Trabajo de Fin de Grado

BulletPoint. Tecnología beacon en entornos universitarios

Alejandro Hernández Padrón

La Laguna, 17 de abril de 2018

D. **Francisco de Sande González**, con DNI nº 42.067.050-G profesor Titular de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutor

C E R T I F I C A

Que la presente memoria titulada:

"*BulletPoint. Tecnología beacon en entornos universitarios*"

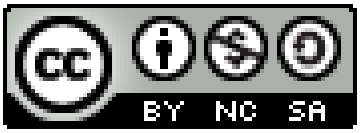
ha sido realizada bajo su dirección por D.ª **Alejandro Hernández Padrón**, con DNI nº 79.089.251-W

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 17 de abril de 2018

Agradecimientos

Mis agradecimientos al profesor Francisco de Sande González por su labor como tutor de este proyecto, orientando este trabajo, compartiendo su conocimiento y exigiendo siempre lo mejor.
Asimismo, me gustaría agradecer a la Universidad de La Laguna, a los Servicios TIC y a Don Juan Carlos Hernández Perdomo de los servicios TIC de la ULL su colaboración en este proyecto y la puesta a disposición de los beacons, sin los cuáles habría sido imposible el desarrollo de este proyecto.
Gracias a Don Alberto Morales de la empresa Galotecnia Redes Sistemas y Servicios SL, por su tiempo y dedicación en la explicación de los requisitos técnicos necesarios para la implantación de los beacons en el sistema de parking de la ULL, de estudio en el desarrollo de este trabajo.
Por último, agradecer también al personal de TITSA por el trato recibido, la participación y la rápida respuesta a las consultas, facilitando con ello el desarrollo del proyecto.

Licencia



Resumen

Este documento constituye el trabajo de investigación del alumno durante el proceso de desarrollo de una aplicación para dispositivos móviles Android mediante el uso de una de las tecnologías más recientes y menos conocidas en el mercado actual: los Beacons.

Partimos de los conocimientos de programación en Java adquiridos en la asignatura: "Diseño arquitectónico y patrones" cursada en el itinerario de Ingeniería del Software. Esta asignatura, impartida en el tercer curso del Grado en Ingeniería Informática de la Universidad de La Laguna, ha sido la que ha sentado los fundamentos a partir de los cuáles se ha desarrollado la aplicación.

Durante este proyecto, el alumno ha conseguido adquirir independencia en su trabajo, visión y planificación realizando tareas de investigación, desarrollo y documentación, que han dado como resultado la obtención de conocimientos durante el proceso de trabajo.

También se ha investigado la reciente tecnología "beacon" que si bien aún no tiene un impacto muy grande, en un futuro inmediato se espera que se empiece a utilizar con naturalidad en distintos ámbitos: turismo, comercio, enseñanza, etc.

Palabras clave: Aplicaciones Android, Java, dispositivos móviles, programación, Beacons.

Abstract

The aim of the project has been the development of an application for Android devices that uses beacon technology for some of its main features.

Based on the knowledge of *Java* programming obtained in the subject: *"Architectural Design and Patterns"* studied in the Software Engineering Branch (given in third year of Computing Engineering degree from *"La Universidad de La Laguna"*). In this work we have acquired the basic knowledge needed to develop Android applications introducing us in the development of applications related to beacon technology.

Moreover, the student has learned independence in her work and gained vision and scheduling aptitudes, developing different labors of research, development and documentation that have come to give her a wide knowledge during the development of this project.

Apart from all this, it's of great value for the student to get to know this new beacon technology. I have investigated and learned from this new technology, which at the moment is not well known, but in the close future it is expected to get more attention in different sectors, such as tourism, trading or learning.

Keywords: *Application for Android, Java, mobile devices, programming, Beacons.*

Índice general

Índice de figuras

Introducción

Este documento comprende el trabajo de investigación y desarrollo realizado por el alumno en la consecución de su Trabajo de Fin de Grado (TFG), con el que culminará sus estudios del Grado en Ingeniería Informática cursados en la Escuela Superior de Ingeniería y Tecnología (ESIT) de la Universidad de la Laguna (ULL).

Capítulo 1

Objetivos

Este TFG tiene los siguientes objetivos principales:

- Por un lado se pretende ampliar los conocimientos en tecnologías móviles en el sistema operativo *Android* [**?**] y en el desarrollo de aplicaciones para este sistema operativo.
- Otro objetivo presente en este TFG es que el alumno investigue y profundice en las tecnicas y tecnologías de reailidad virtual presentes en la actualidad.
- Por otro lado, también se pretende que el alumno se familiarice con el uso de herramientas de control de versiones utilizando *Github* [**?**] y de edición de textos técnicos utilizando *LaTeX* [**?**].
- Por último, tras las labores de investigación y recopilación de información correspondientes, se espera que el alumno aplique los conocimientos adquiridos para desarrollar una aplicación funcional que cubra las necesidades propuestas.

Capítulo 2

Herramientas y Tecnologías

Este capítulo tiene como objetivo presentar las distintas herramientas software y tecnologías empleadas por el alumno en el desarrollo de .

2.1. Introduccion

A continuación se explicarán brevemente las distintas herramientas software utilizadas en el proyecto.

2.1.1. Android Studio

Android Studio [**?**] es el IDE (Entorno de Desarrollo Integrado) oficial para el desarrollo de aplicaciones en Android, basado en IntelliJ IDEA [**?**]. Android Studio ofrece una serie de funcionalidades que han facilitado a la desarrolladora numerosas tareas, entre las cuales podemos destacar:

- Un sistema de compilación basado en Gradle[**?**] que ha simplificado tanto la inserción de dependencias de las distintas librerías que se han tenido que utilizar, como la compilación de la aplicación.
- Un emulador rápido y fácil de utilizar que ha ayudado a visualizar las distintas pantallas durante el desarrollo aunque no ha sido de mucha utilidad para probar el funcionamiento al ser dependiente la app de la tecnología Bluetooth.
- La facilidad para publicar cambios a aplicaciones ya funcionando sin tener que eliminar y volver a crear un nuevo APK parando la app.
- Un sistema de visualización de las diferentes pantallas muy completo, con soporte visual para añadir componentes y cambiar atributos fácilmente.

- Un sistema de depuración, con una interfaz sencilla e intuitiva.

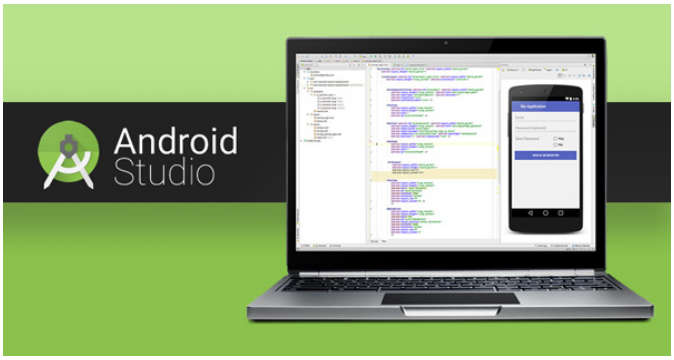


Figura 2.1: Android Studio, un IDE flexible e intuitivo.

Se ha utilizado este IDE frente a otros como Eclipse + ADT [**?**] debido a que en la actualidad es el IDE oficial con soporte de Google. Se ha preferido aprender a utilizar este entorno con vistas al futuro, ya que parece que se consolidará como el preferido para los desarrolladores Android.

2.1.2. LaTex

LaTeX es un sistema de composición de textos, orientado a la creación de documentos que presenten una alta calidad tipográfica. Por sus características y posibilidades, es usado especialmente en la generación de artículos y publicaciones científicas que incluyen, entre otros elementos, expresiones matemáticas, gráficos o figuras.

LaTeX está formado por un gran conjunto de macros de TeX, escrito por Leslie Lamport en 1984, con la intención de facilitar el uso del lenguaje de composición tipográfica, creado por Donald Knuth. LaTeX es software libre bajo licencia LPPL.

Se ha decidido utilizar este sistema debido al carácter profesional que aporta a los documentos. Ha sido una buena oportunidad para aprender a usar un sistema de composición de texto como este, ya que en un futuro puede ser beneficioso el saber manejar esta herramienta.

Si bien es cierto, que el uso de esta herramienta frente a otros editores más familiares ha sido algo tedioso en el inicio, es verdad que una vez acostumbrada a su uso ha resultado ser muy eficaz. En el proceso de aprendizaje se recurrió principalmente a manuales por internet, alguno a destacar en español sería [**?**]

2.1.3. Github

GitHub[**?**] es una forja (plataforma de desarrollo colaborativo) para alojar proyectos que utiliza el sistema de control de versiones Git. Utiliza el framework Ruby on Rails por GitHub, Inc. (anteriormente conocida como Logical Awesome). Desde enero de 2010, GitHub opera bajo el nombre de GitHub, Inc. El código se almacena de forma pública, aunque también se puede hacer de forma privada, creando una cuenta de pago.

Se ha decidido crear un repositorio en esta plataforma para poder llevar un control y una trazabilidad del proyecto. El tutor y el alumno han trabajado en este repositorio de manera conjunta. En el caso del tutor, principalmente para revisar el seguimiento semanal y llevar un control de las tareas. En el caso del alumno, para tener un repositorio donde subir los distintos elementos que se han ido generando a lo largo del trabajo. Aparte de este repositorio, también se ha abierto un segundo repositorio [**?**] asociado a la oficina del software libre (OSL) para subir el código una vez terminado como parte del programa de apoyo a trabajos finales libres (PATFL) [**?**] de la ULL.

Mediante el uso de este repositorio, el alumno ha conseguido ampliar sus conocimientos en Git y familiarizarse con la interfaz de GitHub. Previamente se había utilizado como repositorios GitLab, SVN y RTC en otros proyectos, por lo que no ha sido una complicación mayor utilizar este sistema.

2.2. Tecnologías utilizadas

A continuación se revisan las distintas tecnologías utilizadas en el desarrollo de la aplicación.

2.2.1. El Sistema Operativo Android

Android es un sistema operativo que emplea Linux en la interfaz del hardware. Los componentes del SO subyacentes se codifican en C o C++ pero las aplicaciones se desarrollan en Java. De esta manera Android asegura una amplia operatividad en una gran variedad de dispositivos debido a dos hechos: la interfaz en Linux ofrece gran potencia y funcionalidad para aprovechar el hardware, mientras que el desarrollo de las aplicaciones en Java permite que Android sea accesible para un gran número de programadores conocedores del código.

Este SO fue diseñado principalmente para dispositivos móviles con pantalla táctil: smartphones, tablets y otros dispositivos como televisores o automóviles. Fue desarrollado inicialmente por Android Inc., empresa que fue respaldada económicamente por Google y más tarde adquirida por esta misma empresa.

Actualmente tiene una gran comunidad de desarrolladores creando aplicaciones para extender la funcionalidad de los dispositivos. A fecha de hoy existen más de un millón de aplicaciones disponibles para la tienda oficial de Apps de Android, Google Play [**?**] sin tener en cuenta las aplicaciones de otras tiendas no oficiales, como por ejemplo, la tienda de aplicaciones de Samsung Apps [**?**].

2.2.2. Los Beacons

Los *"Beacons"* [**?**] (la traducción del término sería a *"balizas"* o *"faros"*) son una tecnología emergente que desde hace algunos años intenta abrirse paso en el mercado. Como su propio nombre indica, estos dispositivos intentan ser un mecanismo de guía, dando una solución al posicionamiento en interiores, donde otras tecnologías, como el GPS [**?**] o el Wifi dejan de funcionar o resultan imprecisas. Sin embargo, estos no son los únicos usos de los beacons, actualmente muchas empresas están ampliando sus usos a otros campos.

