Facultad
de
Ciencias

GEOLOCALIZACIÓN EN EL ESPACIO PARA LA GESTIÓN DE INVENTARIO

(Space geolocation for inventory management)

Trabajo de Fin de Grado
para acceder al

GRADO EN INGENIERÍA INFORMÁTICA

Autor: Manuel Fernández Herrera

Director: Patricia López Martínez

Codirector: Sergio Herrera Iglesias

Febrero - 2019

RESUMEN

La utilización de un sistema de localización en exteriores, como puede ser el GPS, proporciona datos útiles para numerosas funcionalidades como podemos ver en aplicaciones existentes en el mercado. Este sistema funciona mediante la utilización de satélites que pierden precisión si el objeto a detectar se encuentra dentro de una estructura como puede ser un edificio. Esta situación hace que los sistemas encargados de la localización en interiores estén ganando una gran popularidad en los últimos años. Sin embargo, muchos de estos sistemas tienen un coste elevado y son complejos.

En este proyecto se presenta un estudio de las diferentes tecnologías existentes en el mercado para implementar esta funcionalidad, junto al posterior diseño e implementación de una aplicación que utilizando alguna de las tecnologías analizadas, obtenga la localización de diferentes activos en un espacio controlado. Es decir, el objetivo de la aplicación es obtener estas posiciones sirviendo como evaluación de la tecnología elegida. Los datos obtenidos por la aplicación podrán ser utilizados para la realización de diferentes acciones con el fin de automatizar tareas sobre el establecimiento en el que se implante el sistema, mejorando así la calidad del edificio.

Palabras clave: Beacons, construcción inteligente, geolocalización, localización en interiores.

SUMMARY

The use of an outdoor location system, like GPS, provides useful data for many functionalities as we can see in existing application in the market. This system works by using satellites which lose accuracy if the object to detect is within a structure such a building. This situation makes systems in charge of indoor location to increase their popularity in recent years. However, many of these systems have a high cost and are complex to use.

This project presents a study of the different technologies available in the market for indoor location, along with the subsequent design, assembly and evaluation of chosen technique, whose final objective is obtaining the location, of different assets in a controlled space is presented. To do this, an application to obtain these positions, whose added functionality will be the evaluation of chosen technology will be made. This data will be used to carry out different subsequent actions to automate different establishment actions where system is implanted, improving their quality. This implementation is known as smart building.

Keywords: Beacons, Smart building, geolocation, indoor location.

ÍNDICE

GLOSARIO

ACID	Atomicity, Consistency, Isolation y Durability. En castellano, Atomicidad, Consistencia, Aislamiento y Durabilidad.
API	Application Programming Interfaces. En castellano, Interfaces de programación de aplicaciones).
Beacons	Dispositivos de bajo consumo que emiten señales broadcast.
Broadcast	Transmisión que será recibida por todos los dispositivos de la red.
CMDB	Base de datos de la gestión de la configuración.
CRUD	Acrónimo de create, read, update and delete. En castellano crear, leer, actualizar y borrar.
CSV	Documento en formato abierto para representar datos en forma de tabla.
DBm	Unidad de potencia, decibelios (dB) respecto a un milivatio (mW).
GNU	Sistema operativo de tipo Unix.
GPL	Licencia pública general de GNU.
GPS	Sistema de Posicionamiento Global.
IDE	Integrated Development Environment. En castellano entorno de desarrollo integrado.
ITIL	Describe un conjunto de mejores prácticas y recomendaciones para la administración de servicios.
NAGIOS	Sistema de código abierto para la monitorización de redes.
NFC	Near Field Communication (NFC) o comunicación de campo cercano.
REST	Representational State Transfer. En castellano Transferencia de Estado Representacional.
RFID	Sistema de almacenamiento y recuperación de datos remotos.
RSSI	Indicador de fuerza de la señal recibida.
SaaS	Software as a Service. En castellano software como servicio.
SLA	Acuerdos a nivel de servicio.
Tags	Etiquetas para la identificación de dispositivos.
GIS	Geographic information system. En castellano, sistema de información geográfica.

1. Introducción

En este apartado se iniciará la memoria del proyecto, comentando la motivación para su realización junto a los objetivos finales que se quieren alcanzar con su desarrollo.

1.1. Contexto

Las constantes innovaciones tecnológicas están cambiando muchos aspectos en la vida de nuestra civilización. Una de estas mejoras es la construcción de los denominados smart buildings, también conocidos como edificios inteligentes [1]. Estas estructuras cuentan con numerosas funcionalidades que nos ayudarán en el día a día, automatizando diferentes funciones como puede ser la apertura de diferentes puertas o la activación de las luces, utilizando únicamente estas funcionalidades cuando sean necesarias. En este proyecto nos centraremos en aquellas que tengan que ver con la localización de diferentes activos en el interior del edificio. Estos sistemas pueden proporcionar datos sobre los que se podrán aplicar una serie de algoritmos para automatizar algunas de las funcionalidades existentes en el edificio, como puede ser el encendido de la calefacción automática.

La empresa CIC está interesada en esta tecnología, siendo su intención inicial implantar el sistema de manera privada en sus estructuras, para ir ganando experiencia en su desarrollo, y posteriormente poder desarrollar soluciones que puedan ser sacadas al mercado.

1.2. Objetivos del proyecto

El presente proyecto aborda el desarrollo de una aplicación de control de posición de activos en el espacio, entendiendo como activo cualquier elemento, objeto o persona que porte algún tipo de identificador, incluyendo como parte del trabajo la investigación de posibles técnicas para llevarlo a cabo. Esta aplicación permitirá obtener y almacenar datos sobre la posición de diferentes activos, con una posterior gestión de la información obtenida para la realización de diferentes acciones, como puede ser la automatización de la calefacción o el aire en función de las personas que se encuentren en un espacio para reducir el consumo.

Dentro del sistema a desarrollar diferenciaremos los siguientes casos de uso:

- Conocer la posición en el espacio de un activo, siendo el propio activo el solicitante de dicha información.
- Conocer la posición de los activos que se encuentren en el rango de espacio que sea detectado por nuestro software.

Lo primero que debemos realizar para poder encontrar a un activo es proporcionar a este elemento la capacidad de transmitir información, ya sea de forma activa o pasiva:

- **Activa:** la transmisión se realiza de modo que ambos elementos de la conexión intercambian datos.
- **Pasiva:** transmisión en la que solo hay un dispositivo encargado de transmitir datos mientras que el otro dispositivo de la conexión aprovecha para obtener estos datos.

1.3. Organización de la memoria

El resto del proyecto se expone a lo largo de los siguientes capítulos. En el capítulo 2 se detallarán las diferentes herramientas utilizadas durante la realización del proyecto. A continuación, en el capítulo 3, se comentarán las diferentes tecnologías existentes en el mercado para la comunicación en interiores junto a la tecnología que finalmente será utilizada para la realización de nuestro sistema. Más adelante, en el capítulo 4, se explicarán los diferentes activos que tendrá en cuenta el sistema, además de posibles técnicas que podremos aplicar para conseguir su posición. Posteriormente, en el capítulo 5, se detallará el primer caso de uso, junto a los diferentes pasos realizados para su implementación. En el capítulo 6 se comentará el segundo caso de uso. Seguidamente, en el capítulo 7, se encontrará la conclusión que obtenemos una vez realizadas las acciones necesarias para el desarrollo del proyecto. Para finalizar, en el capítulo 8, se encontrará la bibliografía utilizada para realizar el proyecto.

2.Herramientas utilizadas

Durante la elaboración del proyecto deberemos utilizar diferentes herramientas que nos ayudarán en el desarrollo de este. Entre el software que hemos utilizado destacamos el siguiente:

- **Android Studio** [2]: es un entorno de desarrollo integrado para Android y su función en el proyecto será la de crear aplicaciones para este sistema. La versión actual de este programa nos proporciona características como:
    - Renderizado en tiempo real.
    - Consejos de optimización, ayuda para la traducción y estadísticas de uso.
    - Refactorización específica de Android y arreglos rápidos.
    - Un editor de diseño enriquecido que permite realizar la interfaz de usuario de forma visual.
    - Plantillas para crear diseños comunes de Android y otros componentes.
    - Un dispositivo virtual de Android que se utiliza para ejecutar y probar aplicaciones.

- **Eclipse** [3]: es uno de los mejores IDE del mercado. Su uso es simple e intuitivo, incluyendo además herramientas de refactorización. Hay que destacar que es un programa gratuito. Se utilizará en el desarrollo de un servicio REST que interactuará con la base de datos, conectando así nuestra aplicación con un componente externo.

- **CMDBuild** [4]: es una aplicación web configurable para modelar y controlar la base de datos que contiene los activos y para manejar las operaciones de flujo de trabajo relacionados. Esta herramienta es open source, implementa las recomendaciones de ITIL y se distribuye bajo licencia GPL. Permite gestionar los activos, relacionar elementos de este y almacenar documentos. Otra de las características que ha sido determinante a la hora de elegirla como herramienta, es que es multidioma.

- **DBeaver** [5]: es un gestor de base de datos universal, es decir, es independiente del servidor de base de datos que tengamos. Permite operar con una gran cantidad de bases de datos, más adelante detallaremos cuál es la elegida para este proyecto. Esta herramienta es multiplataforma y consta de versiones para los principales sistemas operativos. Actúa como cliente SQL y ayuda en la administración de la base de datos. Nuestro caso se basa en una base de datos relacional, y para gestionarla utiliza la API de JDBC para interactuar con las bases de datos a través de un controlador JDBC. La edición comunitaria de DBeaver es gratuita como software de código abierto que se distribuye bajo la licencia Apache.

- **PostgreSQL** [6]: si hablamos de bases de datos relacionales open source, PostgreSQL es una de las opciones más interesantes del mercado, ya que es considerado el motor de base de datos, obtenido de forma gratuita, más avanzado de la actualidad. Una de las características más interesantes de esta herramienta es el control de concurrencias multiversión, esta funcionalidad agrega una imagen del estado de la base de datos a cada transacción, permitiéndonos realizar transacciones consistentes de manera eventual, consiguiendo grandes ventajas de rendimiento. PostgreSQL funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo simultáneamente al sistema.

  Algunas de las características más importantes y soportadas por PostgreSQL son:
  - Es una base de datos ACID.
  - Permite el soporte y creación de numerosos tipos de datos.
  - Realiza copias de seguridad
  - Numerosos métodos de autentificación.
  - Completa documentación.
  - Disponible para Linux, UNIX y Windows.

- **GIS** [7]: conjunto de herramientas cuyo objetivo es el almacenamiento, modificación y análisis de datos relacionados con el mundo real vinculados a una referencia espacial. Este sistema nos proporcionará ventajas como:
  - Almacenar únicamente los datos de los elementos digitalizados por lo que requiere menos memoria para su almacenamiento y tratamiento.
  - Buena salida gráfica.
  - Compatible con entornos de bases de datos relacionales.
  - Datos fáciles de mantener y actualizar.

- **Postman** [8]: esta aplicación permite realizar peticiones HTTP (GET, POST, PUT, DELETE entre otros) a una dirección url. Esta funcionalidad es de gran utilidad a la hora de interactuar con APIs Web o para probar nuestros propios desarrollos. También nos permite modificar tanto los parámetros de la petición como el contenido añadiendo JSON, XML o texto plano entre otros.

- **GitKraken:** interfaz gráfica multiplataforma para Git que nos facilitará el uso de este sistema. Git es un software encargado del control de versiones, en el que iremos detallando cada paso de nuestro proyecto tanto de forma local como de forma remota. Esta interfaz nos proporciona una serie de beneficios respecto a usar Git mediante comandos:
  - No es necesario memorizar comandos, la interfaz gráfica facilita la interacción con los datos.
  - Sistema multiplataforma.

    - Posibilidad de establecer la estructura de diferentes modelos, como la aplicada en nuestro proyecto, “Git flow”.
    - Posibilidad de desarrollo concurrente por parte de diferentes desarrolladores.

- **Wireshark** [9]: programa encargado de analizar distintos protocolos de comunicación que se llevan a cabo en nuestra aplicación. Este sistema es útil a la hora de encontrar posibles errores en las llamadas entre diferentes sistemas que interaccionan en nuestra aplicación.

- **Spring + Hibernate** [10]: Spring es un framework de lenguaje de programación Java que facilitará la creación de un servicio web. Spring se usa en conjunto con Hibernate, herramienta que facilita el mapeo de un modelo de objetos a una base de datos relacional.

3.Elección de tecnologías de soporte

En este capítulo se analizarán algunas de las tecnologías existentes en el mercado para la realización de este proyecto, junto a la elección de las que finalmente serán empleadas.

3.1. Análisis de tecnologías candidatas

En el mercado existen diferentes tecnologías para otorgar una identificación, siendo probablemente la más común el código de barras, la cual se ha introducido en numerosas cadenas de distribución y sistemas de control de acceso.

Para nuestro caso se consideró la utilización de las siguientes tecnologías:

- **Beacons** [11][12][13]: como se observa en la figura 3.1 es un pequeño dispositivo que utiliza la tecnología Bluetooth para transmitir mensajes o avisos directamente a cualquier dispositivo compatible que se encuentre en un rango de acción que varía dependiendo del dispositivo hasta un máximo de 50-70 metros. Estas señales son únicas, lo que proporciona una identificación para cada dispositivo, y además definen su ubicación, lo cual nos ayudará a conocer, a partir de la posición del beacon, la de los diferentes activos que estén en su rango.

Figura 3.1: *Beacons*

Este sistema consta de numerosas ventajas:

- **Instalación:** la instalación de este sistema es simple.
- **Duración:** amplia batería, con funcionalidad hasta dos años.
- **Tecnología:** no intrusiva, la parte activa es el usuario y deberá realizar una serie de configuraciones para recibir información de estos dispositivos.
- **Conexión:** usa Bluetooth, tecnología ampliamente extendida en el mercado.
- **Localización:** aporta la posibilidad de calcular la posición aproximada de un activo.

