Escuela Politécnica Superior de Jaén

Grado en Ingeniería Informática

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Jaén

Trabajo Fin de Grado

SISTEMA PARA LA MONITORIZACIÓN DE AMBIENTES INTELIGENTES A TRAVÉS DE LA GESTIÓN Y ACCESO A SERVICIOS

Alumno: **Daniel Zafra Romero**

Tutor: Dra. Dª. Macarena Espinilla Estévez
Dr. D. Javier Medina Quero
Dpto: Informática

Febrero, 2016

Sistema para la monitorización de ambientes inteligentes a través de la gestión y acceso a servicios

Daniel Zafra Romero

Febrero 2016

Universidad de Jaén

Escuela Politécnica Superior de Jaén

Departamento de Informática

Dra. D^a. Macarena Espinilla Estévez, tutora,

y *Dr. D. Javier Medina Quero*, cotutor,

del Trabajo Fin de Grado titulado:

Sistema para la monitorización de ambientes inteligentes a través de la gestión y acceso a servicios,

que presenta *D. Daniel Zafra Romero*,

autorizan su presentación para defensa y evaluación en la

Escuela Politécnica Superior de Jaén.

Jaén, Febrero de 2016

El alumno:

Daniel Zafra Romero

Los tutores:

Macarena Espinilla Estévez Javier Medina Quero

Este trabajo no hubiera sido posible sin el apoyo de todas aquellas personas que han aportado su experiencia, cariño y apoyo en la que ha sido una etapa de enorme crecimiento personal.

En primer lugar, a mis tutores Javier Medina Quero y en especial a Macarena Espinilla Estévez, por enseñarme los beneficios de la organización y de la motivación ante aquello que nos supone un reto.

A mi familia y a mi pareja Carolina, por su apoyo incondicional, sin el cual nunca hubiera aprendido a creer en mí mismo y por su infinita paciencia en todos aquellos momentos en los que me ayudaron a levantarme. Gracias por permitirme devolveros una parte de todo lo que me habéis dado. Y por último, no quisiera olvidarme de dar las gracias a todos mis compañeros, mis amigos, por todos aquellos momentos, dentro y fuera del aula. Gracias a todos.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Motivación

En los últimos años, el avance de la tecnologías de la información y comunicación junto con la inteligencia artificial han constituido una gran parte de los estudios e investigaciones más destacados [1]. Dicho avance ha estado acentuado por el creciente desarrollo de dispositivos electrónicos con un bajo consumo de energía y un tamaño reducido, así como su bajo coste. Dichos dispositivos se están integrando en nuestra vida cotidiana, implicando su utilización, casi imprescindible, en la mayoría de ámbitos profesionales y personales de la sociedad actual.

Recientemente, a los dispositivos electrónicos se les está dotando de una serie de algoritmos y procesos de razonamiento que permitan proporcionarles *inteligencia* de manera que puedan tomar decisiones por sí mismos en base al contexto de aplicación, la información recabada del ambiente y las interacciones que se realizan con los objetos del ambiente [2].

De la premisa de que el entorno puede trabajar para y por nosotros nace el término *inteligencia ambiental* (Ambient Inteligence - AmI) que propone la creación de entornos o espacios inteligentes que se adapten a las necesidades,

gustos e intereses de las personas que viven, trabajan o transita en ellos [3].

Son inagotables las posibilidades que puede brindar un entorno de inteligencia ambiental, también denominado ambiente inteligente, como reducción del consumo energético, vigilancia o seguridad. Sin embargo, la inteligencia ambiental nace con una especial vocación a mejorar la calidad de vida de las personas en términos de salud, bienestar e independencia [4]. Así, se diseñan y se desarrollan entornos donde los dispositivos son protagonistas y tienen como objetivo asistir a los ocupantes en las tareas que deben realizar cada día, sus costumbres o sus actividades personales, todo ello de una manera inteligente y no invasiva. A dichos entornos inteligentes en el ámbito de la salud, el bienestar y la independencia se les ha denominado *ambientes de vida asistida* (Ambient Assisted Living - AAL) [5].

En un entorno de inteligencia ambiental se pueden distinguir tres pilares: red de sensores, procesos de razonamiento y actuadores [6]. Es de destacar que la materia prima de los entornos de inteligencia ambiental son los datos que recabados por la red de sensores. Dichos datos, por sí mismos, tan sólo ofrecen mediciones aisladas, siendo necesario transformar estos datos y dotarlos de un conocimiento y un lenguaje que cualquier persona de interés en el entorno pueda interpretar de manera sencilla y práctica.

Una vez que se procesan los datos para obtener información útil y conocimiento del entorno de inteligencia ambiental, es de gran valor desarrollar un sistema web de monitorización. De este modo, es posible conocer en tiempo real el estado de cada uno de los sensores del entorno, consultar estados pasados o, incluso, realizar búsquedas de cambios de estado de los sensores en cualquier momento. En este ámbito es donde se fundamenta principalmente la propuesta de este trabajo fin de grado.

Adicionalmente, una fuente poderosa de información son las notificaciones remotas con información relativa a los datos recogidos por los sensores, especialmente en el ámbito de AAL. Así, remotamente pueden ser notificados cambios en el

ambiente debido a interacciones que realice el ocupante con los objetos del mismo o las condiciones ambientes del entorno. Un ejemplo que ilustra tal utilidad puede ser la ubicación de un sensor que mida los niveles de CO2 en la vivienda de una persona que sufra deficiencias en el sentido del olfato y del gusto. La notificación a una tercera persona donde se notifiquen valores anómalos en los niveles de CO2 puede ser una cuestión crítica. Otro ejemplo, es la ubicación de un sensor de interrupción en la puerta principal de la casa que pueda notificar cuando la puerta se abre y se cierra y, por ende, las salidas y entradas que realiza una persona con algún tipo de limitación cognitiva.

Es evidente, por tanto, la utilidad de las notificaciones en un entorno de inteligencia ambiental. Por esta razón, dicho aspecto será abordado en este trabajo fin de grado donde se pretende dar solución a dicha necesidad en contextos donde los ocupantes han visto mermadas algunas de sus capacidades con el objetivo de mejorar sus condiciones de vida y proporcionar tranquilidad a su entorno más cercano como puede ser sus cuidadores o sus familiares.

Una vez mostrada en profundidad la motivación de nuestra propuesta en este trabajo fin de grado, se van a describir determinadas cuestiones tecnológicos sobre la que se sustenta nuestra propuesta.

Se predice que para el 2020 habrá más de 50 billones objetos inteligentes conectados [7] (ver figura 1.1). Los datos generados por esta enorme cantidad de dispositivos necesitarán técnicas especiales de fusión de la información y de acceso de manera eficiente. Normalmente, los datos recogidos se almacenarán en una base de datos específica, que debe cumplir unos patrones para posteriormente facilitar el acceso a esta información.

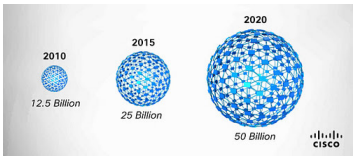

Figura 1.1: Dispositivos conectados (CISCO)

En este trabajo fin de grado, se apostará por desarrollar el sistema de monitorización a través de una arquitectura cliente-servidor, desglosada en los siguientes elementos:

- **API** basada en servicios REST con la tecnología Java, utilizándose en el desarrollo de estos servicios a través del framework Jersey.
- **Sistema web de monitorización de entornos de inteligencia ambiental** para la consulta visual de los servicios ofrecidos por la API anterior, utilizándose varios framework de programación.
- **Persistencia** mediante MySQL, donde se incluirán todos los datos relativos al sistema: entornos, sensores, objetos y usuarios.

Gracias a la interoperabilidad de estos elementos, la propuesta de esta trabajo fin de grado brindará la posibilidad de monitorizar en tiempo real lo que ocurren en diferentes entornos de inteligencia ambiental, así como consultar, mediante distintas vías, los datos históricos almacenados en cada uno de los entornos registrados en el sistema.

1.2. Propuesta

El propósito de este trabajo de fin de grado es desarrollar un sistema web para la monitorización de entornos de inteligencia ambiental junto a un prototipo de aplicación móvil que permita la suscripción y notificación de eventos en un entorno.

1.3. Objetivos

Los objetivos que derivan de la propuesta de este trabajo fin de grado son los siguientes:

1. Definir el conjunto de servicios necesarios para desarrollar el sistema de monitorización de entornos de inteligencia ambiental, también denominados ambientes inteligentes.

2. Desarrollar los servicios Web asociados así como la definición e implementación de la base de datos para el almacenamiento de la información necesaria.

3. Desarrollar un prototipo de aplicación web que permita la monitorización de ambientes inteligentes.

4. Desarrollar un prototipo de aplicación móvil para la notificación de cambios en el ambiente inteligente.

5. Realizar los manuales asociados a los dos prototipos.

6. Redactar una memoria que recoja todo el trabajo desarrollado así como los manuales de instalación y usuario.

1.4. Planificación temporal

A continuación, se indica la estructura de trabajo propuesta en la figura 1.2 junto a la estimación inicial de tiempos que aparece reflejada en la tabla 1.1 y, finalmente, el diagrama de Gant que se ilustra en la figura 1.3.

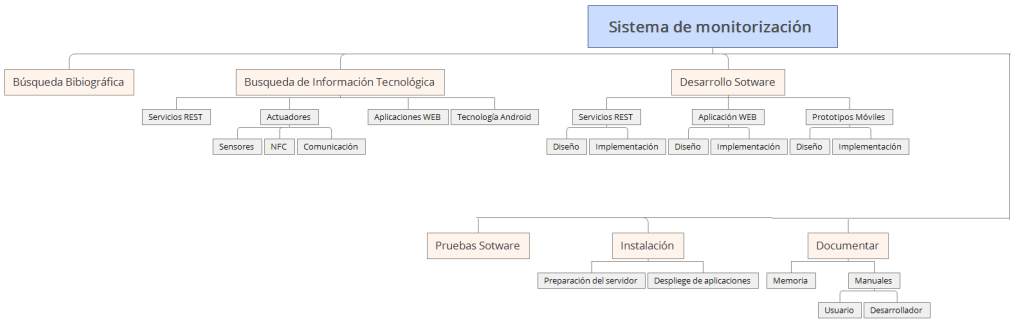

Figura 1.2: Estructura del plan de trabajo

1.4.1. Estimación de tiempos

La estimación de tiempos es una perdición optimista de la duración que tendrá cada uno de los ámbitos del trabajo fin de grado. En la tabla 1.1 se puede ver con detalle cada uno de estas ámbitos acompañado de la duración. Obteniendo como resultado un total de 205 días de trabajo repartidos en 4 horas de trabajo diarias.

Tarea	Tiempo (Días)
Búsqueda bibliográfica	15
Búsqueda de información tecnológica	
Servicios REST	3
Actuadores	5
Aplicaciones WEB	4
Tecnología Android	3
Desarrollo Software	
Servicios REST	58
Aplicación WEB	42
Prototipos Móviles	18
Pruebas Software	10
Instalación	
Preparación del servidor	4
Despliegue de aplicaciones	3
Documentación	
Memoria	30
Manuales	10
Total	205

Tabla 1.1: Estimación de tiempos

1.4.2. Diagrama Gantt

El diagrama de Gantt es una herramienta, el cual expondrá el tiempo y dedicación prevista en las diferentes tareas que esta dividido un proyecto de manera gráfica. A su vez, este diagrama, muestra la duración total y un las fechas establecidas para cada tarea.

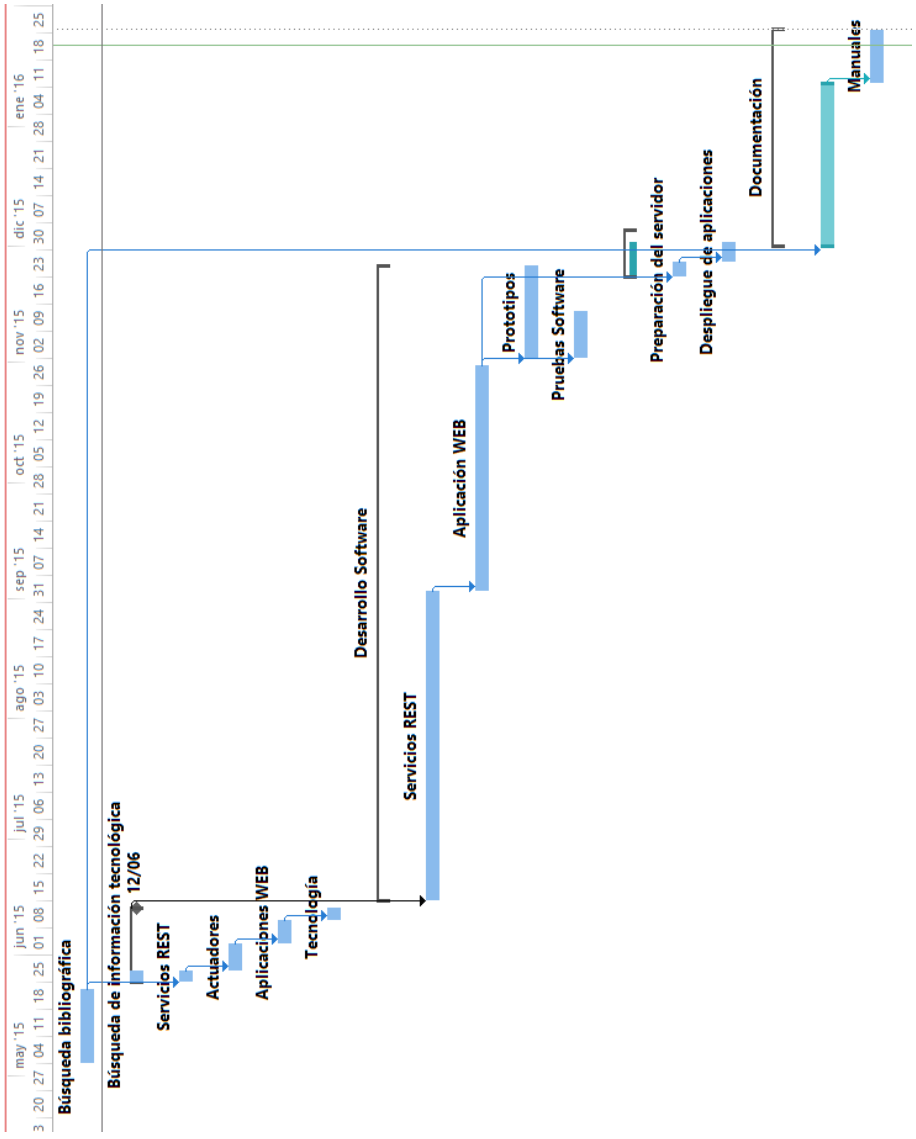

Figura 1.3: Diagrama de Gantt

1.5. Estructura de la memoria

A continuación, se va a describir la estructura de la memoria de este trabajo fin de grado.

- **Capítulo 1.** En este capítulo, se ha expuesto la motivación específica de este trabajo, justificando la realización del mismo, la propuesta concreta que perseguimos y los objetivos que nos marcamos alcanzar. Finalmente, se ha indicado la planificación temporal del trabajo.

- **Capítulo 2.** En este capítulo se va a describir con más detalle el concepto de ambientes inteligentes y el paradigma en el que se apoya, Internet de las Cosas. Además, se van a describir algunos de los laboratorios de inteligencia ambiental más relevantes y, con más detalle, el laboratorio de inteligencia ambiental del Centro de Estudios Avanzados en Tecnologías de la Información y Comunicación (CEATIC), ya que éste, principalmente, ha sido el laboratorio que se ha utilizado en esta propuesta como entorno de inteligencia ambiental. Además, se incluirán las aplicaciones más destacas de los entornos de inteligencia ambiental. Para finalizar el capítulo, se indicarán los aspectos más destacados de tecnologías de sensores y estándares de comunicación, ya que sobre ellos se apoya la monitorización de ambientes inteligentes.

- **Capítulo 3.** En este capítulo se presenta el proceso de ingeniería software que se ha seguido en la elaboración de la propuesta, especificación de requerimientos, análisis, diseño, implementación del sistema y pruebas. Para cada una de las fases se hará una distinción para describir el sistema web de monitorización y el prototipo de aplicación móvil para la notificación de eventos. Dentro de la implementación del sistema se especificará las tecnologías utilizadas y cuales son las herramientas de desarrollo elegidas para la

elaboración de este trabajo.

- **Capítulo 4.** En este último capítulo, se expondrán las conclusiones que nos ha aportado la realización del presente trabajo fin de grado junto a las posibles líneas de trabajo futuras.

- **Bibliografía.** En este punto termina la estructura principal de esta memoria, aquí se plasmará toda bibliografía que aúna todas las referencias que han sido consultadas, para la realización de este trabajo fin de grado.

- **Anexos.** En el anexo A, se visualizará el contenido del CD-ROM adjunto con esta memoria. Se describirá con detalle cada carpeta y archivo, junto con una copia de esta memoria. En los siguientes anexos, Anexos B, C y D, se incluyen los manuales para la instalación, administración y utilización del software elaborado en este trabajo fin de grado.

Capítulo 2

Ambientes inteligentes

Una de las citas más comunes de ambientes inteligentes la encontramos en la definición proporcionada por Alan Steventon y Steve Wright [8]: *Los ambientes inteligentes son sistemas en los que la computación es usada para mejorar las actividades comunes.* Desglosando dicha definición, se aprecia que el principal objetivo es incluir a la tecnología para realizar nuestras tareas diarias de forma inmersa. Para dicho motivo es habitual que se relacione ambientes inteligentes con domótica, un concepto relacionado, aunque más primitivo y menos ambicioso. La domótica, utiliza un conjunto de técnicas para automatizar una vivienda o entorno de inteligencia ambiental, pero si utilizamos estas técnicas junto con una lógica que aprenda y mejore la estancia del habitante, entonces se deriva en un *Ambiente Inteligente*. A lo largo de esta memoria también podemos encontrarnos esta definición como **Entorno de inteligencia ambiental**.

Para aplicar las técnicas que anteriormente se ha mencionado, es necesario un sistema que sea capaz de recabar y monitorizar todo lo que sucede en un ambiente de inteligencia ambiental. Para ello, es fundamental contar con una red de sensores que permitan recoger información del ambiente. Dicha información puede ser ambiental, como temperatura, luz o niveles de CO2, o información relacionada con los cambios que se producen sobre los objetos del ambiente, como

abrir y cerrar puertas o la presión ejercida en una silla o una cama al estar sobre ellos.

Es importante destacar que la red de sensores ubicada en el entorno de inteligencia ambiental que se pretende monitorizar debe estar desplegada de manera transparente para el habitante.

2.1. Internet de las cosas

Internet de las cosas (Internet of Things - IoT) se considera como una evolución de Internet. Dicha evolución dota de conexión a la red de Internet tanto a personas como a dispositivos [9]. Cada vez es mayor la cantidad de dispositivos que tenemos conectados a la red de Internet, automóviles, teléfonos, relojes incluso electrodomésticos. Por dicho motivo, las posibilidades que surgen por la utilización del paradigma de IoT son inmensas, siendo las más prometedoras en el ámbito de la salud, logística, medio ambiente y el hogar [10].

Para entender la repercusión que está teniendo el IoT, se puede mencionar el logro del 10 % mundial en ahorro energético [11]. Un gran ejemplo de este avance reside en grandes edificios donde el consumo de electricidad, agua o otras fuentes de energía son malgastadas. Mediante dispositivos conectados a internet se pueden realizar mediciones inteligentes para reducir dichos consumos.

Por citar otros ejemplos en el ámbito del hogar se pueden destacar varios grupos de aplicaciones.

- **Electrodomésticos Inteligentes**: Funcionan de manera similar a los electrodomésticos convencionales, la gran novedad reside en una red de sensores instalados en puntos clave del electrodoméstico, estos sensores estarán conectados a internet y enviarán información relevante en tiempo real al usuario.
- **Automatización**: La gran novedad en esta aplicación es que podemos conectar todos nuestros dispositivos del hogar a internet y controlarlos remo-

tamente con un simple smartphone o teléfono inteligente.

- **Monitorización**: Por último, una de las aplicaciones más relevantes es la monitorización en tiempo real de una vivienda, recibiendo notificaciones cuando se produzca una variación en cualquiera de los sensores que tenemos conectados.

Esta revolución, se inició a partir del 2007 y, actualmente, no para de crecer. Las empresas más importantes están desarrollando dispositivos e incluso sus propios protocolos para hacerse un hueco en este gran mercado. A continuación se citan las empresas más importantes que están trabajando en proporcionar sus propias soluciones en el paradigma IoT:

- **Samsung SmartThings** [1]: Es una linea de sensores muy sencilla de utilizar los cuales se conectan a un HUB central para monitorizar todo los parámetros que recogen cada uno de los sensores. El principal atractivo de esta oferta es la compatibilidad con más de 300 dispositivos distintos bajo el protocolo de comunicación Z-Wave y ZigBee.

  SmartThings proporciona un control total de la vivienda o entorno de inteligencia mediante nuestro dispositivo móvil o tablet.

- **Google Brillo** [2]: Actualmente se encuentra en desarrollo, siendo un sistema basado en Android, diseñado para los dispositivos IoT, gracias a esta propuesta es posible conectar diferentes dispositivos bajo un mismo lenguaje/protocolo denominado Weave.

- **Beclose** [3]: Gracias a la red de sensores proporcionados por esta compañía, este sistema monitoriza todas las actividades dentro de un ambiente y envía alertas a familiares o cuidadores.

[1] https://www.smartthings.com/
[2] https://developers.google.com/brillo/
[3] http://beclose.com/

2.2. Aplicaciones

A continuación, se indican algunas de las aplicaciones más relevantes que pueden ser desplegadas en los ambientes inteligentes.

- **Teleasistencia**

  Se define teleasistencia como el conjunto de servicios de atención médica o emergencias, prestados a través de dispositivos personales de alarma [12]. Estos servicios, mejoran la calidad de vida de las poblaciones de tercera edad y personas con alguna discapacidad. Uno de los dispositivos más novedosos para esta aplicación se encuentra en el laboratorio de inteligencia ambiental de la universidad de Jaén. Gracias a este dispositivo, podemos detectar caídas o estudiar la movilidad que tiene un individuo dentro del ambiente.

  Estas aplicaciones, además de crear alarmas por si ocurre alguna anomalía en el paciente que están monitorizado, tienen la posibilidad de hacer un seguimiento mucho mas minucioso, conectando directamente el paciente con una persona especializada.

- **Seguridad**

  Otra de las aplicaciones más utilizadas dentro de los ambientes inteligentes es la seguridad. Es un campo muy estudiado, ya que tiene una influencia muy fuerte con la domótica y los sistemas de alarmas. En este ámbito, el ambiente puede detectar automáticamente algunas situaciones e interactuar por sí sólo. Alguno ejemplos en este ámbito, pueden ser la simulación de personas virtuales que residen en la vivienda, o notificar a dispositivos móviles asociados al sistema de que hay individuos cerca o llamando a nuestra puerta.

- **Seguimiento de actividades**

  El análisis del seguimiento de actividades es una tarea novedosa que puede facilitar el tratamiento de muchas enfermedades o demencias. Mediante varios dispositivos instalados por todo el ambiente inteligente, podemos tratar de identificar las tareas cotidianas que realiza un habitante.

  Una vez identificadas, el sistema puede detectar cambios en el comportamiento cuando se realizan diferentes tareas. Estos cambios incluyen el simple hecho de dejar de realizar una tarea, hasta alterar su duración o orden. Para estas aplicaciones, los dispositivos más utilizados suelen ser balizas que detectan la situación del habitante, las cámaras de vídeo, las etiquetas NFC y los sensores de interrupción.

  En el ámbito del seguimiento de actividades es importante destacar el uso de pulseras inteligentes. Gracias a ellas, podemos monitorizar constantes vitales de cada uno de los habitantes. Aquí las posibilidades son muy extensas, desde un simple seguimiento de las rutinas, hasta monitorizar las pulsaciones en tiempo real o motivar al individuo para practicar deporte.

- **Accesibilidad**

  Mediante esta aplicación los ambientes inteligentes se adaptan a todo tipo de personas independientemente de cual sea su limitación o discapacidad. Gracias a esto, se ofrece más autonomía al habitante a la hora de realizar sus tareas y quehaceres cotidianos.

  Algunos dispositivos que se pueden utilizar para esta aplicación son i) los teléfonos implementados con sensores visuales y vibración para personas con discapacidades auditivas, ii) el sonido en cada una de las teclas para discapacidad visual, y iii) dispositivos para facilitar el movimiento como Emotiv Epoc o IRISCOM.

- **Escenas**

  En los entornos de inteligencia ambiental también se pueden configurar escenas donde diferentes actuadores se coordinan para facilitar a los habitantes las tareas que realizan.

  Estas aplicaciones se utilizan con individuos con problemas de sueño o concentración. Tan solo debemos de contar con dispositivos sonoros y luminosos. Estos dispositivos crearán un ambiente especifico dependiendo de la actividad que realice, modificando los atributos de la luminosidad y reproduciendo sonidos en cada una de las estancias.

- **Eficiencia Energética**

  Por último, se encontrará con la área de aplicación la cual gestionará todo el sistema, dependiendo de la temperatura, luminosidad y valores, tanto externos como internos.

  De este modo, no hará falta tener una bombilla encendida si no hay nadie en una habitación, o ajustar la temperatura de la calefacción dependiendo de la temperatura exterior y a las preferencias de los usuarios que se encuentran en las estancias.

2.3. Laboratorios de inteligencia ambiental

Como se ha descrito anteriormente los laboratorios de inteligencia ambiental conciben un espacio donde los objetos cotidianos se encuentran conectados a la red para así dotar a nuestro entorno con nuevas funcionalidades.

Muchos grupos de investigación están desarrollando sus propios laboratorios de inteligencia ambiental para realizar estudios relacionados en este ámbito. A continuación, se mencionan algunos de ellos:

- **Intelligent Systems Lab - Universidad Carlos III de Madrid** [4]: La Universidad Carlos III de Madrid proporciona una linea de investigación sobre sistemas inteligentes. Llevando estos estudios en un laboratorio. Esta linea de investigación además, trabaja con vehículos autónomos o drones.

