Centro Universitario de la Defensa en la Escuela Naval Militar

TRABAJO FIN DE GRADO

Sistema de control de personal a bordo: Implementación de un sistema iBeacon-BLE basado en Raspberry Pi

Grado en Ingeniería Mecánica

ALUMNO: Ione Camacho Lorenzo

DIRECTORES: Belén Barragáns Martínez

Norberto Fernández García

CURSO ACADÉMICO: 2014-2015

Universidade Vigo

Centro Universitario de la Defensa
en la Escuela Naval Militar

TRABAJO FIN DE GRADO

Sistema de control de personal a bordo: Implementación de un sistema iBeacon-BLE basado en Raspberry Pi

Grado en Ingeniería Mecánica

Intensificación en Tecnología Naval

Cuerpo General

Universidad de Vigo

RESUMEN

Este proyecto se plantea como la búsqueda de una solución al control de personal a bordo de los buques de la Armada Española.

Se realiza un análisis de las tecnologías utilizadas en la actualidad con el fin de determinar la problemática encontrada hasta el momento en lo que respecta a la localización en interiores. Tras este análisis, se utiliza la tecnología *Bluetooth Low Energy* como tecnología en el desarrollo de un sistema iBeacon que, basado en placas de ordenador reducido *Raspberry Pi*, permite realizar las funciones de control de personal a bordo mediante la localización, el intercambio de información y alertas en diferentes situaciones incluida la situación de hombre al agua.

Desarrollado el sistema iBeacon de control de personal a bordo se realizan pruebas de verificación con el fin de validar la implementación del sistema en buques, considerando a estos, entornos inteligentes capaces de interactuar con el personal embarcado.

PALABRAS CLAVE

Bluetooth Low Energy, iBeacon, Control de personal, Entorno Inteligente, Hombre al agua.

AGRADECIMIENTOS

Agradezco a mi familia, amigos y compañeros de promoción el apoyo recibido en la realización de este proyecto. Al Sr. Comandante del patrullero “Tabarca”, el TN Valentín Calvar Cerecedo, y su tripulación por la disponibilidad y el trato ofrecidos. Agradezco al TN Javier Molina Martínez toda la ayuda e información prestada. Por último, al Centro Universitario de la Defensa por todo el material ofrecido que ha hecho posible el desarrollo del presente proyecto.

Ione Camacho Lorenzo

CONTENIDO

Contenido 1
Índice de Figuras 4
Índice de Tablas 7
1 Introducción 8
1.1 Motivación 8
1.2 Objetivos 8
1.3 Estructura del trabajo 9
1.4 Recursos utilizados 9
1.4.1 Hardware 9
1.4.2 Software 10
1.5 Organización de la memoria 10
2 Revisión del estado del arte 12
2.1 Tecnologías de radiofrecuencia para localización y posicionamiento 12
2.1.1 GPS, GLONASS, BEIDU y GALILEO 12
2.1.2 Protocolo 802.11 13
2.1.3 Radio de banda ultra ancha (UWB) 14
2.1.4 Telefonía móvil (GSM) 15
2.1.5 ZigBee 16
2.1.6 RFID 16
2.1.7 Bluetooth 17
2.2 Comparativa de tecnologías 17
2.3 Métodos de cálculo para el posicionamiento 19
2.3.1 Método de división del espacio 19
2.3.2 Método de triangulación de magnitudes 19
2.4 Proyectos en la actualidad 20
2.4.1 Sistema de localización por redes neuronales. Universidad de Trento 20
2.4.2 Cricket 21
2.4.3 AeroScout 22
2.4.4 Localización en entornos WiFi. Telefónica I+D 22
2.4.5 Proyecto SHERLOC (Sistema Híbrido de Estrés Remoto y Localización) 23
2.4.6 Cosero 24
2.4.7 iBeacons Apple 24
2.4.8 Naval Automated Personnel Tracking (NAPT) 25

2.4.9 Sistema de Localización de Personal. Sistemas FABA 26
2.5 Conclusión 27
3 Estudio de la tecnología *Bluetooth* 28
3.1 Introducción 28
3.1.1 Historia 28
3.1.2 Special Interest Group (SIG) 29
3.2 *Bluetooth* clásico 31
3.2.1 Pila de protocolos de Bluetooth clásico 33
3.2.2 Controlador Bluetooth 34
3.2.3 Host Bluetooth 38
3.3 *Bluetooth* 4.0: *Bluetooth Low Energy* 40
3.3.1 Pila de protocolos BLE 42
3.3.2 Controlador BLE 43
3.3.3 Host BLE 47
3.3.4 Usos de BLE 50
4 Desarrollo del sistema *iBeacon* 52
4.1 Concepto *iBeacon* 52
4.1.1 Arquitectura 52
4.1.2 Posicionamiento con iBeacon. RSSI 53
4.1.3 Modo LEAd con iBeacon 55
4.2 Configuración de RPi: Operación con BLE 56
4.2.1 Instalación y configuración de Raspbian Debian Wheezy 56
4.2.2 Instalación de controladores de dispositivos USB 57
4.2.3 Instalación de pila Bluetooth Low Energy: BlueZ 58
4.3 Configuración del módulo *Bluetooth*: modo de operación *iBeacon* 59
4.3.1 Generalidades 59
4.3.2 Modo Low Energy Advertising (LEAd) 59
4.3.3 Activación del modo LEAd 60
4.4 Configuración de RPi como escáner BLE 61
5 Escenarios de implementación del sistema *iBeacon* 63
5.1 Escenario de localización a bordo 63
5.1.1 Etapa 1: Análisis del comportamiento de la señal BLE 64
5.1.2 Etapa 2: Análisis de RSSI en el patrullero “Tabarca” 68
5.1.3 Etapa 3: Prueba del sistema iBeacon de localización a bordo 71
5.2 Escenario de envío de alarmas y órdenes 76
5.2.1 Sistema de envío de alarmas 77

