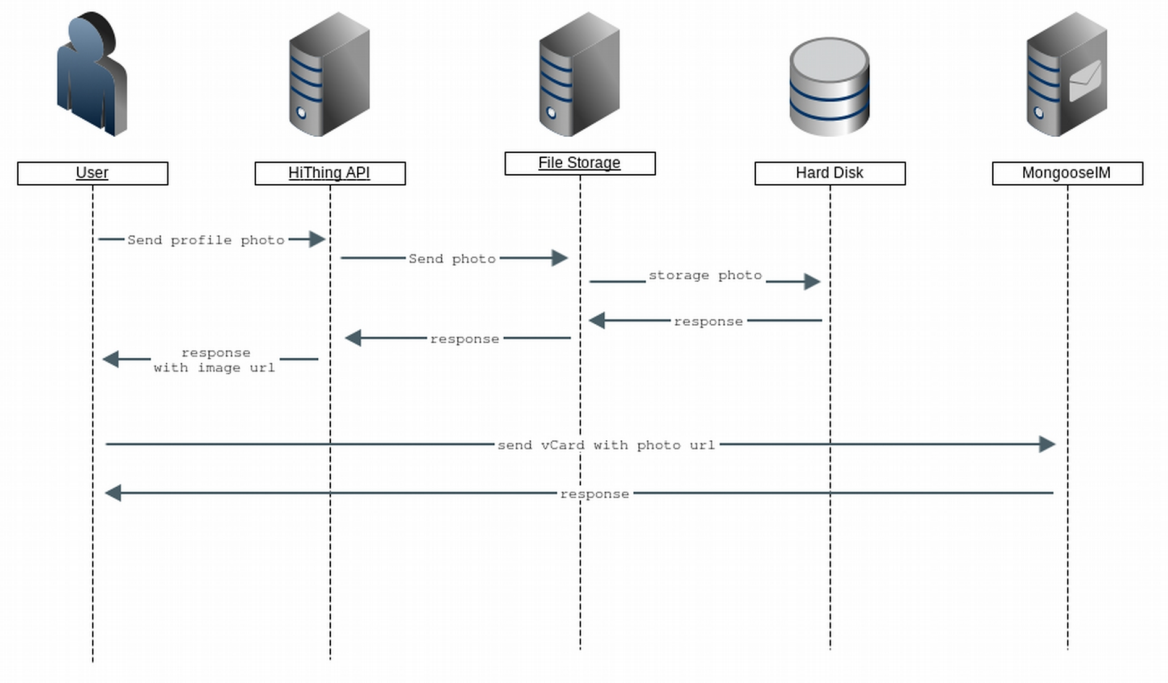


Figura 18. Diagrama de secuencia del servicio del SetvCard

III.3.6 Sincronización de contactos

La sincronización de contactos es necesaria para que el usuario sepa qué contactos suyos están registrados en el servicio y pueda intercambiar mensajería instantánea y archivos multimedia con ellos. En la figura 19 se detalla la secuencia de este servicio. El procedimiento para realizar la sincronización es como sigue. En primer lugar el usuario envía al servicio una lista con toda su agenda de contactos (clasificados por email o teléfono). El servicio obtiene de la base de datos cuáles de esos contactos están registrados en nuestro servicio. Finalmente el servicio le sincroniza estos datos y responde al usuario cuáles de sus contactos están registrados en nuestro servicio y sus username para que puedan comunicarse con ellos.

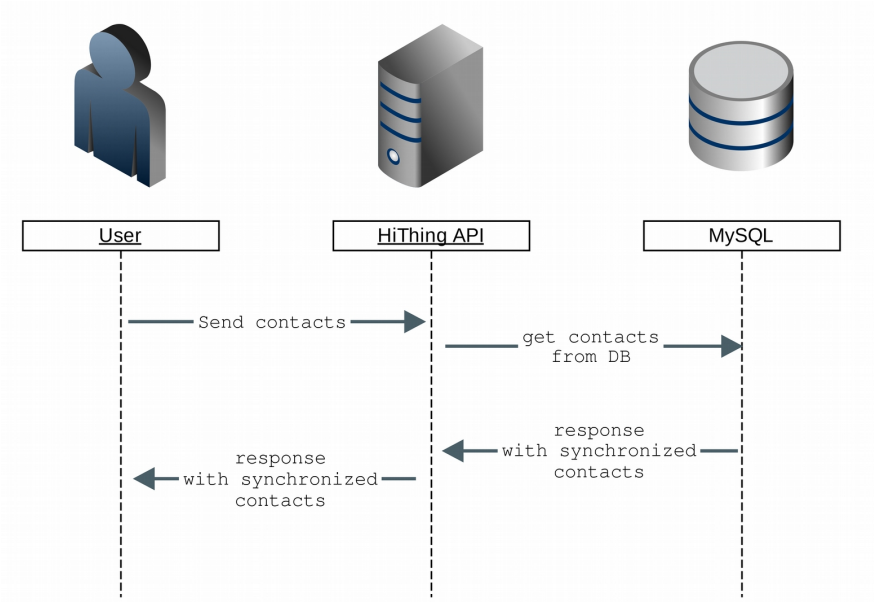


Figura 19. Diagrama de secuencia del servicio para la sincronización de contactos

III.3.7 Store Push Tokens

Para que el servicio reconozca el token individual e impersonal de cada usuario, es necesario reconocer este token en el servicio. Para poder enviar notificaciones vía *push*, cada servicio (Google, Apple, Windows) provee a cada dispositivo un *token*. De esta manera se puede identificar a dónde se va a enviar la notificación. El procedimiento comienza cuando el usuario envía o renueva el token se sesión. Dicho token será almacenado en la base de datos *MySQL*. Cuando el token ha sido almacenado, el servicio responde al usuario con un mensaje de operación satisfactoria. Ver figura 20:

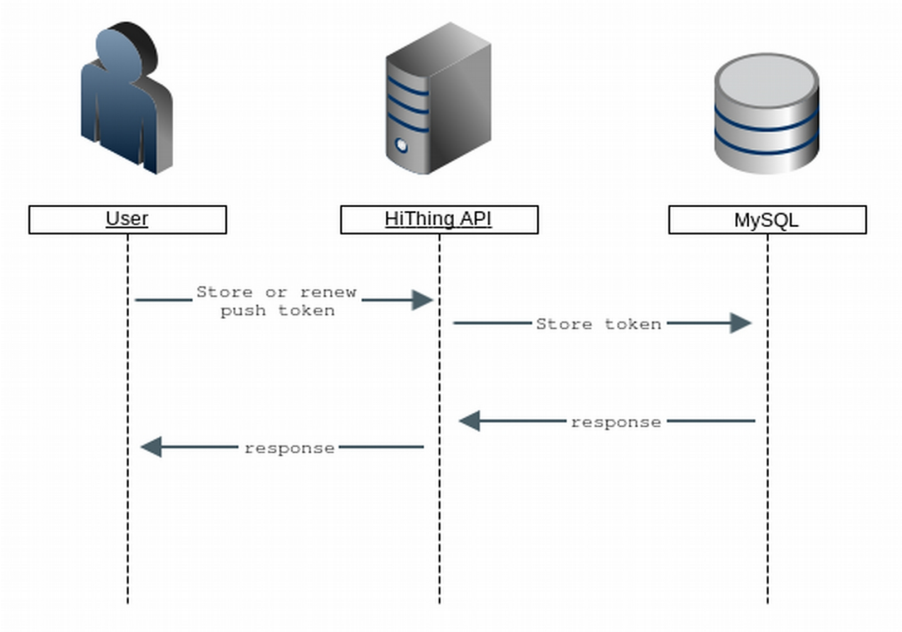


Figura 20. Diagrama de secuencia del servicio Store Push Tokens

III.3.8 Notificaciones Send Push

Este servicio se ha creado para las notificaciones push que son mensajes de texto que se le envía al usuario y que las puede ver aun no teniendo la aplicación im4Things app abierta. Estas notificaciones son posible siempre que el servicio haya registrado previamente el token del usuario (ver apartado III.3.4). Una vez que el servicio sabe cuál es el token de cada dispositivo de un usuario, puede enviarle notificaciones push. En la figura 21 se presenta la secuencia seguida para el envío de la notificación push a los usuarios (clientes). El procedimiento es como sigue: cuando a un usuario se le ha enviado un mensaje y no está en línea, es decir, no tiene activada la aplicación im4Things app, MongooseIM le envía un mensaje la información de ese mensaje a través del servicio *Pushservice*[121]. Este servicio se utiliza básicamente para enviar notificaciones instantáneas a clientes. Para ello, el servicio pushservice obtiene de la base de datos el token del receptor (cliente) a quien le tiene que enviar la notificación push y le pasa el mensaje al proveedor (Google, Apple, Windows...) con el correspondiente token (si el usuario tiene varios dispositivos hay tantos tokens como dispositivos).

121 www.parse.com/docs/android/api/com/parse/PushService.html (accesible el 12 de octubre de 2015)

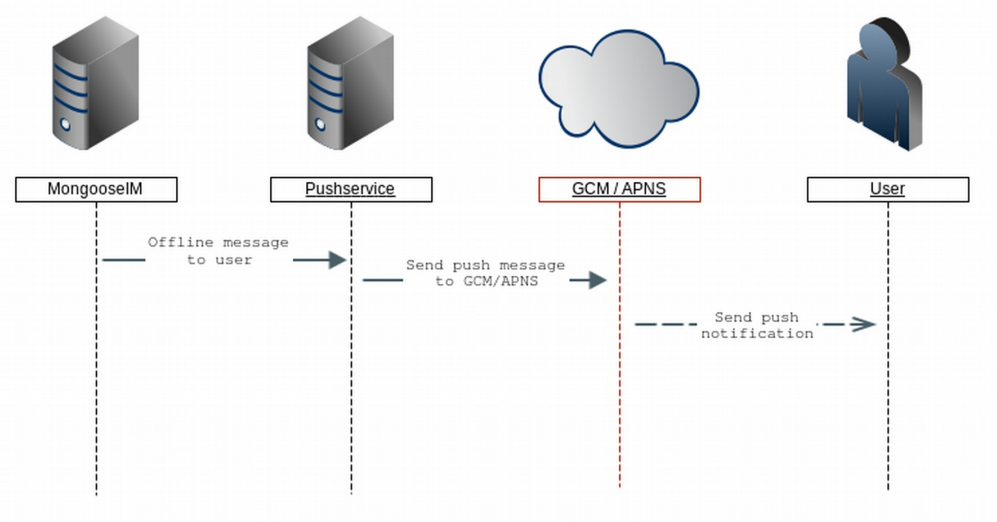


Figura 21. Diagrama de secuencia del servicio Notificaciones Send Push

III.3.9 Registro de la base del conocimiento

Para este registro se ha creado en "userservice" una petición Rest que será llamada desde el programa donde se define la base del conocimiento (BOT) para registrarse. En esta petición, el BOT enviará su MAC y el usuario que ha registrado la máquina. El "userservice" crea el JID de la máquina a partir de su MAC con el formato: MAC + Hash (MAC + contraseña propia del servicio) para evitar registros de usuarios con ese JID, a su vez almacenará los datos de la máquina en la base de datos MySQL, y almacenará el administrador de dicha máquina.

Además, se habilita un módulo específico a MongooseIM para que solo el administrador pueda comunicarse con esa máquina.

III.3.10 Lista de usuarios por máquina

Para ello se ha habilitado un módulo en MongooseIM para que sólo puedan enviar mensajes una máquina desde determinados usuarios. Según las pruebas realizadas llegamos a la conclusión que podemos hacerlo de varias formas:

1. La aplicación del administrador puede enviar una IQ al BOT indicando los usuarios que pueden enviarle mensajes. Posteriormente el BOT realiza una

petición a MongooseIM indicándole la lista de privacidad de los usuarios que van a enviar los mensajes.

2. Creando un servicio que se haga pasar por el BOT. De esta forma será este servicio a petición del administrador, quien envié la petición a MongooseIM con la lista de privacidad.

III.3.11 Notificaciones de funcionamiento del BOT

Cuando un BOT esté realizando alguna actividad, hay que notificar a los usuarios que estén dentro de su lista de privacidad. Esto se puede realizar de distintas formas:

1. Crear un módulo *publish-subscribe* en MongooseIM. La conclusión que llegamos por medio de este paso es que podría afectar al rendimiento del servidor, ya que es un módulo que requiere muchos recursos.
2. Que sea el BOT quien envíe mensajes normales a los usuarios de su lista de privacidad.

Cuando un usuario desea enviar un mensaje a un electrodoméstico, primero deberá "llamar" a ese electrodoméstico ya sea directamente en texto y por medio de un emoticono. A partir de aquí se abre un canal privado de comunicación por el cual el usuario transmitirá una orden a ejecutar por el electrodoméstico. Esa orden no tiene que seguir un determinado listado de comando. Basta con que la orden esté relacionada con la función del electrodoméstico para que éste obedezca. El usuario escribirá un texto en lenguaje natura como por ejemplo "cafetera haz café". Ver figura 22:

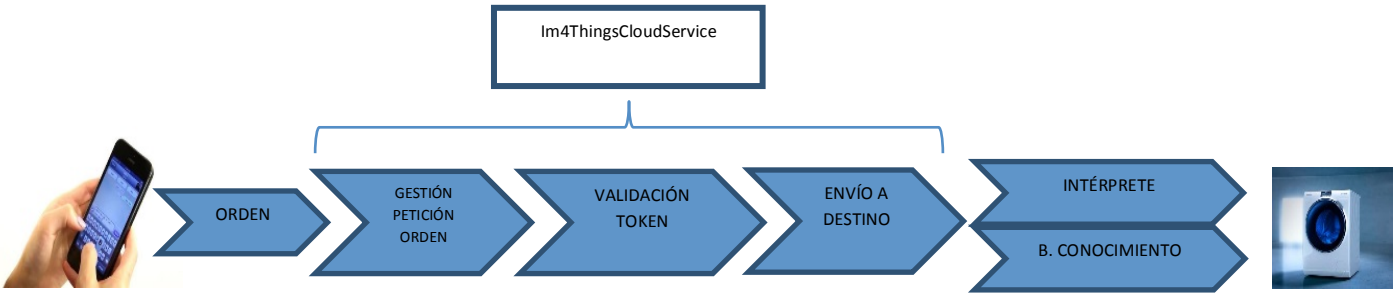


Figura 22. Secuencia por bloques funcionales de la comunicación user-dispositivo

III.3.12 Gestión de colas: prioridad de mensajería

Uno de los puntos críticos de este proyecto es garantizar la calidad y continuidad de servicio. Teniendo en cuenta que, el funcionamiento de los dispositivos depende en cierta medida de la comunicación con el servidor, este debe estar bien dimensionado a efectos de escalabilidad y además debe ser capaz de gestionar el tráfico de la mensajería instantánea de forma adecuada, dando prioridad al tráfico de mensajes entre dispositivos. Hasta el momento es alta la incertidumbre que tenemos en relación a:

1. Factor multiplicador del tamaño del servidor
2. Volumen del tráfico de datos en tiempo real.

En nuestro caso se ha desarrollado el modelo para estudiar la relación de la variación de la intensidad de tráfico de mensajes cursado con la intensidad de tráfico ofrecido por el servidor partiendo del tamaño de la base de datos. Con ello intentamos determinar el tamaño óptimo del servidor (grado de servicio) en cada momento para garantizar la continuidad de servicio.

Para nuestro cálculo hemos supuesto que el proceso de llamadas al sistema sigue una distribución de ***Poisson*** (λ). La distribución de *Poisson* es una distribución de probabilidad discreta que determina a partir de una frecuencia de ocurrencia media de un evento, la probabilidad de que ocurra un determinado número de eventos durante cierto período de tiempo. La herramienta matemática que define la función de probabilidad es:

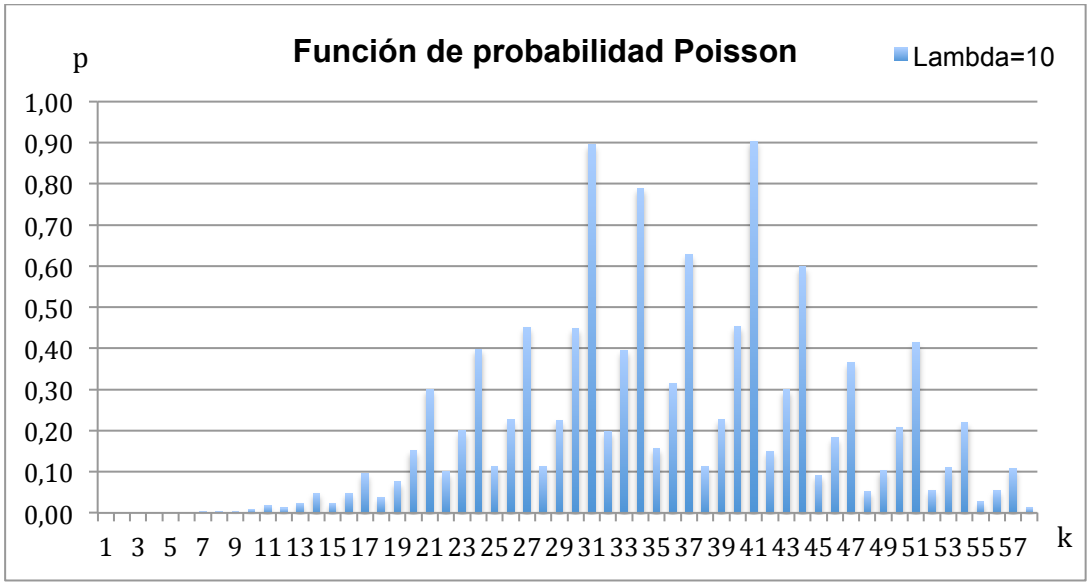
$$f(k,\lambda) = \frac{e^{-\lambda}\lambda^{k}}{k!}$$

Los parámetros de la ecuación son:

k es el número de veces que ocurre un evento en un intervalo temporal.

λ es un parámetro positivo que representa el número de veces que se espera que ocurra el fenómeno durante un intervalo temporal dado.

A título de ejemplo supongamos que tenemos como medida de referencia una tasa de fallo del 0,1% de causado por la capacidad del sistema. Para determinar la probabilidad de que tengamos un número máximo de 5 fallos por 10.000 usuarios usamos la distribución de Poisson, teniendo k=5 y λ=0,001⍰10.000=10. En la gráfica 1 se representa la función de probabilidad de este ejemplo.



Gráfica 1. Función de probabilidad para k=5 y λ=10 del ejemplo. P=0,037833

Para resolver los puntos anteriores, se ha desarrollado un monitor de actividad denominado ModelActivity basado en el modelo Engset. Engset es una unidad adimensional que se utiliza en los servicios telefónicos y sirve como medida estadística del volumen de tráfico en un momento dado. En concreto el modelo Engset se basa en el modelo generalista Erlang pero considerando el primero un número finito de usuarios mientras que el modelo Erlang considera un número infinito.

El aparato matemático empleado es el siguiente. La figura 23 representa el diagrama de la transición de estados para el caso de tener un número de recursos de gestión de mensajes (servidores=m) menor al número de peticiones (N) (N≥m). El modelo escogido expresa en mayor grado el caso contrario, ya que será determinante para seguir escalando la capacidad del servidor cuando este haya

alcanzado la capacidad crítica. Los índices i=0,1,...m corresponden a cada servidor que ofrece los servicios de nuestro proyecto.

