_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE UPV INGENIEROS DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

POSICIONAMIENTO Y NAVEGACIÓN INDOOR: IMPLEMENTACIÓN MEDIANTE SISTEMA INERCIAL.

Autor: Otman Maamori

Tutor: Palau Salvador, Carlos Enrique

Trabajo Fin de Grado presentado en la Escuela Técnica Superior de Ingenieros de Telecomunicación de la Universitat Politècnica de València, para la obtención del Título de Graduado en Ingeniería de Tecnologías y Servicios de Telecomunicación

Curso 2015-2016

Valencia, 13 de septiembre de 2016

Resumen

Este documento es la memoria de un trabajo fin de grado, que tiene como objetivo el desarrollo de una aplicación en Android para el posicionamiento de personas en zonas cubiertas y edificios.

La aplicación desarrollada calcula la posición de la persona mediante los sensores que disponen hoy en día casi todos los dispositivos inteligentes ya sean estos teléfonos móviles o tablets. Los sensores específicamente utilizados en este trabajo son el acelerómetro para la determinación del movimiento de la persona y el giroscopio y el magnetómetro para la determinación de la dirección de movimiento de dicha persona.

Esta aplicación forma parte de un proyecto de investigación de posicionamiento indoor realizado por la ETSIT de la UPV. El objetivo que alcanza esta aplicación en dicho proyecto de investigación es el de afinamiento de su precisión.

Esta memoria se ha estructurado en siete capítulos, a continuación se hará una breve descripción de cada uno.

El primero capítulo trata de describir el contexto del proyecto, sus posibles aplicaciones en nuestra sociedad, así como los objetivos que desea cubrir.

El segundo capítulo trata los diferentes algoritmos, técnicas de estimación y tecnologías que hoy en día se utilizan o son investigados para el posicionamiento en interiores.

El tercer, cuarto y quinto capítulo tratan la tecnología escogida para la implementación de la aplicación, las herramientas utilizadas, lenguajes de programación, la arquitectura interna e implementación de la aplicación y finalmente las pruebas realizadas y resultados obtenidos para la evaluación de la aplicación.

El sexto y séptimo capítulo respectivamente se concluye el proyecto y se cita la bibliografía consultada para el desarrollo de la aplicación.

Resum

Aquest document és la memòria d'un treball fi de grau, que té com a objectiu el desenvolupament d'una aplicació en Android per al posicionament de persones en zones cobertes i edificis.

L'aplicació desenvolupada calcula la posició de la persona mitjançant els sensors que disposen avui dia quasi tots els dispositius intel·ligents, ja siguen aquests telèfons mòbils o tablets.

Els sensors específicament utilitzats en aquest treball són l'acceleròmetre per a la determinació del moviment de la persona i el giroscopi i el magnetòmetre per a la determinació de la direcció de moviment d'aquesta persona.

Aquesta aplicació forma part d'un projecte de recerca de posicionament indoor realitzat per la ETSIT de la UPV. L'objectiu que aconsegueix aquesta aplicació en aquest projecte de recerca és el d'afinament de la seua precisió.

Aquesta memòria s'ha estructurat en set capítols, a continuació es farà una breu descripció de cadascun.

El primer capítol tracta de descriure el context del projecte, les seues possibles aplicacions en la nostra societat, així com els objectius que desitja cobrir.

El segon capítol tracta els diferents algorismes, tècniques d'estimació i tecnologies que avui dia s'utilitzen o són investigades per al posicionament en interiors.

El tercer, quart i cinquè capítols tracten la tecnologia escollida per a la implementació de l'aplicació, les eines utilitzades, llenguatges de programació, l'arquitectura interna i implementació de l'aplicació i finalment les proves realitzades i resultats obtinguts per a l'avaluació de l'aplicació.

En el sisè i setè capítols respectivament es conclou el projecte i se cita la bibliografia consultada per al desenvolupament de l'aplicació.

Abstract

This document is a final year dissertation that aims to develop an Android application to pinpoint people in covered areas and buildings The application calculates the people's position through the use of sensors that almost all smartphones have nowadays, whether they be mobile phones or tablets.

The sensors that are specifically used in this project are the accelerometer to determine the movement of a person, the gyroscope, and the magnetometer to determine the direction of movement of said person.

This application is a part of the indoor positioning research project conducted by the ETSIT from the UPV. The objective of this application in said project is position alignment

This document is structured in 7 chapters. The following paragraphs include a brief description of each one .

The first chapter revolves around describing the context of the project, its possible applications in our society, as well as the objectives it aims to accomplish. The second chapter is about the different algorithms, estimating techniques, and technologies for positioning. The third, fourth, and fifth chapters discuss the technology chosen for the implementation of the application, the tools used, programming languages, internal design, and the implementation of the application, and finally the tests conducted and results obtained. The sixth and seventh chapters respectively conclude the project and provide the bibliography for the sources used.

Índice

Índice de figuras

Índice de tablas

Capítulo 1. Introducción

1.1 Contexto

Desde hace tiempo surgió la necesidad de localizar personas y objetos en interiores de edificios y entornos cerrados, con una variedad de finalidades, que pueden ser desde un simple entretenimiento hasta casos de extrema emergencia. En el caso de entretenimiento, podría ser la posibilidad de ofrecer una información determinada a un visitante de un museo en función de la posición en la que se encuentre, mientras que en el caso de emergencia, podría ser la localización de los miembros de un grupo de bomberos en plena actuación.

La sociedad donde vivemos hoy en día, el entorno donde nos movemos y Nuestro forma de vida son todos ellos factores que exigen el desarrollo de sistemas de localización en interiores(localización indoor) para un alto nivel de desarrollo, una eficaz productividad y alta comodidad.

Las tecnologías existentes hoy en día para la localización en espacio exterior como por ejemplo GPS, GALILEO y GLONASS, no son eficaces para interiores por la pérdida exponencial que experimenta la señal al atravesar los edificios. Ello obliga a desarrollar sistemas específicos para la localización indoor.

Hoy en día todos los edificios, campos y zonas cubiertas disponen de tecnologías de radio frecuencia (RF), como por ejemplo, Wlan conocida como Wi-Fi, red de telefonía móvil como por ejemplo la Femtocelda , como sensores, RFID etc. Así también toda persona dispone de aparatos inteligentes tales como móviles, tabletas etc, que incorporan una variedad de sensores tales como acelerómetro, giroscopio etc ; así como tarjeta de red Wi-Fi. Hoy en día se aprovechan estas tecnologías e infraestructuras ya disponibles y desplegadas para desarrollar estos sistemas y de esta forma se consiguen sistemas de localización con coste menor, sin embargo este bajo coste tiene un coste a nivel de precisión lo que hace que el número de aplicaciones que se le puede dar a estos sistemas sea limitado.

Hoy en día no existe ningún estándar o tecnología de referencia para implementar estos sistemas, de hecho, todos los sistemas están en fase de experimento y/o investigación. Sin embargo los algoritmos que más consenso tienen y que se implementan en la mayoría de estos sistemas es el denominado fingerprinting, este último se basa su funcionamiento en deducir la posición del terminal (objeto) comparando las medidas obtenidas en un momento dado, con otras medidas previas de referencia que han sido guardas y que constituyen la huella radioeléctrica del lugar.

1.2 Objetivos

En este proyecto, se diseña un subsistema que determine la posición de individuos en entornos cerrados, tales como edificios, aeropuertos etc; utilizando los sensores de los

dispositivos inteligentes que disponemos hoy en día, tales sensores pueden ser acelerómetro, giroscopio etc.

Este subsistema forma parte de otro proyecto de investigación que lleva la escuela técnica superior de ingeniero de telecomunicaciones de la universidad politécnica de valencia. El conjunto de ambos brinda un sistema de posicionamiento en interiores empleando sistemas inerciales y tecnología de radio frecuencia, concretamente Wi-Fi.

El subsistema a desarrollar es prácticamente una aplicación Android ejecutada sobre móviles y tablets, utilizando los sensores acelerómetro, giroscopio y magnetómetro, para determinar la posición del individuo.

A continuación se listan los objetivos principales del proyecto:

- Descripción de las principales tecnologías utilizadas o en proceso de investigación para el posicionamiento en interiores, describiendo más la tecnología Wi-Fi, por ser esta la utilizada por el proyecto de investigación.
- Desarrollo de una aplicación Android, que determina la posición de individuos en entornos cerrados mediante los sensores disponibles hoy en día en terminales inteligentes.
- La aplicación desarrollada puede mejorar y refinar la precisión del proyecto de investigación, para poder así brindar un sistema final de alta precisión.

El proyecto empezará con una descripción de las diferentes tecnologías que se están utilizando o investigando hoy en día para este tipo de posicionamiento tanto de radio frecuencia como inerciales, haciendo unas comparaciones entre ellas, y citando para cada una sus ventajas e inconvenientes. En segundo lugar se justificará la solución escogida por este trabajo y en último lugar se explicará su implementación, tecnología utilizada, algoritmos y función dentro del proyecto de investigación.

Capítulo 2. Estado del Arte

En este punto del trabajo se hablará de las diferentes técnicas de estimación de posición así como las tecnologías, tanto las que se utilizan para el posicionamiento indoor como futuras líneas de investigación.

Estas tecnologías se suelen implementar en combinación para formar así sistemas híbridos que puedan brindar posición con alta precisión. Todas las tecnologías a continuación descritas no se implementan como una única solución sino que se combinan varias de las mismas para obtener un alto nivel de precisión y así aumentar el número de aplicaciones que se le puede dar al sistema.

2.1 Técnicas de estimación de posición

En un edifico o entorno indoor se localizan una serie de obstáculos que hacen que la determinación de la posición sea una tarea difícil de hacer.

La propagación de señales de radio frecuencia en entornos cerrados, tales como edificios, experimentan una alta atenuación debido a la estructura interna de los edificios, por la existencia de obstáculos ya sean estos muebles, personal, paredes etc.

Estos obstáculos hacen que la señal experimente difracción lo que provoca que la señal se disperse y se divida en señales con potencia más pequeña en varias dirección provocando así y también sin dispersión reflexiones de señales que el receptor puede recibir con diferentes fases provocando así ltas atenuaciones, esto también hace que haya multi-trayecto entre el transmisor y receptor. Por todo ello podemos concluir que el entono es mucho más influyente que la distancia entre antenas, hecho que afecta más en espacio exterior.

La atenuación debido a lo antes comentado o nula presencia de señal en algunas zonas del edificio hacen complicada la tarea del sistema, provocando así que este último presente una precisión muy baja. Sin embargo en el espacio exterior la señal se ve menos afectada que en interiores, considerando en muchos casos perdidas solo de espacio libre y algunas reflexiones que puede experimentar la señal.

La expresión matemática que permite calcular la potencia recibida en un punto en el espacio libre es la siguiente [1]:

$$Pr = \frac{Pt\ Gr\ Gt}{\left(\frac{4\pi R}{\lambda}\right)^2}$$

Ecuación 2.1 Ecuación de transmisión

Dónde:

Pr: Potencia recibida.
Pt: potencia transmitida.

Gr y Gt: Ganancia de la antena receptora y transmisora respectivamente.

R: Distancia entre transmisor y receptor.
De la formula podemos deducir que las pérdidas por espacio libre son:

$$(4\pi R)^2$$

Ecuación 2.2: Pérdidas en aire libre.

Tanto en interiores como en exteriores existen dos tipos de canales de radio:

Line-of-Sight (LOS) [2]**:** Es decir, visión directa entre el emisor y el receptor, no hay obstáculos físicos. Ver fórmula 1.

Non Line of Sigh (NLOS) [2]: Se usa para describir un trayecto parcialmente obstruido entre la ubicación del transmisor de la señal y la ubicación del receptor de la misma. Los obstáculos incluyen paredes, muebles, personal etc.

Por lo que para este último tipo de canal de radio se desarrollan modelos de propagación que permiten estimar las pérdidas de la señal electromagnética que se propaga en el entorno. Como los entornos son variantes debido por ejemplo al material utilizado, los muebles etc; no existe un modelo de propagación que brinde exactamente las perdidas en diferentes puntos del entono.

A continuación se citan algunos modelos:

- **Modelos Empíricos** [3]**:** Son aquellos basados en medidas realizadas sobre un entono en particular. Se deben tomar varias medidas para obtener resultados acordes a la realidad.

  A su vez existen varios modelos empíricos, a continuación se citan:

  - One-Slope [2]: En este modelo se ajusta la pendiente de pérdidas con el logaritmo de la distancia. De hecho ésta pendiente será mucho mayor que la observada en espacio libre. La expresión que cuantifica las pérdidas es la siguiente.

    $$L = L_0 + 10n \log d$$

    Ecuación 2.3: Formula One-Slope.

    Donde L_0 son las perdidas en espacio libre y d la distancia al transmisor.

  - Basado en un modelo propuesto por Motley-Keenam, intenta ser mucho más realista que el anterior ya que tiene en cuenta las pérdidas adicionales que se generan por el paso de la señal a través de distintas paredes y suelos.

  - Linear-slope Model(LSM) [2]: Este es el modelo más sencillo de los presentados en el COST-231, ya que sólo presenta un parámetro de ajuste. Este modelo asume que la dependencia de las pérdidas de propagación con la distancia es de tipo lineal.

    $$L = L_{FS} + \alpha d$$

    Ecuación 2.4: Linear-slope Model.

- **Medelos semi-deterministas** [3]**:** Se toman en cuenta medidas realizadas, pero éstas luego se ajustan a cierto modelo establecido teóricamente. Los modelos se infieren tomando en cuenta características del terreno, altura de antenas, etc.

  Se consideran distintos escenarios, pero se realiza la predicción entorno a uno de éstos, luego, en los otros escenarios se consideran factores de ajuste (factor de ajuste de altura, altura de obstáculos y densidad de los mismos).

Como se observa, el cálculo de las pérdidas en un entorno Indoor no es en absoluto trivial, por lo que se hace necesaria la experimentación con usuarios finales. Para disminuir los errores de ubicación debidos a estos efectos, se utiliza diversos algoritmos de localización como la trilateración, la triangulación y análisis de escenario. En dichos estudios se habla de los algoritmos tales como el TOA, TDOA, AOA, RTOF, RSS, o Fingerprinting entre otros.

2.1.1 Triangulación

La triangulación es un método que se utiliza para estimar la ubicación de un objeto o usuario final en base a mediciones obtenidas de tres estaciones ubicadas en sitios conocidos, por lo que se puede encontrar la ubicación del objeto y por tanto de la persona que lo lleva mediante la intersección de tres esferas. Se trata de un método de cálculo de las coordenadas de un punto destino sobre la base de las distancias medidas entre el destino y los puntos de referencia fijos de los que se conoce la ubicación. La triangulación cuenta con dos tipos de técnicas: Lateración y Angulación [4].

2.1.1.1 Técnicas de Lateración

Son aquellas que estiman la posición de un objeto (sujeto) midiendo sus distancias desde múltiples puntos de referencia. Estas técnicas utilizan parámetros como el tiempo de llegada (TOA), la diferencia del tiempo entre llegadas (TDOA29), el basado en atenuación de la señal (RSS), el Tiempo de ida y vuelta (RTOF) o el método de fase de llegada (POA32).

- Time Of Arrival (TOA) [5]: Tiene en cuenta que la distancia desde el dispositivo móvil a la unidad de medición es directamente proporcional al tiempo de propagación, con el fin de permitir el posicionamiento en 2D, las mediciones de TOA deben hacerse con respecto a las señales de al menos tres puntos de referencia. En los sistemas basados en TOA, se mide el tiempo de propagación de una sola vía, y se calcula la distancia entre la unidad de medición y el transmisor de señal

  Este algoritmo requiere una sincronización muy alta entre los dos extremos, ya que al contrario el cálculo presentaría un gran error. Al no haber visión directo esto puede hacer que el tiempo de propagación sea alto, lo que se transforma en mayor distancia.

- Time Difference Of Arrival (TDOA) [5]: Se calcula la diferencia en el tiempo en que la señal llega a múltiples unidades de medida, en vez de la hora de llegada

absoluta de TOA. Para estimar la ubicación de un usuario se realiza el cálculo a partir de las intersecciones, similares a las calculadas en la TOA.

Estos dos algoritmos presentan desventajas en entornos Indoor, donde es muy complicado el tener visión directa entre transmisor y receptor, y además la propagación sufre de múltiples efectos, tales como la dispersión, difracción, efecto multipath…, lo que afectaría considerablemente al tiempo y al ángulo de llegada de la señal, por lo que la precisión de la ubicación del sujeto quedaría mermada.

- Método basado en Atenuación de la Señal (RSS) [6]: Para evitar los efectos negativos de los métodos anteriormente expuestos, se propuso un enfoque basado en estimar la distancia del dispositivo móvil a los puntos de acceso, utilizando la atenuación de la intensidad de la señal emitida, de forma que se calcula la pérdida de la señal debida a la propagación.

  Sin embargo este método presenta un gran inconveniente es que la atenuación de la señal no solo depende de la distancia sino que depende de otros factores como pueden ser la reflexión, dispersión, absorción, interferencias etc.

- Reflectron Time Of Flight (RTOF) [6]: Este método consiste en medir el tiempo que tarda una señal en viajar desde el transmisor al receptor móvil y en volver al origen, funcionando de forma similar al sistema radar de telemetría. El funcionamiento es similar al de TOA pero en este caso la sincronización de reloj no es tan crítica. Sin embargo se necesita un reloj con un nivel muy alto de precisión.

2.1.1.2 Técnicas de estimación de ángulo de llegada(AOA)

La angulación es un método por el cual se localiza un objeto mediante el cálculo de los ángulos relativos en relación a puntos de referencia [6]. Se utilizan al menos dos puntos conocidos de coordenadas (x1, y1) y (x2, y2), que disponen de dos ángulos que se calculan con el eje de referencia del punto de referencia, con el eje que lo une con el punto en que se encuentra el dispositivo móvil que lleva el usuario u objeto a localizar.

Como principal desventaja se puede comentar que la medición de los ángulos debe ser precisa, lo que conlleva a una gran complejidad en entornos interiores debido, fundamentalmente a las pérdidas por multi-camino en dichos entornos.

2.1.2 Análisis de escena

En la mayoría de los sistemas de posicionamiento indoor se utiliza esta técnica y concretamente el Fingerprinting [7].

En redes se conoce como análisis de escena a aquellos algoritmos, que se encargan de mapear características de potencia, canal de emisión y direcciones MAC en una zona determinada (escena), para luego realizar la estimación de la ubicación de un dispositivo o sujeto, mediante la comparación de dichas medidas con las recibidas en cada momento, de forma que se asemejen a las de las zonas más cercanas.

En este método, en lugar de determinar las distancias entre el usuario y los puntos de acceso y de usar la triangulación para obtener la ubicación del usuario, dicha ubicación se determina mediante la comparación de los valores de RSSI obtenidos en la etapa de medición, habiendo ya previamente realizado un mapa de potencias y MACs de los dispositivos de acceso inalámbricos en el escenario donde se implementa el sistema.

El algoritmo más empleado de esta técnica es el Fingerprinting, a continuación se describe su funcionamiento.

El primer paso en el fingerprinting es el mapeo de la escena [8], consiste en recorrer el edificio escaneando en cada punto la red, y almacenando en una base de datos los valores de potencia, MACs etc, de la señal recibidos de cada AP accesible junto a las coordenadas a las que corresponden.

Tabla 2.1: Mapeo de la escena.

X_1	Y_1	$RSSI_{AP1,1}$	$RSSI_{AP2,1}$	…	$RSSI_{APm,1}$
…	…	…	…	…	….
Xn	Yn	$RSSI_{AP1,n}$	$RSSI_{AP2,n}$	…	$RSSI_{APm,n}$

Este algoritmo tiene una mayor precisión cuanto mayor es la densidad de puntos en los que se han tomado los valores. La precisión en metros de este algoritmo depende de la distancia entre los puntos que se han tomado durante el proceso de Fingerprinting.

Este proceso se realiza sólo una vez y sirve para todos los clientes. Sólo sería necesario repetirlo en caso de producirse cambios estructurales en el edificio, o severas modificaciones en la infraestructura de la red inalámbrica, por ejemplo, mover todos los puntos de acceso.

Con la base de conocimiento (mapeo) de Fingerprinting realizada, la primera vez que el cliente entra en el edificio y se conecta al servidor, se descarga en su dispositivo móvil los datos tomados. En el caso de que el dispositivo no tuviese la capacidad computacional necesaria para realizar estas operaciones, la descarga de la base de datos no sería necesaria y sería el servidor el encargado de realizar todos los cálculos.

El objeto a localizar (móvil principalmente) escanea la red obteniendo el vector de fuerzas de la señal de los APs accesibles en ese lugar determinado y lo envía en su caso al servidor para determinar la posición.

La principal desventaja, de las técnicas de localización basadas en tecnología Fingerprinting, es que la intensidad de la señal recibida podría verse afectada por efectos como la difracción, reflexión y dispersión Indoor, así como por los cambios que se producen en los canales y en la potencia de las señales de los puntos de acceso.

Para la comparación de los datos mapeados con los obtenidos por el dispositivo y para así determinar la posición del objeto, se utilizan una serie de algoritmos que intentan eliminar los efectos anteriormente comentados, entre estos algoritmos están los métodos determinísticos; el vecino más próximo (NN), el k-vecino más próximo (KNN) **[10]**; K-vecino más próximo ponderado (WKNN) **[10]** ; redes neuronales. Si bien hay que notar que los más utilizados son el kNN y el WkNN, aunque el kNN es un caso específico del

WkNN en el que todos los puntos de acceso tienen la misma ponderación. De igual forma, el NN es el caso de kNN pero con k=1.

La necesidad de uso de estos algoritmos deriva del hecho de que los valores medidos durante la fase online (fase de medida del dispositivo) no suelen tener un emparejamiento sencillo con los valores almacenados en la base de datos.