5.2.2 Sistema de envío de órdenes 78
5.3 Escenario de detección de hombre al agua 79
5.3.1 Sistema de aviso de nivel de batería 80
6 Entorno Inteligente 84
6.1 Infraestructura del Entorno Inteligente 85
6.2 Lógica del sistema de localización de personal por compartimentos o zonas 86
6.3 Lógica del sistema de envío de alarmas y órdenes 87
6.4 Lógica del sistema de alerta de hombre al agua 88
7 Conclusiones y Líneas Futuras 89
8 Bibliografía 91
Anexo I. Manual de configuración de RPi: Operación con BLE 94
Anexo II. Manual de configuración de módulo *Bluetooth*: Operación en modo *iBeacon* 98
Anexo III. Manual de configuración de *iBeacon*: Modo escáner BLE 101
Anexo IV. Manual de configuración de *iBeacon*: Sistema de envío de alarmas y órdenes 104
Anexo V: Manual de configuración de *iBeacon*: Sistema de aviso de nivel de batería 107
Anexo VI: Gráficas y resultados de mediciones de RSSI 109
Anexo VII: Pruebas de consumo energético 114

ÍNDICE DE FIGURAS

Figura 1-1. Placa Raspberry Pi B+. 10
Figura 2-1. Esquema básico de segmento espacial [1]. 12
Figura 2-2. Triangulación de señales por tiempo de llegada. 13
Figura 2-3. Triangulación de señal por ángulo de llegada. 14
Figura 2-4. Gráfica de densidad espectral en UWB. 14
Figura 2-5. Módem GSM Sony Ericsson GM28. 15
Figura 2-6. Red de sensores con tecnología ZigBee. 16
Figura 2-7. Gráfica de consumo de potencia de tecnologías RF 19
Figura 2-8. Gráfica coste /complejidad según tasa de bit para tecnologías de RF. 19
Figura 2-9. Esquema de situación de APs en planta. 21
Figura 2-10. Componentes del sistema de localización Cricket. 21
Figura 2-11. Esquema básico del sistema de localización AeroScout. 22
Figura 2-12. Pulsera de usuario del sistema SHERLOC. 23
Figura 2-13. Robot inteligente COSERO. 24
Figura 2-14. Sistema *iBeacon* implantado en comercio. 25
Figura 2-15. Diferentes tipos de *iBeacon* tipo Estimote 25
Figura 2-16. Etiqueta RFID portable sistema NAPT. 26
Figura 2-17. Infraestructura del Sistema de Localización de Personal. 26
Figura 3-1. Logo tecnología *Bluetooth*. 28
Figura 3-2. Esquema de capacidades de *Bluetooth* 4.0. 31
Figura 3-3. Diferentes tipos de picorredes: (a) *Unicast* y (b) *Multicast*. 32
Figura 3-4. Red WPAN *Bluetooth* dispersa (*scatternet*). 32
Figura 3-5. Técnica *time slot* y salto de frecuencia (*frequency hopping*) entre maestro y esclavo. 32
Figura 3-6. Pila de protocolo *Bluetooth* clásico. 33
Figura 3-7. Transmisor con modulación GFSK. 34
Figura 3-8. Modulación GFSK. 35
Figura 3-9. Procedimientos *inquiry* y *paging*. 37
Figura 3-10. HCI dentro del controlador *Bluetooth*. 38
Figura 3-11. Trama de datos *Bluetooth* [24]. 38
Figura 3-12. *Service Discovery Protocol*. 39
Figura 3-13. Datos de eventos locales con dispositivos BLE. 41
Figura 3-14. Comunicación entre protocolos de dispositivos modo dual y modo simple. 42
Figura 3-15. Pila de protocolos BLE. 42
Figura 3-16. Canales BLE. 43