- **Ceapat-Imserso** [5]: El Centro Estatal de Autonomía Personal y Ayudas Técnicas (CEAPAT), perteneciente al IMSERSO, cuenta con un laboratorio en su sede el cual esta dotado de tecnologías accesibles facilitar las tareas del hogar.

- **Smart Environemnt - Universidad de Ulster** [6]: El Campus Jordanstown de la universidad de Ulster ha creado el grupo SERG (Smart Environment Research Group). Este grupo está compuesto de una serie de espacios de trabajo dedicados al estudio de los entornos inteligentes.

- **SeniorLAB** [7]: Una casa inteligente para personas mayores. La empresa tecnológica CETIEX ha desarrollado el proyecto SeniorLAB para mejorar la calidad de vida de personas mayores que viven de forma autónoma.

- **Apartamento de inteligencia ambiental del CEATIC** [8]

Para este trabajo fin de grado se han utilizado los datos proporcionados por el laboratorio de inteligencia ambiental del CEATIC. Por dicho motivo, se describe en detalle, dicho entorno.

[4] www.uc3m.es/islab
[5] www.ceapat.org/
[6] http://scm.ulster.ac.uk/ scmresearch/SERG/
[7] http://www.cetiex.es/
[8] http://ceatic.ujaen.es/es/smart-lab-0

Apartamento de inteligencia ambiental-CEATIC

El apartamento de inteligencia ambiental del Centro de Estudios Avanzados en Tecnologías de la Información y de la Comunicación (CEATIC) [13] se encuentra ubicado en la dependencia 109 del edificio C6.

La distribución del apartamento de inteligencia ambiental se muestra en la figura 2.1, cuenta con un espacio de 25 metros cuadrados. Este espacio está distribuido en tres zonas cocina, sala de estar y salón.

Entrando en detalle en cada una de estas estancias, la cocina, esta equipada con todos los electrodomésticos necesarios en una vivienda, frigorífico, microondas, vitrocerámica, extractor de humos entre otros.

El salón, esta equipado con una televisión inteligente, a la que se tiene conectado una videoconsola XBOX ONE con el dispositivo Kinect, para su utilización sin controles físicos. A su vez, esta estancia cuenta con una estación de trabajo encargada de recoger la información de todos los dispositivos.

Por último, el dormitorio, esta equipado con un pequeño aseo, e inodoro para simular las tareas cotidianas que se realizan en cualquier vivienda.

Figura 2.1: Laboratorio CEATIC

2.4. Sensores

2.4.1. Tecnología de sensores

Un aspecto fundamental en la red de sensores que nutre a un ambiente inteligente es la tecnología que utilizan cada uno e los sensores para comunicarse y como es su funcionamiento interno.

Si nos centramos desde un punto de vista mas cercano a la electrónica, cada sensor tiene un funcionamiento específico, por lo que podemos agruparlos en dos amplios grupos. [14]

- **Fuente eléctrica**, este primer grupo, nos muestra cómo los sensores tienen dos modos de trabajar, de manera activa o pasiva.
  - *Sensores pasivos* tienen la ventaja de que no necesitan ningún tipo de alimentación externa para funcionar. Directamente, estos sensores generan una señal cuando se realiza un estimulo sobre ellos. Algunos ejemplos podrían ser un simple termómetro de mercurio, donde varía la densidad del metal al aplicar una diferencia de temperatura o un sensor piezoeléctrico [14], que al ser presionado genera una señal eléctrica. Es muy común utilizar este sensor para detectores de presión o para medir la aceleración.
  - *Sensores activos*, que para funcionar requieren de una fuente externa a ellos. Algunos ejemplos más comunes son sensores inductores para detectar la presencia de metales o sensores termistor [15], usados generalmente para la medición de temperaturas.
- **Naturaleza de la señal**, donde se describe la señal que produce cada sensor [14, 16].

- *Señal analógica*, representable con una función matemática continua donde muestra la información de una variable en función al tiempo. Son muy adecuadas para el envío de audio.
- *Señal digital*, a diferencia de la analógica que utilizaba una función continua, aquí la señal tomará unos valores discretos, 0 y 1, representando un bit de dos amplitudes distintas.

Una vez conocido los diferentes modos de funcionamiento de un sensor, mencionaremos los parámetros más frecuentes [16]:

- **Rango:** Conjunto de valores que pueden ser medidos, comprendidos entre el máximo y el mínimo detectable.
- **Sensibilidad:** Mínimo valor, encargado de producir un cambio.
- **Error:** Desigualdad entre el valor obtenido en la medición y el valor verdadero. Aquí encontraremos dos tipos. Error absoluto y relativo.

  Es muy importante destacar que este margen de error, se agrava por la aparición de ruido, señal que no contiene información.
- **Capacidad de respuesta:** Dependiendo del sensor puede ser establecida de manera fija, o depender de la magnitud que esta midiendo, siendo distinta.

Tras entender el funcionamiento desde un punto de vista cercano a la electrónica, se va a describir el comportamiento de los sensores donde se encuentran dos puntos [17], dependiendo del modo en el que los sensores envíen la información. Para poder categorizarlos, se utilizará como criterio la la frecuencia con la que envían la información.

- **Sensores baja frecuencia**. No tienen un intervalo de tiempo definido para enviar la información. Estos sensores, están siempre a la escucha y cada vez que un medio externo altera el valor de los sensores, este

se dispara automáticamente y es en este momento cuando realiza el envío de información. Este tipo de sensores son muy útiles, ya que nos indican cuando un objeto ha sido alterado o manipulado.

- **Sensores alta frecuencia** Están programados para enviar la información en una constante de tiempo establecida al desplegarlos. Antes de realizar un envío, se debe de aplicar una operación estadística (media), a los datos recogidos durante el tiempo de medición.Por lo que finalmente se hace el envío de esta operación.

2.4.2. Estándares de comunicación

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

A continuación se indican los protocolos de comunicación más importantes:

- **ZigBee:** Es un protocolo reciente, fijándonos en la fecha en la que se aprobó, vemos que no ha pasado mucho tiempo. Esto nos da una pista del auge que esta teniendo esta tecnología. En cuanto a su comunicación, es inalámbrica, haciendo uso del estándar IEEE 802.15.4 (Radio-fusión de bajo consumo). [18]

- **Threat:** Este estándar pertenece a la compañía Google, tras realizar en 2014 la compra de la empresa Nest a la que pertenecía. El objetivo principal de este estándar es conectar un gran numero de dispositivos independientemente del la comunicación que venga establecida por sus proveedores (WiFi, Bluethooth o ZigBee). Para ello Threat hace uso de 6LoWPAN [19] bajo el protocolo IEEE 802.15.4.

- **Z-Wave:** Esta comunicación utiliza los mismos principios que la conocida ZigBee, la principal ventaja que diferencia esta tecnología de ZigBee es el rango de frecuencia con el que trabaja [20]. Haciendo uso de la banda 900Mhz frente a la 2.4 GHhz. Anteriormente se ha tratado esta diferencia como una ventaja, ya que a más baja frecuencia obtenemos una mayor facilidad para que las ondas atraviesen paredes u otros obstáculos comunes de los ambientes inteligentes.

- **Bluetooth LE:** (Low Energy) Es un nuevo sistema, versión 4.0, de la conocida comunicación Bluetooth. Emite en la banda 2.4 GHz con un alcance teórico de 100 metros. La gran particularidad de esta tecnología es la poca energía que necesita para su funcionamiento [21].

2.4.3. Sensores del apartamento de inteligencia ambiental del CEATIC

Dado que en este trabajo fin de grado se va a monitorizar el apartamento de inteligencia ambiental del CEATIC, en esta sección se describen los dos tipos de sensores que van a nutrir el sistema y qué tipo de información utilizan.

- **Sunspot:** Son unos sensores ambientales (figura 2.2) de la empresa Sun Microsystems adquirida en 2010 por Oracle. Estos sensores utilizan el estándar de comunicación ZigBee y son capaces de detectar temperatura, luminosidad y movimiento [22].

Figura 2.2: Sensor Sunspot

- **Tynetec:** Sensores de contacto (figura 2.3), utilizados para detectar manipulación en cualquier punto en el que se encuentren instalados (puertas, ventanas, teléfono o cualquier dispositivo), el funcionamiento de este tipo de sensores es muy simple. Cuenta con dos placas, que cuando son separadas se envía una señal (mediante un protocolo propio) a un receptor, el cual la interpreta [23].

Figura 2.3: Sensor Tynetec

Capítulo 3

Proceso de ingeniería del software

En este capítulo se va a realizar una descripción detallada del proceso de ingeniería del software llevado a cabo para culminar con éxito nuestra propuesta consistente en un sistema web de monitorización de entornos de inteligencia ambiental, al cual denominaremos *Sistema web* y un prototipo de aplicación móvil para suscripción y notificación de eventos en entornos de inteligencia ambiental, al cual denominaremos *Prototipo de aplicación móvil*.

Como se indicó en el capítulo 1, nuestro trabajo fin de grado contará con los siguientes pilares:

- **API** basada en servicios REST con tecnología Java. En el desarrollo de estos servicios se ha utilizado Jersey (Implementación de referencia JAX-RS).
- **Sistema Web** para el manejo mediante una interfaz de usuario de los servicios ofrecidos por la API anterior. Para el desarrollo de esta aplicación se han utilizado varios framework de programación que se describirán con más detalle posteriormente.
- **Persistencia** se hará uso de una base de datos relacional MySQL, donde se incluirá todos los datos relativos al sistema, sensores, objetos y usuarios.

Antes de describir el proceso que se ha llevado a cabo para desarrollar el software del presente trabajo fin de grado, se van a citar algunas definiciones de dicho proceso y las fases de las que consta.

Ingeniería del Software es la construcción de software de calidad con un presupuesto limitado y un plazo de entrega en contextos de cambio continuo.

Ingeniería del Software es el establecimiento y uso de principios y métodos firmes de ingeniería para obtener software económico que sea fiable y funcione de manera eficiente en máquinas reales.

3.1. Fases de la ingeniería del software

Las fases del proceso de ingeniería del software son las siguientes:

- **Especificación de Requerimientos:** Requisitos y funcionalidades que tendrá el sistema.
- **Análisis del Sistema:** Especificación formal de los requerimientos del sistema.
- **Diseño del Sistema:** Especificación del funcionamiento para satisfacer los requisitos analizados.
- **Implementación del sistema:** Desarrollo del software del sistema, cumpliendo todos los requerimientos bajo el diseño anteriormente establecido.
- **Pruebas:** Verificar y validar que cumple con los requerimientos y estándares del sistema.

3.2. Especificación de requisitos

3.2.1. Requerimientos del sistema web

Al ser la primera fase en nuestro proceso de ingeniería, vamos a determinar cuáles son las claves donde nuestro sistema debe dar las mejores soluciones. Es muy importante definir bien este punto, ya que será el pilar del software desarrollado. Para ello, vamos a proceder a realizar una descripción completa del comportamiento del sistema, dividiendo los requisitos en dos grupos:

- **Requerimientos funcionales:** Aquellos requisitos que definen una función del sistema o de sus componentes, centrándose en los parámetros de entrada, flujos de datos y respuestas esperados en sus casos de uso.
- **Requerimientos no funcionales:** Todos los demás requisitos que especifican criterios utilizados para estudiar comportamientos específicos. Estos abarcan desde restricciones de tiempo, tipos de licencias de uso, elecciones de lenguaje de programación, etc.

A continuación de describirán con detalle cada uno de estos requerimientos del sistema.

3.2.1.1. Funcionales

Para este trabajo fin de grado, se han considerado los siguientes requerimientos funcionales:

- **Requerimiento funcional 1: Login Administración**

  El sistema debe de proporcionar un formulario para que el administrador pueda acceder a un panel privado para configurar los ambientes que estarán en el sistema. A su vez, el sistema permitirá cerrar la sesión al terminar su uso.

- **Requerimiento funcional 2: Gestionar ambientes**

  El sistema podrá gestionar (visualizar/filtrar/editar/borrar) los ambientes por parte del administrador del sistema. El sistema agrupará cada información en los siguientes cuatro grupos:

  - **Información del ambiente** El sistema guardará información para la futura identificación del ambiente.
  - **Seguridad del ambiente** El sistema guardará si el ambiente es publico o privado
  - **Información de la base de datos** El sistema almacenará la información de la base de datos de cada ambiente.
  - **Mapa**.

- **Requerimiento funcional 3: Gestionar Sensores**

  El sistema debe de proporcionar mediante diferentes paneles la posibilidad de gestionar (visualizar/filtrar/editar/borrar) toda la información referente a los sensores que tendremos dados de alta en todo el sistema. También, el sistema proporcionará un histórico de dónde han estado instalados cada uno de los sensores del sistema. A su vez, el sistema categorizará los sensores según su estado *Activo* o *No Activo*.

- **Requerimiento funcional 4: Seguridad**

  El sistema debe de permitir al administrador del sitio cambiar sus credenciales y dar de alta a nuevos administradores para que gestionen los ambientes del sistema.

- **Requerimiento funcional 5: Gestionar Objetos**

  El sistema debe de permitir a los administradores asociar objetos creados manualmente. Una vez dado de alta un objeto en el sistema, este permitirá

al administrador editar su descripción, sensor al que pertenece o dar de baja dicho objeto.

- **Requerimiento funcional 6: Gestionar Mapa**
  El sistema debe de permitir al administrador situar en el mapa del ambiente cada objeto dado de alta.

- **Requerimiento funcional 7: Visualizar ambientes**
  El sistema debe de ofrecer una lista de ambientes para permitir al usuario elegir cual desea monitorizar. Para realizar esta acción el sistema ofrecerá la siguiente información por cada uno de los ambientes:

  - Mapa
  - Nombre
  - Descripción

- **Requerimiento funcional 8: Visualizar el estado actual**
  El sistema debe permitir visualizar a cualquier usuario el estado actual en el que se encuentra cada uno de los ambientes dados de alta en el sistema. Para realizar esto, el sistema ofrecerá la información de cada uno de los objetos que están asociados al ambiente seleccionado. A su vez, el sistema mostrara toda la información de manera gráfica y resumida sobre el mapa asociado a un ambiente. Por ultimo, el sistema debe de realizar alertas visuales cuando se produzca un evento en uno de los objetos establecidos.

- **Requerimiento funcional 9: Visualizar histórico**
  El sistema deberá permitir al usuario reproducir la situación en la que se encontraba un ambiente un día determinado, seleccionando un día concreto a través de un calendario. Posteriormente, el sistema consultará todos los eventos producidos en dicho día.

  Para recorrer la franja horaria, el sistema debe de ofrecer dos alternativas:

- Manual: El sistema permitirá deslizar una barra de desplazamiento seleccionando la hora dónde quiere visualizar el estado en el que se encontraba el ambiente.
- Automática: El sistema moverá automáticamente la hora en un tiempo constante, representando en este tiempo cómo se encontraba el ambiente.

El sistema realizará alertas visuales cuando se produzca un evento en uno de los objetos establecidos.

- **Requerimiento funcional 10: Visualizar Eventos**
  El sistema debe de permitir al usuario visualizar en modo texto todos los eventos que se ha producido en el ambiente. Para ello, el usuario podrá filtrarlos por cada uno de los objetos que están dados de alta en el ambiente, permitiéndole tres tipos de consulta:
  - Intervalo de fecha
  - Últimos N cambios
  - Últimos N días
- **Requerimiento funcional 11: Internacionalización**
  El sistema debe de permitir al usuario intercambiar en cualquier momento el idioma en el que se visualiza.

3.2.1.2. No funcionales

Como dijimos anteriormente, los requerimientos no funcionales nos describirán las restricciones o limitaciones que han de tenerse en cuenta para que nuestro trabajo fin de grado pueda utilizarse correctamente en un entorno de inteligencia ambiental. Dichos requerimientos serán los siguientes:

- **Requerimientos Físicos**

  Al hacer uso de una arquitectura cliente-servidor, será necesaria una estación fija que ofrezca todos los servicios implementados. A continuación, detallaremos las especificaciones mínimas para poder ejecutar nuestros servicios.

  - Procesador: 2,40 GHz
  - RAM: 1GB
  - HDD: 10 GB
  - Salida a Internet necesaria

  En cuanto al cliente, solo será necesario tener acceso a Internet pero es recomendable cumplir los siguientes requisitos mínimos:

  - Procesador: 1,20 GHz
  - RAM: 2GB
  - HDD: 50 GB
  - Salida a Internet necesaria
  - Resolución pantalla 1024x768

- **Requerimientos Software**

  Al igual que en los Requerimientos Físicos, vamos a dividir estos requerimientos entre el cliente y el servidor.

  Servidor:

  - Distribución libre Linux
  - Java 1.7 o superior
  - Tomcat 1.8
  - MySQL 4.1

- Apache2

Cliente:

- Navegador Web (Firefox o Chrome)
- Sistema operativo Windows, Linux o Mac OS
- Soporte para JavaSript

- **Requerimientos de la interfaz**

  - **Flexibilidad** Se debe adaptar a cada uno de los ámbitos donde el usuario esta visualizando esta aplicación.
  - **Robustez** Debe estar pensada para soportar fallos producidos por los usuarios.
  - **Facilidad** Ha de ser intuitiva y fácil para el usuario que aún no está familiarizado con ella.

3.2.2. Requerimientos en el prototipo de aplicación móvil

3.2.2.1. Funcionales

Para nuestro prototipo de aplicación móvil, se han considerado los siguientes requerimientos funcionales:

- **Requerimiento funcional 1: Identificador del dispositivo**
  El sistema debe tener un mecanismo para que obtener un ID único para registrar nuestro dispositivo móvil en la base de datos del sistema.

- **Requerimiento funcional 2: Monitorizar estado de un ambiente**
  El sistema se conectará a los servicios proporcionados por la aplicación final para obtener los valores de como se encuentra cada uno de los ambientes que estén registrados en el sistema.

- **Requerimiento funcional 3: Suscripción a objetos**

  El sistema ofrecerá al usuario una lista de objetos perteneciente a cada ambiente. De este modo, el usuario elegirá que eventos quiere recibir en su dispositivo en forma de alerta.

- **Requerimiento funcional 4: Notificaciones**

  El sistema creará unas alertas cada vez que reciba una señal de cambio de unos de los objetos a los que está suscrito, de este modo se notificará mediante un mensaje Push al usuario.

3.2.2.2. No funcionales

- **Requerimientos Físicos**

  Al ser un prototipo de aplicación móvil, necesitamos un dispositivo con las siguientes características.

  - Android 4.0 o superior
  - Conexión a Internet
  - Resolución Mínima 800x400 píxeles

3.3. Análisis del sistema

3.3.1. Casos de uso del sistema web

Los diagramas de casos de uso muestran el comportamiento del sistema desde el punto de vista del usuario, representando las funciones que un sistema puede ejecutar.

Cada caso de uso, está formado por los siguientes elementos:

- **Titulo:** Identificador único.
- **Actor:** Usuario principal que utilizará el sistema.

- **Condiciones de entrada:** Precondiciones que se deben de cumplir antes de realizar la acción.
- **Flujo de eventos:** Eventos que ocurrirán al realizar la acción.

Antes de mostrar los casos de uso principales de nuestro sistema, vamos a presentar el diagrama frontera (ver figura 3.1) que recogerá los principales requisitos presentados en la sección anterior.

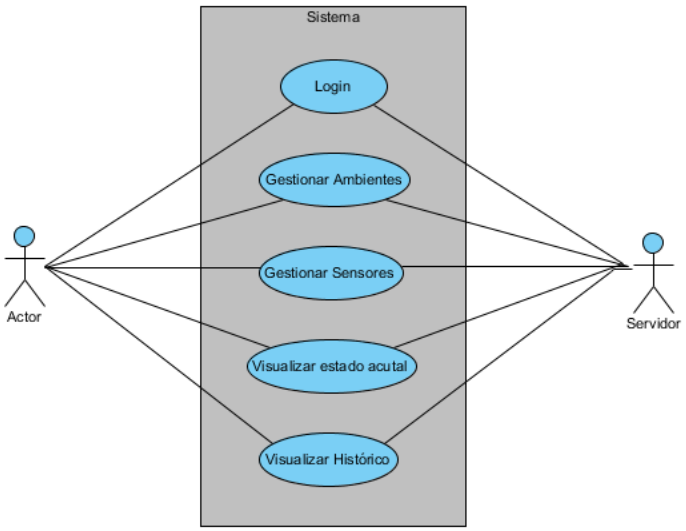

Figura 3.1: Diagrama frontera

En los siguientes diagramas (ver figuras desde 3.2 a 3.8), al detallar cada caso de uso se pueden emplear dos tipos de relaciones:

- **extend:** La dirección de este tipo es hacia el caso de uso y representa alternativas.
- **include:** Relación contraria a extend, representa acciones de uso común.
- **Caso de uso 1: Identificación de usuario**

Figura 3.2: Caso de uso: Identificación de usuario

- **Actores:** Administrador
- **Condiciones entrada:** Deben existir usuarios en el sistema.
- **Eventos:**
  1. El sistema muestra un formulario solicitando los datos de entrada.
  2. El administrador introduce sus credenciales.
  3. El sistema comprueba, mediante la base de datos y los servicios, que el usuario sea correcto. (E1)
  4. El sistema muestra el panel de administración.
- **Excepciones:** E1: Las credenciales no son correctas, se vuelven a pedir los datos.

- **Caso de uso 2: Gestionar Ambientes**

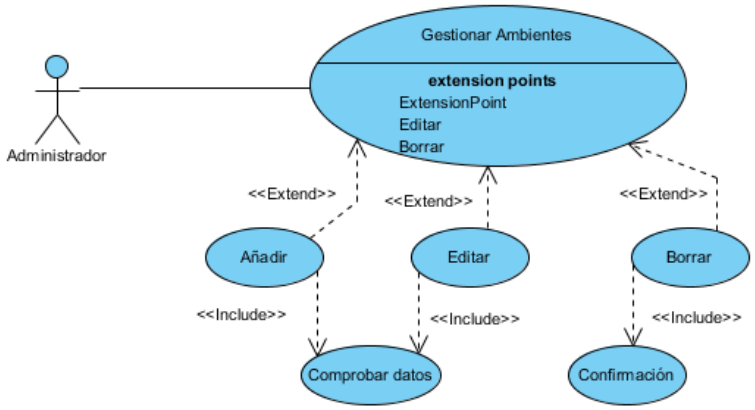

Figura 3.3: Caso de uso: Gestionar ambientes

- **Actores:** Administrador
- **Condiciones entrada:** Debe de estar identificado en el sistema.
- **Eventos:**
  1. El sistema muestra todos los ambientes del sistema.
  2. El administrador podrá crear un nuevo ambiente. (S1)
     a) El sistema muestra un formulario solicitando los datos del ambiente.
     b) El administrador introduce los datos solicitados.
     c) El sistema comprueba si los datos son correctos. (E1)
     d) El sistema muestra el ambiente creado.
  3. El administrador podrá editar un nuevo ambiente. (S2)
     a) El sistema muestra un formulario con los datos actuales del ambiente.
     b) El administrador introduce los cambios deseados.
     c) El sistema comprueba si los datos son correctos. (E1)

d) El sistema muestra el ambiente editado.

4. El administrador podrá borrar un nuevo ambiente. (S3)

a) El sistema muestra una alerta avisando de la perdida de datos.

b) El administrador confirma dicha alerta.

c) El sistema muestra los ambientes restantes.

- **Excepciones:** E1: Los datos del ambiente no son correctos, el sistema vuelve a pedir dichos datos.

- **Caso de uso 3: Gestionar Sensores**

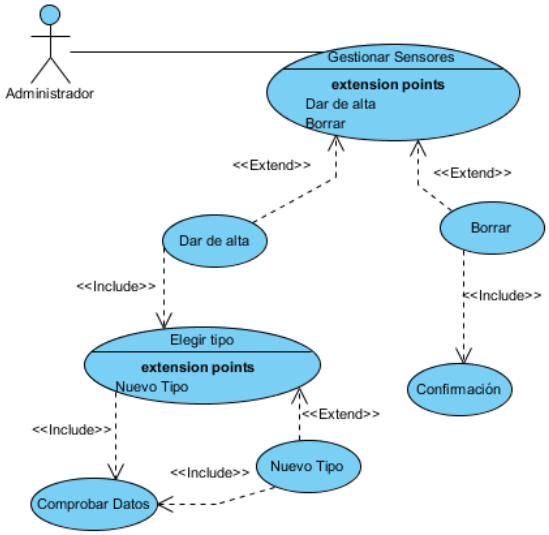

Figura 3.4: Caso de uso: Gestionar sensores

- **Actores:** Administrador
- **Condiciones entrada:** Debe de estar identificado en el sistema.
- **Eventos:**
  1. El sistema muestra todos los sensores dados de alta.
  2. El administrador podrá dar de alta nuevos sensores. (S1)
     a) El sistema muestra un formulario solicitando los datos del sensor.
     b) El administrador introduce los datos solicitados.
     c) El sistema ofrece una lista de tipos de sensores.
     d) El administrador elige el tipo de la lista S1.1
     e) El administrador añade un nuevo tipo a la lista S1.2
        1) El sistema muestra un formulario solicitando datos.
        2) El administrador añade los datos del nuevo tipo.

3) El sistema valida los datos.

f) El sistema comprueba si los datos son correctos. (E1)

g) El sistema comprueba si el sensor no estaba ya dado de alta. (E2)

h) El sistema muestra la lista de sensores.

3. El administrador podrá borrar sensores del sistema. (S3)

a) El sistema muestra una alerta avisando de la perdida de datos.

b) El administrador confirma dicha alerta.

c) El sistema muestra los sensores restantes.

- **Excepciones:**

  E1: Los datos del sensor no son correctos, el sistema vuelve a pedir dichos datos.

  E2: El sensor ya estaba dado de alta, el sistema muestra la lista de sensores.