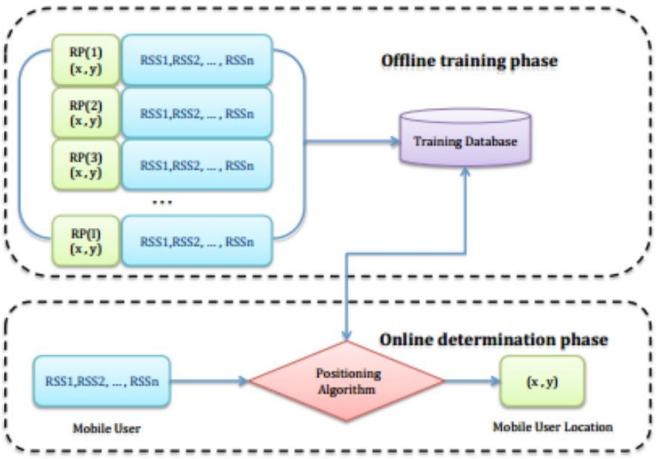

Figura 2.1: Diagrama de método de Fingerprinting.

Offline training phase corresponde a la fase de mapeo del escenario.

Online determination phase corresponde a la fase de navegación.

2.1.3 Algoritmos de proximidad

Hay diversos estudios en los que se habla de otros tipos de algoritmos, que proporcionan información de la ubicación basándose en la proximidad **[9]** de los dispositivos respecto de otros de los que se conoce su ubicación. Estos sistemas tienen la ventaja de la sencillez de implantación pero la desventaja de que resultan caros en comparación con otros sistemas como los Wi-Fi de amplia implantación en redes públicas y corporativas. En estos sistemas se fundamentan los basados en Bluetooth, IrDA y los RFID.

2.2 Sistemas de localizaciones en interiores

En este punto se van a explicar las diferentes tecnologías de radio frecuencias como los sistemas inerciales que se están utilizando para sistemas de posicionamiento indoor, citando para cada una sus ventajas y desventajas.

2.2.1 Sistemas basado en Bluetooth LE

Bluetooth Low Energy (BLE) es la nueva especificación 4.0 **[11]**, 4.1 y 4.2 de la tecnología Bluetooth desarrollado por Bluetooth Special Interest Group (SIG). Se ha diseñado como una tecnología complementaria a Bluetooth clásico para garantizar un consumo de energía bajo, y menor tiempo de establecimiento de conexión. A pesar del uso de la misma banda de frecuencia y las similitudes compartidas, BLE debe

considerarse un nuevo estándar con objetivos y aplicaciones diferentes. Con esta nueva tecnología surgen muchas aplicaciones útiles y en el caso de este proyecto, se explica su aplicación en el mundo de posicionamiento indoor.

Tabla 2.2: Comparativa entre BLE y Bluetooth clásico ***[11].***

	BLE	BR	EDR
Modulación	GFSK 0.45 to 0.55	GFSK 0.28 to 0.35	DQPSK / 8DSPK
Tasa Mbit/s	1	1	2/3
Nº Canales	40	79	79

Arquitectura [12]

La topología de red de BLE es de tipo estrella. Los dispositivos Master pueden tener varias conexiones de capa de enlace con periféricos (Slaves) y simultáneamente realizar búsquedas de otros dispositivos. Por otro lado un dispositivo en rol de esclavo solo puede tener una conexión de capa de enlace con un único Master. Además, un dispositivo puede enviar datos en modo Broadcast, eventos de Advertising, sin esperar ninguna conexión; esto permite enviar datos a los dispositivos en estado Scanning sin necesidad de establecer la conexión Master-Slave.

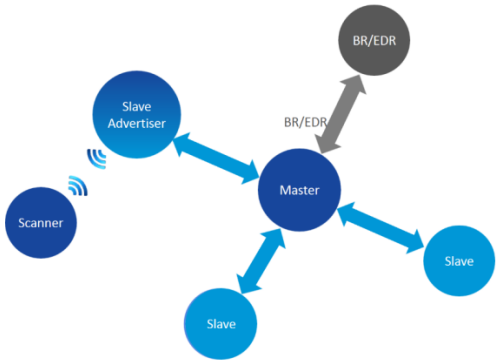

Figura 2.2: Topología estrella de la tecnología BLE.

El BLE funciona bajo una pila de protocolos que se divide en tres partes básicas: Controller, Host y Applications **[11][12]**.

El Controller **[11][12]** es el dispositivo físico que permite transmitir y recibir señales radio e interpretarlas como paquetes con información. Contiene la capa física "Physical Layer", "Direct Test Mode", la capa de enlace "Link Layer" y la interfaz de control de host "Host Controller Interface".

El Host **[11][12]** es la pila de software que administra cómo dos o más dispositivos se comunican entre ellos. Esta parte de la pila contiene una capa de control de enlace lógico, protocolo de adaptación, administrador de seguridad, protocolo de atributo, perfil de atributo genérico (GATT) y perfil de acceso genérico (GAP).

La aplicación es la funcionalidad que se implementa con el BLE, siendo cada sistema operativo quien proporciona el API para implementar dichas aplicaciones.

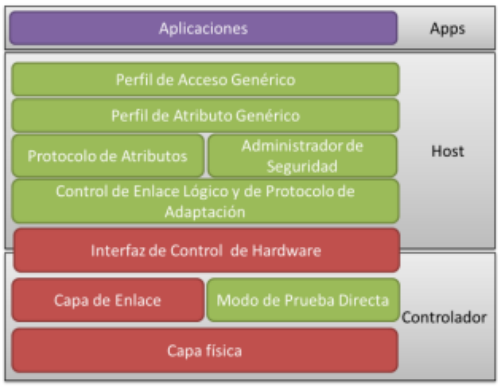

Figura 2.3: Pila de protocolos BLE.

A continuación se va a hacer una descripción básica de alguna de las capas que forman la pila de protocolos.

Capa física [11]

Esta capa se encarga de enviar y recibir las señal de radio, utilizando la banda de frecuencia Industrial Scientific Medical (ISM) 2.4 Ghz que se extiende desde 2402 Mhz hasta 2480 Mh. El número de canales es de 40, tres de los cuales se utilizan para el Advertising y el resto para la transmisión de datos.

La banda de funcionamiento de BLE es compartida por varias tecnologías como IEEE 802 y ZigBee. Esto provoca la aparición de interferencia, para solucionar este problema, BLE utiliza la técnica espectro ensanchado por salto de frecuencia (FHSS), es una técnica de modulación en el que la señal se emite sobre una serie de radiofrecuencias aparentemente aleatorias, saltando de una frecuencia en frecuencia sincrónicamente con el transmisor.

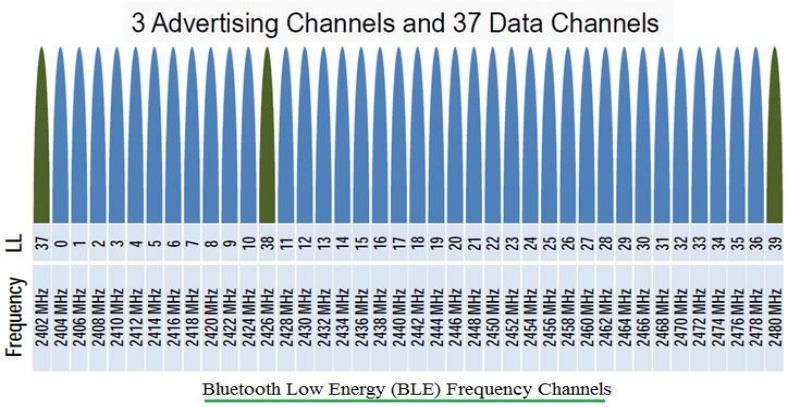

Figura 2.4: Canales BLE

Capa de Enlace [11]

Esta es la capa responsable de los estados de Advertising, Scanning, creación y mantenimiento de las conexiones. También es la responsable de la estructura de los

paquetes. Los estados de la tecnología BLE son, Standby, Advertising, Scanning y Initiating **[11][12]**.

EL BLE utiliza dos métodos para el envío de información, a continuación se describen:

- Advertising y Scan Responses **[11][12]**: Los paquetes de Advertising incluyen campos donde se puede proporcionar información, y el dispositivo central puede solicitar más información con un Scan Request. No hay seguridad esencial para este método ya que en modo Broadcasting la emisión de datos es pública, y la pueden recibir todos los dispositivos.
- Conexiones **[11][12]**: Durante una conexión, dos dispositivos pueden intercambiar información, lo cual requiere procedimientos más complejos comparados con otros métodos.

Después de una pequeña introducción sobre el funcionamiento y las características de la tecnología BLE, procederemos ahora a la explicar su implementación en el mundo de posicionamiento indoor.

Beacons [13]

Son dispositivos que implementan la tecnología Bluetooth LE para la localización indoor.

Son bastante parecidos a un GPS en cuanto al funcionamiento, ya que los beacons cuentan con una señal única por cada dispositivo que son capaces de definir una localización y detectar otros dispositivos con Bluetooth de la misma tecnología como pueden ser los móviles inteligentes. Estos dispositivos al implementar la tecnología LE hace que la batería les dure meses.

Posicionamiento y distancia con iBeacon

El iBeacon **[11][12]** es el nombre del sistema utilizado hoy en día para El posicionamiento en interiores con tecnología BLE. El posicionamiento con el sistema iBeacon se lleva a cabo por la técnica de proximidad. Esta técnica ya explicada en punto anterior, se calcula la distancia a un Beacon mediante la medición de la potencia de la señal recibida (RSS, del inglés Received Signal Strength) [14].

El Beacon transmite de forma aleatoria entre los canales de advertismenet **[12]** paquetes de advertisement de forma repetitiva según el periodo elegido. Estos paquetes de advertisement transportan un identificador numérico, que las aplicaciones instaladas en los receptores mapean en acciones. Como se puede ver el protocolo es muy sencillo, sin embargo la complejidad de este sistema de posicionamiento reside en las aplicaciones.

El intervalo o el periodo de transmisión de los advertising packet es un valor fijo de unos 100 ms, aunque este periodo puede ser reconfigurado. Sin embargo este parámetro tiene una relación directa con la duración de la batería, ya que a menor periodo más consumación de batería.

El beacon puede ser configurado **[11]** para cambiar parámetros de la comunicación como puede ser el periodo de envío etc.

Esta tecnología está destinada más bien a la ubicación descontinua ya que se basa en la técnica de proximidad por lo que su aplicación como sistema de seguimiento sería falto de eficacia y precisión. Sin embargo tiene una variedad de aplicaciones que no necesitan localización sino proximidad, como por ejemplo estar en un centro comercial y al acercar a una determinada tienda salte una aplicación en el dispositivo receptor que nos avise de los descuentos ofrecidos por dicha tienda, y muchas aplicaciones más. Podemos concluir que esta tecnología nos puede transmitir que es lo que nos rodea en un determinado momento.

A continuación describiremos el formato del llamado Advertising Packet **[14]**.

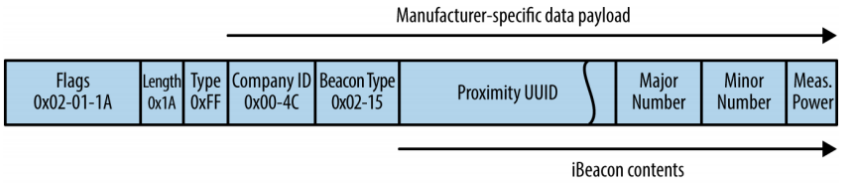

Figura 2.5: Adevertising Packet.

El Advertisign Packet siempre tiene el mismo tamaño y está compuesto por una serie de campos fijos Flags indica si Beacon puede funcionar en modo BLE o modo clásico o ambos, Length informa del tamaño de del resto de campos que le siguen, Type: Identifica el fabricante, Company ID identifica el fabricante mediante un identificador.

Ahora procederemos a explicar la parte del formato denominada Beacon Payload, que es interpretada por el receptor para generar eventos.

Los campos que la componen son cinco y a continuación se describen.

- Beacon type: Su valor es 0x02-15. Informa del protocolo utilizado, siendo el número de protocolos posibles dos e informa también del tamaño de la carga Beacon.
- Proximity UUID: UUID son siglas de Universal Unique Identifier,es un identificador de 128 bits que identifica de forma única la organización que hace uso de este Beacon. Este valor es configurable por el usuario.
- Major and Minor numbers: El primero es un identificador que identifica un subgrupo de una organización mientras que el segundo identifica al mismo dispositivo Beacon.
- Measured power: Indica la potencia que se debe recibir en caso LOS a un metro de distancia. A partir de este punto de referencia se intenta calcular la distancia a la que está un dispositivo de un Beacon.

Como se explicó anteriormente el sistema iBeacon se define como sistema de posicionamiento por proximidad. Aunque se puede calcular la distancia a un Beacon mediante la medición de la potencia de la señal recibida (RSS), esto permite mejorar la posición del objeto, como se ha dicho antes este tecnología solo se utilizaría para posición y no para la navegación.

La estimación de la distancia proporcionada por una baliza se basa en la relación de la potencia de la señal iBeacon recibid (RSS) sobre la potencia calibrada del transmisor

(measured power). Esta relación queda definida por el RSSI (Received Signal Strength Indicator) que toma la siguiente expresión.

$$RSSI(dBm) = -(10n)\log(d) - A$$

*Ecuación 2.5: Cálculo de distancia en Bluetooth **[15]**.*

- n = constante de pérdidas por propagación. Toma valores entre 2.7 y 4. Para propagación en el vacío, n=2.
- d = distancia entre emisor y receptor en metros.
- A = Tx power medida en dBm.

Despejando la expresión anterior, se obtiene la distancia relativa entre emisor y receptor. Por tanto, la precisión en la posición dependerá de la correcta calibración previa del Beacon para tener un ajustado valor de measured power, la calidad de medición del hardware receptor (RSSI) y la correcta ejecución de los cálculos, que son llevados a cabo por aplicaciones.

Hay un factor que ha de tenerse en cuenta a la hora de calcular la distancia entre emisor y receptor y que afectará de forma notable: la fluctuación de RSSI. Esta fluctuación puede producirse por diferentes motivos:

- Reflexión: Es el cambio de dirección de una onda, que al entrar en contacto con un obstáculo grande, como una pared, regresa al punto donde se originó.
- Refracción: En este caso el cambio de dirección de una onda se produce como consecuencia de la diferente velocidad de propagación.
- Difracción.: Cuando una onda llega a un obstáculo de dimensión similar a la longitud de la onda, dicho obstáculo se convierte en un nuevo foco emisor de la onda.
- Dispersión (Scattering): Se produce una separación de las ondas de distinta frecuencia al atravesar un material.
- Multitrayecto: Es un fenómeno consistente en la propagación de una onda por varios caminos diferentes. Esto es debido a los fenómenos de reflexión y de difracción.

Los anteriores factores no se tienen en cuenta en los cálculos de distancia con la expresión anterior, por lo que se ven directamente reflejados en el valor d.

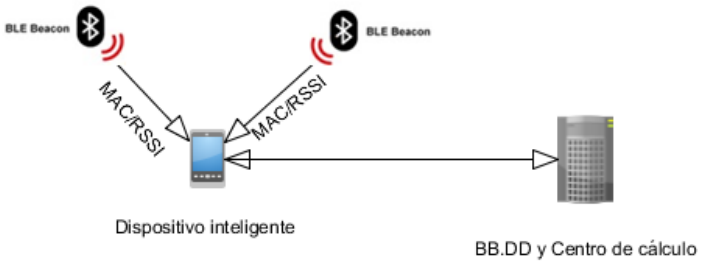

Figura 2.6: Escena tecnología iBeacon.

2.2.2 Sistemas basados en Wi-Fi

En este punto se explicará el uso de la tecnología Wi-Fi para el posicionamiento indoor. Para ello primero se realizara una explicación básica de la tecnología, funcionamiento, protocolos, estándares y un ejemplo de una infraestructura básica.

2.2.2.1 Introducción

Wlan es un sistema de comunicaciones de datos inalámbricos, en lugar del par trenzado, coaxial o fibra óptica utilizados en las LAN convencionales, proporcionando conectividad inalámbrica dentro de un área de cobertura.

La marca Wi-Fi es el resultado de la unión de grandes empresas del sector de las comunicaciones **[16]** y la electrónica para desarrollar un estándar para qué equipos de varios fabricantes dotados de la tecnología Wlan podrán comunicarse.

El primer estándar en el que se basa esta marca es el IEEE 802.11, que define el uso de los dos niveles inferiores de la arquitectura o modelo OSI (capa física y capa de enlace de datos) especificando sus normas de funcionamiento en una red de área local inalámbrica (WLAN).

La expansión y existencia de esta marca en cualquier dispositivo hizo que los estándares Wlan se conociesen por su nombre.

2.2.2.2 Arquitectura

Dispositivos

Los elementos que forman una red Wi-Fi son los siguientes.

- **Adaptadores inalámbricos**: Son tarjetas de red que cumplen con el estándar 802.11 y que permiten a un equipo conectarse a una red inalámbrica..
- **Puntos de acceso**: Permiten a los equipos equipados con tarjetas de red Wi-Fi acceder a una red. El punto de acceso es la unión entre la red y el usuario.

Figura 2.7: Punto de Acceso Cisco.

Figura 2.8: Adaptador de red inalámbrico.

Topología

Existen dos tipos de topología, modo independiente y modo infraestructura, en este proyecto solo se describe el segundo.

- **Modo infraestructura [17]:** Este segmento se forma al menos con un Punto de Acceso (AP). Al segmento básico de red en IEEE 802.11 se le denomina BSS (Basic Service Set).

  Los nodos que pertenecen al mismo BSS pertenecen al mismo medio físico por lo que solo uno puede transmitir en un determinado momento, la transmisión simultánea de dos o más nodos producirá colisión. Cualquier comunicación se hace a través del AP.

  El BSS se puede conectar a un sistema de distribución para así unirse a otros BSSs formando así una arquitectura ESS (Extended Services Set).

  En la figura 2.9, se muestra una arquitectura BSS.

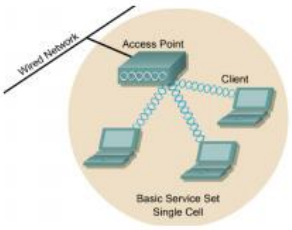

Figura 2.9: Arquitectura BSS.

  Un BSS tiene un identificativo denominado Bsic Service Set ID (BSSID), que se corresponde con la MAC inalámbrica del AP. Además dispone otro identificativo que permite que las personas identifiquen a la red Wlan de la BSS, dicho identificativo denominado SSID (Service Set ID) está formado por una cadena alfanumérica.

  Como mencionamos anteriormente para la ampliación de la red Wlan se suele agrupar varias BSSs, dando lugar a la arquitectura ESS.

- **Modo ad hoc [17]:** Los equipos inalámbricos se conectan entre sí para formar una red punto a punto, es decir, una red en la que cada equipo actúa como cliente y como punto de acceso simultáneamente.

2.2.2.3 Modelo de Referencia.

Como todos los estándares 802 del IEEE, el IEEE 802.11 cubre las primeras dos capas del modelo de OSI (Open Systems Interconnection), es decir la capa física (L1) y la capa de enlace (L2). La sección siguiente describirá lo que implica cada una de esas capas en términos de estándares inalámbricos **[17].**

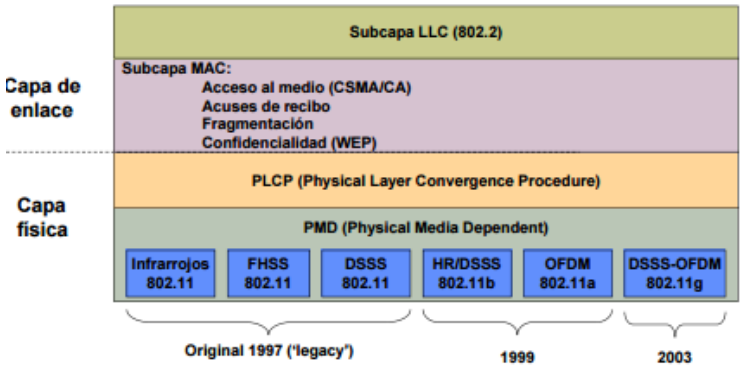

Figura 2.10: Capas de funcionamiento Wlan.

Capara física

La capa física tiene como finalidad transportar correctamente la señal que corresponde a 0 y 1 de los datos que el transmisor desea enviar al receptor. Esta capa se encarga principalmente de la modulación y codificación de los datos.

- **Técnicas de Modulación:** Un aspecto importante que influencia la transferencia de datos es la técnica de modulación elegida. A medida que los datos se codifican más eficientemente, se logran tasas o flujos de bits mayores dentro del mismo ancho de banda.

  A continuación se citan algunas de las técnicas de modulación utilizadas:

  - FHSS (Frequency Hopping Spread Spectrum): se basa en el concepto de transmitir sobre una frecuencia por un tiempo determinado, después aleatoriamente saltar a otra.
  - DSSS (Direct Sequence Spread Spectrum): implica que para cada bit de datos, una secuencia de bits (llamada secuencia seudoaleatoria, identificada en inglés como PN) debe ser transmitida. Cada bit correspondiente a un 1 es substituido por una secuencia de bits específica y el bit igual a 0 es substituido por su complemento.
  - OFDM (Orthogonal Frequency-Division Multiplexing): Una señal OFDM es la suma de un número de subportadoras ortogonales, donde cada subportadora se modula independientemente usando QAM (modulación de fase y amplitud) o PSK (modulación de fase).

- **Subcapa PLCP:** La subcapa PLCP desempeña las funciones que son comunes a todos los medios de transmisión. La subcapa incorpora una cabecera que se antepone a la trama MAC. La trama así construida es la que se transmite en el medio físico **[17]**.

  Las principales funciones que desempeña la cabecera PLCP son:

- Establecer la sincronización entre emisor y receptores a fin de que interpreten correctamente el principio de cada bit y de la trama misma.
- Indicar la velocidad de transmisión utilizada.se transmite el preámbulo y la cabecera a 1Mbps para que todos lo puedan recibir (todos reciben a 1 Mbps como mínimo), el resto a la velocidad indicada en la cabecera.

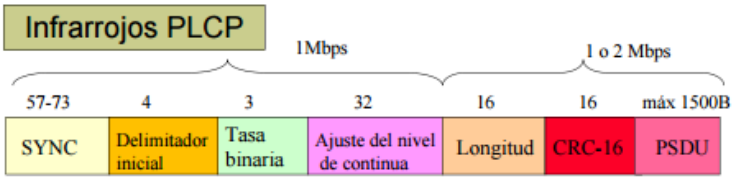

Figura 2.11: Cabecera PLCP.