Figura 3-17. Paquete de datos BLE. 44
Figura 3-18. Evento *advertising* con PDU ADV_IND y SCAN_REQ SCAN_RSP. 45
Figura 3-19. Evento de conexión PDU CONNECT_REQ. 46
Figura 3-20. Características definidas por *Bluetooth* SIG con su dirección *handle* (Derecha). 48
Figura 3-21. Estructura de un perfil GATT. 49
Figura 3-22. Ejemplo de servicio BLE con sus correspondientes características. 49
Figura 4-1. Estructura de una trama de un *iBeacon* sobre BLE. 52
Figura 4-2. Captura de pantalla de *iOS iBeacon Locator App*. 54
Figura 4-3. Rangos de proximidad a un *iBeacon*. 55
Figura 4-4. Captura de pantalla de *iOS Estimote App*. 55
Figura 4-5. Esquema básico de funcionamiento de *iBeacon* en modo LEAd. 56
Figura 4-6. Interfaz de línea de comandos de Raspbian Debian Wheezy. 57
Figura 4-7. Captura de configuración de conexión inalámbrica del módulo WiFi. 57
Figura 4-8. Pila de protocolos BLE. 59
Figura 4-9. Plan de frecuencias de BLE con 37 canales de comunicación y 3 para notificaciones. 60
Figura 4-10. Ejemplo de ejecución de comando *LE Set Advertising Data*. 60
Figura 4-11. Ejemplo de ejecución de *LE Set Advertising Parameters*. 61
Figura 4-12. Ejemplo de ejecución de *LE Set Advertising Enable*. 61
Figura 4-13. Acciones BLE disponibles en la pila de protocolos BlueZ. 61
Figura 4-14. Captura de comando *hcidump* tras ejecutar un escaneo *Low Energy*. 62
Figura 4-15. Captura del terminal de RPI tras ejecutar script de escaneo *Low Energy*. 62
Figura 5-1. Captura de pantalla de *App Particle Detector*. 64
Figura 5-2. Esquema de situación de *iBeacons* y puntos de recogida de medidas (aspas). 65
Figura 5-3. Resultados de estudio de directividad según plano de orientación. 65
Figura 5-4. Disposición de *iBeacons*. Primera fase (azul), segunda fase (rojo) y tercera fase (verde). 66
Figura 5-5. Representación gráfica de fluctuación de RSSI para un PRI de 1500 ms. 68
Figura 5-6. Cubierta baja del patrullero "Tabarca". 69
Figura 5-7. Situación de *iBeacon* en el sollado de dotación del patrullero "Tabarca". 69
Figura 5-8. Fluctuación de RSSI para un PRI de 1500 ms en sollado del patrullero "Tabarca". 70
Figura 5-9. Esquema de situación de escáneres. 72
Figura 5-10. Itinerario de pruebas. 73
Figura 5-11. Esquema de sistema de alarma *iBeacon*. 76
Figura 5-12. Captura de alerta de Zafarrancho de Combate en terminal Debian. 78
Figura 5-13. Captura de orden Acuda a puente. 78
Figura 5-14. Placa *MoPi* [28]. 80

Figura 5-15. Alimentación y conexión de MoPi. 81
Figura 5-16. Configuración de niveles de voltaje en *MoPi*. 81
Figura 5-17. Sistema de aviso de nivel de batería: Nivel alto. 82
Figura 5-18. Sistema de aviso de nivel de batería: Nivel medio. 82
Figura 5-19. Sistema de aviso de nivel de batería: Nivel bajo. 83
Figura 6-1. Esquema de entorno inteligente [30]. 84
Figura 6-2. Esquema básico de entorno inteligente en buques. 85
Figura 7-1. Placa de desarrollo BLE. Serie: DB-CSR1010-10137-1A-ND [29]. 90
Figura AI-1. Página de inicio de Raspberrypi.org. 94
Figura AI-2. Comprobación de conexión de adaptador USB WiFi en Raspbian. 95
Figura AI-3. Edición de la interfaz de red con editor de texto GNU nano. 96
Figura AI-4. Comprobación de conexión del módulo *Bluetooth* USB en Raspbian. 96
Figura AII-1. Módulo *Bluetooth* del *iBeacon* 2 en estado DOWN. 98
Figura AII-2. Módulo *Bluetooth* del *iBeacon* 2 en estado UP RUNNING (Activo). 98
Figura AII-3. Ejemplo de configuración de intervalo de repetición de pulso fijo de 500 ms. 99
Figura AII-4. Ejemplo de activación de transmisión de módulo *Bluetooth*. 100
Figura AII-5. Ejemplo de configuración de UUID, Major y minor. 100
Figura AIV-1. Configuración de audio con interfaz de *alsamixer*. 106
Figura AVII-1. Placa MoPi alimentando a RPi. 114
Figura AVII-2. Configuración de voltajes de la fuente de alimentación en MoPi. 114

ÍNDICE DE TABLAS

Tabla 2-1. Comparativa de tecnologías de RF. 18
Tabla 2-2. Comparativa de proyectos en la actualidad....27
Tabla 3-1. Frecuencias Bluetooth por países....34
Tabla 3-2. Clases de dispositivos Bluetooth. 34
Tabla 3-3. Comparativa entre BLE y Bluetooth Clásico....41
Tabla 5-1. Mediciones de fluctuación de RSSI. 67
Tabla 5-2. Mediciones de fluctuación de RSSI con una sola baliza. 67
Tabla 5-3. Medidas de RSSI en sollado de dotación del patrullero "Tabarca". 70
Tabla 5-4. Valores de RSSI y distancia teórica en los puntos de medida. 74
Tabla 5-5. Comparativa de proximidad real-medida....75
Tabla AII-1. Tabla de conversión de intervalos de repetición. 99

1 INTRODUCCIÓN

1.1 Motivación

El desarrollo experimentado por las nuevas tecnologías, junto con la integración del entorno, tecnología e informática ha llevado, en la actualidad, a la aparición de los entornos inteligentes.