- **Caso de uso 4: Visualizar estado actual**

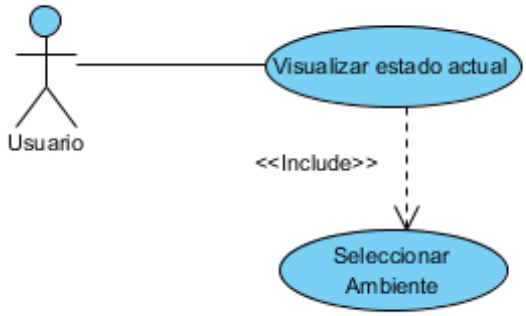

Figura 3.5: Caso de uso: Visualizar estado actual

- **Actores:** Usuario
- **Condiciones entrada:** El sistema debe de tener algún ambiente dado de alta.
- **Eventos:**
  1. El sistema muestra todos los ambientes del sistema. (E1)
  2. El usuario selecciona el ambiente que quiere monitorizar. (E2)
  3. El sistema consulta los datos de todos los objetos del ambiente.
  4. El sistema devuelve el estado actual de cada uno de los objetos.
- **Excepciones:**

  E1: El sistema no tiene ambientes dados de alta. Se muestra un mensaje de información.

  E2: El sistema no puede realizar la conexión con el ambiente. Se muestra un mensaje de información.

- **Caso de uso 5: Consultar histórico de eventos**

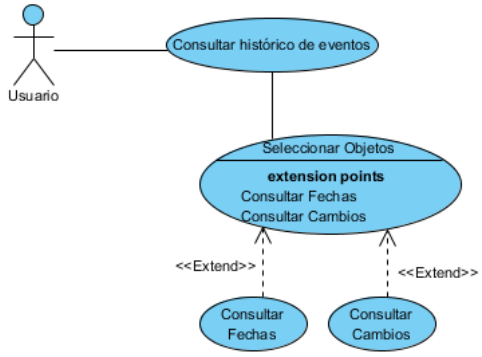

Figura 3.6: Caso de uso: Consultar histórico de eventos

- **Actores:** Usuario
- **Condiciones entrada:**

  El usuario ha debido de seleccionar previamente un ambiente.

  El sistema debe de tener objetos asociados al ambiente.

- **Eventos:**

  1. El sistema muestra un formulario con todos los objetos del sistema.
  2. El usuario elige los objetos a los que le desea aplicar la consulta.
  3. El sistema muestra un formulario con el tipo de consulta.
  4. El usuario realiza la consulta. (E1)
  5. El sistema muestra los datos asociados con la consulta del usuario.

- **Excepciones:**

  E1: Los objetos seleccionados no tienen ningún evento. Se muestra un mensaje de error.

3.3.2. Casos de uso del prototipo de aplicación móvil

- **Caso de uso 1: Visualizar estado actual**

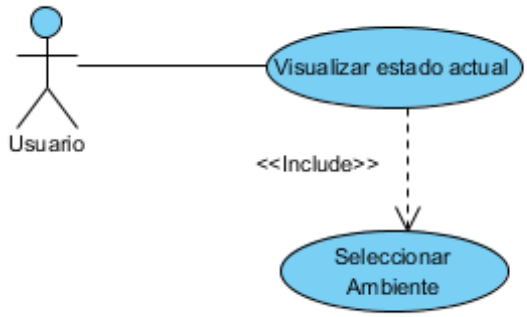

Figura 3.7: Caso de uso: Visualizar estado actual

- **Actores:** Usuario
- **Condiciones entrada:**

  El dispositivo móvil tiene salida a internet. El dispositivo móvil esta registrado en el sistema. El sistema debe de tener algún ambiente dado de alta.

- **Eventos:**

  1. El sistema muestra todos los ambientes. (E1)
  2. El usuario selecciona el ambiente que quiere monitorizar. (E2)
  3. El sistema consulta los datos de todos los objetos del ambiente.
  4. El sistema devuelve el estado actual de cada uno de los objetos y los muestra en una tabla.

- **Excepciones:**

  E1: El sistema no tiene ambientes dados de alta. Se muestra un mensaje de información.

  E2: El sistema no puede realizar la conexión con el ambiente. Se muestra un mensaje de información.

- **Caso de uso 2: Suscribirse a objetos**

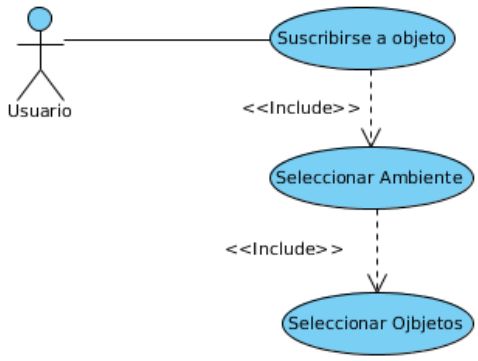

Figura 3.8: Caso de uso: Visualizar estado actual

- **Actores:** Usuario
- **Condiciones entrada:**

  El dispositivo móvil tiene salida a internet. El dispositivo móvil esta registrado en el sistema. El sistema debe de tener algún ambiente dado de alta.

- **Eventos:**

  1. El sistema muestra todos los ambientes. (E1)
  2. El usuario selecciona el ambiente que quiere monitorizar. (E2)
  3. El sistema consulta los datos de todos los objetos del ambiente.
  4. El sistema devuelve el estado actual de cada uno de los objetos y los muestra en una tabla.

- **Excepciones:**

  E1: El sistema no tiene ambientes dados de alta. Se muestra un mensaje de información.

  E2: El sistema no puede realizar la conexión con el ambiente. Se muestra un mensaje de información.

3.3.3. Escenarios del sistema web

En primer lugar, hay que especificar que los escenarios son una descripción parcial de cómo se comporta nuestro sistema en una situación concreta. Cada uno de los escenarios que vamos a elaborar, debe cumplir los siguientes elementos:

- Identificador único.
- Breve descripción.
- Actores.
- Eventos.

Al igual que en la sección anterior, los escenarios más importantes de esta aplicación son los siguientes:

- **Escenario 1: Identificarse en el sistema**
  - **Descripción:**
    El administrador Pedro quiere identificarse en el sistema para poder gestionar un ambiente inteligente que él administra.
  - **Actores:**
    Pedro, Aplicación, Servidor
  - **Eventos:**
    1. Pedro visita la web de monitorización de ambientes inteligentes.
    2. Pedro pulsa el botón *Administrador* para acceder al formulario de acceso.
    3. La aplicación carga una nueva página, incluyendo un formulario pidiendo un usuario y una contraseña.
    4. Pedro teclea sus credenciales, usuario y contraseña.

5. El servidor comprueba las credenciales comparándolas con las que tiene almacenadas en la base de datos, si estas son correctas Pedro puede entrar en el panel de administración.

- **Escenario 2: Crear un nuevo ambiente**

  - **Descripción:**

    Un nuevo administrador (Matt Farrell) quiere dar de alta un nuevo ambiente para monitorizar todos los sensores que tiene instalados dentro de este.

  - **Actores:**

    Matt Farrell, Aplicación, Servidor

  - **Eventos:**

    1. Matt pulsa el botón *Nuevo Ambiente* dentro del panel de administración.
    2. La aplicación crea una ventana modal con un formulario para que Matt pueda rellenar los datos correspondientes al nuevo ambiente.
    3. Matt completa todos los datos requeridos por el formulario y enlaza un archivo de imagen correspondiente al mapa del ambiente.
    4. El sistema comprueba que todos los campos son correctos y envía la petición al servidor.
    5. El servidor comprueba que no existe un ambiente con los mismos datos y devuelve una lista con todos los ambientes del sistema.

- **Escenario 3: Eliminar un ambiente**

  - **Descripción:**

    El Dr. Emmett quiere borrar un ambiente del sistema, ya que abandonaron el proyecto de inteligencia ambiental con el que estaba trabajando.

- **Actores:**

  Emmett, Aplicación, Servidor

- **Eventos:**

  1. Emmett dentro del panel de administración pulsa el botón borrar (Icono papelera) asociado a uno de los ambientes que él administraba.
  2. La aplicación despliega una alerta preguntándole si esta seguro de realizar esta acción, ya que es irreversible.
  3. Emmett acepta el dialogo.
  4. El sistema manda al servidor la petición de borrar el ambiente gestionado por Emmett.
  5. El servidor comprueba que existe dicho ambiente y lo eliminada de la base de datos.

- **Escenario 4: Dar de alta un sensor**

  - **Descripción:**

    El administrador Sheldon quiere dar de alta un nuevo sensor en el sistema. Este sensor aun no tiene definido ningún tipo, por lo que a su vez también deberá de crearlo.

  - **Actores:**

    Sheldon, Aplicación, Servidor

  - **Eventos:**

    1. Sheldon pulsa en el botón *Nuevo sensor* perteneciente al apartado Sensores en el panel de administración.
    2. La aplicación creará una ventana modal con un formulario, pidiéndole a Sheldon el identificador del sensor y su tipo.

3. Sheldon pulsa en el botón *Nuevo tipo*, para añadir un nuevo tipo de sensor al sistema.
4. La aplicación abre una nueva ventana modal con un formulario para la creación del nuevo tipo de sensor.
5. Sheldon completa todos los datos del formulario y acepta la ventana modal.
6. La aplicación envía la orden de crear un nuevo tipo de sensor al servidor junto con sus datos.
7. El servidor acepta la petición y devuelve una lista con todos los tipos de sensores.
8. La aplicación actualiza los tipos de sensores.
9. Sheldon completa el identificador del nuevo sensor y acepta la ventana modal.
10. La aplicación comprueba los datos introducidos por el administrador Sheldon y los envía al servidor.
11. El servidor comprueba que el sensor no este ya dado de alta y devuelve una lista con todos los sensores actualizada.

- **Escenario 5: Consultar el estado actual de un ambiente**

  - **Descripción:**

    Paul quiere consultar el estado de un ambiente donde están realizado prácticas de inteligencia ambiental.

  - **Actores:**

    Paul, Aplicación, Servidor

  - **Eventos:**

    1. Paul abre la aplicación web en su navegador y selecciona el ambiente que quiere consultar.

2. La aplicación consulta en el servidor el estado actual del ambiente.
3. El servidor comprueba todos los objetos asociados a un ambiente y devuelve el estado actual de cada uno de ellos.
4. La aplicación convierte los datos obtenidos por el servidor y los muestra en la vista para el usuario Paul.

- **Escenario 6: Consultar los eventos en un rango de fechas**

  - **Descripción:**

    Juan quiere consultar los eventos que ocurrieron en su ambiente desde el 20 de enero hasta el día 25 del mismo mes.

  - **Actores:**

    Juan, Aplicación, Servidor

  - **Eventos:**

    1. Juan pulsa el botón *Eventos* dentro de la aplicación web principal.
    2. La aplicación web carga una nueva página con un formulario asociado para realizar la consulta.
    3. Juan selecciona todos los objetos que aparecen en el formulario, selecciona consulta por fecha e introduce la indicada en el titulo.
    4. El sistema envía al servidor la petición, junto con los objetos y fechas seleccionadas en el formulario.
    5. El servidor comprueba si hay eventos asociados a los objetos en esa fecha y devuelve una lista con cada uno de ellos.
    6. El sistema convierte los datos recibidos del servidor a la vista del usuario.

3.3.4. Escenarios del prototipo de aplicación móvil

Los escenarios más importantes para nuestro prototipo de aplicación móvil son:

- **Escenario 1: Registrar un nuevo dispositivo**

  - **Descripción:**

    Jorge se ha comprado un nuevo móvil y quiere monitorizar el estado en el que se encuentra el ambiente que administra.

  - **Actores:**

    Jorge, Aplicación Móvil, Servidor

  - **Eventos:**

    1. Juan instala la aplicación en su nuevo dispositivo, y la ejecuta.
    2. La aplicación se intenta conectar al servidor enviando a su vez la ID única asociada a su dispositivo.
    3. El servidor comprueba si la ID corresponde a alguna ya dada de alta en el sistema, si no esta dada de alta la registra, si ya lo estaba de antes devuelve una lista de objetos a los que esta suscrito ese usuario.
    4. Jorge pulsa en el menú el botón *Estado Actual*.
    5. La aplicación móvil realiza la petición al servidor y este devuelve una lista de objetos junto con los últimos valores asociados.

- **Escenario 2: Suscribirse a nuevos objetos**

  - **Descripción:**

    Pablo quiere realizar una subscripción a varios objetos de un ambiente, de este modo sabrá a tiempo real que es lo que sucede en dicho ambiente.

  - **Actores:**

    Jorge, Aplicación Móvil, Servidor

  - **Eventos:**

    1. Pablo pulsa el botón Suscripciones dentro el menú de la aplicación móvil
    2. La aplicación móvil muestra los objetos a los que Pablo esta suscrito.
    3. Pablo marca las casillas de cada uno de los objetos deseados y confirma dicha acción.
    4. La aplicación móvil envía a servidor dicha petición,
    5. El servidor guarda los objetos que a los que el usuario esta suscrito.

3.4. Diseño del sistema

El diseño del sistema es la tarea donde se identifican los objetivos finales del sistema y se plantean las diversas estrategias para alcanzarlos en la actividad de implementación [24].

Este diseño, lo dividiremos en dos partes, diseño de clases y diseño de datos.

3.4.1. Diseño de clases

Como se ha especificado a lo largo de esta memoria, nuestra aplicación utiliza una arquitectura cliente-servidor, por lo tanto a la hora de detallar el proceso de diseño de clases, lo haremos diferenciando cada una de las partes de esta arquitectura.

Sistema web y prototipo de aplicación móvil: El cliente utilizará un esquema de diseño MVC (Modelo-Vista-Controlador) (ver figura 3.9) [25]. La importancia de este esquema, es mantener el mayor grado de independencia posible entre el modelo de la vista y el controlador.

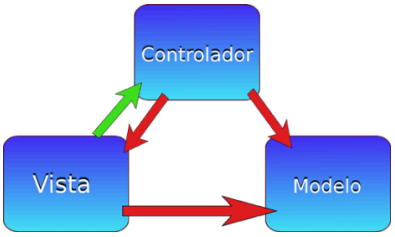

Figura 3.9: Modelo-Vista-Controlador

- **Modelo:** El modelo es el conjunto de datos ofrecido por el servidor, que en este caso, estará en formato JSON.
- **Vista:** Al tratarse de una aplicación WEB, las vistas son todas las clases donde se mostrarán los datos del modelo. Cada una de las diferentes páginas HTML.

- **Controlador:** El controlador se encargará de dirigir el tráfico entre el usuario y la aplicación. Es la parte vital de este esquema, ya que se encarga de realizar las peticiones del servidor y traducirlas para que la vista pueda interpretarlas. En este trabajo fin de grado los controladores están diferenciados para la parte de administración y principal. Con el objetivo de simplificar su dificultad se ha utilizado AngularJS.

Servidor: Aquí se realizará una explicación más detallada, ya que es la parte vital de nuestro trabajo fin de grado. Un buen diseño en la misma hará que el sistema sea más seguro y eficaz.

Debido a la gran cantidad de clases con las que nos encontramos en nuestro servidor, se van agrupar en tres paquetes, con el objetivo de visualizar de una manera clara las relaciones entre las clases y los paquetes y, dando como resultado final, el siguiente diagrama de paquetes (ver figura 3.10):

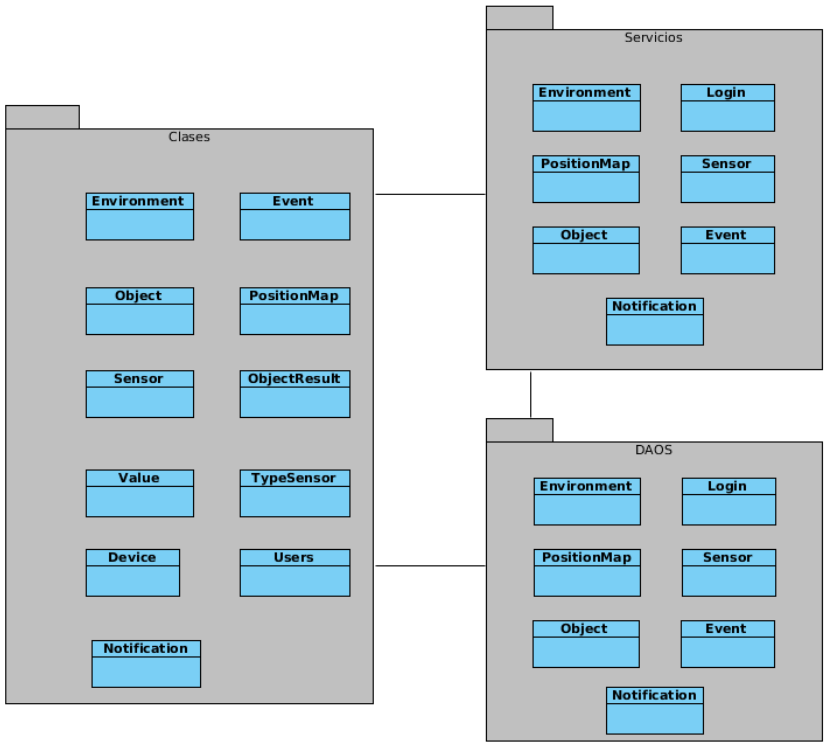

Figura 3.10: Diagrama de paquetes

A continuación, en la figura 3.11 se de talla el diagrama de clases sin establecer una asociación con los servicios o DAOS.

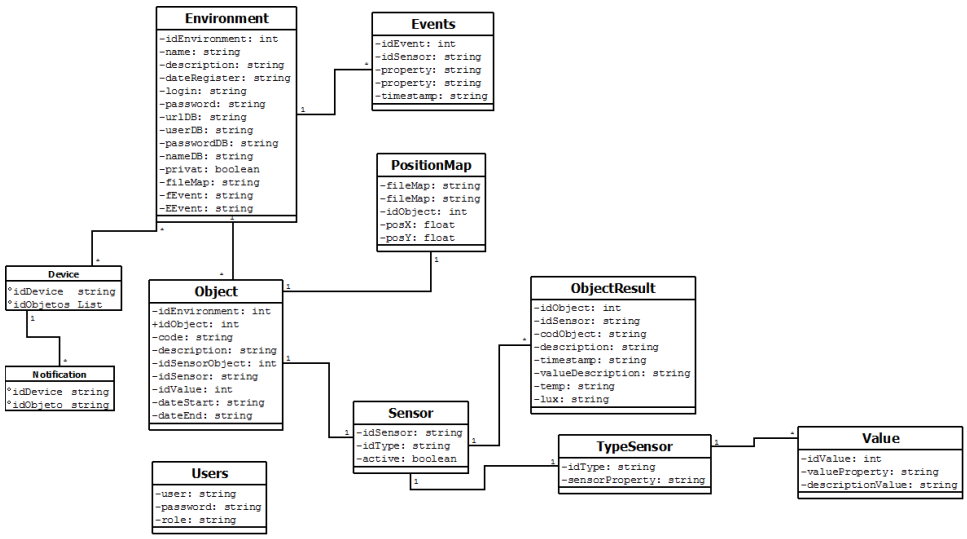

Figura 3.11: Diagrama de Clases

Cada uno de estos cuenta con las siguientes clases, atributos y operaciones ver figuras desde 3.12 a 3.14:

- **Paquete Clases:**

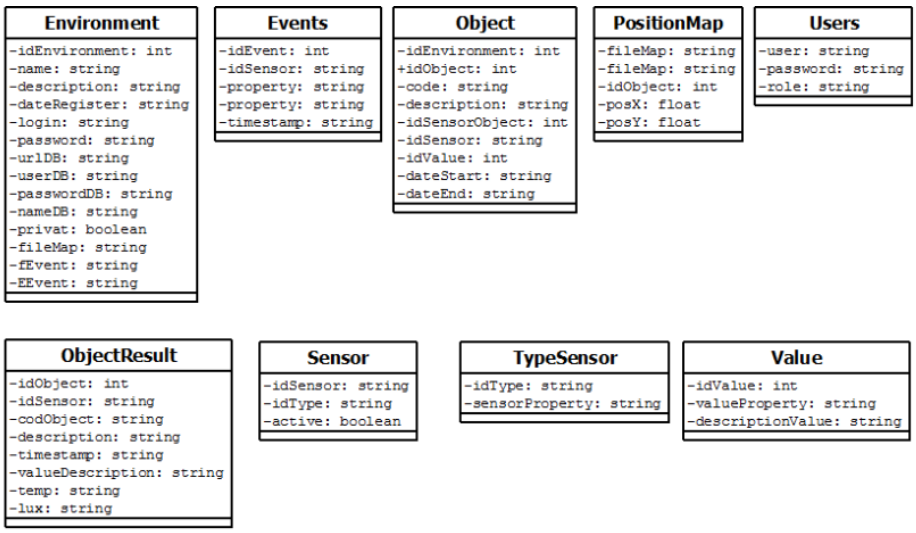

Figura 3.12: Clases

- **Paquete Servicios:**

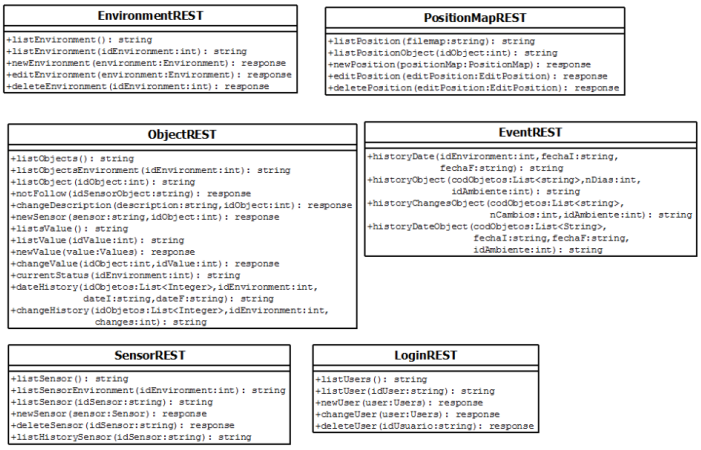

Figura 3.13: Servicios

- **Paquete DAOS:**

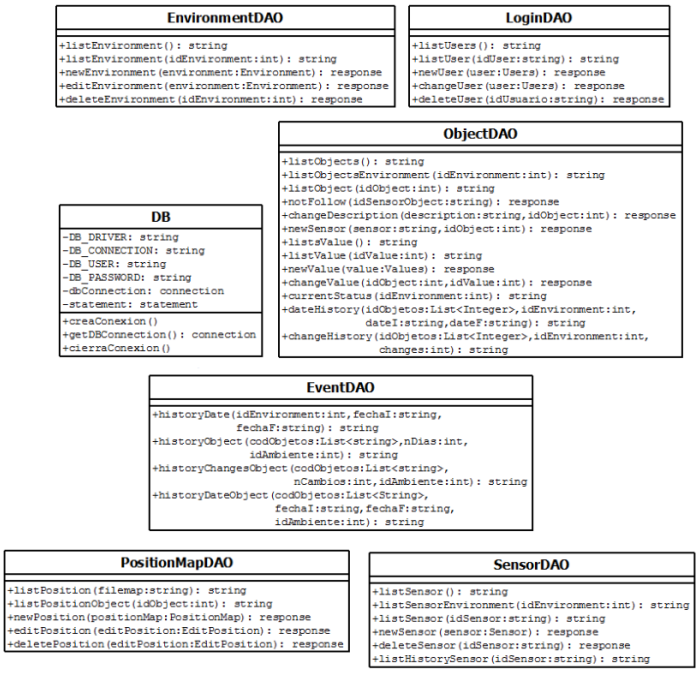

Figura 3.14: DAOS

3.4.2. Diseño de datos

La idea principal de esta fase, dentro del diseño de software, es determinar qué estructura se utilizará para realizar la persistencia de todos los datos que serán de vital importancia para el uso de nuestra aplicación.

Antes de entrar en detalle, mencionaremos que el diseño de datos ha sido dividido en dos paquetes (ver figura 3.15), los cuales trabajaran juntos para proporcionar significado a nuestros servicios de aplicación.

Paquetes:

Figura 3.15: Paquetes diseño de datos

- **Eventos:** Es la parte más importante de nuestra aplicación. Este paquete contiene una sola tabla, y en ella reside toda la información enviada por los sensores de cada uno de los ambientes que monitorizará la aplicación.

- **Sistema:** Este paquete será el encargado de traducir toda la información proporcionada por los sensores a un lenguaje natural. De este modo, al usuario le resultará más fácil interpretar todos los datos. Además, en este reside toda la lógica de la aplicación y servicios.

Modelo entidad-relación Es una herramienta, conocida normalmente por las iniciales E-R, utilizada para el modelado de datos que permite representar las tablas relevantes de un sistema, así como sus relaciones y atributos. El modelo de datos entidad-relación se asemeja al lenguaje que utilizamos para comunicarnos y está provisto de una serie de objetos que llamaremos entidades. Cada una de

estas entidades se relacionará para crear entre ellos un modelo donde realizar la persistencia de todos los datos.

Además de las entidades, relaciones y atributos, es importante destacar otros conceptos que serán utilizados en este modelo.

- **Cardinalidad:** Puede ser de tres tipos, cada uno de estos corresponde con el número de solicitudes implicadas en cada entidad. Para entenderlo con una mayor facilidad vamos a suponer que tenemos dos entidades, A y B.
  - **Uno a muchos:** A corresponde con una o más solicitudes de B. Se representa mediante (1:*)
  - **Muchos a muchos:** Cualquiera de ambas estancias se relacionan entre sí. Representada por (*:*)
  - **Uno a uno:** Tan solo una sola instancia de A se relaciona con B.
- **Claves:** Pueden estar formadas por uno o más atributos, el principal uso de estas claves es la unión entre distintas entidades.

En la siguiente figura 3.16 se mostrará el esquema conceptual del sistema web de este trabajo.