A continuación se intentará responder a las preguntas más frecuentes que nos pueden surgir con respecto a esta tecnología:

- ¿Qué es un beacon?
- ¿Cómo funcionan estos dispositivos?
- ¿Qué rango de alcance poseen?
- ¿Con qué dispositivos móviles son compatibles?
- ¿Qué ventajas y desventajas tienen con respecto a otras tecnologías?
- ¿Qué usos se le ha dado a esta tecnología hasta ahora?

¿Qué es un Beacon?

Para quienes no conozcan este término, en el marco en el que nos movemos, hace referencia a un pequeño dispositivo (sus tamaños varían de uno a otro, pero siempre de tamaño reducido) que emite señales de onda corta utilizando la tecnología Bluetooth [**?**]. Estas señales contienen una pequeña cantidad de información que es recibida por dispositivos móviles con tecnología Bluetooth dentro de un rango de cobertura variable dependiendo del beacon y su configuración. Normalmente, la intensidad de esta señal y su frecuencia son configurables.

El funcionamiento de un beacon es sencillo: El dispositivo emite una señal ininterrumpida (véase Figura **??**) que es captada por los dispositivos móviles dentro de su radio de cobertura. La señal nos ofrece información que sirve para detectar y

Figura 2.2: Uno de los beacons de la compañía Estimote

Figura 2.3: Uno de los beacons de la compañía Estimote en formato Pegatina

Figura 2.4: Representación de un beacon emitiendo mediante Bluetooth a un dispositivo móvil

localizar estos dispositivos. Esta señal es captada por una aplicación previamente instalada en un dispositivo móvil que esté programada para recibirla y reaccionar de manera acorde a la información recibida.

Hay que tener en cuenta que esta señal es unidireccional: los beacons son capaces de enviar pero no están preparados para recibir. También hay que tener en cuenta, que la mayoría de los beacon actuales en el mercado transmiten información preconfigurada, confiando en la aplicación móvil para utilizar la información; sin embargo es muy posible que esto cambie en un futuro, ampliando las posibilidades de los beacons.

¿Como funcionan estos dispositivos?

Los beacons usan Bluetooth Low Energy (BLE) [**?**], una versión del protocolo Bluetooth diseñada para usar mucha menos energía y enviar menos información.

Los beacons funcionan con baterías cuyo tiempo de vida depende de la configuración establecida, teniendo en cuenta la emisión de la señal (intensidad y frecuencia) y tiempo de hibernación. Sus tiempos de vida son variables, pudiendo durar desde un mes hasta varios años.

Figura 2.5: Interior de un beacon de Estimote

Independientemente de lo que se pueda pensar, los beacons en sí mismos no transmiten información significativa, transmiten identificadores cortos (a modo de información configurable), que son interpretados por una aplicación que sepa lo que ha de hacer cuando detecte esta información y que es la que se encarga de procesarla y realizar la acción pertinente.

Este identificador se divide en tres partes:

- *"'UUID"'* [**?**] : corresponde con una ID dada por el fabricante e identifica el beacon en cuestión, nosotros utilizaremos el término MAC para referirnos a este identificador.
- ID Superior : configurables y utilizadas con un significado específico que puede identificar una acción o parámetro.
- ID Inferior: configurables y utilizadas al igual que la superior con un significado específico que se puede usar para identificar una acción o parámetro.

¿Qué rango de alcance poseen?

Actualmente los beacons en el mercado presentan un rango de aproximadamente 70 metros de radio sin obstáculos. Está demostrado que este rango disminuye significativamente al atravesar paredes de metal o ladrillo; otros materiales disminuyen en menor medida el rango.

Figura 2.6: Números identificativos de los beacons

Las aplicaciones que trabajan con beacons suelen definir acciones en tres rangos de distancia (véase Figura **??**) principalmente:

- Lejos: diseñado para que el dispositivo móvil pueda lanzar una acción cuando el usuario se encuentre en el rango exterior de un beacon, es decir, cuando se entra en el rango del beacon.
- Cerca: diseñado para que el dispositivo móvil pueda lanzar una acción cuando el usuario se encuentre en el rango interior del beacon.
- Inmediato: diseñado para que el dispositivo móvil pueda lanzar una acción cuando el usuario se encuentre muy cercano al beacon.

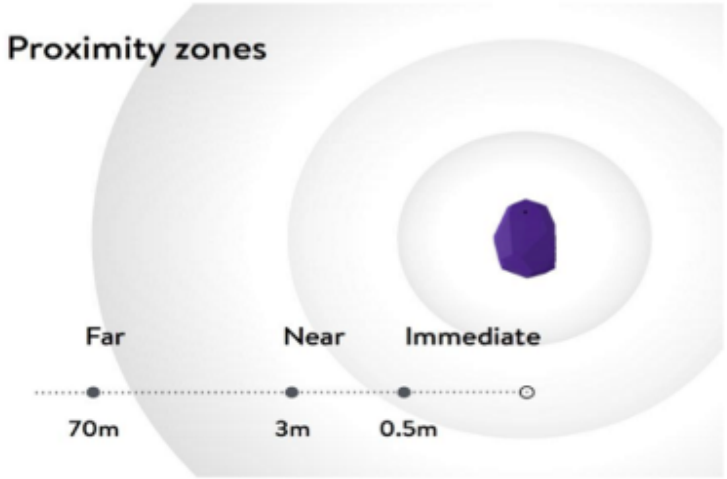


Figura 2.7: Ejemplificación del rango de un beacon

Sin embargo, esto depende de como esté diseñada la aplicación, es posible lanzar acciones a una distancia determinada sin tener en cuenta estos rangos mencionados anteriormente, ya que en todo momento es posible conocer la distancia a la que nos encontramos del beacon.

¿Con qué dispositivos funcionan?

Las beacons son compatibles con todos los dispositivos que soporten el protocolo Bluetooth Low Energy, pero para que las señales de los beacons sean detectadas por un dispositivo, se ha de tener activado el Bluetooth.

En dispositivos con IOS7 [**?**] o superior el dispositivo móvil puede estar constantemente buscando dispositivos BLE. El sistema operativo se encarga de activar las aplicaciones implicadas cuando registra la entrada de un beacon en su rango incluso estando cerradas las aplicaciones.

En dispositivos Android [**?**] el sistema operativo no está preparado para escanear dispositivos BLE, por lo que son las aplicaciones las que tienen que encargarse de escanear las proximidades buscando beacons. Esto supone que las aplicaciones tienen que estar funcionando y despiertas (aunque sea en segundo plano). Existen librerías que solucionan esta limitación haciendo que la aplicación escanee cada cierto tiempo. Sin embargo, no es muy eficaz y suele tener incompatibilidades con el sistema operativo. Un ejemplo de estas incompatibilidades lo podemos ver en el hilo de discusión [**?**] de los desarrolladores de la librería AltBeacon que se ha utilizado para el desarrollo de `BulletPoint` y que mencionaremos más tarde.

Por último en dispositivos Windows Phone [**?**] o Blackberry [**?**] existen diferentes niveles de compatibilidad pero en los que soportan BLE, su funcionamiento es similar al de los dispositivos Android, por lo que no nos pararemos a analizarlo.

¿Qué ventajas y desventajas tienen con respecto a otras tecnologías?

A la hora de hablar de los beacons existen una serie de ventajas pero también podemos encontrar algunas desventajas que revisaremos a continuación.

Las principales ventajas que se distinguen a la hora de hablar de los beacons son las siguientes:

- A diferencia de la tecnología GPS, la activación del Bluetooth consume mucha menos batería.
- Las aplicaciones desarrolladas suelen ser dependientes de la red de datos al necesitar información.
- A diferencia de la tecnología GPS, sigue funcionando con precisión en el interior de los edificios.

En cuanto a las desventajas, podemos destacar las siguientes:

- Dependen de aplicaciones instaladas en el dispositivo móvil para funcionar.
- Es necesario tener el Bluetooth activado, lo que consume batería durante el tiempo que esté activado.

- Su utilidad depende de la voluntad de terceros de utilizar estos dispositivos, configurarlos y distribuir las aplicaciones.

¿Qué usos se le ha dado a esta tecnología hasta ahora?

Por ahora esta tecnología se ha utilizado en entornos muy diversos y con distintas funcionalidades. Entre los más conocidos podríamos destacar los siguientes:

Clevedon School

Este ejemplo es bastante significativo para nosotros ya que se aplicó en el mismo entorno en el que queremos trabajar, una institución de enseñanza universitaria. Después de desplegar cerca de 1200 iPads la universidad de Clevedon (Figura **??**) utilizó esta tecnología junto con su aplicación universitaria ya existente.

Figura 2.8: La aplicación de Clevedon School

Han sido capaces de crear una manera fácil para que los profesores puedan añadir recursos que se envían automáticamente a los alumnos transitando diferentes zonas en diferentes horarios. Para realizar este trabajo de manera eficiente fue necesaria la creación de una interfaz para la gestión de los recursos en las diferentes beacons.

Esta interfaz junto con la aplicación móvil es capaz de:

- Programar los recursos para distribuirse a una hora del día especificada.
- Programar el material para ser distribuido en un momento determinado durante una clase o evento.
- Poner los recursos a disposición del alumnado que se encuentre en una localización específica.

Utilizando estos tres recursos, la aplicación, la interfaz y los beacons han sido capaces de crear un entorno interactivo y eficiente motivando tanto al profesorado como al alumnado.

Cleveland Cavaliers Stadium y Levi's Stadium

Dos de los ejemplos más conocidos han sido los despliegues que se han realizado en estos dos estadios (Ver Figura **??**).

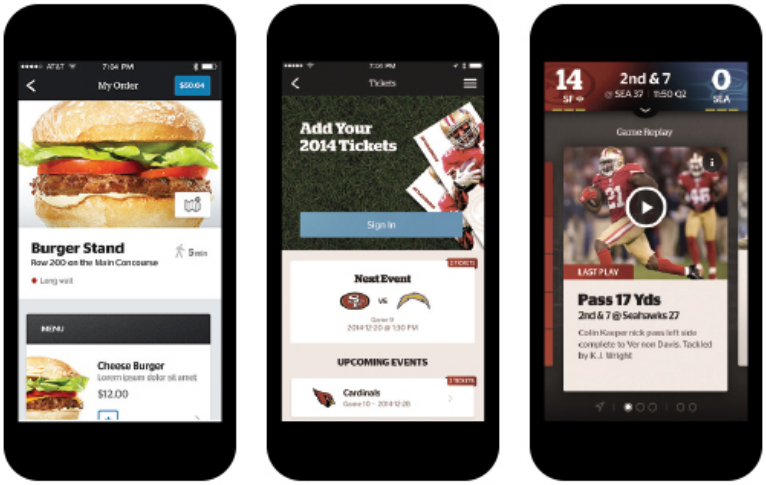


Figura 2.9: La aplicación de Levi's Stadium

Por un lado tenemos el despliegue del estadio de Levi's , cuya intención ha sido la de ayudar a sus seguidores a navegar por el estadio dadas sus dimensiones. En este caso los beacons (de la compañía Aruba Networks) se utilizan en conjunto con puntos de acceso y repetidores situados por toda la infraestructura de manera que queda el estadio cubierto. Con la aplicación los fanes también son capaces de ver repeticiones de las jugadas y pedir comida directamente desde sus dispositivos móviles.

Un punto importante de este despliegue ha sido la monitorización continua del funcionamiento de los beacons, incluyendo si están en funcionamiento o necesitan batería nueva. Los beacons son también más económicos que los puntos de acceso WiFi, lo cual les ha beneficiado.

En el caso del estadio de Cleveland, los beacons se encargan de proveer al usuario de información personalizada dependiendo del lugar y la hora. En algunos casos vídeos, ofertas promocionales y contenido adicional.

Orlando Int'l Airport

Otro despliegue exitoso de esta tecnología ha sido en el aeropuerto internacional

de Orlando, donde mediante el uso de los beacons y de una aplicación móvil propia han sido capaces de proporcionar una serie de funcionalidades de vital importancia en una infraestructura como el aeropuerto:

- Navegación paso por paso a través de cerca de 1000 establecimientos o servicios dentro del aeropuerto.
- Actualizaciones inmediatas de la información de los vuelos.
- Instrucciones a puntos de interés críticos como puntos de recogida de equipaje, puertas de embarque o puestos de información.