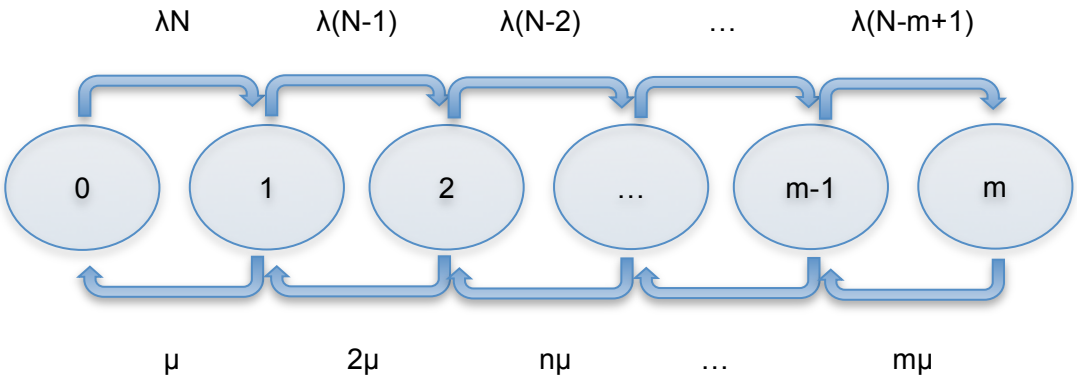


Figura 23. Diagrama de diagrama de la transición de estados Engset

Las restricciones que impone este método son:

1. El tiempo de servicio está distribuido exponencialmente con tasa μ (distribución de Poisson).
2. El número de peticiones es limitado por cada servidor de servicios (m).

Un servidor de servicios (en adelante servidor m), corresponde a parte de la solución del sistema capaz de gestionar una petición de servicio. Una petición de servicio lo puede constituir por ejemplo la gestión de mensajería instantánea de un usuario, la gestión de datos multimedia, etc.

Cuando el sistema está gestionando una petición de mensajería, está un tiempo ocupando el recurso de gestión, tiempo que no podrá atender a otra petición y cuando finaliza el proceso quedará libre para atender otra petición. Gráficamente se representa en la figura 24:

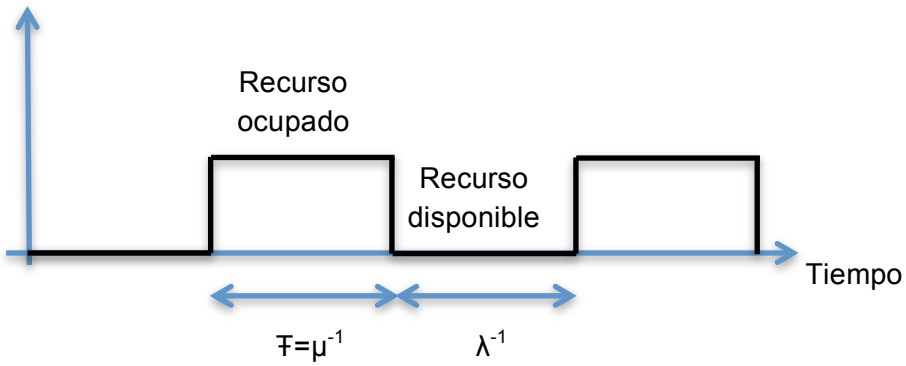


Figura 24. Diagrama temporal que representa la ocupación del servidor atendiendo a una petición

El parámetro que queremos determinar la probabilidad de bloqueo del sistema (p(i)) para atender peticiones de los usuarios y dispositivos de forma que nunca se llegue al 1. La determinación de esta probabilidad viene determinada por las ecuaciones de balance de flujo según el siguiente conjunto de relaciones:

$$N \cdot \lambda \cdot p(0) = p(1) \cdot \mu$$
$$(N-1) \cdot \lambda \cdot p(1) = p(2) \cdot 2\mu$$
$$(N-2) \cdot \lambda \cdot p(2) = p(3) \cdot 3\mu$$
$$\dots$$
$$(N-i) \cdot \lambda \cdot p(i) = p(i+1) \cdot (i+1)\mu$$

El inicio del cálculo lo hacemos a partir de la probabilidad de bloqueo para el primer servicio **p(0).** A partir de este valor vamos a determinar el resto de probabilidades de bloqueo, es decir para los distintos estados i: i->p(i):

$$p(m) = p_0 \cdot \prod_{i=0}^{m-1} \cdot \frac{\lambda_i}{\mu_{i+1}}$$

$$p(1) = p(0) \cdot m \cdot \frac{\lambda}{\mu} = p(0) \cdot \binom{m}{1} \cdot (\frac{\lambda}{\mu})^1$$

$$p(2) = p(0) \cdot \binom{m}{2} \cdot (\frac{\lambda}{\mu})^2$$

…

$$p(m) = p(0) \cdot \binom{m}{m} \cdot (\frac{\lambda}{\mu})^m$$

Como el total de todas las probabilidades debe ser igual al a unidad, obtenemos:

$$1 = p(0) \cdot \left\{1 + \binom{m}{1} \cdot \left(\frac{\lambda}{\mu}\right)^0 + \binom{m}{2} \cdot \left(\frac{\lambda}{\mu}\right)^2 + \cdots + \binom{m}{m} \cdot \left(\frac{\lambda}{\mu}\right)^m\right\} = p(0) \cdot (1 + \frac{\lambda}{\mu})^m$$

(ecuación 1)

En función de p(0) y m podemos determinar la relación entre λ y μ, y como consecuencia podemos determinar la velocidad de proceso en el número de peticiones que debe tener el servidor para atender todas las peticiones de los usuarios. Esta velocidad de procesar información sin quedar bloqueado el servidor es lo que se denomina **tráfico ofrecido** (TO) y que está representado por:

$$TO = \lambda_m \cdot \frac{1}{\mu} = \frac{N \cdot \beta}{1 + \beta \cdot (1 - P(i))}$$ (en unidades Erlang[122]) (Ecuación 2)

siendo:

- Intensidad de tráfico: $\lambda_m = N \cdot \omega$ (ecuación 3)
- ω =número de peticiones/tiempo
- $\beta = \frac{\lambda}{\mu}$

Una unidad erlang tiene un significado de unidad de intensidad de tráfico. Un erlang es la intensidad de tráfico de un conjunto de servicios cuando solo uno de ellos está ocupado de manera continua. Cuando el tráfico es de un (1) erlang significa que el elemento de red está totalmente ocupado durante el tiempo de medición.

[122] www.kt.agh.edu.pl/~brus/kolejki/ErlangBTables.pdf (accesible el 12 de octubre de 2015)

El problema que pretendemos resolver con esta herramienta matemática es la tráfico ofrecido TO en erlang que equivale al número de peticiones o eventos realizados con éxito por el servicio Im4ThingsCloudService en función de una probabilidad de fallo (probabilidad de bloqueo) establecida y un número de servidores disponibles. Para las pruebas de concepto realizadas hemos supuesto los siguientes valores de partida para un determinado momento del día donde los requisitos de servicio son:

Queremos determinar el número de servidores disponibles m capaces de gestionar la información que a continuación se detalla durante una prueba de estrés fijada en 1 hora (hora cargada):

P=0,1 (probabilidad de que una petición sea gestionada por encima de un tiempo umbral)

N=10.000 peticiones

w=1 gestión_petición/segundo.

Demanda de servicio siguiendo un distribución exponencial (Poisson)

Los resultados obtenidos son los siguientes:

De la ecuación 3 obtenemos el valor de la intensidad de tráfico:

$$\lambda_m = N \cdot \omega = 10.000 \cdot 1 = 10.000$$

en base a que los cálculos los estamos realizando sobre una hora cargada el tráfico ofrecido es:

$$TO = \lambda_m \cdot \frac{1}{\mu} = \frac{10.000}{3.600} = 2{,}7777\ Erlang$$

de la ecuación 1 obtenemos el número de servidores necesarios para atender a la petición de servicio:

$$1 = p(0) \cdot \left(1 + \frac{\lambda}{\mu}\right)^m = 0{,}1 \cdot (1 + 2{,}77777)^m$$

Despejando m obtenemos:

$$m = \frac{\log(10)}{\log(3{,}277777)} = 1{,}7323$$

En el caso de que m=2 despejando P(1) de la ecuación 1 obtenemos P=0,073, del que el primer servidor estará al 100% operativo y el segundo al 73,23%.

La interpretación de estos datos arroja que para garantizar la gestión de la demanda solicitada serían necesarios dos servidores para que la tasa de fallo no supere el 0,1%. En términos de peticiones la solución anterior no discrimina si los mensajes son de un usuario o de un dispositivo. Una solución que proponemos en este proyecto es que la base de datos MySQL se organice en función de la prioridad del mensaje.

Aunque este punto no es fundamental para nuestra proyecto, hemos creído conveniente realizar este cálculo estadístico porque nos da una idea de la relación que existe entre la calidad del servicio frente a los recursos disponibles. Para trabajos futuros, este punto se podrá desarrollar en profundidad cuando tengamos una masa crítica de usuarios determinante.

Los datos en la base de datos MySQL están organizada según una estructura de **cola de prioridades**. Una cola de prioridades es una estructura de datos en la que los elementos se atienden en el orden indicado por una prioridad asociada a cada uno. Si varios elementos tienen la misma prioridad, se atenderán de modo convencional según la posición que ocupen. El tipo de cola atiende al algoritmo de ordenación descendente de forma que el dato a extraer es el de mayor prioridad.

El objetivo de esta técnica es determinar tanto la estructura de la base de datos empleada como su dimensionamiento desde el punto de vista de los datos generados por cada mensaje transferido frete a la prioridad que cada mensaje

tiene frente al resto. Esto se hace desde la perspectiva de distinguir los tipos de datos encolados. El La clasificación realizada está catalogada según cuatro escalones de prioridad yendo desde la menos prioritaria (0) a la más prioritaria (3).

- Mensaje prioridad 0 como aquel que simplemente es un mensaje de texto no proveniente del Bot.
- Mensaje prioridad 1 como aquel que es un mensaje reconocido como orden hacia el Bot.
- Mensaje prioridad 2 como aquel que es un mensaje de alarma proveniente del Bot hacia otro Bot.
- Mensaje prioridad 3 como aquel que es un mensaje de alarma proveniente del Bot hacia el usuario.

La estructura básica del algoritmo para la operación básica de la creación de la cola vacía es:

```
Cola(){
        elementos_cola = k; //Distribución Erlang
while (elementos_cola != 0) pop();
    }
    void push(const T& elem){
       Prioridad* aux = new Prioridad;
       aux->elemento = elem;
       if (elementos_cola == 0) primero = aux;
       else ultimo->siguiente = aux;
       ultimo = aux;
       ++elementos;
    }
    void pop(){
       Nodo* aux = primero;
       primero = primero->siguiente;
       delete aux;
       --elementos;
    }
    T consultar() const{
       return primero->elemento;
    }
    bool vacia() const{
       return elementos_cola == 0;
    }
    unsigned int size() const{
       return elementos_cola; }
```

Algoritmo 3. Algoritmo para el encolamiento de los mensajes según prioridad de servicio

III.4 DISPOSITIVO

Cada dispositivo tiene un hardware específico formado por sensores y un sistema de control y comunicación que interactúa con el software para ejecutar los comandos y enviar el estado del dispositivo de una manera eficiente. Este hardware está basado en Raspberry Pi que se ha utilizado en múltiples sistemas de control y reconocimientos de voz como el presentado en *(Noguera, 2015)*. Dentro del software del dispositivo también se implementa una versión de la Im4Things app que es gestionada por el módulo de diálogo que se en carga de mantener la conversación entre el dispositivo y otros usuarios.

El módulo de diálogo inteligente permite ejecutar comandos y consultar el estado actual del mismo. Este módulo también permite detectar alertas del dispositivo, transformarlas a lenguaje natural y enviarlas al servicio Im4ThingsCloudService para que sean comunicadas a los usuarios que tengan suscrito ese servicio. Un ejemplo de esta alerta puede ser que la ejecución de un comando ha finalizado.

En la figura 25 se ilustran los distintos bloques constructivos que conforman la base del conocimiento que junto a la parte Hardware denominamos como Bot.

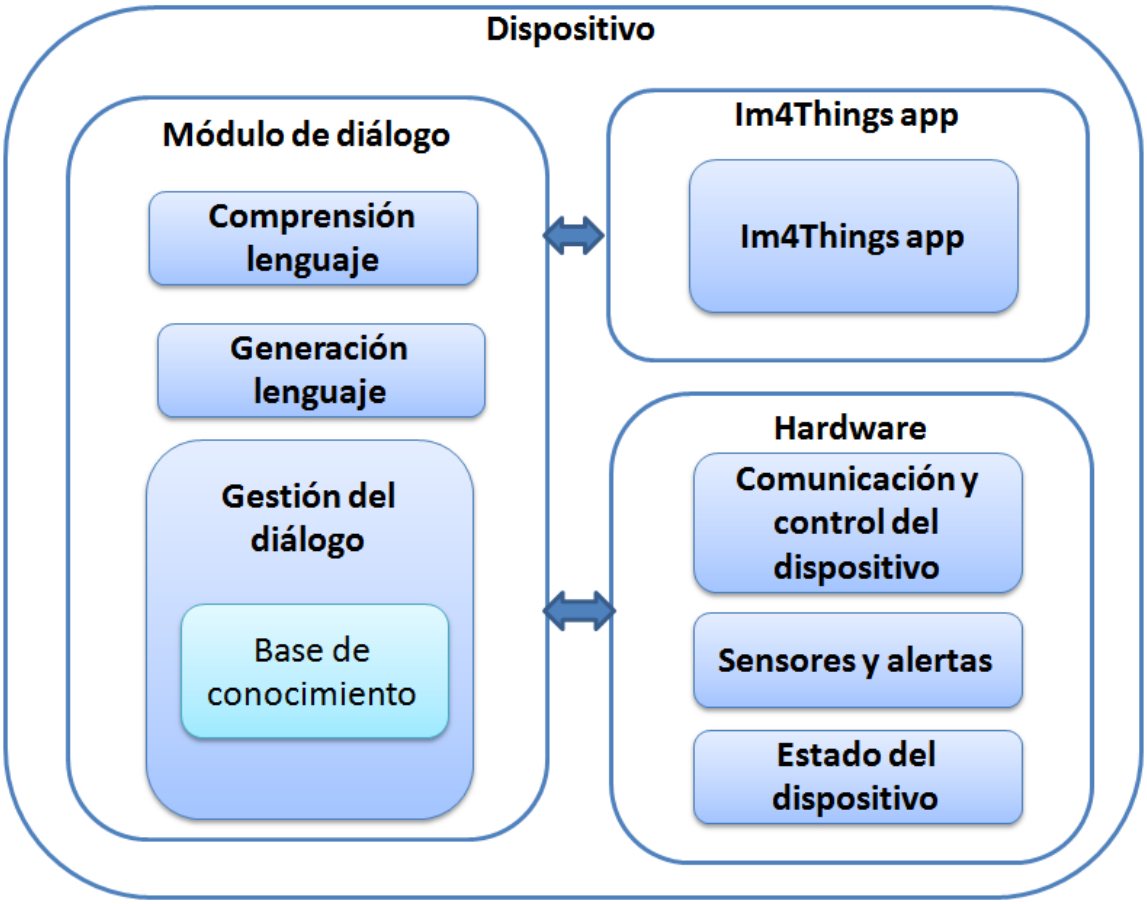


Figura 25. Arquitectura funcional del dispositivo

III.4.1 Hardware del dispositivo

La elección del hardware para esta tesis doctoral fue determinante. La filosofía seguida fue la misma que rige a la Internet de las Cosas, es decir, que cada dispositivo a controlar tenga su propia dirección IP a la que conectarse para comunicarse con el dispositivo desde los servicios de la nube y gestionar distintos eventos. Los eventos pueden ser de varios tipos en función si el destinatario de la información es un usuario o un electrodoméstico. Un ejemplo de evento puede ser cuando un usuario envía un mensaje al electrodoméstico, por ejemplo con la finalidad de encenderlo o apagarlo. Otro tipo de evento puede ser cuando un electrodoméstico envía un mensaje a otro electrodoméstico, por ejemplo cuando la cafetera se queda sin agua y envía un mensaje a una válvula con la orden abrir/cerrar.

Las funcionalidades deseables que debía cumplir el hardware al inicio del proyecto eran las siguientes:

- Servir de interface entre el electrodoméstico y la capa superior de aplicación. Entendemos como capa superior de aplicación todo lo relacionado con la conexión de datos entre el electrodoméstico e Internet.
- Gestionar las peticiones de mensajería instantánea desde el servidor y desde las señales de campo, como por ejemplo la botonera del electrodoméstico.
- Gestión de la base de conocimiento basada en ontologías.
- Gestión de la comunicación inalámbrica.
- Integración dentro de la electrónica propia del electrodoméstico.

Como punto de partida se seleccionaron dos posibles tipos de hardware, de arquitecturas distintas, sobre las que hacer las pruebas de validación. La primera solución empleada está basada en un microcontrolador (Arduino) y la segunda solución seleccionada está basada en la arquitectura de microprocesador (Raspberry Pi). Ambas soluciones comerciales, deben de tener la capacidad de gestión los distintos bloques de entrada y salida digitales así como los puertos de comunicaciones serie e inalámbrica: WiFi y Bluetooth.

A continuación se explica brevemente cada propuesta con las que se realizaron los prototipos de la presente tesis doctoral.

III.4.2 Hardware Arduino para IoT

Arduino[123] es una plataforma de desarrollo electrónico "open-source" basada en el concepto "fácil de usar" tanto en lo referente al software IDE como al propio hardware. Este tipo de plataforma está especialmente orientada a profesionales y aficionados a la electrónica con el denominador común de que posibilita el intercambio de proyectos a través de Internet. Por otro lado, esta plataforma hardware está especialmente diseñada para ampliar de forma modular la placa

[123] www.arduino.cc (accesible el 12 de octubre de 2015)

original o placa base por medio de los denominados "*shields*[124]". Un shields es una placa electrónica que dispone de los sensores y elementos de comunicación inalámbrica que se pueden acoplar a la placa base del ardiuno con el objetivo de ampliar sus capacidades. Por su construcción los shields pueden ser apilados unos encima de los otros obtenido la alimentación eléctrica directamente de la placa base. En nuestro caso particular empleamos la siguiente configuración: Arduino DUE+ Arduino WiFi Shield (ver figuras 26 y 27)

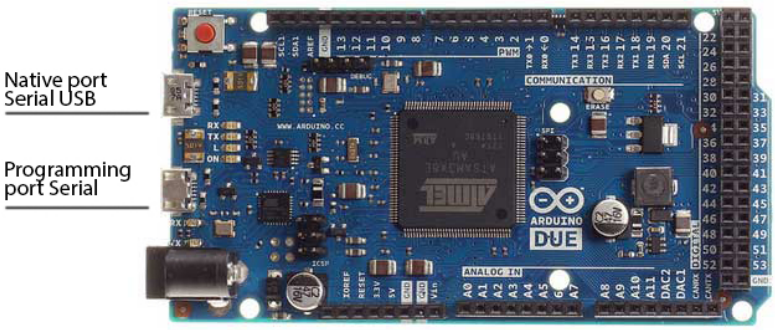


Figura 26. Imagen de la placa Ardino DUE. En la imagen aparecen los puertos de comunicaciones así como los conectores PCB para la inserción de Shields. (fuente http://arduino.cc)

Figura 27. Imagen de la placa Arduino WiFi Shield incluyendo la antena de comunicaciones para la banda de frecuencia de 2 GHz (fuente www.aliexpress.com)

Las principales características de estos dispositivos son las siguientes:

Arduino DUE

- Microcontrolador: AT91SAM3X8E

[124] www.arduino.cc/en/Main/ArduinoWiFiShield101 (accesible el 12 de octubre de 2015)

- Tensión de operación: 3.3V
- Rango de tensión para las líneas de entradas analógicas: 7-12V
- Rango de tensión para las líneas de entradas/salidas digitales: 6-16V
- Consumo de corriente para las líneas de E/S: 130 mA
- Memoria Flash: 512 KB
- SRAM: 96 KB (two banks: 64KB and 32KB).
- Velocidad de reloj: 84 MHz

Arduino WiFi Shield

- Tensión de alimentación 5Vcc (esta tensión de alimentación la suple directamente el Arduino DUE).
- Arduino Due compatible
- Vía de conexión (canal y frecuencia): por medio del estándar 802.11b/g networks
- Tipo de encriptación de los datos: WEP y WPA2 Personal
- Conexión con el Arduino DUE: por medio del puerto SPI
- Micro SD slot
- Puerto de programación serie: ICSP
- Puerto serie FTDI para debuging
- Mini-USB para actualización del firmware

III.4.3 Hardware Raspberry Pi para IoT

Raspberry Pi[125] es un ordenador de placa reducida o (placa única) (SBC) de bajo coste, desarrollado con el objetivo de estimular la enseñanza de la informática y la computación.