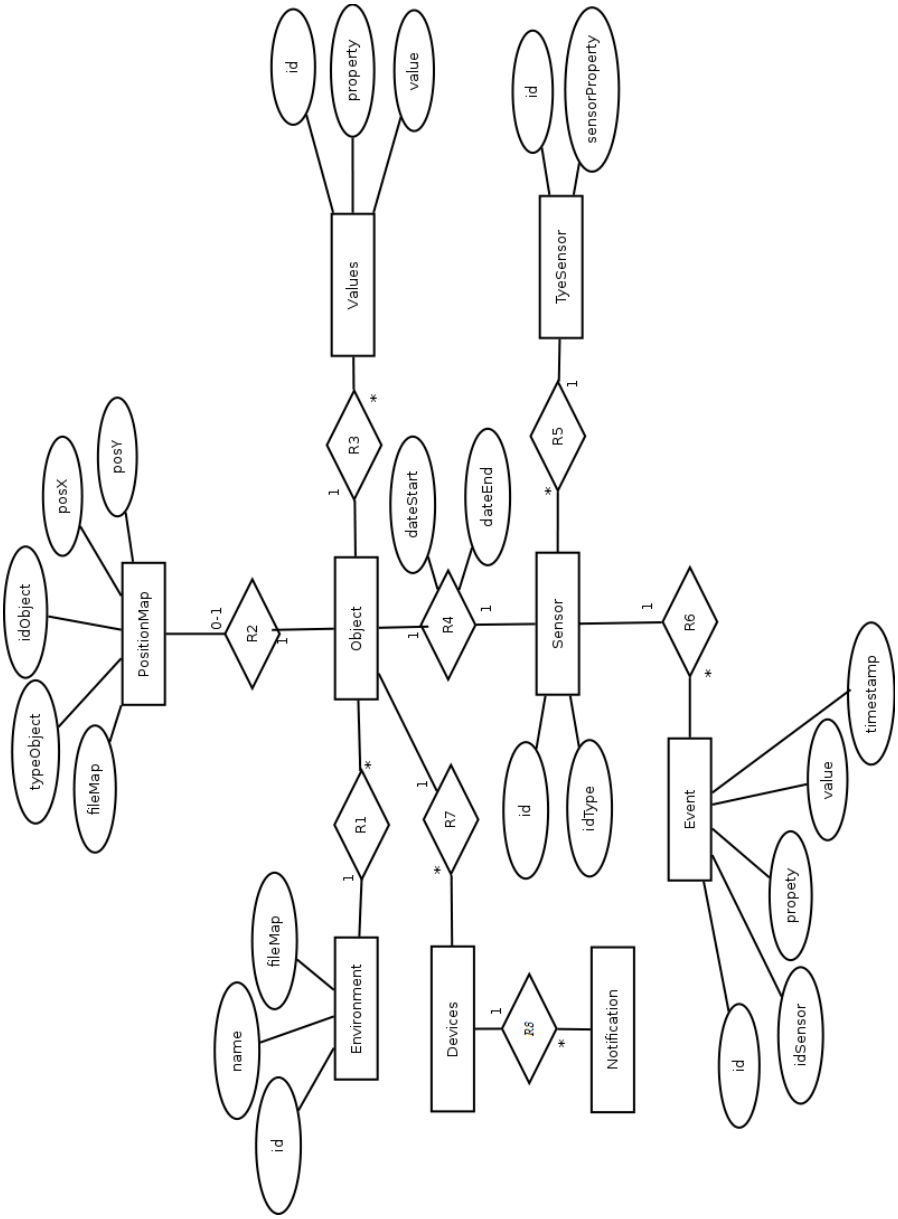

Figura 3.16: Esquema Conceptual

- **R1:** A un ambiente le pertenecen muchos objetos.
- **R2:** Un objeto puede tener una o ninguna sola posición en el mapa
- **R3:** En un objeto se definen uno o varios valores.
- **R4:** Un objeto pertenece a un único sensor.
- **R5:** Un tipo de sensor, puede tener muchos sensores.
- **R6:** Un sensor enviará múltiples eventos.
- **R7:** Un objeto puede estar en muchos dispositivos móviles
- **R8:** Un dispositivo móvil puede tener muchas notificaciones. sensores.

Una vez entendido este punto, debemos realizar una nueva tarea. Normalizar el modelo entidad-relación. De este modo, evitaremos la redundancia de datos, ahorrando espacio y haciendo las consultas mucho más eficientes. Vamos a detallar las diferentes formas de normalización, hasta llegar a un nivel 3. Llamado **Tercera forma normal (FN3)**.

Primera forma normal (FN1): Una tabla está en primera forma normal si todos los atributos, dependen de la clave primaria.

Segunda forma normal (FN2): Una tabla está en segunda forma normal si además de estar en Primera forma normal, todos los atributos que no pertenecen a la clave dependen de ésta.

Tercera forma normal (FN3): Una tabla está en tercera forma normal si además de estar en Segunda forma normal, no existen atributos que dependan transitivamente de la clave.

Esquema Conceptual Modificado Para obtener el esquema conceptual modificado (ver figura 3.17), debemos eliminar todas las entidades débiles y las relaciones muchos a muchos. Realizando estas tareas obtenemos el siguiente esquema:

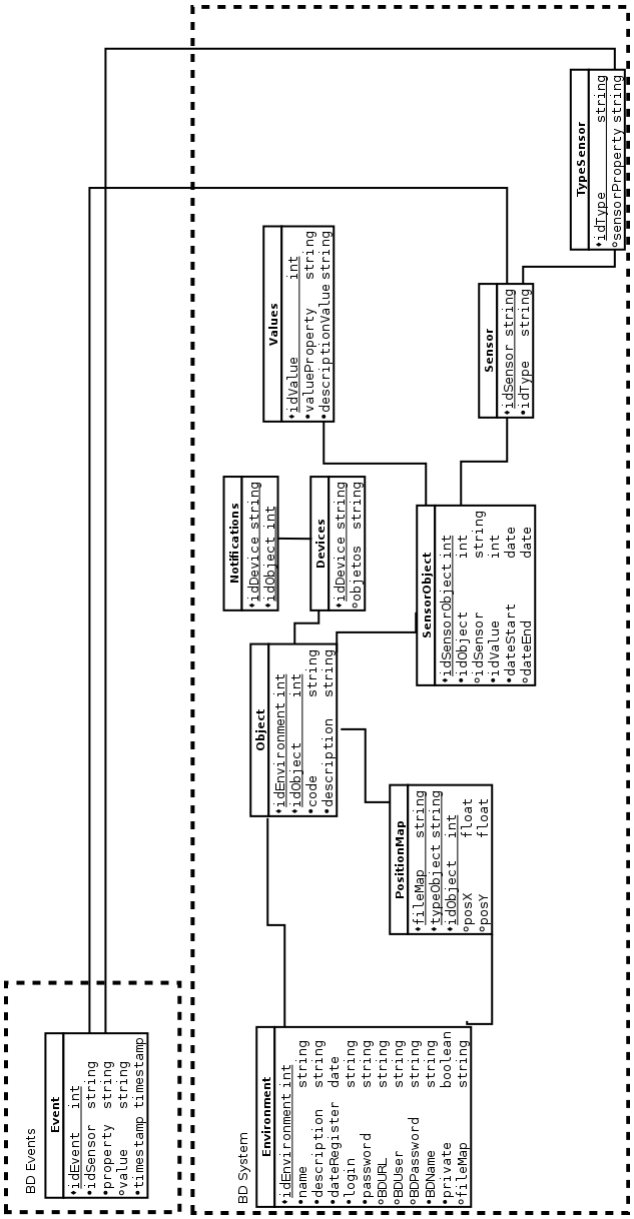

Figura 3.17: Esquema Conceptual

Tablas Una vez obtenido el esquema conceptual modificado, podemos instanciar las tablas que se utilizarán en nuestra base de datos, quedando como resultado las siguientes tablas:

Events: Esta tabla (ver figura 3.18) se encuentra dentro del paquete Eventos, como se mencionó anteriormente en ella se almacenará toda la información que los sensores recogen continuamente en el ambiente. Al ser una tabla con un volumen de datos muy alto, se deben de utilizar técnicas para consultar los datos de una manera eficiente.

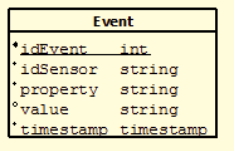

Figura 3.18: Tabla Events

Atributos:

Atributo	Descripción
idEvents	Identificador numérico auto-incremental para identificar cada uno de los eventos.
idSensor	Identificador único del sensor (Mac, ID)
property	Propiedad valor del sensor(Estado, Luminosidad, Temperatura...)
value	Valor de una propiedad.
timestamp	Marca de tiempo

Tabla 3.1: Tabla Events

Environment: En esta tabla (ver figura 3.19) se guardará toda la información importante del ambiente que vamos a monitorizar, obteniendo tres grandes grupos de datos: La información y descripción del ambiente, un estado que define si el ambiente es público o privado, junto con sus datos de acceso, y toda la información para conectarnos a la base de datos de los eventos generados por los sensores.

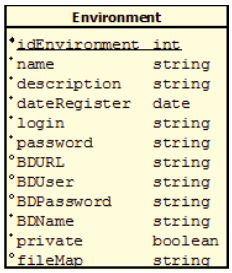

Figura 3.19: Tabla Environment

Atributos:

Atributo	Descripción
idEnvironment	Identificador numérico auto-incremental para identificar un entorno de inteligencia ambiental.
name	Nombre del entorno de inteligencia ambiental.
description	Descripción corta del entorno de inteligencia ambiental.
dateRegister	Fecha en la que se da de alta el entorno de inteligencia ambiental.
login	Usuario principal el entorno de inteligencia ambiental.
password	Contraseña del entorno de inteligencia ambiental.
BDURL	Dirección de la base de datos del entorno de inteligencia ambiental.
BDUser	Usuario para conectar a la base de datos.
BDPassword	Contraseña para conectar a la base de datos.
BDName	Nombre de la base de datos.
private	Seguridad del entorno de inteligencia ambiental.(Privado o público)
fileMap	Archivo de imagen correspondiente al plano del entorno de inteligencia ambiental.

Tabla 3.2: Tabla Evironment

Sensors y Typesensor: Para entender mejor el funcionamiento de estas tablas (ver figura 3.20) es mejor explicarlas en conjunto. Gracias a ellas, todos los sensores serán registrados en el sistema. Además, por cada uno de los tipos de sensores tendrá una información asociada, de este modo sabremos que tipo de datos esta recogiendo cada uno de estos sensores. Por este motivo, antes de dar de alta un nuevo sensor, debemos definir a que tipo pertenece y que valores permitiremos obtener.

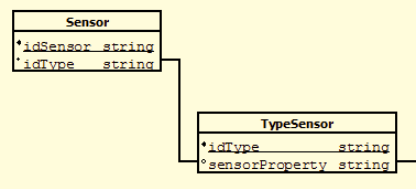

Figura 3.20: Tabla Sensors y Typesensor

Atributos:

Atributo	Descripción
idSensor	Identificador único del sensor
idType	Identificador correspondiente al tipo de sensor. Ej:Tynetec, Sunspot..
sensorProperty	Propiedades que medirá el sensor (Temperatura, Luminosidad...)

Tabla 3.3: Sensors y Typesensor

PositionMap: Gracias a esta tabla (ver figura 3.21), podemos situar todos los sensores en el plano asociado al ambiente al que pertenecen.

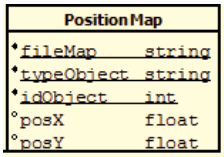

Figura 3.21: Tabla PositionMap

Atributos:

Atributo	Descripción
fileMap	UUID del archivo de imagen correspondiente al plano del entorno de inteligencia ambiental
typeObject	Tipo de objeto que a representar en el mapa. (Actividad, Objetos...)
idObject	Identificador correspondiente al objeto.
posX	Coordenada X de posición.
posY	Coordenada Y de posición.

Tabla 3.4: PositionMap

Object y SensorObject: Table Object (ver figura 3.22): Define un objeto y su descripción. Además, esta tabla se encarga de asociar el objeto con un ambiente.

Table SensorObject: Tabla muy importante en nuestro sistema que nos sirve de unión entre un sensor y un objeto. Aprovechando esta unión, utilizaremos esta tabla para saber si el objeto se encuentra activo en el sistema. Para realizar esta comprobación solo debemos consultar el atributo DateEnd y, siempre que este sea NULO, significa que aún está siendo utilizado ese objeto en el ambiente.

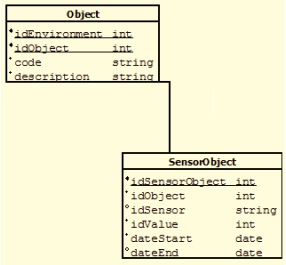

Figura 3.22: Tabla Object

Atributos:

Atributo	Descripción
IdEnvironment	Identificador del entorno de inteligencia ambiental donde se añadirá el objeto
IdObject	Identificador numérico auto-incremental para identificar un objeto
Code	Identificador corto(3 letras) para nombrar un objeto
Description	Descripción de un objeto
IdSensorObject	Identificador numérico correspondiente al SensorObjeto
IdSensor	Identificaor del sensor
IdValue	Identificador del valor
DateStart	Fecha de registro
DateEnd	Fecha de finalización

Tabla 3.5: Sensor y Object

Values: Es una tabla (ver figura 3.23) muy sencilla, contiene la información necesaria para traducir a un lenguaje natural cada uno de los datos que han sido enviados por un tipo de sensor.

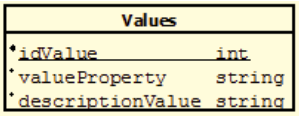

Figura 3.23: Tabla Values

Atributos:

Atributo	Descripción
idValue	Identificador numérico auto-incremental para identificar un valor
valueProperty	Propiedad a traducir (Estado, NFC, Manipulación...)
descriptionValue	Descripción de un valor

Tabla 3.6: Tabla Values

Device y Notifications: Por último, nos quedan las dos tablas utilizadas para el prototipo de aplicación móvil. Estas tablas se centrarán en registrar cada dispositivo al sistema junto con los objetos que esta suscrito y guardar cada uno de los eventos que el interactuaron con cada objeto.

Atributos:

Atributo	Descripción
idDevice	Identificador único de cada dispositivo
objetos	Identificador del objeto al que esta suscrito

Tabla 3.7: Tabla Device y Notifications

3.4.3. Diseño de la interfaz

En esta fase definiremos cómo será la apariencia visual que tendrá la parte cliente de nuestra aplicación. Es muy importante implementar una buena arquitectura en nuestro sistema para que éste sea eficiente pero, por otro lado, el usuario final es quien estará delante de nuestra aplicación, por lo que si ésta no tiene un buen diseño, puede llegar a fracasar.

Para realizar el diseño de la interfaz, debemos de pensar cuáles son todos los puntos donde el usuario interactuará con la aplicación: estilo, mensajes y metáforas.

3.4.3.1. Estilo

En este punto al tratarse de una aplicación web, vamos a definir una guía de estilo para que todos los elementos de la interfaz tengan coherencia y consistencia.

Una **Guía de estilo**, es una lista de reglas y elementos gráficos que los desarrolladores web deben seguir para crear una experiencia cohesiva al final de la aplicación diseñada. Los puntos en la guía de estilos son los siguientes:

- **Tipografía:** El diseñador Oliver Reichenstein, asegura que la tipografía es el 95 por ciento del diseño web [26]. Una buena elección de esta tipografía es uno de los principales factores para que el usuario final encuentre apropiado el contenido que lee.

  En nuestra aplicación web se ha utilizado la siguiente tipografía para el cuerpo:

  - **Fuente:** Helvetica Neue
  - **Tamaño de letra:** 14px
  - **Interlineado:** 1.428

En cambio, en la aplicación móvil el tamaño se ha adaptado al medio donde se esta visualizando.

- **Fuente:** Arial
- **Tamaño de letra:** 14px
- **Interlineado:** 1.6

- **Colores:** En este proceso decidimos utilizar una paleta (ver figura 3.24) de color proporcionada por la aplicación Color de la empresa Adobe. Gracias a esta paleta, nos aseguramos que los colores combinan entre ellos, creando una armonía en el diseño.

Figura 3.24: Paleta de colores

- **Iconos:** En este apartado se ha utilizado una galería de iconos en formato texto. Esta galería se cargará antes de visualizar el contenido de la web. Gracias a los iconos, los usuarios finales podrán tener una idea clara de que simboliza cada una de las acciones con las que están asociados.

  - **Galería de iconos:** Font Awesome 4.5.0

- **Botones:** Los botones son una mezcla de paletas de colores y formas, todos los botones utilizados son proporcionados por el framework de diseño *Bootstrap*, pero se han definido unas reglas con el color de estos (ver figura 3.25).

Figura 3.25: Botones

3.4.3.2. Mensajes

Los mensajes se utilizarán para indicar al usuario advertencias o situaciones de error. En la mayoría de las ocasiones, estos mensajes vienen dados por un fallo de entendimiento a la hora de utilizar la aplicación, pero también puede darse el caso de que el sistema funcione de manera anómala.

Para expresar todos estos mensajes, es importante utilizar las siguientes tres reglas.

- **Mensaje especifico:** No todos los errores o advertencias que provoque el sistema deben ser iguales, por lo que es importante ser específicos para que el usuario final entienda que esta ocurriendo en el sistema.
- **Breve:** Si extendemos el mensaje, aumentamos su dificultad, buscando que el mensaje sea breve y especifico, sin dejar información que pueda ser de utilidad.
- **Denotación Positiva:** Muchos usuarios pueden tener sensación de culpabilidad cuando el sistema avisa de un error. Para que esto no ocurra, usaremos una denotación positiva e indicaremos qué pasos deben seguir para que no vuelva a darse esa situación de error.

Además de estos tres puntos, se han definido unos estilos de colores para agrupar los mensajes: Rojo, para mensajes de error; y amarillo, para advertencias. A continuación en las figuras 3.26 a 3.28 mostramos unos ejemplos de mensajes del sistema:

- **Usuario o contraseña incorrecta:**

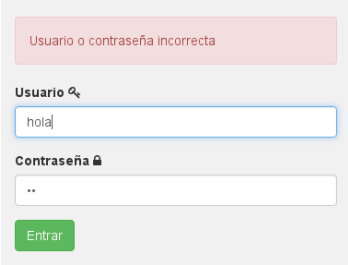

Figura 3.26: Mensaje usuario o contraseña incorrecta

Este mensaje es a causa de que el usuario o la contraseña no es la que esta dada de alta en el sistema, se utiliza el rojo para indicar que es un error.

- **Fecha incorrecta:**

Figura 3.27: Mensaje fecha incorrecta

En este caso, el usuario a introducido una fecha anterior a cuando se dio de alta el entorno de inteligencia ambiental, por lo que en ese rango no podría realizar la consulta.

- **Información consulta:**

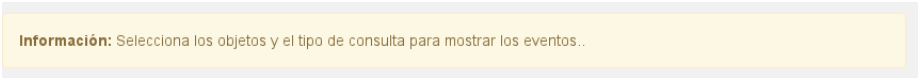

Figura 3.28: Mensaje información consulta

Aquí el usuario no ha tenido que interactuar con el sistema para que aparezca el mensaje, tan solo se le advierte que debe de hacer para poder utilizar el sistema. Para representarlo se ha utilizado el color amarillo.

3.4.3.3. Metáforas

La R.A.E. nos define una metáfora como: *Aplicación de una palabra o de una expresión a un objeto, a un concepto, al cual no denota literalmente, con el fin de sugerir una comparación (con otro objeto o concepto) y facilitar su comprensión.*

Extrapolando está definición a nuestro diseño, representaremos muchas funcionalidades de nuestra aplicación mediante estas metáforas. En nuestro caso, la mayoría de estas metáforas se representan mediante una serie de iconos que harán más sencillo el manejo de la aplicación.

Las metáforas más importantes utilizadas son:

- **Edición de ambientes:**

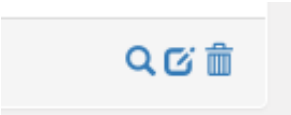

Figura 3.29: Metáfora edición

Gracias a esta metáfora (ver figura 3.29), podemos tener tres acciones sin necesidad de introducir texto. Cada uno de estos botones tiene asociado una definición: Buscar, Editar y Borrar.

- **Reproducir:**

Figura 3.30: Metáfora reproducir

Está metáfora (ver figura 3.30) viene influenciada por los años 80 cuando, en la mayoría de los dispositivos la utilizaban para reproducir una película, vídeo VHS o vinilo. En nuestro caso la utilizamos para reproducir como se encontraba el ambiente un día concreto.

- **Objeto Ambiental:**

Figura 3.31: Metáfora objeto ambiental

En este caso (ver figura 3.31) tenemos una combinación de metáforas utilizadas para representar mucha información con un simple golpe de vista. Como se ha mencionado en el punto 2.4.3 los sensores ambientales nos calculan la temperatura luminosidad y movimiento. Para representar cada uno de estos valores se ha utilizado el icono correspondiente con *Encendido/Apagado*, una nube representado la temperatura, y un sol representado luminosidad. Para poder saber el grado de cada uno de ellos se ha utilizado una gama de colores asociadas a un intervalo.

- **Objeto Manipulación:**

Figura 3.32: Metáfora objeto manipulación

Es muy similar a la anterior, pero en este caso solo se utilizará el icono *Encendido/Apagado* (ver figura 3.32) para representar si hubo manipulación en el objeto asociado al icono. Utilizaremos los colores rojo, indicando no, y verde, para el sí.

3.4.3.4. Storyboard del sistema web

Para entender que es un storyboard, debemos retroceder al año 1940, cuando este sistema se hizo muy popular en la producción de películas animadas. Tiene un funcionamiento muy simple, en el que se han de crear una serie de ilustraciones, cada una de ellas representando una acción asociada a una historia realizada por el usuario.

Estas ilustraciones representarán una secuencia similar a cómo se va a comportar la aplicación web, y cuáles son las alternativas que tenemos al realizar una acción.

- **Panel Principal:** En este storyboard (ver figura 3.33), vamos a representar como será la vista al entrar en la página principal, la cual nos representará los distintos ambientes dados de alta en el sistema. Por cada uno de los ambientes podemos realizar tres acciones: ver el estado actual, consultar el histórico del ambiente y, por último, mostrar todos los eventos que han ocurridos desde que éste se dio de alta.

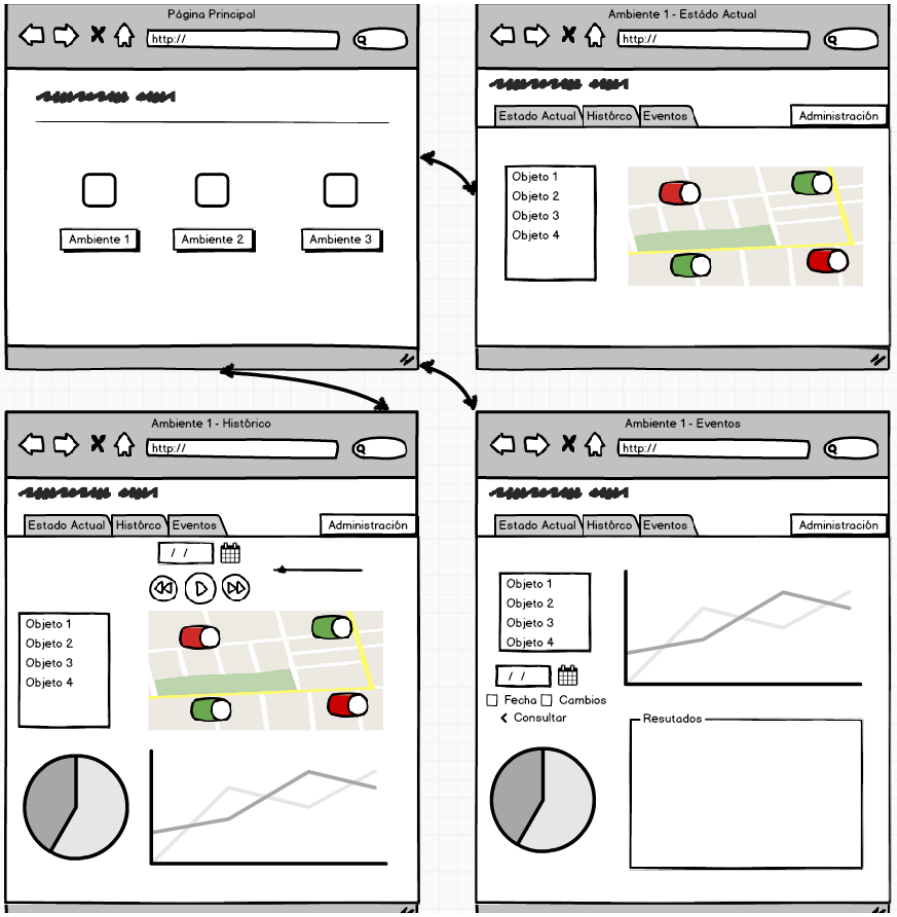

Figura 3.33: StoryBoard Panel principal

- **Panel Administración:** En las figuras 3.34, a 3.37 se van a plasmar las secuencias más importantes que tendrá la administración del sistema.