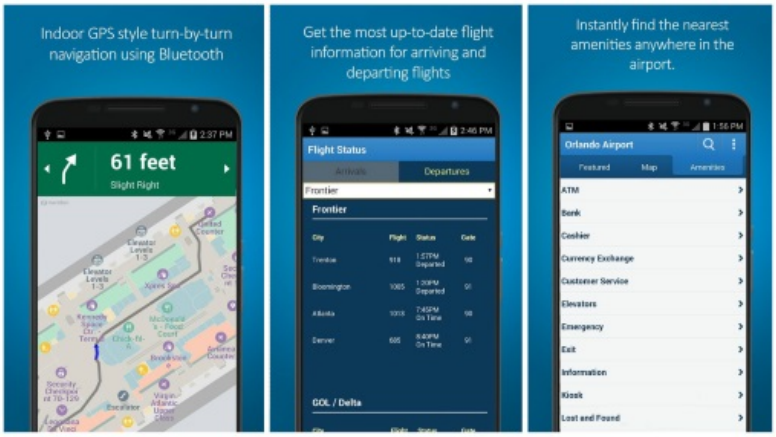


Figura 2.10: La aplicación del Aeropuerto Internacional de Orlando

El siguiente punto sería ampliar la opción a los establecimientos de ofrecer anuncios o promociones, opción que mantienen abierta y no se descarta en un futuro.

Esta información ha sido extraída de: [**?**]

2.2.3. CouchBase Server

Couchbase Server [**?**] es una base de datos NoSQL [**?**] con una arquitectura distribuida orientada al rendimiento, escalabilidad y disponibilidad. Permite desarrollar aplicaciones de manera sencilla y rápida combinando la flexibilidad del JSON [**?**] y la tecnología NoSQL.

¿Por qué utilizar Couchbase Server?

Hemos decidido utilizar esta tecnología por su flexibilidad y potencia para almacenar documentos fácilmente. Además resulta muy sencillo integrarla con

la tecnología móvil mediante el uso de una base de datos reducida dentro del dispositivo móvil (véase Figura **??**). La sincronización con la base de datos principal del servidor se realiza mediante una puerta de sincronización.

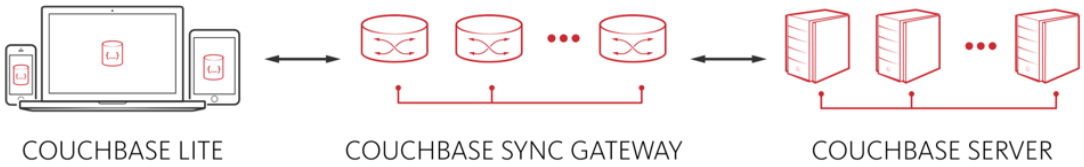


Figura 2.11: Sincronización de CouchBase Server con CouchBase Lite mediante Sync Gateway

En este caso se ha configurado el servidor en un ordenador portátil haciendo uso de las indicaciones de la página web del producto [**?**]. Se procederá a desglosar brevemente los pasos seguidos a la hora de configurar el servidor.

Configuración de la arquitectura

Para configurar el servidor se han seguido los siguientes pasos:

- Descargar la versión Community del producto desde la página web siguiendo el enlace [**?**].
- Seguir los pasos de la página de desarrolladores para la instalación y configuración [**?**].
- Una vez configurado CouchBase Server procederemos a descargar Sync Gateway que será el servicio que se encargue de sincronizar el contenido del nuestra aplicación al servidor, para ello lo descargaremos de la página al igual que el servidor siguiendo el enlace [**?**] .
- Para vincular el servidor con Sync Gateway es necesario hacer uso de un fichero de configuración con el que lanzaremos el servicio Sync Gateway.
- Una vez configurado Sync Gateway, ya tenemos el canal de configuración entre el servidor y la aplicación, para utilizar la base de datos móvil seguiremos los pasos desglosados en: [**?**] .

En este caso se ha tenido que conectar el dispositivo al ordenador. Tanto el ordenador como el dispositivo móvil han de estar en la misma red y hemos utilizado la dirección IP de la máquina para realizar las peticiones del móvil al servidor (alojado en el portátil) al utilizar la API. De esta manera se ha comprobado el funcionamiento del servidor, del servicio de sincronización y de la base de datos versión Lite en el dispositivo móvil.

2.2.4. La librería AltBeacon

¿Qué es AltBeacon?

Se puede definir AltBeacon como una especificación que:

- Define el formato del protocolo publicitario que los beacons transmiten a través de Bluetooth Low Energy (BLE).
- Intenta crear un mercado abierto y competitivo para implementaciones usando proximidad con los beacons.
- Puede ser utilizada gratuitamente, sin cuotas ni compromisos.
- No favorece a ningún proveedor sobre otro. Las limitaciones vienen determinadas por los estándares técnicos de cada uno.

A continuación se profundizará en el funcionamiento y configuración de la librería AltBeacon, que cumple con la especificación AltBeacon y que se ha utilizado para trabajar en Android.

Configuración

Para trabajar con esta librería en Android Studio solo hemos tenido que importarla mediante el uso de Gradle a nuestro proyecto como se explica en [**?**].

También es posible descargarla desde la página oficial, donde además podremos encontrar diferentes versiones de la misma. Para ello podemos hacer uso del siguiente enlace [**?**] y seleccionar la versión deseada.

Funcionamiento

La funcionalidad de esta librería se centra en dos elementos principales:

- Monitorización (*"Monitoring"*) que sería algo como supervisar, saber qué beacons se encuentran en una región o si ha entrado o salido un beacon de una región.
- Rastreo (*"Ranging"*), que permite saber a que distancia se encuentran los beacons en todo momento dentro de una región.

Utilizando estas dos funcionalidades la aplicación es capaz de controlar, monitorizar y rastrear los distintos beacons en una determinada región. En la página web de la librería [**?**] se pueden encontrar ejemplos básicos de cómo se realizan estas dos funciones en la sección *"Samples"*, además en la sección *"Documentación"* se pueden encontrar también algunos artículos, que pueden resultar interesantes dependiendo del tipo de aplicación que se esté desarrollando.

2.2.5. El algoritmo de trilateración

Durante el desarrollo de este trabajo se nombrará en diferentes puntos el término trilateración. A continuación explicaremos qué es la trilateración y cómo se ha aplicado el algoritmo al trabajar con beacons en `BulletPoint`.

¿Qué es la trilateración?

La trilateración es un método matemático para determinar las posiciones relativas de objetos utilizando la geometría de triángulos de forma análoga a la triangulación. La triangulación utilizada en la tecnología GPS, utiliza medidas de ángulo junto con al menos una distancia para calcular la posición del sujeto.

A diferencia de ésta, aunque muchas veces se confundan los términos por la similitud, la trilateración utiliza las localizaciones conocidas de dos o más puntos de referencia y las distancias entre el sujeto y cada punto de referencia (véase Figura **??**). Para determinar de forma única y precisa la localización relativa de un punto en un plano bidimensional utilizando solo la trilateración, se necesitan generalmente al menos 3 puntos de referencia.

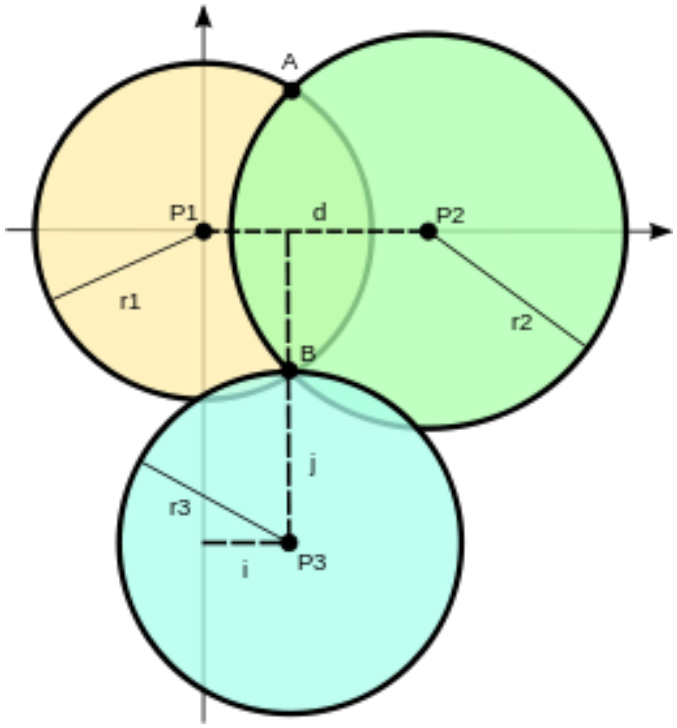


Figura 2.12: En el algoritmo de trilateración se utilizan las localizaciones conocidas de varios puntos de referencia y las distancias entre el sujeto y cada punto de referencia.

```
1  public Point calculatePosition(double[][] positions, double[] distances ) {
2        //Call trilateration algorithm
3        TrilaterationFunction trilaterationFunction = new TrilaterationFunction(positions,
             distances);
4        //Use non linear least squares solver using levenberg-marquardt method.
5        NonLinearLeastSquaresSolver nlSolver = new NonLinearLeastSquaresSolver(
             trilaterationFunction, new org.apache.commons.math3.fitting.leastsquares.
             LevenbergMarquardtOptimizer());
6
7        LeastSquaresOptimizer.Optimum optimum = nlSolver.solve();
8        // The result is the center of the position of the User, given in X Y coordinates of a
             Point.
9        double[] centroid = optimum.getPoint().toArray();
10       return new Point(centroid[0],centroid[1]);
11 }
```

Listado 2.1: Usamos el algoritmo de trilateración, utilizando las posiciones de los beacons y las distancias. El algoritmo los resuelve y devuelve una posición "X" e "Y" que transformamos en un objeto de la clase *Point*

En el desarrollo de la aplicación se ha utilizado un algoritmo de trilateración implementado en Java que utiliza el algoritmo de Levenberg-Marquardt [**?**] de Apache Commons Math [**?**] tal y como explica el autor en [**?**].

En nuestro caso, las posiciones de los beacons son determinadas en una imagen separados entre sí a menos de 70 metros. Los beacons se sitúan en estas posiciones elegidas en la imagen y se obtiene una posición "X" e "Y" que queda registrada para cada beacon en la aplicación (Listado **??**).

Las distancias medidas hasta la posición de cada beacon vienen dadas por la distancia registrada por la aplicación. Aparte de estos dos datos, para obtener una posición "X" e "Y" donde representar el punto, es necesario aplicar una escala que convierta los metros a píxeles. Tras la aplicación del algoritmo de trilateración, se consigue como resultado la posición del usuario en la imagen representado por "X" e "Y" en píxeles.

Capítulo 3

Beacons en entornos universitarios

En este capítulo realizaremos un análisis de los posibles casos de uso de la tecnología beacon en el ámbito universitario.

3.1. Aplicaciones móviles en entornos universitarios

Actualmente las posibilidades de las aplicaciones móviles para entornos universitarios se presentan amplias. Cada universidad intenta tener su propia aplicación siguiendo un patrón similar, puesto que todas parten de necesidades similares. Realizando una investigación general de las aplicaciones disponibles en el mercado observamos que estas aplicaciones se centran en ofrecer servicios propios (servicio de correo, Moodle, chat entre usuarios, etc.), mantener al alumnado informado y agilizarle los trámites mayoritariamente.