Capa de enlace [17]

La capa de transmisión de datos de 802.11, se compone de dos partes:

- **Control de acceso al medio (MAC):** En un medio inalámbrico no se puede detectar colisiones pues la señal recibida es enmascarada por la transmitida debido a su bajo nivel de potencia, por lo que se deberían implementar algoritmos que permitan evitar las colisiones.

  El protocolo utilizado para evitar colisiones, CA (Collision Avoidance), es el CSMA/CA. A continuación se describe su funcionamiento.

  En primer lugar se escucha el medio:

  - Si está libre espera un tiempo DIFS, si tras este tiempo el medio sigue libre transmitimos nuestra trama.
  - Si está ocupado espera hasta que el canal esté libre. Cuando el canal esté libre escucha durante DIFS más un tiempo aleatorio dado por el algoritmo de backoff. Posteriormente, si el canal aún está libre transmite.

  Las transmisiones se realizan con reconocimiento, y no se envían más datos hasta que se reciba reconocimiento aunque la capa PHY indique que el medio está libre.

- **Control lógico del enlace (LLC):** gestiona los enlaces lógicos de nivel 2 y proporciona una interfaz común para el nivel de red, ocultando las diferencias relativas a la topología, al medio físico y a las técnicas de acceso al canal **[17]**.

  Se definen tres tipos de operación:

  - Tipo 1: Soporta un Servicio No Orientado a la Conexión sin reconocimiento.
  - Tipo 2. Soporta un Servicio Orientado a la Conexión.
  - Tipo 3. Soporta un Servicio No Orientado a la Conexión con reconocimiento.

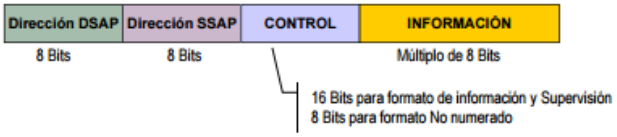

Figura 2.12: Cabecera LLC

DSAP y el SSAP especifican los usuarios LLC destino y origen. Además, indica dirección individual o de grupo o si es una orden o una respuesta.

2.2.2.4 Implementación Wi-Fi en interiores

Ésta tecnología se basa en una combinación de mapas de potencia Wi-Fi RSS combinado con un análisis del entorno, tiene tres fases **[8]** que se indican en la Figura 2.13.

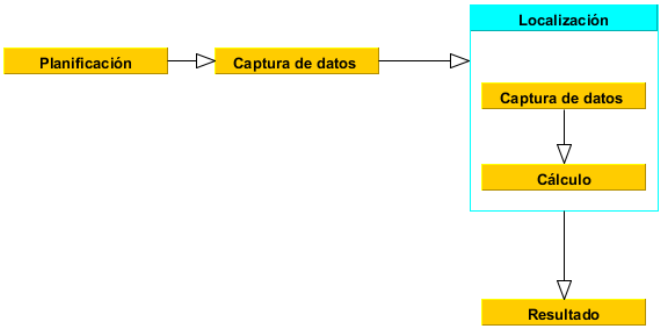

Figura 2.13: Diagrama de flujos de posicionamiento indoor por Wi-Fi.

- Planificación
- Captura de datos
- Localización

En la primera etapa denominada Planificación **[8]**, se determinan por un lado los puntos de acceso fijos a emplear por el sistema de posicionamiento, y por otro lado, se determinan cuáles son las posiciones relevantes dentro del escenario.

En la segunda fase denominada Captura de datos **[8]** se utiliza la técnica de Fingeprinting, ya explicada en puntos anteriores.

El fundamento de esta técnica es crear una base de datos de fuerzas tomadas en diferentes puntos del área de interés donde se implementará el sistema.

Los dispositivos a localizar medirán la potencia recibida y otros datos de los puntos de acceso situados en el área y los enviarán a un servidor que consulta la base de datos y mediante algoritmos de estimación se calcularán las coordenadas del dispositivo.

Como sabemos del funcionamiento del estándar IEEE 802.11, el punto de acceso transmite de forma periódica una trama llamada trama Beacon, en la que anuncia la presencia de la red WLAN dentro de una estructura BSS.

El contenido de la trama Beacon es:

Timestamp: Sirve para la sincronización de equipos, ya que todos los equipos que la reciben actualizan su reloj interno a esta hora.

SSID: Es una cadena alfanumérica que identifica a la red WLAN.

BSSID: Es la dirección MAC del punto de acceso que gestiona la infraestructura BSS

Tasas Soportadas: Indica las tasas que la red WLAN soporta, esto viene limitado por el punto de acceso.

Información de Capacidad: Indica los requerimientos que las estaciones necesitan para conectarse a la red.

Los dos campos más importantes de la trama beacon son el SSID y el BSSID, ya que estos son los utilizados por el dispositivo a ubicar junto con el RSSI. El RSSI(Received Signal Strength Indicator) es un término utilizado para medir de forma relativa la calidad de la señal recibida por el dispositivo. Cada fabricante define el rango de su escala, por ejemplo Cisco tiene un rango de 0 a 100, siendo el 0 la mejor potencia recibida y el 100 la peor. Por tanto para el mapeo del área se utiliza la información de las tramas Beacon.

El mapeo se puede realizar de dos formas diferentes, a continuación se describen los dos métodos existentes.

El primer método es manual, es decir lo realiza una persona con un aparato de medida, midiendo así en cada punto del espacio la potencia de los diferentes puntos de acceso durante un determinado tiempo y calculando la media de los valores medidos. Una vez medidos los valores RSSI en un punto del espacio, a dicho punto se le asigna una coordenada (x,y), luego a una distancia fija de este punto se calcula el siguiente punto. El procedimiento se repite hasta que se mapea toda la zona.

Como se puede deducir el tiempo de medida y la distancia entre punto interviene mucho en la precisión del sistema, por lo que se deben escoger parámetros que hagan que la precisión sea alta. De normal la distancia entre puntos es de 0.5 metros y el tiempo medio de medición es de 10 segundos.

Como se puede deducir cada punto tendrá más de RSSI y BSSID (cada uno corresponde a un punto de acceso).

El segundo método consiste en emplear herramientas de simulación de propagación, es decir se mider la potencia de la señal en determinados puntos y en base a estos estimar el resto de puntos y a diferentes tipos de canales indoor. Como ejemplo de este tipo de herramienta podemos citar Ekahau.

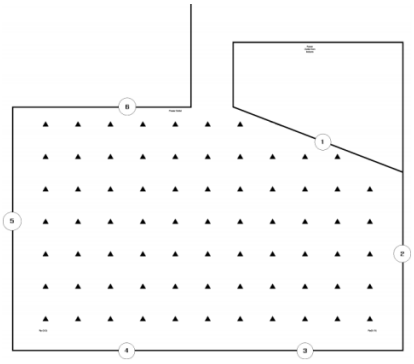

Figura 2.14: Ejemplo mapeo de área indoor.

Los círculos representan los APs y los triángulos representan coordenadas (x,y).

Con todos los puntos calculados se crea la base de datos **[8]**, que se aloja en un servidor que además de contener el mapeo implementa también algoritmos que permiten estimar la posición del dispositivo y que corresponde a la fase de localización.

Cuando un dispositivo navegue por el área mapeada, recibirá las tramas Beacons de los distintos puntos de acceso, la información de estas tramas junto con los niveles RSSI son enviados al servidor encargado de estimar la posición (x,y). En la siguiente figura se representa el procesa anteriormente descrito **[8]**.

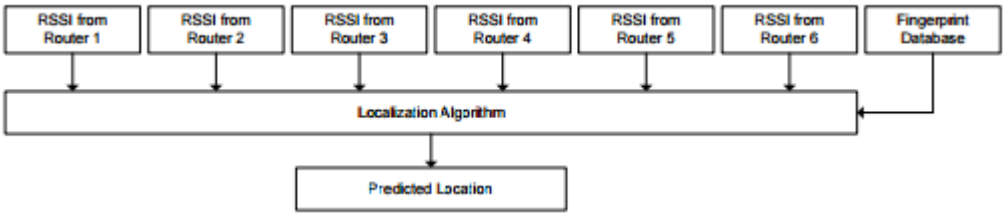

Figura 2.15: Diagrama de predicción de la ubicación.

El problema de la estimación de la posición de un dispositivo de comunicaciones móvil (en adelante móvil) a partir de las medidas del nivel de señal recibida (RSSI) de los distintos Access Points (APs) presentes en el edificio, se puede modelar matemáticamente de la siguiente forma **[10]**:

Sea $X(t)=\{x(t),y(t)\}$ la posición del móvil en tiempo t (posición en un plano).

Sea $S(t)=\{s_1(t),\ldots, s_p(t)\}$ el vector de las RSS medidas en dicho punto, siendo P la cantidad de APs presentes en el edificio.

Se busca un estimador de la posición en t de la forma:

$$X(t) = f(S(t)).$$

Ecuación 2.6: Estimador de posición **[10]**.

Por lo tanto el objetivo es encontrar una función $f(\cdot)$ que sea la mejor estimación posible, es decir que minimice el error cuadrático medio (MSE).

Además es posible mejorar el estimador para el problema de seguimiento, donde se agrega la información adicional de la posición en uno o varios instantes anteriores $t-1, t$

− 2, ..., t − N, por lo que el estimador queda de la forma: X(t) = f(S(t), X(t − 1), X(t − 2), ..., X(t − N)).

Existe una variedad de algoritmos que permiten estimar o más bien comparar los valores medidos por el dispositivo a ubicar con los guardas en la base de datos para poder así determinar su posición. A continuación describiremos los más utilizados.

Vecino más cercano [10]

La técnica de clasificación de los K vecinos más cercanos corresponde a un enfoque determinístico y permite calcular a partir de S(t) cuáles son los K valores más cercanos de las huellas S1, S2, ..., SN. Para ello es necesario definir una medida de distancia en el espacio de las señales recibidas (RSS). Las medidas de distancia más utilizadas son Manhattan (norma-1), norma euclideana (norma-2) o Mahalanobis4 A partir de los K vecinos más cercanos S1, .., SK correspondientes a las posiciones X1, .., XK, puedo calcular la posición X(t) como una combinación lineal de dichas posiciones, de la forma **[10]**:

$$X = \sum_{i=1}^{K} \frac{\omega_i}{\sum_{j=1}^{K} \omega_j} X_i$$

Ecuación 2.7: Vecino más cercano.

El método correspondiente a pesos $\omega_i = 1$ se denomina KNN (K-Neareast Neighbor) y simplemente estima la posición como un promedio de la posición de los K puntos de referencia con valores de RSSI más similares (cuya distancia es menor) a las medidas de RSSI obtenidas.

Cuando se usan pesos distintos para cada punto, el método se denomina WKNN (Weighted K-Neareast Neighbor) **[10]**. Una posibilidad es usar como pesos los inversos de las distancias a cada vecino, es decir wi = 1/d(Si , S) **[10]**, siendo d(Si , S) la distancia de S al vecino i. De esta forma la posición se estima como el baricentro de los K vecinos más cercanos.

Como podemos ver este algoritmo no utiliza posiciones anteriores para el cálculo de la posición actual, con esto podemos concluir que este método se utiliza más bien para situación de localización sin seguimiento continuo.

Estimación de máxima verosimilitud[10] [18]

La estimación de máxima verosimilitud corresponde a un enfoque probabilístico o bayesiano. En este caso se asume que la posición a estimar es una variable aleatoria y se busca la estimación que maximice la probabilidad condicional (posterior) dada la información que brindan las huellas **[10]**.

Redes neuronales, SVM y árboles de decisión [10] [18]

El problema de la estimación de la posición a partir de huellas puede verse como un problema de clasificación. En este enfoque las clases corresponden a los puntos de referencia (huellas) y se busca decidir a qué clase pertenece un punto dadas las RSSI recibidas. Otro enfoque posible es encarar la estimación como un problema de regresión funcional, donde lo que se desea es la mejor estimación posible de la función que mapea

RSSI con la posición y para ello se cuenta la función evaluada en una serie de puntos, en este caso las huellas. Las redes neuronales, SVM y los árboles de decisión son precisamente herramientas ampliamente utilizadas para problemas de clasificación, así como métodos de aprendizaje, en particular para el aprendizaje de funciones no lineales como es el caso en el problema de localización.

Los algoritmos a continuación descritos utilizan la información de las anteriores posiciones para predecir de forma correcta la siguiente posición.

Filtro bayesiano[10] [18]

Este método corresponde a la extensión de la estimación de máxima verosimilitud vista anteriormente, pero ahora agregando la información del pasado de la trayectoria. En este caso el modelo de movimiento del móvil se define en la elección del prior.

Filtro de Kalman[10] [18]

El filtro de Kalman corresponde a un caso particular de filtro bayesiano, el cual considera un modelo lineal y ruido blanco gaussiano de media nula. Dado un modelo de esta forma, el filtro de Kalman es un estimador recursivo óptimo si se considera el MSE como medida de error. Existen diversas formas de encarar el problema de seguimiento basado en huellas mediante el filtro de Kalman.

Filtro de partículas [10]

El filtro de partículas es un método de Monte Carlo secuencial que genera muestras aleatorias (partículas) según cierto modelo de movimiento y estima sus respectivas densidades de probabilidad a partir de las huellas. A diferencia del filtro de Kalman permite trabajar con modelos no lineales y ruidos no gaussianos. Este enfoque permite además incorporar la información del mapa del edificio para mejorar la estimación, eliminando las partículas que hacen movimientos imposibles como atravesar paredes. La principal desventaja de estos métodos es el elevado costo computacional, una limitante importante para su uso en aplicaciones de tiempo real con móviles de baja capacidad. En este trabajo no se utilizó este método, por más detalle ver.

En la siguiente figura se representa de forma gráfica un sistema de posicionamiento indoor basado en tecnología Wi-Fi.

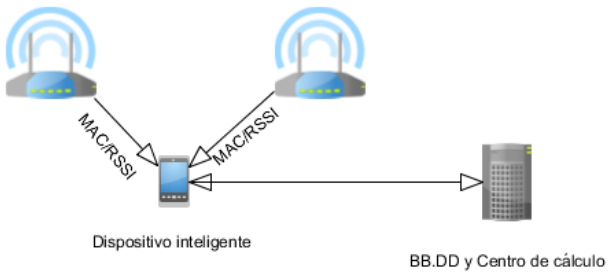

Figura 2.16: Sistema indoor por WI-FI

Conclusión

El mayor problema de esta tecnología es la propagación de la señal, zonas sin cobertura, las fluctuaciones que esta puede sufrir debido a cambios en el entorno y las

interferencias debido a señales de la misma frecuencia al utilizar una frecuencia sin licencia y ruido introducido por equipos eléctricos. Debido a todos estos inconvenientes esta tecnología presenta niveles de precisión muy bajaos por lo que su utilización como un sistema único sería ineficaz por lo que se debería utilizar con otros sistemas, en la mayoría de las veces se utiliza con los sistemas inerciales. A este sistema se le denomina sistema híbrido.

La tecnología Wi-Fi puede ser utilizada con cualquiera de las técnicas de localización anteriormente descrita, sin embargo la idónea es la técnica de análisis de Fingerprinting.

2.2.3 Basados en sistemas inerciales

Los sistemas de navegación inerciales (INS) surgen para evitar la dependencia que tienen los sistemas inalámbricos de posicionamiento de la infraestructura de red previa y del hardware específico del receptor.

Los sistemas INS, se basan en unidades de medida inercial llamados IMUs mediante las cuales se realiza el seguimiento y el posicionamiento relativo de un individuo en relación a su punto de partida, orientación y velocidad de marcha.

Estos sistemas usan sensores inerciales como acelerómetros, giroscopios, contador de pasos y brújulas que determinan la distancia recorrida y la orientación de movimiento del usuario. Además, requieren de una inicialización o calibración en la que se indique o se mida la posición inicial sobre la que se va a aplicar el movimiento y con ello se obtiene la posición (referida siempre a la posición inicial) de forma autónoma.

La gran ventaja de los INS es que no precisan de infraestructura de red externa, con lo que su despliegue es muy barato. Además, en la actualidad la mayor parte de los teléfonos móviles cuentan con sensores de la IMU, por lo que se puede emplear este sistema en cualquier entorno indoor. Sin embargo, cuenta con dos inconvenientes principales: en primer lugar, requiere una inicialización de gran exactitud para que las medidas obtenidas mediante estos sistemas sean correctas. Por otro lado, el error en la medida de estos sistemas es elevado y acumulativo, por lo que es necesario "reinicializar" o recalibrar periódicamente el sistema y/o combinarlo con otros sistemas de posicionamiento en interiores.

A continuación haremos una pequeña descripción de todos aquellos sensores que intervienen en el posicionamiento indoor basado sistemas inerciales.

Acelerómetro

Un acelerómetro es un dispositivo que mide la aceleración de un elemento. No está necesariamente relacionado con el cambio de velocidad, sino que a veces se asocia con el fenómeno de peso experimentado por una masa de referencia conocida por el dispositivo de medida. Generalmente, el acelerómetro mide la aceleración mediante la medida de cuánto presiona la masa sobre algo cuando una fuerza actúa sobre ella **[19]**.

Su funcionamiento se basa en el movimiento de diminutas barras metálicas que se desplazan con el propio aparato y provocan un cambio en la capacidad eléctrica, generando unos datos que después son interpretados en clave tridimensional.

Existen varios tipos de tecnologías (piezo-eléctrico, piezo-resistivo, galgas extensométricas, láser, térmico…) y diseños que aunque todos tienen el mismo fin

pueden ser muy distintos unos de otros según la aplicación a la cual van destinados y las condiciones en las que han de trabajar.

Los acelerómetros han pasado de estar dedicados a un uso industrial (medir vibraciones y oscilaciones) y de investigación a estar presentes en muchos aparatos cotidianos.

Figura 2.17: Acelerómetro de móvil.

La fuente de error principal de un acelerómetro **[19]** es el "bias" (m/s2) (offset de la señal de salida sobre el valor real), el cual difiere de cada acelerómetro concreto. Un error de bias constante de valor ε provoca un error en la posición que al ser doblemente integrado supone un crecimiento cuadrático a lo largo del tiempo. El error acumulado a lo largo del tiempo en la posición estimada es: $s(t) = \varepsilon * t^2 /2$, siendo t el tiempo en el que se considera. Es posible estimar el valor del error bias mediante una medida del valor medio de la salida del acelerómetro a largo plazo cuando no se experimenta ninguna aceleración. Con ello, se puede recalibrar la medida de aceleración periódicamente, de forma que el error se reduzca. Los errores de bias no corregidos son los que limitan, principalmente, el rendimiento del sistema inercial de localización.

Giroscopio[19]

Un giroscopio es un dispositivo de medida de la orientación, basado en los principios de conservación del momento angular. Un giroscopio convencional suele ser mecánico, y consiste en 3 ruedas giratorias montada en un eje de simetría, permitiéndole la rotación en los 3 ejes. Cuando el giróscopo se somete a un cambio de orientación de su eje, cambia de orientación (o experimenta un momento angular) girando respecto de un tercer eje, perpendicular tanto a aquel respecto del cual se lo ha empujado a girar, como a su eje de rotación inicial.

En los teléfonos móviles, por el contrario se emplean la tecnología MEMS (Micro Electro-Mechanical System), que mide el cambio angular y con ello el giro. Los giroscopios de tecnología MEMS contienen elementos vibrantes que miden el efecto Coriolis, que permite calcular la velocidad angular mediante la medida de rotación en diferentes ejes.

En el giroscopio, al igual que en el acelerómetro, la fuente de error principal es el "bias", que es el offset entre la medida de orientación indicada y la real cuando no se está realizando ningún giro. Un error de bias constante de valor ε provoca un error en la medida que crece linealmente a lo largo del tiempo. El error acumulado a lo largo del tiempo es: $\Theta(t) = \varepsilon * t$, siendo t el tiempo en el que se considera. Es posible estimar el valor del error bias mediante una medida del valor medio de la salida del giroscopio a largo plazo cuando no se experimenta ninguna rotación. Una vez conocida, es fácil corregir las medidas de salida, simplemente restándosela al valor medido.

Magnetómetro (brújulas) **[19]**

Un magnetómetro es un instrumento para medir la fuerza y dirección del campo magnético en el área cercano al dispositivo. En los dispositivos móviles, en general, se emplean magnetómetros vectoriales, que tienen la capacidad de medir la componente de campo magnético en una determinada dirección relativa a la orientación espacial del propio dispositivo.

Las fuentes principales de errores de medida de este dispositivo son la contaminación magnética del sensor y la presencia de elementos férreos en el instrumento. Este error es o bien intrínseco de la fabricación o bien del estado o entorno en el que se encuentra el dispositivo, por lo que es difícil de corregir. Sin embargo, mediante medidas a largo plazo en una situación estática, se puede estimar cuál es el error en la medida y corregirlo en la salida.

Step Counter[19]

Es un sensor tipo software que utiliza como base el acelerómetro, por lo que su implementación sería sola en aquellos dispositivos que tienen el acelerómetro.

Para detectar un paso se monitoriza el acelerómetro y cuando se detecta un pico o un salto grande en la aceleración en una determinada dirección se cuenta el paso.

Todas las plataformas móviles proporcionan APIs que permiten usar este sensor sin la necesidad implementar la lógica y el cálculo de la detección del paso.

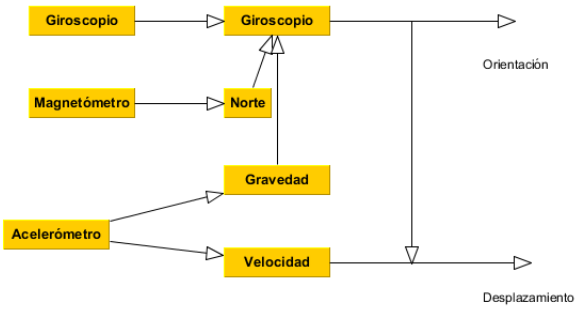

Figura 2.18: Sistema de posicionamiento inercial.

Los sensores más utilizados de los anteriormente explicados son el acelerómetro y el giroscopio.

El primero calcula la posición del objeto en movimiento mediante la doble integración de la aceleración. A continuación se explica un algoritmo sencillo para una precisión normal, para una mejor precisión se han de tomar unas consideraciones que pueden hacer que este algoritmo sea un poco más preciso.