  - **Ambientes:**

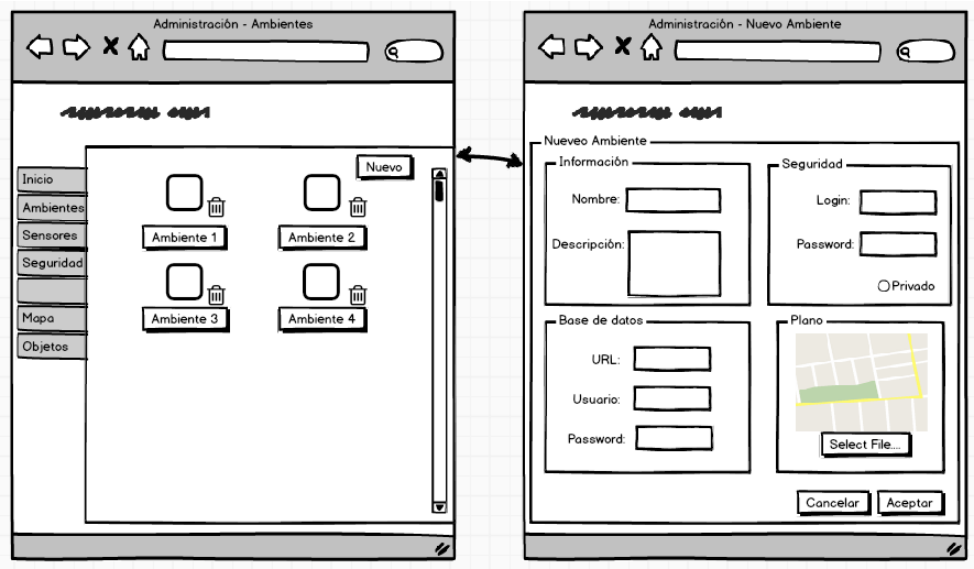

Figura 3.34: StoryBoard Ambientes

  - **Registra Sensor:**

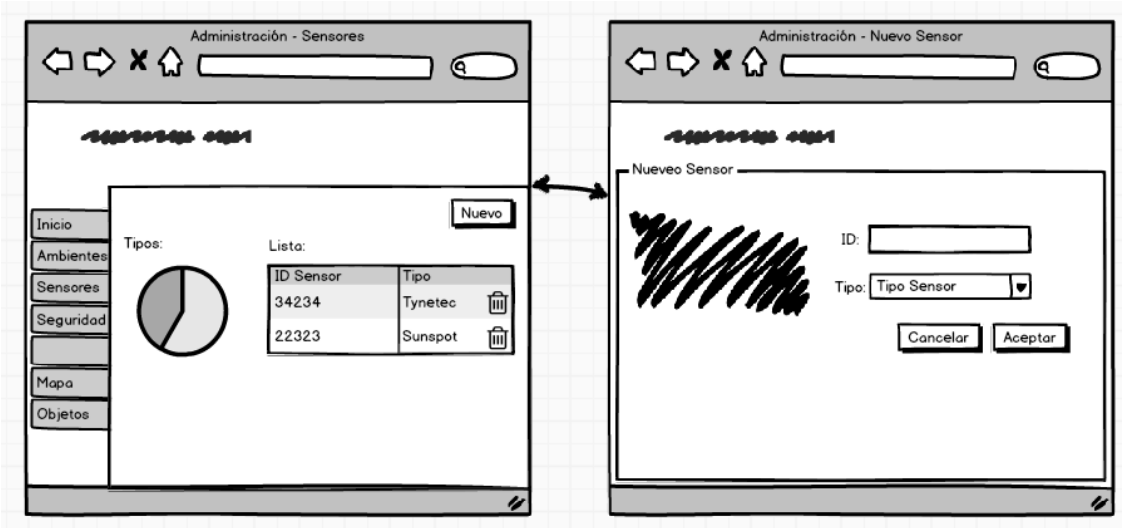

Figura 3.35: StoryBoard Sensores

- **Seguridad:**

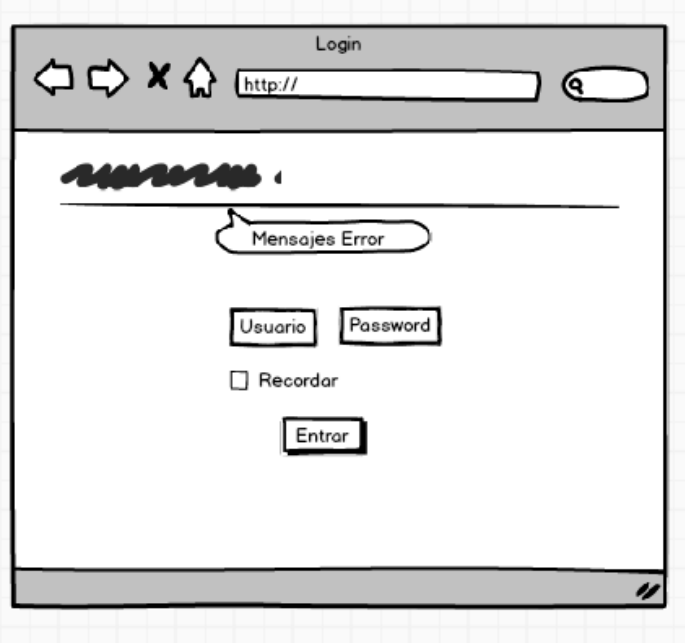

Figura 3.36: StoryBoard Página Acceso

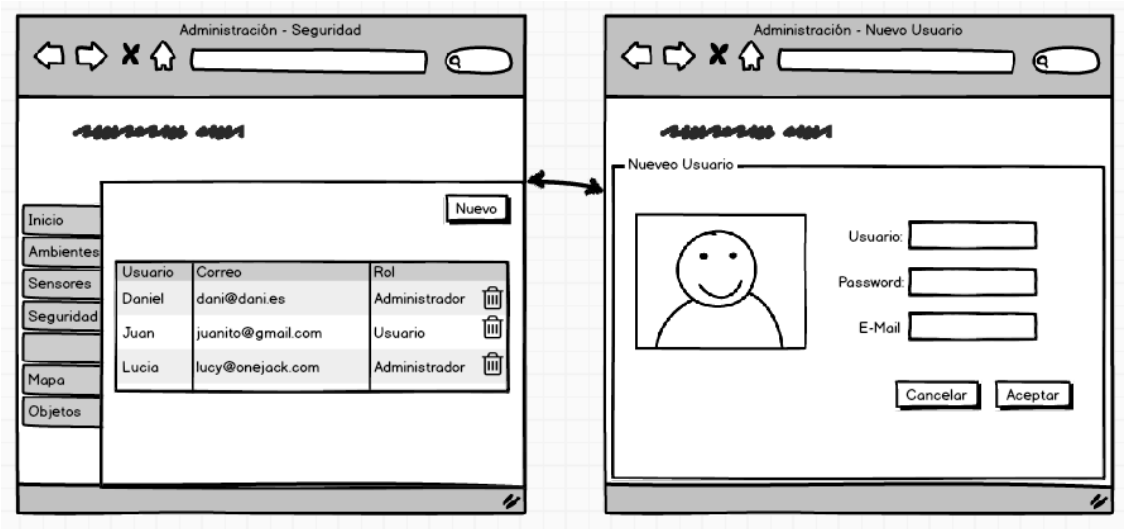

Figura 3.37: StoryBoard Añadir usuarios

- **Mapa:**

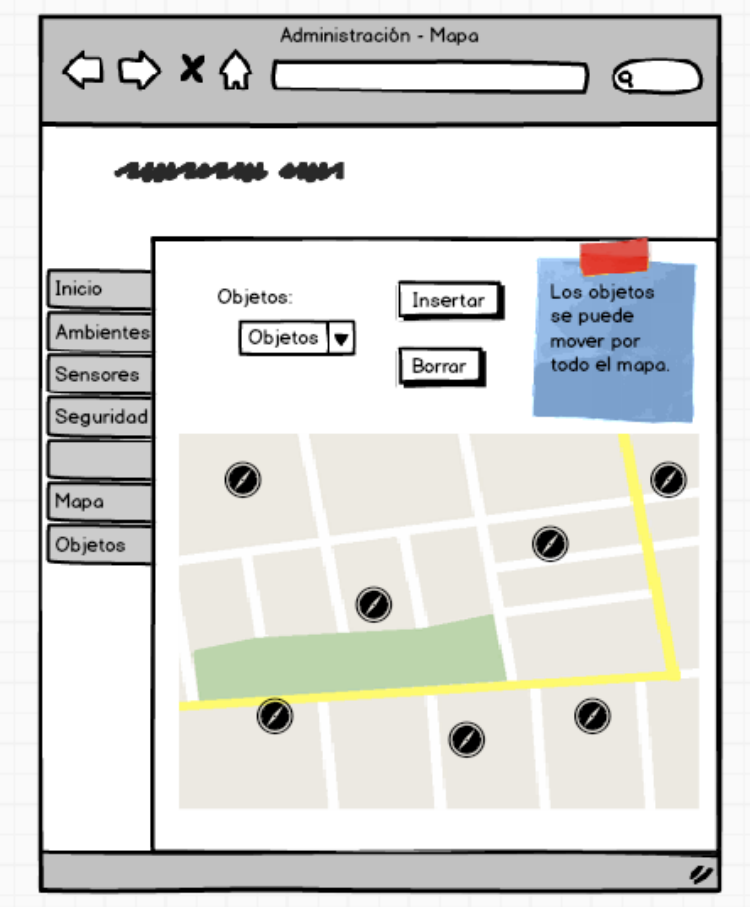

Figura 3.38: StoryBoard Mapa

3.4.3.5. Storyboard del prototipo de aplicación móvil

A continuación en las figuras, 3.39, a 3.42 se va a presentar la historia de funcionamiento del prototipo de aplicación móvil

- **Vista principal:**

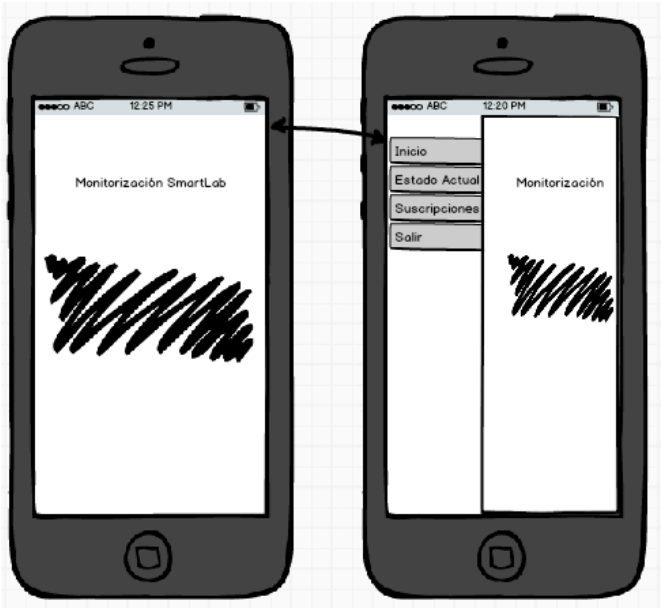

Figura 3.39: StoryBoard prototipo de aplicación móvil: Principal

- **Estado actual:**

Figura 3.40: StoryBoard prototipo de aplicación móvil: Actual

- **Suscripciones:**

Figura 3.41: StoryBoard prototipo de aplicación móvil: Suscripciones

- **Notificaciones:**

Figura 3.42: StoryBoard prototipo de aplicación móvil: Notificaciones

3.5. Implementación del sistema

3.5.1. Arquitectura del sistema web

Ya que conocemos todo el proceso de Ingeniería, solo nos queda una última actividad antes de llegar a las pruebas. Esta actividad será la que más tiempo nos lleve, pero una correcta elaboración será trivial para ahorrar costes y tiempos en el futuro. Es importante realizar un minucioso estudio sobre cuál es el mejor lenguaje de programación que se adapta a nuestro trabajo fin de grado.

En este trabajo fin de grado, se ha utilizado una arquitectura cliente-servidor, esta arquitectura, se puede simplificar con el esquema proporcionado en la figura 3.43.

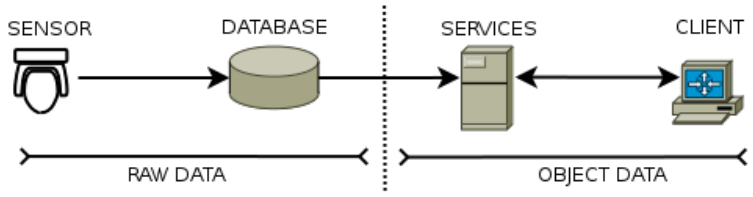

Figura 3.43: Esquema del sistema

En la parte que más nos centraremos en este apartado será la arquitectura usada en *OBJECT DATA* (Servidor/Cliente), para esto, se despliega nuestra aplicación en un servidor central, este servidor ofrecerá una API a través de servicios web.

La definición proporcionada por el World Wide Web Consortium (W3C) [27] sobre los servicios web explica que son un sistema software diseñado para soportar una comunicación entre diferentes equipos en red. Estos servicios son accedidos desde la red mediante un protocolo especifico y ejecutados en los equipos que los alojan, cumpliendo una función determinada. La implementación utilizada para el servidor es REST se resume en la siguiente figura 3.44.

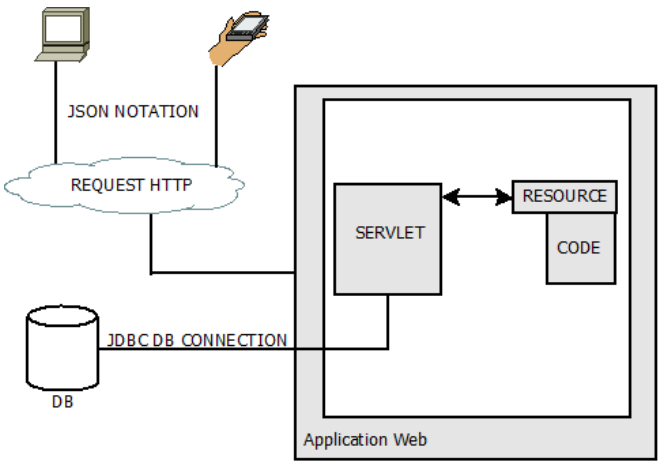

Figura 3.44: Esquema REST

El protocolo REST (Representational State Transfer), propone una arquitec-

tura cliente-servidor donde cada uno de los servicios actúa como un recurso. Para identificarlos se utiliza una URL con unos parámetros que especifican que servicio se va a producir o consumir. Para establecer la comunicación entre el cliente y el servidor, podemos guiarnos por la figura 3.44, donde cada cliente envía una serie de mensaje en formato JSON a través del protocolo HTTP, haciendo uso de los cuatro métodos disponibles en dicho protocolo (GET, PUT, DELETE y POST).

- **GET:** Leer
- **POST:** Crear
- **PUT:** Actualizar
- **DELETE:** Borrar

Como se mencionó a modo de introducción, en el intercambio de comunicación entre el cliente y el servidor se utilizará un lenguaje ligero basado en texto. Para realizar esto, utilizamos JSON, el cual es un lenguaje cuya sintaxis está basada en un subconjunto de funcionalidades y palabras reservadas para un lenguaje de script Web. [28]

3.5.2. Servicios

A continuación vamos a nombrar cuales son los servicios más importantes de nuestro sistema:

3.5.2.1. Eventos

- **Lista eventos por ambiente:**
  - **URL:** /event/environment/*ambiente*
  - **Parámetros:** idAmbiente

  - **Tipo:** GET

- **Consulta el evento de un sensor:**
  - **URL:** /event/*evento*/environment/*ambiente*
  - **Parámetros:** idEvento, idAmbiente
  - **Tipo:** GET
- **Nuevo Evento:**
  - **URL:** /event/environment/*ambiente*
  - **Parámetros:** idAmbiente, Objeto Event
  - **Tipo:** POST
- **Histórico de eventos:**
  - **URL:** /event/environment/*ambiente*/history/date/*fechaInicio*/to/*fechaFin*
  - **Parámetros:** idAmbiente, fechaInicio, fechaFin
  - **Tipo:** GET

3.5.2.2. Sensores

- **Lista todos los sensores del sistema:**
  - **URL:** /sensors
  - **Parámetros:**
  - **Tipo:** GET
- **Lista los sensores de un ambiente:**
  - **URL:** /sensors/environment/*ambiente*
  - **Parámetros:** idAmbiente

    - **Tipo:** GET

- **Lee un sensor:**

    - **URL:** /sensors/*idSensor*
    - **Parámetros:** idSensor
    - **Tipo:** GET

- **Nuevo Sensor:**

    - **URL:** /sensors
    - **Parámetros:** Objeto sensor
    - **Tipo:** POST

- **Borrar Sensor:**

    - **URL:** /sensors/*idSensor*
    - **Parámetros:** idSensor
    - **Tipo:** DELETE

- **Consultar tipos:**

    - **URL:** /sensors/types
    - **Parámetros:**
    - **Tipo:** GET

- **Nuevo Tipo:**

    - **URL:** /sensors/types
    - **Parámetros:** Objeto Type
    - **Tipo:** POST

- **Borrar tipo:**
  - **URL:** /sensors/types/*idType*
  - **Parámetros:** idType
  - **Tipo:** DELETE

3.5.2.3. Objetos

- **Lista todos los objetos del sistema:**
  - **URL:** /object
  - **Parámetros:**
  - **Tipo:** GET
- **Lista objetos por ambiente:**
  - **URL:** /object/environment/*idAmbiente*
  - **Parámetros:** idAmbiente
  - **Tipo:** GET
- **Nuevo Objeto:**
  - **URL:** /object/*idObject*/idValue
  - **Parámetros:** Objeto object, idObjeto, idValue
  - **Tipo:** POST
- **Dejar de seguir un objeto:**
  - **URL:** /object/notfollow
  - **Parámetros:** idSensorOject
  - **Tipo:** PUT

- **Cambiar descripción:**
  - **URL:** /object/*idObject*/description
  - **Parámetros:** descripcion
  - **Tipo:** PUT
- **Listar el estado actual de un ambiente:**
  - **URL:** /object/environment/*idEnvironment*/currentstatus
  - **Parámetros:** idAmbiente
  - **Tipo:** GET
- **Listar histórico de un ambiente:**
  - **URL:** /object/environment/*idEnvironment*/history/date/*fechaInicio*/to/*fechaFin*
  - **Parámetros:** idEnvironment, fechaInicio, fechaFin
  - **Tipo:** GET

3.5.3. Tecnología en el servidor del sistema web

- **JAVA:** Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems [29]. Este lenguaje tiene muchas sintaxis de C y C++, aunque funciona con un modelo de objetos simplificado, reduciendo así el número de errores producidos por los programadores. El gran potencial de JAVA se encuentra en su complicación, ya que se realiza a través de bytecode, permitiendo que todas las aplicaciones se ejecuten mediante una maquina virtual java. Esta ventaja, nos proporciona ejecutar el código en cualquier maquina, independientemente del sistema operativo que utilice.

  Las razones por las que se ha usado este lenguaje:

  - Orientado a objetos

- Flexible
- Independiente a la plataforma
- Fuente abierta

- **JERSEY:** Para el desarrollo de la API, se ha utilizado el framework de programación JERSEY.Jersey RESTful da soporte para JAX-RS y JAX-R [30]. La configuración de este fremework, es la siguiente:

  - **Seguridad:** Consultará los usuarios de la base de datos junto con un rol asociado a ellos, para permitir el acceso a los servicios protegidos.

    ```
    <Realm  className="org.apache.catalina.realm.DataSourceRealm"
    dataSourceName="jdbc/smartlab"
    localDataSource="true"
    roleNameCol="role"
    userCredCol="password"
    userNameCol="user"
    userRoleTable="Users"
    userTable="Users" />
    ```

  - **Filtro CORS:** Cross Origin Sharing Resource (CORS) permite al navegador realizar una petición web a otro dominio que no cumpla con la política SOP siempre y cuando el dominio destino agregue la cabecera Access-Control-Allow-Origin, especificando a qué orígenes permite la petición. Es de gran utilidad configurar el servidor con este filtro para futuras aplicaciones o prototipos móviles [31].

    ```
    <filter>
    <filter-name>CorsFilter</filter-name>
    <filter-class>...CorsFilter</filter-class>
    <init-param>
    ```

```
<param-name>cors.allowed.methods</param-name>
<param-value>GET,POST,HEAD,PUT,DELETE</param-value>
</init-param>
</filter>
<filter-mapping>
<filter-name>CorsFilter</filter-name>
<url-pattern>/*</url-pattern>
</filter-mapping>
```

- **JDBC:** (Java Database Connectivity) es una API del lenguaje JAVA. Proporciona un conjunto de operaciones las cuales permiten crear una interfaz para realizar la comunicación entre a base de datos y nuestra aplicación (ver figura 3.45).

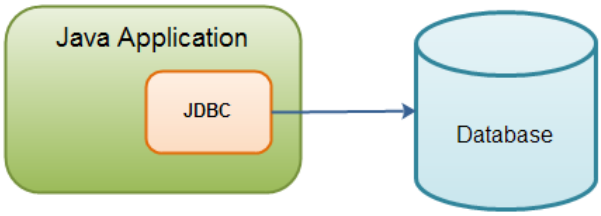

Figura 3.45: Esquema JDBC

De este modo, JDBC proporciona nuestras aplicaciones JAVA nuevas instrucciones para acceder a bases de datos SQL (Structured Query Language) [32]. Así, al utilizar esta API, no tenemos que tener en cuenta qué sistema de bases de datos estamos utilizando, ya que JDBC traduce nuestras consultas al sistema con el que nos conectamos (Oracle, MySQL, MariaDB...).

- **MySQL:** Es un sistema multiplataforma de administración de bases de datos relacionales (SGBDR) [33], desarrollado por Sun Microsystem. Es uno

de los más utilizados, ya que se adapta de una manera sencilla y eficiente en entornos clientes-servidor.

MySQL está orientado a consulta no transnacionales (MyISAM), por lo que su gran potencial reside en las aplicaciones web, ya que estas no tienen una alta concurrencia de modificación de datos y sí un gran número de lecturas.

3.5.4. Tecnología en el cliente del sistema web

- **HTML5 y CSS3: HTML** (Hyper Text Markup Language), como su propio nombre indica, es un lenguaje Markup, esto quiere decir que para transformar el contenido a una web hace uso de unas reglas con múltiples etiquetas que definen la estructura con las que se visualizará el contenido.

  Con el avance de la Web 2.0 y los nuevos requerimientos se planteó una actualización del estándar, creando así el actual HTML5. Este nuevo estándar ha proporcionado una mayor compatibilidad entre navegadores, una mejor visualización en distintas resoluciones y como principal novedad, la inserción de contenido multimedia reduciendo de este modo la dependencia con plugins.

  CS3 (Cascading Style Sheets) HTML define la estructura de la web mediante etiquetas, ya que tenemos definida la estructura podemos dotarla de un estilo personalizado (colores, disposición...) mediante hojas de estilo.

  Al igual que con el lenguaje HTML, CSS utiliza un lenguaje Markup para definir cada una de las hojas de estilo que tendrá nuestra aplicación web [34, 35].

- **Bootstrap:** Es un framework de programación (HTML5 y CSS3) desarrollado por Twitter. Es uno de los más utilizados, ya que se adapta con facilidad a cualquier resolución.

  Las características más importantes de este framework:

- Soporte completo con HTML5 y CSS3
- Inserción de imágenes responsive.
- Sistema GRID: Gracias a este sistema, podemos diseñar nuestra web mediante columnas. De este modo, podremos definir con una gran facilidad cuanta se verán dependiendo de la resolución.

Además, este framework asegura tener una compatibilidad total con los navegadores web actuales, por lo que muchos CMS (WordPress, Drupal, Joomla...) lo están utilizando para crear sus plantillas.

- **ANGULARJS:** Es un framework del lenguaje JavaScript, desarrollado por la conocida empresa Google. La principal características de este framework es el uso del patrón MVC (Modelo-Vista-Controlador). Gracias a este, podemos dotar al lenguaje HTML de nuevas etiquetas, las cuales facilitarán el desarrollo de nuestra aplicación.

  Dentro de cada controlador, podemos definir una serie de complementos:

  - **Scopes (Ámbitos):** Aquí se guarda la información de los modelos que se representan en la vista, además también se pueden utilizar como atributos para manejar algoritmos o la lógica del sistema.
  - **Directivas:** Es uno de los que más atractivo hacen a AngularJS, al definir una nueva directiva, podremos usarla en nuestra vista como si fuese una etiqueta más de HTML, de este modo toda la funcionalidad de esta reside dentro del controlador.
  - **Filtros:** nos permiten modificar la manera en el que se representa la información en la vista de nuestra aplicación, tiene un funcionamiento similar a los *Pipeline* de Unix: {{expresion — filtro }}

    Una de las funciones más utilizadas en ANGULARJS es el uso de peticiones. Un ejemplo de una de ellas es el siguiente. Al recibir el JSON

del servidor, lo almacenará en un ámbito especifico.

```
//Leer ambiente
  $scope.leerAmbiente = function() {
          $http.get(url).success(function(data) {
                  $scope.ambiente = data;
                  $scope.currentStatus();
          });
  };
```

3.5.5. Arquitectura en el prototipo de aplicación móvil

Haciendo uso de la parte servidora de nuestra aplicación, se ha desarrollado un prototipo de aplicación para dispositivos móviles. Para realizarla se ha utilizado el framework de desarrollo Cordova. Este framework, convertirá una pequeña parte de nuestro cliente web a una aplicación Android.

Resumiendo, definimos Cordova como un framework que sirve para convertir aplicaciones desarrolladas con HTML+Javascript+CSS en aplicaciones nativas para dispositivos móviles.

Gracias a este proceso, mediante AngularJS, podemos acceder a funcionalidades propias del dispositivo como cámara, sensores o notificaciones.

Un esquema del funcionamiento de AngularJS se muestra en la figura 3.46:

Figura 3.46: Esquema Cordova

Para la implementación de este prototipo, una de los objetivos más importantes era poder recibir notificaciones en Android.

Para conseguir recibir notificaciones, se ha utilizado Google Cloud Menssaging (GCM). Este servicio hará de intermediario entre nuestro dispositivo Android y nuestro Servidor de aplicaciones REST.

Cuando un dispositivo móvil se conecta con el servicio de mensajería GCM, Google le proporciona un identificador de registro único. Posteriormente nuestro sistema guardará este identificador y todas las notificaciones serán enviados a este dispositivo.

El esquema final para este objetivo se ilustra en la figura 3.47

Figura 3.47: Esquema Notificaciones

Par obtener este servicio proporcionado por Google, crearemos un nuevo proyecto dentro de la consola de desarrolladores de Google https://cloud.google.com/console .

Al crear este proyecto, se nos proporcionara un **Número de proyecto** y una **Clave de conexión**, es importante anotar ambos códigos, ya que nuestro prototipo móvil los utilizará para la conexión.