En un principio, estas aplicaciones se enfocaban a atraer estudiantes, centrándose en la calidad de la universidad y mostrando las posibilidades que ofrecían. Sin embargo, con el paso de los años y el desarrollo creciente de las aplicaciones móviles, se muestra un cambio en esta estrategia. Ahora las aplicaciones tienen una doble función y no sólo buscan el acceso de nuevos estudiantes, sino que también intentan mejorar la experiencia del alumnado ya matriculado y acercar a los nuevos a la experiencia de la universidad. Algunos ejemplos posibles los encontramos en el marketplace de Google [**?**, **?**, **?**]

Uno de los hechos que podemos observar es que las universidades están intentando obtener una solución rápida para desarrollar su app. Una de estas soluciones es la creación de plantillas web optimizadas de su sitio web, lo que podemos considerar una opción rápida con un coste bajo.

Hoy en día casi todos los estudiantes tienen acceso a un dispositivo móvil y

cuentan con una tarifa de internet. En un futuro próximo con la aparición de estos dispositivos ya nos surge la pregunta ¿Serán capaces estos dispositivos de transformar la educación? y en caso afirmativo ¿De qué manera?

3.2. Posibles casos de uso de la tecnología beacon en entornos universitarios

Como se mencionaba antes, una de las posibilidades que se presentan para explotar esta tecnología se encuentra en las instituciones de enseñanza, las cuales podrían utilizar los beacons para facilitar a su alumnado, profesorado y demás personal involucrado una serie de servicios de gran utilidad.

Sin embargo, para utilizar esta tecnología es necesario cumplir una serie de condiciones:

- Tener instalada la aplicación en su dispositivo móvil.
- Tener activado el protocolo Bluetooth.
- La aplicación ha de estar activa.
- Los beacons han de estar desplegados y configurados correctamente en lugares clave donde el rango sea óptimo.

En el caso de dispositivos Apple no es necesario tener activado el Bluetooth ya que el SO se encarga de captar las señales BLE. Tampoco es necesario que la app esté despierta ya que nuevamente el SO se encarga de despertar a la aplicación involucrada. Sin embargo Apple no ha desarrollado un IBeacon físico aún, aunque en un futuro, se espera que sus dispositivos móviles puedan funcionar como un beacon bidireccional. Cabe destacar que existen librerías que se encargan de realizar estas mismas funcionalidades para mantener el móvil despierto a la escucha de posibles beacon para otros sistemas operativos, pero a diferencia de Apple esta funcionalidad no está integrada directamente en el SO por lo que no presenta el mismo nivel de control.

Asimismo, podemos afirmar que prácticamente la mayoría de las universidades cuentan con una disposición amplia en lo que se refiere a servicios y despliegue de medios. Como ejemplo, podemos tomar la Universidad de la Laguna, la cual cuenta con una red WiFi con un rango de cobertura casi completo en sus instalaciones y una amplia carta de servicios disponibles para sus alumnos. Además cuenta con una serie de beacons, que podrían ser instalados fácilmente en lugares estratégicos.

Partiendo de esta base, procederemos a explorar posibles casos de uso para los beacons tomando el contexto universitario y del alumnado como referente:

3.2.1. Guía a través del campus de la universidad

Este caso de uso cubre la funcionalidad principal de un beacon. El posicionamiento y guía tanto en exteriores como en interiores.

Como interesados podríamos destacar:

- Personal invitado a jornadas o eventos en instalaciones de la universidad.
- Alumnado de intercambio en programas internacionales.
- Estudiantes de nuevo acceso.
- Personas con discapacidad.

Figura 3.1: Servicios de localización a través de la Universidad

El funcionamiento sería el siguiente: el usuario transita por las inmediaciones del campus universitario. El usuario accede al sistema de navegación dentro de la aplicación, la cual le muestra entonces su ubicación como un punto de color sobre el mapa del campus. Este mapa tiene marcados puntos de interés que contienen información de diferente tipo dependiendo del punto marcado: nombre, historia, página web, teléfono de contacto, trámites asociados... son algunos de los datos que podría mostrar. El mapa se va actualizando dependiendo de la posición del usuario permitiendo volver a la vista más alejada en cualquier momento para una visualización más general.

3.2.2. Descarga automática de material

Este caso de uso resultaría muy útil para personal lectivo y para estudiantes, los cuales accederían de manera más sencilla al material disponible. También sería aplicable para ponentes de charlas quienes no tendrían que alojar sus apuntes en alguna plataforma externa o llevarlos consigo en un almacenamiento externo para compartirlo al finalizar la actividad.

El funcionamiento sería el siguiente: el profesor/a o ponente lleva consigo un beacon y sus estudiantes u oyentes tienen instalados en sus dispositivos la aplicación. El profesor es capaz de introducir en su aplicación con el perfil de profesor (el ponente con su correspondiente perfil), indicaciones del material a utilizar en el evento. El profesor/a carga consigo el pequeño dispositivo e indica al alumnado que conecten el Bluetooth y abran la aplicación. Al entrar en el rango, la app pedirá permiso al alumno para descargarse el contenido indicado por el profesor. Si el alumno acepta, la aplicación pasaría a abrir el contenido indicado por la página correspondiente.

3.2.3. Acceso al parking y recuento de número de plazas disponibles

Este caso de uso proporcionaría información muy útil a los usuarios del parking de la universidad, informando del número coches estacionados en el parking y de las plazas restantes a ocupar en tiempo real.

El funcionamiento sería el siguiente: el personal de la ULL tendría la aplicación en su móvil; al acercarse a la barrera del parking, el usuario activaría el Bluetooth de su móvil. La app registraría un nuevo punto entrando en el rango de acción del parking. La aplicación comprobaría conectando con un servidor institucional que el usuario está autorizado a entrar en el parking y procedería a abrir la puerta del parking dejando entrar al vehículo. Cuando el vehículo saliese del rango del beacon por el rango interior, la aplicación registraría entonces un nuevo acceso al parking y contabilizaría otro vehículo dentro de parking. Al salir del parking el proceso sería el mismo, por lo tanto la aplicación sería capaz de informar al usuario de las plazas ocupadas en tiempo real.

3.2.4. Gestión de eventos e información, entrada automática

El funcionamiento sería el siguiente: el alumnado transita los interiores de la Universidad de camino a sus clases. Los beacons están desplegados en las inmediaciones de lugares de interés, tipo aulario, paraninfo, clases que se utilicen a modo de salas de reuniones o seminarios. Al pasar por las inmediaciones de estos lugares de interés, la aplicación sería capaz de proporcionar al usuario información

de diversa índole: ponentes, tema de la charla, acceso, teléfono de contacto u otra información similar. Al mismo tiempo, la aplicación también cuenta con un tablón donde se muestran posibles eventos futuros. Estos eventos pueden ser muy variados y corresponder a diferentes tipos de actividades. Al mismo tiempo se podría confirmar la entrada al evento en el caso de haberla, mediante un código de acceso identificativo generado al realizar la inscripción en el evento.

Figura 3.2: La aplicación presenta información al usuario sobre el evento que está teniendo lugar.

3.2.5. Despacho del profesorado e información

El funcionamiento podría ser abarcado de dos maneras. Por un lado, podría utilizarse para proveer al alumnado de información acerca del grupo de despachos, aclarando que profesorado tiene el despacho en la zona, horario de tutorías, correo electrónico de contacto, horario de corrección de exámenes, etc. De esta manera el alumnado al acercarse a la zona sería capaz de acceder a información de todo el profesorado, o si buscase a alguno en particular, la aplicación le daría la opción de elegir su nombre de una lista y simplemente comprobar si tiene su despacho en esa zona.

Por otro lado, este caso de uso podría ampliarse para proporcionar una información adicional, comprobando si el profesorado está en la zona en ese momento y se encuentra disponible. El profesor tendrá un perfil de la aplicación con un código identificativo que le distingue de los demás profesores. Estos datos se guardarían en un servicio externo, y la app sería la encargada acceder a este servicio consultando las entradas y salidas del profesorado. De esta manera el alumnado sería capaz de saber cuando el profesor se encuentra disponible en las cercanías mediante la aplicación. En cuanto al estado de disponibilidad, sería un dato que actualizaría el profesor desde su perfil en la aplicación.

3.2.6. Información y descuentos para usuarios de la app

Este caso de uso no solo dependería de la universidad, sino de establecimientos comerciales interesados. La idea sería la siguiente: la universidad en colaboración con un establecimiento comercial le entrega un beacon. La aplicación contaría con un perfil para el dueño del establecimiento, donde sería capaz de introducir información que desea que se muestre al usuario al pasar cerca de su establecimiento, mensajes de información, descuentos u ofertas especiales por ejemplo.

El usuario al pasar por las inmediaciones del establecimiento recibe en su aplicación una notificación del establecimiento con la información introducida por el dueño anteriormente. Al aceptar la notificación, el usuario podría ser redirigido a la página web del establecimiento para ver las ofertas. En cualquier caso el establecimiento ha conseguido captar la atención de un posible cliente, y el usuario se beneficiaría de ofertas y descuentos.

3.2.7. Control de asistencia

Este caso de uso puede ir ligado al de descarga automática de material (véase epígrafe **??**), el funcionamiento sería el siguiente: el alumno conectaría el Bluetooth de su móvil al iniciar la clase. En ese momento la aplicación detectaría los dispositivos y permitiría al usuario registrar su asistencia a la clase en el horario lectivo. El profesorado sería capaz en todo momento, desde el perfil del profesor, de consultar la asistencia de los alumnos. Si lo unimos a la descarga automática de material, proporcionaría comodidad tanto al alumnado como al profesorado. Sin embargo, un impedimento podría ser el rango del beacon o la necesidad de activar el Bluetooth ya que, si el alumno no tiene batería en el móvil, habría que recurrir a un método secundario.

3.2.8. Control de acceso a instalaciones

El control de acceso a las aulas y edificios puede ser un tema abordable mediante el uso de estos dispositivos. Los lectores de tarjetas pasarían a ser algo innecesario. El alumno simplemente tendría que activar el Bluetooth cerca del punto de entrada, se comprobaría su identidad y se procedería a darle a acceso o a informarle de su falta de permiso. Los permisos para acceder a estos puntos podrían quedar almacenados en alguna plataforma donde se puedan visualizar.

3.2.9. Biblioteca informativa

Otro posible uso de la tecnología beacon tiene que ver con las bibliotecas o lugares de almacenamiento de material. El estudiante se acercaría a la biblioteca

buscando un libro específico. Partimos de que en la app estaría registrada la localización de los libros disponibles en los estantes. De esta manera la aplicación indicaría al alumno o alumna la posición del libro que busca. Para lugares amplios donde hay gran cantidad de material (véase la Figura **??**), incluso podría guiar al usuario por las instalaciones hasta llegar a su objetivo, informarle del número de ejemplares disponibles o de la fecha prevista de entrada de algún título.

Figura 3.3: La biblioteca de la Universidad de Salamanca contiene más de 1.000.000 de ejemplares lo que puede dificultar la localización de algunos títulos.

3.2.10. Actividades interactivas por el campus, jornadas de acogida u otros eventos

En un ámbito más recreativo, se podría tener en cuenta el uso de los beacons para organizar juegos o actividades de ocio para el alumnado. Por ejemplo, rutas a través del campus con adivinanzas o puzzles relacionados con diferentes temáticas, lo que fomentaría el trabajo en equipo. Al mismo tiempo se podría aplicar algún tipo de recompensa para los ganadores, descuentos o bonos tramitados por medio de la aplicación. Estos eventos dependerían de los organizadores y el alumnado tendría que registrarse con su identificador.

3.2.11. Localización de transporte público, horarios e información de la parada

Uno de los transportes más utilizados por el alumnado de la universidad es el autobús. Este medio de transporte puede llegar a ser el día a día de muchos de los estudiantes que no cuentan con vehículo propio o que simplemente prefieren utilizar este medio de transporte.

Este caso de uso contempla lo siguiente: el estudiante llega a una parada de autobús y utilizando su dispositivo móvil, consulta los autobuses que van a pasar por la parada. La aplicación le permite obtener información de los diferentes autobuses, el itinerario y el tiempo restante para que llegue a la parada. Asimismo enlazaría con la web de la compañía de transporte para información adicional sobre líneas e itinerarios.