- **Uso:** su uso será a distancia con un amplio rango, esto nos otorgará la capacidad de contactar con él a una distancia considerable.
- **Batería:** a diferencia de otras tecnologías no se drenará batería del dispositivo que actúe como pasivo, al transmitir la señal por frecuencias muy bajas.

- **NFC:** Se trata de una tecnología inalámbrica que funciona en la banda de los 13.56 MHz. [14] Su transferencia puede alcanzar los 424 kbit/s por lo que su enfoque está centrado en una comunicación instantánea. Sin embargo, su gran déficit es el rango de acción, siendo este un máximo de unos 10-20 centímetros.

Figura 3.2: *NFC*

Esta tecnología aportaría una serie de características a nuestro sistema, entre las que destacamos:

- **Conexión:** la conexión entre los dos dispositivos es extremadamente rápida, ya que tan solo lleva 0,1 segundos y se produce automáticamente.
- **Comodidad:** la comodidad y la rapidez de poder interaccionar con NFC únicamente acercando el dispositivo como se observa en la figura 3.2, es su mayor ventaja.
- **Distancia de uso:** el gran hándicap de esta tecnología es la distancia máxima entre los dispositivos, que no debe exceder de unos 20 cm.
- **Disponibilidad:** como es evidente, únicamente los dispositivos dotados con esta tecnología podrán realizar estos usos. Se comenta esta característica ya que el porcentaje de dispositivos que la poseen no es próximo a la totalidad del mercado.

- **RFID** [15]: un sistema para la comunicación sin cables entre dos o más objetos, donde uno emite señales de radio y el otro responde en función de la señal recibida. La información transmitida suele ser mínima, limitándose en la mayoría de las ocasiones a un simple identificador. El rango de estos sistemas varía dependiendo de su tipo, siendo el alcance comercial un máximo de seis metros, aunque se han creado casos específicos en el que llegan a alcanzar unos cientos de metros. Un ejemplo de estos sistemas se encontraría en las etiquetas de productos, como puede ser la ropa en una tienda, observable en la figura 3.3.

Figura 3.3: *RFID*

Los sistemas RFID están basados en la comunicación bidireccional entre un lector y una etiqueta, como se detalla en la figura 3.6. La etiqueta es un elemento compuesto por silicio, que está unido al objeto o producto que se va a identificar. Únicamente contiene una antena y un chip, como se observa en la figura 3.4, con la capacidad de almacenar información. [16]

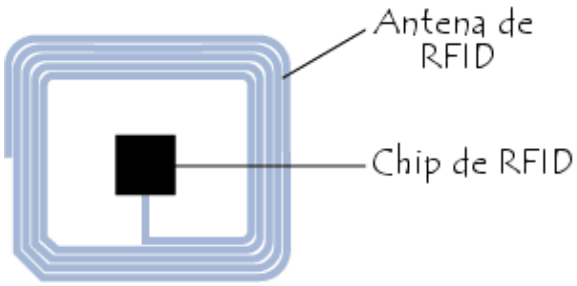


Figura 3.4: *Etiqueta RFID*

El lector, mostrado en la figura 3.5, es el mecanismo que permite leer la etiqueta pegada en el objeto a identificar, logrando así un intercambio de la información contenida en la etiqueta, de acuerdo con la programación de esta.

Figura 3.5: *Lector RFID*

La comunicación entre estos elementos es constante siempre que se encuentren en un rango de distancia óptimo para ella. Cuando la etiqueta se

encuentra en este rango, el lector le envía señales, esta las recibe a partir de su antena y envía su información al lector.

El proceso para esta transmisión de datos es simple:

1. El lector envía señales de radiofrecuencia en forma de ondas a la etiqueta, la cual gracias a su antena capta esta señal.
2. El chip de la etiqueta transmite por radiofrecuencia la información que tiene contenida en memoria.
3. El lector recibe la información transmitida por la etiqueta y lo envía a una base de datos, en la que previamente se han registrado las características del producto.

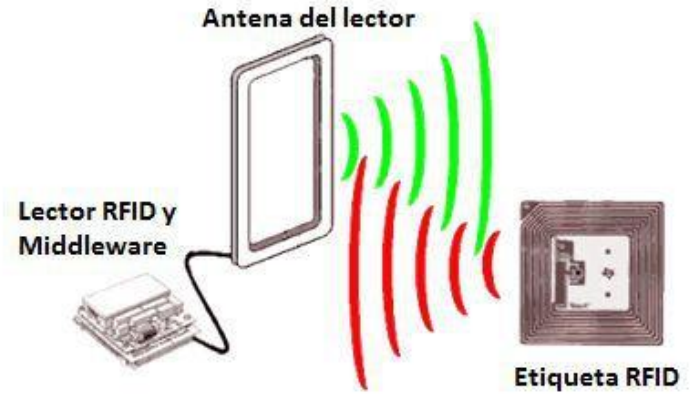


Figura 3.6: *Funcionamiento tecnología RFID*

La frecuencia de estos dos elementos debe ser la misma, con el fin de establecer una comunicación continua entre ambos. Existen diferentes rangos de frecuencias, pero las más comunes oscilan desde 125KHz hasta 2.4GHz.

En esta tecnología podemos encontrar las siguientes características:

- **Seguridad:** los identificadores tienen una compleja duplicación, aumentando así la certeza de encontrar lo que realmente se está buscando.
- **Uso:** su uso será a distancia con un rango amplio, esto nos otorgará la capacidad de realizar la operación en cualquier posición dentro del espacio accesible.
- **Lecturas simultáneas:** esta tecnología permite que numerosos dispositivos sean leídos de manera concurrente.
- **Tags:** no tienen contacto directo con el lector y tienen una vida útil prolongada.
- **Uso:** los activos funcionarán de forma pasiva en el sistema, por lo que no será necesario ningún tipo de actividad por su parte.

- **GPS** [17]: es un sistema de navegación compuesto por una red de satélites colocados en órbita. Estos sistemas transmiten su tiempo y posición continuamente con una alta precisión. Las diferentes localizaciones se calcularán aplicando el principio matemático de trilateración, teniendo como variables las posiciones de los satélites que participan en el cálculo, junto a las diferentes distancias del punto que pide la posición con respecto a los satélites. Estas distancias serán calculadas conociendo el tiempo que tarda en llegar la señal de un dispositivo a otro, sabiendo de antemano la velocidad a la que viaja la señal. Una vez conocidos estos valores, se necesitará un mínimo de cuatro satélites funcionales para conseguir un sistema funcional. El método de trilateración será explicado de forma más detallada en el capítulo 4.2. El GPS tiene dificultades en interiores, al disminuir la calidad en su intento de atravesar las estructuras de edificios, aunque existen medidas para aumentar la precisión en este contexto.

Una vez detalladas algunas de las conexiones que podrían sernos más útiles para nuestra situación procederemos a elegir la más adecuada para cada caso.

3.2. Tecnologías elegidas

En este apartado deberemos diferenciar el caso de uso en el que nos encontremos, ya que no utilizaremos la misma tecnología para ambas situaciones.

En el primer caso de uso, la opción elegida ha sido la utilización de beacons. Sus ventajas ya han sido comentadas en el apartado anterior, pero básicamente ha sido elegido por su bajo coste, fácil implantación y duración en el tiempo.

Dentro de la tecnología beacon existen dos modalidades [18]:

- Eddystone.
- iBeacon.

La principal diferencia entre estos dos tipos es la empresa que la desarrolla. Eddystone es desarrollada por Google, mientras que si hablamos de iBeacon su empresa es Apple. Otra de las diferencias entre estas modalidades es la capacidad de Eddystone para interactuar a través del envío de URLs, sin la necesidad de una aplicación móvil, funcionalidad que iBeacon no implementa. Durante la realización de este proyecto los beacons han sido implementados por Eddystone, al tener más familiaridad con los sistemas de Google que con sistemas pertenecientes a Apple, facilitando su configuración y desarrollo.

En el segundo caso, por el contrario, nuestra preferencia hubiera sido utilizar el sistema RFID, pero al calcular el posible coste total que llevaría su implementación se descartó la opción. Por ello, se decidió reutilizar los beacons obtenidos para el primer caso, pero modificando la forma de implementar el sistema ayudándonos de otros dispositivos, como puede ser el caso de Raspberrys.

Las Raspberrys [19], como se observa en la figura 3.7, son dispositivos de pequeño tamaño formados por una placa base en la que se puede encontrar un procesador, un chip gráfico, una memoria RAM y entradas para conexiones con diferentes dispositivos.

Figura 3.7: *Raspberry*

Básicamente se trata de un ordenador de tamaño reducido con el que detectaremos la distancia a la que se encuentra el dispositivo. Cada uno de estos aparatos mandarán la información obtenida a un servidor central que gestionará esta información. El uso más detallado de este dispositivo se comentará en el capítulo 6.

4. Preparación de entorno

En este capítulo se comentarán los primeros pasos llevados a cabo para la implementación del proyecto, como será la definición del inventario, entendiendo como inventario aquellos elementos cuyos datos serán almacenados y gestionados por el sistema, y el análisis de las posibles técnicas que se podrían utilizar para el cálculo de la posición de los activos.

4.1. Inventario del proyecto

Para la correcta funcionalidad de nuestro proyecto necesitaremos diferentes objetos, que diferenciaremos entre elementos fijos y elementos móviles.

Los elementos fijos son aquellos cuya posición no se verá alterada en el tiempo, excepto en muy contadas ocasiones, mientras que los elementos móviles serán aquellos cuya posición será frecuentemente alterada.

Dentro de los elementos fijos podremos diferenciar:

- Raspberrys.
- Beacons.

Mientras que en los elementos móviles las posibilidades son mayores:

- Personas.
- Ordenadores.
- Monitores.
- Tablets.
- Teléfonos.

A cada uno de estos elementos deben ir asociados diferentes atributos que los caractericen y que dependerán del elemento que estén identificando, entre ellos podemos encontrar:

- Identificador.
- Nombre.
- Fuerza de señal.
- Coordenadas de localización.
- Fecha de localización.
- Descripción.
- Identificador propio del beacon. Los beacons cuentan en su propio sistema con un identificador propio, el cual almacenaremos, pero será independiente al identificador genérico que tendrá cualquier elemento en la base de datos.
- Los beacons también tendrán un atributo “active” que identificará si hay que tener en cuenta este dispositivo para realizar los cálculos. Este valor será útil a la hora de realizar pruebas.

En este momento del desarrollo, se cuenta con un modelo de inventario (modelo de dominio) como el que se muestra en la figura 4.1:

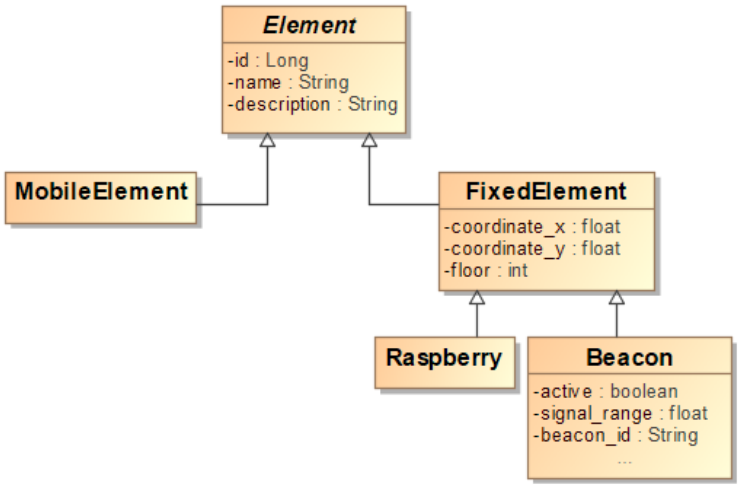


Figura 4.1: *Modelo del inventario*

Este modelo será modificado en el futuro, pero será suficiente para la realización del primer caso de uso.

4.2. Técnicas de cálculo de posición

Para ambos casos de uso es necesario transformar la información obtenida de los correspondientes dispositivos en una posición en el espacio. Para realizar esta transformación podemos diferenciar varias técnicas, entre las que destacamos:

- **Conectividad:** se basa en la proximidad entre los dispositivos. A cada receptor se le establecería un rango limitado de recepción y se comprobaría si el objeto al que queremos asignarle una posición se sitúa dentro de este rango. Si esto sucediera, la posición del objeto sería considerada la misma que la del receptor, que ya es conocida. Esta técnica realiza un posicionamiento simbólico, por lo que si queremos más precisión deberemos optar por otra.
- **Triangulación:** para realizar esta técnica necesitamos que los receptores siempre se encuentren en una posición fija conocida y debemos obtener el ángulo de la señal que queremos rastrear. Por ejemplo, en la figura 4.2 los receptores serían las montañas, cuya posición y ángulo respecto a nuestra posición, como se ha ido comentado anteriormente, es conocida. Por lo tanto, será necesario que los receptores utilicen antenas direccionales, las cuales determinan la dirección con respecto al receptor. Gracias a esto, se podría realizar una recta ficticia en esa dirección, que unidas a las diferentes rectas que realizarán el resto de los receptores que estén en ese rango, permitirán obtener un punto de inserción que posicionará el objetivo en el mapa.