A diferencia que la anterior solución, este dispositivo consta de una única placa electrónica con los correspondientes puertos de entrada y salida (I/O) y comunicaciones según se detalla en la figura 28:

[125] www.raspberrypi.org (accesible el 12 de octubre de 2015)

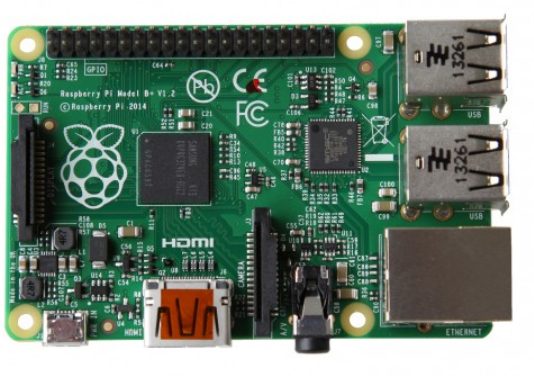


Figura 28. Imagen del ordenador de placa reducida Raspberry Pi (RB Pi). La interface para comunicar RB Pi con el electrodoméstico es el bloque de pines configurable destinados a entradas y salidas digitales. (Fuente www.aliexpress.com)

Las principales características de este dispositivo son:

- Arquitectura ARM.
- System-on-a-chip Broadcom BCM2835.
- Procesador central (CPU) ARM1176JZF-S a 700 MHz. Esto permite implementar firmware para la gestión de diferentes modos de funcionamiento permitiendo realizar la función *overclock*[126] de hasta 1 GHz de velocidad de proceso.
- Procesador gráfico (GPU) VideoCore IV.
- 512 MB de memoria RAM.
- Tarjeta SD para el almacenamiento permanente.
- Sistema operativo RISC OS 5, Arch Linux ARM (derivado de Arch Linux) y Pidora (derivado de Fedora).
- Lenguajes de programación soportados: Python.5, Tiny BASIC,12 C, Perl5 y Ruby.13

La experiencia adquirida del empleo de estos dispositivos para la fase de prototipado ha sido muy enriquecedora ya que por un coste reducido se obtiene grandes resultados reduciendo así el tiempo de desarrollo del hardware.

Tanto Arduino como Raspberry Pi ofrecen distintas posibilidades, enfoques y soluciones para su empleo en Internet of Things. Lo que buscamos en estos dispositivos para su utilización en nuestro proyecto fue principalmente dos cosas:

[126] Aumento del rendimiento del dispositivo mediante el aumento de la frecuencia de trabajo.

conectividad inalámbrica y posibilidad de embeber la base de conocimiento y algoritmos de inteligencia artificial. Por otro lado también nos interesa tener en cuenta otros importantes aspectos como son el económico y la comunidad de desarrolladores que sigue a cada producto. Esto último es importante en cierto modo, no desde el prisma científico pero sí técnico porque la comunidad de desarrolladores ofrece el bien incuestionable de avanzar en la técnica y alcanzar mejores resultados en el futuro. Y eso será posible siempre que estos desarrollos estén al alcance económico de una gran mayoría.

Ambas soluciones ofrecían, durante la elaboración de este proyecto idénticas, posibilidades de comunicación. Estamos hablando claro está de comunicación Wifi y Bluetooth con sus correspondientes shields. Por otro lado, el repertorio de librerías y comandos de programación son tan amplias que a día de hoy se podrían embeber sin problema a muchos electrodomésticos sin inteligencia previa, es decir electrodomésticos exentos de "electrónica".

Pero una imperiosa necesidad que demanda nuestro proyecto y que obligatoriamente debe cumplir la electrónica es tener la capacidad de llevar instalado un sistema operativo encargado de llevar a cabo la incuestionable labor de dotar de inteligencia al electrodoméstico, ya no desde el punto de vista funcional que de ello se encarga el fabricante sino de dotar de personalidad al electrodoméstico. No en vano el objetivo primero de este proyecto es humanizar las máquinas.

En el siguiente punto se describe los pasos que se han seguido para validar el hardware que finalmente utilizamos.

III.4.4 Validación del hardware

Para poder evaluar las dos soluciones hardware propuestas anteriormente se formuló un programa de puntos de validación (PPV) diseñado para conocer si la combinación Arduino DUE+ Arduino WiFi Shield y el hardware Raspberry Pi serían capaces de responder con éxito a las siguientes cuestiones:

- PPV1. Capacidad de integración con el software de gestión de diálogo inteligente.
- PPV2. Gestión de puertos.
- PPV3. Capacidad de gestión de las entradas y salidas para atender las peticiones por parte del usuario.
- PPV4. Capacidad de integración de la encriptación de seguridad en las comunicaciones.
- PPV5. Consumo y estabilidad eléctrica.
- PPV6. Conectividad WiFi y Bluetooth.
- PPV7. Establecimiento y tráfico de información entre el servidor y los electrodomésticos.
- PPV8. Retardos.
- PPV9. Robustez frente a fallos y pérdidas de comunicación.
- PPV10. Compatibilidad electromagnética.

Tanto el montaje como los ensayos realizados se efectuaron en los laboratorios del departamento empleando para ello los siguientes equipos como: un osciloscopio, un ordenador portátil, UIs de programación, espectro analizador, voltímetro, estación de soldadura, placa *protoboard*[127] y otro pequeño material y cableado:

[127] www.circuitoselectronicos.org/2007/10/el-protoboard-tableta-de-experimentacin.html (accesible el 12 de octubre de 2015)

PPVi	Arduino	Raspberry Pi
PPV 1	✓	✓✓✓✓✓
PPV 2	✓✓✓	✓✓✓✓✓
PPV 3	✓✓✓✓✓	✓✓✓
PPV 4	✓✓✓	✓✓✓✓✓
PPV 5	✓✓✓✓	✓✓✓✓✓
PPV 6	✓✓✓✓✓	✓✓✓✓✓
PPV 7	✓✓✓	✓✓✓✓✓
PPV 8	✓✓✓✓✓	✓✓✓✓
PPV 9	✓✓✓✓	✓✓✓✓
PPV 10	✓✓✓✓	✓✓✓✓

Tabla 9. Resultados obtenidos de la validación del hardware

Como se pueden observar en la tabla 9 los resultados que obtuvimos fueron determinantes. Raspberry Pi (en adelante RP Pi) ofrece una ventaja fundamental frente a Arduino y es que se le puede instalar un sistema operativo Linux. En concreto nosotros hemos empleado la versión Raspbian[128] (*Foundation's official supported operating system*). Por otro lado, la capacidad de gestionar el tráfico de información server-user aventaja esta solución frente al Arduino. Todo ello tiene su base en la gestión de la información. Por un lado, la capacidad de un microcontrolador se acerca más a soluciones industriales donde la secuencia de pasos del proceso es fundamental. En cambio el microprocesador tiene una mayor capacidad de gestionar la información de los datos y más concretamente para la gestión de programas que funcionan y son compatibles con el sistema operativo.

Respecto al resto de característica, ambas soluciones son válidas porque tiene que ver con el hardware. En este caso el hardware de ambas soluciones tiene un alto potencial para ser integrados en soluciones de IoT.

Por lo tanto y a tenor de los resultados obtenidos se llega a la conclusión de que la solución RB Pi es más idónea que la solución Arduino entre otros motivos

[128] www.raspberrypi.org/downloads/raspbian/ (accesible el 12 de octubre de 2015)

porque la PPV1, es decir que el microcontrolador no tiene suficiente capacidad para la gestión simultánea del firmware de inteligencia artificial implementado en el módulo de diálogo inteligente y las señales del interface.

III.5 Im4Things App

Como interface hombre-máquina, se ha diseñado una aplicación App denominada Im4Things app para terminales Android. La interface ofrece al usuario la posibilidad de enviar y recibir mensajes multimedia, archivos de datos y multimedia. Por medio de esta aplicación, el usuario puede mantener y gestionar la conversación del dispositivo con otros usuarios y otros dispositivos similar a otras redes sociales en formato chat.

III.5.1 Módulo de diálogo

Tradicionalmente, los sistemas de diálogo escrito están compuestas por tres tareas distintas de procesamiento del lenguaje natural: comprensión del lenguaje natural, la gestión del diálogo y la generación de lenguaje. En el sistema presentado en esta tesis doctoral además es necesario incluir un nuevo módulo que permita la comunicación con los dispositivos para consultar su estado, ejecutar comandos o gestionar la comunicación entre los usuarios y los dispositivos.

La figura 29 muestra la arquitectura del sistema de diálogo. Como se puede observar, el sistema de diálogo se basa en una ontología del dispositivo que representa y formaliza la información y conocimiento de tal dispositivo como las acciones que pueden ejecutarse en el mismo, los sensores que contiene, los servicies y alertas que pueden ejecutarse en el dispositivo, sus estados posibles y otros dispositivos que pueden integrarse en el dispositivo.

Por otro lado, existen 4 módulos principales que forman el sistema: el módulo de comprensión, el módulo de gestión del diálogo, el controlador del dispositivo y el módulo de generación de respuestas

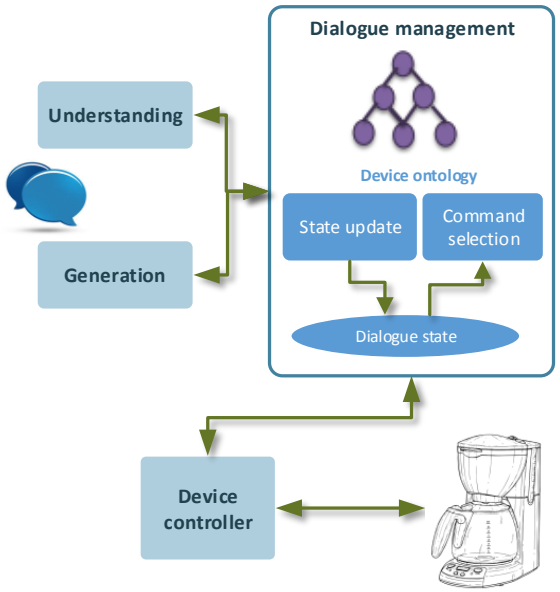


Figura 29. Arquitectura del sistema de diálogo

A continuación se describen los componentes más importantes del sistema de diálogo.

III.5.1.1 Ontología del dispositivo

La figura 30 muestra un ejemplo de la ontología que representa una cafetera. En este ejemplo, la cafetera puede estar en 6 estados distintos (CoffeeEmpty, DrinkReady, Off, MakingDrink, SelectDrink, WaterEmpty and Paused). Además, la cafetera tiene tres sensores (WaterLevel, Temperature, CoffeeLevel) que lanzan alertas de que el café o el agua se han acabado o cuando la bebida está lista (NoCoffee, NoWater, Drink_Ready). Por último, se pueden ejecutar tres acciones diferentes en el dispositivo (DrinkSelection, Start, and Stop).

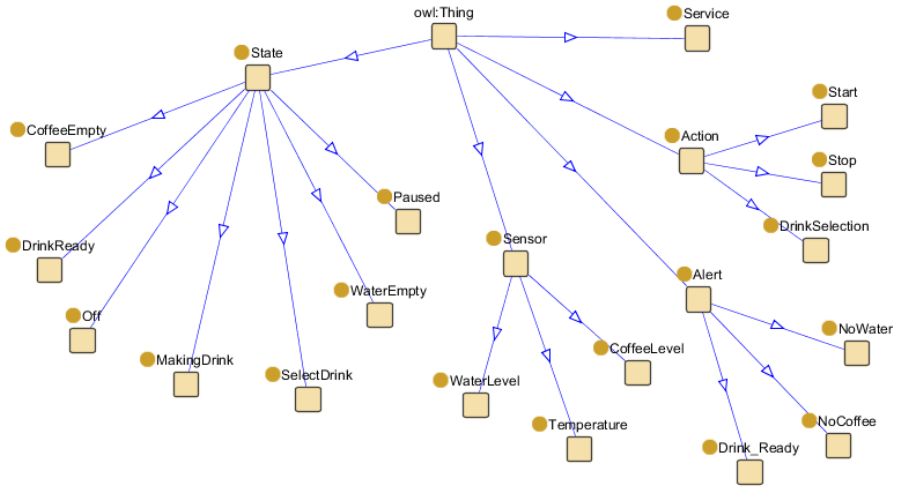


Figura 30. Un ejemplo de ontología para una cafetera

Esta ontología tiene X clases principales que se explican a continuación:

- State: Representa todos los posibles estados que puede tener el dispositivo.
- Sensor: Representa los sensores que contiene el dispositivo.
- Action: Representa las posibles acciones que puede realizar el dispositivo.
- Alert: Representa todas las alertas que puede detectar el dispositivo e informar de ellas.
- Service: Representa los servicios que permite ejecutar el dispositivo.
- Variable: Representa otras variables que puede tener el dispositivo y que puede ser interesante consultar.

III.5.1.2 Módulo de comprensión

El objetivo principal del sistema de comprensión del lenguaje natural es analizar si el texto recibido es una pregunta o un comando y procesar el significado del texto para poder gestionar este mensaje. Este módulo se basa en trabajos anteriores del grupo de investigación como *(Paredes-Valverde et. al., 2015)*, donde el texto recibido se representa mediante un modelo ontológico del mensaje que almacena información lingüística como el verbo principal, el foco del mensaje y

otra información como los elementos de la ontología referenciada. Más concretamente, la ontología que se instancia de muestra en la figura 31:

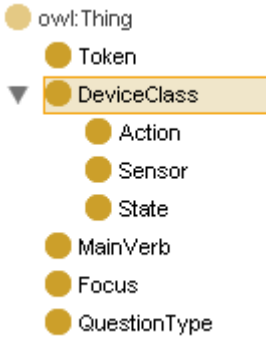


Figura 31. Modelo ontológico del mensaje

Esta ontología representa si el mensaje enviado puede ser una consulta o comando e identifica los elementos más importantes de la misma que se describen a continuación:

- **Token**: Tokens encontrados en el mensaje.
- **DeviceClass**: Son entidades que han sido encontradas en la ontología del dispositivo.
    - **Action**: Se ha encontrado una acción para ejecutarse en el dispositivo.
    - **Sensor**: Se ha hecho referencia a algún sensor.
    - **State**: Se ha hecho referencia a algún estado del dispositivo.
- **MainVerb**: Verbo principal de la oración.
- **Focus**: Elemento principal sobre la que versa el mensaje.
- **QuestionType**: Tipo de pregunta que se está realizando. Ejemplo: Qué, cómo, cuándo, porqué, etc.

III.5.1.3 Módulo de gestión del diálogo

El módulo de gestión de diálogo se basa en el lenguaje AIML Artificial Intelligence Mark-up Language) *(Wallace, 2003)*, que se basa en el lenguaje XML para desarrollar sistemas conversacionales. El lenguaje AIML es capaz de capaz de reconocer patrones predefinidos en un texto, como por ejemplo mensajes en un chat, y generar respuestas apropiadas en función de diversas condiciones y estados del sistema.

Este sistema de gestión del diálogo permite dar respuestas al usuario de forma aleatoria siguiendo un patrón preestablecido de comportamiento. Por otro lado el sistema de diálogo ofrece, además, un nivel de lógica tal como aleatoriedad entre una lista de posibles respuestas, condicionalidad, manejo de variables, redirección entre patrones sinónimos, etc.

El lenguaje AIML se compone de patrones de expresiones regulares junto con las respuestas que se deben proporcionar en el caso de cumplirse uno de estos patrones. Por ejemplo, el algoritmo 4 muestra un ejemplo de la definición de un sistema conversacional con dos patrones distintos. En el primer caso, si el intérprete identifica la frase "ERES UN ROBOT" definida dentro del elemento *<pattern>*, éste responderá con la frase definida dentro del elemento *<template>*, en este caso, "Sí, soy el robot más inteligente del mundo". En el segundo caso, si se obtiene cualquier conjunto de tokens y seguidamente "ES TU NOMBRE", entonces se activará este segundo patrón. Como también se puede observar la respuesta de este segundo patrón está parametrizada mostrando el atributo *"name"* del BOT creado con AIML.

```
<category>
    <pattern>ERES UN ROBOT</pattern>
    <template>
            Sí, soy el robot más inteligente del mundo.
    </template>
</category>
<category>
    <pattern>* ES TU NOMBRE</pattern>
    <template>
            Mi nombre es <bot name="name"/> y estoy encantado de hablar
    contigo.
    </template>
</category>
```

Algoritmo 4. Algoritmo ejemplo de definición de 2 patrones en AIML

A partir de la descripción de los patrones, los intérpretes actuales lo transforman a una máquina de estados. Por ejemplo, en la figura 32 se muestra un ejemplo de máquina de estados para AIML.

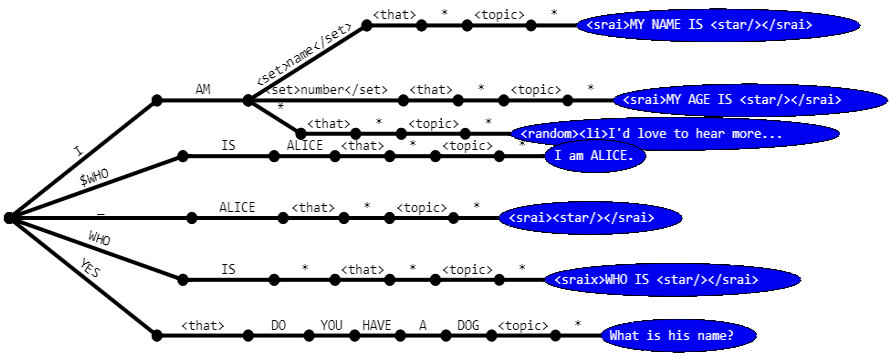


Figura 32. Ejemplo de representación de estados en el lenguaje de marcas que se utiliza en AIML

En definitiva y como se ha visto anteriormente AIML se utiliza para definir, fácilmente plantillas de patrones que producen una respuesta basada en una correspondencia simple de tokens de entrada. Sin embargo, AIML no fue diseñado para el procesamiento del lenguaje o comprender frases, por lo que se diseñó una versión ampliada del lenguaje AIML en la que no se utilizan tokens, sino que se utilizan conceptos de la ontología del dispositivo y el modelo ontológico del mensaje procesado por el módulo de comprensión.