Capítulo 4

La aplicación BulletPoint

Basándonos en los casos de uso revisados en el capítulo anterior, en este capítulo se discutirán los casos de uso que han sido elegidos e implementados en la aplicación `BulletPoint`. Comentaremos la aplicación centrándonos en el desarrollo de la misma, así como en diferentes partes a destacar del código que puedan resultar interesantes.

4.1. Casos de uso elegidos

Como ya hemos mencionado previamente, estos casos de uso se incluyen como parte de la aplicación que se ha desarrollado en este TFG, donde cada caso de uso se considera un módulo. La integración de cada uno de estos módulos con el resto de la aplicación se ha realizado mediante el desarrollo de un menú de funcionalidades donde es posible seleccionar qué acción se desea. Para almacenar la información necesaria para algunos módulos, se ha introducido un menú adicional de ajustes. Estos datos se utilizan entre otras cosas para identificar al usuario y confirmar ciertas acciones o dejar constancia de otras.

A continuación se desglosarán los distintos casos de uso en los que tendremos que pensar de ahora en adelante como módulos, junto con la explicación del caso de uso y su funcionamiento se incluirán algunos de los detalles más importantes de la implementación de cada uno.

4.1.1. Localización de transporte público, horarios e información de la parada

Objetivo

El objetivo de este caso de uso es el conocer, en tiempo real, qué autobuses pasan por la parada en la que nos encontramos, hacia dónde se dirigen, y cuánto

tiempo falta para que lleguen a la parada. En caso de necesidad de información adicional, la aplicación está preparada para remitirnos a navegar por la página web [**?**] de la empresa Transportes interurbanos de Tenerife, S.A.(TITSA), donde podemos ver datos adicionales sobre el autobús seleccionado.

Figura 4.1: Información mostrada por TITSA actualmente en su página web.

Despliegue

En este caso, sería necesario colocar un beacon en cada parada de autobuses. De esta manera cada parada de autobús queda identificada por un beacon. Si el usuario se encuentra cercano a dos paradas, la aplicación considera que le interesa la información de la parada más cercana a su posición y le proporcionará información únicamente de esta parada con el objetivo de no confundir al usuario.

Funcionamiento

Mediante el uso de los beacons, permitimos a la aplicación identificar en que parada se encuentra el usuario. La aplicación asocia la MAC **??** de un beacon con el número identificativo de la parada. Este número identificativo de la parada es lo que utiliza TITSA para identificar sus paradas en la API y en toda su web.

La aplicación ha sido programada enlazando los números identificativos de la parada con la dirección MAC de los beacons. En el listado **??** se puede apreciar como se relaciona cada MAC con el número identificativo de la parada.

Cuando la aplicación detecta un beacon, es capaz de reconocer la parada en la que se encuentra el usuario. Una vez la aplicación conoce los datos de la parada, se procede a realizar dos peticiones de comunicación con el servicio de TITSA:

- La primera petición se hace sobre la API de TITSA y nos proporciona la información de los autobuses y el tiempo restante para que llegue dicho autobús a la parada. Cabe destacar que para poder utilizar esta API se contactó con TITSA, cuyo personal nos proporcionó una API KEY para poder trabajar con ella. Podemos ver el método principal en el listado **??**.

```java
//The class BeaconBusStop, stores the information that relates each beacon to a stop identifier.
public class BeaconBusStop {

    private static final Map<String, String> stopsMapId;
    static {
        /*Each becon MAC address is associated with the stop ID*/
        Map<String,String> auxMap = new HashMap<String,String>();
        //Intercambiador La Laguna
        auxMap.put("20:C3:8F:F1:AA:11", "2625");
        //Intercambiador Santa Cruz
        auxMap.put("20:C3:8F:F1:52:ED", "7140");
        //Intercambiador Costa Adeje
        auxMap.put("D4:F5:13:7A:1D:24", "7142");

        stopsMapId = Collections.unmodifiableMap(auxMap);
    }

    public static String getStopId(String maccAddress){
        return stopsMapId.get(maccAddress);
    }
}
```

Listado 4.1: *BusBeaconStop*. Esta clase contiene la información que asocia cada MAC con el ID de la parada de TITSA. Añadir un nuevo beacon a una parada implica añadir un nuevo elemento a *stopsMapId*.

- La segunda petición se realiza sobre la web de TITSA directamente. El objetivo de esta petición es obtener para cada autobús la información de su recorrido. Para ello se obtiene el código de la página en HTML extrayendo la información relativa a su itinerario. La información no se encuentra en un formato óptimo en algunos casos, pero esta información proviene de TITSA y ofrece una información necesaria para usuario.

Aparte de esta información, por cada autobús se incluye un enlace a modo de botón que remite al usuario a la página web de TITSA con el identificador de la línea. Así el usuario puede obtener más información adicional en caso de precisarla.

En un principio nuestra idea era utilizar simplemente la API de TITSA, puesto que se creía que proporcionaría toda esta información; sin embargo, en algunos casos los destinos no se correspondían con la realidad y las rutas aparecían erróneas. Ante esta situación, se contactó con personal de TITSA involucrado en el desarrollo de esta API quien nos comentó que esto ocurría en algunos casos por la manera en la que estaba planteada la API.

La solución que se adoptó, fue utilizar ambos métodos para obtener la información completa. Dichos métodos serán desglosados a continuación:

En el primer método, utilizando una librería de peticiones HTTP, se envía una consulta por GET a la API de TITSA que devuelve una respuesta en XML. Esta respuesta se parsea con un handler de XML (véase Listado **??**).

```java
package com.bulletpoint.ull.bulletpoint.busclasses;

public class Arrival {
    //This class is used to store the data comming from TITSA.
    private String stopCode; //Stop code, each beacon is associated with one of these.
    private String stopName;
    private String destination; //It's comming faulty from TITSA, so we will parse TITSA webpage
         looking for it.
    private String hour;
    private String travelId;
    private String lineNumber;
    private String minutesForArrival;


    public String getStopCode() {
        return stopCode;
    }

    public void setStopCode(String stopCode) {
        this.stopCode = stopCode;
    }

    public String getStopName() {
        return stopName;
    }

    public void setStopName(String stopName) {
        this.stopName = stopName;
    }

//...
}
```

Listado 4.2: La clase *Arrival* donde quedan contenidos los datos de cada llegada.

```java
public class XmlHandler extends DefaultHandler {

    String elementValue = null;
    Boolean elementOn = false;
    Arrival arrival;
    public static XMLGettersSetters data = new XMLGettersSetters();

    public XmlHandler(){
        data= new XMLGettersSetters();
    }

    public static XMLGettersSetters getXMLData() {
        return data;
    }

    public static void setXMLData(XMLGettersSetters data) {
        XmlHandler.data = data;
    }
    /**
     * This will be called when the tags of the XML starts.
     **/
    @Override
    public void startElement(String uri, String localName, String qName,
                             Attributes attributes) throws SAXException {
        elementOn = true;
        if (localName.equals("llegada"))
        {
            Log.i("INFO", "Creando nuevo elemento llegada...");
            arrival = new Arrival();
        }

    }
    /**
     * This will be called when the tags of the XML end.
     **/
    @Override
    public void endElement(String uri, String localName, String qName)
            throws SAXException {
        elementOn = false;
        /**
         * Sets the values after retrieving the values from the XML tags
         * */
        if (localName.equalsIgnoreCase("codigoparada"))
            arrival.setStopCode(elementValue);
        else if (localName.equalsIgnoreCase("denominacion"))
            arrival.setStopName(elementValue);
        /*else if (localName.equalsIgnoreCase("destinolinea"))
            arrival.setDestination(elementValue); Destination not correct, we will get the route
                from parsing titsa webpage*/
        else if (localName.equalsIgnoreCase("hora"))
            arrival.setHour(elementValue);
        else if (localName.equalsIgnoreCase("idtrayecto"))
            arrival.setTravelId(elementValue);
        else if (localName.equalsIgnoreCase("linea"))
            arrival.setLineNumber(elementValue);
        else if (localName.equalsIgnoreCase("minutosparallegar"))
            arrival.setMinutesForArrival(elementValue);
        else if (localName.equalsIgnoreCase("llegada")) {
            Log.i("Setting arrival:", arrival.toString());
            data.setArrivals(arrival);
        }
    }
}
```

Listado 4.3: El handler se encarga de transformar el fichero XML en elementos de tipo *Arrival* (véase Listado **??**).

```
1  public class HttpClientTitsa {
2      private static final String BASE_URL = "http://apps.titsa.com/apps/apps_sae_llegadas_parada.
           asp?";
3      //...
4      public List<Arrival> getTimetablesBasedOnLineNumber(final String stopNumber) {
5
6          RequestParams paramettersGet = new RequestParams();
7          String relativeUrl = "IdApp=" + APIkey + "&idParada=" + stopNumber;
8          Log.i("INFO", "Realizando peticion: " + BASE_URL + relativeUrl);
9          get(relativeUrl, paramettersGet, new SaxAsyncHttpResponseHandler(new XmlHandler()) {
10             @Override
11             public void onSuccess(int statusCode, Header[] headers, DefaultHandler defaultHandler)
                   {
12                 Log.i("INFO", "Finalizada peticion!");
13             }
14             @Override
15             public void onFailure(int statusCode, Header[] headers, DefaultHandler defaultHandler)
                   {
16                 resultados = null;
17             }
18             @Override
19             public void onPostProcessResponse(ResponseHandlerInterface instance, HttpResponse
                    response) {
20                 super.onPostProcessResponse(instance, response);
21                 resultados = XmlHandler.getXMLData().getArrivals();
22                 Log.i("INFO", "Procesada peticion!");
23             }
24         });
25         return resultados;
26     }
27     //...
28 }
```

Listado 4.4: El método *getTimetablesBasedOnLineNumber()* del cliente de TITSA se encarga de obtener los datos de su API y devolver una lista de llegadas.

Al mismo tiempo, se inicia la petición a la página de TITSA y utilizando la librería Jsoup [**?**], se obtiene el contenido de la página web. De este contenido en HTML, recopilamos únicamente el recorrido de los autobuses y el resto se descarta como podemos ver en el listado **??**. Esta petición es bastante más pesada que la del XML previa, ya que tiene que obtener todo el contenido de la página web en la que aparece el desglose del itinerario en formato HTML. El parseo de la página también es más lento que en el paso previo, ya que debe ir elemento por elemento iterando en los elementos hijos del HTML.

Utilizando ambos conjuntos de datos, se va creando una estructura con los mismos, donde se almacena la información sobre cada elemento. En última instancia vamos añadiéndolos a la vista utilizando un adaptador y una estructura para visualizarlos.

```
1
2  public class HttpClientTitsa {
3    private static final String BASE_URL = "http://apps.titsa.com/apps/apps_sae_llegadas_parada.asp?
         ";
4    //...
5    public static Map<String,String> getCorrectRoute(String stopNumber){
6      //...
7      Map<String,String> destinationMap = new HashMap<String,String>();
8      Document doc = Jsoup.connect("http://www.titsa.com/correspondencias.php?idc=" + stopNumber).
           get();
9      Log.i("INFO","Realizando peticion");
10     Element table = doc.select("table").get(0);
11     Elements rows = table.select("tr");
12     for (int i = 1; i < rows.size(); i++) { //first row is the col names so skip it.
13         Element row = rows.get(i);
14         Elements cols =row.select("td");
15         //Second element corresponds to Destination route.
16         String[] elementos = cols.get(1).toString().split("-", 2);
17         destinationMap.put(Html.fromHtml(elementos[0].replaceAll(" ","")).toString(),Html.fromHtml(
               elementos[1].replaceAll(" ","")).toString());
18     }
19     return destinationMap;
20     //...
21 }
```

Listado 4.5: El método *getCorrectRoute()* nos permite obtener el itinerario correcto del autobús.