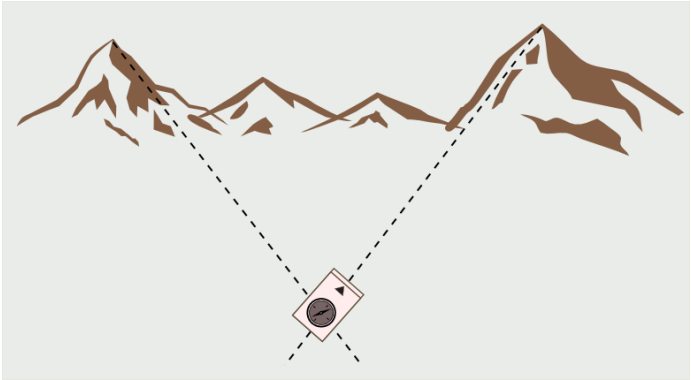


Figura 4.2: *Triangulación*

El mínimo número de receptores para conseguir una posición precisa es dos, siempre que conozcamos la distancia entre ellos, con lo que se conocen dos ángulos del triángulo formado y la distancia de uno de los lados. Con estos datos, mediante una serie de cálculos se podría calcular la posición deseada.

- **Trilateración** [20]: posible técnica válida si nuestras antenas no son direccionables. En este caso se necesitarán al menos tres receptores que representarán tres circunferencias con un radio variable, como se representa en la figura 4.3. Si existe un punto en el que se unan al menos tres radios de diferentes receptores es la posición exacta de nuestro activo. Si el número de radios que pasan por ese punto es menor de tres no se podría conocer la posición exacta del elemento, pero sí identificar un rango de posiciones entre las que se encuentra.

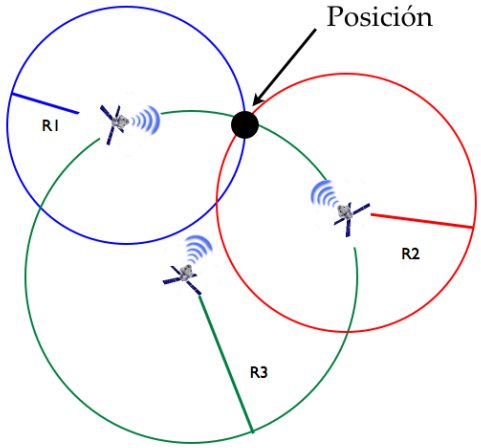


Figura 4.3: *Trilateración*

Este caso se puede extrapolar también a tres dimensiones, ya que el receptor muestra, si la alcanza, la distancia a la que se encuentra el activo, pero no su dirección. Un ejemplo de esta tecnología en nuestras vidas es el cálculo de nuestra posición por GPS, los satélites cubren una "esfera" de un radio máximo, y nuestra posición en el plano se puede calcular si se encuentra dentro de, al menos, cuatro de estas esferas en planos distintos, pero si hablamos en dos

dimensiones con tres circunferencias sería suficiente. Estas situaciones se pueden observar en la figura 4.4.

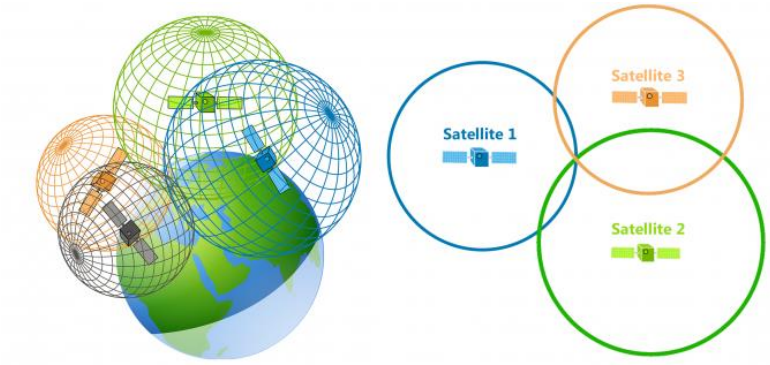


Figura 4.4: *Trilateración 3D/2D*

- **Fingerprint** [21]: el fingerprinting emplea medidas de las señales recibidas desde diferentes emisores para proveer una "huella dactilar" o "fingerprint" de las condiciones de radio en una posición geográfica específica. Normalmente, la posición real estará determinada por algún tipo de medidas de referencia. De esta forma, se crea un mapa de las condiciones de radio y así la posición de un terminal puede ser encontrada después de comparar las medidas reales con las condiciones de referencia.

  En la aplicación del método fingerprinting se pueden diferenciar dos etapas:
  - **Fase de calibrado (o fase offline):** genera las medidas de referencia que se han de almacenar en una base de datos.
  - **Fase de medida en tiempo real (o fase online):** se realizan las localizaciones de los terminales móviles mediante la comparación y análisis de las medidas obtenidas frente a las almacenadas.

  Esta técnica permite aumentar la exactitud de la medida obtenida por las técnicas anteriormente comentadas, teniendo en cuenta los negativos efectos del multitrayecto. Además, no necesita sincronización alguna y, en principio, no es necesario el recalibrado. Se asume como punto de partida que las estaciones emisoras son estáticas, y que se conoce con exactitud su emplazamiento real.

  Hay que encontrar algoritmos que permitan relacionar de forma fiable los valores reales medidos con los valores almacenados en la base de datos, que muy probablemente, no coincidirán nunca o casi nunca por distintos motivos (el móvil no se ubica normalmente en puntos concretos que hemos almacenado, la señal presenta variaciones temporales, etc.).

  Los numerosos estudios realizados sugieren que una de las técnicas de localización más adecuada para interiores es la basada en las medidas RSSI-fingerprinting, pero su costosa implementación hace poco atractiva su implantación.

5. Implementación del caso de uso 1

Recordamos que este caso de uso tiene por objetivo conocer la posición en el espacio de un activo, siendo el activo el que solicita conocer su posición. Además de almacenar la última posición del activo calculada por el sistema, el sistema permitirá guardar un histórico de las posiciones calculadas.

Para la implantación de este sistema necesitaremos una serie de beacons que actuarán como elementos pasivos, cuya posición fija en el espacio deberemos conocer, y un dispositivo con Bluetooth, en este caso un dispositivo móvil, será el encargado de conseguir la distancia entre él y los beacons, siempre que éstos se encuentren en su rango de conexión. Una vez obtenidos estos datos, mediante el cálculo que se explicará a continuación, se obtendrá la posición del activo.

La transmisión entre los beacons y el activo usando esta tecnología es simple:

1. Los beacons emiten señales por broadcast.
2. Estas señales serán captadas por dispositivos móviles que cuenten con tecnología Bluetooth, siempre y cuando estén adaptados al reconocimiento de estas transmisiones.
3. Estas señales portarán una información que será utilizada por el dispositivo móvil para realizar el cálculo de su propia posición, y posteriormente transmitir este dato calculado a un servidor central.

5.1. Requisitos del sistema

Los requisitos funcionales de la aplicación a desarrollar son los siguientes:

- Una vez arrancada, la aplicación calculará de manera continuada la posición del dispositivo, siempre y cuando se encuentre dentro del rango de acción detectado por el sistema.
- La aplicación mostrará en un mapa la última posición calculada para el dispositivo.

Los requisitos no funcionales de la aplicación son:

- La aplicación será compatible con sistema operativo Android 4.0.3 o superior.

5.2. Diseño de la aplicación

En este apartado se detallará la estructura del sistema.

5.2.1. Arquitectura del sistema

La figura 5.1 esquematiza la arquitectura del sistema. Existen diferentes subsistemas involucrados:

- **Aplicación móvil:** sistema que lanzará la aplicación, realizará los cálculos necesarios para obtener la posición y mostrará dicha posición en una interfaz.
- **API REST**: actuará como intermediario entre la aplicación móvil y la base de datos para realizar el intercambio de datos entre ambos.
- **Base de datos**: almacenará los datos necesarios para el correcto funcionamiento del sistema (beacons, elementos móviles, localizaciones…).
- **GeoServer**: servidor sobre el que representaremos nuestros datos geográficos.
- **Web**: página web sobre la que no solo pintaremos los datos de un activo, como sucede en la aplicación móvil, si no que se verán representados los datos de los diferentes activos generados por la aplicación móvil. Cabe destacar que la realización de esta web no ha formado parte de este proyecto.

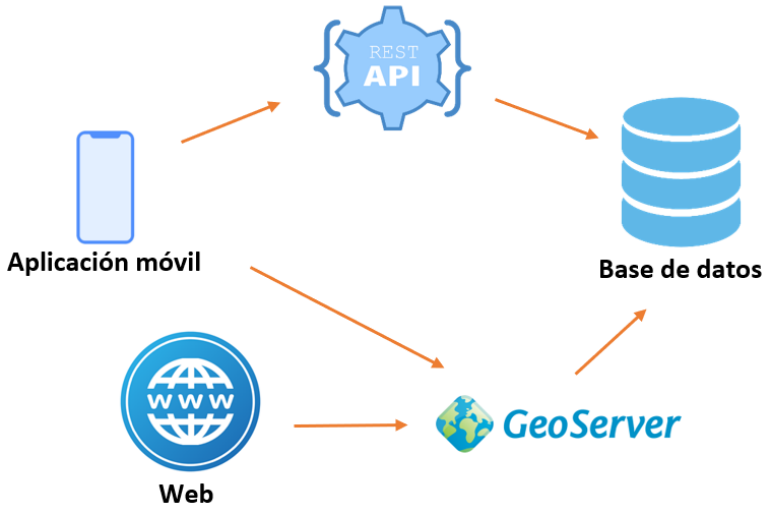


Figura 5.1: *Relación entre sistemas*

Estas relaciones a alto nivel se explican de manera más detallada mediante el diagrama de secuencias representado en la figura 5.2. Este diagrama de secuencias muestra las acciones que se desencadenan a la hora de ejecutar la aplicación móvil. Al arrancar la aplicación, esta comenzará obteniendo datos, como son los planos del edificio, que serán obtenidos del GeoServer para su posterior representación en la interfaz. A continuación, se obtendrán las posiciones de los beacons que se encuentren en la base de datos, utilizando la API REST para realizar esta comunicación. Por último, se calculará la posición en la que se encuentra el activo que ha lanzado la aplicación representándola en la interfaz y mandando estos datos a la base de datos, de nuevo, a través de la utilización de la API REST.

En el diagrama se observa la falta de interacción con la web, esto se debe a que la web interacciona de forma asíncrona a la propia ejecución de la aplicación. La web únicamente se relaciona con el GeoServer, al que mandará una petición para obtener las diferentes capas que posteriormente se representarán sobre su interfaz.

Algo parecido sucede con el GeoServer, este sistema además de las relaciones comentadas o representadas anteriormente, pedirá a la base de datos la información necesaria para una correcta representación de los datos.

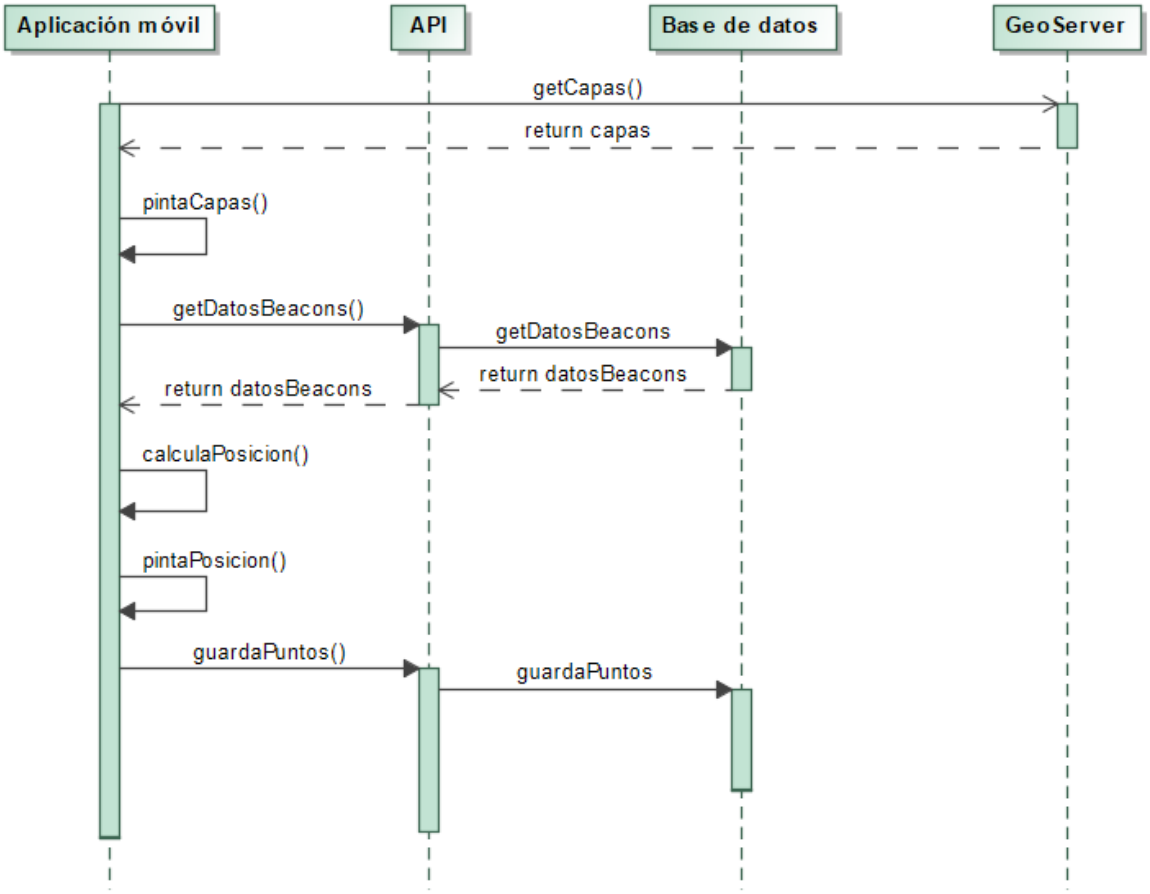


Figura 5.2: *Flujo de interacción básico del sistema*

5.2.2 Arquitectura de la aplicación móvil

La aplicación móvil ha sido desarrollada con Android Studio siguiendo el patrón modelo vista controlador (MVC) [22]. La base de este modelo consiste en separar el código implementado en tres capas:

- **Modelo:** capa en la que se trabaja con los datos. En este sistema, se accede al modelo a través del servicio REST.
- **Vista:** capa que implementa la visualización de las interfaces de usuario.
- **Controlador:** contiene las operaciones a realizar con el modelo, funcionará de enlace entre las dos capas comentadas con anterioridad.