Los entornos inteligentes permiten a las personas interactuar con el medio que las rodea, principalmente, tras un intercambio de información. El entorno analiza la localización, comportamiento y contexto en el que se encuentran las personas y les envía información de utilidad.

Considerando a los buques de la Armada Española como entornos inteligentes, se propone en el presente trabajo la implementación de un sistema de control de personal a bordo, ya sea en la cámara de máquinas, compartimentos interiores o incluso en cubierta que pueda proporcionar un mecanismo de intercambio de información de interés, por ejemplo: localización de personal por compartimentos, alarma por entrada en zonas restringidas, alarma por situaciones de emergencia, identificación de situación de hombre al agua, etc.

La localización en interiores es el área principal de este sistema, siendo un ámbito estudiado e investigado en los últimos años sin llegar a una solución tan aceptada y extendida como es el GPS para el posicionamiento en exteriores.

Este hecho es la principal motivación de este trabajo de fin de grado: la implementación de balizas *iBeacon* con tecnología *Bluetooth* basadas en *Raspberry Pi* para la localización y control de personal en interiores e intercambio de información.

1.2 Objetivos

El objetivo principal de este trabajo es configurar un sistema de radiobaliza sencillo. Para ello se utilizará la tecnología *Bluetooth* como método de transmisión de datos, y una placa de desarrollo *Raspberry Pi* (RPi) para el control de la baliza. Una vez instalado el sistema, se realizarán medidas encaminadas a valorar su posible utilización en escenarios de control de personal.

En este trabajo de fin de grado se dejan sentadas las bases teóricas, técnicas y de desarrollo del sistema de balizamiento, para la continuación y aplicación en un futuro trabajo en el ámbito de control de personal a bordo.

1.3 Estructura del trabajo

Con el fin de llevar a cabo el objetivo, se ha dividido el desarrollo del trabajo en cuatro etapas:

1. Documentación previa, con el fin de obtener la información y conocimientos necesarias de las tecnologías a manejar.

2. Configuración y prueba de los dispositivos *iBeacon*:
   - Instalación de drivers *Bluetooth* en sistema operativo Raspbian.
   - Instalación y configuración de Raspberry Pi (RPi) para utilizar los módulos USB de *Bluetooth Low Energy* (BLE).
   - Configuración de los módulos BLE en modo operación *iBeacon*.
   - Prueba de los módulos en salas del CUD.
   - Prueba de los módulos en buques de la Armada.

3. Configuración y prueba de los dispositivos para escenarios con intercambio de información:
   - Configuración de los dispositivos para intercambio de información sobre BLE.

4. Mejoras del sistema:
   - Medida del consumo energético del sistema *iBeacon*.
   - Análisis de RSSI y comportamiento de la señal BLE.
   - Aplicaciones en entornos inteligentes.

1.4 Recursos utilizados

En el presente apartado se exponen los componentes de *hardware* utilizados en el sistema *iBeacon*, así como el *software* empleado en el desarrollo:

1.4.1 *Hardware.*

1) Placa *Raspberry Pi* modelo B+:

   - SoC: Broadcom BCM2835 (CPU, GPU y SDRAM)
   - CPU ARM1176JZF-S a 700 MHz
   - GPU Broadcom VideoCore IV
   - Memoria RAM 512 MB (compartidos con la GPU)
   - Conexiones:

     - 4 x USB 2.0
     - 1 x Salida audio mini jack 3.5 mm
     - 1 x Salida audio/vídeo HDMI
     - 1 x Micro USB
     - 1 x RJ45 10/100 Ethernet RJ45

   - Conectividad:

     - LAN Red local 10/100
     - WiFi (mediante adaptador USB WiFi compatible, no incluido)

- Slot MicroSD para tarjetas.
- Alimentación: 5V/600 mA (3.5 W) via microUSB
- Dimensiones: 85.6 mm x 53.98 mm
- Cable HDMI.
- Alimentación por cable USB a micro USB.
- Carcasa protectora.

Figura 1-1. Placa Raspberry Pi B+.

2) Batería portátil externa RAVpower con capacidad de 10400 mAh / 38.4 Wh y 2 salidas USB de 5V 1A/2A.
3) Kingston MicroSD de 8 GB.
4) Adaptador nano USB Wireless 802.11.b/g/n de 150 Mbps.
5) Adaptador USB *Bluetooth* CSR 4.0 serie BTA-CSR4B5.
6) Teclado por cable USB.
7) Pantalla con entrada HDMI.
8) Placa controladora de alimentación *MoPi: hot-swap and 24/7 power for the Raspberry Pi*.
9) Iphone 5C.