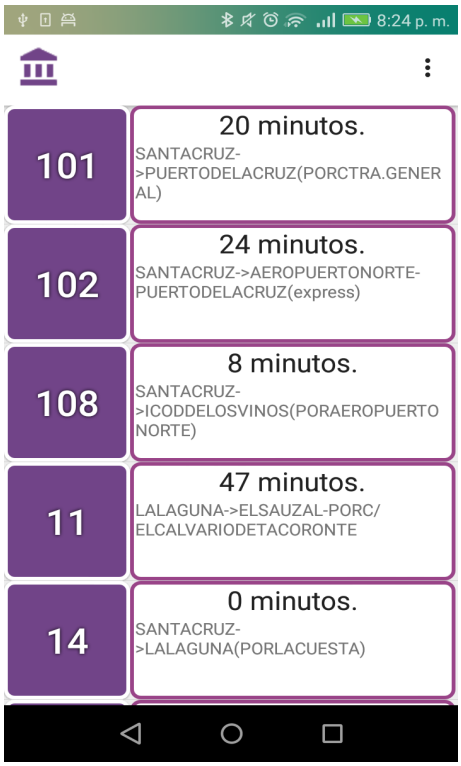


Figura 4.2: `BulletPoint` permite conocer en tiempo real, autobuses que se acercan a la parada, junto con su destino y tiempo de llegada de la parada identificada por un beacon.

Dificultades

Al plantear este caso de uso se han detectado las siguientes dificultades:

- A la hora de presentar los datos, la generación de la estructura fue un proceso con el que no se estaba familiarizado y que llevó más tiempo del previsto inicialmente; sin embargo, ha servido de piedra angular para el desarrollo de los demás casos de uso ya que presenta elementos comunes: peticiones, parseo de datos, visualización de listas, etc.
- Otra dificultad ha sido el no poder obtener todos los datos necesarios con una sola petición y tener que hacer uso de dos peticiones distintas con el tratamiento posterior de los respectivos datos, ya que no eran en absoluto similares.

Ampliación

Ampliar este caso de uso sólo conllevaría asociar los nuevos identificadores de los beacons a nuevos identificadores de parada. Por ello podemos afirmar que sería relativamente sencillo desplegar una red de beacons en las paradas de autobús de TITSA.

4.1.2. Gestión de eventos e información

Objetivo

El objetivo de este caso de uso sigue siendo el de proporcionar información de eventos e información de interés para el usuario; sin embargo, se ha desligado el módulo de entrada automática, al que hacíamos mención en el análisis previo, ya que en la mayoría de los casos esta funcionalidad no es necesaria.

Despliegue

En este caso, sería necesario colocar un beacon en sitios de interés donde puedan tener lugar eventos. Al mismo tiempo, también es necesaria una fuente de datos donde alojar la información de los eventos. De esta manera cada beacon hace referencia a una fuente de información de eventos. Si el usuario se encuentra cercano a dos beacons, la aplicación considera que le interesa la información del beacon más cercano a su posición. Si se desplaza a otra posición y se acerca a otro beacon, la información sería diferente.

```
1  public class RssBeaconInfo {
2
3      private static final Map<String, String> rssMap;
4      static {
5          Map<String,String> auxMap = new HashMap<String,String>();
6          //Deportes y ocio
7          auxMap.put("20:C3:8F:F1:AA:11", "http://eventos.ull.es/rss/category/5/1001/deporte-y-ocio-
               general.rss");
8          //Informatica y telecomunicaciones
9          auxMap.put("20:C3:8F:F1:52:ED", "http://eventos.ull.es/rss/category/13/1001/informatica-y-
               telecomunicaciones-general.rss");
10         //Musica teatro y danza
11         auxMap.put("D4:F5:13:7A:1D:24", "http://eventos.ull.es/rss/category/17/1001/musica-teatro-y
               -danza-general.rss");
12
13         rssMap = Collections.unmodifiableMap(auxMap);
14     }
15
16     public static String getRssLink(String maccAddress){
17         return rssMap.get(maccAddress);
18     }
19 }
```

Listado 4.6: La información de la MAC del beacon queda asociada a un enlace RSS.

Funcionamiento

Mediante el uso de los beacons, permitimos a la aplicación identificar en qué localización se encuentra el usuario. En este caso, hemos utilizado la información RSS del portal web eventosULL [**?**]. Esta página posee diferentes enlaces RSS que actualmente no están separados por ubicación sino por categorías variadas (ocio, eventos, deportes, tecnologías,etc.). Sin embargo lo hemos tomado como ejemplo porque constituye un portal que ya está en funcionamiento y proporciona información sobre eventos actuales relacionados con la Universidad de la Laguna.

La aplicación ha sido programada enlazando la dirección MAC de los beacons con diferentes enlaces RSS de la página web de eventos como podemos apreciar en el listado **??**. Al detectar un beacon, se reconoce a qué información queremos acceder y se procede a realizar una petición utilizando el enlace identificado por el beacon:

La petición se lanza sobre el enlace, el cual nos devuelve un archivo XML que procedemos a tratar para extraer la información que nos interesa. Para no sobrecargar la aplicación con demasiada información del evento, hemos decidido utilizar sólo los datos más relevantes: el título y la fecha del evento. Al mismo tiempo, sobre cada evento se incluye un enlace que, al ser seleccionado, remite a la página web del evento para suministrar información adicional sobre él (localización en un mapa, manera de inscribirse al evento, etc.). Sin embargo, esta información

no está disponible para todos los eventos.

En un principio, se planteó el filtrar los eventos para obtener los cercanos a una posición geográfica; sin embargo, no todos los eventos poseen esta información, y sería necesario realizar una petición por cada RSS para obtener los datos y el proceso sería más lento. Si en el futuro se planteara, cada RSS debería relacionarse con la localización del evento.

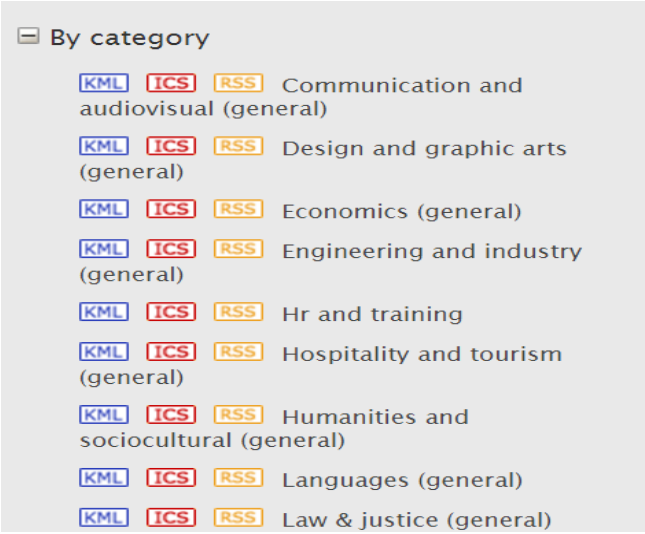


Figura 4.3: Las diferentes categorías de los RSS del portal web de eventos.ull.es

La petición se realiza utilizando la misma librería de peticiones HTTP que en el caso anterior: se envía una consulta por GET al enlace RSS de la página web de eventosULL el cual nos devuelve una respuesta en XML. Esta respuesta se parsea con un handler de XML para seleccionar únicamente los datos que nos interesan. Esta petición es bastante rápida y genera una vista con la información de los distintos eventos. Cada elemento evento de esta lista es capaz de remitir a la página web de eventosULL para recibir información adicional en caso de estar interesados en el evento en cuestión.

Figura 4.4: El portal web de eventos, donde podemos obtener información adicional de un evento concreto.

Figura 4.5: Usando `BulletPoint` podemos conocer eventos asociados a un RSS del portal web.

Dificultades

A la hora de implementar este caso de uso no ha habido grandes dificultades. Sin embargo, la fuente de datos no ha sido la óptima, ya que lo ideal hubiese sido poder filtrar los eventos por localización en lugar de por categorías. Aún así, se ha conseguido el objetivo principal de mostrar diferentes eventos en función del beacon detectado.

Ampliación

Ampliar este caso de uso sólo conllevaría asociar los nuevos identificadores de los beacons a nuevos identificadores de ficheros RSS. Es por ello que podemos afirmar que sería relativamente sencillo desplegar una red de beacons en diferentes puntos del campus universitario asociados a un RSS. En caso de utilizar otro medio al RSS, sólo habría que adaptar la obtención y la visualización del nuevo contenido, modificando la dirección y añadiendo un nuevo parseo.

4.1.3. Control de asistencia

Objetivo

Este caso de uso tiene como objetivo controlar la asistencia de los alumnos a las clases u otro evento que requiera control de asistencia. En un principio se había descartado por las complicaciones que presentaba:

- Era necesario tener un servidor en el que poder almacenar la información de las asistencias con la información de los participantes.
- Se necesitaba un modo de almacenar los datos del usuario para poder identificarle a la hora de registrar la asistencia.
- El área de acción para registrar la asistencia podía ser variable, sin ser muy exacta. El rango podía salirse del aula o localización de la actividad.
- Debía de haber algún tipo de control para no registrar asistencias erróneas o fraudulentas.

Para solucionar estos problemas se ha optado por las siguientes soluciones:
Como servidor se ha utilizado *"Couchbase Server"*, una base de datos distribuida no-SQL, orientada a documentos y de código abierto. Por otro lado, se ha configurado en la aplicación móvil una base de datos *"Couchbase Lite"* que sirve de paso intermedio e interactúa con *"Sync Gateway"* para sincronizar los datos de la base de datos del dispositivo al servidor y viceversa.

```
public class SettingsActivity extends AppCompatActivity {
    //...
    public static class SettingsFragment extends PreferenceFragment{
        @Override
        public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            // Load the preferences from an XML resource
            addPreferencesFromResource(R.xml.preferences);
        }
    }
    //...
}
```

Listado 4.7: Para cargar la pantalla de preferencias de un fichero XML en el fragmento *Settings*.

Para almacenar los datos del usuario, se ha hecho uso de la API de preferencias de Android como podemos apreciar en los listados **??** y **??**, el cual almacena los datos en el dispositivo móvil en un espacio compartido para toda la aplicación. De esta manera, se han almacenado datos como *"Nombre de Usuario"*, *"Número Das"* o *"DNI"*, los cuáles solo serán visibles para el usuario de la aplicación y servirán para registrar las asistencias con estos datos.

En cuanto al área de acción a partir de la cual es posible registrar la asistencia, se ha optado por establecer un perímetro dentro del aula delimitado por las paredes de la misma, descartando cierto margen y de manera que el aula quede cubierta y los exteriores no se tengan en cuenta.

Despliegue

En este caso, para poder realizar este proceso es necesario utilizar la trilateración [**?**] por lo que, es necesario desplegar al menos 3 beacons. Para las pruebas se ha utilizado el aula 2.1 del Edificio de Ingeniería Informática.

Estos beacons no tienen que ser necesariamente por aula ya cada beacon cubre un área de 70 metros de radio aproximadamente; teniendo en cuenta que las paredes crean interferencias en la señal, probablemente se podrían utilizar 3 beacons para cubrir dos aulas, pero habría que hacer un estudio de la localización y las interferencias para poder asegurarlo.