5.2.3. Diseño del servicio REST

La API REST ha sido desarrollada de manera íntegra y su diseño (o interfaz) es el que se muestra en la tabla 5.1.

Tabla 5.1. *Diseño del servicio REST*

Recursos	URI	MÉTODOS
Beacons	/beacons	POST
Beacon	/beacons/{idBeacon}	PUT GET DELETE
Última localización de un elemento móvil	/elementoMovil/{idElemento}/Localizacion	PUT
Localización histórica de un elemento móvil	/elementoMovil/{idElemento}/LocalizacionesHistorico	POST

En el futuro se podrán aumentar los elementos existentes en esta tabla, es decir, los recursos accesibles desde el servicio REST, pero en este momento, con el desarrollo de esta disposición, es suficiente para implantar nuestro sistema.

5.2.4. Diseño detallado

Para dar soporte a la funcionalidad de la aplicación, es necesario extender el modelo de dominio que se mostró en la sección 4.1, siendo el diagrama de clases resultante el que se muestra en la figura 5.3.

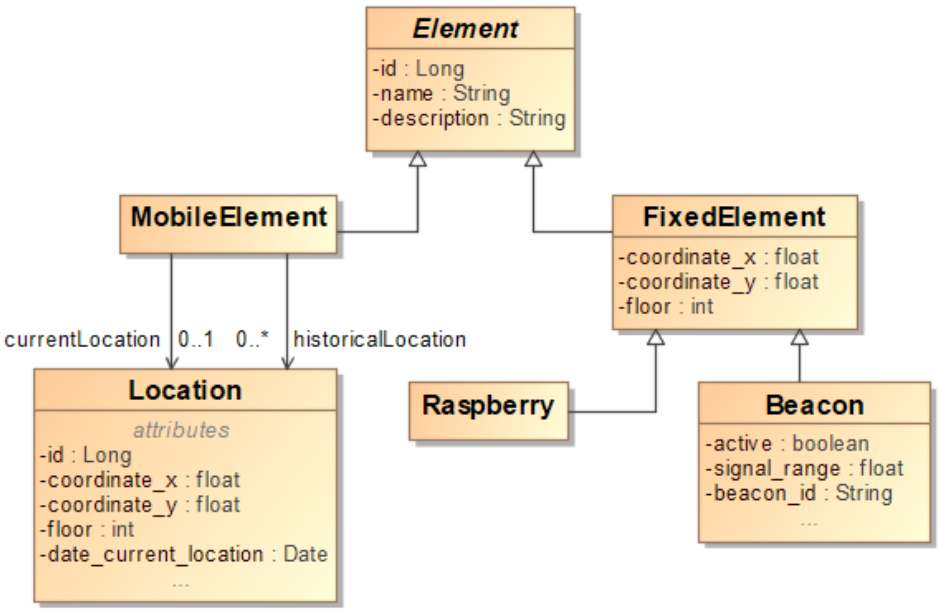


Figura 5.3: *Diagrama de la base de datos con localizaciones*

5.3. Implementación del sistema

Durante este capítulo se explicarán algunos de los detalles sobre la implementación de la aplicación móvil y del servicio REST.

5.3.1. Implementación de la aplicación móvil

En este apartado se detallará la forma en que se han implementado los algoritmos necesarios para obtener las distancias a los dispositivos, así como las posibles variables que se pueden modificar o ajustar para poder obtener una localización más próxima a la realidad.

Fórmula para obtener la distancia

Los beacons no proporcionan directamente su posición respecto a nosotros, pero ofrecen datos que nos permiten calcularla. Entre estos datos se encuentran el RSSI y el poder de transmisión.

Una vez conseguidos estos datos podemos aplicar dos fórmulas diferentes para el cálculo de la distancia en metros entre el beacon y el elemento móvil:

- Distancia = x * (RSSI/Poder de transmisión) ^y+z

  En esta ecuación encontramos las variables 'x','y','z'. Estas variables son calculadas en función de las características del servicio desde el que se mande la petición a los beacons. En el caso de no reconocer las características del dispositivo se calcularán con unas variables por defecto.

- Distancia = 10 ^ ((Poder de transmisión - RSSI) / 10n)

  De nuevo, en esta ecuación encontramos una variable desconocida, en este caso 'n'. Con esta variable representamos la pérdida por propagación, que será un valor variable entre 2 y 4, siendo 2 la pérdida por propagación en el vacío.

Se optará por la segunda fórmula, al tener únicamente una variable dependiente del entorno y no del dispositivo desde el que se realice la acción.

Diferencias entre trilateración y cálculo con eje radical

La distancia entre el dispositivo móvil y los beacons la calcularemos mediante la fórmula anteriormente comentada. En el caso en el que el cálculo de estas distancias fuera preciso, optar por la técnica de trilateración para el cálculo de la posición del activo sería lo más acertado.

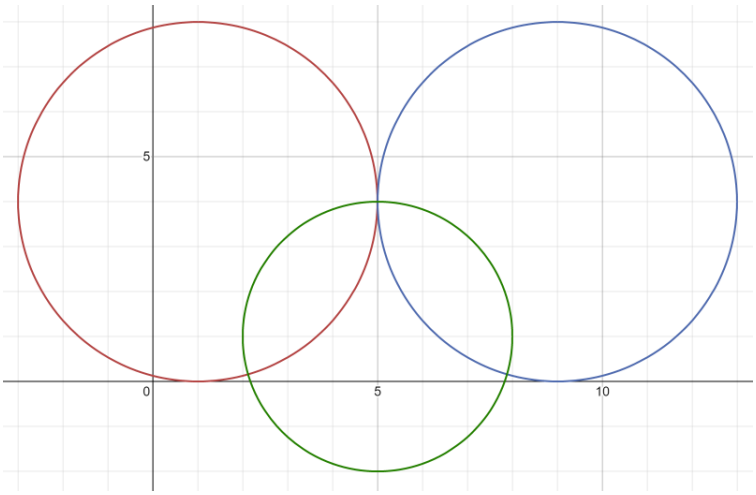


Figura 5.4: *Trilateración 2D supuesto ideal*

Como ejemplo, al ejecutar nuestro cálculo sobre una situación como la que se muestra en la figura 5.4, donde cada circunferencia tiene por centro la posición de un beacon y por radio la distancia obtenida entre dicho beacon y el activo, la posición obtenida para el activo por nuestro programa tal y como se muestra a continuación, como ha sido comentado, sería correcta:

```
El punto en el que nos encontramos es:
Punto [coordenadaX=5.0, coordenadaY=4.0]
```

Uno de los factores de la fórmula utilizada para el cálculo de la distancia es el RSSI, este valor en interiores no será preciso, ya que no se tiene en consideración los obstáculos, ni las refracciones ni difracciones, por lo que las distancias obtenidas no serán precisas haciendo que esta situación ideal sea poco probable.

Una situación más realista es la representada en la figura 5.5.

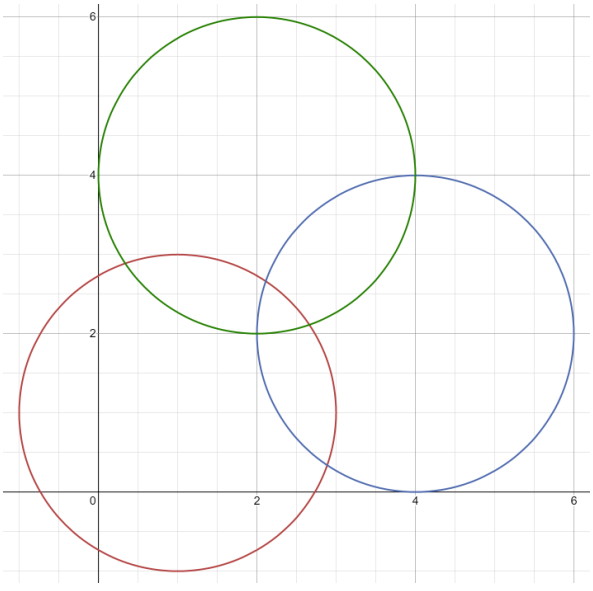


Figura 5.5: *Trilateración 2D supuesto real*

Si aplicamos la técnica sobre este supuesto, las coordenadas del punto que nos calcula ni siquiera se encuentran dentro del área de los tres círculos:

```
El punto en el que nos encontramos es:
Punto [coordenadaX=2.58113883008419, coordenadaY=1.7905694150420948]
```

Una técnica que nos puede ayudar a la hora de ajustar este punto es calcular los ejes radicales [23] entre las circunferencias dos a dos. Un eje radical es una recta perpendicular al segmento determinado por los dos centros de las circunferencias. Teniendo estas rectas se calcularán los puntos de corte entre ellas.

Si se representan los ejes radicales sobre el escenario anterior, la situación obtenida es la que se muestra en la figura 5.6.

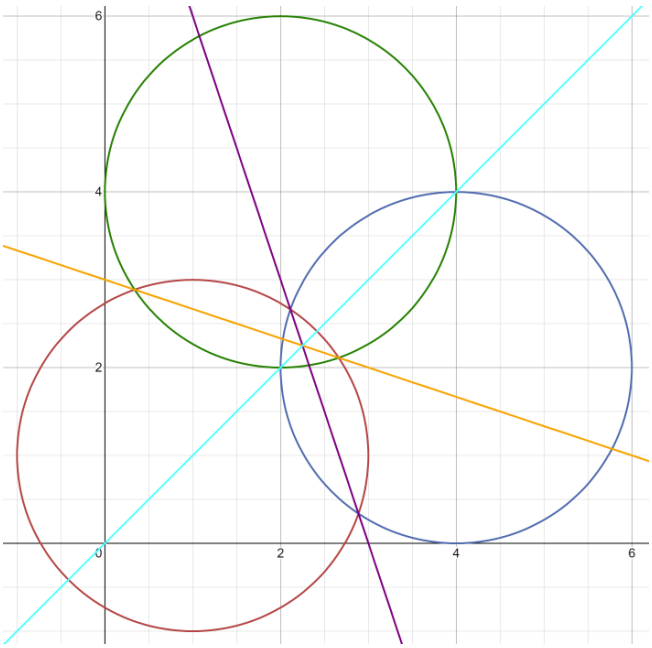


Figura 5.6: *Ejes radicales 2D tres círculos supuesto real*

En este caso, para obtener la posición que queremos calcular, se debe tener en cuenta los puntos de corte de los diferentes ejes. Con este contexto el punto calculado con nuestro sistema será:

```
El punto en el que nos encontramos es:
Punto [coordenadaX=2.25, coordenadaY=2.25]
```

Se observa que el punto obtenido en este caso no solo se encuentra dentro del área perteneciente a los tres círculos, sino que lo posiciona en una zona centrada en esta área.

Además, aplicando esta técnica, se amplía el número de circunferencias que podemos tener en cuenta en el sistema, teniendo un mínimo de tres como en el caso anterior. En la figura 5.7 se expone una situación en la que se detectarán cuatro beacons.

Una vez identificada la posición y la distancia a la que se encuentran los beacons, deberemos calcular sus respectivos ejes radicales, que serán representados en la figura 5.8.

El número de rectas representando a los ejes radicales dependerán del número de circunferencias que tengamos en nuestro sistema y se relacionará con la fórmula $\frac{n*(n-1)}{2}$ siendo 'n' el número de circunferencias de nuestro sistema.