Para ello se extendió el elemento *<pattern>* para poder representar distintas expresiones que contentan más semántica que simplemente un conjunto de tokens. Además, se añadió un nuevo elemento *<action>* que indica una codificación única a realizar por el dispositivo y que se le enviará al controlador del dispositivo para que se ejecute. Además, dentro del elemento *<template>* donde se incluyen las respuestas se incluirán distintas respuestas si el comando se ha ejecutado correctamente o no.

El algoritmo 5 se muestra un ejemplo de tres patrones del dispositivo cafetera.

```
<category>
    <pattern> ACTION==DrinkSelection && STATE==Pause</pattern>
    <action>MAKE_COFFEE</action>
```

```
    <template>
        <success>Preparando su café</success>
        <fail>Lo siento, pero ha habido un problema con la selección de la
    bebida</fail>
    </template>
</category>
<category>
    <pattern> QUESTION_TYPE==What && STATE </pattern>
    <template>
        El estado de la cafetera es <bot state/>.
    </template>
</category>
<category>
    <pattern> ALERT==NoCoffee </pattern>
    <template>
        Se ha acabado el café de la máquina. Por favor, rellénelo cuando
    pueda.
    </template>
</category>
```

Algoritmo 5. Algoritmo ejemplo de definición de 3 patrones en el dispositivo cafetera

El primer patrón del algoritmo 5 representa que se ha identificado en el mensaje la acción de seleccionar bebida y que el estado de la cafetera es que está en pausa. En este caso se enviará un comando al controlador del dispositivo para que haga el café. En el caso en el que el comando se ejecute correctamente se devolverá la respuesta incluida en el elemento *<success>* y en el caso de error se mandará la respuesta incluida en el elemento *<fail>*. En el segundo patrón se pregunta por el estado del dispositivo. Como se puede observar esta respuesta mostrará el estado del dispositivo con el texto "El estado de la cafetera es <bot state/>." En este caso se obtendrá el estado de la variable interna del bot denominada *state*. Por último, el tercer patrón indica que si llega una alerta de que se ha acabado el café (NoCoffee) entonces el módulo de generación de respuestas mostrará al usuario el mensaje contenido en la etiqueta *<template>*.

Resumiendo, el sistema funciona de la siguiente manera. Cuando llega una entrada, el módulo de diálogo obtiene los posibles patrones (comandos o consultas) que se asemejan más al texto recibido. También analiza y chequea si es posible llevar a cabo el comando o consulta y ejecutarlo cambiando su estado y comunicándose con el controlador de dispositivo. Una vez la operación se ha realizado correctamente, el módulo de generación del lenguaje obtiene una de las posibles respuestas para esa acción o consulta de ese dispositivo de manera aleatoria. Por ejemplo, si llega un comando, este se ejecuta por el controlador del dispositivo que indica si el comando se ha ejecutado correctamente. Entonces el sistema de gestión del diálogo establece una comunicación con el módulo de generación que obtiene la respuesta en lenguaje natural adecuada y la envía al usuario a través de la aplicación de chat.

Es importante destacar que el controlador del dispositivo no ejecuta solamente el comando, sino que también monitoriza las alertas que se envían desde los sensores del dispositivo y gestiona su estado. En este caso, el controlador avisará al sistema de gestión de diálogo con que existe una alerta para que pueda avisar al usuario.

III.5.2 Controlador del dispositivo

El controlador del dispositivo lo compone el hardware destinado al control del funcionamiento del dispositivo. En este proyecto se ha diseñado y fabricado un controlador basado en entradas y salidas digitales que se conectarán por un lado a la RB Pi por medio de un puerto de comunicaciones serie y por el otro a una cafetera de goteo marca Ufesa tal y como se presenta en la figura 33 conectadas a una placa electrónica equipada con en entradas y salidas digitales.

Figura 33. Imagen del tipo de cafetera "automatizada" en este proyecto

Para el control de esta cafetera se ha empleado una señal digital para saber el nivel de agua, otra entrada para conocer el estado (on/off) y una señal digital para encender o apagar la cafetera.

III.5.3 Módulo de generación de respuestas

Como se ha visto en el módulo de gestión del diálogo, en este módulo, sólo se obtienen los mensajes predefinidos que se han definido en los patrones definidos por el lenguaje AIML extendido que hemos definido para esta tesis doctoral.

III.5.4 Configuración inicial y registro del Bot

Para la configuración inicial del es necesario por un lado dar de alta al dispositivo en la aplicación del usuario y por otro lado enviarle la base de conocimiento que está formada por la ontología del dispositivo y un fichero con los patrones definidos en el lenguaje AIML extendido que se ha explicado anteriormente.

El usuario debe descargar la aplicación Im4Things del servidor/market (1). Una vez descargada e instalada en el terminal móvil/tablea, podrá ejecutarla y darse de alta en la plataforma (2). El procedimiento para el alta se puede realizar de dos formas distintas:

Opción 1: introduciendo la dirección de correo electrónico y contraseña.

Opción 2: Extraer los datos de otra red social (Facebook o Twiter) y contraseña.

Una vez registrado el usuario, a partir de entonces, tendrá a su disposición el chat con las funcionalidades de **envío de mensajería instantánea y multimedia**.

Cuando se conecta el Bot (4) a la red eléctrica, comienza en modo configuración. El usuario deberá activar el Bluetooth para localizar el Bot, sincronizarse y enviar los datos de:

- *Username* administrador.
- Parámetros de la WiFi.
- *Password.*

Una vez registrado estos datos en el Bot, automáticamente sincroniza los datos por HTTP en el servicio Im4ThingsCloudService a la vez que recibe un *username* y *password*. Cuando el Bot está registrado, el servidor envía al administrador de este dispositivo un "mensaje de bienvenida" para informar que el registro se ha realizado de forma exitosa. Llegado a este punto, el canal establecido por Bluetooth (4) se cierra y queda dispuesto para cualquier otra labor de administración o reconfiguración. Cuando finaliza el alta del Bot en el servidor, este pasa a tener entidad propia con un nombre o *nick*. De esta forma el usuario podrá invitar a este Bot en el chat por medio del nombre designado. Además, podrá identificar al Bot por medio de una fotografía que quedará reflejada en la base de datos del servidor. A continuación se explican distintos detalles de este proceso.

III.5.5 Intercambio de mensajes Bluetooth para la configuración del Bot

Para la configuración inicial del Bot se ha implementado un protocolo sobre la tecnología Bluetooth (comunicación User-Bot), así como sobre el protocolo HTTP (comunicación Bot-Server), que permitirá a un usuario que haya adquirido un Bot añadirse como administrador del mismo y transferirle los datos necesarios para que el Bot pueda conectarse y registrarse en el servidor.

El proceso de configuración del Bot se realizará siguiendo los siguientes pasos:

1. El Bot estará en modo configuración inicial (bluetooth abierto esperando conexiones).
2. El usuario desde su dispositivo móvil selecciona al Bot de la lista de dispositivos bluetooth y le envía un mensaje con los parámetros de configuración necesarios al Bot para su conexión con el servidor.
3. El Bot guarda la información recibida del usuario y utiliza la misma para conectar con el servidor, enviándole un mensaje HTTP de registro en el sistema.
4. El servidor valida la información recibida del Bot y lo da de alta en el sistema, informando al Bot con un mensaje de respuesta que el registro fue realizado, en el cual se incluirá la datos necesarios del Bot para iniciar la sesión en el servidor XMPP. Si se produce algún error, le responderá con un mensaje de error indicándole el motivo del mismo.
5. El Bot recibe el mensaje de respuesta del servidor con la información necesaria para conectarse al servidor XMPP e informa al usuario con un mensaje sobre Bluetooth. Si se produjo un error en el registro, informará al usuario con un mensaje de error y volverá al estado de configuración inicial.
6. El usuario recibe el mensaje del Bot que le indica si el proceso finalizó correctamente o no. En el primer caso, guardará la información recibida del Bot para poder comunicarse con el mismo a través del servidor XMPP, o puede enviarle un mensaje para cancelar el proceso, obligando al Bot a volver a su configuración inicial.

III.5.6 Seguridad en el servidor para evitar registros fraudulentos de Bots

Se ha desarrollado un protocolo de seguridad utilizado para el registro de Bots pensado para evitar fraudes y ataques contra el servidor. La seguridad en este tipo de sistemas es fundamental porque si se viola el servicio por personas ajenas al sistema podría hacer funcionar de forma malintencionada los dispositivos. Sirva de ejemplo que se violase la seguridad y se tomase el control de los dispositivos asociados a una vivienda. El hacker podría tomar el control de la vivienda y

provocar el caos. Como consecuencia el usuario dejaría de utilizar este tipo de plataformas.

Este protocolo nos proporciona la oportunidad de que en caso de ataque al servidor podamos recuperar el servicio de registro de Bots de forma fiable y segura sin necesidad de tener que modificar código en el Bot, con la consiguiente ventaja de que los Bots no se verán comprometidos al ataque. Cuando el servidor detecta que una llamada a un bot procede de un usuario no registrado para este cometido, automáticamente cambia la llamada al código de seguridad que valida la acción, es decir, la orden que envía el hacker no sería ejecutada porque primero se debe superar el registro del usuario.

Descripción de nomenclaturas:

• **Token:** cadena de caracteres que el Bot proporciona al servidor para que éste lo autentique como un Bot único y seguro.

• **Secreto:** cadena de caracteres aleatorios solamente conocido por el servidor con el que se crearán los tokens.

Se procede a describir el protocolo:

1. Implementación de un número conveniente de algoritmos aleatorios para proporcionar un token de verificación al Bot. El número de algoritmos será equivalente a las oportunidades de recuperar los tokens en caso de ataque al servidor.
2. Estos algoritmos son guardados en un sitio ajeno a los Bots y al servidor junto a un secreto único por algoritmo y su numeración, que nos servirá el índice para encontrarlo.
3. En el servidor está implementado solamente uno de esos algoritmos para comprobar el token.
4. En cada Bot se guardarán tantos tokens como algoritmos hayan, estos tokens serán producidos por dichos algoritmos (cada Bot tiene tokens

únicos).

5. Para saber qué token tiene que utilizar para autenticarse, el Bot realizará una consulta HTTP al servidor, preguntándole qué algoritmo está usando, el servidor devolverá el número del algoritmo que está utilizando (índice) y el Bot se autenticará con el token relacionado a ese número de algoritmo.
6. Cuando el Bot se registra, enviará el token apalabrado con el servidor, y si todo es correcto le dejará registrarse correctamente.

En caso de ataque:

1. Cuando el servidor es atacado y consiguen el algoritmo para sacar los tokens, lo primero que hay que hacer es proteger el servicio, expulsar al atacante y asegurar de nuevo el servidor para que no vuelva a entrar.
2. Una vez realizados los casos claves, se cambiará el algoritmo en el servidor a otro previamente pensado e indexado.
3. El atacante ya no sabe cuál es el nuevo algoritmo.
4. El Bot puede registrarse normalmente. Simplemente utilizará otro token de los que él tiene.

III.5.7 Diseño de protocolos específicos basados en el estándar

Las comunicaciones del sistema Im4Things se basan en el intercambio entre entidades de diferentes mensajes definidos en el protocolo de comunicación XMPP, el cual ha sido extendido para permitir añadir diferentes entidades (humanos y Bots) al sistema, así como para permitir la interacción entre ellos. En concreto, se han desarrollado las siguientes características:

III.5.8 Configuración del perfil de los Bots

Todas las entidades del sistema tendrán un perfil, que estará comprendido por un nickname y su avatar. El administrador del Bot podrá cambiar el perfil del Bot siempre que lo desee; tanto el avatar como el nickname del Bot. Para realizar este proceso, se realizan los siguientes pasos:

1. El administrador del Bot enviará un mensaje con la petición de cambio de perfil al Bot.
2. El Bot recibe el mensaje, obtiene su nueva configuración de perfil y actualiza la información en el servidor enviándole un mensaje con su nueva configuración de perfil.
3. El servidor almacena la información del Bot, almacenando su nuevo perfil y respondiendo al Bot con un mensaje indicándole el resultado de su petición.
4. El Bot recibe el mensaje con el resultado:

   4.1 Si el cambio se realizó correctamente, el Bot envía un mensaje de notificación sobre el cambio realizado a las distintas entidades interesadas en los cambios de estado del Bot.

   4.2 Si se produjo algún error, el Bot informará al administrador sobre el mismo con un mensaje de error.

III.5.9 Administración de una lista de control de acceso para los Bots

Cada Bot gestionará una lista de control de acceso (en adelante LCA) donde almacenará las distintas entidades de Im4Things que tienen permiso para comunicarse con el Bot. En un principio, la única entidad que puede interactuar con el Bot es el administrador del mismo.

Así mismo, sólo el administrador podrá modificar la LCA, añadiendo o eliminando otras entidades a la misma.

El proceso para añadir/eliminar entidades de la LCA es el siguiente:

1. El administrador envía un mensaje al Bot indicándole las entidades de desea añadir/eliminar del Bot.
2. El Bot procesa la lista recibida y envía un mensaje con la nueva LCA al servidor.
3. El Bot recibe la respuesta sobre los cambios realizados en el servidor, dándose los siguientes casos:

3.1 El servidor almacenó correctamente la LCA. A continuación el servidor envía un mensaje de notificación a los usuarios pertenecientes a su LCA, indicándoles las nuevas entidades que han sido añadidas o eliminadas.

3.2 El servidor informó de un error. En este caso el Bot envía un mensaje a su administrador sobre el error producido, dejando sin modificar su LCA.

Además de añadir/eliminar entidades a la LCA, cualquier entidad asociada a un Bot podrá consultar su LCA. El proceso es el siguiente:

1. Una entidad envía un mensaje al Bot para obtener su LCA.
2. El Bot envía un listado con las distintas entidades añadidas actualmente a su LCA.

Si una entidad no tuviera permiso para comunicarse con el Bot, el servidor descartará dicha petición.

III.5.10 Consulta del estado de los Bots y envío de comandos a los mismos

Una entidad (humano o máquina) puede comunicarse con un Bot y obtener el estado actual del mismo, así como la lista de comandos que puede realizar el Bot. Cualquier usuario que se encuentre en la LCA del Bot podrá consultar el estado del Bot y enviarle comandos. En cuanto a la comunicación entre Bots, se añade una primera fase de "descubrimiento", en la cual ambos Bots interactúan entre sí para ver si son compatibles y pueden comunicarse a través del envío de comandos.

Un usuario podrá conocer el estado del Bot enviándole un mensaje de petición de estado, al cual el Bot responderá con un mensaje indicando el estado actual en el que se encuentra.

Como se comentó previamente, un usuario podrá enviar comandos a un Bot usando el lenguaje natural, pero también podrá pedirle al Bot una lista de

comandos que permitirá al usuario ejecutarlos sin tener que escribir esas órdenes en lenguaje natural. El proceso sería el siguiente:

1. El usuario envía un mensaje al Bot pidiéndole las acciones que realiza.
2. El Bot responde al usuario con un mensaje incluyendo sus acciones.

En cuanto a la comunicación entre Bots, dado que podrán haber máquinas de diferentes fabricantes, con funcionalidades distintas entre ellas, se añade un primer proceso de descubrimiento entre Bots para conocer si ambos Bots entienden el mismo lenguaje o acciones, En concreto, este proceso se basa en la implementación, por parte de los Bots, de "interfaces", y es definido de la siguiente manera:

1. Un Bot (BOT-A) recibe de su **admin** un mensaje para incluir en su LCA a otro Bot (BOT-B), tal y como se define en los puntos [III.3.5.4 y III.3.5.5] de este capítulo.
2. Una vez añadido, el BOT-A envía un mensaje al BOT-B preguntándole si implementa la "INTERFAZ-1".
3. El BOT-B recibe la petición. Si implementa dicha interfaz, responderá al BOT-A con un mensaje afirmativo, en caso contrario responderá con un error.
4. Si el BOT-A recibe una confirmación sobre la interfaz implementada, puede pedir los comandos del BOT-B usando el mecanismo definido anteriormente.

III.5.11 Actualización de la base de conocimiento del Bot

Las respuestas o acciones de los Bot a las distintas entradas enviadas por los usuarios se encuentran almacenadas en una base de conocimiento que está formada por la ontología del dispositivo y un fichero con los patrones AIML extendidos que puede ser actualizada por el administrador del Bot. El proceso de actualización es el siguiente:

1. El administrador debe obtener la URL desde la que podrá descargarse la nueva base de conocimiento, enviando un mensaje al servidor.
2. El servidor responde con la URL y el administrador la reenvía dentro de un mensaje al Bot.
3. El Bot recibe la URL desde donde descargar la nueva base de conocimiento para instalarla.
4. Una vez instalada, el Bot responderá con mensaje al administrador indicando que se realizó la actualización.
5. Si se produjo algún error, se enviará un mensaje al administrador con dicho error.

III.5.12 Resetear el Bot a su estado de configuración inicial

En cualquier momento un administrador podrá restablecer un Bot a su configuración inicial. El proceso de reseteo el siguiente:

1. El administrador envía un mensaje al Bot indicándole que debe restablecerse a la configuración inicial.
2. El Bot envía al servidor una petición para que se eliminen todos los datos referentes a él, y elimina toda la información local para poder iniciar una nueva configuración del Bot.
3. El Bot comunica al administrador que se ha completado el reseto enviándole un mensaje de confirmación. En este punto el Bot no puede recibir ningún tipo de mensaje hasta que vuelva a ser configurado.
4. Si todas las operaciones fueron correctas, el Bot devolverá una IQ de tipo 'result':
5. Si se produjo algún error, se enviará un mensaje al administrador con dicho error.

Capítulo IV.- Validación del sistema

IV.1 INTRODUCCIÓN

En este capítulo se detalla la validación del sistema propuesto en esta tesis doctoral por usuarios finales interesados en los resultados del proyecto.

Aunque, como se ha visto en el capítulo II el sistema propuesto por esta tesis está formado por varios módulos, aplicaciones y servicios, la validación de este apartado se centra en el dispositivo y más concretamente en la parte del sistema de diálogo.

Durante el desarrollo de toda la tesis doctoral se hicieron pruebas unitarias, de seguridad y eficiencia en cada componente desarrollado, pero además, era necesario validar el sistema desde el punto de vista de su usabilidad por los usuarios y la precisión y exhaustividad del sistema propuesto para reconocer las órdenes y consultas sobre el estado de distintos dispositivos.

En los siguientes apartados se explican las medidas de evaluación utilizadas para validar el sistema y cómo se realizó dicha evaluación.

IV.2 MEDIDAS DE EVALUACIÓN

La evaluación del rendimiento de los sistemas de control de comandos y los sistemas de diálogo se suele realizar mediante la utilización de un conjunto de métricas de evaluación típicas de la disciplina de extracción de información: "precisión", "cobertura" y "medida-F". Estas métricas, son de aplicación muy común en los procesos de evaluación de sistemas de procesamiento del lenguaje natural y de recuperación y extracción de información *(Salton, et. al 1983)*. Estas medidas se diseñaron para evaluar el rendimiento de sistemas de búsqueda y recuperación de información como buscadores, pero se han adaptado para medir

el rendimiento en casi todas las aplicaciones de procesamiento del lenguaje natural.