Como sabemos, la aceleración es la variación de la velocidad de un objeto con el tiempo, al mismo tiempo la velocidad es la variación de la posición de un objeto con el tiempo. Con otras palabras la velocidad es la derivada del desplazamiento y la aceleración es la derivada de la velocidad. Dicha relación queda expresada en las siguientes formulas.

$$\vec{a} = \frac{d\vec{v}}{dt} \ , \vec{v} = \frac{d\vec{s}}{dt} \ \ por\ tanto\ \ \vec{a} = \frac{d(\overrightarrow{ds})}{dt^2}$$

Ecuación 2.8: Formula de aceleración y velocidad **[20]**.

Por lo tanto conocida la aceleración (valor que obtenemos del acelerómetro) hay que hacer la inversa de la segunda derivada para sacar el desplazamiento, lo que viene a ser la doble integral.

$$A = \int (\int \vec{a}\, dt) dt$$

Ecuación 2.9: Formula de desplazamiento **[20]**.

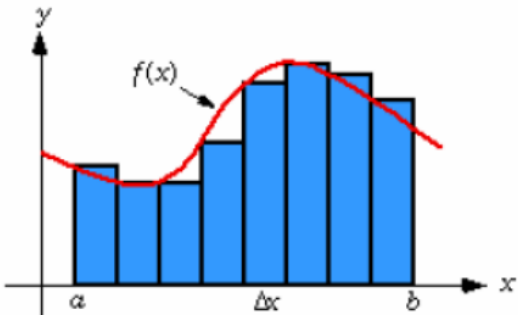

Figura 2.19: Ejemplo Integral.

La mejor aproximación para el cálculo de la integral sería:

$$área = (muestra_n + \frac{muestra_n - muestra_{n-1}}{2}) * T$$

Ecuación 2.10: Cálculo de desplazamiento **[20]**.

La representación gráfica de la formula antes descrita se refleja en la siguiente figura **[20]**.

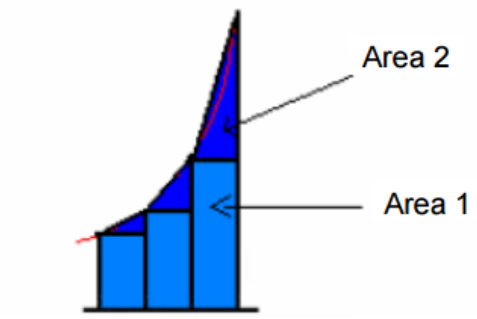

Figura 2.20: Cálculo aproximado del desplazamiento.

Este sistema necesita un punto de partida o referencia para a partir del cual determinar la dirección del desplazamiento mediante el giroscopio y el magnetómetro y el desplazamiento mediante el acelerómetro, ver figura 2.20.

Como se explicó antes el acelerómetro tiene un offset que corresponde a los valores que se obtienen cuando el móvil está en reposo, entonces una forma de eliminar este ruido se realiza mediante la medida de la salida del acelerómetro cuando el móvil está en

reposo durante un tiempo y establecer la media de estos valores como el valor del offset. Este offset calculado sería valor cero de nuestros datos por lo que habría que restarlo siempre a los valores obtenidos.

El segundo sensor es el giroscopio o giróscopo, este nos proporciona la velocidad de rotación en rad/s del objeto respecto al eje x, y y z. Por lo que para obtener el ángulo de giro se integra la salida de este en el tiempo.

Como hemos comentado anteriormente este sensor no es perfecto y presenta unas pequeños errores en sus medidas, que integradas en el tiempo pueden generar errores considerables, este error se denomina "drift over time". Para reducir este error, dicho sensor se suele fusionar con otros dos sensores, el magnetómetro y el acelerómetro, esta fusión se lleva a cabo mediante el filtros, siendo el más efectivo el filtro Kalman.

Para el cálculo de la orientación Android facilita la librería que permite su cálculo, por lo que le resta mucha complejidad matemática a este punto.

La gran ventaja del posicionamiento por sistemas inerciales es su coste nulo, ya que aprovecha los recursos de nuestros móviles para posicionarnos.

Sin embargo este sistema tiene varias desventajas que hacen que su uso individual como sistema de posicionamiento sea inviable. Una de sus desventajas es el error acumulativo de las medidas ya sea debido al error del offset, al cálculo aproximado y al ruido generado por los sensores, otro también es la consumación de la batería por lo que su uso incontrolado es una gran desventaja.

Un forma de remediar estas desventajas es utilizar sensores inerciales en combinación con otros sistemas de posicionamiento absoluto, de forma que los IMU únicamente provean la información necesaria para completar y corregir la posición del usuario en términos (X, Y), con datos acerca de su orientación, sentido y velocidad de marcha.

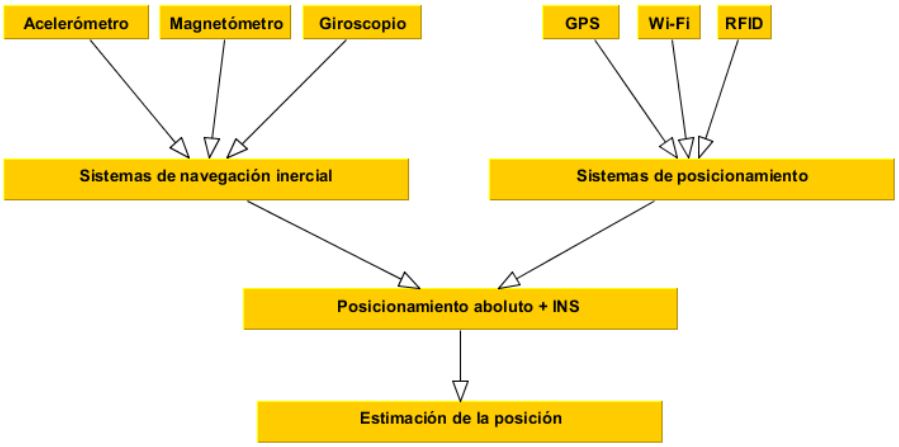

Figura 2.21: Arquitectura híbrida.

Como podemos observar en la figura 2.21 un sistema combinado de estas características será capaz de proveer información de posicionamiento de gran precisión y por tanto ampliar considerablemente el abanico de posibles aplicaciones prácticas con técnicas de localización de personas.

Capítulo 3. Solución escogida

La solución que brinda este proyecto es la basada en sistemas inerciales, sin embargo esta solución no se implementará como un sistema absoluto de posicionamiento sino será un subsistema de apoyo a otro proyecto que utiliza la tecnología Wi-Fi. La fusión de estos dos subsistemas es la solución final que brindará un alto nivel de precisión y permitirá su uso en varias aplicaciones.

A continuación se describen las dos formas o soluciones con las que se pretendía implementar el subsistema. Las dos soluciones emplean el mismo hardware, sin embargo se diferencian en la forma de utilizar dicho hardware. La primera solución ha resultado inviable al evaluarla debido a la gran imprecisión que esta presentaba. Mientras que la segunda solución brinda buenos resultados, por lo que será la solución de implementación de este subsistema.

3.1 Primera solución

La primera solución que se dio al proyecto, utiliza los sensores acelerómetro, magnetómetro y giroscopio. Con el acelerómetro se calcula el desplazamiento del dispositivo en línea recta, en la dirección del eje, esto se consigue mediante la monitorización del de las salidas del acelerómetro del móvil cada 20 ms, lo que viene a ser muestrear a 20 ms. Para minimizar el error del offset (ver 2.2.3 Basados en sistemas inerciales) de dicho sensor se resta un valor ya calculado (que se calcula mediante la media aritmética, cuando el móvil está en reposo horizontalmente sobre una mesa, ver punto calibración). Para el cálculo de una aceleración del el objeto en un determinado instante se calcula la media de las valores obtenido durante medio segundo. Una vez calculada la aceleración se utiliza la formula (Ecuación 2.10: Cálculo de desplazamiento ***[20]***.).

Mientras que con el giroscopio se detecta el giro del dispositivo y con ello se obtiene el ángulo de giro. En realidad para el cálculo del giro no se utiliza solo el giroscopio, sino que se utiliza también el acelerómetro y el magnetómetro para mejorar la precisión y para la obtención del ángulo con referencia del norte, esto se consigue mediante el empleo del filtro complementario. Sin embargo el elemento o el sensor fundamental de ambos para el ángulo es el giroscopio.

Para la obtención del ángulo se minimiza de la misma forma que el acelerómetro el error del offset y se elimina el ruido de la los datos mediante un filtro de media, el filtro se basa en el cálculo de la media durante un tiempo (denominado la ventana del filtro) y se considera el valor obtenido de esta forma se consigue rebajar el error generado por el sistema. Para un mejor resultado de la medición se utilizan los filtros Kalman, sin embargo la implementación de estos filtros es muy compleja por lo que en este proyecto no se ha llevado a cabo. En este caso para los cálculos, no se necesita emplear un método aproximado para hacer las integrales como en el caso del acelerómetro, si no que se emplean librerías facilitadas por Android, que facilitan mucho la implementación de esta parte.

Para una mejor monitorización y estudio del funcionamiento de cada sensor y para así detectar los errores y valorar la viabilidad de las medidas y de los cálculos de los sensores se ha diseñado una aplicación para cada caso, es decir una aplicación que solo implementa el acelerómetro para obtener el desplazamiento y otra para obtener el ángulo del giro del dispositivo y su orientación respecto al norte.

En primer lugar se evalúa la viabilidad del acelerómetro (desplazamiento). Para ello se han definido tres escenarios de prueba concretos:

- Escenario 1: El móvil permanece en reposo horizontal sobre una mesa.
- Escenario 2: El móvil permanece en reposo horizontal sobre la mano de una persona
- Escenario 3: El móvil se mantiene quieto y horizontal sobre la palma de la mano de un individuo que se desplaza hacia el norte 4 metros.

Tabla 3.1: Evaluación del acelerómetro.

Escenario	**Desplazamiento (m)**		
	X	Y	Z
1	0	0	0
2	0.06	0.15	0.53
3	9.0	9.6	15

En el escenario 1, las medidas obtenidas son las correctas, ya que en este caso el móvil está en reposo, no existe ningún desplazamiento. Este escenario nos puede ayudar a comprobar la eficacia de la eliminación del error de offset.

En el escenario 2, el móvil permanece en reposo pero esta vez sobre la mano del individuo, sin embargo en este caso el desplazamiento es diferente a 0 m, en el caso ideal el resultado debería ser 0, por lo que las medidas obtenidas representan el error, este error es principalmente debido a la sensibilidad que tienen los sensores al movimiento.

En

Tabla 3.1: Evaluación del acelerómetro.

anterior podemos ver que la dirección que mayor error introduce es la Z (dirección hacía el suelo). El cálculo de este error se hizo durante 4 segundos, por lo que el error va aumentando en un valor medio de 5 cm cada segundo aproximadamente para los tres ejes. Con esto podemos deducir la gran sensibilidad que tiene este sensor a los movimientos y el ruido que este presenta.

En el escenario 3, el móvil es desplazado 4 m en dirección lineal, en los tres ejes: Podemos ver en la tabla 3.1**¡Error! No se encuentra el origen de la referencia.** anterior, el desplazamiento obtenido, en este caso, podemos ver que el resultado obtenido defiere mucho de la realidad, el error es demasiado alto, siendo el error en el plano paralelo a la tierra de 5 m en cada dirección, mientras que la vertical al plano de la tierra (la altura, el eje Z) tiene un error de 11 m.

Podemos concluir del escenario 2 y 3 que el cálculo del desplazamiento con el acelerómetro queda inviable, ya que presenta un gran error en las medidas. Este error, como se ha comentado anteriormente es debido al ruido generado por el sistema, por la sensibilidad del sensor, y la integración doble de los datos hace que el error sea doble, invalidando de esta forma esta solución propuesta.

En la siguiente figura se muestra la aplicación que calcula el desplazamiento.

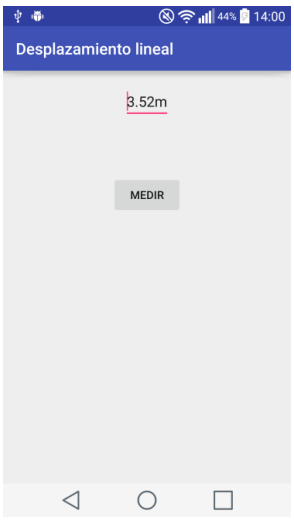

Figura 3.1: Aplicación que calcula desplazamiento.

Aunque se ha concluido de la evaluación anterior, que esta solución es inviable debido al error del acelerómetro, se ha decidido seguir con la evaluación de la segunda parte de la solución (giróscopo, obtención de la orientación) para saber de esta forma su viabilidad en el caso de su implementación en la segundo solución.

Al igual que en el caso del cálculo del desplazamiento, se hizo una aplicación concretamente para evaluar las medidas del giroscopio. Esta aplicación nos da el ángulo de giro del dispositivo en los tres ejes. Para ello se han definido tres escenarios:

- Escenario 1: Situar el móvil horizontalmente sobre una mesa en dirección norte
- Escenario 2: Situar el móvil horizontalmente sobre una mesa y lanzar varias veces la aplicación, también en dirección norte.

Tabla 3.2: Evaluación del giroscopio.

Escenario	**Lanzamiento Nº**	**Ángulo (º)**		
		X	Y	Z
1		1,99	0.45	0.4
2	1	2.80	1.72	0.0
	2	2.44	2.28	1.04
	3	1.92	1.84	0.61

En el escenario 1 se ha querido evaluar la precisión del giroscopio para determinar el norte, la dirección del norte queda determinada por la variable X. Como podemos ver en la tabla anterior, el error generado por el giroscopio es de aproximadamente 2 grados, es un error despreciable. Al estar el móvil situado horizontalmente sobre una mesa, las variables Y y Z deberían tener 0 grados, sin embargo existe un error, pero este es despreciable.

Para la evaluación del escenario 1, se ha comparado el resultado de nuestra aplicación con varias aplicaciones descargadas de Play Store, que presentan un alto nivel de precisión.

En el segundo escenario, se ha lanzado la actividad varias veces para ver la diferencia entre los grados obtenidos en cada lanzamiento, y de esta forma ver su precisión y su error, ya que el móvil no se mueve. Como podemos ver en la tabla 3.2, la diferencia en grados entre los tres lanzamientos de la actividad es muy pequeña, esto nos informa de que la aplicación no tiene casi error.

Se concluye de los resultados de los tres escenarios, que la implementación del giroscopio es viable, ya que presenta errores despreciables por lo tanto se empleará en la segunda solución.

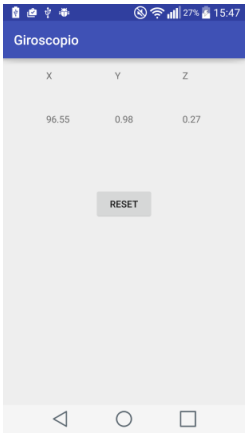

Figura 3.2: Aplicación que calcula ángulo de giro.

Debido a la imprecisión de la solución causada por los datos que nos brinda el acelerómetro, se decidió cambiar de solución ya que esta queda inviable.

3.2 Segunda Solución

La segunda solución propuesta, consiste en utilizar el mismo hardware que la anterior, es decir, los mismos sensores, sin embargo se diferencian en la forma en que estos sensores son tratados. Esta solución utiliza la misma implementación del giroscopio que la anterior, ya que este como se pudo ver ha resultado viable. Por ello podemos concluir que las dos soluciones solo se diferencian en la forma del trato de los datos del sensor acelerómetro.

La segunda solución propuesta consiste en utilizar como sensores el acelerómetro y el giroscopio, sin embargo, esta vez, el acelerómetro es usado para detectar el paso de un ser humano, es decir detectar un cambio de aceleración brusco, a esta implementación o función en nuestro proyecto se le denomina detector de paso.

El detector de paso ya explicado en el punto anterior, utiliza el acelerómetro para contar nuestros pasos. Para contar el paso se monitoriza la salida del acelerómetro y cuando se detecta un pico alto se considera un paso, de esta forma el error generado por el acelerómetro es mínimo. Para la minimización del error del offset y así la eliminación del ruido interpolado en los datos se utilizan los mismos métodos y filtros que los utilizados en la primera solución para el cálculo del acelerómetro y que se explican en siguientes puntos (3.3 Calibración y 3.4 Filtro complementario).

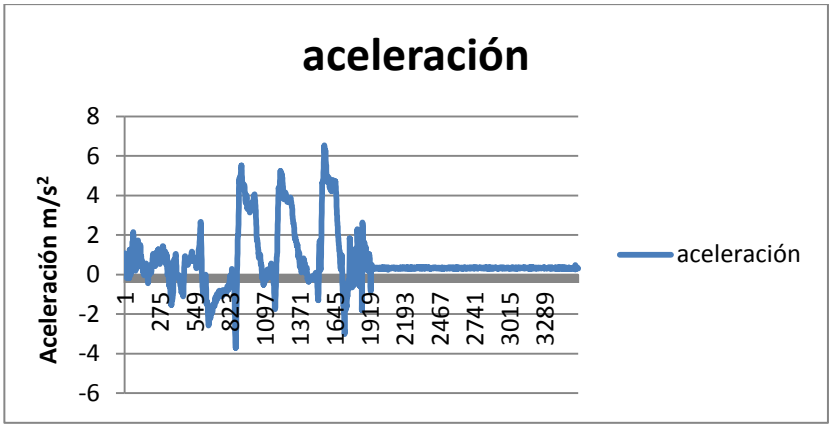

Figura 3.3: Detección de paso.

En la figura 3.3, se puede ver unos picos de aceleración en la salida del acelerómetro que reflejan que un paso, en este caso podemos ver tres picos, lo que corresponde a tres pasos. Como se puede ver también, cada pico tiene cierta duración, esto es debido a que el periodo de muestreo del acelerómetro es muy pequeño, es decir más pequeño que la duración del paso, concretamente cada microsegundo, debido a esto se calcula la media de las muestras cada segundo, de esta forma se elimina también el ruido generado por el sistema.

El paso medio del ser humano es de 0.4 m, por lo que cada vez que se detecta un paso se desplaza el objeto en el mapa 0.4 metros en la dirección indicada por el giroscopio.

La metodología empleada en esta solución para evaluar la precisión de las medidas ofrecidas por el sensor es la misma que en la primera solución, es decir se crea una aplicación precisamente para evaluar las medidas del detector, la evaluación de este elemento de la arquitectura de la aplicación es realizada o explicada en el punto pruebas y resultados.

Evaluando las medidas del sensor y los cálculos se concluye que son aceptables y presentan una mejor precisión que la primera solución. Por lo cual se confirma que esta solución es adecuada, por lo que será la implementada en este trabajo.

El coste de esta implementación es casi nulo, ya que se utiliza tecnología que disponemos hoy en día, sin embargo este coste mínimo produce un coste en precisión, al ser la tecnología de no alta gama, y esto a su vez conlleva a que número de aplicaciones que se le pueden dar sea limitado.

3.3 Calibración

El objetivo de la calibración es la eliminación del offset presente en los datos obtenidos de los distintos sensores. El offset es el valor que entregan los sensores cuando el dispositivo está en reposo (estático), por lo que debería ser eliminado para alcanzar una mejor precisión. El offset suele ser un error variable no fijo, por lo cual su eliminación es aproximada.

A continuación en la siguiente figura se muestra el offset del sensor acelerómetro en el eje y.

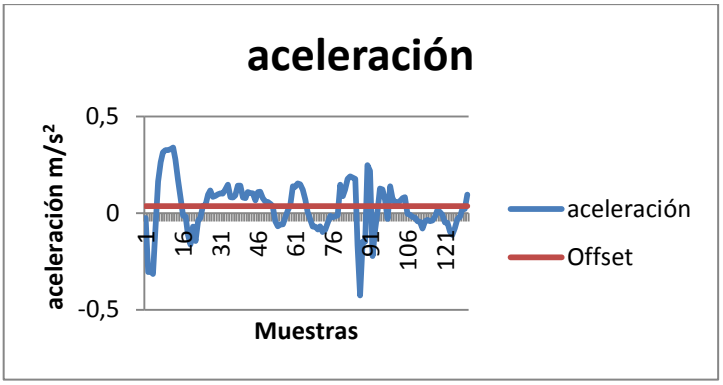

Figura 3.4: Offset del acelerómetro.

Para la realización de la calibración, se inmoviliza el móvil sobre una mesa en una superficie plana, y se monitoriza la salida del sensor durante un periodo de tiempo determinado, denominado tiempo de muestreo. Una vez hecho lo anterior se calcula la media de las muestras monitorizadas durante el tiempo de muestro, la media se convierte en el offset que presenta dicho sensor, que luego se utiliza como el umbral a partir del cual nuestro sistema empieza a considerar los datos válidos. En la figura anterior se muestra la salida del sensor acelerómetro durante 3 segundos y el offset calculado.

Este método de calibración es utilizado en todos los sensores para minimizar así el error del offset, aunque como se puede observar no es del todo efectivo.

3.4 Filtro complementario

Los filtros complementarios son una solución determinista del problema de estimar un conjunto de variables a partir de mediciones provenientes de diferentes sensores, planteada en el dominio frecuencial. Esta clase de filtros han sido estudiados extensivamente en el área de procesamiento digital de señales, de donde surge su definición formal

Filtro Complementario[14]: si FPB es un filtro pasa-bajo cuya función de transferencia está definida por un polinomio de la variable compleja "s" ($H_{PB}(s)$) y FPA un filtro pasa-alto con función de transferencia ($H_{PA}(s)$) , entonces se dice que forman un filtro complementario si cumplen alguna de las siguientes condiciones :

$$H_{PB}(s) + H_{PA}(s) = 1 \quad \mathbf{o} \quad |H_{PB}(s)|^2 + |H_{PA}(s)|^2 = 1$$

Ecuación 3.1: Filtro complementario.

En particular, cuando se desea estimar un ángulo a partir de los datos provenientes de un acelerómetro y un giróscopo se diseña un filtro complementario para aprovechar los datos dentro del rango de frecuencias donde el ruido y las perturbaciones de los sensores son menores. Al pasar los datos del acelerómetro por un filtro pasa-bajo, se eliminan las perturbaciones de alta frecuencia y se aprovecha la precisión de los resultados a largo plazo. En forma análoga, cuando se filtra la señal del giróscopo con un filtro pasa-alto se eliminan los efectos negativos del drifting bias (en Inglés: polarización variable) y se aprovecha su inmunidad a las vibraciones y otras aceleraciones externas que perturban las estimaciones provenientes del acelerómetro. Por lo tanto, el esquema de fusión sensorial basado en un filtro complementario es el que se ilustra en la Figura 3.5.