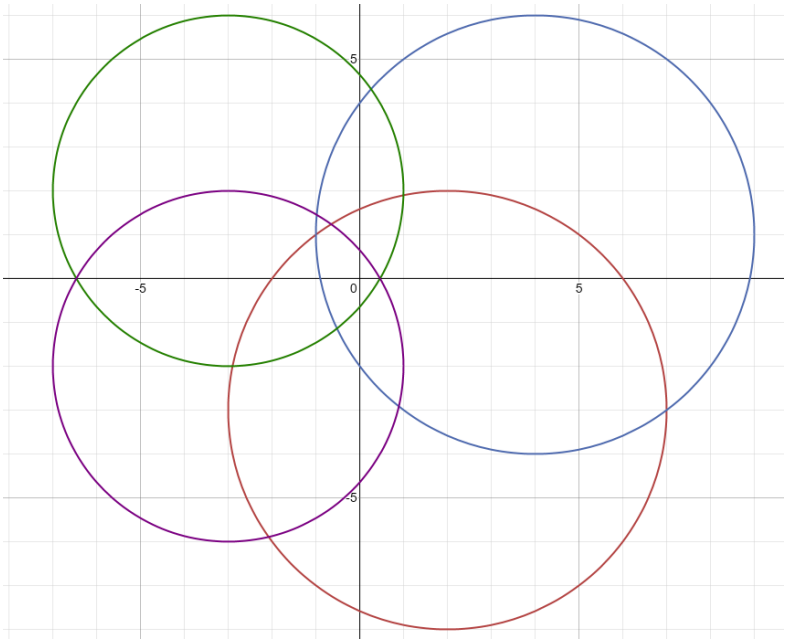


Figura 5.7: *Cuatro círculos 2D*

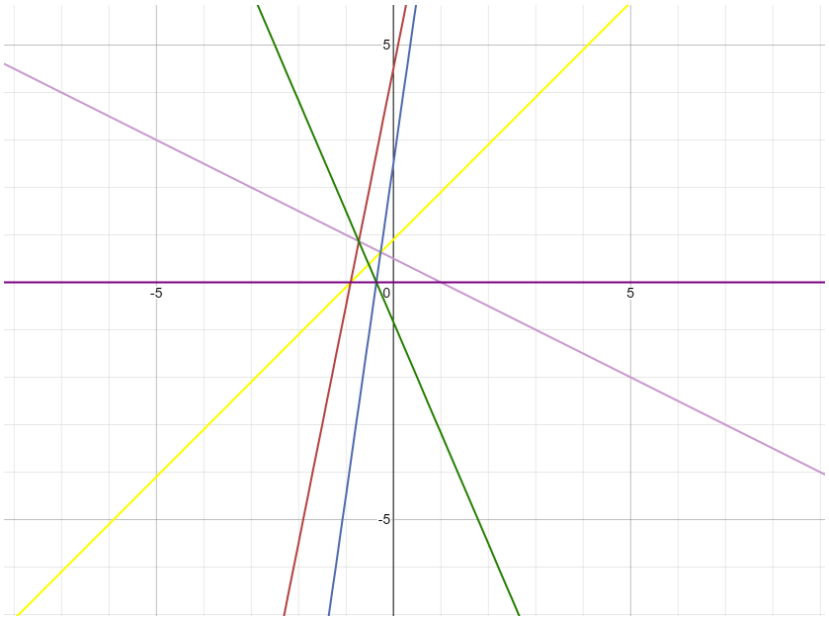


Figura 5.8: *Rectas de ejes radicales*

La unión de las dos gráficas anteriores tiene como resultado la figura 5.9:

Una vez obtenidas tanto las distancias a los beacons, que serán representadas con los círculos, como sus ejes radicales, se podrá ejecutar nuestro sistema de cálculo, obteniendo el siguiente punto:

```
El punto en el que nos encontramos es:
Punto [coordenadaX=-0.35154978354978345, coordenadaY=0.9580606060606062]
```

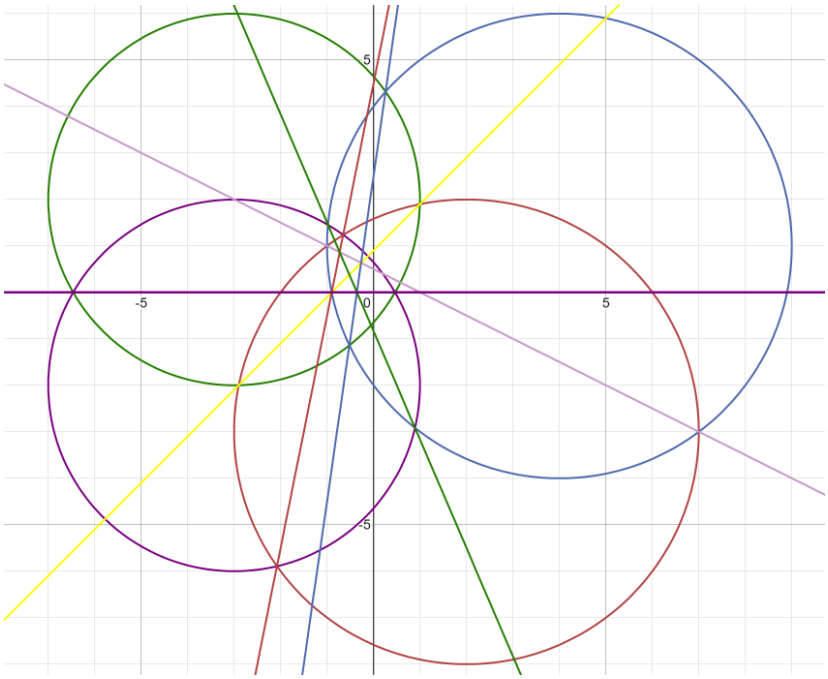


Figura 5.9: *Cuatro círculos 2D con ejes radicales*

Los puntos de corte entre los ejes radicales en este caso tienen diferentes valores, pero el resultado final solo puede ser un punto, por lo que nos quedaremos con la media geométrica de estos resultados.

Durante la aplicación de este sistema también podemos encontrar complicaciones, como la que se produce cuando dos ejes radicales son prácticamente paralelos, lo que hará que el punto de corte entre ambos tenga una posición muy alejada de la real.

A continuación, se simulará una situación en la que la posición del activo es el origen de coordenadas y los datos de los beacons obtenidos son los representados en la figura 5.10.

En esta situación, como se ve en la representación gráfica, la posición que calcularía el sistema sería:

```
El punto en el que nos encontramos es:
Punto [coordenadaX=2.125, coordenadaY=0.5]
```

Este es un caso simplificado, ya que únicamente se cuenta con los datos de tres beacons, pero ya se puede observar que alguno de los puntos de corte de los ejes puede alejarse demasiado de la realidad.

Si se añade un beacon más a esta situación, se puede encontrar un escenario como el que se representa en la figura 5.11:

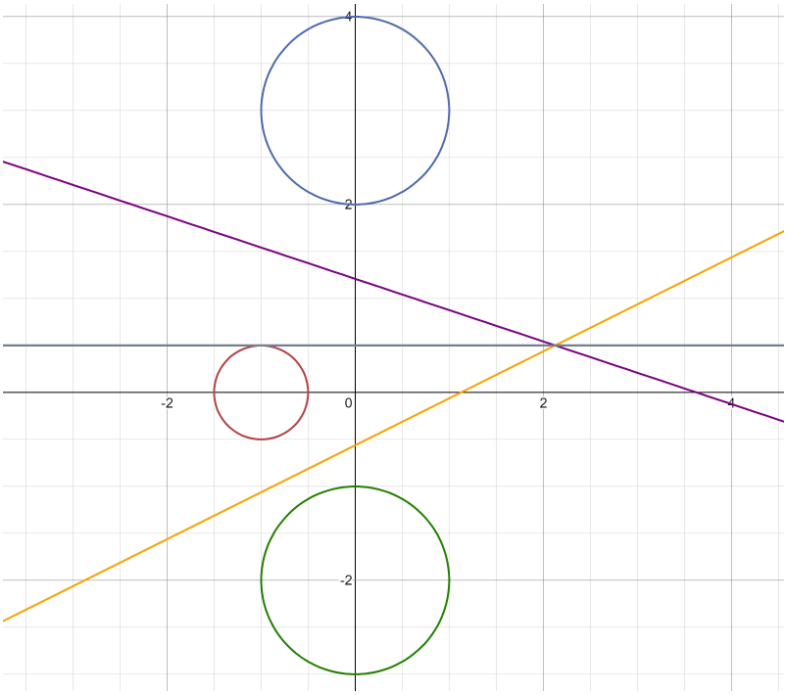


Figura 5.10: *Ejes radicales 2D tres círculos supuesto real*

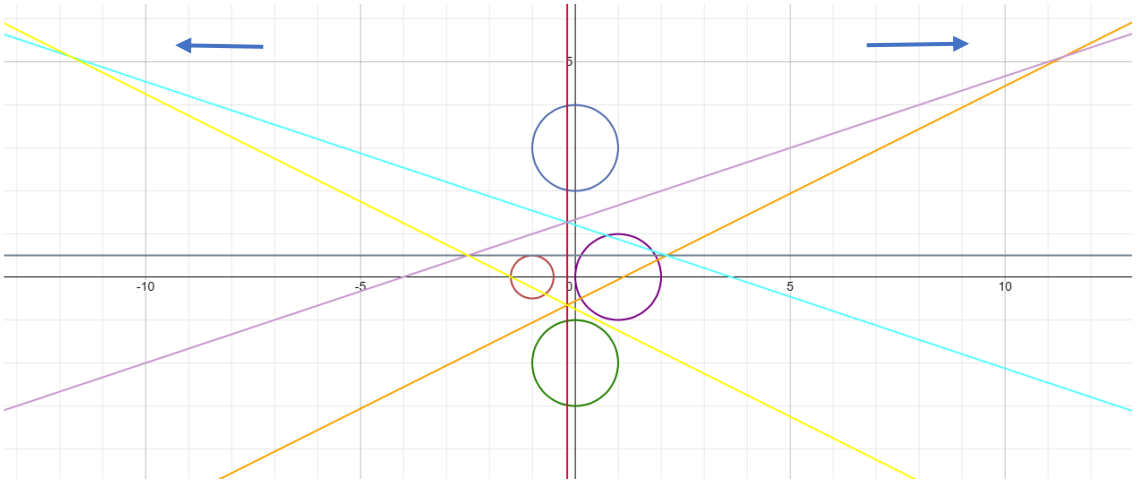


Figura 5.11: *Ejes radicales 2D cuatro círculos supuesto ideal*

Se puede observar que la mayoría de los puntos de corte se sitúan en torno al eje de coordenadas, posición en la que se encuentra el activo, pero en este caso se encuentran dos puntos bastante alejados de esta zona, haciendo que la media se vea alterada. En el cálculo final de la posición se despreciarán estos valores, obteniendo finalmente el valor:

```
El punto en el que nos encontramos es:
Punto [coordenadaX=-0.1875, coordenadaY=1.0395833333333333]
```

Implementación del cálculo de la posición

Al arrancar la aplicación se activará un manejador de evento que será lanzado cada vez que se reciba al menos una señal de beacon (la especificación de este manejador la define la librería propia de los beacons). Este manejador es el encargado de realizar los cálculos necesarios que obtienen la posición del dispositivo. El pseudocódigo del algoritmo implementado por el manejador es el mostrado a continuación:

```
For (beacons detectados)
      Calcula y guarda distancias
End for
if (Beacons detectados>=3)
      Calcula punto(beacons detectados)
      Pinta punto
      POST REST

End if
```

Se ha implementado la clase "Operacion", la cual contiene los principales métodos utilizados para el cálculo de la posición. Esta clase será utilizada desde el manejador previamente comentado. Los métodos existentes en esta clase se representan en la figura 5.12.

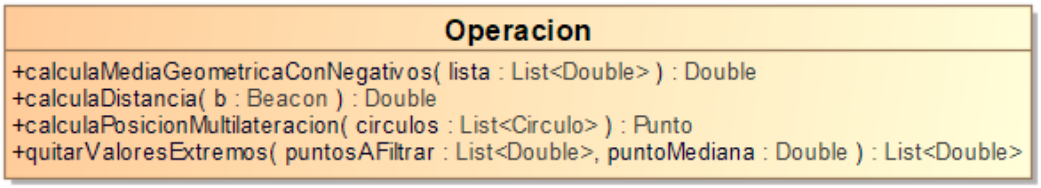


Figura 5.12: *Métodos clase Operación*

Ajustes para incrementar la precisión

El sistema de localización calculado con el RSSI de los beacons no tiene una alta precisión, por lo que es necesario realizar algunos ajustes en el sistema para conseguir que el resultado se aproxime más a la realidad. Estos ajustes se han implementado como una sección en la aplicación móvil que se muestra en la figura 5.13.

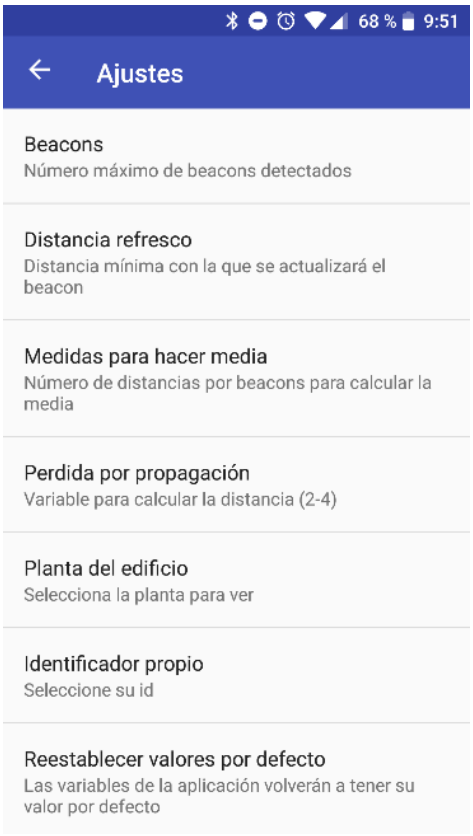


Figura 5.13: *Pantalla de ajustes en aplicación móvil*

Beacons considerados para realizar el cálculo

Como se ha comentado anteriormente, el número mínimo de beacons para realizar el cálculo de la posición mediante ejes radicales es tres, pero no se ha detallado el número máximo. Se ha explicado que el número de rectas que calculará el sistema será $\frac{n*(n-1)}{2}$ siendo 'n' el número de beacons cuya distancia contemplamos, por lo que la cantidad de rectas a calcular irá creciendo de forma exponencial junto a sus correspondientes puntos de corte por cada dispositivo. La elección de este valor depende del trabajo que le queramos dar al terminal y de la precisión con la que pretendamos dotar al sistema, aunque cabe destacar que la aplicación calculará la posición a partir de obtener los valores de tres beacons.

En nuestro caso, se ha elegido contemplar como máximo cinco, los cinco más cercanos, al entender que superar ese valor sería una cantidad de operaciones excesivas para nuestro terminal. La aplicación nos permitirá ajustar este valor en un umbral entre tres y cinco, como se muestra en la figura 5.14. Para implementar este cambio se añadirá una condición al if del pseudocódigo comentado anteriormente, por la cual, en caso de conseguir distancias de seis o más beacons, únicamente se tendrán en cuenta las cinco distancias cuyo valor sea menor.