Según *(Salton, et. al 1983)* la precisión se define como una medida de exactitud y determina la fracción de documentos relevantes de todos los documentos recuperados en un sistema de recuperación de información. Acorde con esta definición, la precisión se puede calcular a partir de la fórmula (1).

$$precisión = \frac{|\{documentos\ relevantes\} \cap \{documentos\ recuperados\}|}{|\{documentos\ recuperados\}|} \quad \textbf{(1)}$$

Por otro lado, la cobertura es una medida de la integridad y determina el número de documentos relevantes recuperados de todos entre todos los documentos relevantes de esa búsqueda. Acorde con esta definición, se puede calcular a partir de la fórmula (2).

$$cobertura = \frac{|\{documentos\ relevantes\} \cap \{documentos\ recuperados\}|}{|\{documentos\ relevantes\}|} \quad \textbf{(2)}$$

Por último, la media-F es la media armónica de los valores de precisión y exhaustividad *(Yang, y Liu, 1999)*. En concreto, la fórmula de la medida-F (del inglés, 'F-Measure, balanced F-Score o F1 measure') se utiliza para evaluar el rendimiento global de la precisión y cobertura. Acorde con esta definición, la medida-F se puede obtener a partir de la fórmula (3). Al igual que ocurre con la precisión y la cobertura, el rango del valor de la medida-F es un número real entre 0 y 1.

$$medida - F = 2 \cdot \frac{precision \cdot exhaustividad}{precision + exhaustividad} \quad \textbf{(3)}$$

Estas métricas también se pueden especificar mediante la terminología de falsos positivos y falsos negativos *(Olson, y Delen, 2008).* En las ecuaciones 4 y 5 se puede observar las fórmulas de precisión y cobertura en términos de falsos positivos y falsos negativos y sería el total de aciertos positivos. Aquí la precisión es el resultado de los aciertos del sistema divididos por todos los aciertos y los falsos positivos. Por el contrario la cobertura se calcula como el número de aciertos dividido por el número de aciertos y los falsos negativos. En el caso de la evaluación de la tesis doctoral, los falsos positivos representa las veces que el sistema ha ejecutado algún comando o consulta que no era la que solicitó el usuario y los falsos negativos representan los comandos en los que se ha equivocado el sistema además de en los que el sistema no ha realizado ninguna acción y debería haberlo hecho.

$$precisión = \frac{TP}{TP + FP} \quad \textbf{(4)}$$

$$cobertura = \frac{TP}{TP + FN} \quad \textbf{(5)}$$

El sistema de diálogo y control de comandos propuesto en esta tesis doctoral se evaluó adaptando estas métricas. La **precisión** mide entonces el número de comandos y consultas que el sistema detecta correctamente y realiza una acción o devuelve una respuesta adecuada, dividido por el número de comandos y consultas que fueron ejecutadas o respondidas por el sistema. Por otro lado, la **cobertura** mide el número de comandos y consultas que el sistema resolvió correctamente dividido por el número total de comandos y consultas realizadas por el usuario. Más concretamente las medidas utilizadas para precisión y cobertura se muestran en las ecuaciones 6 y 7.

$$precisión = \frac{RespuestasCorrectas}{TotalRespuestasEjecutadas} \quad \textbf{(1)}$$

$$cobertura = \frac{RespuestasCorrectas}{TotalConsultasYComandosRealizadas} \quad \textbf{(2)}$$

Así, si un valor de precisión de un 80%, significa que 80 de cada 100 comandos o consultas que se ejecutaron en el sistema, se ejecutaron de forma correcta. Y un valor de cobertura de 70% indica que 70 de cada 100 comandos o consultas realizadas por el usuario tuvieron un comportamiento correcto en el sistema.

IV.3 EXPERIEMENTO DE VALIDACIÓN

Lo primero que se realizó para la validación fue el diseño de un test sociolingüístico para 5 dispositivos, a saber, cafetera, sistema de riego, control de iluminación interior, control de persianas y calefactor de aire (ver Anexo 1. Test sociolingüístico) que permitiese detectar sin influir en los usuarios finales, cuál sería la forma natural de interactuar por cada uno de los dispositivos. Una vez diseñado este test, se les pasó a 50 personas incluyendo desarrolladores de la empresa y potenciales clientes finales. Se les realizó una presentación del sistema y se les solicitó que rellenasen el test sociolingüístico antes de utilizar el sistema.

A los 50 testeadores se le facilitó la descarga de la aplicación App im4Things a través de los distintos markets en función de la tecnología de móvil/tableta (Android E iOS). El acceso a la aplicación es restringido y solo se le dio acceso a los testeadores. Para ello el registro se efectuó a través de número de teléfono tal y como se describe en el capítulo III. En la figura 34 se puede ver una conversación de chat entre uno usuario y la cafetera. Durante esta conversación se le solicita ciertas acciones a la cafetera y esta responde en función del estado. Toda la conversación se mantiene en lenguaje natural.

El aspecto gráfico del chat es muy similar a otros famosos chat ya que este proyecto ha intentado seguir la línea de facilidad de uso y experiencia previa del usuario. En la figura 35 se puede observar distintas área del chat como son la fotografía del usuario/dispositivo, que en esta ocasión es la de la cafetera, estado de la cafetera, es decir si está en línea o no. En lo que podríamos llamar escritorio, que es la región del chat donde se suceden los distintos mensajes entre el usuario y la cafetera se describe el historial los últimos mensajes con la información añadida de si el mensaje ha sido enviado y leído por el destinatario. Ya en la parte baja del chat, aparece el cuadro donde el usuario escribe el texto a enviar a la cafetera. Según la versión de móvil/tableta, el usuario dispone de corrector ortográfico. En caso de que el dispositivo no entienda el mensaje que le ha enviado el usuario, este le contestará de forma aleatoria con mensajes del tipo *"por favor repita de nuevo el mensaje"* o *"no le he entendido bien".*

Para mejorar la experiencia de uso se ha dotado a la aplicación App im4Thing el envío de emoticonos. Los emoticonos son muy útiles ya que a través de una imagen, por ejemplo la taza de café, la cafetera entiende que el usuario ordena que se ponga en funcionamiento y prepare un café.

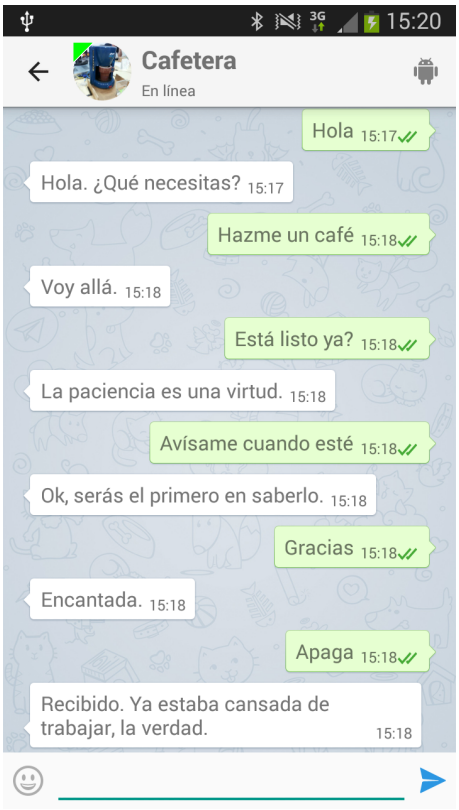


Figura 34. Consola del chat im4Things durante una conversación con la cafetera. Los mensajes que aparecen en fondo amarillo son los enviados por el usuario.

Como se ha comentado, se rellenaron 50 test con las formas en los que los usuarios podrían interactuar con cada uno de los dispositivos propuestos. Por ejemplo, en la tabla 10 se muestran distintas formas de realizar acciones para los dispositivos **cafetera, sistema de riego, control de iluminación interior, control de persianas y calefactor de aire**:

Dispositivo	Comando en lenguaje natural	Comando / Consulta
Cafetera	Hazme un café	Comandos: DrinkSelection + Start
	Prepara un café corto	
	Empieza a hacer café	
	¿Tienes café?	Consulta estado: EmptyCoffee
	¿Te queda café?	
	¿Tengo que ponerte café?	
	¿Has terminado con el café?	Consulta estado: DrinkReady
	¿Está listo ya?	
	Avísame cuando esté	
	Apaga	Comando: TurnOff
	Descansa	
Sistema de riego	Riega las plantas	Comandos: Water + Start
	Enciéndete	
	Actívate	
	A regar	
	¿Hay agua?	Consulta estado: ThereWater
	¿Tienes agua?	
	¿Está la válvula abierta?	
	¿Has terminado?	Consulta estado WaterReady
	¿Estás encendido?	
	¿Estás regando?	
	Apágate	Comando: TurnOff
	No riegues más	
	Corta el grifo	
Control de iluminación interior	Enciéndete	Comandos: Light + Start
	Quiero luz	

	Enciende derecha	
	Enciende izquierda	
	Nivel de luz	Consulta estado: StatusLight
	Cuánto tiempo llevas encendida	
	En que escena estás	
	Estás averiada	
	¿Hay alguna lámpara rota?	
	Apágate	Comando: TurnOff
	Apaga escena 1	
	Apaga escena 1	
	A negro	
Control de persianas	Hace sol	Comandos: Blind + Start
	Sube	
	Baja	
	¿estás subida?	Consulta estado: StatusBlind
	¿cómo estás?	
	¿estás bajada?	
	¿estás rota?	
Calefactor de aire.	Enciéndete	Comandos: Heater + Start
	A trabajar	
	Tengo frío	
	Regúlate a 26 grados	
	¿estás funcionado?	Consulta estado: StatusHeater
	¿a qué temperatura estás?	
	¿cuánto tiempo llevas encendido?	
	Apágate	Comando:

	Vete a dormir	TurnOff
	Corta el aire	

Tabla 10. Ejemplos de formas de interacción con los dispositivos obtenidos.

Todos los comandos y consultas en lenguaje natural obtenidos a partir de los tests sociolingüísticos se introdujeron en el sistema utilizando la App im4Things y se comprobó si el comportamiento realizado era el mismo que el comportamiento esperado por el dispositivo para poder calcular las medidas de **precisión, cobertura y medida-F**. En la tabla11 se puede observar el número de consultas y comandos que se ejecutaron, cuántos de estos comandos y consultas tuvieron el comportamiento esperado, cuántos comandos y consultas tuvieron un comportamiento erróneo y cuántos comando y consultas no tuvieron respuesta para los dispositivos cafetera, sistema de riego, control de iluminación interior, control de persianas y calefactor de aire.

Dispositivo (ordenados por orden del experimento)	**Comandos y consultas totales**	**Comandos y consultas con comportamiento esperado**	**Comandos y consultas con comportamiento erróneo**	**Comandos y consultas sin respuesta**
Cafetera	200	185	10	5
Sistema de riego	160	148	8	4
Control de iluminación interior	185	165	15	5
Control de persianas	80	68	6	6
Calefactor de aire	120	106	12	2
Total	745	672	51	22

Tabla 11. Estadísticas sobre comandos y consultas ejecutados.

Como se puede observar en la tabla11, el dispositivo cafetera falló en 10 ocasiones a las órdenes dadas por el usuario y en 5 ocasiones no obtuvo respuesta a una acción solicitada. Los fallos detectados fueron del tipo “haz una cosa” y ésta

no se realizó. Por ejemplo detectamos que estos fallos aparecían principalmente cuando el usuario enviaba la misma orden repetidamente durante un espacio de tiempo muy pequeño, por ejemplo *"haz café", "quiero café", "enciéndete"*.

De igual forma actuamos con el resto de dispositivos obteniendo unos resultados muy prometedores. Observamos que aquellos sistemas donde peor responde el sistema son aquellos donde al hardware se le asocian más de un elemento. En ejemplo lo encontramos en el control de iluminación interior. En este caso probamos instalamos en una habitación 4 luminarias asociadas de 2 en 2 para distinguir entre derecha e izquierda. El control consistía en encendido y apagar todo o sólo las del lado derecho o izquierdo.

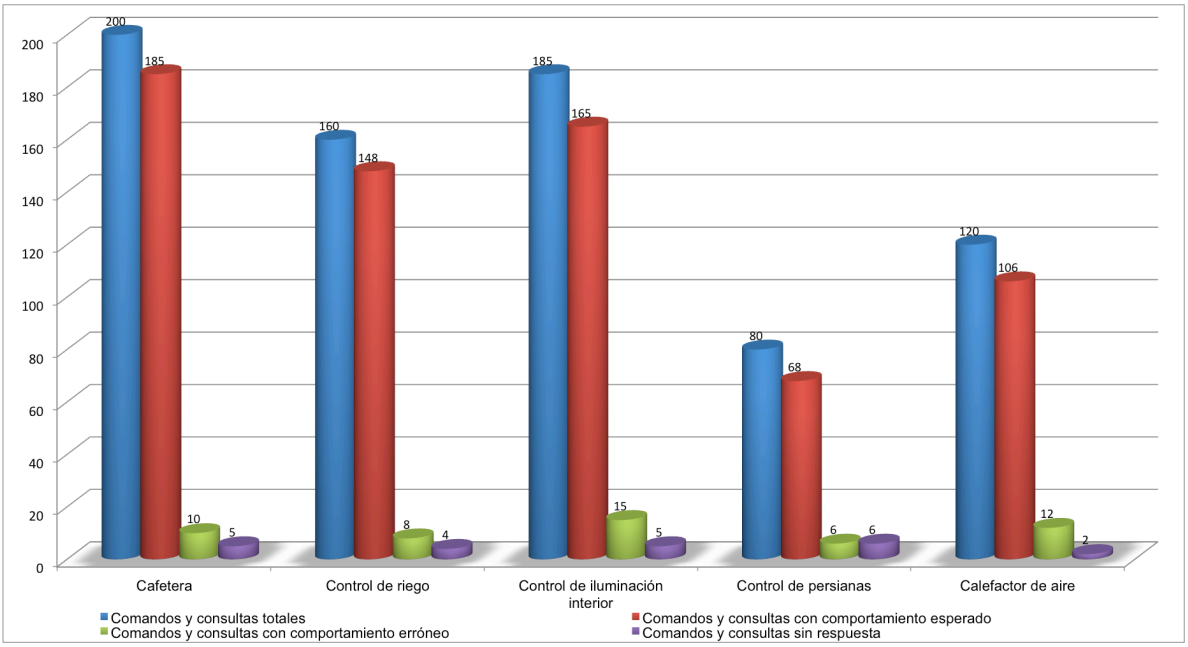
En este caso el sistema tuvo 15 ejecuciones erróneas. Pudimos observar que el 62% de estos fallos estaban relacionados con la posición en la habitación del usuario. Cuando los usuarios indicaban al sistema, por ejemplo encender izquierda, el sistema no sabe si el usuario está entrando o saliendo de la habitación y se daba el caso de que la orden de control encendiese el lado contrario.

Este hecho abre un nuevo campo de investigación futuro y es integrar en el sistema im4Things la posición del usuario en el interior de una vivienda o edificio por medio de posionamiento interior (*InDoor*), por ejemplo empleando tecnología Bluetooth.

Respecto al resto de sistemas los resultados obtenidos fueron muy similares a los obtenidos en el caso de la cafetera. En el caso particular del calefactor de aire introducimos la novedad de incorporar al dispositivo la señal analógica de temperatura de la sala donde hicimos los experimentos. En este caso el control del calefactor de aire estaba supeditado a la temperatura de la sala de forma que se programó la función <termostato> en el bot. El ratio de órdenes no ejecutadas de forma correcta fueron del tipo: *si la temperatura de la sala está supera cierto valor ponte en enciéndete.* Observamos que el 76% de los fallos correspondían a una media errónea de la lectura de la temperatura.

De nuevo se abre una nueva línea de investigación para buscar la integración óptima de sensores de campo en el sistema im4Things. Esta línea de investigación es ambiciosa en el sentido de que la información registrada por los sensores vuelca la información en el servidor para que sean gestionados y como consecuencia se envíe la orden de control al dispositivo.

En la gráfica 2 se resume el resultado de los 5 experimentos en formato código de barras.



Gráfica 2. Resultado parcial de comandos y consultas

Con los datos obtenidos en la tabla 11 se han calculado los datos de **precisión, cobertura y medida-F** que se muestran en la tabla 12. Como se puede observar la cafetera y el sistema de riego obtiene la mayor **precisión** con un 94,87% frente al 91,89% del controlador de persianas, al 91,67% del control de iluminación y al 89,83% del calefactor de aire. El resultado demuestra que el sistema es preciso en general para la gran variedad de sistemas probados y especialmente precisos en aquellos sistemas donde el control depende de algunas señales de estado digitales.

Otro dato importante es que el sistema responde bien ante diversos escenarios. Cabe destacar que no solo interviene la tecnología desarrollada en la App im4Things sino que el sistema está formado por muchos más sistemas que no vemos pero que están ahí. Nos referimos que el sistema alcanza una **cobertura**

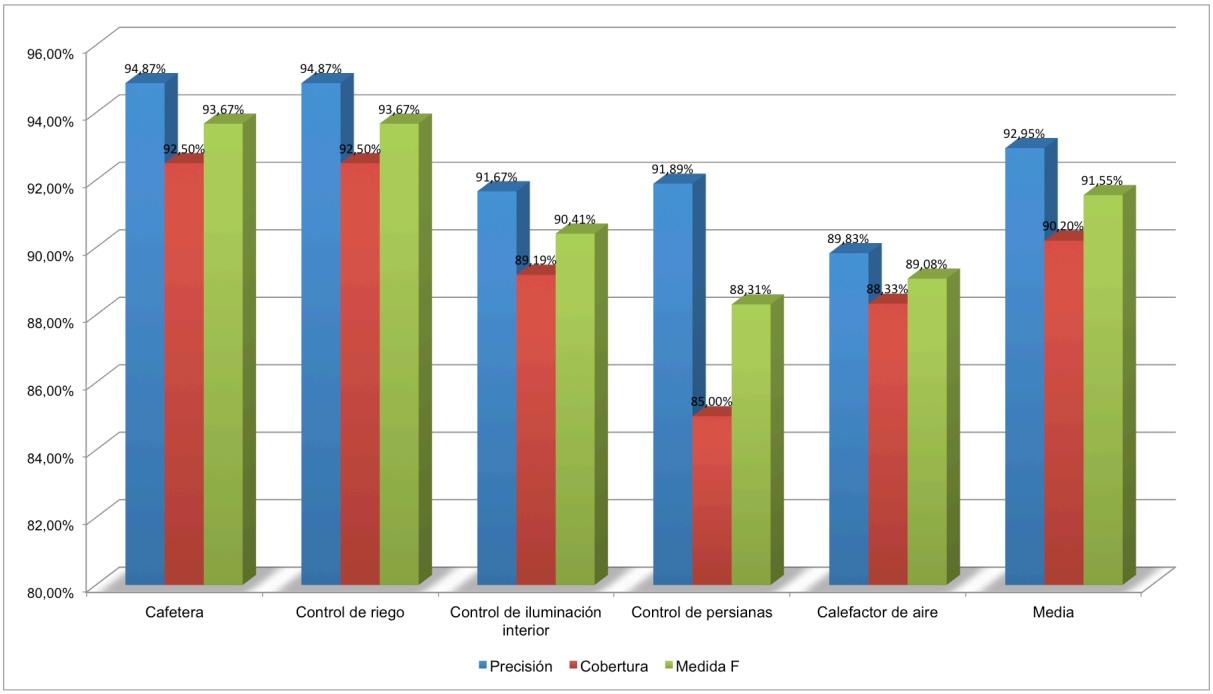
muy alta en el control de distintos sistemas y cabe decir que de un experimento a otro no se han realizado ajustes algunos de adaptación hasta que no se ha completado el mismo. Por lo tanto la adaptabilidad es muy buena con un ratio de entrono al 90% (**medida F**).

Por otro lado, observamos que a medida que íbamos realizando nuevos experimentos siendo el primero el realizado con la cafetera y el último con el calefactor de aire, se ha ido ganando experiencia en esta disciplina porque el reto tecnológico ha ido creciendo y la precisión se ha mantenido estable.