Funcionamiento

En cuanto se accede al módulo de asistencia, la aplicación comienza a escanear las inmediaciones buscando beacons. Cuando detecta más de 2 beacons, la aplicación carga la imagen de dicha localización y comprueba que el alumno se encuentra dentro de la zona delimitada para registrar la asistencia, listado **??**.

```
<PreferenceScreen
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:widgetLayout="@layout/custom_preference_layout"
    >
    <PreferenceCategory
        android:title="@string/user_data">
        <EditTextPreference
            android:id="@+id/userPref"
            android:key="userName"
            android:title="@string/user_name"
            android:summary="@string/summary_user_name"
            android:dialogTitle="@string/dialog_user_name"
            android:defaultValue="@string/user_name"/>
        <EditTextPreference
            android:key="userAlu"
            android:title="@string/user_alu"
            android:summary="@string/summary_user_alu"
            android:dialogTitle="@string/dialog_user_alu"
            android:defaultValue="@string/user_alu"/>
        <EditTextPreference
            android:key="userDni"
            android:title="@string/user_dni"
            android:summary="@string/summary_user_dni"
            android:dialogTitle="@string/dialog_user_dni"
            android:defaultValue="@string/user_dni"/>
    </PreferenceCategory>
</PreferenceScreen>
```

Listado 4.8: El fichero XML de donde se cargan todas las preferencias.

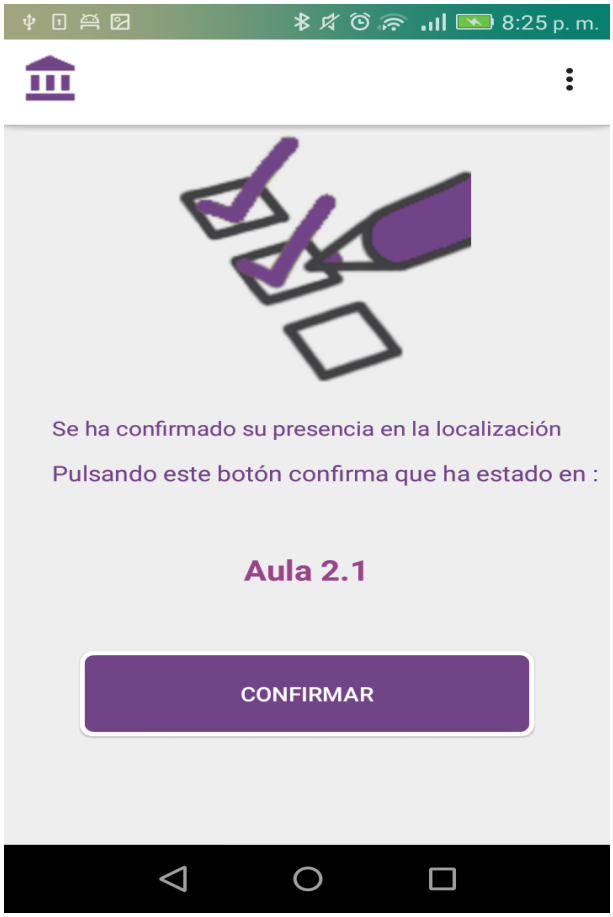


Figura 4.6: Pantalla de confirmación de asistencia en la localización.

Si no se encuentra en la zona delimitada, igualmente la aplicación muestra su ubicación en el mapa para que pueda corregir su posición. Una vez que acceda al área, se muestra un mensaje de confirmación con el aula en la que se encuentra. En este momento se debe confirmar (Véase Figura **??**) pulsando un botón de que el usuario desea registrar la asistencia. Previamente el usuario ha debido introducir sus datos en la aplicación y éstos deben estar almacenados. Estos datos junto con la dirección MAC del dispositivo desde el cual se envía la petición de registro son los que utiliza la aplicación. Otro control es almacenar la hora junto con los demás datos cuando se confirme la asistencia, asegurándonos así de que realmente el usuario ha estado en el aula a dicha hora.

Los datos que se almacenan en el servidor tienen la estructura que se muestra en la Figura **??**, sin embargo esta estructura puede ser fácilmente modificada si se detecta que es necesario añadir más datos.

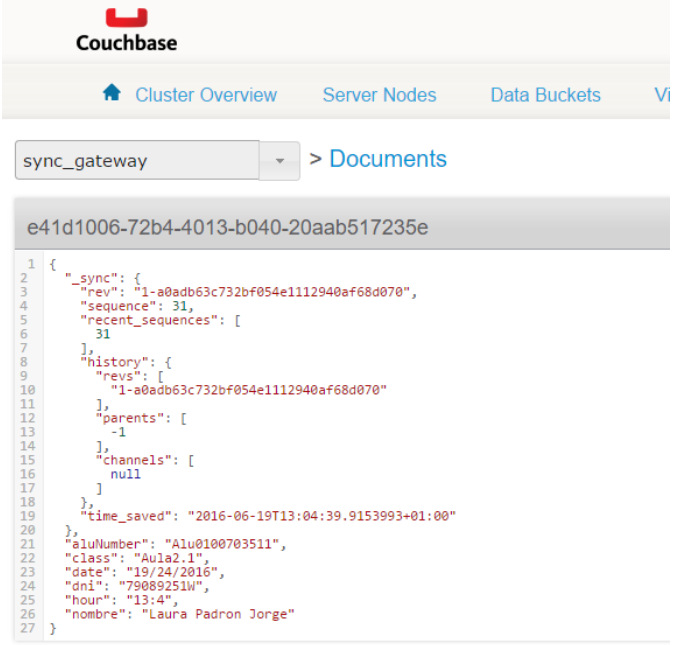


Figura 4.7: La estructura de los datos del usuario en CouchBase Server.

Dificultades

En este caso las dificultades han sido de diversa índole: se ha tenido que configurar un servidor con el que no se había trabajado previamente, se ha tenido que trabajar con la API de preferencias de Android, y había que idear un método que permitiese comprobar de manera fiable que el alumno se encontraba o había estado en la localización a la hora correcta. La configuración del servidor y Sync Gateway ha sido un poco tediosa, pero una vez configurada ha sido muy fácil de

```
public class ScanFragment extends Fragment implements BeaconConsumer {

    private BeaconManager beaconManager;
    protected int functionality;
    private static final Region myregion = new Region("All", null,null,null);
    //...
    @Override
    public void onBeaconServiceConnect() {
        //...
        beaconManager.setRangeNotifier(new RangeNotifier() {

            @Override
            public void didRangeBeaconsInRegion(Collection<Beacon> beacons, Region region) {
                String message = "Comprobando regiones";
                if (region.getUniqueId().equals("All")) {
                    if (beacons != null && beacons.iterator().hasNext()) {
                        //...
                        else if(functionality==5){
                            if (beacons.size() >= 3) {
                                int count = 0;
                                String[] ids = new String[beacons.size()];
                                double[] dist = new double[beacons.size()];
                                for (Beacon selBeacon : beacons) {
                                    ids[count] = selBeacon.getBluetoothAddress();
                                    dist[count] = selBeacon.getDistance();
                                    count++;
                                }
                                try {
                                    checkPerimeter(beacons,ids, dist, BeaconAttInfo.getAreas(),
                                        staticVars.SCALEETSII, BeaconAttInfo.getImage());
                                } catch (Exception e) {
                                }
                            }
                        }
                        //...
                    }
                }
            }
        });
    }
    //...
}
```

Listado 4.9: Utilizando *ScanFragment* se comprueba que el usuario se encuentra dentro de la zona delimitada.

utilizar.

Ampliación

Para ampliar este caso de uso se deberían incluir nuevos beacons, nuevas imágenes de las localizaciones y nuevas áreas de registro; sin embargo, el envío de datos sería prácticamente el mismo. Si se decidiese cambiar en un futuro los datos a almacenar en el servidor, sería posible sin mucho esfuerzo.

4.1.4. Guía a través del campus de la universidad

Objetivo

Este caso de uso se centra en ofrecer una pequeña guía al usuario mostrándole su posición en un edificio y algo de información útil. Para ello hemos tomado a modo de ejemplo el edificio de Matemáticas y Física ya que poseíamos los planos.

Se han tomado como referencia algunas zonas del edificio y se han creado algunas indicaciones básicas para acompañar la posición en el mapa y los puntos de interés. Es necesario destacar que las pruebas se han realizado utilizando 3 beacons, con lo cual sólo se ha cubierto una pequeña parte del edificio, sin embargo a modo de ejemplo muestra perfectamente las posibilidades de esta tecnología.

Despliegue

En este caso, también es necesario utilizar el algoritmo de trilateración. En el listado **??** podemos observar los métodos principales para utilizar la trilateración. Hemos tenido que desplegar 3 beacons en el lugar que queríamos visualizar, en el edificio de Física y Matemáticas en la primera planta, se han colocado en las esquinas de la zona principal, conserjería, cerca de los ascensores y en la entrada a los pasillos principales.

Funcionamiento

En cuanto se accede al módulo de guía, la aplicación comienza a escanear las inmediaciones buscando beacons. Cuando detecta más de 2 beacons, la aplicación carga la imagen de dicha localización, en este caso el plano del edificio de Física y Matemáticas, una versión reducida para la parte en la que nos encontremos y comprueba que el usuario se encuentra dentro de algunas de las posibles zonas. En cuanto se entre en alguna se mostrará la información relacionada con esta posición. En el estado actual de `BulletPoint` las acciones de la zona se limitan a mostrar un pequeño texto con información generada teniendo en cuenta que el usuario necesite guiarse en las instalaciones, es decir, no conoce las instalaciones.

```
1   public Point calculatePosition(double[][] positions, double[] distances ) {
2
3       TrilaterationFunction trilaterationFunction = new TrilaterationFunction(positions,
            distances);
4       NonLinearLeastSquaresSolver nlSolver = new NonLinearLeastSquaresSolver(
            trilaterationFunction, new org.apache.commons.math3.fitting.leastsquares.
            LevenbergMarquardtOptimizer());
5
6       LeastSquaresOptimizer.Optimum optimum = nlSolver.solve();
7       // The center of the point where the user is.
8       double[] centroid = optimum.getPoint().toArray();
9       return new Point(centroid[0],centroid[1]);
10  }
11
12  public void drawPointInPosition(final double posX, final double posY, final double radius,
        final List<Area> areas, final int imageResource) {
13      getActivity().runOnUiThread(new Runnable() {
14          @Override
15          public void run() {
16              BitmapFactory myFactory= new BitmapFactory();
17              BitmapFactory.Options opt = new BitmapFactory.Options();
18              opt.inScaled = false;
19              opt.inMutable = true;
20
21              Bitmap bitmap = myFactory.decodeResource(getResources(), imageResource,opt);
22              //...
23              Canvas canvas = new Canvas(bitmap);
24              for (Area area: areas) {
25                  canvas.drawRect(new Rect(area.getLeft(),area.getTop(),area.getRight(),area.
                        getBottom()), paintZone);
26              }
27              canvas.drawCircle((float) posX, (float) posY, (float) radius, paint);
28
29              //...
30              ImageView imageView = (ImageView) getActivity().findViewById(R.id.location);
31              imageView.setAdjustViewBounds(true);
32              imageView.setImageBitmap(bitmap);
33          }
34      });
35  }
36  }
```

Listado 4.10: Los métodos principales para utilizar el algoritmo de trilateración con los beacons, calcular la posición del usuario y dibujar en la imagen.

Ya que este caso de uso podría ser útil para un rango más amplio de usuarios por su naturaleza de guía, se ha decidido ofrecer soporte en inglés a la hora de mostrar las indicaciones.

Dificultades

Las dificultades de este caso de uso radican principalmente en situar al usuario en el edificio. Dentro de un edificio existen muchos factores que pueden distorsionar la calidad de la señal que emite un beacon (paredes, personas, etc.), y por tanto influir en su posición estimada en el mapa. A la hora de enfrentarnos a este problema hemos comprobado que la localización presenta un rango de error y que es necesario darle un margen de tiempo a la aplicación para calcular la posición del usuario. Para intentar paliar esta situación, las áreas en las que se intercambia información con el usuario aparecen resaltadas en el mapa, de esta manera el usuario es capaz de saber con certeza que tiene un área de información e intentar corregir su posición si está interesado.

Ampliación

Para ampliar este caso de uso se deberían incluir nuevos beacons, nuevas imágenes de las localizaciones (exteriores o interiores) y nuevas áreas. Las áreas podrían ofrecer diferentes funcionalidades aparte de la información de la localización.