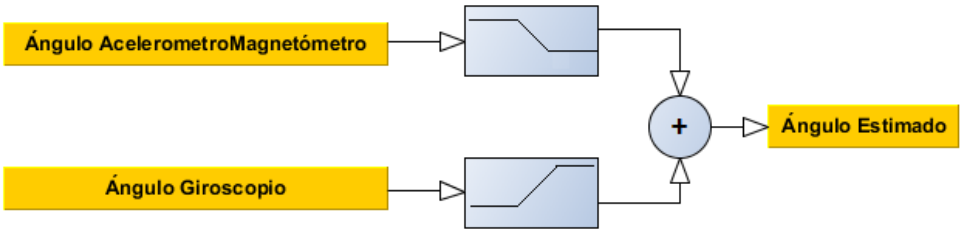

Figura 3.5: Filtro Complementario.

Como se puede ver en la figura 3.5, el filtro complementario implementado en este proyecto es utiliza para mejorar la precisión del giroscopio. Para ello, por una parte se calcula la orientación del móvil mediante los sensores acelerómetro y magnetómetro, el resultado obtenido da la orientación del dispositivo es en azimut, roll y pitch. Por otra parte se calcula a su vez la orientación con el giroscopio, obteniendo un ángulo que tiene como referencia el norte (ángulo 0 grados el norte), al igual nos devuelve que los sensores anteriores, nos brinda la orientación en las tres dirección.

En este proyecto solo se tiene en cuenta la orientación en azimut, al ser la ubicación en plano 2D.

Una vez obtenidas las dos orientación de los dos distintos sensores, estas se ponderan con un factor denominado coeficientes del filtro cada una de tal forma que la suma de los coeficientes es 1, una vez ponderadas son sumadas y de esta forma se obtiene orientación del dispositivo.

Android facilita librerías y clases para el cálculo de la orientación de los distintos sensores, hecho que resta una escasa complejidad al proyecto.

Capítulo 4. Desarrollo del sistema

En este punto de del proyecto se va a explica la implementación de la solución final escogida para llevar a cabo el subsistema que tiene como objetivo abordar el posicionamiento indoor. Así como las tecnologías y herramientas utilizadas.

4.1 Tecnologías y Herramientas Utilizadas

En esta sección se describirán las diferentes tecnologías utilizadas, tales como lenguajes de programación, formatos de intercambio de información, plataformas móviles, herramientas de desarrollo y compilación etc.

4.1.1 Plataforma Android

Se optó por esta plataforma por su presencia en el mercado, ya que ha tenido una amplia aceptación como sistema operativo de dispositivos inteligentes entre los usuarios y la industria, clasificándose como el sistema operativo móvil más utilizado en la actualidad. Ver figura 4.1**[21]**.

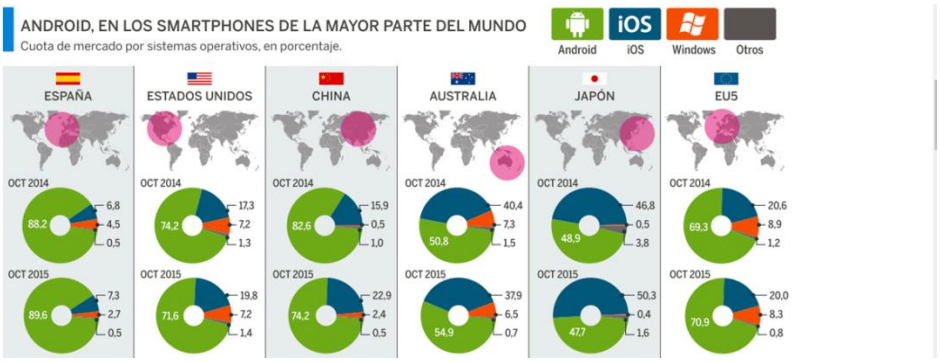

Figura 4.1: Plataformas móviles más utilizadas.

Android es un sistema operativo móvil desarrollado inicialmente por Android Inc. y en 2005 fue adquirido por Google.

Esta plataforma se basa en la versión 2.6 de Linux para el sistema de servicios básicos, tales como la seguridad, la gestión de memoria, la gestión de recursos, la pila de red y el modelo del controlador.

4.1.1.1 Arquitectura [22]

La siguiente figura muestras los diferentes niveles de los que se compone la arquitectura Android.

- **Aplicaciones:** es el último nivel de esta tecnología, está formado por el conjunto de aplicaciones instaladas. Existen varios lenguajes para programas estas aplicaciones como por ejemplo java, C/C++, JavaScript etc. En este proyecto hemos optado por Java.
- **Armazón o entorno de aplicaciones:** Gracias a esta capa, los desarrolladores tienen acceso a los APIs que usan las aplicaciones base. Su diseño permite que unas aplicaciones puedan utilizar componentes de otras, manejar Hardware del dispositivo con cierta facilidad etc.

- **BIBLIOTECAS NATIVAS:** En esta capa se encuentra un conjunto de bibliotecas en C/C++ que el sistema operativo utiliza, entre las que se encuentran: codecs de audio, imágenes y vídeo, encriptación de datos, manejo de gráficos 2D y 3D y manejo de BBDD (Bases de datos) entre otras, estás bibliotecas suelen ser utilizadas para cuando haya repitición.
- **Manejadores en tiempo de ejecución:** Aquí es donde se ejecutan las aplicaciones programadas en java, estas aplicaciones son ejecutadas sobre la máquina virtual Dalvik. El código es compilado a .class y luego a .dex, este último es el interpretado por la máquina virtual. La razón que tiene google de comiplar de .class a .dex es la rapidez y pequeño tamaño de estos últimos, dos factores que hacen que la ejecución de la aplicación sea rápida y sin errores.
- **Kernel De Linux:** Android está construido sobre el núcleo de Linux, pero se ha modificado dramáticamente para adaptarse a dispositivos móviles.

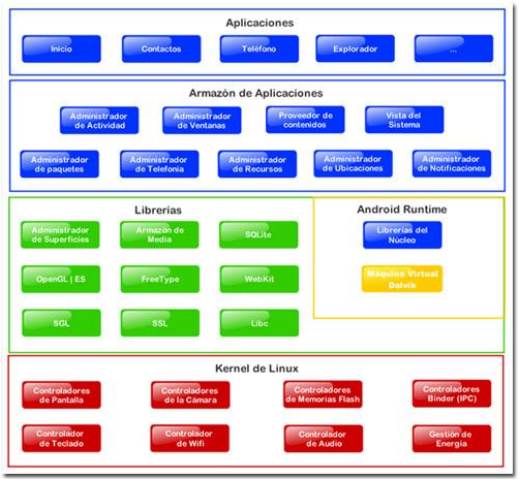

Figura 4.2: Arquitectura Android.

4.1.1.2 Versiones [23]

Como era de esperar las actualizaciones del sistema operativo han ido evolucionando progresivamente desde su lanzamiento hasta la 7.0 correspondiente al API 24, siendo la más reciente.

En la siguiente tabla se hace un pequeño repaso de todas las versiones existentes y resaltando alguna de sus características [11].

Tabla 4.1: Versiones de Android.

Nombre código	Número de versión	Fecha de lanzamiento	Nivel de API
Apple Pie	1.0	23 de septiembre de 2008	1
Banana Bread	1.1	9 de febrero de 2009	2
Cupcake	1.5	27 de abril de 2009	3
Donut	1.6	15 de septiembre de 2009	4

Eclair	2.0–2.1	26 de octubre de 2009	5–7
Froyo	2.2–2.2.3	20 de mayo de 2010	8
Gingerbread	2.3–2.3.7	6 de diciembre de 2010	9–10
Honeycomb1	3.0–3.2.6	22 de febrero de 2011	11–13
Ice Cream Sandwich	4.0–4.0.4	18 de octubre de 2011	14–15
Jelly Bean	4.1–4.3.1	9 de julio de 2012	16–18
KitKat	4.4–4.4.4, 4.4W–4.4W.2	31 de octubre de 2013	19–20
Lollipop	5.0–5.1.1	12 de noviembre de 2014	21–22
Marshmallow	6.0–6.0.1	5 de octubre de 2015	23
Nougat	7.0	15 de junio de 2016	24

4.1.2 Java

Java es un lenguaje de programación que nació en 1991 por la empresa Sun Microsystems (adquirido por Oracle en 2010). Su propósito principal fue crear un lenguaje que cumpliese con la filosofía conocida, por sus siglas en inglés, como WORA ("Write Once, Run Anywhere"), que implica que el código pueda ser ejecutado por cualquier dispositivo sin necesidad de tener que recompilar o adaptar el código.

Esto es posible porque el código es pseudocompilado, es decir, se traduce a Java bytecodes que puede ser interpretado por cualquier máquina virtual de Java (JVM). Por contra, pierde la utilidad de poder utilizar recursos de bajo nivel, que otros lenguajes como C o C++ sí permiten

Entre sus numerosas ventajas se pueden destacar **[24]**:

- **Independiente de la plataforma:** Como ya hemos comentado, su código es ejecutado por la máquina virtual de Java lo que permite que no tenga que ser recompilado para cada plataforma.
- **Orientado a objetos:** Nos permite crear programas modulares y reusar el código escrito.
- **Distribuido:** Java fue diseñado con capacidad de comunicarse por red de forma inherente, de forma que se pueden crear aplicaciones distribuidas y compartir recursos de forma sencilla.
- **Robusto:** A la hora de compilar el código a Java bytecodes, el compilador intenta asegurar al máximo que pueda ser ejecutado sin problemas, detectando los posibles errores que puedan ocurrir en tiempo de ejecución.
- **Multihilo:** Java tiene integrada la habilidad de ejecutar código concurrente, y realizar múltiples tareas de forma simultánea.

- **Dinámico y extensible:** En Java, cada clase está, normalmente, contenida en un único fichero, lo que permite cargarlas en tiempo de ejecución de forma dinámica. Ésto permite crear programas modulares con la habilidad de ser extendidas.
- **Eficiente:** Aunque el código no sea compilado directamente a instrucciones del procesador, Java permite, en la mayoría de los casos ejecutarse con la suficiente eficiencia para minimizar el impacto. Además, gracias al "recolector de basura" evita por completo las fugas de memoria, ya que elimina los objetos que dejan de ser referenciados.

Java se distribuye en diferentes formatos, cada uno orientado a diferentes propósitos y así poder dar mayor soporte **[24]**:

- **Java SE:** Java Standard Edition, es la distribución de propósito general, está orientado al usuario final sin ofrecer recursos enfocados a usos específicos.
- **Java EE:** Java Enterprise Edition está orientado para diseñar software para empresas, ofrece soporte para crear aplicaciones en red y servicios web escalables, fiables y seguros.
- **Java ME:** Java Micro Edition es utilizado para integrar Java en sistemas embebidos que sean poco potentes. Se enfoca en consumir los menos recursos posibles. Fue utilizado sobre todo por teléfonos móviles, pero hoy en día está cayendo en el desuso debido a sus limitaciones.
- **JavaFX:** JavaFX actualmente se encuentra integrado en la octava versión de JavaSE, pero hasta entonces ha sido distribuida de forma separada. Ofrece la posibilidad de crear interfaces gráficas mucho más completas que las que ofrece por defecto JavaSE. Utiliza un esquema modelo-vista-controlador y también tiene incorporado la posibilidad de crear aplicaciones web gráficas.

Además, el entorno de ejecución de Java se distribuye de forma separada al entorno de programación, de forma que cada uno de los formatos vistos es distribuido mediante un JDK (por sus siglas en inglés, Java Develpment Toolkit) y puede ser ejecutado por un JRE (por sus siglas en inglés, Java Runtime Environment) y así el usuario final sólo debe instalar el JRE para poder ejecutar una aplicación Java desarrollada en cualquiera de sus formatos.

Para este proyecto, se ha decidido utilizar JavaSE para desarrollar la aplicación

4.1.3 XML [25]

XML es un lenguaje para la definición de lenguajes de marcado desarrollado por el W3C (World Wide Web Consortium):

"El Lenguaje Extensible de Marcado (XML) describe una clase de objetos llamados documentos XML y parcialmente describe el comportamiento de programas de computador que pueden procesarlos".

Proviene del lenguaje SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML) para estructurar documentos grandes.

Se propone como estándar de intercambio de información estructurada entre plataformas. Usado en BB. DD., editores de texto, hojas de cálculo, ect.

A continuación citaremos algunas de sus características

- Define la sintaxis y los requisitos que deben cumplir los lenguajes de marcado que especifica.
- Marcado: señales añadidas a la información para ayudar a su procesado.
- Extensible: permite definir nuestras propias marcas
- General: cualquier clase de objetos de datos
- Descriptivo: describe datos, pero no qué hacer con ellos
- Implica tener la información estructurada

Gracias a su flexibilidad, XML puede ser utilizado para el intercambio de información estructurada entre diferentes plataformas, y permitir la compatibilidad entre sistemas distintos.

En este documento se utilizará XML principalmente para el diseño de la interfaz gráfica del usuario, como por ejemplo layouts, animaciones etc; y también para las definiciones de las características principales de la aplicación (módulos principales, permisos necesarios, nombre, icono etc.) en el fichero AndroidManifest.

4.1.4 UML [26]

Lenguaje unificado de modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group).

Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados.

La primera versión de lenguaje es la UML 1.1, siendo la última la UML 2.5

4.1.5 Herramientas

En este punto del proyecto se explicaran las herramientas necesarias para llevar a cabo el del desarrollo del subsistema. Sin embargo no son las únicas herramientas existentes sino que existen otras muchas herramientas como alternativa. Se han elegido estas por comodidad.

4.5.1.1 Android Studio

Android Studio **[28]** es un entorno de desarrollo integrado (IDE), basado en IntelliJ IDEA de la compañía JetBrains, que proporciona varias mejoras con respecto al plugin ADT (Android Developer Tools) para Eclipse. Android Studio utiliza una licencia de software libre Apache 2.0, está programado en Java y es multiplataforma.

Fue presentado por Google el 16 de mayo del 2013 en el congreso de desarrolladores Google I/O, con el objetivo de crear un entorno dedicado en exclusiva a la programación de aplicaciones para dispositivos Android.

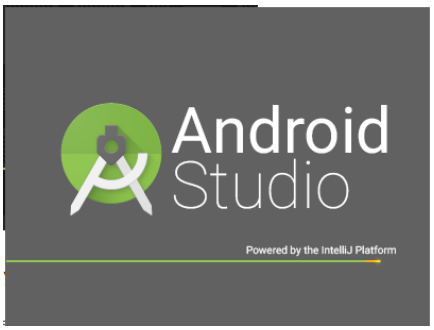

Figura 4.3: IDE Android Studio.

La versión utilizada en este proyecto es 2.1.2, siendo esta la última versión de dicho IDE.

Principales características que incluye Android Studio:

- Soporte para programar aplicaciones para Android Wear (sistema operativo para dispositivos corporales como por ejemplo un reloj).
- Herramientas Lint (detecta código no compatible entre arquitecturas diferentes o código confuso que no es capaz de controlar el compilador) para detectar problemas de rendimiento, usabilidad y compatibilidad de versiones.
- Utiliza ProGuard para optimizar y reducir el código del proyecto al exportar a APK (muy útil para dispositivos de gama baja con limitaciones de memoria interna).
- Integración de la herramienta Gradle encargada de gestionar y automatizar la construcción de proyectos, como pueden ser las tareas de testing, compilación o empaquetado.

A continuación se va a explicar con brevedad los aspectos más relevantes del IDE. Ver figura 4.5.

- **1:** Es el árbol de directorios de la aplicación a desarrollar, donde se muestra la estructura de ficheros del proyecto y las librerías referenciadas.
- **2:** Ésta es la vista principal del área de trabajo, donde se puede editar el código fuente del archivo seleccionado. Mediante el uso de pestañas se facilita el cambio de un archivo a otro.
- **3:** Aquí es donde se puede visualizar la salida estándar y la salida de los errores resultantes de la compilación, instalación y también donde se pueden monitorizar las acciones y estados de nuestra aplicación.
- **4:** Es la barra de herramientas, su posición es fija pero las herramientas de la barra, es configurable, entre ellas destacan las herramientas que permiten compilar/ejecutar/debugear el proyecto.

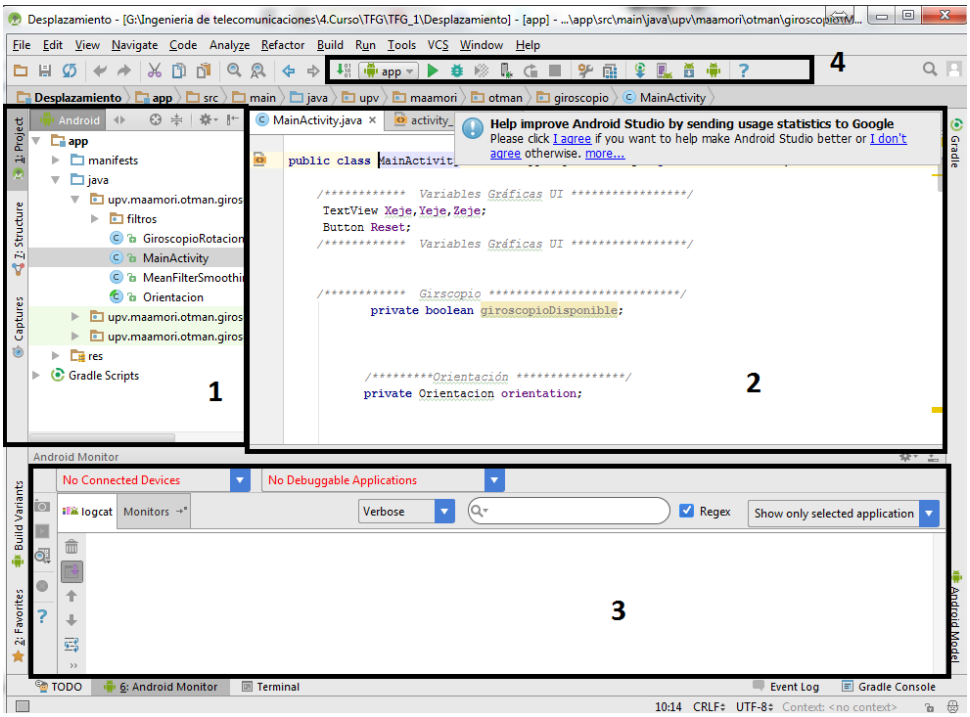

Figura 4.4: Entorno de trabajo Android Studio.

Además de la interfaz anteriormente comentada existe otra interfaz del IDE que nos permite diseñar la interfaz gráfica de nuestra aplicación, ver figura 4.6.

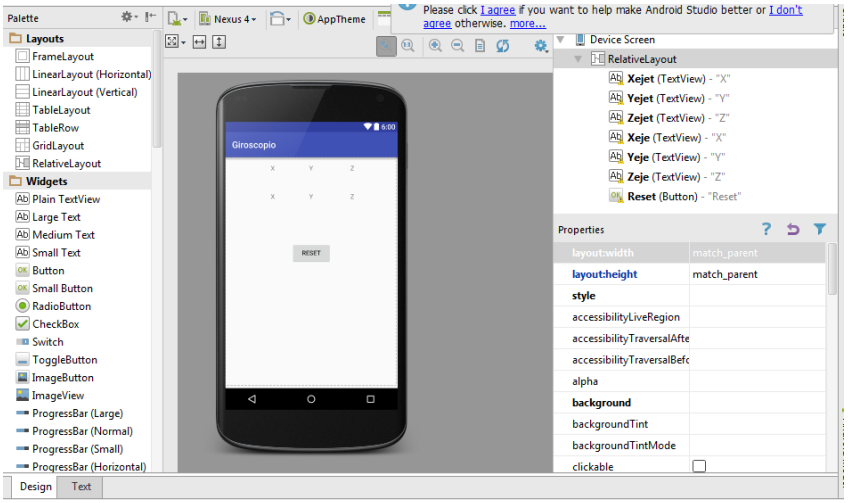

Figura 4.5: Interfaz del IDE para el diseño de la interfaz gráfica de la aplicación.

Podemos decir que el IDE Android Studio es un entorno que facilita el desarrollo de aplicaciones Android. Sin embargo la instalación solo del IDE Android Studio no es suficiente, sino que se necesita una serie de software que permita el desarrollo de aplicaciones. Dicho software es el Kit de desarrollo de software (SDK, siglas en inglés de Software Devlopment kit) y el Java Devlopment Kit (JDK).

En la actualidad, en la página oficial de Android Studio existen dos modos de descarga, uno que integra el SDK y otro sin el SDK, en el segundo caso habría que

descargar el SDK e instalarlo y configurar el IDE para que utilice dicho SDK. El primer caso es el más fácil por lo que es el modo aconsejable.

El SDK de Android, incluye un conjunto de herramientas de desarrollo. Comprende un depurador decódigo, APIs, bibliotecas, un emulador de teléfono basado en QEMU, documentación, ejemplos de código y tutoriales **[27]**.

El SDK está formado principalmente por tres herramientas.

- Androis SDK Tools: permite las construcciones de APKS, depuración, optimización etc.
- Android SDK Platform-tools: permite emular los dispositivos Android con las versiones existentes.
- Android SDK Build-tools: generación de R.java, APKs sin optimizar, conversión de Java bytecode a Dalvik bytecode.

Cada cierto tiempo Google publica una nueva versión de Android, por lo que también publica su correspondiente SDK. Por lo cual hay que instalar estos SDKs para poder trabajar con la nueva versión y poder explotarla al máximo. Para esto existe una herramienta denominada SDK Manager, que nos permite agregar los componentes que deseamos para adaptar nuestro entorno de desarrollo.