Una vez configurado este servicio, configuraremos la aplicación del prototipo móvil desarrollado en Cordova.Para esto debeos de realizar dos pasos, activar el Plugin de notificaciones (com.adobe.plugins.pushplugin), y configurar el fichero javascript AppNotificaciones.js con el ID proporcionado anteriormente en el servicio de Google.

3.5.6. Herramientas de desarrollo

Una vez descritos los lenguajes de programación empleados para este trabajo fin de grado, vamos a especificar cada unas de las herramientas que se han utilizado

para poder desarrollar cada una de las aplicaciones.

Este trabajo fin de grado tiene dos grandes partes cliente y servidor, por lo que se han tenido que utilizar distintas herramientas de desarrollo, a continuación detallaremos las más importantes.

- **NetBeans 8:** Es un entorno de desarrollo libre y multiplataforma creado por la empresa Sun Microsystems. Este IDE soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles), además de múltiples lenguajes de programación como (C, C++, PHP o Python).

- **Atom:** Es un editor de código libre multiplataforma, está desarrollado por GitHub. Se decidió se uso ya que su gran potencial reside en lo personalizable que puede llegar a ser. Atom soporta varios lenguajes de programación web, incluyendo Sass y LESS, por lo que para aplicaciones web, puede ser una buena elección.

3.6. Pruebas

Como sabemos las pruebas son el ultimo punto dentro del proceso de la ingeniera del software. Realizar estas pruebas es una actividad desarrollada para evaluar la calidad del producto, y si es necesario poder mejorarlo.

En este ámbito, uno de los *gurus* de la informática, E. W. Dijkstra dice la siguiente frase:

Program testing can be a very effective way to show the presence of bugs, but it is hopelessly inadequate for showing their absence. [36]

Por lo que consideramos con un alto nivel de interés este punto. Las pruebas que vamos a realizar (ver tablas desde 3.8 a 3.27), están basadas en los requisitos funcionales descritos en el la sección 3.2.2.1:

3.6.1. Casos

Tabla 3.8: Test 1: Identificación correcta como administrador

Condiciones	Administrador Registrado en el sistema
Acción	El administrador introduce sus credenciales (usuario y contraseña)
CheckPoint	El sistema valida los datos y muestra el panel de administración.

Tabla 3.9: Test 2: Identificación incorrecta como administrador

Condiciones	Administrador Registrado en el sistema
Acción	El administrador introduce sus credenciales (usuario y contraseña)
CheckPoint	El sistema valida los datos y un mensaje advirtiendo que el usuario o contraseña es erroneo.

Tabla 3.10: Test 3: Creación correcta de ambiente

Condiciones	Identificado como administrador
Acción	El administrador rellena los formularios del nuevo ambiente
CheckPoint	El sistema valida los datos y muestra el nuevo ambiente creado.

Tabla 3.11: Test 4: Creación incorrecta de ambiente

Condiciones	Identificado como administrador
Acción	El administrador rellena los formularios del nuevo ambiente dejando el campo nombre vacio
CheckPoint	El sistema valida los datos y muestra un mensaje de error alertando al usuario.

Tabla 3.12: Test 5: Creación incorrecta de ambiente BD

Condiciones	Identificado como administrador
Acción	El administrador rellena los formularios del nuevo ambiente y el sistema no puede conectar con la base de datos introducida
CheckPoint	El sistema valida los datos y muestra un mensaje de error alertando al usuario.

Tabla 3.13: Test 6: Borrado correcto de un ambiente

Condiciones	Identificado como administrador
Acción	El administrador pulsa el icono de borrar de un ambiente y confirma el borrado
CheckPoint	El sistema valida la petición y elimina dicho ambiente

Tabla 3.14: Test 7: Inserción correcta de un nuevo sensor

Condiciones	Identificado como administrador
Acción	El administrador rellena el formulario con los datos de un nuevo sensor
CheckPoint	El sistema valida la petición y muestra la lista de sensores del sistema

Tabla 3.15: Test 8: Inserción repetida de un sensor

Condiciones	Identificado como administrador
Acción	El administrador rellena el formulario con los datos de un nuevo sensor que ya estaba dado de alta en el sistema
CheckPoint	El sistema valida la petición y devuelve un mensaje de error indicando que no se pudo crear el sensor

Tabla 3.16: Test 9: Inserción sin tipo de un sensor

Condiciones	Identificado como administrador
Acción	El administrador rellena el formulario con los datos de un nuevo sensor y olvida seleccionar su tipo
CheckPoint	El sistema valida la petición y devuelve un mensaje de error indicando que no se pudo crear el sensor

Tabla 3.17: Test 10: Inserción correcta de nuevo usuario

Condiciones	Identificado como administrador
Acción	El administrador rellena el formulario con los datos del nuevo usuario
CheckPoint	El sistema valida la petición y devuelve la lista de usuarios

Tabla 3.18: Test 11: Borrado correcto de un usuario

Condiciones	Identificado como administrador
Acción	El administrador rellena los campos del formulario pulsa el icono de borrado de un usuario
CheckPoint	El sistema valida la petición y devuelve la lista de usuarios

Tabla 3.19: Test 12: Inserción correcta de un nuevo objeto

Condiciones	Identificado como administrador y tener ambientes dados de alta en el sistema
Acción	El administrador rellena los campos del formulario y selecciona un objeto inactivo al nuevo objeto
CheckPoint	El sistema valida la petición y devuelve la lista de objetos

Tabla 3.20: Test 13: Inserción de un nuevo objeto con un sensor activo

Condiciones	Identificado como administrador y tener ambientes dados de alta en el sistema
Acción	El administrador rellena los campos del formulario y selecciona un objeto activo al nuevo objeto
CheckPoint	El sistema valida la petición y devuelve un mensaje de error indicando que ese sensor ya esta en uso

Tabla 3.21: Test 14: Dejar de seguir un objeto

Condiciones	Identificado como administrador y tener ambientes dados de alta en el sistema
Acción	El administrador pulsa sobre el botón dejar de seguir en el objeto deseado y confirma las advertencias
CheckPoint	El sistema valida la petición y devuelve la lista de objetos

Tabla 3.22: Test 15: Situar objetos en un mapa correctamente

Condiciones	Identificado como administrador, tener ambientes y objetos dados de alta en el sistema
Acción	El administrador añade un nuevo objeto al mapa y lo ubica dentro de este
CheckPoint	El sistema guarda la nueva posición del objeto

Tabla 3.23: Test 16: Situar objetos repetidos en un mapa

Condiciones	Identificado como administrador, tener ambientes y objetos dados de alta en el sistema
Acción	El administrador añade un objeto repetido al mapa
CheckPoint	El sistema comprueba que el objeto ya estaba en el mapa y no realiza ninguna acción

Tabla 3.24: Test 17: Consultar estado actual

Condiciones	Tener ambientes y objetos dados de alta en el sistema
Acción	El usuario selecciona un ambiente en el sistema
CheckPoint	El sistema muestra el mapa junto con sus objetos y estado correspondiente

Tabla 3.25: Test 18: Conectar dispositivo móvil con el servidor.

Condiciones	Tener servidor ejecutado y dispositivo con internet
Acción	El usuario abre la aplicación
CheckPoint	La aplicación móvil cambia el estado a conectado

Tabla 3.26: Test 19: Conexión errónea entre el dispositivo móvil y el servidor.

Condiciones	Tener servidor ejecutado y dispositivo con internet
Acción	El usuario abre la aplicación
CheckPoint	La aplicación móvil notifica al usuario mediante un mensaje de error de que no puede conectar con el sistema

Tabla 3.27: Test 20: Recibir notificaciones de un evento.

Condiciones	Tener servidor ejecutado Conexión a internet en el dispositivo Suscrito mínimo a un objeto
Acción	El sistema recibe un evento
CheckPoint	La aplicación móvil alerta al usuario mediante una notificación del evento recibido

3.6.2. Resultados

A continuación se presenta (ver tabla 3.28) los resultados obtenidos tras realizar cada una de las 20 pruebas vistas en el punto anterior.

Test	Resultado	Problemas detectados	Final
Test 1	Superado		Superado
Test 2	Superado		Superado
Test 3	No superada	Campo formulario no definido	Corregido
Test 4	Superado		Superado
Test 5	Superado		Superado
Test 6	Superado		Superado
Test 7	Superado		Superado
Test 8	Superado		Superado
Test 9	Superado		Superado
Test 10	Superado		Superado
Test 11	Superado		Superado
Test 12	Superado		Superado
Test 13	Superado		Superado
Test 14	Superado		Superado
Test 15	Superado		Superado
Test 16	Superado		Superado
Test 17	Superado		Superado
Test 18	No superada	Transformar las URL a URI	Corregido
Test 19	Superado		Superado
Test 20	Superado		Superado

Tabla 3.28: Resultados Test

Uno de los errores más importantes se encontraba en el Test 18. El prototipo de aplicación móvil necesitaba acceder a una URL la cual contenía espacios y caracteres no validos. La solución a este problema fue transformar dicha URL a una URI, de este modo sería legible para el cliente.

Capítulo 4

Conclusiones y líneas de trabajo futuras

En este trabajo fin de grado se ha llevado a cabo un sistema de monitorización de ambientes inteligentes junto a un prototipo de aplicación móvil para la suscripción de eventos en un ambiente.

La propuesta presentada puede ser crítica, si tenemos en cuenta que el número de personas de edad avanzada alcanzará los dos mil millones para el año 2050, siendo una cuestión clave para las personas mayores poder vivir el mayor tiempo posible en sus hogares, con el fin de tener un envejecimiento activo y positivo [37].

Es de destacar que una de las enfermedades más comunes en las personas mayores es la demencia. En la actualidad se estima que la demencia afecta a unos 7,3 millones de personas residentes en Europa, la mayoría, mujeres [38]. La demencia se define como una pérdida de habilidades cognitivas y emocionales que interfieren sorpresivamente en las actividades de la vida diaria de la persona que la sufre [39].

Por tanto, la propuesta de este trabajo fin de grado es de gran la utilidad, ya que, entre otros, puede proporcionar a las personas con demencia en estados iniciales y a su entorno más cercano importantes ventajas frente al desarrollo de

su enfermedad.

Una primera ventaja que se puede citar es que un ambiente inteligente monitorizado con el sistema propuesta se pueden detectar cambios en las rutinas de sus habitantes, haciendo más fácil la detección temprana de la demencia y su tratamiento de las distintas fases.

Otra ventaja está relacionada con la cantidad de personas que se prevén que estén afectadas por la demencia en sus estados iniciales.

Dicha cuantía hace inviable el cuidado de forma profesional en instituciones ya que, la mayoría de las plazas de dichas instituciones están dedicadas a personas con una demencia avanzada [40]. Por tanto, en estados de demencia inicial es necesario el cuidado por parte de los propios familiares de los enfermos. En esta última cuestión reside la ventaja de nuestra propuesta, ya que permiten una mejor conexión con el hogar de la persona enferma, ofreciendo la oportunidad a los cuidadores de alejarse físicamente del enfermo sin reducir la vigilancia sobre éste.

Así, para poder mejorar la calidad de vida y el bienestar de las personas es necesario modificar los entornos donde viven actualmente y convertirlos en entornos de inteligencia ambiental..

El objetivo de este trabajo fin de grado fin de grado ha sido realizar una aplicación multiplataforma capaz de monitorizar el estado de cualquier ambiente inteligente, generando y manejando el flujo de eventos de los sensores que se encuentran desplegados en el ambiente.

Para llevar a cabo el software, se ha contando, principalmente, con el apartamento de inteligencia ambiental del Centro de Estudios Avanzados de Tecnologías de la Información y Comunicación (CEATIC).

El software desarrollado en este trabajo fin de grado ha generado un prototipo en el que los dispositivos de un ambiente inteligente se comunican mediante una arquitectura abierta basada en un modelo mixto de cliente-servidor y de publicación/suscripción de notificaciones en tiempo real. Este esquema permite la

visualización, consulta y persistencia de los sensores del laboratorio inteligente independientemente del dispositivo que se conecte a nuestra aplicación.

Destacar que gracias al esfuerzo por desarrollar una arquitectura y servicios escalables, cualquier tipo de nuevo sensor que se desee incorporar puede integrarse sin alterar el resto de componentes.

Aunque el sistema sea escalable, actualmente solo trabaja con dos tipos de sensores, por lo que en un futuro se podría añadir otros tipos con nuevas funcionalidades para monitorizarlos en nuestro sistema. También, es importante especificar como líneas de trabajo futuras, las continuas funcionalidades que se le puede añadir al software elaborado en este trabajo. Alguna de estas funcionalidades en las que se está trabajando para un futuro próximo es identificar y monitorizar actividades o exportar los datos generados por el sistema.

Otro punto importante en esta linea, es añadir todas las funcionalidades que contempla el actual sistema web dentro de nuestro prototipo de aplicación móvil.

Sin embargo, en un futuro se pretende incluir nuevas técnicas para que hagan más sencilla la integración de dispositivos inteligentes y su traducción al modelo unificado de sensores de nuestro trabajo.

Cabe destacar que en este trabajo fin de grado, como se puede observar, se han alcanzado todos los objetivos que se propusieron al comienzo de la memoria, siguiendo detenidamente, y con constancia, los pasos de desarrollo de Ingeniería del Software. A su vez, el carácter innovador de este trabajo nos ha permitido aprender nuevas técnicas de desarrollo, nuevos lenguajes de programación y múltiples conceptos en al ámbito de ambientes inteligentes.

Por último, como última conclusión personal de la memoria, me gustaría expresar mi grata satisfacción al haber podido desarrollar este trabajo fin de grado, ya que me ha permitido aplicar los conocimientos adquiridos durante años en las asignaturas de Ingeniería Informática, y también, aprender y complementar mis conocimientos gracias a las recientes tecnologías de los ambientes inteligentes, que

forman parte ya de mi vida profesional, y que abren para mí nuevas puertas de conocimiento.

Capítulo 5

Anexos

A.1. Contenido CD-ROM

En este Anexo se mostrará la estructura del contenido del disco adjunto en la documentación de esta memoria de trabajo fin de grado.

- **Aplicaciones Finales**

  Para facilitar el futuro uso del software desarrollado, dentro de esta carpeta se encuentran las tres aplicaciones desarrolladas en todo el trabajo.

  - **Cliente Web**: Aplicación HTML
  - **Cliente Móvil**: Fichero con extensión .apk para instalar en un sistema Android
  - **Servidor.war**: Fichero con extensión .war para desplegar en un sistema Tomcat.

- **Base de datos**

  El contenido que se encuentra dentro de esta carpeta es un único fichero con extensión .sql. Este fichero contiene una copia actual de toda la base de datos del sistema web.

- **Código Fuente**

  Dentro de esta carpeta, se encuentran los tres proyectos desarrollados en este trabajo fin de grado. Cada uno de ellos se abre con el IDE de programación NetBeans 7.5 o superior.

- **Memoria.pdf**

  Documento con extensión .pdf con la documentación de este trabajo.

- **Demostración.mp4**

  Vídeo en el que se muestra el sistema web y el prototipo de aplicación móvil.

B.2. Manual de Instalación

En el presente Manual de Instalación, se describe cómo desplegar el Sistema web para la Monitorización de ambientes inteligentes. Al tratarse de un sistema con una arquitectura cliente-servidor vamos describir los pasos para la instalación de estas dos partes.

B.2.1. Servidor

La aplicación desarrollada en este trabajo fin de grado necesita el siguiente software instalado y configurado en un servidor.

- Base de datos MySQL 5.5
- Java 1.7 o superior
- Tomcat 1.8

A continuación se detallará como instalar y configurar cada aplicación.

B.2.1.1. Base de datos

Para la persistencia de los datos necesitamos instalar MySQL en nuestra maquina servidora.

MySQL se puede instalar en cualquier plataforma, (Linux, Windows o MAC OS) pero para esta guía se va utilizar el sistema operativo Linux con la distribución Debian 8.0

Para realizar la instalación solo abriremos nuestra consola y tecleamos la siguiente sentencia:

```
apt-get install mysql-server mysql-client
```

Una vez instalado, los siguientes comandos son genéricos para cualquier sistema operativo, ya que son propios de MySQL.

Durante la instalación nos aparecerá un mensaje indicando que introduzcamos una contraseña de administrador.

Tras la instalación de MySQL el siguiente paso que vamos a realizar es subir la base de datos a nuestro servidor. Para realizar esta acción vamos a utilizar el cliente Client que nos ofrece MySQL para un entorno UNIX.

El primer paso es conectarnos mediante el protocolo SSH a nuestra maquina. Para ello, solo es necesario teclear el siguiente comando:

```
ssh usuario@URL -p PUERTO
```

El sistema nos pedirá la clave de acceso y una vez dentro podremos identificarnos en nuestro cliente de base de datos del siguiente modo:

```
mysql -u root -p
```

De nuevo, escribimos la contraseña de la base de datos y obtenemos la siguiente ventana (ver figura B.1).

```
root@debian:~# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 25057
Server version: 5.5.46-0+deb8u1 (Debian)

Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Figura B.1: PRONT SQL 1

Ahora vamos a importar nuestro fichero SQL a la base de datos remota que estamos conectados. Para esto hay que hacer los siguientes pasos:

1. Crear la base de datos: CREATE DATABASE Smartlab;
2. Seleccionar la base de datos: USE Smartlab;
3. Importar el fichero SQL: fichero.sql

Para comprobar que la inserción se realizó correctamente podemos teclear el siguiente comando (ver figura B.2)., el cual nos mostrará todas las tablas que tiene la base de datos a la que estamos conectados.

```
mysql> SHOW TABLES from SmartLab;
+--------------------+
| Tables_in_SmartLab |
+--------------------+
| Activity           |
| Environment        |
| Habitant           |
| HabitantActivity   |
| Notification       |
| Object             |
| PositionMap        |
| Sensor             |
| SensorObject       |
| TypeSensor         |
| Users              |
| Values             |
+--------------------+
12 rows in set (0.02 sec)

mysql>
```

Figura B.2: PRONT SQL 2

En este punto, ya tenemos instalado el servidor de base de datos junto con las tablas que vamos a utilizar en nuestras aplicaciones. Ahora debemos de configurar nuestra aplicación con los datos del anterior servidor.

Para realizar esta modificación, tan solo necesitamos un editor de texto. En este, debemos de abrir el fichero de configuración de nuestro proyecto SmartLab y navegar hasta el siguiente directorio:

/SmartLab/src/main/webapp/META-INF/context.html

Al abrir el fichero anteriormente indicando, podremos cambiar los valores de los atributos de configuración para conectarnos a la base de datos. Deben de quedar del siguiente modo (ver figura B.3)

```
<!-- BBDD Exterior  -->
<Resource auth="Container" driverClassName="com.mysql.jdbc.Driver"
        maxActive="25"
        username="root"
        password="password"
        name="jdbc/smartlab"
        type="javax.sql.DataSource"
        url="jdbc:mysql://localhost:3306/SmartLab"
/>
```

Figura B.3: Configuración base de datos

B.2.1.2. Servidor Tomcat

Una de las aplicaciones que se necesitan en nuestro servidor es Tomcat, la cual es multiplataforma.

Como se ha comentado anteriormente en esta guía vamos a explicar como se instalaría en una máquina LINUX con la distribución Debian 8.0.

El primer paso es abrir nuestra consola de comandos e introducir:

apt-get install tomcat8 apt-get install tomcat8-admin tomcat8-examples tomcat8-docs

La primera sentencia nos instala tan solo Tomcat8 y la segunda, nos proporcionará un panel de administración para controlar Tomcat desde el navegador Web, Ejemplos y manuales.

Una vez instalado, se define un usuario y contraseña para poder entrar en el panel de administración anteriormente mencionado.

nano /etc/tomcat8/tomcat-users.xml

En este fichero añadimos la siguiente sentencia, cambiando los valores admin y password por los deseados.

```
<tomcat-users>
        <user username="admin" password="password"
                roles="manager-gui,admin-gui"/>
</tomcat-users>
```

Para activar esta configuración reiniciamos nuestro servicio Tomcat8 mediante la siguiente instrucción:

service tomcat8 restart

Al finalizar toda la configuración de nuestra aplicación de servicios. debemos de compilar la versión final, al realizar esta acción, nuestro entorno de desarrollo nos genera un archivo con extensión .war. Este fichero se adjuntara en el CD, pero se puede acceder a este tras realizar la compilación en la siguiente ruta:

../SmartLab/target/SmartLab-1.0.war

Una vez que tenemos el fichero localizado debemos de entrar al panel de administración web proporcionado por Tomcat

http://servidor:8080/manager/html

Este panel nos pedirá un usuario y contraseña, después de introducir las credenciales correctas, nos aparece este panel de administración (ver figura B.4).

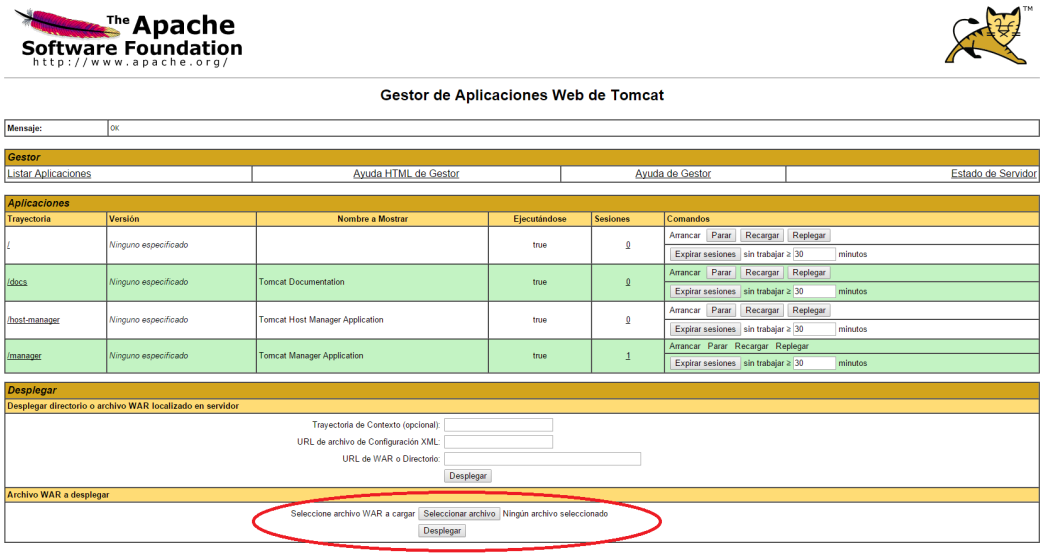

Figura B.4: Desplegar aplicación

En ella, se selecciona el archivo con extensión .WAR que se ha generado anteriormente y se pulsa el botón *Desplegar*. Extrapolando esta acción a un entorno de escritorio podría ser como *Instalar una nueva aplicación*. Después de realizar la espera de la subida del archivo, se aprecia que aparece junto a las demás aplicaciones desplegadas. Aquí (ver figura B.5) se visualizan algunas opciones como Pararla, Reiniciarla o Eliminarla.

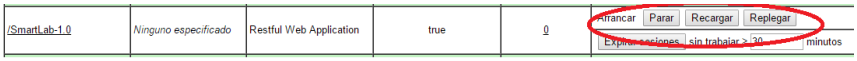

Figura B.5: Opciones aplicación Tomcat

Una vez desplegada, ya podemos acceder a todos los servicios que ofrece nuestra aplicación. Si queremos obtener la ruta donde se encuentra instalada sólo debemos hacer click en el nombre Smartlab1.0 que podemos apreciar en la imagen superior.

B.2.1.3. Acceso a servicios

http://URL:8080/SmartLab-1.0/

Para comprobar el funcionamiento podemos acceder a uno de los servicios que nos ofrece la aplicación, en la figura B.6 mostramos un ejemplo de como devuelve los datos proporcionados.

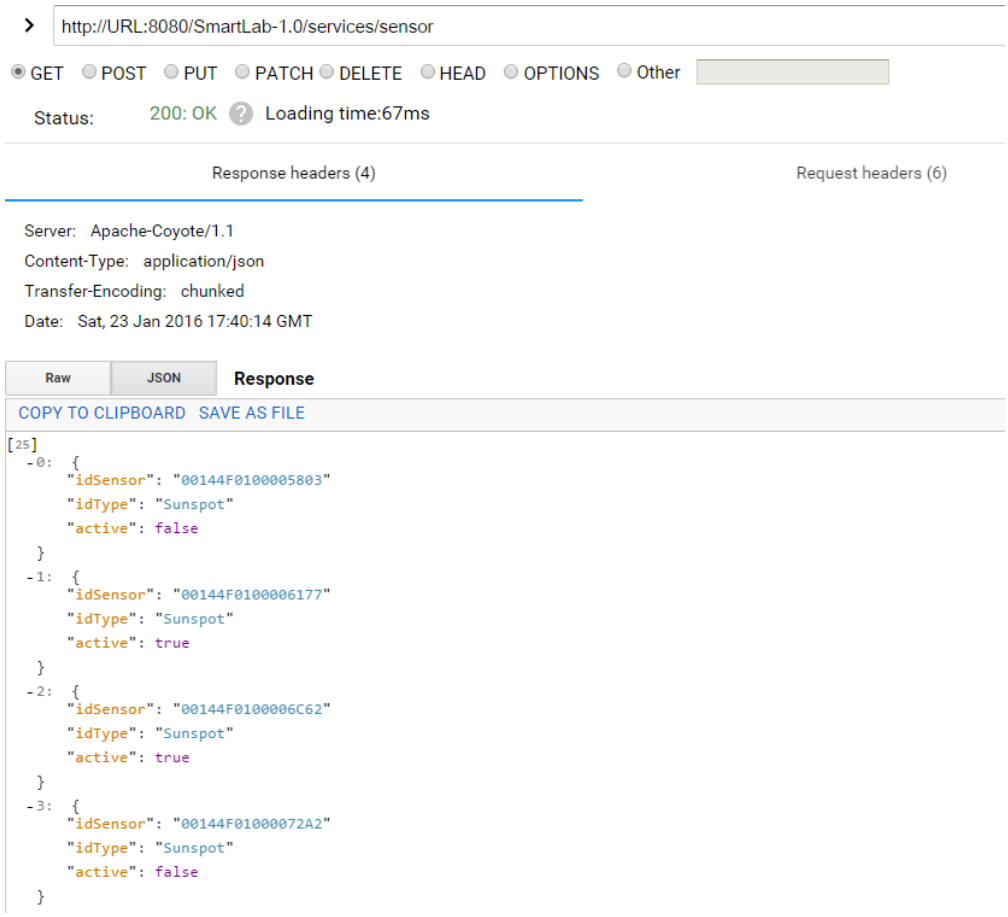

Figura B.6: Ejemplo de un servicio

B.2.1.4. Apache

Para poder instalar nuestro cliente necesitamos tener instalado Apache en nuestra maquina servidora, de este modo desde cualquier navegador podremos

conectarnos a la aplicación web.

La instalación de apache es sencilla, sólo debemos de teclear en la consola de nuestro servidor:

apt-get install apache2

B.2.2. Cliente

El despliegue de la parte del cliente es más simple que el anterior. Antes de desplegar nuestra aplicación web en el servidor, debemos de cambiar la dirección URL a la que se conectará cada uno de los servicios. La dirección que vamos a utilizar en este punto es la que nos proporcionó Tomcat en Acceso a servicios.

Para realizar dicha configuración debemos de abrir los siguientes dos ficheros con nuestro editor de texto.

SmartLabClient/public_html/js/admin/app.js
SmartLabClient/public_html/js/main/app.js

En estos ficheros se encuentra la siguiente variable, la cual debemos de cambiar por la dirección donde se encuentre nuestra aplicación de servicios, (ver figura B.7).