1.4.2 *Software*

1) Sistema operativo Raspbian Wheezy para Raspberry Pi.
2) *Wind32DiskImager* (Creación de MicroSD con SO ejecutable).
3) *Putty* (Conexiones remotas).
4) Notepad ++ (Editor de líneas de código de programación).
5) Apps iOS (*iBeacon Locator y Particle Detector).*

1.5 Organización de la memoria

Esta memoria está estructurada en nueve capítulos acompañados de siete anexos, según se indica a continuación:

En el Capítulo 1 se realiza una introducción al proyecto, se expone la motivación, objetivos y estructura del mismo así como una descripción del *hardware* y *software* utilizados para el desarrollo.

En el Capítulo 2 se realiza una revisión del estado del arte:

- En primer lugar, se realiza un análisis de las diferentes tecnologías utilizadas para la localización en interiores.
- En segundo lugar, se presentan los métodos utilizados para el cálculo de la posición, centrados en la triangulación y el análisis del patrón de comportamiento de señales.
- En tercer lugar, se presentan varios proyectos actuales para la localización en interiores.

En el Capítulo 3 se presenta un análisis en profundidad tanto de la tecnología *Bluetooth* como de la tecnología *Bluetooth Low Energy,* que serán la base de desarrollo del presente proyecto.

En el Capítulo 4 se describe la configuración de los ordenadores RPi para el funcionamiento como sistema *iBeacon*, analizando teóricamente y técnicamente las funcionalidades del sistema. Además, se realiza la configuración de RPi en modo escáner de señales BLE y se describe el funcionamiento de este modo.

En el Capítulo 5 se plantean los diferentes escenarios de aplicación del sistema *iBeacon*: Un escenario de localización de personal a bordo, un escenario de envío de alertas y órdenes y, por último, un escenario de detección de la situación de hombre al agua. Además, en este capítulo, se incluyen diferentes pruebas y experimentos de verificación de la aplicación del sistema *iBeacon* en los diferentes escenarios propuestos.

En el Capítulo 6 se introduce el concepto de *Entorno Inteligente* y su aplicación teórica a buques utilizando el sistema *iBeacon* de control de personal a bordo. Se describen, en este capítulo, las diferentes lógicas a seguir por el sistema *iBeacon* para la implementación de los escenarios propuestos anteriormente.

En el Capítulo 7 se exponen las conclusiones obtenidas tras el desarrollo del presente proyecto así como las líneas futuras de investigación y mejora del sistema *iBeacon*.

La última sección, incluye las referencias bibliográficas y los recursos web consultados para la ejecución del proyecto.

Por último, se adjuntan siete anexos que presentan los manuales de configuración del sistema *iBeacon* para implementar las diferentes funcionalidades de éste, acompañados de un anexo incluyendo los resultados gráficos de las pruebas realizadas en el Capítulo 5 y un anexo de estudio del consumo energético del sistema *iBeacon* en los diferentes modos de funcionamiento.

2 REVISIÓN DEL ESTADO DEL ARTE

En este capítulo de la memoria se exponen las tecnologías existentes en la actualidad para resolver el problema de la localización y posicionamiento en interiores, los métodos de cálculo de posición más utilizados, así como los principales proyectos desarrollados en el presente.

2.1 Tecnologías de radiofrecuencia para localización y posicionamiento

2.1.1 *GPS, GLONASS, BEIDU y GALILEO*

Los sistemas de localización por satélite, ya sea GPS (EE.UU.), GLONASS (Rusia), BEIDU (China) o GALILEO (Europa) utilizan el mismo principio de funcionamiento. Un segmento espacial, formado por 24 satélites, da cobertura global de señal al segmento terrestre formado principalmente por receptores de señal satélite.

Un receptor satélite recibe señal de al menos 4 satélites con información de posición y hora de emisión de cada uno, datos que permiten llevar a cabo la triangulación de la posición absoluta (latitud y longitud) y cálculos de velocidad y altitud con la señal de un cuarto satélite. Corregidos los retardos de propagación de la señal y utilizando más de cuatro satélites, se obtiene precisión de hasta 10 metros siendo un sistema idóneo para la localización a nivel global.

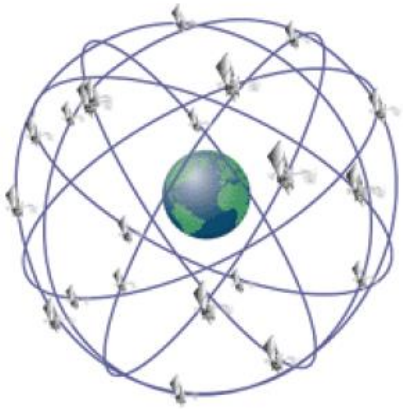


Figura 2-1. Esquema básico de segmento espacial [1].

Sin embargo, presenta deficiencias en posicionamiento a pequeña escala, como podría ser el interior de un edificio o buque, debido a las siguientes causas:

- **Necesidad de línea de visión directa** (*LoS*): Existen determinadas situaciones en las que no se establece recepción directa de la señal de al menos 4 satélites debido a la posición del receptor, por ejemplo, en ciudades con edificios altos, calles estrechas, túneles y compartimentos.