Dispositivo	Precisión	Cobertura	Medida F
Cafetera	94,87%	92,50%	93,67%
Sistema de riego	94,87%	92,50%	93,67%
Control de iluminación interior	91,67%	89,19%	90,41%
Control de persianas	91,89%	85,00%	88,31%
Calefactor de aire	89,83%	88,33%	89,08%
Media	**92,95%**	**90,20%**	**91,55%**

Tabla 12. Medidas de precisión, cobertura y Medida-F del experimento.

A modo de resumen en la gráfica 3 se compara las medidas entre ellas de cada experimento.



Gráfica 3. Resultados del cálculo de precisión, cobertura y medida-F de los 5 experimentos

IV.4 VALIDACIÓN CON USUARIOS FINALES

Una vez realizada esta validación, se seleccionaron a un conjunto de 50 usuarios finales a los que se les dejó estos dispositivos y la aplicación para que la evaluaran. El perfil de los usuarios seleccionados (ver tabla 13) fue confeccionado en función de la edad, sexo y experiencia de uso en redes sociales tipo chat.

Perfil/Usuario	Perfil 1	Perfil 2	Perfil 3	Perfil 4
Edad	15-25	26-40	41-50	50-65
Experiencia previa	Alta	Alta	Media	Baja
Número	22	18	6	4

Tabla 13. Perfil de los usuarios.

Respecto a la edad hubo paridad, es decir el 50% de usuarios del perfil 1 eran mujeres y así con el resto de perfiles. Los usuarios pudieron utilizar el sistema durante una semana para poder determinar la precisión y cobertura del sistema que obtuvo unos resultados muy similares a los mostrados en el apartado anterior.

De este experimento también se obtuvo como resultado el uso que hicieron del sistema fue bajo con una media del 9,36 usos al día por todos los usuarios. En la tabla 14 se muestra el número de comandos y consultas por dispositivo y por usuario durante todo el periodo. Como se puede observar, los dispositivos más utilizados fueron la Iluminación Interior con una media de 101 comandos y consultas seguido del Control de persianas, cafetera, calefactor de aire y sistema de riego con una media del 95, 94, 93 y 85 de usos respectivamente.

Usuario	Cafetera	Sistema de riego	Iluminación interior	Persianas	Calefactor de aire	Media Uso
U1	0	1	5	2	3	11
U2	1	4	2	2	2	11
U3	3	0	0	3	1	7
U4	0	1	1	2	1	5
U5	4	0	2	2	0	8
U6	0	3	1	0	1	5
U7	3	2	1	4	3	13
U8	1	2	1	2	2	8
U9	2	1	2	1	1	7
U10	2	0	2	1	2	7
U11	3	2	5	3	2	15
U12	2	1	1	1	0	5
U13	0	0	0	2	3	5
U14	1	2	3	1	3	10
U15	2	0	2	1	3	8
U16	2	3	5	2	2	14
U17	1	5	2	1	1	10
U18	2	0	2	2	1	7
U19	1	1	2	2	3	9
U20	2	2	2	1	1	8
U21	4	0	1	0	1	6
U22	1	1	5	2	3	12
U23	3	2	2	2	4	13
U24	4	2	1	2	1	10
U25	1	3	1	3	1	9
U26	1	0	0	4	2	7
U27	1	4	1	0	3	9
U28	4	2	1	2	2	11
U29	1	2	1	2	3	9
U30	2	1	3	3	0	9
U31	0	2	2	1	1	6
U32	1	1	4	3	3	12
U33	3	4	5	1	1	14
U34	1	0	4	0	1	6
U35	2	3	1	2	0	8
U36	2	1	0	1	3	7
U37	4	1	0	1	4	10
U38	2	1	3	4	5	15
U39	4	1	3	3	0	11
U40	1	2	2	1	2	8
U41	1	1	3	2	0	7
U42	1	3	1	1	2	8
U43	2	0	1	1	4	8
U44	2	2	3	4	1	12
U45	3	3	3	2	3	14
U46	0	4	2	4	0	10
U47	1	2	1	2	2	8
U48	5	1	2	1	2	11
U49	2	4	3	4	2	15
U50	3	2	1	2	2	10
TOTAL=	94	85	101	95	93	9,36

Tabla 14. Usos realizados por los usuarios.

También se les pidió que rellenaran una encuesta de satisfacción con el objetivo de conocer datos relativos al grado interés, grado de utilidad, facilidad de uso, qué tipo de electrodomésticos les gustaría fuesen controlados por el sistema y si recomendaría el sistema im4Things a otros usuarios. Toda esta información se adjunta como Anexo II. Encuesta de satisfacción, en esta tesis doctoral.

En general la satisfacción con los resultados obtenidos por la aplicación fue bastante alta y casi todos los usuarios estaban muy satisfechos con los resultados obtenidos. Por un lado los usuarios percibieron la novedad del sistema y la innovación subyacente del mismo. Eso hizo que acogieran el sistema con gran expectación lo cual al principio supuso un reto ya que la experiencia nos dice que si un sistema no está a la altura de la esperada caerá en el olvido. Por otro lado, los fallos que produjo el sistema pueden ser corregidos con lo que conseguiremos como resultado robustez del mismo. Respecto a la apariencia de la interface de usuario, será necesario mejorarla en lo relativo a colores y gráficos de los iconos aunque los usuarios valoraron positivamente la ergonomía de la misma, es decir, la funcionalidad que ofrece el chat es intuitiva y de fácil interpretación. Respecto a los electrodomésticos que los usuarios ven con más potencial de uso son principalmente los que tiene tareas programadas y con capacidad a ser programadas refiriéndose a cafetera, lavadora, secadora, horno, sistema de riego, y por otro lado otros sistemas relacionados con el confort como son aire acondicionado, alumbrado y persianas. Respecto a los sistemas de seguridad, los usuarios reflejan que estarían muy interesados en recibir información de detectores de presencia y humo principalmente así como aplicaciones para discapacidad y dependencia como por ejemplo un botón de ayuda para informar de casos de emergencia como caídas o enfermedad. Los usuarios reflejaron su recelo en el uso de im4Thing para el uso controlado por el sistema de eventos de una vivienda como por ejemplo la apertura de puertas y cajas de seguridad que en definitiva podemos resumir que de momento todo aquello que no intervenga directamente por el usuario.

IV.5 CONCLUSIONES

Los datos que hemos obtenido de los usuarios tanto de la fase de validación como de la encuesta de satisfacción nos indican que el sistema funciona muy bien pero sólo hemos validado los 5 sistemas comentados que son recordamos son cafetera, sistema de riego, control de iluminación interior, control de persianas y calefactor de aire y se deberían evaluar el sistema con más dispositivos para seguir mejorando tanto el funcionamiento del sistema como mejorar la experiencia de uso.

Respecto a los fallos de comandos y consultas con funcionamiento erróneo llegamos a la conclusión que cuando esto ocurría, el servidor enviaba repetidamente todas las órdenes al hardware del dispositivo para que este ejecutase la orden. En función de la congestión de red los retardos (tanto de gestión como de propagación) de la información son variables. Para reducir este efecto sobre el envío de información en nuestro sistema se decidió hacer un proceso de encolamiento de las peticiones en el servicio Im4ThingsCloudService de forma que cuando el servidor reciba una petición para realizar una acción, este gestione la primera y encole las que lleguen después (solo para aquellas que realizan la misma acción). Este procedimiento de encolado está controlado por un tiempo de espera de 0,8 segundos. Si durante este tiempo el servidor recibe respuesta positiva del dispositivo, es decir que se ha ejecutado la acción, borra el contenido de la cola de peticiones. Con ello conseguimos dos efectos: el primero es dar el tiempo suficiente a la red para enviar la información y segundo es no saturar al hardware con la misma acción.

Para las 5 ocasiones en que el usuario no obtuvo respuesta a una acción solicitada a la finalización de esta tesis doctoral no tenemos unas conclusiones concluyentes de por qué se producen estos fallos. Según nuestro estudio pueden ser consecuencia de limitación del hardware, posibles microcortes temporales del websockets u otros fallos relacionados con el camino que debe recorrer la información hasta llegar al usuario. No obstante estos fallos son una parte muy reducida de todo el tráfico gestionado no afectando al funcionamiento del sistema.

Durante la fase de elaboración del software de grupos de usuarios *(groupchat)* encontramos el problema mayor que cuando en un grupo conviven tanto usuarios como dispositivos, el sistema debe discriminar aquellas palabras habituales en una conversación entre usuarios pero que para el dispositivo significan una orden. Por ejemplo si en una conversación entre usuarios se dice café refiriéndose a "*cafetería café Madrid*" por ejemplo, la cafetera se pondrá a preparar un café. La solución que implementamos en este proyecto fue implementar grupos virtuales de notificaciones de forma que la comunicación fuese directa usuario-dispositivo pero las notificaciones procedentes de los dispositivos se informasen a un grupo de usuarios. Por ejemplo, supongamos el grupo virtual "cafetera familia" en el que dos personas puedan controlar la cafetera. Cuando una de ellas da una orden a la cafetera el sistema notifica también al otro usuario esa orden, por medio de notificaciones **push** (ver capítulo III, apartado III.8) y todos los mensajes generados desde la cafetera se notificará simultáneamente a ambos usuarios. Así si un usuario da la orden de preparar café el otro usuario está informado, no volviendo a preparar otro café y además también sabrá si la cafetera ha tenido un fallo.

Esta metodología ha supuesto posibilitar otras líneas de actuación enfocadas a personas con discapacidad. El sistema im4Things posibilita a una persona discapacitada a solicitar e informar a un grupo de personas sobre una necesidad. Por ejemplo en caso de caída, adaptando im4Things por medio de un botón de emergencia (similar al botón de llamada de emergencia) la persona necesitada puede enviar un mensaje simultáneo a un gran número de usuarios de un grupo virtual.

Capítulo V. – Conclusiones y líneas futuras.

V.1 CONCLUSIONES

Este trabajo representa un intento en la mejora de los sistemas de interacción con dispositivos y más concretamente, un intento de mejora en los sistemas de diálogo en lenguaje natural con distintos dispositivos y electrodomésticos. En la actualidad, existen diferentes tipos de propuestas basadas en diálogo inteligente para el manejo y la consulta de estado de distintos dispositivos como se ha detallado en el capítulo II. Sin embargo, la mayoría de ellas presentan algunas desventajas que dificultan su establecimiento como solución estándar. Entre todas estas carencias, la más difícil abordar es que se pueda consultar el estado del dispositivo, gestionar las alarmas y mantener la gestión del diálogo. Además de mantener la información sobre los posibles estados que puede tener el dispositivo y las posibles alarmas que se pueden ejecutar, es necesario poder mantener distintas variables para saber el estado en el que se encuentra el diálogo. Como se ha visto en los diferentes enfoques analizados en el estado del arte, se han seleccionado distintas características para comparar estos trabajos como si permiten la interacción en lenguaje natural, si es oral o escrito, si se permite la consulta de estado, si tiene gestión del diálogo, si el control de los dispositivos es centralizado o distribuido, qué estándares domóticos utiliza y si permite la gestión de alertas.

Entre los trabajos analizados, se vio que el sistema Mayordomo *(Espejo et. al 2010)* era uno de los sistemas más completos ya que permite la consulta del estado de los dispositivos, gestiona el diálogo y permite órdenes en lenguaje natural. No obstante, tenía algunas limitaciones como que el control es centralizado y no permite la gestión de alertas.

El sistema de diálogo propuesto en esta tesis proporciona un entorno estable y escalable a la vez que maduro y formal para poder comunicarse con los dispositivos a través de mensajería instantánea. Además, gracias al uso de

tecnologías de representación de conocimiento como son las ontologías, lo hace fácilmente configurable para cada dispositivo.

Además, como se ha comentado en el capítulo V, este sistema ha sido validado con diversos dispositivos y con usuarios finales obteniendo resultados prometedores que permiten concluir que este sistema está maduro para su futura comercialización.

Además, el sistema desarrollado cumple con un conjunto de características deseables y que han dado lugar a un conjunto de aportaciones de esta tesis doctoral que comentamos a continuación:

- α Desarrollo de un sistema de consulta y control de dispositivos mediante diálogo en lenguaje natural que satisface cada una de las características que han sido analizadas anteriormente. Entre las características que dispone este sistema se destaca que permite el envío de órdenes en lenguaje natural escrito, permite la consulta del estado de los dispositivos, realiza una gestión del diálogo y permite el tratamiento de alarmas. Todas estas características se realizan de manera distribuida y cada dispositivo es el encargado de controlar y gestionar las acciones y señales del propio dispositivo.
- α La utilización de protocolos seguros de mensajería instantánea entre dispositivos y humano/dispositivo a través de lenguaje natural. Actualmente, no existen estándares en el IoT que permitan comunicar los dispositivos de forma segura y eficiente.
- α Proporcionar un hardware barato que permita la comunicación y control distribuido de cada dispositivo. Además, el hardware debe ser y adaptable para ser integrable en los distintos electrodomésticos de cada fabricante. Para ello esta tesis doctoral propone utilizar hardware con puertos de comunicaciones estándar por ejemplo puerto serie y USB. Respecto a los protocolos de comunicación entre el hardware de im4Things y el electrodoméstico hemos hecho que este sea plug and play por medio del empleo de protocolos estándar como puede ser ModBus TCP.

- α Representación del conocimiento e información del dispositivo mediante ontologías. La base de conocimiento de los dispositivos son fácilmente configurables y reutilizables ya que para ello se ha desarrollado un modelo ontológico compuesto por distintos conceptos que definen un conjunto de propiedades de cada dispositivo que permite modelar todas las acciones y respuestas que debe tener cada dispositivo de manera sencilla.
- α Validación del sistema. El proceso de validación del sistema se ha realizado con distintos dispositivos. Para ello se realizaron distintos test sociolingüísticos para recabar la manera que los usuarios desearían comunicarse con los dispositivos. Además, el proceso de validación requirió de la participación de varios usuarios para validar los resultados y usabilidad del sistema proporcionando conclusiones muy interesantes y útiles, reafirmando la consecución de los objetivos definidos al inicio de la investigación.

V.2 LÍNEAS FUTURAS

En lo que respecta a las líneas futuras, en esta tesis doctoral se citan varios tópicos que no han sido considerados como parte relevante en el desarrollo de la misma, pero que proporcionan nuevas líneas de investigación que a continuación se señalan como trabajos futuros:

δ *Integración de tecnologías de reconocimiento del habla.*

El sistema descrito en esta tesis doctoral está basado en el reconocimiento del lenguaje natural escrito. Casi todos los sistemas para control de dispositivos en lenguaje natural lo realizan a través de tecnologías de reconocimiento del habla. Como línea futura se propone la ampliación de la funcionalidad de nuestro sistema introduciendo tecnologías de reconocimiento y generación del habla.

δ *Inclusión de un reconocimiento multilingüe.*

Para facilitar la comercialización del sistema propuesto en distintos mercados y distintas regiones es necesario integrar tecnologías de reconocimiento multilingüe que permita interactuar con los sistemas en distintos idiomas. Actualmente existen muy pocas soluciones que incorporan multilingüismo debido a que las reglas gramaticales difieren mucho de unas lenguas a otras y también es necesario un mantenimiento de la base de conocimiento en estos idiomas. Un ejemplo es la aproximación presentada en *(Jordan & Araki, 2014)* donde se presenta una aproximación independiente del idioma y ha sido probada en Inglés, Francés y Japonés.

Como trabajo futuro se plantea utilizar tecnologías de traducción automática para poder traducir y adaptar las bases de conocimiento a distintos idiomas de manera sencilla como realizan trabajos actuales de localización de ontologías *(Mejía et. al 2012)*.

δ *Mejora del sistema de diálogo para convertirlo en multimodal.*

El sistema propuesto en esta tesis doctoral permite la comunicación entre dispositivos y humanos mediante texto en lenguaje natural escrito. Los sistemas de diálogo cada vez incorporan otras maneras de interactuar que los vuelven

multimodales. Las entradas de un sistema de diálogo multimodal pueden ser un subconjunto de las diversas modalidades como el habla, texto escrito, las expresiones faciales, gestos, miradas, indicaciones, etc. *(Trung, 2006)*. En un futuro, se tratará de proporcionar otros modos de interacción como gestos o indicaciones para poder controlar dichos dispositivos.

- *Detección y resolución de ambigüedades.*

Actualmente los dispositivos se autogestionan de manera distribuida por lo que es complicado que exista una ambigüedad en una orden o consulta de un dispositivo. Sin embargo, cuando se crean grupos de dispositivos que pueden tener acciones parecidas y puedan comportarse distintamente con la misma orden es necesario desarrollar métodos para resolver las posibles ambigüedades. Por lo tanto, un dispositivo no deberá contener solamente información sobre su propio funcionamiento y sus funciones, sino que también podría ser interesante que este dispositivo conociese también las funcionalidades y posibilidades de otros dispositivos para determinar si la orden o la consulta que se ha ejecutado es más cercana a él o a otro de sus dispositivos. Para ello será necesario mantener distintas bases de conocimiento en los dispositivos y tal vez pueda ser interesante que los mismos dispositivos intercambien estas bases de conocimiento entre ellos.

- *Desarrollo de un catálogo de bases de conocimientos para dispositivos.*

Debido a que la construcción de las bases de conocimiento es una de las tareas más complicadas, sería adecuado proporcionar a los usuarios un catálogo donde se incluyese el modelado de las bases de conocimiento de distintos dispositivos en distintos idiomas para que se pudieran reutilizar. Así, se ha planeado que se van a construir distintas bases de conocimiento para todos los electrodomésticos comunes que puedan incluir el hardware desarrollado para esta tesis doctoral que incluyen: hornos, lavadoras, microondas, frigoríficos, tostadora, etc.

- *Desarrollo de un editor con asistentes para la configuración y creación de bases de conocimiento.*

Como se ha comentado en el punto anterior, la creación de ontologías y bases de conocimiento es una tarea muy complicada. Es por esta razón por la que un

usuario para poder crear una base de conocimiento o modificar una existente necesita conocimientos de modelado conceptual y ontologías. Como estos conocimientos no son muy sencillos de aprender, se ha planteado la creación de editores sencillos para poder editar las funcionalidades, estados y alarmas de los dispositivos para que sean fácilmente configurables y que creen de manera transparente a los usuarios la base de conocimiento necesaria para el funcionamiento del sistema de diálogo dentro del dispositivo.

En resumen, el sistema de diálogo y control de dispositivos de la IoT mediante lenguaje natural escrito presentado en esta tesis doctoral se puede situar dentro de las áreas de Internet de las Cosas, Procesamiento del Lenguaje Natural y Sistemas Basados en Conocimiento. En primer lugar, es relevante para el área de Internet de las Cosas porque proporciona nuevas tecnologías y protocolas para la interacción entre los dispositivos y los humanos En segundo lugar es relevante para el área de Procesamiento del Lenguaje Natural porque proporciona una tecnología sencilla para la interpretación de órdenes y comandos en lenguaje natural que es eficiente y puede aplicarse en el control de dispositivos. Además, se ha presentado un sistema de gestión de diálogo en lenguaje natural mediante tecnologías de mensajería instantánea. Finalmente, esta metodología presentada en esta tesis contribuye a mejorar el desarrollo de sistemas de basados en conocimiento ya que permite el modelado de dispositivos y sistemas de diálogo de manera sencilla mediante ontologías.

Capítulo VI. – Contribuciones científicas.

El desarrollo de esta tesis doctoral ha dado lugar a las siguientes contribuciones científicas que han sido aceptadas y publicadas antes de la presentación de la presente tesis doctoral.