```
var url = 'http://URL:8080/SmartLab-1.0/services';
```

Figura B.7: Variable servidor

Al terminar la configuración, para ejecutar nuestra aplicación web necesitamos tener un servidor con Apache2 instalado y configurado. Si cumplimos con este requisito, solo debemos de copiar la carpeta adjuntada en el CD *SmartLabClient* en la siguiente ruta de nuestra maquina Linux

/var/www/

Al terminar de copiarse, introduciremos la siguiente ruta en el navegador y se abrirá nuestra aplicación web (ver figura B.8) con todos los ambientes que se encuentran en la base de datos.

http://URL/SmartLab

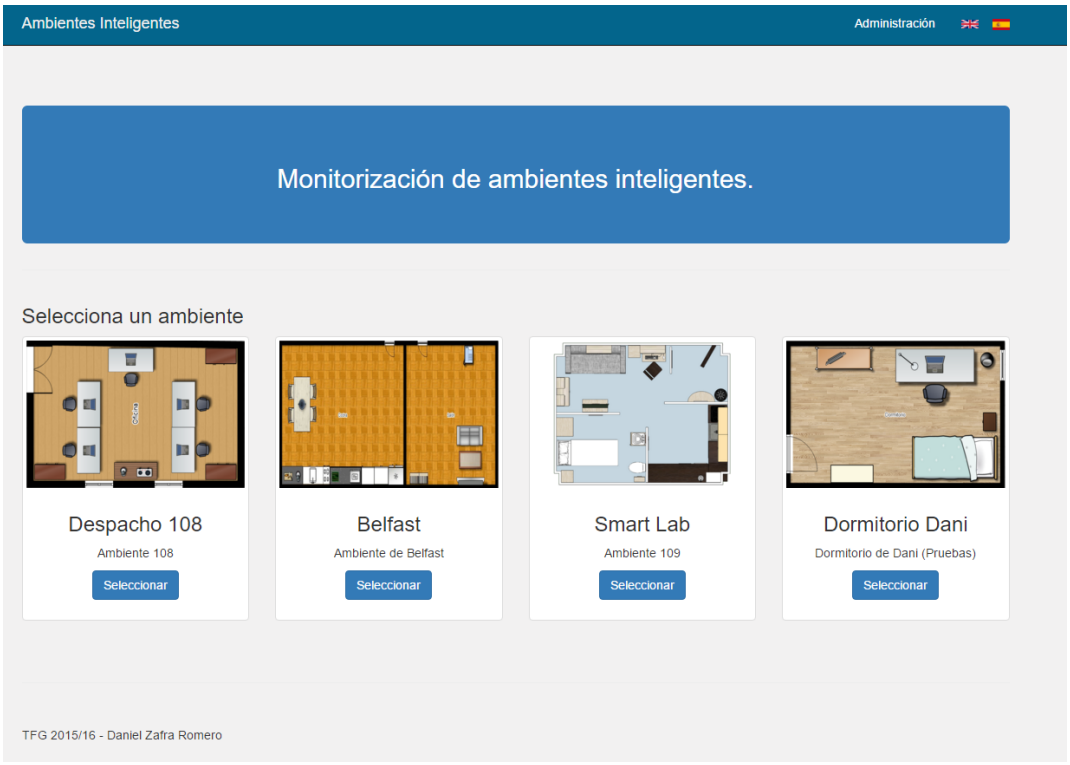

Figura B.8: Aplicación final

B.2.3. Prototipo de aplicación móvil

Para el desarrollo de este prototipo se ha utilizado la plataforma Android y el framwework de programación Cordova. Para instalar esta aplicación en nuestro dispositivo, debemos de ejecutar el archivo Cliente-Móvil.apk incluido en el CD-ROM de este trabajo fin de grado.

Si por el contrario, queremos instalar la aplicación en nuestro dispositivo directamente con el framework Cordova, solo debemos de activar el modo depuración y conectarlo por USB.

Tras ello, nos dirigimos mediante la consola a la carpeta *SmartLabMovil* dentro de nuestro trabajo fin de grado. En ella ejecutamos la siguiente instrucción:

cordova run android –device

Al terminar, se abrirá la aplicación en el terminal conectado al equipo.

C.3. Manual de administrador

C.3.1. Acceso

Para entrar al panel de administración solo se deberá de pulsar sobre el botón Administrador dentro del panel principal, al hacer click se nos desplegará el siguiente formulario (ver figura C.9) . En este formulario debemos de introducir nuestras credenciales de acceso.

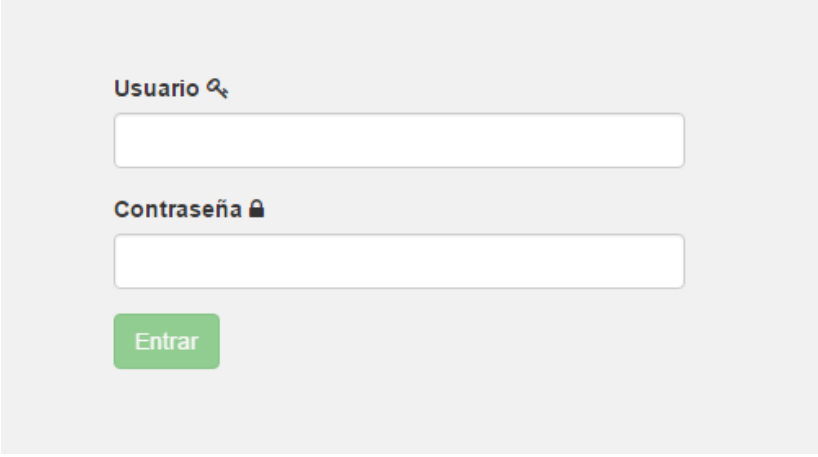

Figura C.9: Acceso panel de adminitración

Si los datos son correctos, automáticamente el sistema nos redirigirá hacia el panel de administración (ver figura C.10).

C.3.2. Gestionar ambientes inteligentes

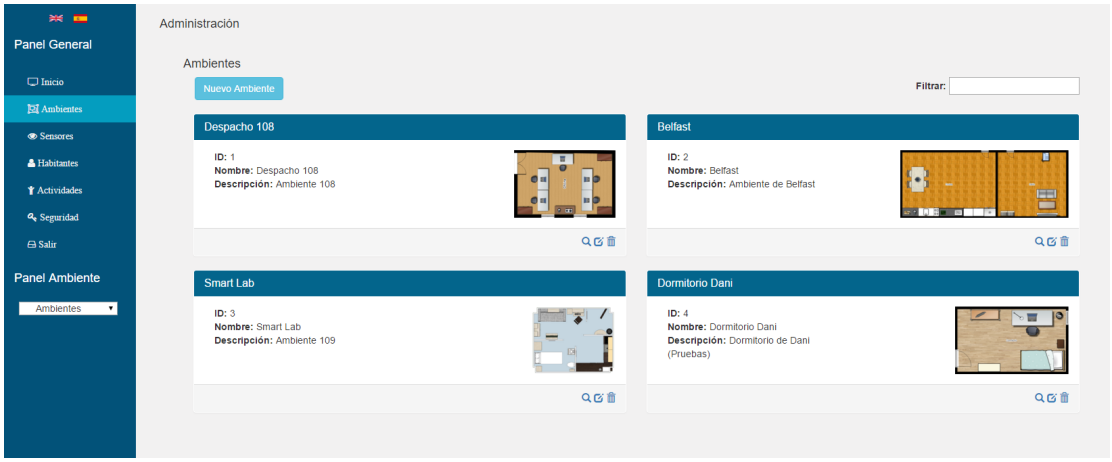

Figura C.10: Panel de administración

Al cargar el panel de administración, nos presenta todos los ambientes que tenemos dados de alta en nuestro sistema. Las operaciones que podemos hacer en este apartado son las siguientes:

C.3.2.1. Crear un nuevo ambiente

Al pulsar el botón de nuevo ambiente, el sistema abrirá la siguiente ventana modal (ver figura C.11). En esta nueva ventana introduciremos toda la información relevante al ambiente que queremos crear, junto con las credenciales para acceder a la base de datos donde se encuentran los eventos de los sensores y una fichero de imagen para el plano o mapa del mismo.

Figura C.11: Nuevo ambiente

C.3.2.2. Editar un ambiente

Del mismo modo que podemos crear un nuevo ambiente, también es posible editar los datos que pertenecen este, para realizar esta acción, solo debemos de pulsar en el icono *Editar* que hay en el inferior de la tabla de cada ambiente. Al pulsarlo se abrirá la siguiente ventana modal (ver figura C.12).

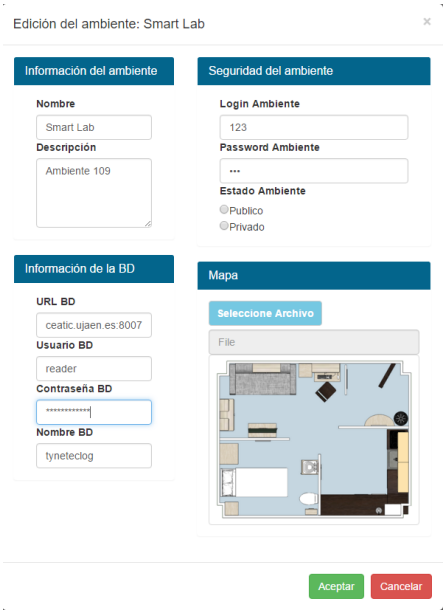

Figura C.12: Editar ambiente

C.3.2.3. Borrar un ambiente

Por último, para terminar la gestión de los ambientes inteligentes, podemos eliminar toda la información que pertenece a un ambiente. Para ello debemos pulsar el icono *Eliminar* que hay en el inferior de cada tabla del ambiente. Al pulsar este botón el sistema nos mostrará una alerta (ver figura C.13) para que confirmemos si queremos realizar esta acción.

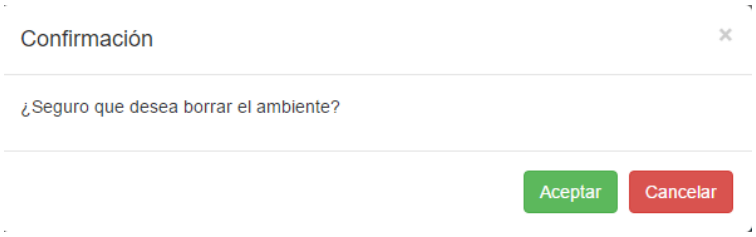

Figura C.13: Eliminar ambiente

C.3.3. Gestionar sensores

Dejando a un lado la gestión de ambientes, nos pasamos a un nuevo punto, en este podemos consultar todos los sensores que tenemos en el sistema, dar de alta nuevos sensores o eliminarlos. Cada uno de los sensores que demos de alta en este punto, estarán disponibles para todos los ambientes que se encuentren en el sistema. En la figura C.14 se visualiza la información de cada sensor.

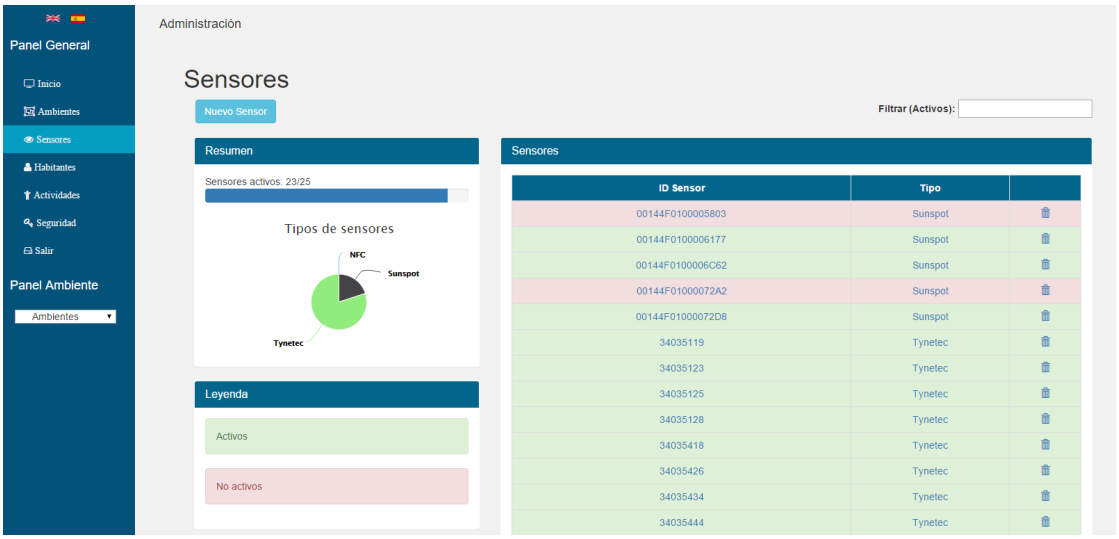

Figura C.14: Gestión de sensores

A la hora de presentar los sensores en el sistema se han utilizado dos colores para indicar si se encuentra en ese momento activos o inactivos. Esto quiere decir que cuando un sensor se encuentra activo es porque pertenece a un objeto, mientras que el sensor esta inactivo, solo esta dado de alta en el sistema pero no se está utilizando.

C.3.3.1. Dar de alta sensor

Si añadimos un nuevo sensor, debemos asegurarnos que sabemos el identificador único que lleva asociado (puede ser una dirección MAC) y cual es el tipo de

sensor. Después de pulsar sobre *Nuevo sensor* se nos despliega la siguiente ventana modal (ver figura C.15).

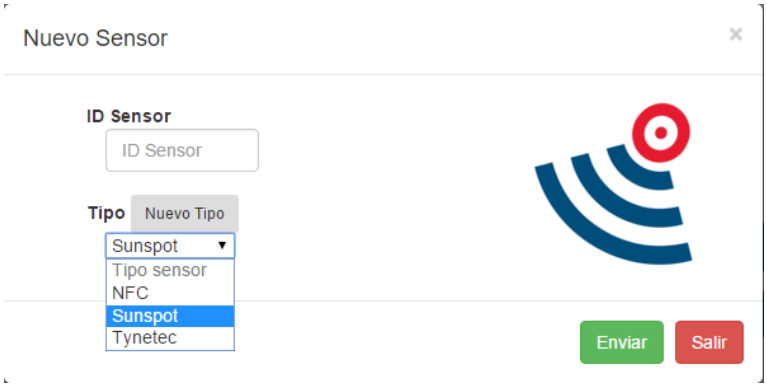

Figura C.15: Nuevo sensor

Si el tipo de sensor que vamos a añadir no se encuentra en nuestra base de datos debemos de crear un nuevo tipo. Para realizarlo, sin cerrar la ventana modal en la que nos encontramos pulsamos en *Nuevo Tipo*, aquí se abrirá otra nueva ventana modal (ver figura C.16) con los siguientes campos:

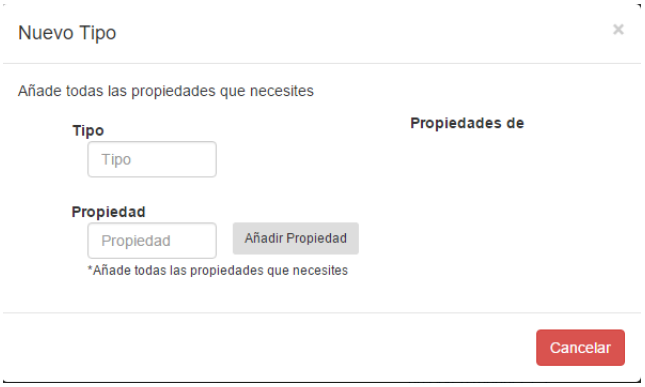

Figura C.16: Nuevo tipo

En este punto, podemos añadir tantas propiedades como tenga el nuevo tipo de sensor. Por ejemplo, si queremos añadir el tipo Sunspot, debemos de añadir todas las propiedades que utiliza este sensor, Temperatura, Luminosidad y Manipulación.

C.3.3.2. Borrar un sensor

En cambio, borrar un sensor de nuestro sistema es una tarea sencilla, tan solo debemos de buscar el ID correspondiente en la lista de sensores y pulsar en el icono borrar (Papelera). Al realizar esta acción se despliega (ver figura C.17) una nueva ventana preguntándonos si estamos seguros de realizar esta acción.

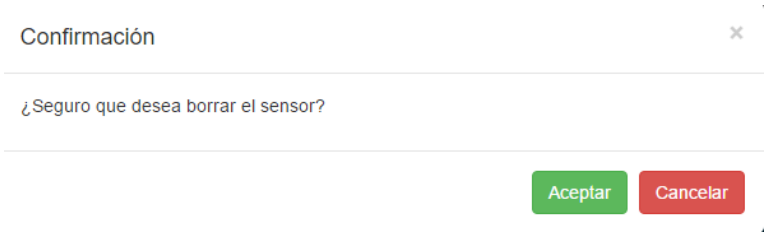

Figura C.17: Borrar sensor

C.3.4. Gestionar usuarios

En este sistema se pueden crear tantos usuarios o administradores como deseemos. Tan solo tenemos que hacer click en Seguridad dentro de nuestro menú. Y aparecerá la siguiente pantalla (ver figura C.18).

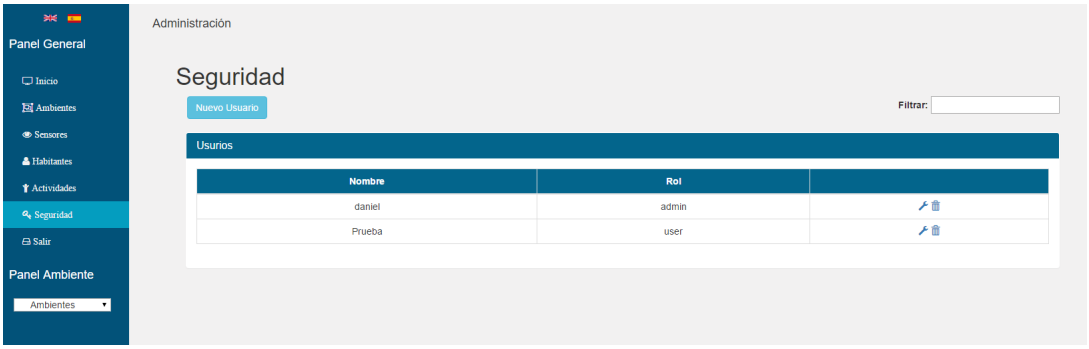

Figura C.18: Gestionar usuarios

Aquí, podremos editar, borrar o dar de alta nuevos usuarios o administradores. Es importante recordar que a la hora de crear un nuevo usuario debemos de

asignarle un rol, pudiendo elegir entre Usuario normal, o Administrador. Cada uno de los nombres del usuario se tomará como identificador único.

C.3.5. Gestionar elementos del mapa

Es una de las partes más atractivas de nuestro sistema, al elegir un ambiente nos aparecerán dos nuevos botones que estaban ocultos en nuestro menú. Una de las nuevas opciones que podemos hacer es gestionar elementos del mapa, pulsando sobre *Mi mapa*.

Al entrar en este aparatado vemos el mapa asociado con el ambiente que tenemos seleccionado. Aquí podemos situar cada uno de los objetos que tenemos en el ambiente, para realizarlo, solo debemos de desplegar el menú en el que aparecen todos los objetos y pulsar la el botón *Insertar*. Al realizar esta acción, un nuevo icono aparece en el mapa (ver figura C.19), este icono tiene la particularidad e que es móvil, por lo que podemos ubicarlo en el mapa de una manera similar a como se encuentra en el ambiente. Esto nos facilitará el visualizado del ambiente.

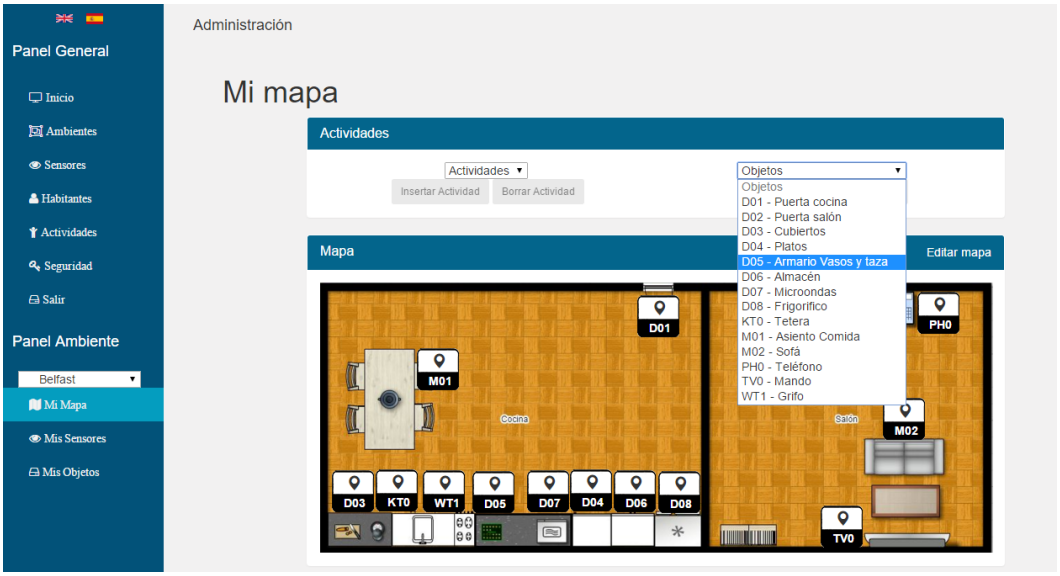

Figura C.19: Gestión elementos visuales

C.3.6. Gestionar Objetos

Es la parte más compleja del panel de administración. Aquí el administrador de cada ambiente puede asociar un sensor con un objeto concreto. Esta asociación guardará automáticamente una marca de tiempo ligada al momento en el que se crea. Este apartado se visualiza del siguiente modo (ver figura C.20)

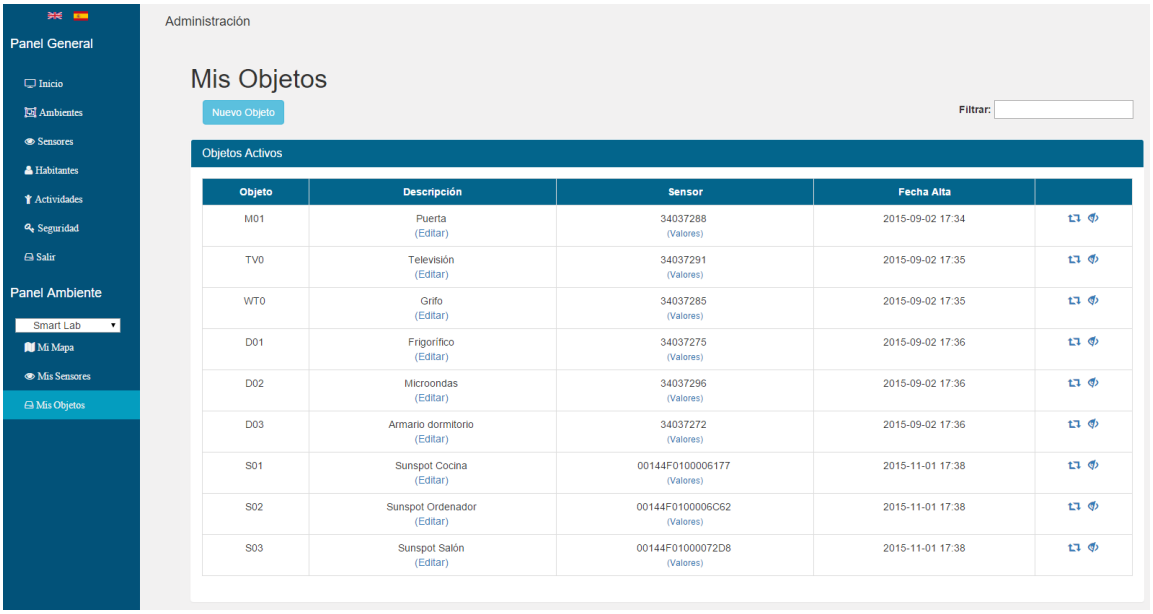

Figura C.20: Gestionar objetos

C.3.6.1. Crear nuevo objeto

Si deseamos crear un nuevo objeto, tenemos que complementar el formulario ilustrado en la figura C.21.

Figura C.21: Nuevo objeto

Aquí, debemos de introducir un código de objeto, único para cada ambiente, una breve descripción y el sensor con el que estará ligado. Además también tenemos que incluir de que modo se van a traducir los valores de este objeto.

Por ejemplo si nos encontramos con un sensor Tynetec instalado en un teléfono. Cada vez que el objeto reciba un evento, este nos convertirá el estado numérico del sensor a un estado mucho más fácil de interpretar. En la figura C.22 se puede apreciar un esquema de este funcionamiento.

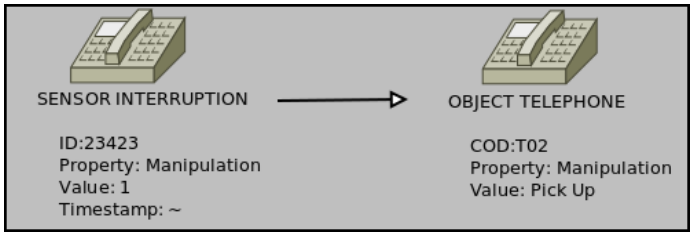

Figura C.22: Conversión sensor a objeto

C.3.6.2. Crear nuevos valores

El anterior ejemplo, convertíamos los valores numéricos del sensor a una orden en lenguaje natural, para poder realizar esto, utilizamos los valores. Estos valores se asocian al crear un nuevo objeto, y podemos dar de alta tantos valores como

deseemos. Para ello solo debemos de saber el tipo de sensor que estamos ligando al objeto y que información envía al dataset.

Los sensores utilizados del tipo Tynetec utilizan los siguientes valores numéricos para indicar su estado:

- 1 Abierto
- 9 Cerrado
- 0 Batería baja

De este modo si tenemos el sensor asociado a una puerta podemos crear los valores Abierta/Cerrada, asociando cada uno de ellos al 1 ó 9.

C.3.6.3. Dejar de seguir un objeto

Si por alguna razón, queremos dejar de observar como se encuentra un objeto, solo debemos dejar de seguirlo. De este modo el objeto no aparecerá en la visualización del estado actual. Y solo podremos consultar sus eventos en el histórico. Para realizar esta acción debemos de pulsar en el icono asociado a la acción Desvincular (Ojo tachado).

Al realizarla, nos saldrá la alerta inferior indicando que esta acción es irreversible. Después el objeto pasará a la tabla de *Objetos Inactivos* (ver figura C.23)·

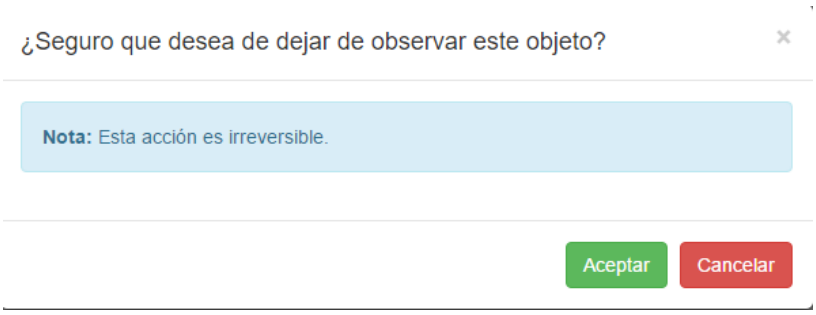

Figura C.23: Dejar de observar

C.3.6.4. Otras opciones

Otras opciones que se pueden realizar en la gestión de objetos son:

- Cambiar descripción
- Cambiar sensor

D.4. Manual de usuario

D.4.1. Cambiar de idioma

Toda la aplicación web esta traducida en dos idiomas Español e Inglés. Automáticamente la aplicación detecta cual es el idioma de tu navegador, pero si quieres forzar el cambio, tan solo debemos pulsar en el icono asociado a la bandera del país (ver figura D.24). Todo el texto de la aplicación cambiará al idioma deseado sin recargar de nuevo el contenido de la página que se esta visualizando.