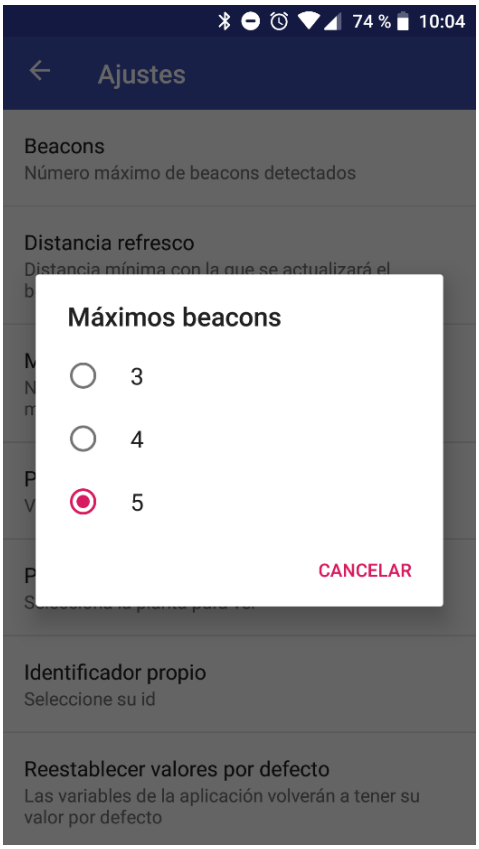


Figura 5.14: *Elección de máximo número de beacons a tener en cuenta*

Rango de los beacons

Existen diferentes posibilidades a la hora de obtener un tipo de beacon, en nuestro caso la empresa elegida ha sido Estimote, cuyo diseño es el mostrado en la figura 3.1. Este tipo de dispositivo tiene una serie de parámetros que podemos configurar a nuestro gusto, entre los que podemos observar alguno meramente informativo como tags o nombres que se le pueden otorgar, u otros que se pueden tener más en cuenta en su funcionamiento como son el tipo de tecnología que usará (Eddystone o iBeacon) o la fuerza de transmisión de su señal medida en dBm, siendo esta segunda mucho más importante para nuestra aplicación.

Esta potencia la podremos variar desde un mínimo de -40dBm hasta un máximo de 10dBm. Al aumentar este valor el rango de emisión de la señal Bluetooth emitida por el dispositivo es mayor, aumentando a su vez el consumo, por lo que la batería interna tendrá una menor duración. Lo lógico en un principio, si despreciamos el tema de la batería, es otorgar al beacon la mayor potencia posible, pero al hacer las pruebas se observa que cuanto mayor sea la distancia entre el beacon y el elemento a localizar mayor oscilación puede haber entre la medida de distancia obtenida y la medida real.

Buscando siempre obtener la mayor precisión posible, se ha optado por limitar el rango máximo de los beacons hasta un máximo de unos siete metros, detectando así únicamente los beacons en una distancia media, cuya medición será más aproximada.

La representación de este valor dentro de los ajustes del beacon se muestra en la figura 5.15.

Figura 5.15: *Ajustes rango máximo beacons*

Distancia de refresco

Otra de las variables que se tienen en cuenta en la aplicación tiene que ver con la representación del punto que simboliza la localización del activo. Se trata de la distancia necesaria entre diferentes cálculos de posición para que este cambio se vea representado en el sistema. Parece necesaria la implantación de esta variable, ya que la distancia que obtenemos entre los beacons y el activo sufre continuas variaciones como se ha detallado anteriormente. Estos cambios se verían reflejados en nuestro mapa, representando, aunque nuestra posición real no variara, un continuo movimiento ficticio. Esta variable tendrá un valor por defecto de dos metros.

Medidas para hacer la media

Se ha comentado anteriormente que los valores de distancia obtenidos por los beacons no tienen una alta precisión y se pueden ver alterados por diferentes factores externos independientes a nosotros.

Para intentar disminuir esta dificultad se calculará la media de una serie de distancias, que posteriormente será suministrada al sistema como distancia entre el activo y el beacon reduciendo así la variación y la posible diferencia entre el valor real y el obtenido. El número de valores necesarios para realizar la media de estos valores se ha establecido en cinco. Este cambio se reflejará en el bucle del pseudocódigo comentado anteriormente, donde solo se considerará válida una distancia entre el activo y un beacon después de obtener cinco distancias entre estos dispositivos, pudiendo realizar la media entre estos valores. Cabe resaltar, que en base a esto, la localización del activo se calcula, en el mejor de los casos, cada 1.5 segundos, ya que los beacons emiten señales broadcast cada 300ms.

Diferencias entre media aritmética y geométrica

Normalmente cuando hablamos de media, entendemos que nos estamos refiriendo a la media aritmética. Como sabemos, esta se calcula sumando todos los valores y dividiéndolo por el número total de ellos. Esta media en nuestro caso puede verse alterada, ya que nuestro sistema posee numerosos valores extremos (ocasionados por ruido o diferentes circunstancias), por lo que hemos optado por utilizar la media geométrica, que se verá menos alterada por la existencia de estos valores.

Desechar valores obsoletos

Hay que tener en cuenta que la media de diferentes valores puede verse afectada por factores temporales o espaciales.

Pensemos en un aula con dos puertas “A” y “B”, con una amplia distancia entre ambas, en la que obtenemos cuatro valores de distancia entre el dispositivo y un beacon sin obtener el quinto necesario para obtener la media, salimos por la puerta “A”, perdiendo el rango de detección del beacon. Si tiempo después volvemos al aula y al rango de visión del beacon utilizando la puerta “B”, obtendremos una nueva medida, con lo que ya se puede calcular la media. En este contexto cuatro de los valores de la media se corresponderían a una parte del área, la próxima a la puerta ‘A’, mientras que el último ha sido obtenido de la zona cercana a la puerta ‘B’, por lo que el resultado no será preciso. Esto se soluciona llevando la cuenta de las interacciones cada vez que detectamos algún beacon, eliminando cálculos de distancias cuya interacción en la que fue obtenida y la actual difiera de un número variable, en nuestro caso diez. Esta situación estará representada en las figuras 5.16 y 5.17.

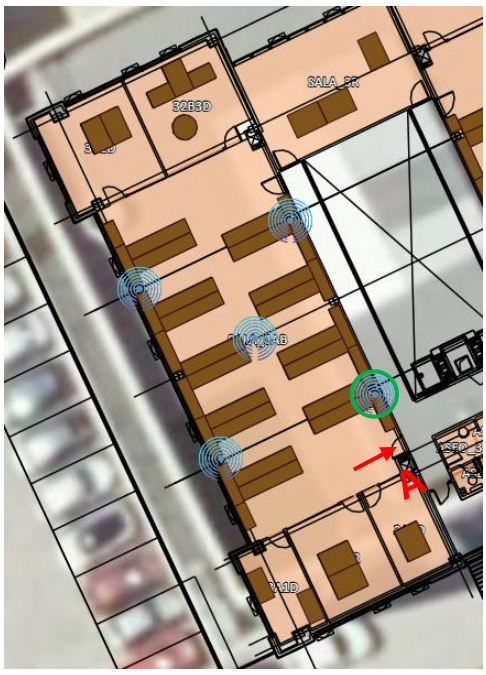


Figura 5.16: *Beacons conectados puerta A*

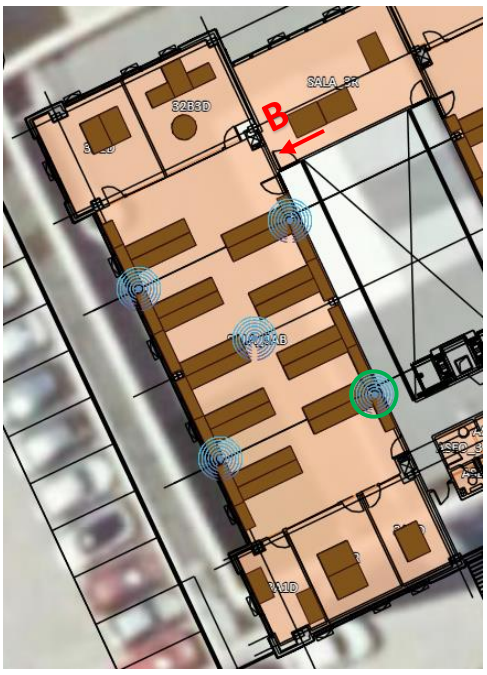


Figura 5.17: *Beacons conectados*

5.3.2. Implementación de servicio REST

En el punto 5.2.3. se comentó a alto nivel la interfaz del servicio REST, en la figura 5.18 se mostrará el diagrama de clases correspondiente a la implementación de dicho servicio.

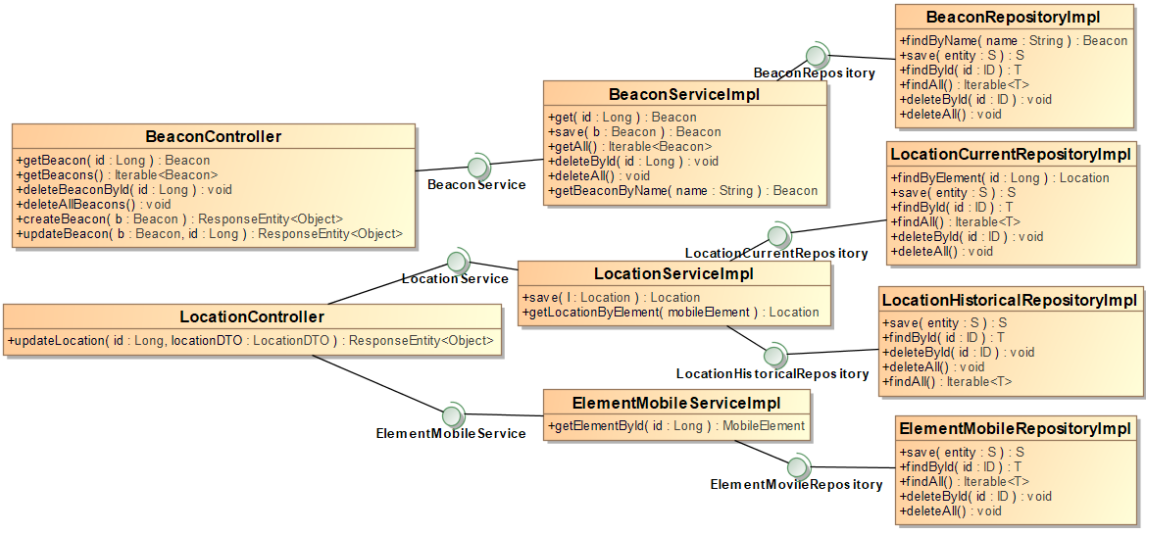


Figura 5.18: *Diagrama de componentes*

El sistema ha sido implementado utilizando Spring siguiendo una arquitectura de tres capas. Podemos diferenciar las capas Controller, Service y Repository. Las clases pertenecientes a la capa Controller serán las encargadas de implementar el servicio REST, las clases Repository interactúan con la base de datos y, por último, las clases pertenecientes a la capa Service relacionan las clases de las otras capas comentadas anteriormente.

5.4. Pruebas del sistema

En esta sección se pondrá a prueba el correcto funcionamiento del sistema mediante la realización tanto de pruebas unitarias como de integración.

5.4.1 Pruebas unitarias

Se comprobará si los valores obtenidos por los métodos que calculan la posición retornan los resultados deseados, para ello se realizarán pruebas unitarias sobre dichos métodos con un conjunto de valores de entrada previamente definidos.

Inicialmente se comprobará el correcto funcionamiento de métodos que participan en el cálculo de la multilateración. Estos métodos son los existentes en la clase “Operación”, los cuales se observan en la figura 5.12. La correcta funcionalidad de cada uno de estos métodos ha sido comprobada.

El método “calculaPosicionMultilateracion” utiliza el resto de las funciones existentes en su clase, por lo que al corroborar su correcto funcionamiento comprobaríamos cada uno de los métodos que contiene, aunque también se han definido pruebas para probar dichos métodos de manera independiente.

El método “calculaPosicionMultilateracion” será comprobado para tres, cuatro o cinco beacons. Un ejemplo de esta comprobación se representa en la figura 5.19.

```java
@Test
public void multilateracion4Beacons_isCorrect()
{
    List<Circulo> circulos = new ArrayList<>();
    circulos.add( new Circulo( origenX: 2, origenY: -2, floor: 3, radio: 2.0));
    circulos.add( new Circulo( origenX: -2, origenY: -2, floor: 3, radio: 2.0));
    circulos.add( new Circulo( origenX: -2, origenY: 2, floor: 3, radio: 2.0));
    circulos.add( new Circulo( origenX: 2, origenY: 2, floor: 3, radio: 2.0));

    assertEquals( expected: 0.0, Operacion.calculaPosicionMultilateracion(circulos)
            .getCoordenadaX(), delta: 0.000001);
    assertEquals( expected: 0.0, Operacion.calculaPosicionMultilateracion(circulos)
            .getCoordenadaY(), delta: 0.000001);
}
```

Figura 5.19: *Pruebas unitarias android*

5.4.2. Pruebas de integración

Se corroborará el correcto funcionamiento del servicio REST creado para la obtención y manipulación de la base de datos. Para su prueba realizaremos invocaciones sobre los métodos básicos que forman el servicio. Por lo tanto, se realizarán test ejecutando POST, GET, PUT y DELETE. En este documento únicamente se mostrará la prueba del método GET representado en la figura 5.20, pero los cuatro métodos han sido comprobados. Estas pruebas serán llevadas a cabo utilizando el programa “Postman”.