- **Intensidad de señal en recepción**: Debido a la distancia entre el segmento espacial y el segmento terrero se produce atenuación de la señal GPS, siendo demasiado débil para traspasar superficies y alcanzar receptores localizados en interiores.

- **Sistema de referencia**: El sistema GPS proporciona una referencia absoluta o "*datum*" de tipo cartesiano X, Y, Z; mientras que, en un sistema de localización en interiores, se buscan referencias locales y adaptadas al contexto en el que se encuentra el receptor, por ejemplo: se encuentra usted en la cámara de máquinas.

2.1.2 *Protocolo 802.11*

El protocolo 802.11 (WiFi) establece un sistema de comunicación centralizado, es decir, existe una red con uno o varios puntos de acceso (*APs: Access Points*) a los que se conectan diferentes clientes representados por tarjetas de red inalámbricas. Periódicamente, es enviada una señal por los puntos de acceso a modo de baliza (*beacon*) que el cliente recibe, permitiéndole saber qué redes inalámbricas se encuentran disponibles en su entorno.

Según [2], esta emisión y recepción de señal permite la localización en interiores utilizando tres técnicas:

- **Técnica "tiempo de llegada"** (*ToA: Time of Arrival*): Basada en el principio de que una señal electromagnética tarda un tiempo en propagarse una cierta distancia. Mediante el cálculo espacio/tiempo y triangulación de señales, se puede obtener la distancia entre emisor y receptor.

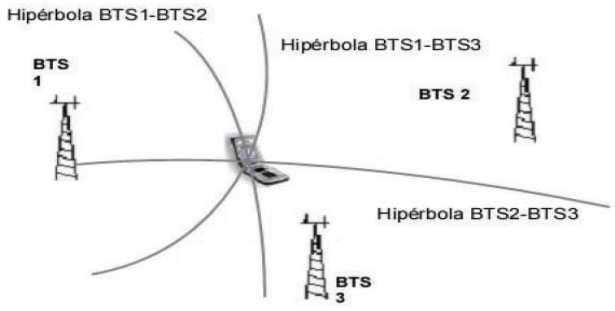


Figura 2-2. Triangulación de señales por tiempo de llegada.

- **Técnica "ángulo de llegada"** (*AoA: Angle of Arrival*): Basada en el principio de procesado de señal mediante *arrays*. Se utilizan varias antenas en recepción y se triangula la señal con respecto al ángulo con el que se recibe. El protocolo 802.11 utiliza dispositivos de antena única, por lo que esta técnica ha de aplicarse con hardware adicional. Presenta deficiencias en presencia de multitrayectos debido al error en recepción de ángulos, así como en entornos con obstáculos que no permitan la comunicación *LoS*.

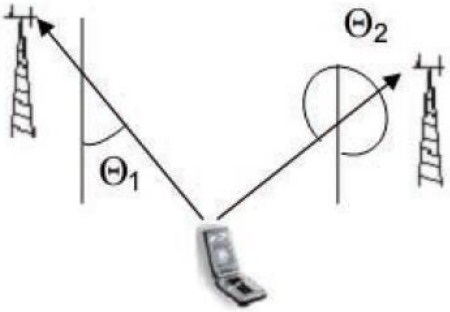


Figura 2-3. Triangulación de señal por ángulo de llegada.

- **Técnica "*nivel de señal recibida*"** (*RSS: Received Signal Strength)*: Basada en el principio de atenuación de la señal recibida. De forma teórica, se relaciona la distancia entre un terminal y un punto de acceso tomando como dato la diferencia entre la señal recibida (atenuada) y la teórica (no atenuada). Es necesario disponer de un mapa radio de la señal para cada dispositivo según su antena, ángulo de emisión, nivel de batería, etc. Presenta grandes limitaciones con la aparición de multitrayectos, reflexiones, difracciones y fenómenos que hacen compleja la estimación y cálculo de la relación distancia/atenuación.

Esta tecnología presenta como ventajas inmediatas la estandarización y el bajo coste. Sin embargo, ha sido diseñada para conseguir altas velocidades de transmisión de datos a costa de elevado consumo de energía.

La precisión de localización está comprendida entre 1-5 metros dependiendo de la densidad de emisiones RF en el área, los tiempos promediados y la complejidad del entorno.

2.1.3 *Radio de banda ultra ancha (UWB)*

La tecnología UWB (*Ultra WideBand)* tiene su origen para aplicaciones militares de RADAR y comunicaciones seguras permitiéndose el uso civil a partir de 1990.