Figura D.24: Cambiar idioma

D.4.2. Monitorizar estado actual

El uso principal de esta aplicación es monitorizar como se encuentra cada uno de los ambientes que tenemos dados de alta en el sistema. Nada más entrar en nuestra aplicación seleccionamos el ambiente deseado y obtenemos la siguiente ventana (ver figura D.25).

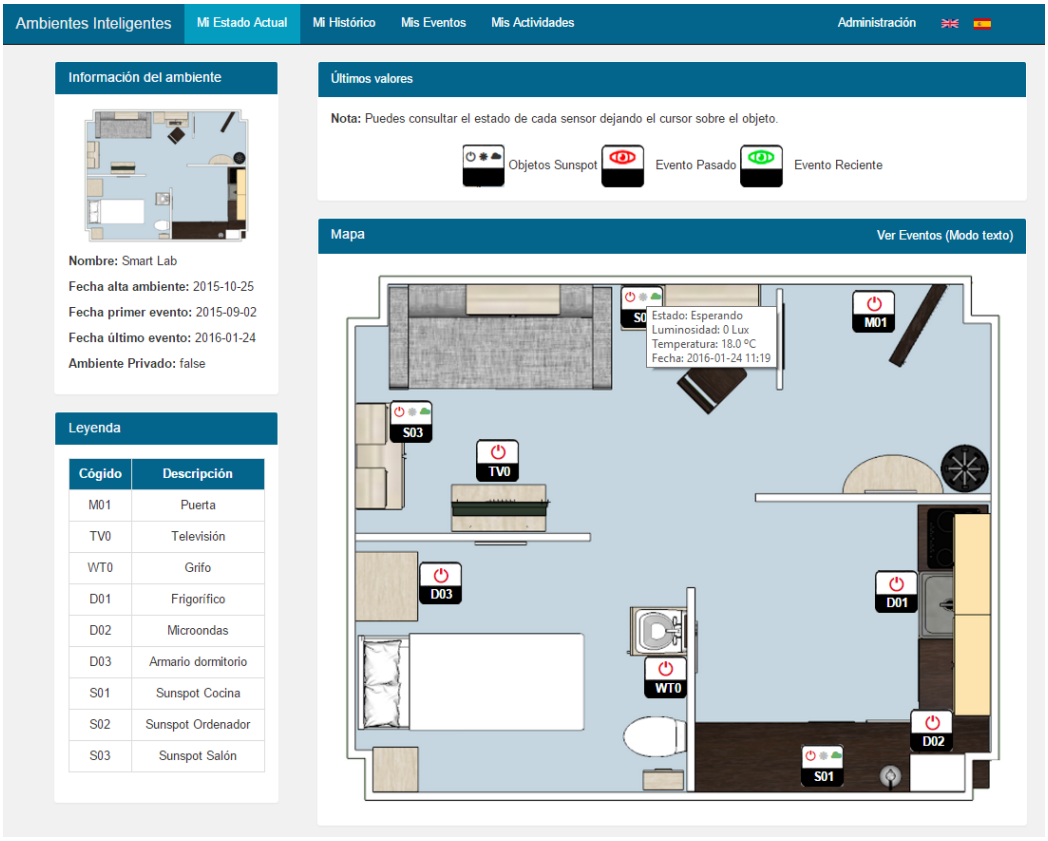

Figura D.25: Estado actual

Aquí encontraremos todos los objetos que están dados de alta en el sistema y una breve descripción del ambiente. Por cada uno de los objetos que se muestran en el mapa, hay unos iconos. Estos iconos simbolizan el estado actual del sensor al que esta asociado el objeto. Si queremos obtener más información tan solo debemos de dejar el curso del ratón encima de un objeto.

D.4.3. Consultar histórico

Visualiza el estado actual de un ambiente es interesante para saber como se encuentra en ese momento, pero quizás sea mucho más importante poder visualizar como se encontraba un día concreto a una hora precisa. En este punto es donde entra el juego el apartado *Mi Histórico*, en el que podemos visualizar todo lo que

ha ocurrido en el ambiente desde que fue dado de alta.

Al entrar en esta pestaña, se visualiza una pantalla muy similar a la del *Estado actual*, pero en esta tenemos muchas más opciones.

En primer lugar, para visualizar todos los eventos, pulsamos sobre la fecha y se nos despliega el siguiente calendario (ver figura D.26).

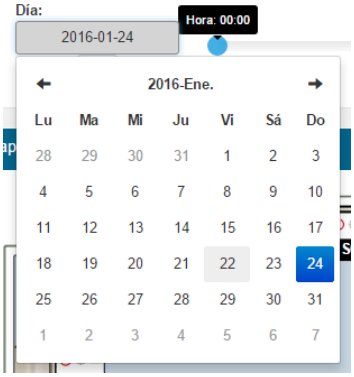

Figura D.26: Calendario histórico

Al elegir el día deseado, automáticamente la aplicación carga todos los eventos que ocurrieron. Para visualizar estos eventos tenemos dos opciones (ver figura D.27).

- Mover la barra de tiempo, de este modo mostraremos como se encontraba el ambiente en la hora seleccionada.
- Pulsar el botón Play, en esta opción la barra se moverá automáticamente, y el sistema reproducirá el estado actual del ambiente.

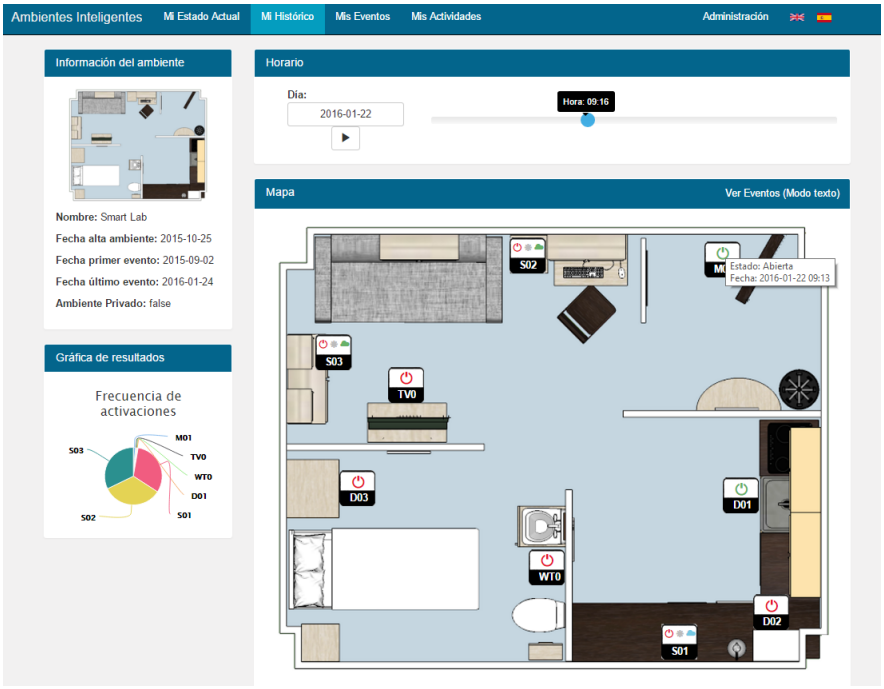

Figura D.27: Histórico

Además de poder visualizar el estado al ambiente, el sistema agrupara los datos y mostrará dos gráficas (ver figura D.28), representado la temperatura y la luminosidad durante las 24 horas seleccionadas.

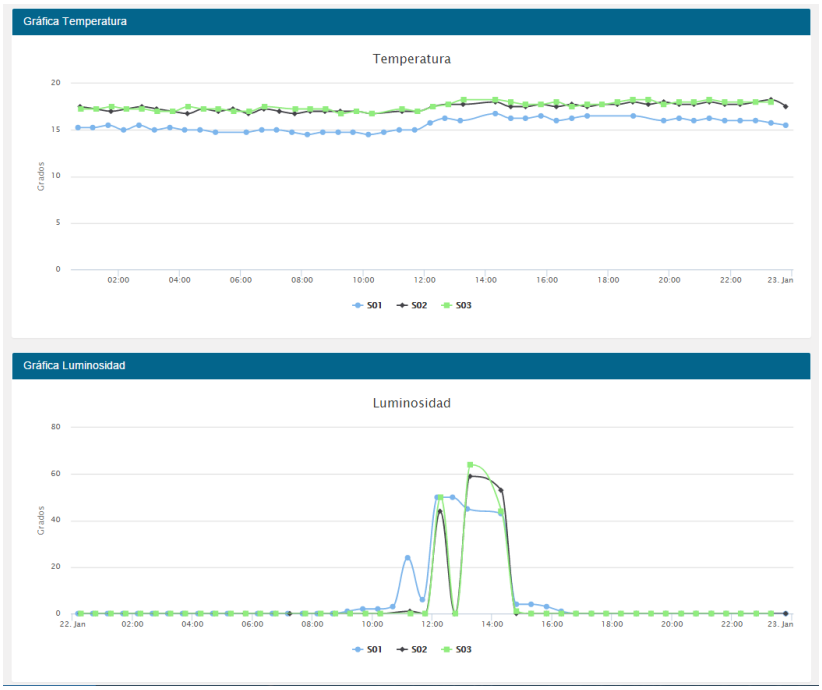

Figura D.28: Gráficas histórico

Por último, podemos visualizar todos los eventos ocurridos en dicha fecha pulsando *Ver eventos (Modo texto)* obteniendo la siguiente ventana (ver figura D.29).

Eventos (Modo Texto)

Objeto	Fecha Evento	Estado	Descripción
M01	2016-01-22 09:13	Abierta	Puerta
M01	2016-01-22 12:04	Abierta	Puerta
M01	2016-01-22 12:57	Abierta	Puerta
M01	2016-01-22 13:05	Abierta	Puerta
TV0	2016-01-22 12:23	Encendido	Televisión
TV0	2016-01-22 14:19	Encendido	Televisión
WT0	2016-01-22 11:51	Encendido	Grifo
D01	2016-01-22 09:13	Abierta	Frigorífico
D01	2016-01-22 11:51	Abierta	Frigorífico
D02	2016-01-22 11:52	Abierta	Microondas
M01	2016-01-22 11:50	Abierta	Puerta

Figura D.29: Histórico (Modo Texto)

D.4.4. Visualizar eventos

Para finalizar este manual, mostramos la pestaña *Visualizar Eventos*, en ella consultaremos el histórico de eventos de un ambiente en modo texto. Para realizar esta acción tenemos tres métodos:

- Fecha: Marcaremos un rango desde/hasta.
- Tiempo: Utilizaremos un tiempo establecido, (Última semana/día o mes)
- Cambios: Consultará los últimos N cambios de los objetos seleccionados.

Tras decidir que tipo de consulta vamos a realizar, solo nos queda seleccionar los objetos y pulsar sobre el botón *Consultar*. El sistema mostrará todos los eventos de los objetos seleccionados y una gráfica indicando el número de eventos de cada objeto agrupados en un tiempo de 24 horas. De este modo podemos saber cuales

son los objetos que más eventos producen en una hora determinada (ver figura D.30).

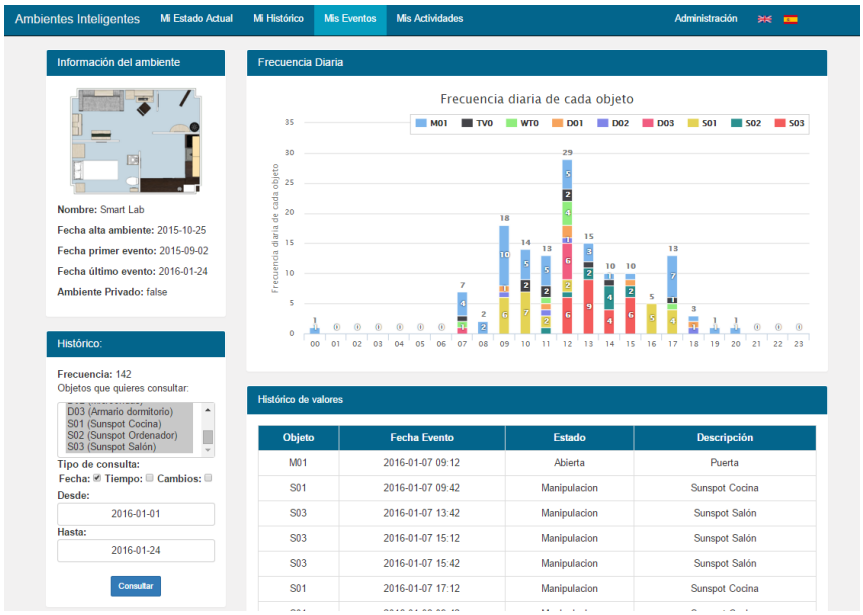

Figura D.30: Eventos

D.4.5. Prototipo de aplicación móvil

Al iniciar nuestra aplicación móvil nos encontramos con la siguiente pantalla (ver figura D.31), para que aparezca el menú, solo pulsamos en el icono superior izquierdo. En este menú tenemos las siguientes opciones:

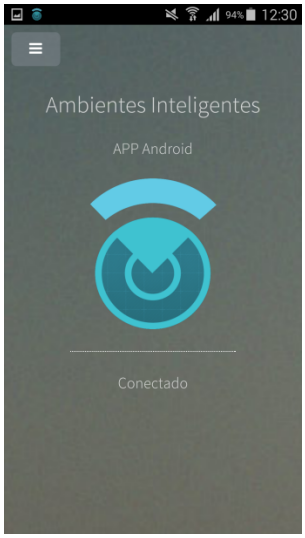

Figura D.31: Principal - Estado Actual

1. Consultar estado actual.

   Seleccionamos el ambiente que queremos consultar el sistema visualizará el estado de todos los sensores ligados con el ambiente del siguiente modo, (ver figura D.32).

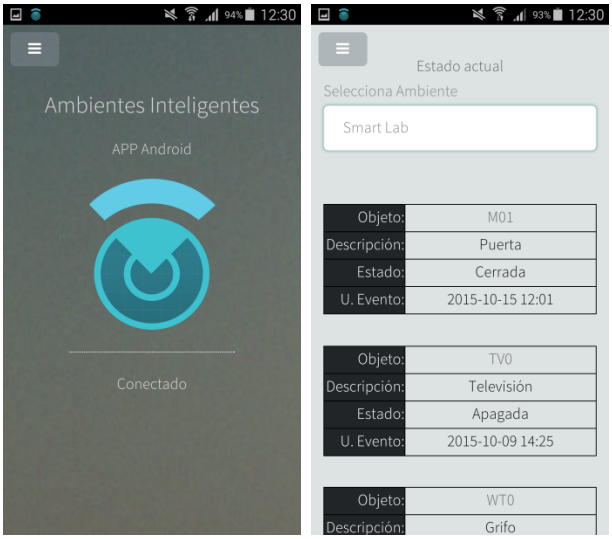

Figura D.32: Principal - Estado Actual

2. Suscribirse a objetos.

   Para la suscripción, solo seleccionamos que objetos queremos monitorizar a tiempo real. As su vez, también podemos eliminar suscripciones pulsando el botón asociado a cada objeto.

   En la figura D.33 se puede visualizar el proceso para configurar a que objetos queremos suscribirnos.

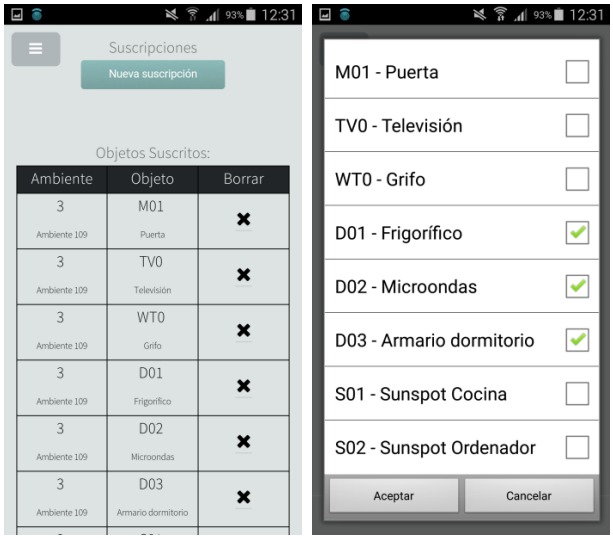

Figura D.33: Suscripciones

3. Notificaciones.

   Cada vez que un objeto envía un evento al sistema, nuestra aplicación móvil informará de dicho evento mediante una notificación Push (ver figura D.34).

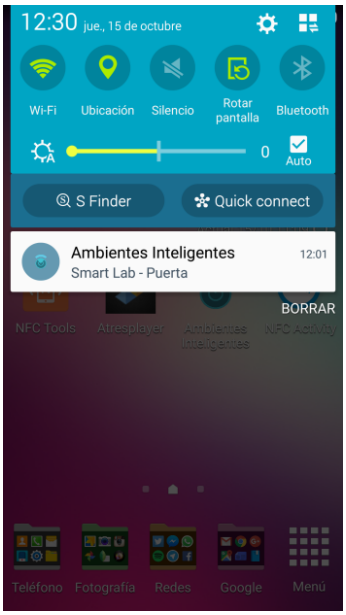

Figura D.34: Notificaciones

Bibliografía

[1] D. Navarro, L. G. Cruz, J. G. R. Torres, A. R. Saldivar, S. E. Schaeffer, J. Torres-Jimenez, A. Rodriguez-Cristerna, R. F. Brena, E. García-Ceja, R. Z. Cabada, *et al.*, "Inteligencia artificial: una reflexión obligada," *Inteligencia artificial: una reflexión obligada*, 2014.

[2] G. de Inteligencia Computacional, "Inteligencia ambiental, sistemas ubícuos y visión por computador en telefonía móvil. estado del arte,"

[3] J. A. Agreda and E. Gonzalez, "Ambient intelligence based multi-agent system for attend elderly people," in *Computing Colombian Conference (9CCC), 2014 9th*, pp. 115–120, IEEE, 2014.

[4] I. de Investigación de Enfermedades Raras *et al.*, "Instituto de salud carlos iii-ministerio de ciencia e innovación," *Guemes Carcaga I., Martín Arribas M. Canal Bedia R., Posada De La Paz, M.,«Evaluación de la eficacia de las intervenciones psicoeducativas en los trastornos del espectro autista». Madrid: IIER-Instituto de Salud Carlos III*, 2009.

[5] P. Georgieff, "Ambient assisted living," *Marktpotenziale IT-unterstützter Pflege für ein selbstbestimmtes Altern, FAZIT Forschungsbericht*, vol. 17, pp. 9–10, 2008.

[6] D. Martínez, F. Blanes, J. Simo, and A. Crespo, "Redes de sensores y actuadores inalámbricas: Una caracterización y caso de estudio para aplicaciones

médicas en espacios cerrados," *Pendiente de publicación: XXIX Jornadas de Automática. Universidad Rovira i Virgili de Tarragona, España*, 2008.

[7] E. D., "The internet of things: How the next evolution of the internet is changing everything," 2011.

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

[9] P. Soraya, "Internet de las cosas," 2012.

[10] D. Evans, "Internet de las cosas," *Cómo la próxima evolución de Internet lo cambia todo*, 2011.

[11] C. Principal, "El internet de las cosas," *Fundación de la Innovación Bankinter*, vol. 1, no. 1, p. 1, 2011.

[12] P. A. C. Pérez, J. R. García, J. J. R. Ibáñez, and C. S. Meléndez, *Telemedicina.: Ingeniería biomédica*, vol. 56. Universidad de Castilla La Mancha, 2009.

[13] "Smartlab ceatic[online]." urlhttp://ceatic.ujaen.es/es/smart-lab-0.

[14] R. P. Areny, *Sensores y acondicionadores de señal*. Springer-Verlag, 2004.

[15] L. Nashelsky, *Teoría de circuitos y dispositivos electrónicos*. Pearson, 2003.

[16] J. C. R. N. Antonio Serna Ruiz, Francisco Antonio Ros Garcia, *Guía práctica de sensores*. Creaciones Copyright SL, 2004.

[17] A. Bateman, *Comunicaciones digitales*. Marcombo, 2003.

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

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

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

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

[22] "Sunspotworld." urlhttp://www.sunspotdev.org/, 2008.

[23] "Tynetec." urlhttp://www.tynetec.co.uk/, 2010.

[24] E. Freeman, E. Robson, B. Bates, and K. Sierra, *Head first design patterns*. .ºReilly Media, Inc.", 2004.

[25] A. T. Espinosa, J. G. C. Sagredo, M. M. Reyes, and M. d. L. L. García, "Automatización de la codificación del patrón modelo vista controlador (mvc) en proyectos orientados a la web," *CIENCIA ergo-sum*, vol. 19, no. 3, pp. 239–250, 2012.

[26] O. Reichenstein, "Web design is 95 % typography," *Retrieved from Information Architects: http://informationarchitects. net/blog/the-web-is-all-about-typography-period*, 2003.

[27] "Web services architecture[online]." urlhttp://www.w3.org/TR/ws-arch/.

[28] "Json, o'reilly, t: "what is web 2.0: design patterns and business models for the next generation of software"[online]." urlhttp://www.oreillynet.com/go/web2.

[29] D. Arnow and G. Weiss, *Introducción a la programación con Java*. Pearson Publications Company, 2001.

[30] J. Sandoval, *Restful java web services: Master core rest concepts and create restful web services in java*. Packt Publishing Ltd, 2009.

[31] M. Hossain, *CORS in Action*. Manning, 2014.

[32] G. Reese, *Database programming with JDBC and Java*. .º'Reilly Media, Inc.", 2000.

[33] C. Thibaud, *MySQL 5: instalación, implementación, administración, programación*. Ediciones ENI, 2006.

[34] "Html(5) tutorial[online]." urlhttp://www.w3schools.com/html/.

[35] "Css3 introduction[online]." urlhttp://www.w3schools.com/css/.

[36] E. W. Dijkstra, "The humble programmer," *Communications of the ACM*, vol. 15, no. 10, pp. 859–866, 1972.

[37] G. Smith, S. Del Sala, R. H. Logie, and E. A. Maylor, "Prospective and retrospective memory in normal ageing and dementia: A questionnaire study," *Memory*, vol. 8, no. 5, pp. 311–321, 2000.

[38] C. Europea, *Salud pública*. 2009.

[39] J. Arango, S. Fernández, and A. Ardila, "Las demencias: aspectos clínicos, neuropsicológicos y tratamiento," *México: Manual*, 2003.

[40] "Imserso (2014). informe 2012. las personas mayores en españa. datos estadísticos estatales y por comunidades autónomas. madrid: Ministerio de sanidad, servicios sociales e igualdad.."