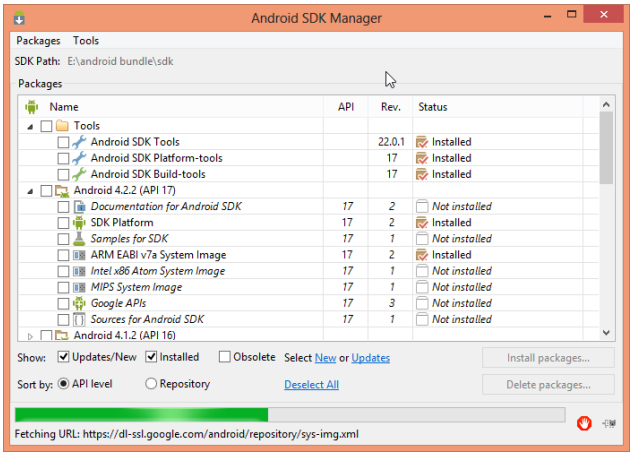

Figura 4.6: Android Manager.

Otro componente que se necesita también integrar en el IDE para poder hacer el desarrollo es el JDK, que es un software que provee herramientas de desarrollo para la creación de programas en Java.

4.1.6 StarUML

StarUML es una herramienta para el modelamiento de software basado en los estándares UML (Unified Modeling Language) y MDA (Model Driven Arquitecture), que en un principio era un producto comercial y que hace cerca de un año paso de ser un proyecto comercial (anteriormente llamado plastic) a uno de licencia abierta GNU/GPL.

Figura 4.7: Logotipo de StarUML.

El software heredó todas las características de la versión comercial y poco a poco ha ido mejorando sus características, entre las cuales se encuentran:

- Diagrama de casos de uso
- Diagrama de clase
- Diagrama de secuencia
- Diagrama de colaboración.
- Diagrama de estados
- Diagrama de actividad.
- Diagrama de componentes
- Diagrama de despliegue.
- Diagrama de composición estructural (UML 2.0)

La capacidad de generar código a partir de los diagramas y viceversa, actualmente funcionando para los lenguajes c++, c# y java. Generar documentación en formatos Word, Excel y PowerPoint sobre los diagramas.

Esta herramienta es utilizada en este proyecto para el modelado de la implementación mediante el lenguaje UML

4.2 Arquitectura de la aplicación

En este punto del trabajo se va a explicar el modo de funcionamiento de la aplicación, su interacción con el usuario (objeto a localizar), la arquitectura interna de la misma así como la arquitectura del sistema del que forma parte.

4.2.1 Funcionamiento

Como se ha comentado en puntos anteriores, la aplicación es un subsistema de posicionamiento indoor, su finalidad es mejorar la precisión del sistema conjunto.

Nuestra aplicación se apoya de unas coordenadas iniciales, estas son ofrecidas por el sistema conjunto, que utiliza la tecnología Wi-Fi. Una vez conocida la posición inicial, la aplicación calcula de forma aproximada la posición del objeto cuando detecta movimiento y dibuja su posición en el mapa del edificio, ver figura 4.9.

En las pruebas y resultados realizados en este proyecto, se elige el punto de inicio manualmente, ya que el otro subsistema no está aún implementado, dicho punto de inicio se elige apretando en el mapa sobre el punto de partida. Ver siguiente figura 4.9.

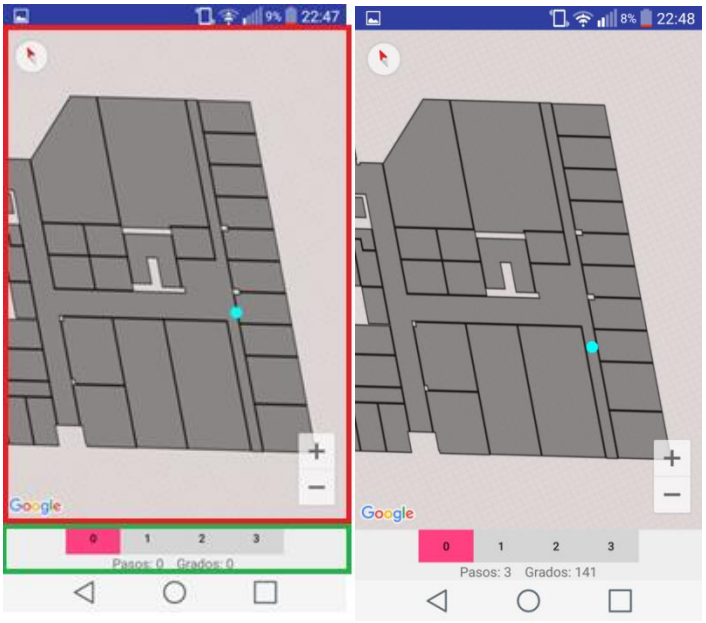

Figura 4.8: Desplazamiento de un objeto.

La figura anterior, representa la interfaz gráfica de la aplicación. En dicha interfaz, podemos diferenciar dos zonas, una para la actualización de la posición (área roja de la figura 4.9 izquierda), que dispone del mapa del edificio y la otra para el control (área verde de la figura 4.9 izquierda). Las funciones de control que permite la aplicación son, el cambio de mapa, esto conlleva el cambio de planta; y la selección del punto de inicio (punto de partida).

Como se puede ver en la figura anterior en la parte de la izquierda, se elige el punto de partida manualmente (en la parte baja de la IU de la aplicación, se refleja el número de pasos, Pasos=0), cuando el objeto, empieza a moverse, la aplicación detecta dicho movimiento mediante los sensores y así calcula su nueva posición. En la parte derecha de la figura, se refleja la posición del objetivo después de dar tres pasos (Pasos=3).

La aplicación, dispone de botones que nos permiten cambiar de planta. En este caso la aplicación se ha desarrollado para el edificio antiguo de teleco de la upv, por lo que habrá cuatro botones. Además de los botones la aplicación dispone de dos campos que nos muestran el número de pasos dados y la dirección de movimiento respecto al norte.

4.2.2 Esquema de la aplicación

En este punto se va a introducir tanto el esquema general del sistema del cual nuestra aplicación forma parte, explicando cual son los elementos que lo forman, su función dentro del sistema así como la relación entre ellos; así como el esquema interno de la aplicación objetivo de este proyecto, explicando también sus elementos y la relación que se desarrolla entre ellos para un buen funcionamiento del subsistema.

El esquema general y básico del sistema conjunto, sería el mostrado en la siguiente figura.

Figura 4.9: Arquitectura del sistema de posicionamiento indoor.

A continuación se explica los elementos que forman dicho sistema:

- **Puntos de Acceso:** Estos pueden ser puntos de acceso o simplemente elementos llamados Beacon, su función, es transmitir la trama beacon que debe recibir el dispositivo inteligente (objeto a localizar), de la cual saca información que se enviará al Centro de Cálculo para la determinación de la posición. En la planificación (ver 2.2.2.4 Implementación Wi-Fi en interiores), es donde se decide el número de estos y su ubicación en la área a cubrir.
- **Dispositivo Inteligente:** Este es el elemento que utiliza para ubicarse, puede ser desde un móvil, Tablet, aparatos electrónicos,etc. En este caso puede ser un móvil, o Tablet con sistema operativo Android, su utilización es para ubicar a personas.
- Es este el entorno de nuestra aplicación. La aplicación se ejecuta sobre éste para utilizar sus sensores y así estimar la posición y de esta forma utilizarla para la actualización de la posición del objeto en el mapa o enviarla al centro de cálculo.
- **BBDD y Centro de Cálculo:** Es aquí donde se guarda el mapa de potencias del escenario y es donde se estima la posición del objeto, comparando las medidas enviadas por el dispositivo inteligente junto con la estimación de este por el sistema inercial.

  Mediante los datos recibidos se consulta la base de datos y se aplican algoritmos de estimación para determinar así la posición del dispositivo inteligente, esta estimación es enviada al elemento anterior para mostrarla a la persona a ubicar.

Además de la anterior explicación básica de la arquitectura del sistema conjunto, como se puede deducir, no es el objetivo de este proyecto, se explica la arquitectura interna de la aplicación, donde se muestra la interacción y la relación existente entre los distintos elementos que la componen.

Como se puede observar en la figura 4.11, la arquitectura de la aplicación a desarrollar es muy simple.

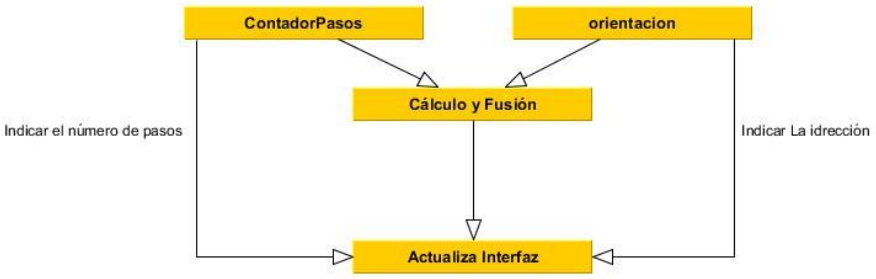

Figura 4.10: Arquitectura interna de la aplicación.

- **orientacion:** Éste elemento de la arquitectura determina la dirección de desplazamiento del dispositivo, obteniendo así el ángulo en grados respecto al norte. Una vez obtenido el ángulo este es enviado al elemento Fusión y Estimación donde es procesado junto con la información del anterior elemento para determinar la posición.

  A su vez este elemento se puede comunicar directamente con la Interfaz de usuario para monitorizar en todo momento el ángulo del dispositivo.

- **Cálculo y Fusión:** Éste elemento se encarga de estimar la posición del dispositivo a partir de los datos entregados por los dos elementos anteriores y las ultimas coordenadas del dispositivo, mediante una serie de cálculos se generan de esta forma las nuevas coordenadas (nueva posición) que serán luego enviadas al último elemento de esta arquitectura.

  Una pequeña aclaración respecto a este elemento, es que si solo se recibe información de solo uno de los elementos anteriores, la posición no queda determinada, es decir, se necesita información de ambos elementos para que este elemento determine la posición actual del objeto.

- **Actualiza Interfaz:** Es aquí donde se implementa la lógica del diseño de la IU, es la encargada de interactuar con el usuario y actualizar la IU. Éste elemento se encarga de recibir las coordenadas del elemento Fusión y Estimación para luego actualizar la posición de la persona en el mapa así como los datos de los dos primeros elementos explicados, para monitorizar en todo momento el número de pasos y la dirección del objeto.

  También aquí es donde se implementa la lógica necesaria para responder a los eventos generados por la persona, tales eventos pueden ser, el cambio de planta, que genera un cambio de mapa en la interfaz, elección del punto de partida.

4.3 Implementación de la aplicación

En este punto se aborda la fase de implementación del proyecto, en concreto la arquitectura interna de la aplicación descrita en el punto anterior, aquí es dónde se emplea el entorno de desarrollo y lenguajes adecuados para llevar a cabo el proyecto. También se citarán las librerías y APIs utilizados.

La explicación se apoyará tanto en trozos de código extraídos como diagramas UML.

4.3.1 Lenguaje, entrono y estructura del proyecto

El lenguaje utilizado para la implementación de la aplicación es Java (Explicado en el punto 4.1.2 Java). Por un lado se optó por este lenguaje por su mínima complejidad y por otro lado porque existen muchos APIs y librerías para llevar a cabo aplicaciones Android que hacen que la implementación sea menos compleja. Existen sin embargo otros lenguajes que pueden ser menos potentes que java como por ejemplo JavaScript, CCS y HTML5 y más potentes como C/C++ ya que este acelera la ejecución de las instrucciones al ser el lenguaje nativo del sistema Android.

El IDE utilizado para la implementación de la aplicación es Android Studio (ya explicado en el punto 4.5.1.1 Android Studio), es la herramienta más adecuada a este proyecto ya que hace que la implementación de este sea menos compleja y fácil de llevar. Sin embargo existen otras herramientas como eclipse etc.

En la siguiente figura se refleja la estructura de directorios de nuestro proyecto, donde se puede ver de forma calara la separación de código por directorios, donde cada directorio corresponde a un elemento de la arquitectura (Figura 4.10: Arquitectura interna de la aplicación.).

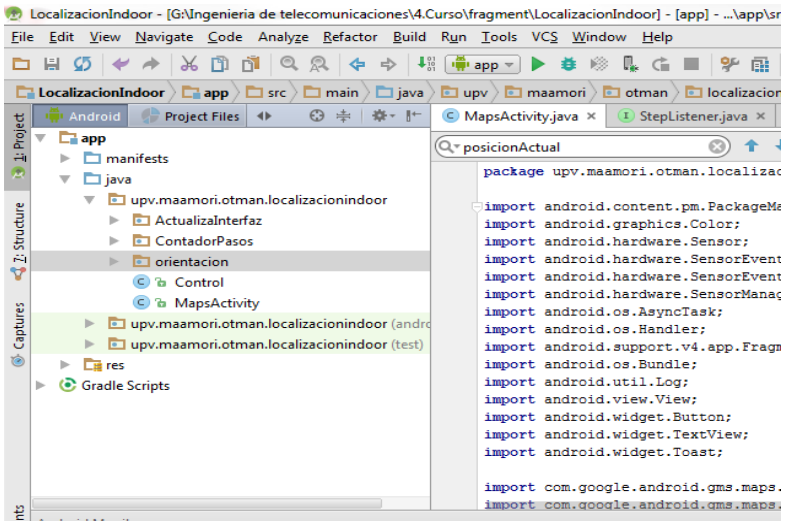

Figura 4.11: Estructura de directorios.

A continuación se describe el contenido de la figura anterior.

- El directorio ContadorPasos implementa las clases (atributos y métodos e interfaces de comunicación) que modelan el funcionamiento del elemento Contador de pasos de la arquitectura interna de la aplicación.

  En la siguiente figura se representan las clases que componen dicho directorio.

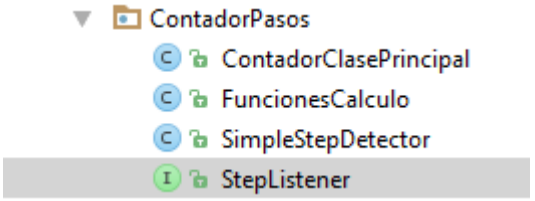

Figura 4.12: Clases Contador de Pasos.

Como podemos apreciar en la figura anterior, este elemento se modela mediante cuatro clases, donde cada una desempeña una función dentro de dicho elemento:

- **ContadorClasePrincipal:** Implementa el sensor acelerómetro para la detección del paso, y enviar datos al elemento Fusión y Cálculo.
- **SimpleStepDector:** Implementa la lógica necesaria para saber si se ha producido un paso, utiliza la clase funcionesCalculo para hacer cálculos matemáticos.
- **FuncionesCalculo:** Implementa las funciones que necesita SimpleStepDector para determinar un paso, tales funciones son por ejemplo la media de los datos del acelerómetro durante un periodo, la norma de un vector etc.
- **StepListener**: Es una interfaz, es implementada por ContadorClasePrincipal.

- El directorio orientación implementa las clases (atributos y métodos e interfaces de comunicación) que modelan el funcionamiento del elemento Giroscopio, es decir el cálculo de la orientación del objeto.

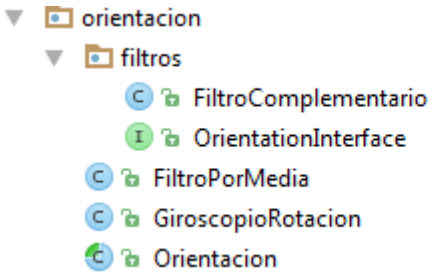

Figura 4.13: Clases Giroscopio (orientacion).

Este elemento de la arquitectura se modela mediante cinco clases, ver la figura 41, cada clase desempeña una función dentro de dicho elemento.

- El directorio ActualizaInterfaz implementa las funciones que modelan el funcionamiento del elemento Fusión y Estimación.

  Este directorio está compuesto por una serie de clases y librerías ofrecidas por el API google map para el manejo de mapas. Su función principal es calcular las nuevas coordenadas a partir del ángulo de orientación, coordenadas actuales y la distancia recorrida.

  Como se puede ver en la siguiente figura, dicho elemento o directorio está formado por cuatro clases.

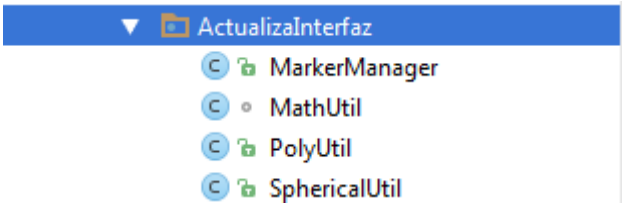

Figura 4.14: Clases ActualizaInterfaz.

- Control y MapsActivity son las clases que implementan funciones que gestionan la actividad, funciones para el lanzamiento de hilos, la actualización y la gestión de la interfaz de usuario, así como gestores de eventos producidos por los actores que pueden ser tanto el usuario como alguno de los elementos de la arquitectura anteriormente descrita

Más adelante se explicarán con más profundidad las clases que forman esta estructura.

Una vez explicada la estructura de la lógica de cálculo, manejo y gestión de la aplicación, ahora se explicara la estructura del diseño de la interfaz usuario.

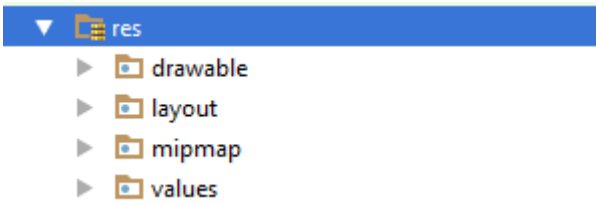

Figura 4.15: Estructura de la IU.

La estructura de la IU en este proyecto se compone prácticamente de tres directorios fundamentales, siendo estos los dos primeros y el último directorios del directorio res, tal y como se muestra en la figura anterior.

A continuación se explicaran las funciones básicas de cada uno y de sus componentes

- Drawable: En este proyecto es el directorio donde se ubican los planos de las distintas plantas del edificio, así como el icono de nuestra aplicación. Ver figura 44.

- Layout: Es aquí donde se define el diseño gráfico de cada una de las actividades de la aplicación, el fichero que define el diseño es un fichero XML.

  En este caso solo existe una única actividad dividida en dos fragmentos. El primero fragmento representa el mapa y los botones de control de la misma, mientras que el segundo representa los botones que permiten el cambio de planta y la visualización de los pasos dados y la orientación. Por lo que este documento está formado por dos ficheros XML siendo estos; activity_maps y control. Ver figura 4.17.

A cada fichero XML le corresponde una clase para su manejo, el primero es manejado con la clase MapsActivity y el segundo con la clase Control.

- Google_maps_api: En este fichero se encuentra una clave que identifica nuestra aplicación frente al servicio de google maps. Sin esta clave el mapa de nuestra aplicación no funcionaría. Con esta clave google maps identifica las aplicaciones y establece el límite de uso del API maps. Ver figura 44.

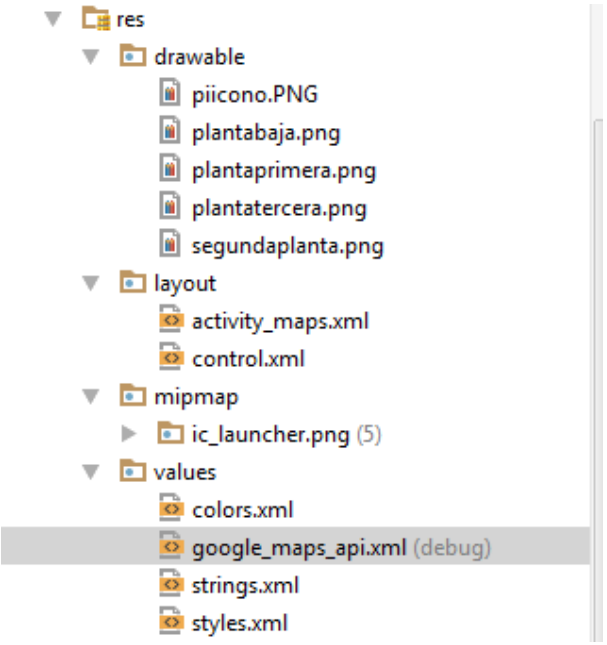

Figura 4.16: Contenido de la estructura UI.

4.3.2 Diagrama de clases UML

En esta sección se explicará cómo se ha implementado la aplicación siguiendo el esquema de bloques de la arquitectura interna. Este punto es una explicación profunda del contenido, relaciones y comunicaciones de las distintas clases descritas en el punto anterior.

4.2.3.1 Diagrama de clases reducido

Con el diagrama de clases reducido se obtendrá una visión del conjunto de clases de la aplicación, en el que agruparemos dichas clases en bloques, dónde cada bloque representa a un elemento de la arquitectura interna de la aplicación.

Se puede observar en la figura siguiente (Figura 4.17: Diagrama de clases reducido) que algunas clases, abarcan varios bloques. Se ha decidido realizarlo de esta manera para simplificar la interacción entre clases y la organización del código.

En la misma figura se refleja la relación entre las distintas clases, la agrupación de las clases ha sido tal que cada grupo realiza una tarea concreta dentro de la aplicación. A continuación se hará una breve explicación de cada agrupación o bloque.

1 : En este bloque de la agrupación podemos ver las clases que se encargan de dibujar el mapa y la interfaz de usuario. También es donde se localiza la clase principal que gestiona toda la aplicación (MapActivity). Podemos observar las relaciones existentes entre las clases según la sintaxis UML.

2 : Este bloque muestra las clases que llevan a cabo las tareas para detectar los pasos así como la relación existente entre ellas. La clase ContadorClasePrinipal es la que instancia todas las otras clases y es la misma que se encarga de comunicarse con el bloque 1.

3 : Éste bloque nos muestra la relación entre las clases que permiten el cálculo del orientación del objeto, es decir el ángulo del dispositivo respecto al norte. La clase que se comunica con el bloque 1 es la clase Orientación, y la que instancia todo el resto de clases para llevar a cabo los cálculos.

4 : En este bloque se refleja la relación existente entre las clases que calculan la posición nueva del objeto y dibujar dicha posición en el mapa. Estas clases son facilitadas por el API del google map, por lo que facilita bastante la implementación.

5 : En este bloque se muestra la relación existente entre las clases que permiten atender a los eventos generados por el usuario, tal como el cambio de piso, elección del punto de partida, visualización del número de pasos y la dirección de movimiento. En esta agrupación es necesaria la intervención de la clase MapActivity.