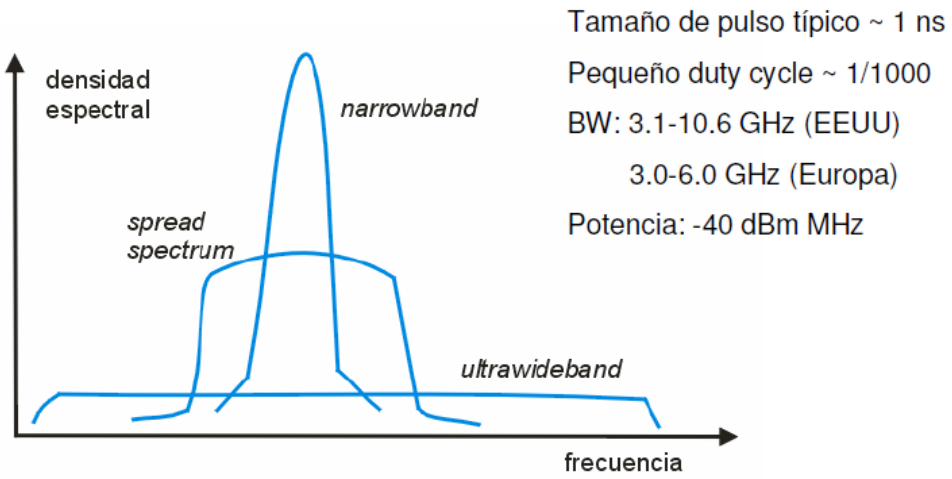


Figura 2-4. Gráfica de densidad espectral en UWB.

Debido a su elevado rango de frecuencias y paramétrica, como se puede observar en la Figura 2-4, presenta grandes ventajas en diferentes aspectos:

- Menor interferencia con otros sistemas de RF debido a su baja densidad espectral.
- Buena capacidad de detección debida a la modulación de la señal UWB.
- Buena propagación por interiores y penetración en materiales.
- Precisión de hasta 1 metro.

Como desventajas principales, se encuentran:

- Se requiere sincronización precisa entre lectores.
- Localización estimada por técnicas *ToA*.
- Tecnología en trámite de estandarización.

Debido a sus características, esta tecnología podría presentar grandes opciones para la localización en interiores, sin embargo, su arquitectura y su elevado ancho de banda la convierten en óptima para otros usos como reemplazar la conexión de cables vía USB entre dispositivos.

2.1.4 *Telefonía móvil (GSM)*

La telefonía móvil actual utiliza módem GSM que posibilita la localización del dispositivo portador mediante dos métodos [4]:

- **Método "*Cell ID*":** Basado en una célula que recibe la señal del teléfono móvil identificándolo y localizándolo. La precisión depende del tamaño de la célula receptora.

- **Método "*Diferencia de tiempos mejorada (E-OTD)*":** Basado en la localización mediante el cálculo del tiempo de propagación de señales transmitidas entre el teléfono móvil y las estaciones base. La precisión está comprendida entre 50-200 metros, lo que representa una deficiencia para la efectiva localización en interiores.

Figura 2-5. Módem GSM Sony Ericsson GM28.

2.1.5 *ZigBee*

La tecnología ZigBee está basada en el estándar IEEE 802.15.4, estándar que define el nivel físico y el control de acceso al medio de redes inalámbricas de área personal [5].

Es una tecnología diseñada para ser el estándar de comunicación de las redes inalámbricas de sensores como se puede observar en la Figura 2-6, presentando tres principios básicos de trabajo: *low cost, low power* y *low data rate.*

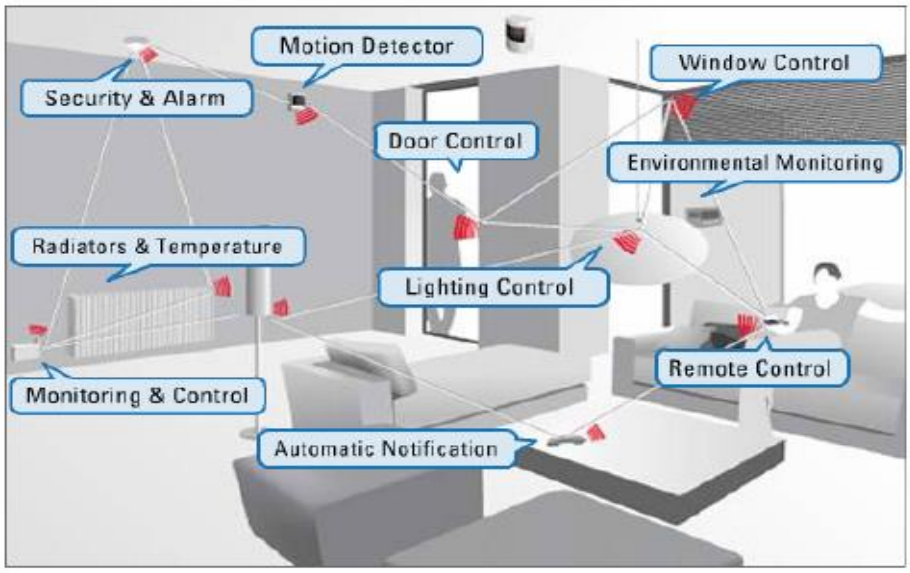


Figura 2-6. Red de sensores con tecnología ZigBee.

Diversos trabajos realizados en la actualidad con esta tecnología ponen en evidencia la viabilidad y opciones que presenta esta tecnología para la localización en interiores, debido principalmente a su bajo coste y baja potencia de emisión.