- Noguera-Arnaldos, J. Á., Paredes-Valverde, M. A., Valencia-García, R., & Rodríguez-García, M. Á. (2015). Sistema de diálogo basado en mensajería instantánea para el control de dispositivos en el internet de las cosas. Procesamiento del Lenguaje Natural, 55, 173-176.
- Paredes-Valverde, M. A., Noguera-Arnaldos, J. Á., Rodríguez-Enríquez, C. A., Valencia-García, R., & Alor-Hernández, G. (2015). A Natural Language Interface to Ontology-Based Knowledge Bases. In Distributed Computing and Artificial Intelligence, 12th International Conference (pp. 3-10). Springer International Publishing.
- Noguera-Arnaldos J.A, Rodríguez-García M.A., Ochoa J.L., Paredes-Valverde M.A., Alcaraz-Mármol G., Valencia-García R. (2015) Ontology-Driven Instant Messaging-Based Dialogue System for Device Control. Lecture Notes in Computer Science 9416: 299-308

Además, se va a enviar una publicación a un número especial de la revista Mobile Information Systems dedicada a sistemas móviles de interacción hombre-máquina basados en el uso de tecnologías de procesamiento del lenguaje natural.

El proyecto im4Things presentado en esta tesis doctoral ha tenido la suficiente entidad para ser patentada. Tras la preparación de toda la documentación necesaria se procedió a su registro en agosto de 2015 con número de solicitud P201531171.

Bibliografía

Arshdeep Bahga, Vijay Madisetti (2014). *"Internet of Things: A Hands-On Approach Hardcover".*pp. 19-50, 65-93.

Bernard M.E. Moret, H.D. Shapiro (1991). *"Algorithms From P to NP: Design and Efficiency"*, pp. 152-180.

Gabillaud Jerome (2010). *"SQL Server 2010 Administración de una base de datos con SQL Server Management",* Edic. ENI.

Armstrong R. Virding y Wikstrom M. W (1996). *"Concurrent Programming in Erlang",* Edic. Prentice Hall.

Hopkins Bruce y Antony Ranjith (2013). *"Bluetooth for Java",* Edic Apress.

Huidobro Moya José M. y Roldan Martínez (2005). *"Comunicaciones en redes Wifi, VoIP, Multimedia seguridad, WLan".* Edit. Creaciones copyright. pp. 84-135.

Amaro Soriano y José Enrique (2012). *"Programación Avanzada con Android"*. Edic. Marcombo.

Conway Joe y Hillegass Aarone (2011). *"Programación iOS"*. Edit. Anaya.

Lutz Mark (2001). "*Programming Python Solutions for Python Programmers*". Edit. O´reilly and Associates.

Benítez Raúl, Escudero Gerad, Kanaan Samir y Masip Rodó David (2011). *"Inteligencia Artificial Avanzada",* Edit. Adobe DRM.

Vázquez Daniel, Junestrand Stefan y Passaret Xabier (2014). *"Domotica y hogar digital"*. Edit. Paraninfo.

Maestre Torreblanca Jose Mª. (2015). *"Domótica para Ingenieros".* Edit. Paraninfo.

Salvatore Gaglio y Giuseppe Lo Re (2014). *"Advances onto the Internet of Things. How Ontologies Make the Internet of Things Meaningful".* Edit. Springer. Pp. 2194-5357.

Biswas G, Roscoe R, Jeong H y Sulcer B (2009). *"Promoting self-regulated learning skills in agent- based learning environments".* Proceedings of the 17th international conference on, computers in education. Hong Kong: Asia-Pacific Society for Computers in Education.

Pert Hájek (2001). *"Mathematics of Fuzzy Logic"*. Edit. Kluwer Academic Publishers. ISSN 1-4020-0370-6. pp. 261-271.

Atzori, L., Iera, A., y Morabito, G. (2010). "The internet of things: A survey. Computer networks". Pp. 2787-2805.

Xia, F., Yang, L. T., Wang, L., y Vinel, A. (2012): "Internet of things. International Journal of Communication Systems", pp. 1101.

Troyano, Albert Batiste (2011). "Protocolos de encaminamiento en redes inalámbricas Mesh: un estudio teórico y experimental". Pp. 79-104.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web.Scientific american, 284(5), 28-37.

Guido R. Hiertz, Dee Denteneer, Sebastian Max, Rakesh Taori, Javier Cardona, Lars Berlemann y Bernhard Walke (2010): "IEEE 802.11s: the Wlan mesh standard". IEEE. 1536-1284

Harper, R. (Ed.). (2006): "Inside the smart home". Springer Science & Business Media.

Ibrahim Dogan & Tessel Renzenbrink (2012) "The Internet of Things (IoT). An Introduction with PIC Microcontrollers". Elector. 17-45.

Jose Ignacio Moreno, Ignacio Soto y David Larrabeiti (2001). "Protocolos de Señalización para el transporte de Voz sobre redes IP". Departamento de Ingeniería Telemática Universidad Carlos III de Madrid.

Jokinen, K., & McTear, M. (2009). Spoken dialogue systems. Synthesis Lectures on Human Language Technologies". Pp. 1-151.

López-Cózar, R., Callejas, Z., y Gea, M. (2005). "Análisis de metodologías de evaluación de sistemas de diálogo multimodal."Procesamiento del lenguaje natural". Pp. 34.

Lison, P., y Meena, R. (2014). "Spoken dialogue systems: the new frontier in human-computer interaction". The ACM Magazine for Students". Pp. 46-51

Aust, H. y Oerder, M. (1994). "Database query generation from spoken sentences". Proc. of 2nd Workshop on Interactive Voice Technology for Telecommunications Applications (IVTTA). Kyoto, Japan. pp. 141–144.

Bonafonte, A., Aibar, P., Castell, N., Lleida, E., Mariño, J., Sanchis, E. y Torres, M., (2000). "Desarrollo de un sistema de diálogo oral en dominios restringidos". Actas de las I Jornadas en Tecnología del Habla. Sevilla, España.

Zue, V., Seneff, S., Glass, J., Polifroni, J., Pao, C., Hazen, T. y Hetherington, L., (2000). "JUPITER: A telephone-based conversational interface for weather information". IEEE Transactions on Speech and Audio Processing. vol. 8 (1), pp. 85–96.

Turunen, M., Sonntag, D., Engelbrecht, K. P., Olsson, T., Schnelle-Walka, D., y Lucero, A. (2015). "Interaction and Humans in Internet of Things. In Human-Computer Interaction–INTERACT". Springer International Publishing. pp. 633-636.

Vivancos-Vicente, P., Castejón-Garrido, J.S. y García-Gil, D. (2009). "INVOX – Control domótico por voz mediante lenguaje natural". DRT4all, pp 11-17

Vanus, J., Smolon, M., Koziorek, J., y Martinek, R. (2015). "Voice Control of Technical Functions in Smart Home with KNX Technology" Computer Science and its Applications. Springer Berlin Heidelberg. pp. 455-462).

Katsamanis, A., Rodomagoulakis, I., Potamianos, G., Maragos, P., y Tsiami, A. (2014). "Robust far-field spoken command recognition for home automation combining adaptation and multichannel processing". Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on. pp. 5547-5551.

Espejo, G., Ábalos, N., López-Cózar, R., Callejas, Z., y Griol, D. (2010). "Sistema Mayordomo: Uso de un Entorno de Inteligencia Ambiental a Través de un Sistema de Diálogo Multimodal". Procesamiento del lenguaje natural. pp. 309-310.

Kopp, S., Jung, B., Leßmann, N., y Wachsmuth, I. (2003). "Max - A Multimodal Assistant in Virtual Reality Construction". Pp. 11.

Aguado, L. R., Serrano, A. M. G., Martínez Fernández, P. M., y de Datos, G. D. B. (2002). "Planteamiento semántico y pragmático para gestión de diálogos en asistentes virtuales". Procesamiento del lenguaje natural. Pp. 28.

Amy S. Wu, Rob Farrell y Mark K. Singley (2002). "Scaffolding group Learning in a Collaborative Networked Environment". Computer Support for Collaborative Learning: Foundations for a CSCL Community. Pp. 245.

Brian J. Reiser (2002). "Why scaffolding should sometimes make tasks more difficult for Learners. Computer Support for Collaborative Learning". Foundations for a CSCL Community. Pp. 255.

Tetsuo Shinozaki, Yukiko Yamamoto y Setsuo Tsuruta (2013). "Context-based counselor agent for software development ecosystem". Edit. Springer. *Volume 97, Issue 1, pp 3-28*

AREA, M. y ADELL, J. (2009). "e-Learning: Enseñar y aprender en espacios virtuales". En J. De Pablos (Coord): Tecnología Educativa. Aljibe, Málaga. Pp. 391-424.

Ioannis Doumanis y Serengul, (2013). "Beyond Artificial Intelligence Markup Language (AIML) Rapid prototyping of a Q&A system". Smith de la Universidad de Middlesex University. Conference: Proceedings of the 9th International Conference on Intelligent Environments 2013, At Athens/Greece.

J. S. R. Jang, C. T. Sun, y E. Mizutani (1997). "Neuro-Fuzzy and Soft Computing: a Computational Approach to Learning and Machine Intelligence". (Englewood Cliffs, NJ: Prentice-Hall). Reviewed by Yu-Chi Ho.

Stefan Junestrand, Xavier Passaret y Daniel Vázquez, (2004). "Domótica y Hogar Digital". Thomson Paraninfo, pp. 65.

Sáez, D. (2004). "Domótica". Actualidad TIC - Revista del Instituto Tecnológico de Informática. Pp 4-5.

Noguera-Arnaldos, J. Á., Paredes-Valverde, M. A., Valencia-García, R., & Rodríguez-García, M. Á. (2015). "Sistema de diálogo basado en mensajería instantánea para el control de dispositivos en el internet de las cosas" Procesamiento del Lenguaje Natural, pp. 173-176.

Portet, F., Vacher, M., Golanski, C., Roux, C., & Meillon, B. (2013). "Design and evaluation of a smart home voice interface for the elderly: acceptability and objection aspects". Personal and Ubiquitous Computing. Pp. 127-144.

Soda, S., Nakamura, M., Matsumoto, S., Izumi, S., Kawaguchi, H., & Yoshimoto, M. (2012, December). "Implementing virtual agent as an interface for smart home voice control". In Software Engineering Conference (APSEC), 2012 19th Asia-Pacific (Vol. 1, pp. 342-345). IEEE.

Kumar, S. (2014). "Ubiquitous smart home system using android application". arXiv preprint arXiv: 1402- 2114.

Santos-Pérez, M., González-Parada, E., & Cano-García, J. M. (2011). AVATAR: An Open Source Architecture for Embodied Conversational Agents in Smart Environments. In Ambient Assisted Living (pp. 109-115). Springer Berlin Heidelberg.

Studer, R., Benjamins, V. R., & Fensel, D. (1998). "Knowledge engineering: principles and methods". Data & knowledge engineering. Pp. 161-197.

Schreiber, G. (2000). "Knowledge engineering and management: the CommonKADS methodology". MIT press.

Angele, J., Fensel, D., Landes, D., & Studer, R. (1998). "Developing knowledge-based systems with MIKE". Automated Software Engineering, pp. 389-418.

Eriksson, H., Shahar, Y., Tu, S. W., Puerta, A. R., & Musen, M. A. (1995). Task modeling with reusable problem-solving methods. Artificial Intelligence,79(2), 293-326.

Sowa, J. F. (1999). Knowledge representation: logical, philosophical, and computational foundations.

Studer, R., Benjamins, V. R., & Fensel, D. (1998). "Knowledge engineering: principles and methods". Data & knowledge engineering. Pp. 161-197.

Ruiz-Martınez, J. M., Minarro-Giménez, J. A., Castellanos-Nieves, D., Garcıa-Sánchez, F., & Valencia-Garcia, R. (2011). "Ontology population: an application for the E-tourism domain". International Journal of Innovative Computing, Information and Control (IJICIC), 6115-6134.

Fernández-Breis, J. T., Castellanos-Nieves, D., & Valencia-García, R. (2009). Measuring individual learning performance in group work from a knowledge integration perspective. Information Sciences, pp. 339-354.

García-Sánchez, F., Fernández-Breis, J. T., Valencia-García, R., Gómez, J. M., & Martínez-Béjar, R. (2008). "Combining Semantic Web technologies with Multi-Agent Systems for integrated access to biological resources". Journal of biomedical informatics, pp. 848-859.

Esteban-Gil, A., Garcia-Sanchez, F., Valencia-Garcia, R., & Fernandez-Breis, J. T. (2012). "SocialBROKER: A collaborative social space for gathering semantically-enhanced financial information". Expert Systems with Applications, pp. 9715-9722.

Lupiani-Ruiz, E., GarcíA-Manotas, I., Valencia-GarcíA, R., GarcíA-SáNchez, F., Castellanos-Nieves, D., FernáNdez-Breis, J. T., & CamóN-Herrero, J. B. (2011). Financial news semantic search engine. Expert systems with applications, pp. 15565-15572.

García-Sánchez, F., Valencia-García, R., Martínez-Béjar, R., & Fernández-Breis, J. T. (2009). An ontology, intelligent agent-based framework for the provision of semantic web services. Expert Systems with Applications, 36(2), 3167-3187.

Valencia-García, R., Fernández-Breis, J. T., Ruiz-Martínez, J. M., García-Sánchez, F., & Martínez-Béjar, R. (2008). A knowledge acquisition methodology to ontology construction for information retrieval from medical documents. Expert Systems. Pp. 314-334.

Carrer-Neto, W., Hernández-Alcaraz, M. L., Valencia-García, R., & García-Sánchez, F. (2012). Social knowledge-based recommender system. Application to the movies domain. Expert Systems with Applications, 39(12), 10990-11000.

Colombo-Mendoza, L. O., Valencia-García, R., Rodríguez-González, A., Alor-Hernández, G., & Samper-Zapater, J. J. (2015). RecomMetz: A context-aware knowledge-based mobile recommender system for movie showtimes. Expert Systems with Applications, 42(3), 1202-1222.

Lenat, D. B. (1995). CYC: A large-scale investment in knowledge infrastructure.Communications of the ACM. Pp. 33-38.

Contreras, J., & Martínez, C. A. Tutorial Ontologías. Universidad Complutense de Madrid. Accesible en http://www.sedic.es/gt_normalizacion_tutorial_ontologias.pdf (último acceso 7/10/2015)

Miller, G. A. (1995). "WordNet: a lexical database for English". Communications of the ACM. Pp. 39-41.

Paolo Baronti, Prashant Pillai, Vince W.C. Chook, Stefano Chessa, Alberto Gotta, Y. Fun Hu, (2007). "Wireless sensor networks: A survey on the state of the art and the 802.15.4 and ZigBee standards". Computer Communications 30 (2007) 1660

Salton, G., Fox, E. A., & Wu, H. (1983). Extended Boolean information retrieval. Communications of the ACM, 26(11), 1022-1036.

Yang, Y., & Liu, X. (1999, August). A re-examination of text categorization methods. In Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval (pp. 42-49). ACM.

Olson, D. L., & Delen, D. (2008). Advanced data mining techniques. Springer Science & Business Media.

Jordan, A., & Araki, K. (2014, October). A framework for multilingual real-time spoken dialogue agents. In Awareness Science and Technology (iCAST), 2014 IEEE 6th International Conference on (pp. 1-7). IEEE.

Mejía, M. E., Montiel-Ponsoda, E., de Cea, G. A., & Gómez-Pérez, A. (2012). Ontology localization. In Ontology Engineering in a Networked World (pp. 171-191). Springer Berlin Heidelberg.

Trung, H. (2006). Multimodal dialogue management-state of the art. Human Media Interaction Department, University of Twente, 2.

ANEXO I.- Test Sociolingüístico

Imagina que pudieras controlar mediante un chat algunos de los aparatos electrónicos o dispositivos mecánicos que hay en tu casa, como por ejemplo la cafetera, sistema de riego, control de iluminación interior, control de persianas y calefactor de aire.

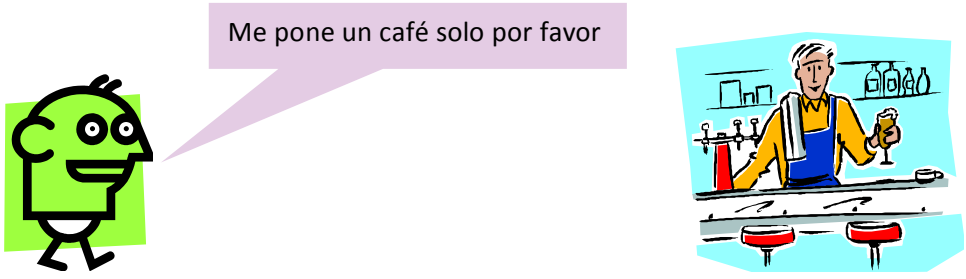
Con sólo unas palabras sería posible que estos aparatos hicieran lo que tú les estás pidiendo, como si realmente te entendieran.

El objetivo de este test sociolingüístico es determinar cuáles son las expresiones que te gustaría utilizar para "comunicarte" con los dispositivos de tu casa, de manera que te resulte lo más cómodo posible.

Para ello te proponemos una serie de acciones que se han automatizado y que es posible controlar mediante un comando en lenguaje natural.

En cada imagen se presenta una situación concreta, con un breve texto que la describe, junto con un personaje, al que llamaremos Pepe, que es el que debe ordenar o pedir que se realice la acción. Si tú fueras esa persona ¿qué te gustaría decir en cada situación?

Por ejemplo, Pepe se acaba de levantarse y se dirige al bar de la esquina para tomar un café, una vez en el bar el camarero llega y Pepe hace su petición. Existen varias opciones a la hora de pedir un café.