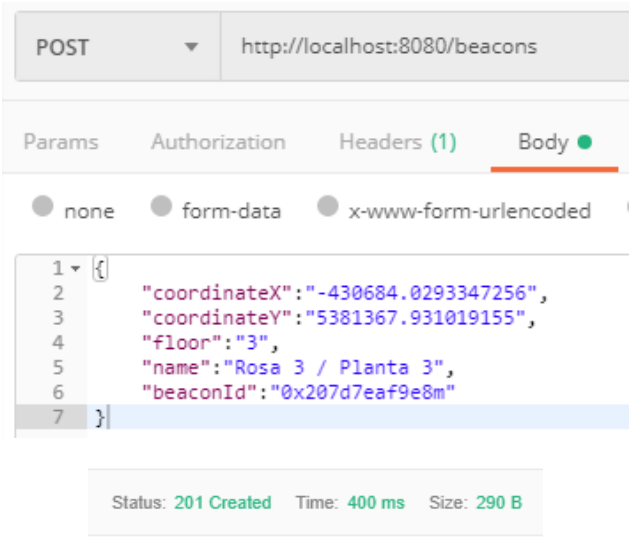


Figura 5.20: *Prueba de la creación de un beacon*

Este sistema tendrá conexión con la base de datos y a su vez con el GIS, por lo que se certificará que la conexión entre estos elementos existe y tiene además un resultado satisfactorio. Para ello, se ejecutará la aplicación en un lugar cubierto por el rango de los beacons. Y se observará que las localizaciones obtenidas se añaden a la base de datos, como se detalla en la figura 5.21.

id	coordinate_x	coordinate_y	floor	id_element_mobile	date_historical_location
583	-430.682,131772709	5.381.357,3172957655	3	2	2019-01-25 13:52:19
585	-430.683,1518446913	5.381.357,476614926	3	2	2019-01-25 13:53:08
586	-430.682,424978591	5.381.358,78708568	3	2	2019-01-25 13:55:09
587	-430.682,7447174551	5.381.357,741481857	3	2	2019-01-25 13:55:12
589	-430.682,7916950851	5.381.357,758422688	3	2	2019-01-25 13:55:30
591	-430.682,1710426479	5.381.358,600774905	3	2	2019-01-25 13:56:50
594	-430.682,6588217388	5.381.357,737255234	3	2	2019-01-25 13:57:15
596	-430.682,4210021303	5.381.357,988588183	3	2	2019-01-25 13:57:52
598	-430.682,4566570588	5.381.358,6690908	3	2	2019-01-25 13:58:39
599	-430.682,7909509263	5.381.357,67997756	3	2	2019-01-25 13:58:48
601	-430.682,5671999174	5.381.357,964905432	3	2	2019-01-25 14:00:47
603	-430.682,9024917866	5.381.357,738072823	3	2	2019-01-25 14:01:18
604	-430.681,8202422984	5.381.357,432119667	3	2	2019-01-25 14:01:56

Figura 5.21: *Posiciones almacenadas en la base de datos*

Posteriormente, se observará el GIS para confirmar que los datos guardados en la base de datos corresponden a los representados sobre el mapa, como se detalla en la figura 5.22.

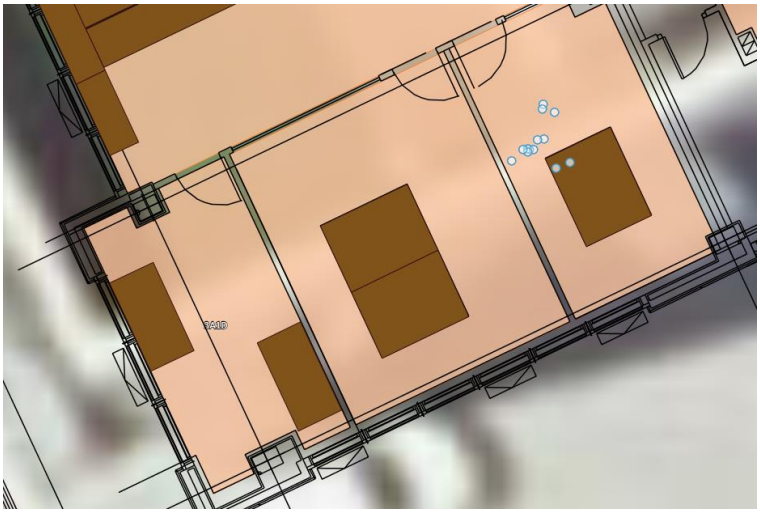


Figura 5.22: *Posiciones representadas en el mapa*

5.4.3. Pruebas funcionales

Una vez comprobado el correcto funcionamiento de los diferentes subsistemas se pueden hacer pruebas del sistema completo sobre diferentes escenarios.

Empezaremos colocando un dispositivo en el centro de un aula de unos 24m^2 con un beacon en cada una de las esquinas de esa área. Obteniendo el resultado representado en la figura 5.23.

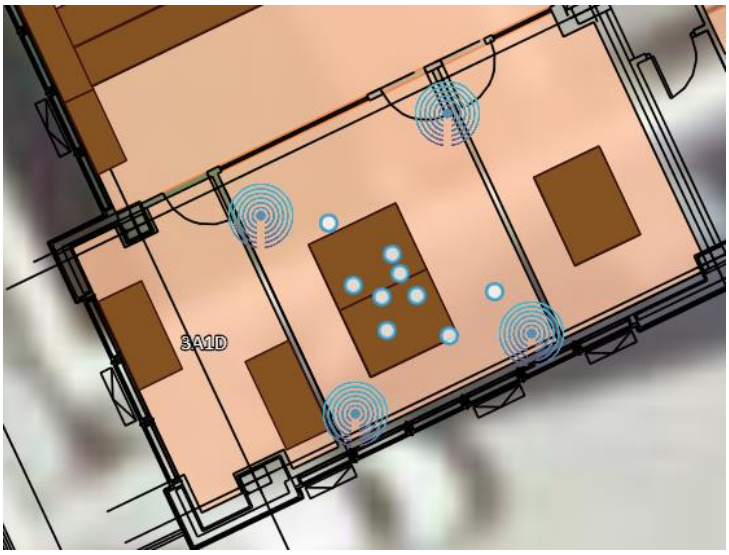


Figura 5.23: *Posiciones en un aula con 4 beacons*

Aunque nuestra posición real no ha variado, vemos como si se ha modificado la posición calculada en el sistema. Como punto positivo se observa que ningún punto lo posiciona fuera del aula en el que nos encontramos.

Se realizará la misma prueba, pero esta vez utilizando únicamente tres de los beacons, posicionados en tres de las esquinas. En esta situación obtenemos los resultados mostrados en la figura 5.24.

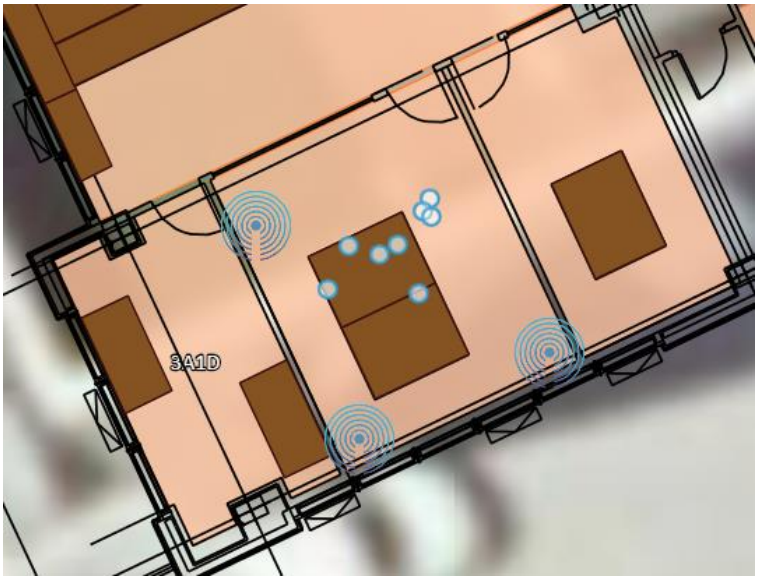


Figura 5.24: *Posiciones en un aula con 3 beacons*

En este caso se observa una dispersión de las posiciones calculadas similar al caso anterior, encontrando la totalidad de los puntos dentro del aula nuevamente.

Ahora se debería probar el comportamiento de los beacons si se quiere cubrir un área separada por paredes. Para realizar esta situación modificaremos el anterior caso

colocando dos beacons en la sala contigua y colocando el dispositivo que realizará el cálculo en esta nueva sala. Los resultados obtenidos son los reflejados en la figura 5.25.

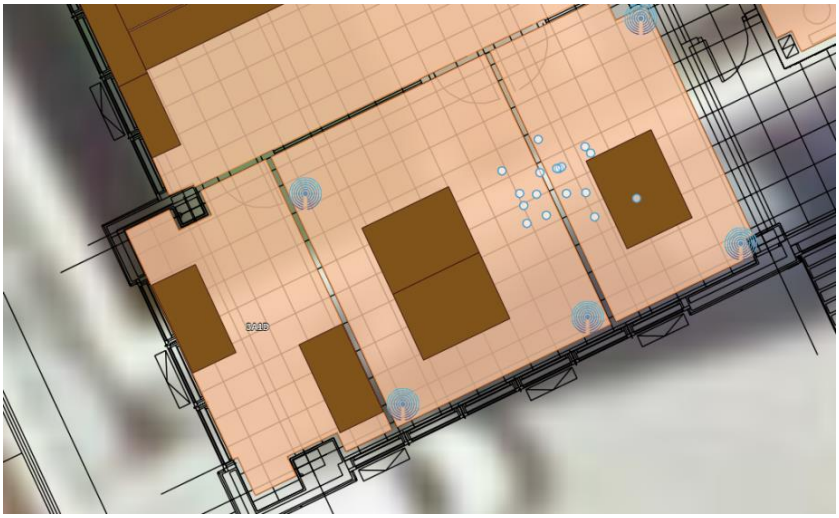


Figura 5.25: *Posiciones en dos aulas con 5 beacons*

Se observa que los resultados obtenidos no cuentan con una alta precisión, por lo que se deberán realizar modificaciones para intentar dotar al sistema de mayor exactitud.

Se trabajará sobre un caso en el que el área a cubrir consta de tres salas independientes separadas por paredes, después de probar diferentes distribuciones de beacons para que cubran las tres salas podemos diferenciar dos como las más funcionales:

- Una en la que utilizamos siete beacons, tres en la sala central y dos en cada una de las contiguas, como se representa en la figura 5.26:

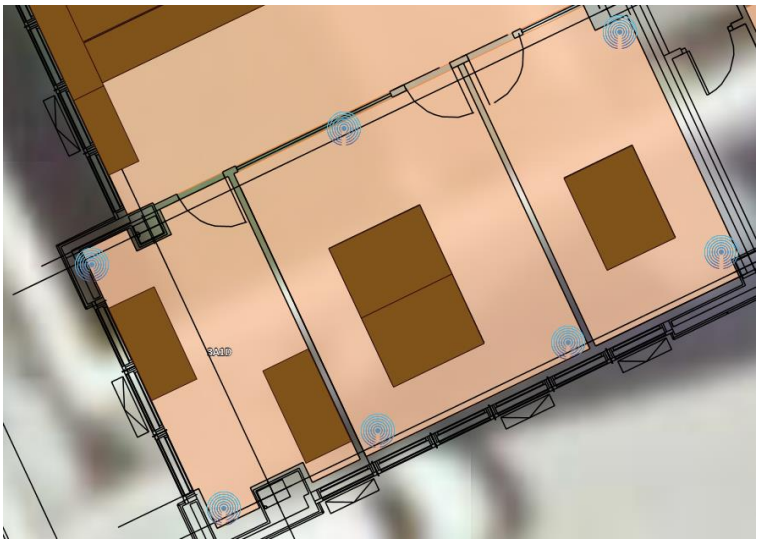


Figura 5.26: *Posiciones en tres aulas con 7 beacons*

- Otra en la que utilizamos nueve beacons, repartiendo de forma equitativa los beacons por cada una de las salas, como se ve representado en la figura 5.27:

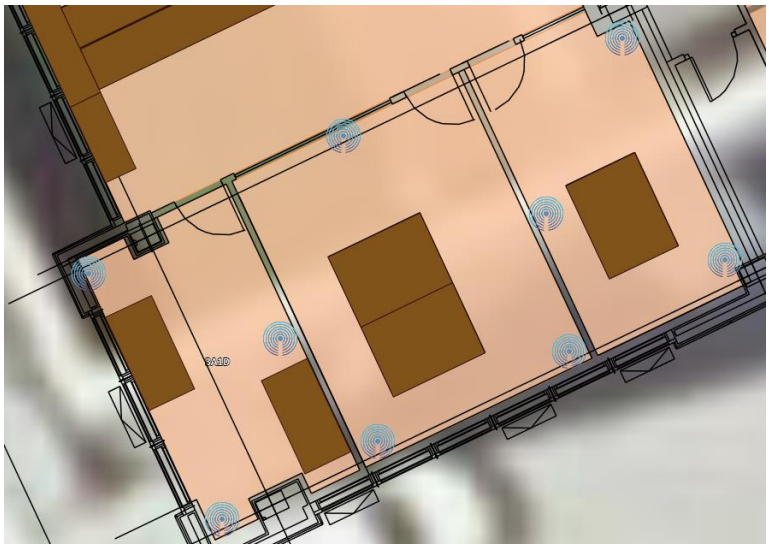


Figura 5.27: *Posiciones en tres aulas con 9 beacons*

Si el activo se encuentra en la sala central los resultados obtenidos para estas distribuciones son los mostrados en las figuras 5.28 y 5.29.