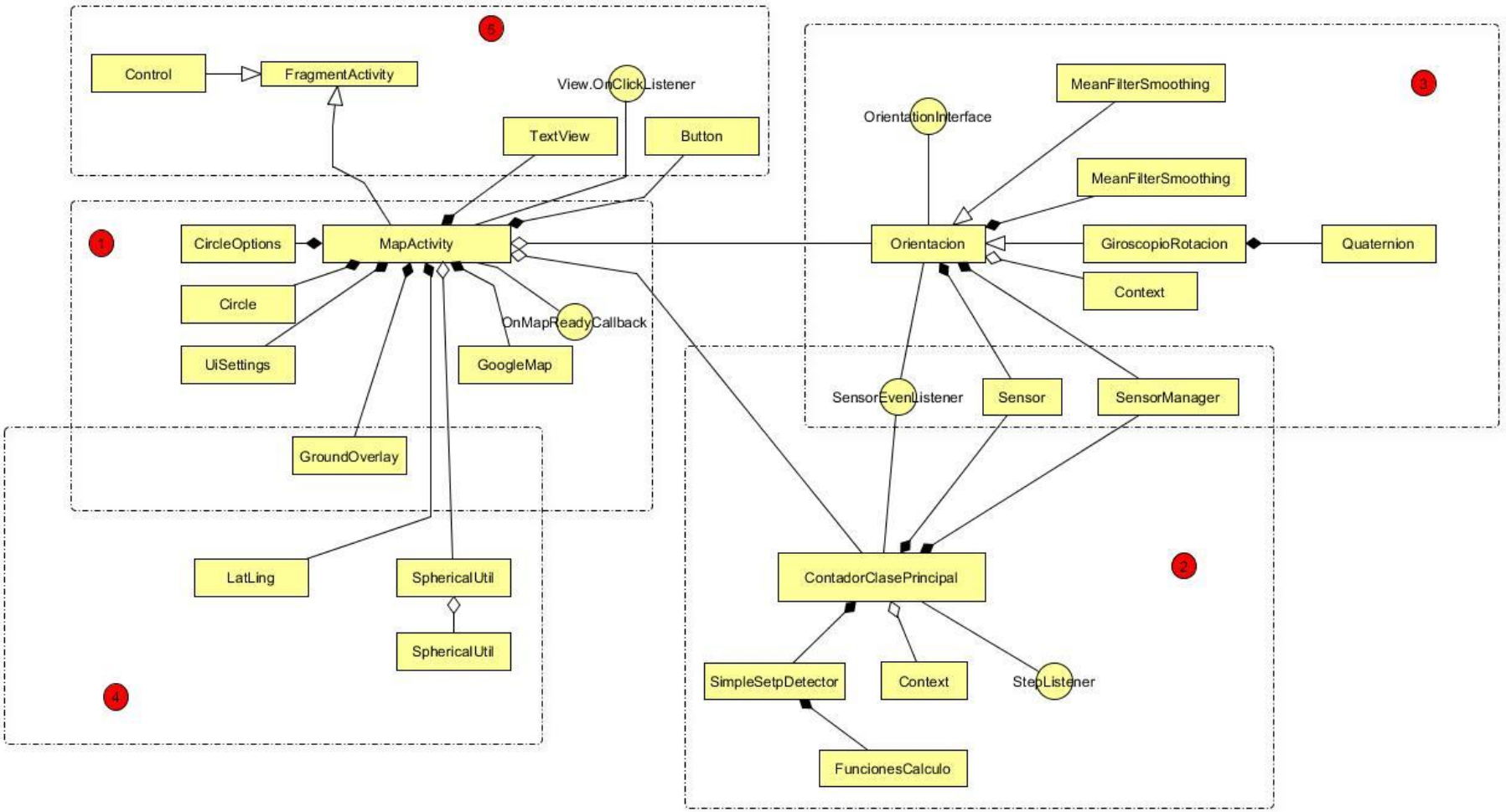

Figura 4.17: Diagrama de clases reducido

4.2.3.2 Diagrama de clases UML expandido

A continuación se detallan las distintas clases vistas en el diagrama de clases reducidas, mostrando así los métodos y atributos de cada una de ellas y agrupándolas según sus respectivos bloques.

1 y 5

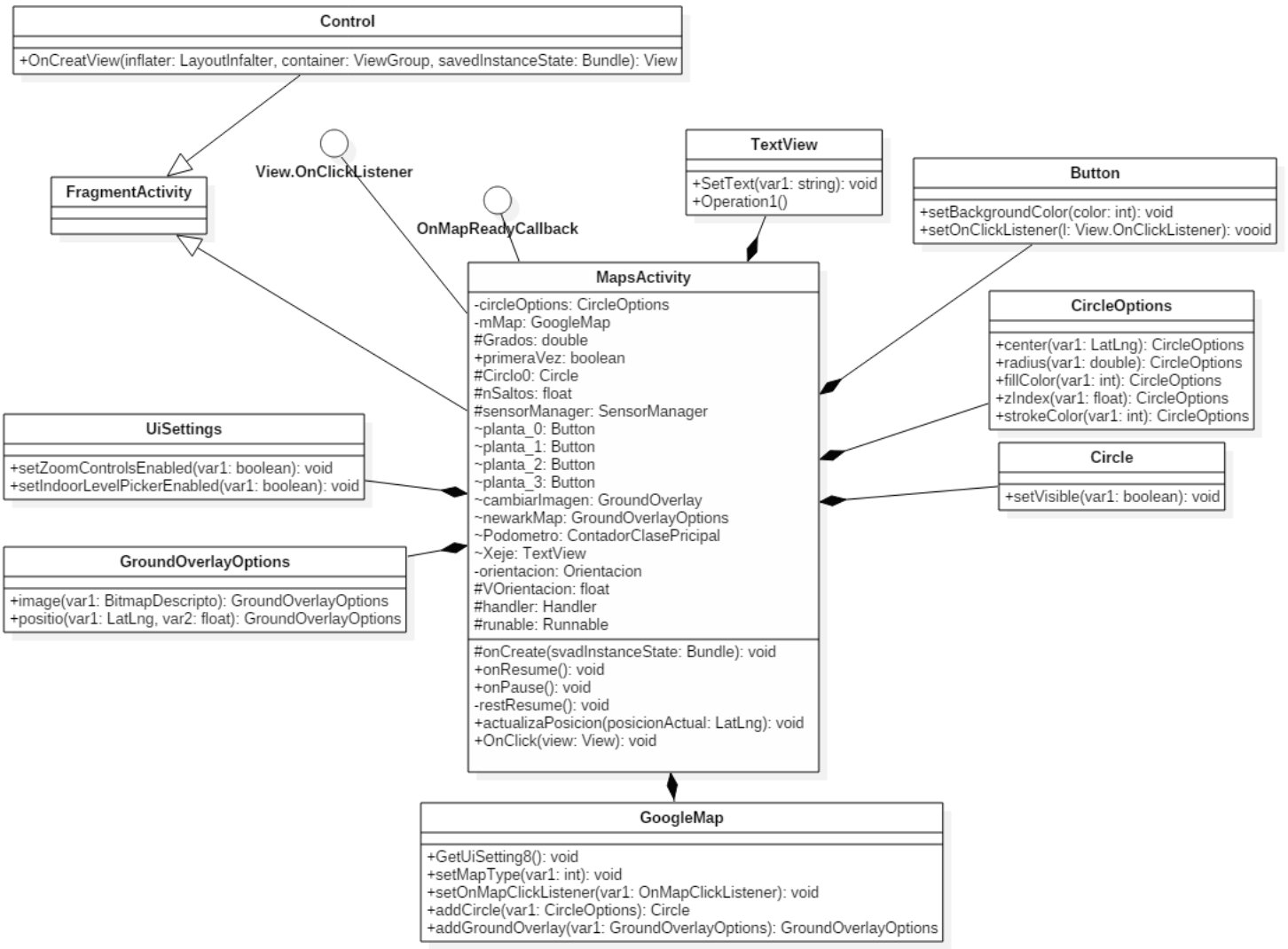

Figura 4.18: Diagrama de clases primer y quinto bloque.

- MapsAcitivity es la clase principal de la aplicación, es la encargada de comunicarse con las otras clases de los distintos bloques anteriormente explicados. La mayoría de los atributos y métodos que implementa son de otras clases padre, facilitadas por la plataforma Android.

Las funciones que esta clase realiza son:

- Cargar, gestionar y actualizar la IU de la aplicación así como su ciclo de vida.
- Cargar, gestionar y actualizar los distintos mapas del edificio y ubicar el punto de partida del objeto.
- Comunicarse con los distintos bloques explicados en el punto anterior para obtener datos relevantes a la ubicación.
- Permite responder a eventos generados por el usuario mediante la implementación de la interfaz View.OnCliclListener y OnMapReadyCallback.

- Control: Esta clase representa el otro fragmento de la interfaz de usuario donde se dibujan los cuatro botones para el cambio de plano del edificio, así como los textos para mostrar el número de pasos y el ángulo.

Estos dos bloques como se puede ver en la figura anterior contienen muchas clases facilitadas por Android y que son instanciadas principalmente por la clase MapsActivity para llevar a cabo su funciones anteriormente descritas. Por ello a la hora de la descripción de estas clases con UML, se ha limitado a nombrar solo aquellos métodos y atributos utilizados en este proyecto.

A continuación se comentado de forma breve el uso de cada una de las clases:

- FragmentActivity: Es una interfaz implementada por la clase Control y MapsActivity para la división de la actividad en fragmentos.
- UiSttings: Añade funcionalidades a la aplicación para el tema de manejo de mapa, tales como el zoom, el cambio de cámara etc.
- CircleOptions: Ofrece los métodos para la construcción de objetos de figuras, para su dibujo en el mapa. En este proyecto se utiliza para dibujar la figura círculo para indicar el punto de partida.
- GoogleMap: Ofrece los métodos para el manejo de mapas, para su carga en la aplicación, para el dibujo de figuras sobre mapas, así como manejadores de eventos tales como detección de pulsación de una determinada zona del mapa.
- Button: Se utiliza para el manejo de botones.
- TextView: Se utiliza para el manejo de texto indicadores
- View.OnClickListener: Es una interfaz implementada por la clase MapsActivity para atender a los eventos generados por acciones sobre los botones.
- OnMapReadyCallback: Es una interfaz implementada por la clase MapsActivity para la detección de pulsación sobre algún punto del mapa.

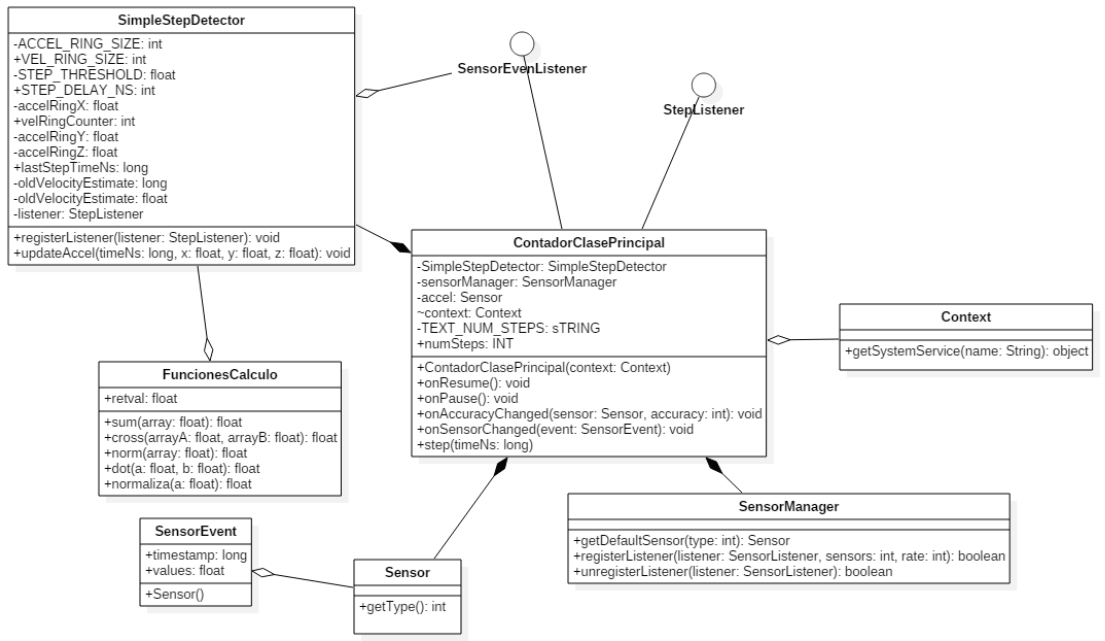

Figura 4.19: Diagrama de clases segundo bloque.

En la figura anterior se reflejan de forma detallada las distintas clases que componen el bloque dos, que corresponde con el bloque ContadorPasos de la arquitectura de la aplicación (Figura 4.10: Arquitectura interna de la aplicación.).

Como sabemos de otros puntos anteriores, este bloque se encarga de calcular los pasos del objeto a ubicar. A continuación procederemos a explicar las clases que forman este bloque.

- ContadorClasePrincipal: Esta clase implementa una interface para el manejo de los sensores, en concreto el Acelerómetro, dicha interfaz es la SensorEvenListener, así como otras clases para la interacción con este. Además de lo anterior, esta clase se basa en el resto de clases para el cálculo y para la detección del paso. Otra de sus funciones es la comunicación con la actividad principal (MapActivity) para facilitar los datos del paso.
- SimpleStepDetector: Esta clase simplemente sirve para utilizar los métodos que ofrece la clase FuncionesCalculo para hacer los cálculos necesarios para la detección del paso.
- FuncionesCalculo: En esta clase es donde se ubican todos los métodos necesarios para hacer los cálculos, como por ejemplo sacar la norma de un vector, la media etc.

El resto de las clases son facilitadas por la plataforma Android, por lo que a la hora de su explicación mediante UML solo se han citado los métodos que son utilizados en este proyecto. A continuación se explica de forma breve la funcionalidad de cada una.

- SensorEvent: Se utiliza para la obtención de los valores de la aceleración (en este caso) de los diferentes ejes del acelerómetro así como la marca de tiempo de cuando se ha producido la obtención.
- Sensor: Se utiliza para la elección del tipo de sensor así como la definición de algunos parámetros de los mismos.
- SensorManager: Nos permite instanciar lo sensores disponibles en el dispositivo para poder trabajar en ellos.
- Context: Como la MapsActivity es la única clase que extiende la clase activity, por la tanto es la única que puede instanciar las sensores en este caso. Entonces para dotar las otros elementos de esta capacidad, se extiende esta último a los otros elementos mediante la clase Context, de esta forma las otras clases pueden instanciar los sensores.

El resto de clases es explicado en el bloque anterior.

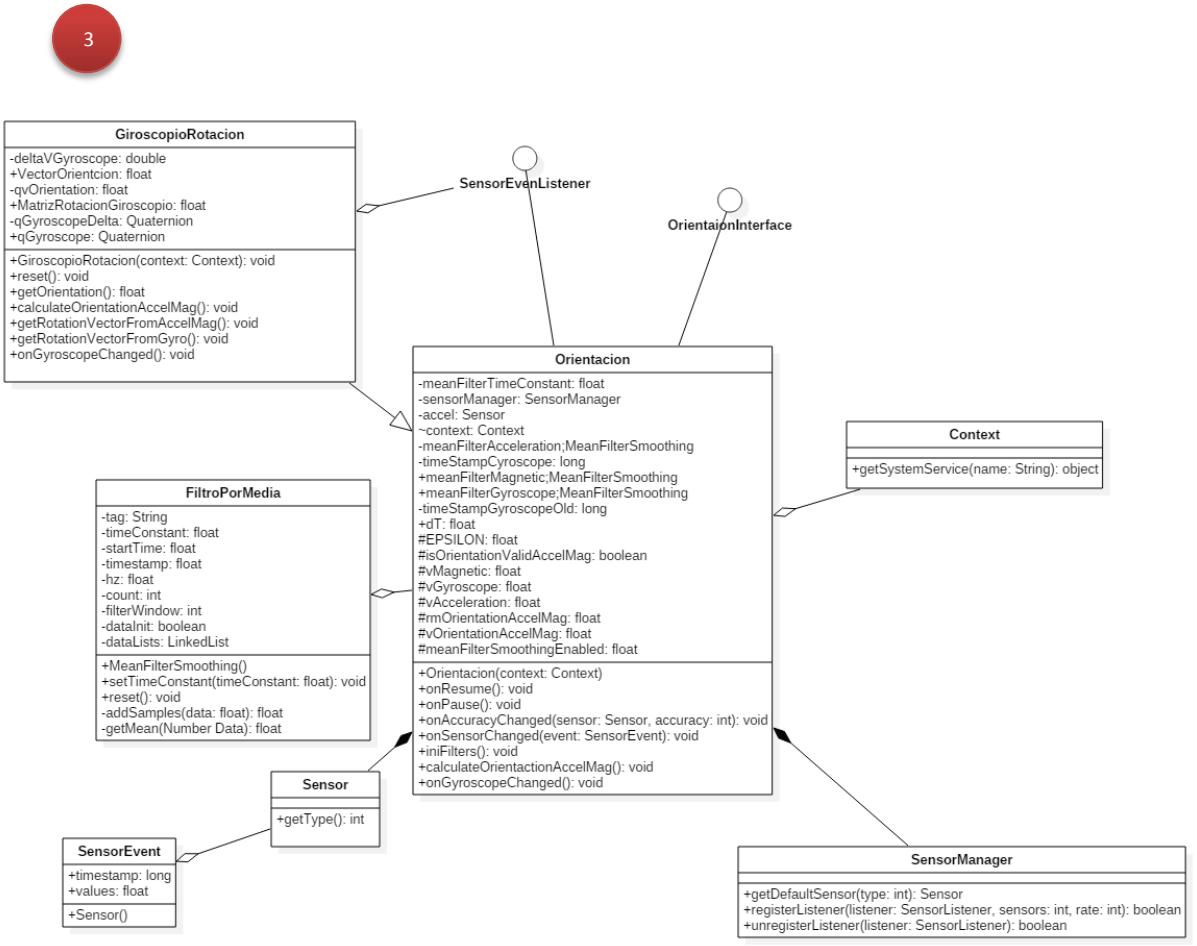

Figura 4.20: Diagrama de clases cuarto bloque.

Este bloque corresponde al bloque Giroscopio de la arquitectura interna de la aplicación (Figura 4.10: Arquitectura interna de la aplicación.). En dicho bloque reside la funcionalidad para la obtención de la orientación del objeto respecto al norte. A continuación se explicarán de forma detallada las clases que lo componen.

- Orientación: Esta clase implementa una interfaz como otras clases para el manejo de sensores del dispositivo, concretamente para el manejo del Acelerómetro, Giroscopio y el Magnetómetro. La clase usa otras clases como MeanFilterSmoothing para el filtrado de las medidas de los distintos sensores para así obtener una precisión alta así como la clase GiroscopioRotacion para el cálculo de la orientación.
- MeanFilterSmoothing: Esta clase implementa un filtro paso alto para la eliminación del ruido de las medidas obtenidas de los sensores y poder así el sistema brindar una precisión alta.
- GiroscopioRotacion: En esta clase es donde se implementan las funciones para el cálculo de la orientación así como métodos para la fusión de los datos de los tres sensores para la mejora de la precesión. La mayoría de las funciones y librerías de esta clase son facilitados por la plataforma Android.

El resto de las clases ha sido explicado en bloque anterior.

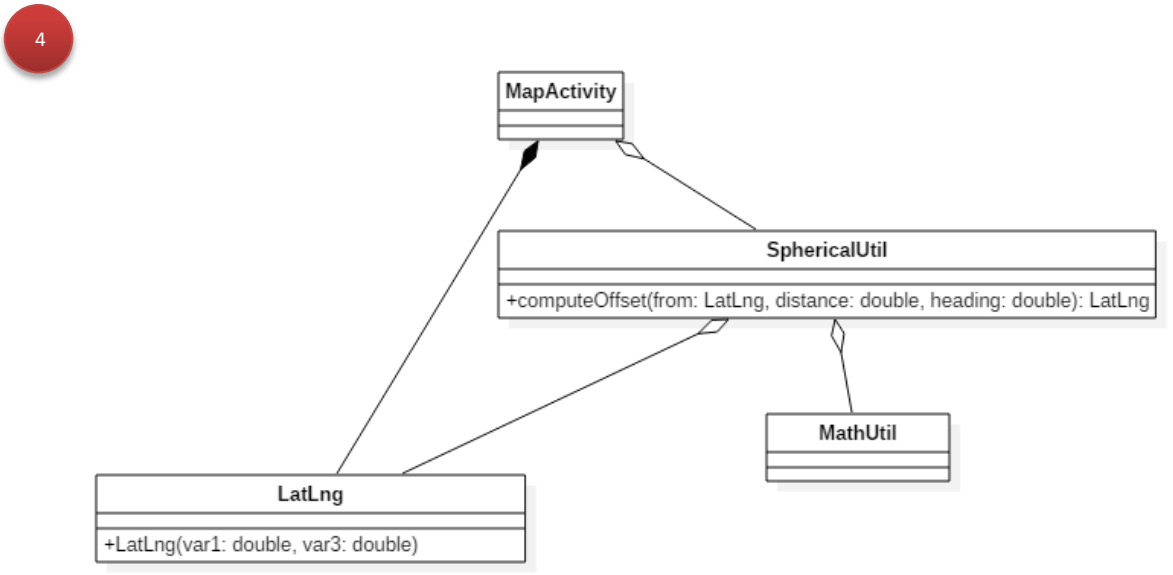

Figura 4.21: Diagrama de clase del bloque 4.

En este bloque podemos decir que la mayoría de las clases que lo forman son clases facilitadas por GoogleMaps. A continuación se hará una pequeña descripción de cada una de ellas.

- SphericalUtil: Esta clase se encarga de generar las nuevas coordenadas del objeto a partir de las coordenadas anteriores, la distancia desplazada y el ángulo. Esta clase es utilizada prácticamente para la actualización de la posición del objeto.
- LatLang: Esta clase es utilizada para la formación de las coordenadas Longitud y la Latitud del objeto.
-

4.3.3 Ciclo de vida de la aplicación

La aplicación emplea un hilo para cada bloque de su arquitectura interna, salvo para para los elementos Fusión y Estimación y la Actualiza Interfaz que se llevan a cabo sobre el mismo hilo. Todo ello es para realizar las distintas tareas de forma concurrente. El empleo de hilos en esta arquitectura es necesario ya que aceleran el proceso de posicionamiento al haber procesos de cálculo pesados y evitan errores en la aplicación.