Pero Pepe también podría haber dicho:

Un café por favor

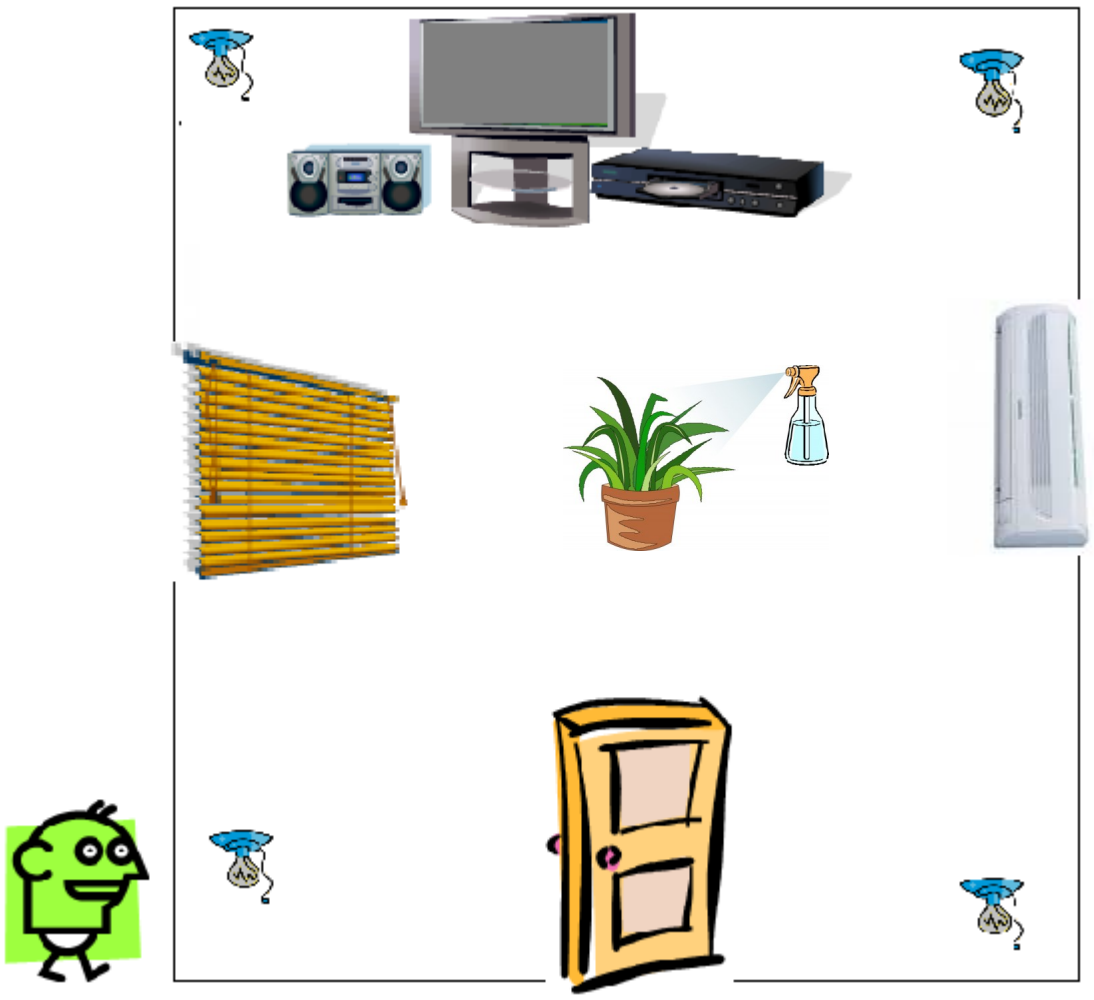
Un solo

Me gustaría tomar un café

Quiero un café

…

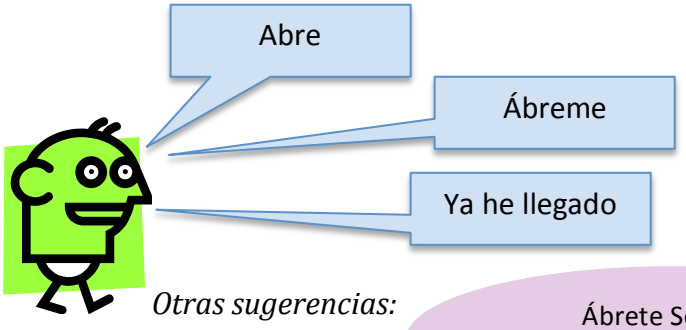
A continuación se muestran otros elementos de una casa los cuales podemos controlar por medio de un chat en lenguaje natural. Entre ellos podemos encontrar un sistema de iluminación, un calefactor de aire, un sistema de riego y un controlador de persianas, elementos de la casa que funcionan de manera automática, y a los que Pepe les puede ordenar o pedir lo que quiere que hagan.



Ahora se plantearán una serie de situaciones relacionadas con ellos en donde Pepe les puede hablar para conseguir lo que desea.

Las expresiones que se podrían utilizar para que estos elementos funcionaran son muchas, así que puedes escribir todas las que consideres.

1. Pepe ha vuelto de tomar el café y acaba de llegar a casa, pero todavía está en la calle, ¿Qué le podría decir Pepe a  para que funcionara?



Otras sugerencias:

Ábrete Sésamo
Vilma ya he llegado

2. Pepe ya está en su salón, y quiere que entre la luz del sol, qué podría decirle a  para tener un poco más de luz?

Persiana súbete, arriba, sube un poco, quiero luz

Otras sugerencias.

Enróllate, ábrete

3. Pepe está un poco aburrido, así que decide utilizar  ¿Qué podría decir Pepe para que funcionara?

Enciéndete, ponte en marcha

Otras sugerencias: Mi canal favorito, youtube, internet

4.  ya funciona, pero aún no ha empezado su programa favorito así que decide hacer un poco de zapping, el mando a distancia no es necesario, porque puede utilizar la voz. ¿Qué diría Pepe en este caso?

Pasar canales, mis canales favoritos, documentales

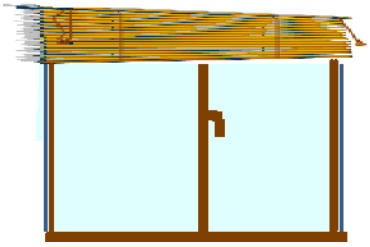
Otras sugerencias: No quiero ver anuncios, canal deportes

5. Pepe ha estado bailando un rato y ya se ha cansado, así que quiere poner en marcha

¿Qué palabras podría utilizar Pepe para que este aparato funcionara?

Enciéndete, temperatura habitual, tengo calor

Otras sugerencias: Ponte a 25 grados, apágate

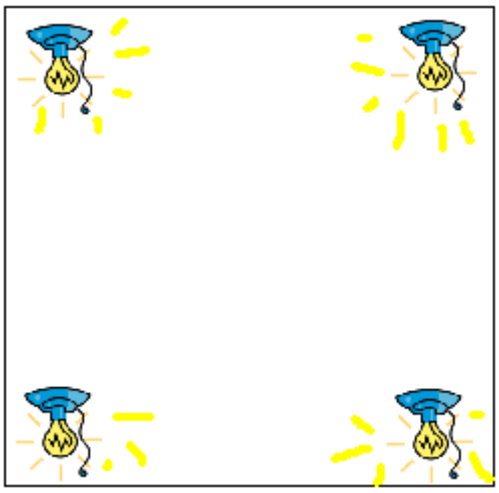
6. Es la hora de la siesta, y Pepe está acostado en el sofá, pero le molesta la luz que entra del exterior, porque  entonces, le gustaría que hubiera un poco menos de luz (pero no quedarse a oscuras). ¿Qué podría decir Pepe para conseguir  ?

Persiana bájate, abajo, baja un poco, no quiero luz

Otras sugerencias: Ciérrate, desenróllate, veta abajo

7. Ha anochecido en casa de Pepe, así que el salón se ha quedado a oscuras ¿Qué podría decir Pepe para pasar de esta situación

a esta otra situación



?

Enciéndete

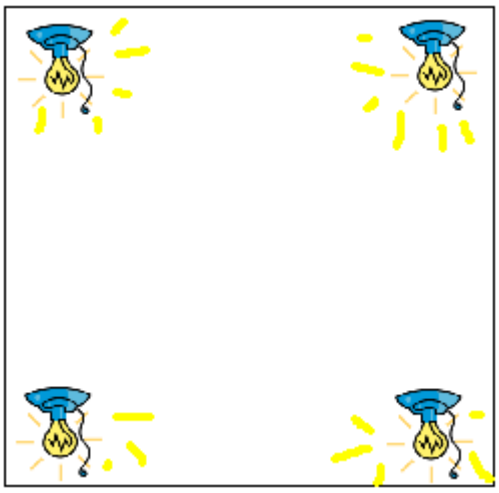
Otras sugerencias: Luz, quiero luz

8. El nivel de luminosidad de  se puede regular, entonces si por ejemplo quisiera menos luminosidad qué diría.

Baja luz, reduce luz

Otras sugerencias: Escena 1, vete a la mitad

9. Finalmente Pepe decide salir un rato, recuerda que Pepe ha puesto en funcionamiento varios dispositivos, que ahora deberían dejar de funcionar. Estos dispositivos son:



Pepe puede elegir entre que deje todo de funcionar al mismo tiempo o bien ir ordenándoselo dispositivo a dispositivo.

Apagar todo, descansar todo, apagar luz y aire, dejar de regar

Otras sugerencias:

Bajar persianas y apagar aire, todo a dormir

2º PARTE

Aunque los dispositivos de la casa de Pepe son bastante listos, puede ocurrir que no funcionen correctamente cuando se les pide algo.

Esto puede pasar porque el dispositivo no ha escuchado lo que Pepe ha dicho (por ejemplo Pepe dice "Encender calefactor" pero el sistema escucha "Encender luz") o bien porque, aunque lo haya entendido, esa orden se puede aplicar a varios dispositivos y el sistema no sabe a cuál de ellos se está refiriendo. Por ejemplo cuando decimos "Apagar" esto se puede aplicar a más de uno de los elementos que se han mostrado: la cafetera, el aire acondicionado, las luces, etc.

Entonces, si tú fueras Pepe ¿qué te gustaría que pasara si el dispositivo que quieres que funcionara no lo hiciera?

Se plantean a continuación dos situaciones con varias opciones sobre lo que te gustaría que pasara. Señala una o varias de estas opciones, y si no encuentras la que se adecúa a tus preferencias la puedes añadir.

SITUACIÓN 1

1. Pepe ha dicho "Subir la persiana" pero el sistema ha entendido "encender luz" ¿Qué te gustaría que pasara?

a) Que no pase nada, es decir, que el sistema no haga absolutamente nada. Ya repetiré la orden cuando me dé cuenta.
b) Que el sistema me avise de que no me ha entendido diciéndome "No he entendido ¿Puedes repetir?"
c) Que el sistema me pregunte utilizando los términos que sí ha entendido, en este caso diría "¿Qué acción quieres ejercer sobre el dispositivo persiana?

d) Que el sistema me pregunte lo que sería más razonable que quisiera hacer. Por ejemplo si en este caso la persiana estuviera bajada, el sistema podría preguntarme "¿Quieres subir la persiana?
e) Que el sistema haga lo que considere más razonable sin preguntarme.

De los 50 probadores estos fueros los resultados a la situación 1:

Opción	Resultados	Media
a	2	4,00%
b	38	76,00%
c	6	12,00%
d	4	8,00%
e	0	0,00%
Total=	50	100,00%

Tabla 15. Datos de la encuesta realizada a los 50 usuarios sobre la situación 1

La tabla 15 presenta los datos de los usuarios respecto a las distintas opciones de la situación 1. Como se puede observar el 76% de las personas consultadas desea que el sistema le avise ante la situación: "Que el sistema me avise de que no me ha entendido diciéndome "No he entendido ¿Puedes repetir?". Eso indica que desean estar seguros de que el sistema no haga algo distinto a la orden enviada lo que podemos traducir que el sistema tiene que ser fiable y seguro. La fiabilidad del sistema es un aspecto global. Por un lado está el aspecto de cómo de fiable sea el sistema im4Things en sí que como vemos la precisión media es del 92,95% y por otro lado está cómo de fiable es el conjunto de sistemas ajenos al nuestro, refiriéndonos a los distintos nodos por donde discurre la información. En el futuro las siguientes líneas de investigación a partir de esta tesis doctoral deben mejorar la tecnología para garantizar que esta tasa pase del 92,95% al 100%. De esta forma los usuarios percibirán que este sistema es además de útil necesario para mejorar la vida de las personal y por consiguiente las empresas se verán motivadas para comercializar con estos sistemas.

SITUACIÓN 2

Pepe dice "Apagar", pero en su salón hay varios dispositivos a los que se les podría aplicar este comando, ¿Qué te gustaría que pasara en este caso?

a) Que el sistema no apague nada, ni me pregunte nada. Cuando me dé cuenta ya seré más específico.
b) Que el sistema me pregunte "¿Qué quieres apagar?"
c) Que el sistema vaya preguntándome uno a uno por los dispositivos que en ese momento están funcionando y sobre los que se puede aplicar el comando apagar. Por ejemplo si están funcionando el calefactor de aire, la persiana y una luz, el sistema podría preguntarme "¿Quieres apagar el calefactor de aire? ¿Quieres regular la persiana? ¿Quieres apagar la luz?", y apague sólo aquello para lo que obtenga una respuesta afirmativa.
d) Que el sistema apague todo lo que esté funcionando sin preguntar.

Opción	Resultados	Media
a	3	6,00%
b	21	42,00%
c	20	40,00%
d	6	12,00%
Total=	50	100,00%

Tabla 16. Datos de la encuesta realizada a los 50 usuarios sobre la situación 2

En la tabla 16 presenta los datos de los usuarios respecto a las distintas opciones de la situación 2. Aquí se produce una dualidad de opciones entre la b) con el 42% y la c) con el 40%. De nuevo los datos revelan que las personas consultadas desea que el sistema le avise ante la situación presentada. Por lo tanto los trabajos futuros deben estar enfocados en mejorar la **fiabilidad y seguridad** del sistema. Esto lo podemos conseguir realizando más sistemas como el presentado para cubrir el resto de necesidades de los usuarios.

ANEXO II.- Encuesta de satisfacción

En este anexo se exponen los resultados obtenidos de la encuesta de satisfacción realizadas a los 50 usuarios betatester realizadas en dos tablas. En la primera tabla (ver tabla 17) se preguntan por el grado de utilidad, facilidad de uso, si recomendaría el sistema im4Things a otros usuarios, grado interés y finalmente la satisfacción media expresada por el usuario. La ponderación fue del 0 al 5 siendo el 0 la puntuación menor valor y el 5 la máxima puntuación.

En la segunda tabla los usuarios son preguntados por el tipo de electrodomésticos que referirían contaran con el sistema im4Things.

Usuario	Utilidad	Facilidad de uso	Recomendaría im4Things	Interés	Satisfacción General
U1	5	2	5	4	4
U2	4	5	4	4	4,25
U3	5	4	4	5	4,5
U4	4	3	3	3	3,25
U5	5	5	3	1	3,5
U6	2	3	2	2	2,25
U7	2	3	2	4	2,75
U8	1	3	4	1	2,25
U9	3	4	3	3	3,25
U10	3	2	3	5	3,25
U11	2	1	4	3	2,5
U12	2	5	2	1	2,5
U13	2	4	4	5	3,75
U14	2	5	3	3	3,25
U15	0	4	3	4	2,75
U16	3	2	2	5	3
U17	0	4	5	1	2,5
U18	2	2	2	2	2
U19	2	2	2	4	2,5
U20	3	3	2	1	2,25
U21	0	1	2	2	1,25
U22	5	4	2	4	3,75
U23	2	2	5	5	3,5
U24	4	5	5	4	4,5
U25	2	5	3	2	3
U26	5	3	4	5	4,25
U27	3	4	3	3	3,25
U28	2	5	3	4	3,5
U29	2	3	4	4	3,25
U30	5	4	3	5	4,25
U31	0	4	2	5	2,75
U32	3	3	3	3	3
U33	5	4	3	5	4,25
U34	1	1	3	3	2
U35	1	4	1	4	2,5
U36	4	2	5	3	3,5
U37	2	4	4	5	3,75
U38	2	2	5	2	2,75
U39	4	2	3	4	3,25
U40	1	3	5	2	2,75
U41	3	2	2	3	2,5
U42	1	3	3	4	2,75
U43	2	5	4	3	3,5
U44	0	4	5	3	3
U45	1	3	5	4	3,25
U46	1	3	5	4	3,25
U47	3	3	3	4	3,25
U48	4	2	2	2	2,5
U49	2	5	2	0	2,25
U50	2	4	4	4	3,5
TOTAL=	124	165	165	166	3,1

Tabla 17. Votación de los usuarios obtenidos de la encuesta de satisfacción

Los resultados fueron los siguientes: el grado de satisfacción general fue del 3,1 (respecto a 5) lo que equivaldría a una satisfacción alta. Esto nos anima a pensar que en caso de comercializarse el producto podría tener buena aceptación por parte de los usuarios. No obstante la población elegida es de 50 usuarios lo que será necesario realizar más investigación de mercado en el futuro.

En la tabla 18 se presentan los resultados obtenidos a la consulta sobre funcionalidad y recomendaciones de uso de la aplicación resultando:

Total de puntuación posible=250

Utilidad	Facilidad de uso	Recomendaría im4Things	Interés
124	165	165	166
49,60%	66,00%	66,00%	66,40%

Tabla 18. Ratios en porcentajes de los preguntas orientadas a aspectos funcionales

Los usuarios indican que el grado de utilidad de la aplicación es cercana al 50% obteniendo un valor de 49,60% votada sobre los sistemas probados que recordamos son **cafetera, sistema de riego, control de iluminación interior, control de persianas y calefactor de aire**. Esto nos lleva a pensar que los usuarios perciben esta aplicación con un grado medio de utilidad. Este ratio podrá incrementarse aportando más soluciones así como enriquecer la base de conocimiento con nuevas funcionalidades. Respecto a la facilidad de uso los usuarios votaron con un 66% su percepción lo que podríamos decir que esta cota alcanzada es satisfactoria e indicadora que la solución adoptada está alineada con los requerimientos de los usuarios. Los usuario también votaron con el 66% recomendar esta solución a otro usuarios. Este punto es un buen indicador ya que este tipo de redes sociales su promoción por medio del bis a bis es muy efectivo. Finalmente el grado de interés global de los usuarios es del 66,40%. Este dato indica que la solución planteada, aún cuando se realizó con el objetivo de investigar y obtener nuevos avances técnico-científicos en mejorar los sistemas de mensajería instantánea y ontologías en IoT, está alineada con las necesidades del usuario.

Para la segunda encuesta, se pidió a los usuarios que respondiesen a la pregunta ¿qué tipo de sistemas ve nuestro sistema de utilidad? Para facilitar la respuesta se agrupó electrodomésticos y sistemas por grupos quedando:

Grupo 1: cafetera, horno, lavadora y secadora

Grupo 2: Alumbrado

Grupo 3: Sistemas de seguridad

Grupo 4: Discapacidad/dependencia

Los resultado fueron los siguientes (ver tabla 19 y 20):

Usuario	Grupo 1	Grupo 2	Grupo 3	Grupo 4
U1	x	x	x	x
U2	x	x		x
U3	x			x
U4	x	x		x
U5	x			x
U6	x	x	x	x
U7	x			x
U8	x			x
U9	x	x		x
U10	x	x	x	x
U11	x	x		x
U12	x			x
U13	x		x	x
U14	x		x	x
U15	x	x		x
U16	x			x
U17	x	x		x
U18	x			x
U19	x		x	x
U20	x	x	x	x
U21	x			x
U22	x			x
U23	x	x	x	x
U24	x			x
U25	x	x		x
U26	x	x		x
U27	x		x	x
U28	x			x
U29	x	x		x
U30	x			x
U31	x	x		x
U32	x			x
U33	x	x	x	x
U34	x	x	x	x
U35	x			x
U36	x		x	x
U37	x	x		x
U38	x	x		x
U39	x	x		x
U40	x			x
U41	x	x		x
U42	x			x
U43	x	x	x	x
U44	x	x		x
U45	x			x
U46	x			x
U47	x		x	x
U48	x	x		x
U49	x			x
U50	x	x	x	x
TOTAL=	50	25	15	50

Tabla 19. Resultado a la pregunta *¿qué tipo de sistemas ve nuestro sistema de utilidad?*

Total de puntuación posible=50

Grupo 1	Grupo 2	Grupo 3	Grupo 4
50	25	15	50
100,00%	50,00%	30,00%	100,00%

Tabla 20. Ratios resultantes a la pregunta *¿qué tipo de sistemas ve nuestro sistema de utilidad?*

Al realizar esta encuetas intuíamos que la respuesta de los usuarios iba a ser dispar en lo que a aspectos de seguridad y funcionalidad se refiere. Los usuario ven claramente que esta aplicación encaja con el 100% de confianza en los grupos 1 y 4 (cafetera, horno, lavadora, secadora y aplicaciones de discapacidad/dependencia respectivamente) mientras que el grupo 2 con el 50% y el grupo 3 con el 30% son los que menos seguridad ofrece a los usuarios. Respecto a los resultados del grupo 3, seguridad, entendemos que el sistema im4Things debe demostrar con el tiempo la fiabilidad y robustez ante fallos y pirateo para que los usuarios confíen en el sistema.