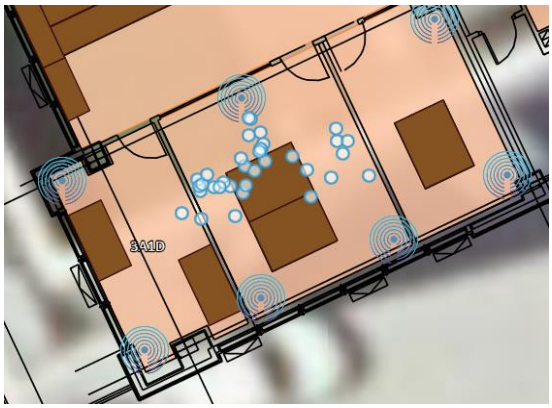
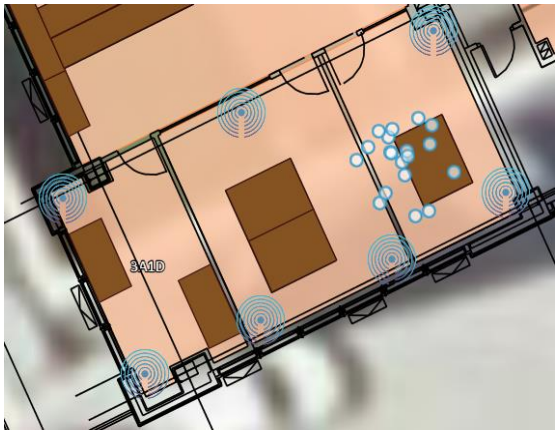


Figura 5.28: *Sala central siete beacons*

Figura 5.29: *Sala central nueve*

Se observa que la mayoría de los puntos obtenidos se encuentran dentro del aula correcta.

Con las mismas disposiciones, se comprobarán los datos obtenidos al situar el activo en una de las salas contiguas a la anterior, las posiciones calculadas son las representadas en las figuras 5.30 y 5.31.



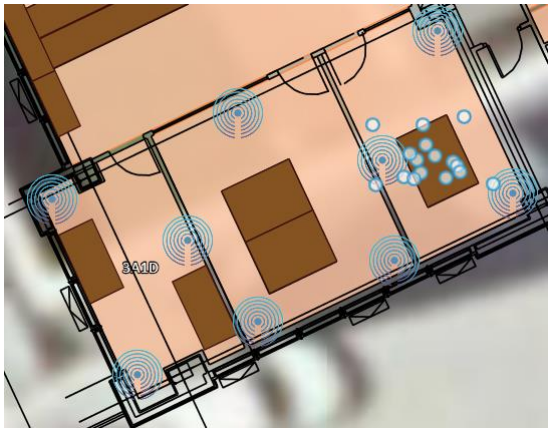
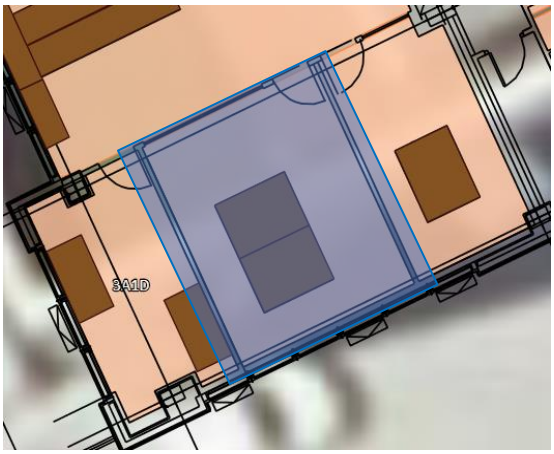


Figura 5.30: *Sala derecha siete beacons* ***Figura 5.31:*** *Sala derecha nueve beacons*

En este caso se consiguen unos datos similares a los obtenidos en la situación anterior, en ambas situaciones se observa que la gran mayoría de posiciones se sitúan dentro del aula en la que se encuentra el activo, aunque también se observa un porcentaje de puntos que se encuentran al limite entre aulas o incluso sobrepasando este límite. A la hora de gestionar los datos, para conocer la sala en la que se encuentra la persona, se podrá considerar como válida su posición en el aula si un alto porcentaje de posiciones calculadas se encuentra dentro de ella. Otra posibilidad es que el sistema considere como área de la sala una superficie algo superior a la real, como se detalla en las figuras 5.31 y 5.32.



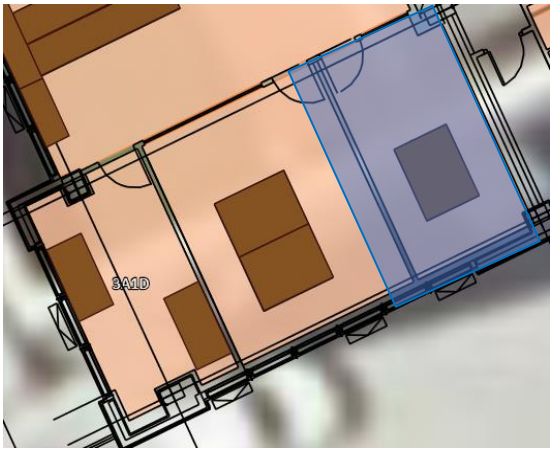


Figura 5.31: *Superficie central a tener en cuenta* ***Figura 5.32:*** *Superficie lateral a tener en cuenta*

Estas medidas pueden ser ejecutadas de forma independiente o de forma complementaria.

6.Implementación del caso de uso 2

Recordamos que este caso de uso trata de conocer la posición de los activos que se encuentren en el rango de espacio que sea detectado por nuestro software sin que éstos pidan ser localizados.

Los requisitos de la aplicación en este caso son los siguientes:

- El sistema deberá realizar una búsqueda de un activo identificado para conocer si este se encuentra en un plano del recinto controlado.
- El sistema permitirá conocer la sala en la que se encuentran los usuarios.
- El sistema permitirá conocer las posiciones de un usuario a lo largo del tiempo.
- El sistema permitirá conocer la última posición obtenida de los usuarios.
- El sistema permitirá conocer el número de activos existentes en una sala.

Se creará una implementación que tendrá un enfoque inverso al anterior caso. Los elementos fijos en el mapa serán las Raspberrys, mientras que el elemento móvil será un dispositivo que emita Bluetooth, en nuestro caso un beacon.

Cada Raspberry, con un mínimo de tres para que el sistema funcione correctamente, contactará con el mismo beacon que quiere detectar y mandará esta información a un sistema central que recopilará estos datos. Una vez obtenidas las distancias entre el beacon y las Raspberrys, conociendo las posiciones de estas últimas se podrá calcular la posición mediante el mismo sistema aplicado en el caso anterior y explicado en el punto 5.1 de este documento.

El principal problema de esta implementación es la sincronización de los diferentes valores obtenidos de diferentes Raspberrys a lo largo del tiempo. En el anterior caso al obtener todos los datos desde el mismo dispositivo no nos encontrábamos con este inconveniente.

Por otro lado, cabe destacar que para detectar dispositivos móviles asociados a una persona, se podría reutilizar el mismo sistema implementado en el caso anterior, siempre que la aplicación fuese ejecutada en segundo plano. Cada vez que se calcula una localización es guardada en base de datos, por lo que la información se seguiría obteniendo de igual manera.

La implementación en este caso no ha sido llevada a cabo por falta de tiempo. Para su desarrollo se continuaría con los elementos desarrollados aplicándoles una serie de ampliaciones para poder realizar las acciones necesarias.

7. Conclusión

El objetivo de este proyecto es el desarrollo de un sistema que permita conocer y almacenar la posición de diferentes activos en un área determinada dentro de un edificio. Este proceso cubre desde la investigación de diferentes tecnologías hasta su posterior implantación y prueba sobre el espacio.

Para obtener una conclusión sobre el cumplimiento de los objetivos del proyecto deberemos tener en cuenta una serie de informaciones obtenidas durante su desarrollo:

- El posicionamiento en el espacio mediante la utilización de beacons tiene un alto grado de variación en el cálculo de distancias, lo que producirá un desajuste entre la posición real y la calculada.
- La implantación de diferentes ajustes para aumentar la precisión de este sistema hará que el sistema sea más fiable al permanecer en un punto a lo largo del tiempo y tenga más problemas si la funcionalidad que se busca es representar el movimiento del activo en tiempo real.
- El sistema será eficiente si el objetivo final es conocer el número de activos existentes en un área sin importar la posición precisa que ocupen en el espacio, gestionando estos datos para diferentes utilidades como se comentó al inicio de este documento.
- A pesar de realizar diferentes acciones para otorgar mayor precisión al sistema, no se conseguirá una funcionalidad similar a la obtenida en sistemas de posicionamiento en exteriores como puede ser el GPS.

Por lo tanto, se concluye que la funcionalidad óptima alcanzada tras la realización de este proyecto es el conocimiento del número de activos en un área limitada para un posterior uso de estos datos y no la localización a tiempo real de estos activos, al carecer de la precisión necesaria que otorgaría al sistema una utilidad real.

Como trabajo futuro, podría resaltarse que recientemente ha sido presentada la nueva versión de Bluetooth, Bluetooth 5.1, cuya principal mejora será la precisión a la hora de calcular distancias entre objetivos, lo cual puede mejorar las prestaciones del sistema desarrollado. También existe la posibilidad de desarrollar nuestros propios beacons con la utilización de los chips ESP32, los cuales cuentan tanto con tecnología Bluetooth como con tecnología wifi.

8.Bibliografía

[1] Blatem, "¿Qué es un smart building y cómo gestiona la eficiencia energética?", http://www.blatem.com/es/actualidad/noticias/que-es-un-smart-building-y-como-gestiona-la-eficiencia-energetica (último acceso: 13/02/2019)

[2] Android Studio, https://developer.android.com/studio/?gclid=Cj0KCQiAnY_jBRDdARIsAIEqpJ2fiyb_aKQH3kCQUBG5feqZ4LX7EbK-D8VbqirExU4qkUL5an3ahEoaAmGyEALw_wcB (último acceso: 13/02/2019)

[3] Francisco J. Gil Gala, "Eclipse, qué es y cómo se instala", https://rootear.com/desarrollo/eclipse (último acceso: 13/02/2019)

[4] CMDBuild, http://www.cmdbuild.org/en (último acceso: 13/02/2019)

[5] DBeaver Community, "Universal Database Tool", https://dbeaver.io/ (último acceso: 13/02/2019)

[6] PostgreSQL, "The World´s Most Advanced Open Source Relational Database" https://www.postgresql.org/ (último acceso: 13/02/2019)

[7] Esri España, "Descubre los GIS", https://www.esri.es/descubre-los-gis/ (último acceso: 13/02/2019)

[8] Paradigma digital, "Postman:gestiona y construye tus APIs rápidamente", https://www.paradigmadigital.com/dev/postman-gestiona-construye-tus-apis-rapidamente/ (último acceso: 13/02/2019)

[9] Wireshark, https://www.wireshark.org/ (último acceso: 13/02/2019)

[10] Hibernate, "Everything data", http://hibernate.org/ (último acceso: 13/02/2019)

[11] Carlos Cabello, "9 usos reales para comprender qué son los "beacons"", https://www.nobbot.com/redes/9-usos-reales-comprender-los-beacons/ (último acceso: 13/02/2019)

[12] Beaconer, "Ventajas y desventajas de la tecnología Beacon", http://www.usingbeacons.com/ventajas-y-desventajas-de-la-tecnologia-beacon/ (último acceso: 13/02/2019)

[13] Grupo Fractalia, "Beacons, modo de funcionamiento, ventajas y potenciales aplicaciones", https://www.fractaliasystems.com/beacons-modo-de-funcionamiento-ventajas-y-potenciales-aplicaciones (último acceso: 13/02/2019)

[14] Javier Penalva, "NFC: qué es y para qué sirve", https://www.xataka.com/moviles/nfc-que-es-y-para-que-sirve (último acceso: 13/02/2019)

[15] Dipole RFID, "Tecnología RFID", http://www.dipolerfid.es/es/tecnologia-RFID (último acceso: 13/02/2019)

[16] UDLAP, "Capítulo 3. Principios de la tecnología RFID", https://docplayer.es/2773682-Capitulo-3-principios-de-la-tecnologia-rfid.html (último acceso: 13/02/2019)

[17] Carvalza, “Qué es un GPS (Sistema de Posicionamiento Global)”, https://www.carvalza.es/que-es-un-gps (último acceso: 13/02/2019)

[18] Agnieszka Gąsiorek, “iBeacon and Eddystone – The (Small) Defference Between Them”, https://kontakt.io/blog/ibeacon-vs-eddystone/ (último acceso: 13/02/2019)

[19] Alfredo Pascual, “Dos millones de razones para saber qué es exactamente Raspberry Pi”, https://www.elconfidencial.com/tecnologia/2013-11-22/dos-millones-de-razones-para-saber-que-es-exactamente-raspberry-pi_56003/(último acceso: 13/02/2019)

[20] Gabri, “¿Cómo funcionan los dispositivos GPS? Trilateración vs Triangulación”, https://acolita.com/como-funcionan-los-dispositivos-gps-trilateracion-vs-triangulacion/(último acceso: 13/02/2019)

[21] Félix Barba Barba, “Estudio de algoritmos de localización en interiores, para tecnologías móviles de última generación”, Trabajo Fin de Master, Universidad Politécnica de Madrid, http://www.dit.upm.es/~posgrado/doc/TFM/TFMs2011-2012/TFM_Felix_Barba_2012.pdf(último acceso: 13/02/2019)

[22] Miguel Ángel Álvarez, “Qué es MCV” https://desarrolloweb.com/articulos/que-es-mvc.html(último acceso: 13/02/2019)

[23] Juan Díaz Almagro, “Potencia, eje radical y sección áurea”, https://es.slideshare.net/jdalmagro/potencia-eje-radical-y-seccin-urea(último acceso: 13/02/2019)