En la siguiente figura se refleja el número de hilos que emplea nuestra aplicación, así como las tareas y la secuencia de acciones que ejecutadas por cada uno.

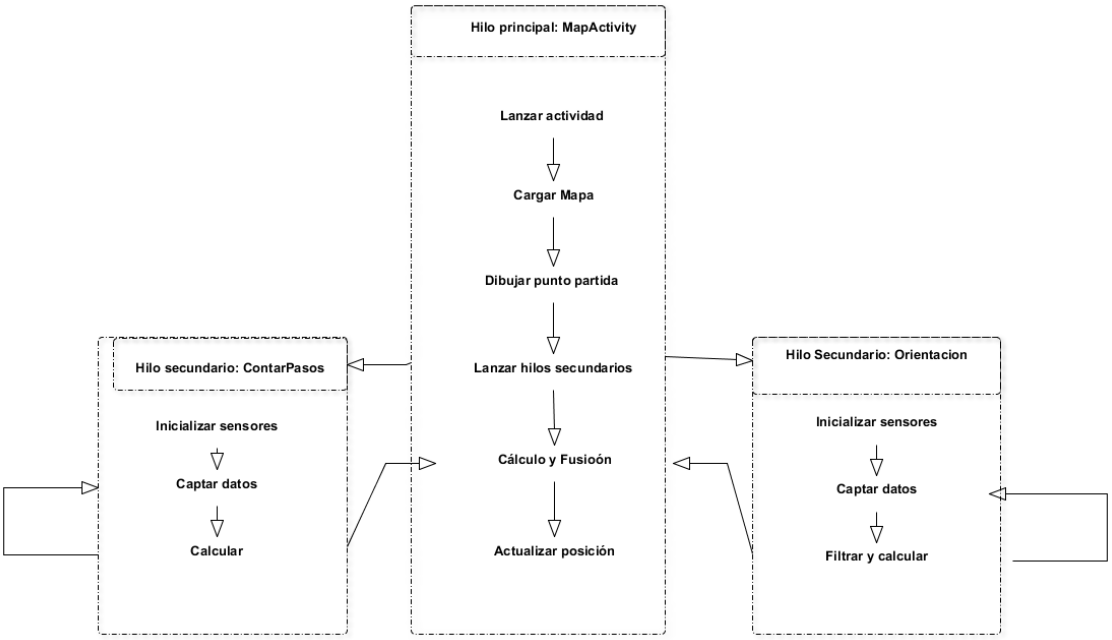

Figura 4.22: Hilos de la aplicación.

Hilo principal

Este hilo se encarga de lanzar la única actividad de la aplicación, y gestionar la interfaz de usuario y así mismo cargar el mapa del edificio y la interfaz de control. Una vez hecho lo anterior espera la pulsación del usuario para la elección del punto de partida, una vez elegido el punto lanza los dos otros hilos y se pone a esperar datos de los otros sensores para la actualización de la posición

Hilo secundario: Orientación

Este hilo es iniciado a partir del hilo principal, es utilizado para llevar a cabo las funciones de cálculo de la orientación. Primero se instancian los sensores giroscopio, acelerómetro y magnetómetro y se captan los datos cuando el giro es detectado, una vez

detectado se filtran los datos y se calcula la posición, que será comunicada en cada cálculo al hilo principal para la actualización de la posición.

Hilo secundario: ContarPasos

Se inicia a partir del hilo principal, es utilizado para llevar a cabo las funciones de cálculo del paso. Primero se instancia el sensor acelerómetro y luego se captan los datos cuando el usuario se mueva, una vez detectado el movimiento se filtran los datos y se estima si es un paso, si es así se le indica al hilo principal para que este determine la posición.

Capítulo 5. Pruebas de ejecución y resultados

En este punto del proyecto, se van explicar las pruebas realizadas sobre el subsistema (la aplicación) para evaluar su funcionamiento, y ver así, si es viable. Pero en primera posición se explican los elementos que han intervenido para la realización de las pruebas, como el dispositivos inteligentes utilizados, el escenario de la prueba etc.

Para la evaluación, se ha decidido evaluar separadamente cada elemento del subsistema (elementos de la arquitectura) para saber cuál elemento del mismo es el responsable de imprecisión, y una vez evaluados estos, evaluar el subsistema total.

5.1 Entorno de las pruebas

Los elementos intervinientes en las pruebas son los siguientes:

- **Dispositivo inteligente:** Es un dispositivo móvil, de marca LG, modelo G3, con una memoria interna de 16 GB, procesador Quad-Core 2.5 Ghz, Ram de 2 GB, pantalla de 5.5 pulgadas y con la versión 5.0 de Android correspondiente al API 21. La aplicación ha sido desarrollada con el API 15, por lo que solo es instalable en versiones de Android iguales o superiores a la 4.0.3.
- **Edificio:** El edificio donde se han realizado las distintas pruebas, es el edificio 4D de la etsit de la upv, concretamente en la segunda planta. Se ha elegido este por cercanía al lugar de trabajo.

5.2 Detector de pasos

El detector de paso es el que se encarga de determinar la distancia recorrida por la persona, por lo que su precisión juega un papel importante en este proyecto. A continuación se va a proceder a evaluar su funcionamiento.

Para la evaluación se ha decidido comparar los pasos reales con los que calcula la aplicación, para distintos escenarios.

Escenario 1: caminar en línea recta, para distintos números de pasos.

Escenario 2: caminar en líneas curvas, para distintos números de pasos.

Tabla 5.1: Escenario 1 Contador Pasos.

Pasos reales	Pasos calculados	Error en metros
4	6	0.8
4	6	0.8
4	5	0.4
4	8	1.8
10	14	1.6
10	13	1.2
10	16	2.4

La tabla anterior refleja la evaluación del contador de pasos en el escenario 1. Como se puede ver en la misma tabla, el error en metros más alto que presenta la aplicación es de 2.4 m. El error anterior es considerable pero es bastante menor que el presentado por la solución 1, dónde se calcula el desplazamiento con la doble integral de la aceleración obtenida del acelerómetro. El error básicamente es debido a la sensibilidad que presenta el acelerómetro a los movimientos y al ruido, una buena corrección puede ser el subir el umbral de paso.

Tabla 5.2: Escenario 2 Contador Pasos

Pasos reales	Pasos calculados	Error en metros
4	8	1.6
4	8	1.6
4	7	1.2
4	7	1.2
10	13	1.2
10	12	0.8
10	15	2.0

En la tabla anterior se reflejan los resultados del escenario 2, como se puede ver se obtiene casi el mismo error que en el caso anterior. El error es pequeño pero considerable, su eliminación puede ser llevada a cabo mediante el cambio del umbral para la detección, igual que en el caso anterior.

5.3 Orientación

Para la comprobación de la viabilidad del cálculo de la orientación se han realizado los dos escenarios comentados en el punto "3.1 Primera solución", donde se obtuvo la Tabla 3.2: Evaluación del giroscopio. y se pudo comprobar que el sistema presentaba un error insignificante, haciendo que esta implementación sea viable.

5.4 Aplicación

En los puntos anteriores, se ha evaluado cada uno de los elementos de la arquitectura por separado, se hizo de esta forma para poder hacer una evaluación con profundidad y así conocer los errores que añade cada bloque a la aplicación. En este punto se va a proceder a evaluar la funcionalidad completa de la aplicación, es decir la evaluación de los bloques anteriores de forma conjunta. Para ello se van a definir varios escenarios.

Escenario 1: esta evaluación se realiza en la planta baja del edificio viejo de la etsit de la upv y consiste en ir de un punto de origen a un punto destino, en la siguiente figura se muestra el camino a recorrer entre dichos puntos.

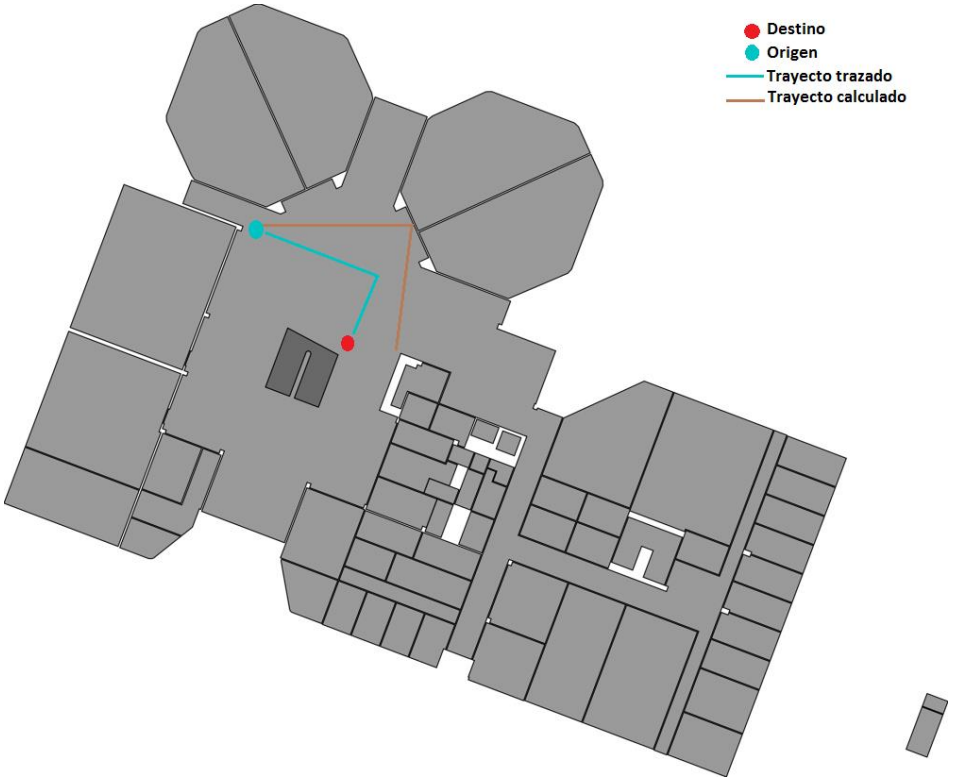

Figura 5.1: Escenario 1 Aplicación.

Para la realización de escenario 1, se elige el punto de partida y se procede a seguir el recorrido trazado en la figura anterior. En la misma figura se puede observar el camino que la aplicación ha calculado (camino de seguimiento del individuo). Como se puede observar, el trayecto calculado presenta cierto error respecto al trayecto trazado, dicho error es debido al error acumulativo generado por el cálculo de la orientación. El error final entre el punto destino y el calculado por la aplicación es de un 1.5 m.

Escenario 2: la evaluación se realiza en la misma planta que el escenario 1, con la misma metodología de evaluación, solo que en este caso se recorre un camino con cierto número de cambios de dirección, en la figura siguiente se presenta el camino trazado.

Figura 5.2: Escenario 2 Aplicación.

En el segundo escenario se ha intentado girar el dispositivo cierto grado para corregir así el error obtenido en el escenario 1, el resultado de recorrer el camino trazado es el presentado en la figura anterior. Como se puede observar existe un error tanto en la orientación como la distancia recorrida, por lo tanto para su corrección se necesita lo mismo que en el caso anterior.

Capítulo 6. Conclusiones y líneas futuras

6.1 Conclusiones

El objetivo de este proyecto como se ha explicado en el apartado de introducción (1.2 Objetivos) es hacer una introducción de las tecnologías de posicionamiento indoor y el diseño de una aplicación indoor basada en sistema inercial. En este trabajo no se han podido cubrir todas las tecnologías existentes, algoritmos y técnicas de estimación de posición debido al tiempo y la extensión del trabajo.

Como se pudo deducir del trabajo, existe una variedad de tecnologías que disponemos y que nos rodean y que tal vez desconozcamos su utilidad, siendo su utilidad un recurso que nos pueda facilitar, mejorar y acomodar la vida. Se ha podido concluir que estos sistemas de posicionamiento indoor, su coste es mínimo, al aprovechar dichas tecnologías.

Respecto a la precisión de las tecnologías utilizadas se ha concluido que estas presentan una precisión baja, limitando así el número de aplicación que se les puede dar, sin embargo para mejorar dicha precisión se ha concluido que una buena solución puede ser la utilización de sistemas híbridos, en los que se utiliza más de una tecnología a la vez, la mejora en estos sistemas es considerable.

Respecto a la aplicación, se concluye que responde al objetivo planteado en el proyecto, no obstante, lo hace con cierta imprecisión, que se puede considerar aceptable. Dicha imprecisión es debida al error generado por los sensores utilizados, este error se debe básicamente a la baja gama de sensores integrados en los dispositivos inteligentes. Por lo que para una buena precisión sería necesaria la implementación de filtros que eliminan el ruido generado por los sensores, los errores de la doble integración y el offset que estos dispositivos presentan. Sin embargo el filtrado tiene una complejidad tanto a nivel de programación como a nivel de ejecución (hardware), así como un coste temporal, todos ellos pueden hacer que la aplicación sea inviable. Por lo que la precisión no solo depende de los sensores utilizados, sino también del hardware y software que estos dispositivos tienen.

Se ha concluidos que estos sistemas tienen una gran repercusión sobre el consume de la batería, sobre todo cuando se utilicen sistemas híbridos, como es el caso del proyecto de investigación, que utilizará los sensores y la red Wi-Fi. El consumo de la batería es un factor importante a tener en cuenta, ya que por ejemplo en el caso de los sensores aunque la aplicación pase a segundo plano, el sensor sigue funcionando, lo que hace que el consumo de la batería aumente. Una buena manera de resolver esto es mediante una implementación inteligente, que inhabilite los sensores cuando estos no son usados.

6.2 Líneas futuras

En líneas futuras procederemos a citar las mejores que se pueden hacer a este proyecto tanto a nivel de descripción de las tecnologías de posicionamiento utilizadas como las que están en proceso de investigación; como el diseño de la aplicación.

Capítulo 7. Bibliografía

Respecto a las tecnologías que se utilizan, como se ha podido notar, no han sido todas citadas y explicadas en este proyecto, por lo que una posible mejora, puede ser la investigación de las nuevas tecnologías usadas.

Algunas de las tecnología a tratar en líneas futuras son las femtoceldas utilizadas por redes móviles, GPS pseuloditos, esta tecnología consiste en la creación de una constelación en la zona indoor, sin embargo presenta un coste elevado, que limita su uso solo en zonas industriales y finalmente la tecnología que utiliza Led, usando la cámara del dispositivo.

Las mejoras que se introducirían a la aplicación diseñada serian básicamente a nivel de filtrado en primer lugar, para así eliminar los errores generados por los sensores y poder así brindar una precisión alta.

El filtro que se desearía implementar ya que es el más sofisticado para este tipo de problemas es el filtro Kalman, aunque este presenta una gran complejidad matemática que requiera tanto de hardware para reducir el coste de procesamiento como conocimiento matemáticas para dicha implementación. El filtro Kalman calcula el error de cada medida a partir de las medidas anteriores, eliminarlo y dar el valor real del ángulo. En cierto modo es un algoritmo que aprende en cada iteración.

Otra de las mejoras es mejorar el diseño de la interfaz de usuario y brindarle al usuario una aplicación de mucha comodidad, una de las comodidades puede ser la programación de la aplicación para que gire el mapa cada vez que el usuario gire el móvil, de esta forma se le brinda una comodidad alta al usuario final, ya que se evitaría girarla manualmente.

Una mejora destacable y grande a nivel de localización seria la detección de cambio de piso, es decir identificar que el individuo está subiendo o bajando por una escalera o ascensor y cambiar de forma automática el mapa en visualización de la planta y dibujar su punto de partida automáticamente en la nueva planta, valiéndose en la posición última del individuo en el planta anterior. Esto se puede conseguir mediante la monitorización de la salida del acelerómetro, concretamente en el eje z (el eje perpendicular a la pantalla del dispositivo).

Otras mejoras serían tanto a nivel de consumo de la batería, alargando la vida de esta como a nivel de interfaz de usuario para brindarle al usuario una aplicación sencilla y fácil de manejar automatizando una serie de rutinas.

Capítulo 7. Bibliografía

[1]: Lorenzo Rubio Arjona y Francisco Ramos Pascual.” Mecanismos de propagación radioeléctrica”. Páginas: 22-25.

https://www.dropbox.com/s/n2la04j80pj1tkp/Tema%202_1.pdf?dl=0

[2]: Lorenzo Rubio Arjona y Francisco Ramos Pascual. ”Modelos de la propagación radioeléctrica”. Páginas: 53-59.

https://www.dropbox.com/s/91grmardsejsrzb/Tema%203%20Modelado%20propagacion.pdf?dl=0

[3]: Jeison Marín Alfonso. “Modelo de propagación empírico para predicción de pérdidas de potencia en señales inalámbricas”.

https://www.dropbox.com/s/7wvv6c0lr8kt4iy/771-2338-1-PB.pdf?dl=0

[4]: Francisco Javier Coret Gorgonio. “Tesis Francisco Javier Coret Gorgonio año 2015”. Páginas 29-32.

https://www.dropbox.com/s/ooblv1tpm9kme49/Tesis_Francisco_Javier_Coret_Gorgonio_2015.pdf?dl=0

[5]: Eva Lagunas Targarona.”Estimación conjunta de TOA y DOA en sistemas UWB para localización”. Páginas: 29-31.

http://upcommons.upc.edu/bitstream/handle/2099.1/7325/PFC_EvaLagunas.pdf

[6]: Pablo Suazo Fernández. ”Desarrollo de una aplicación de posicionamiento indoor”. Páginas: 26-28

http://es.slideshare.net/peterbuck/desarrollo-de-una-aplicacin-de-posicionamiento-mediante-wifi?next_slideshow=1

[7]: Francisco Javier Coret Gorgonio. “Tesis Francisco Javier Coret Gorgonio año 2015”. Páginas 32-35.

https://www.dropbox.com/s/ooblv1tpm9kme49/Tesis_Francisco_Javier_Coret_Gorgonio_2015.pdf?dl=0

[8]: Pablo Suazo Fernández. ”Desarrollo de una aplicación de posicionamiento indoor”. Páginas: 28-33

http://es.slideshare.net/peterbuck/desarrollo-de-una-aplicacin-de-posicionamiento-mediante-wifi?next_slideshow=1

[9]: Felix Barba Barba.” ESTUDIO DE ALGORITMOS DE LOCALIZACIÓN EN INTERIORES, PARA TECNOLOGÍAS MÓVILES DE ÚLTIMA GENERACIÓN”.

Página: 14.

https://www.dropbox.com/s/jzq7soax1hzi39i/TFM_Felix_Barba_2012.pdf?dl=0

[10]: Claudio Avallone, Germán Capdehourat. “Tratamiento estadístico de señales”.

https://www.dropbox.com/s/qokbvregkiytsqz/Posicionamiento_indoor_con_WiFi_Avallone_Capdehourat%20%28%20referencia%203%29.pdf?dl=0

[11]: “Bleutooth Low Energy”. Páginas 16-20

https://www.dropbox.com/s/67ptiyaeeb6u2fg/bleutooth%20low%20energy.pdf?dl=0

[12] Mtthew S.Gast “Building Aplicacions with iBeacon, proximity and location sercices with Bluetooth low energy”. Páginas 16-18.

http://file.allitebooks.com/20151203/Building%20Applications%20with%20iBeacon.pdf

[13]: A Guide on Technologies and Use Cases. “INDOOR POSITIONING & NAVIGATION”. Página: 10

http://www.setesca.com/blog/wp-content/uploads/742ef817-0e87-4064-ac22-4e0d6b3854de.pdf

[14]: Improving Indoor Localization Using Bluetooth Low Energy Beacons file:///G:/Ingenieria%20de%20telecomunicaciones/4.Curso/TFG/2083094.pdf

[15]: Alfonso Fernández Sanitaria.” Sistemas de orientación en espacios interiores mediante en balizas digitales”. Página: 26.

https://www.dropbox.com/s/uxfy8k0kpf4wc7t/TFG_FERNANDEZ_SANTAMARIA_ALFONSO_1_2016.pdf?dl=0

[16]: Artículo de Wikipedia https://es.wikipedia.org/wiki/Wifi

[17]: Redes de Area Local. “Redes Wlan” paginas 24-28.

https://www.dropbox.com/s/ogielppr4t34ggb/RAL_05a.pdf?dl=0

[18]: Pablo Suazo Fernández. ”Desarrollo de una aplicación de posicionamiento indoor”. Páginas: 31-33

http://es.slideshare.net/peterbuck/desarrollo-de-una-aplicacin-de-posicionamiento-mediante-wifi?next_slideshow=1

[19] Cristina Regueir Senderos. “PROYECTO DE INVESTIGACIÓN BÁSICA O APLICADA”. Universitat Oberta de Catalunya. Página 5.

http://openaccess.uoc.edu/webapps/o2/bitstream/10609/34081/6/cregueiroTFM0614memoria.pdf.

[20]: Kurt Seifert and Oscar Camacho. “ Implementing Positioning Algorithms Using Accelerometers”.

https://www.dropbox.com/s/8il1b2gpo6z91l0/c%C3%A1lculo%20de%20desplazamiento.pdf?dl=0

[21]: Artículo del periódico Expansión “economía digital, ¿Android o iOS? Los sistemas operativos más usados según el continente”. http://www.expansion.com/economia-digital/companias/2015/12/09/56684be1ca474151018b4590.html

[22]: Artículo de Wikipedia. “Android”.

https://es.wikipedia.org/wiki/Android#Arquitectura

[23]: Artículo Wikipedia “Historial de versiones Android”.

https://es.wikipedia.org/wiki/Anexo:Historial_de_versiones_de_Android

[24]: Artículo Wikipedia “Java”.

https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)

[25]: Artículo Wikipedia “Extensible Markup Lenguage”.

https://es.wikipedia.org/wiki/Extensible_Markup_Language

[26]: Artículo Wikipedia “Lenguaje unificado modificable”.

https://es.wikipedia.org/wiki/Lenguaje_unificado_de_modelado

[27] Desarrollo de programas para Android, recurso de Wikipedia. https://es.wikipedia.org/wiki/Desarrollo_de_programas_para_Android#Android_SDK

[28]: Megan Bambara. “Introducción a Android Studio”

https://www.raywenderlich.com/120508/beginning-android-development-tutorial-android-studio