4.1.5. Acceso al parking

Objetivo

El objetivo principal de este caso de uso es permitir a usuario acceder a los aparcamientos universitarios utilizando la aplicación móvil. Actualmente el acceso al parking se formaliza utilizando tarjetas de acceso magnetizadas. El objetivo principal es poder complementar este método con uno más sencillo y más sostenible a largo plazo.

Despliegue

En este caso, también es necesario utilizar el algoritmo de trilateración, por lo que se ha tenido que desplegar 3 beacons en la entrada del parking, se ha utilizado el parking del edificio central, disponiendo los beacons en posiciones previamente seleccionadas como se muestra en la imagen de la Figura **??**.

Figura 4.8: Se puede apreciar la estructura del parking central. Los puntos A, B y C señalan las posiciones de los beacons.

Funcionamiento

En cuanto se accede al módulo de parking de `BulletPoint`, la aplicación muestra una lista con los diferentes aparcamientos y el número de plazas de cada recinto. Esta lista se obtiene mediante una consulta a una API diseñada en colaboración con Alberto Morales, quien se ha encargado de configurar el servicio para facilitar a la aplicación la información necesaria. De la lista de aparcamientos, el usuario ha de seleccionar al que quiere acceder, identificando así la imagen y zonas a dibujar. Esta selección da paso a un escáner que comienza a detectar la posición del usuario y a situarlo en la imagen. Si el usuario se encuentra en una de las zonas designadas para entrar o salir del parking, se lanzará la acción, que en este caso consiste en una petición de apertura contra el servidor. En este caso cada zona posee un identificador de barrera que es el que usa la petición para determinar que barrera debe abrir.

Por otro lado esta petición debe ir autentificada, para poder acceder, por lo que la aplicación ha negociado previamente un secreto con el servidor que la identifica y le permite realizar la acción de apertura utilizando un número identificativo basado en la MAC del dispositivo de pruebas.

Dificultades

En este caso de uso las dificultades han radicado en el hecho de comunicarse con el servidor para realizar las llamadas. La trilateración se ha realizado de la misma manera por lo que no ha habido grandes problemas. Además de estas dificultades, cabe añadir una adicional y es que la seguridad del parking depende del sistema single log in de la ULL, por lo que ha habido que buscar la manera de abordar este tema. Sin embargo, Alberto Morales se ha implicado en todo momento intentado resolver las necesidades de la aplicación de manera óptima, lo que ha sido una gran ayuda.

Ampliación

Para ampliar este caso de uso se deberían incluir nuevos beacons, 3 mínimo por cada recinto, nuevas imágenes de las localizaciones (exteriores o interiores) y nuevas áreas; Las áreas serían esenciales para definir las acciones de apertura de las barreas. Si las zonas acaban siendo muchas, sería recomendable crear una base de datos para almacenar las localizaciones, las zonas y las imágenes a cargar.

4.2. Despliegue

El despliegue de estos dispositivos es variable como hemos podido comprobar con los distintos casos de uso. Sin embargo hay ciertas consideraciones comunes a en todos los casos, si lo que deseamos es un sistema que detecte entradas y salidas, independientemente de la posición exacta del usuario, simplemente necesitaremos un único beacon como regla general.

Sin embargo, si se pretende conocer la localización más precisa del usuario necesitaremos recurrir a un método como la trilateración [**?**]. Serán necesarios un mínimo de 3 beacons. Incrementando el número de beacons es posible aumentar la precisión con la que se percibe la posición del usuario. El algoritmo no es totalmente exacto, puesto que depende de las distancias calculadas a los beacons, sin embargo estas distancias pueden variar. Existen diversos factores que pueden influir en la calidad de la señal del Bluetooth y por tanto distorsionar estas distancias, entre otros factores podemos destacar principalmente:

- Físicos: paredes, personas o elementos del entorno principalmente.
- Intangibles: interferencias con otras ondas.
- Configurables: la intensidad de la señal y por tanto su rango es configurable.

Todos estos factores han de ser tomados en cuenta a la hora de realizar el despliegue de los beacons. Otro factor a tener en cuenta es la altura. Los dispositivos es recomendables levantarlos cierta altura sobre el nivel del suelo. A la hora de probarlos se ha intentado, en la medida de lo posible, mantenerlos a un nivel elevado sobre la altura de las personas y siempre intentando mantener esta medida de altura para los diferentes beacons. En cuanto al despliegue de la aplicación, el código fuente de `BulletPoint` se encuentra disponible para su descarga en [**?**], bajo licencia Creative Commons Reconocimiento-NoComercial-CompartirIgual 4.0 Internacional [**?**].

Capítulo 5

Conclusiones y líneas de trabajo futuras

En este capítulo se presentarán las conclusiones a las que se ha llegado tras realizar este TFG y discutiremos posibles líneas de trabajo futuras.

5.0.1. Conclusiones

Actualmente la tecnología beacon se encuentra aún en fase de desarrollo. Por sí sola puede llegar a tener muchas limitaciones debido a su funcionamiento. Las restricciones físicas y de posicionamiento requieren que se realice un análisis para posicionar los dispositivos de la mejor manera. Por otro lado cada distribuidor específico de beacons intenta que se utilicen sus SDKs para desarrollar las aplicaciones. Dependiendo del caso, incluso existen algunos sujetos a cuotas, con lo que desarrollar utilizando esta tecnología puede no resultar económico y poner las cosas difíciles a muchos desarrolladores. La solución a este problema la hemos encontrado en la librería AltBeacon que intenta abrir las puertas a los desarrolladores que pretenden utilizar esta tecnología.

Gracias a esta librería se están desarrollando diferentes aplicaciones, que podrían llegar a tener funcionalidades interesantes para el día a día como la automatización de algunas tareas. Hasta ahora se han explorado las posibilidades más beneficiosas a la hora de vender la tecnología como es la publicidad, pero al desarrollar este trabajo, hemos comprobado que el mercado es mucho más amplio, abarca mucho más de lo que se ha desarrollado hasta ahora.

La consecución de este TFG muestra que existen muchas aplicaciones que podrían interactuar con esta tecnología y dar lugar a un producto útil y cómodo para los usuarios. En unos años probablemente seamos capaces de explotar al máximo las posibilidades que nos ofrecen los beacons, dando paso a aplicaciones que formen parte de nuestra rutina, sin embargo por ahora seguiremos

experimentando con esta tecnología y viendo como evoluciona.

5.0.2. Conclusions

Nowadays, beacon technology is still on a development phase. By itself can be quite limitated due to to its functioning.The physical and positional limitations require a deep analysis in order to place devices the best way. On the other hand, each specific beacon provider tries to use his own SDKs to develop the aplications. Depending on the case, fees might be applied so, working with this technology can end up being not money-saving and make things difficult for many developers.

The solution to this problem has been found on the AltBeacon library, which tries to bring close to developers this technology. Thanks to this library, many applications, which might have handy functionalities for day-to-day usage, have been developed and many related to the automation of tasks. Until now just the advertising possibilites have been explored, since it is the most profitable option.

However, during this project we have realized that market is far more wide and it covers much more than what has been developed right now. The achievement of this Final Year Project is to show that, many apps able to work with this technology exist and they can give as a result an useful and convenient product for users.

In a few years, we will probably be able to take advantage of all posibilities beacons offer and we will be able to develop apps, which would end up beeing part of our routine; for now, we will keep on testing this technology and watching it grow.

5.0.3. Líneas de trabajo futuras

En un futuro se podrían ampliar las funcionalidades desarrollando los demás casos de uso planteados que no se han elegido en este trabajo. En entornos universitarios existen multitud de servicios que podemos relacionar con esta tecnología, entre otros podemos destacar:

- Acceso a instalaciones deportivas, educativas o de diversa índole.
- Contratación de servicios en diferentes puntos: alquiler de bicicletas en puntos de alquiler, pago de entradas en las inmediaciones de un evento.
- Anuncios o promociones de diverso tipo, relacionados tanto con la universidad como con establecimientos comerciales cercanos al campus.
- Servicios de guía dentro del campus, con puntos de interés para el alumnado.

Son múltiples las posibilidades que se le pueden dar a esta tecnología. En otros sectores como el turismo o la medicina también están empezando a tener relevancia, con lo que podemos confirmar que no es una tecnología aislada, sino que poco a poco se está abriendo paso en el mercado.

Capítulo 6

Presupuesto y puesta en marcha

En este capítulo se expondrán las estimaciones de recursos necesarios para poner en práctica este despliegue teniendo en cuenta el estado actual del proyecto.

El desglose del presupuesto de la aplicación se puede separar en dos partes: por un lado, la compra de los distintos dispositivos, y por otro el desarrollo y mantenimiento de la aplicación. Se puede contabilizar de la siguiente manera:

- Para el caso de uso de los autobuses, sería necesario un beacon por parada. Si sólo se cubriesen las paradas dentro de la zona universitaria, obtendríamos unos 20 beacons a distribuir entre Anchieta y Guajara.
- Siguiendo con el caso de uso de los eventos, se podrían calcular 10 zonas de eventos principales nuevamente a distribuir entre Guajara y Anchieta. Estas zonas abarcarían desde el paraninfo, aulas magnas o bibliotecas hasta lugares destinados específicamente para la realización de eventos.
- Para los casos de uso que dependen de la localización, debemos realizar un análisis de cada recinto: los aparcamientos para el módulo del aparcamiento, las aulas para el módulo de asistencia y los edificios o exteriores para el módulo de guía.

Teniendo en cuenta que cada dispositivo de Aruba lo comercializamos por 25 euros:

- Para cubrir el primer caso de uso sería necesaria una inversión de 500 euros y cubriríamos 20 paradas de autobús.
- Para las 10 zonas de eventos se necesitarían 250 euros (aunque es posible añadir más beacons)

- Los casos de uso que dependen de la localización son relativos: como hay 8 aparcamientos y se necesitan como mínimo 3 beacons por recinto, podríamos hablar de un mínimo de 600 euros a desembolsar. En el caso de la asistencia y guía, si calculamos que para cubrir todos los edificios de la ULL (facultades en su mayoría) se necesitarían unos 100 beacons, el importe asciende a 2500 euros.

Esta primera parte del presupuesto suma en total 3.850 euros y con ellos quedarían cubiertos los edificios, aparcamientos y paradas de autobús cercanas al campus. La instalación y configuración de los dispositivos se ha calculado en base a la suma de 12,40 euros por hora trabajada. Se necesitarían 2 especialistas trabajando durante un mes a jornada completo (8 horas) para realizar la instalación.

El total del despliegue ascendería a 3.968 euros. Si lo sumamos al precio de los dispositivos obtenemos la suma de 7.818 euros. Los dispositivos cuentan con una garantía de un año por lo que, si el dispositivo fuese considerado defectuoso en ese plazo, se procedería al cambio del mismo sin coste para el cliente.

Aparte de los dispositivos hay que añadir la segunda parte del coste, derivada del mantenimiento de la aplicación y del coste de un servidor para almacenar los datos y realizar las consultas. Para esta parte se ofrecen dos opciones:

- Compra del servidor dedicado junto con el soporte y configuración servidor (cuota anual) : 1300 + 250 = 1.550 euros.

- Configuración de un servidor ya disponible en la Universidad (sin cuota anual de soporte ni mantenimiento): 550 euros.

Por otro lado, sería recomendable el mantenimiento de la aplicación. Ofrecemos un especialista programador en Android que se haría cargo de las incidencias que pudieran surgir con la app en horario laboral a jornada completa, con un tiempo de respuesta de 24 horas y solución de la incidencia en 72 horas a partir de la confirmación de la misma.

El coste final ascendería a 21.600 euros anuales netos durante el primer año y a 18.000 euros en adelante. Este mantenimiento no incluye nuevas funcionalidades, únicamente el mantenimiento y el correcto funcionamiento de los módulos disponibles a día de hoy.

El presupuesto final habría que calcularlo en función de las distintas opciones presentadas.