Sin embargo, presenta grandes limitaciones de utilidad para aplicaciones donde sea necesario un cierto volumen de transmisión de datos debido a su limitada tasa de bit. A mayores, se producen fluctuaciones de la señal debido a la configuración del entorno o el movimiento de personal, que se ven reflejadas en pérdida de precisión.

2.1.6 *RFID*

La tecnología de identificación por radiofrecuencia (*RFID*) está basada en la transmisión de una señal RF por parte de un emisor *(etiqueta o tag*) que posteriormente es registrada y analizada por un lector.

Las etiquetas o *tag*, que consisten en microchips adjuntos a antenas radio, pueden ser de tres tipos diferentes:

- **Etiquetas RFID pasivas:** Este tipo de *tag* no lleva fuente de alimentación propia, utilizando la energía inducida por el lector en el escaneo para responder. Presentan ventajas de tamaño y coste, ya que se pueden encontrar etiquetas de 0,5 x 0,5 milímetros a 10 céntimos la unidad. Sin embargo, debido a su pasividad, el tiempo de vida de la señal respuesta es bastante corto, limitando el alcance a unos 7 metros.
- **Etiquetas RFID semi-activas:** Este tipo de *tag* integra una batería de limitada capacidad que posibilita la emisión de una respuesta en radiofrecuencia sin que la etiqueta deba

reutilizar la energía recibida por el escaneo del lector. La principal limitación es la escasa transferencia de datos frente a la autonomía.

- **Etiquetas RFID activas:** Este tipo de *tag* lleva integrada una fuente de alimentación propia que le permite emitir señales en frecuencias de 455 MHz, 2,45 GHz y 5,8 GHz consiguiendo alcances operativos de hasta 100 metros. Presentan un elevado coste y un aumento de tamaño del dispositivo, pero su mayor ventaja viene determinada por su uso como baliza, permitiendo una localización en tiempo real tras la emisión periódica de una señal de identificación.

En la actualidad, la tecnología RFID presenta numerosos usos: control de inventario en tiendas, peajes, identificación de ganado y mascotas, control de acceso a edificios, pasaportes digitales e, incluso, localización en interiores. En este último aspecto, de interés para este trabajo, la tecnología RFID presenta el inconveniente del elevado coste, ya que es necesaria la instalación de numerosos dispositivos tanto de control y lectura, como etiquetas activas para permitir la localización y evitar las interferencias mutuas.

Por otro lado, a nivel técnico, la tecnología RFID proporciona tasas de transmisión de datos variables y dependientes de la frecuencia de transmisión. Para conseguir tasas aceptables para una comunicación full-dúplex es necesario trabajar con frecuencias de 2,4 GHz a 5,8 GHz, limitando el alcance de localización a 1 metro [10].

2.1.7 *Bluetooth*

La tecnología *Bluetooth* define un estándar, tanto hardware como software, de comunicación inalámbrica. Este estándar posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia en distancias cortas, ya sea en entornos cerrados (compartimentos, despachos, etc.) como en entornos públicos.

Uno de los objetivos de esta tecnología es la posibilidad de reemplazar o eliminar el cableado y conectores que enlazan unos dispositivos con otros. Además, esta tecnología pretende facilitar la interacción y sincronización de los diferentes dispositivos, tanto móviles como fijos, sin necesidad de visión directa entre ellos. Otro objetivo es el de obtener una tecnología de bajo coste y potencia que posibilite dispositivos baratos consiguiendo a la vez la comunicación confiable y segura por radio.

Esta tecnología *Bluetooth* aporta grandes opciones y soluciones tanto a la localización en interiores como a la transmisión de datos full-dúplex entre dispositivos. Está siendo utilizada como tecnología de referencia en diversos proyectos de localización e interactuación entorno/usuario, como, por ejemplo, el proyecto *iBeacon* desarrollado por Apple Inc [11]. Por tanto, es el estándar elegido para el desarrollo del presente proyecto, en el cual se describirá y desarrollará dicha tecnología en profundidad.

2.2 Comparativa de tecnologías

Tras el análisis de diversas tecnologías, se presenta en este apartado un resumen de los cuatro estándares de más interés en el desarrollo de este trabajo: *UWB (IEEE 802.15.3), ZigBee (IEEE 802.15.4), WiFi (IEEE 802.11.a/b/g) y Bluetooth (IEEE 802.15.1).*

ESTÁNDAR / CATACTERÍSTICAS	***Bluetooth***	**UWB**	**ZigBee**	***WiFi***
Banda de frecuencia	*2.4 GHz*	*3.1- 10.6 GHz*	*868/915 MHz;2.4GHz*	*2.4 GHz; 5GHz*
Máxima tasa de señal	*1 Mb/s*	*110 Mb/s*	*250 Kb/s*	*54 Mb/s*
Alcance nominal	*10 m*	*10 m*	*10-100 m*	*100 m*
Potencia nominal de TX	*-10 - 0 dBm*	*-41.3 dBm/MHz*	*(-25) - 0 dBm*	*15 - 20 dBm*
Número de canales RF soportados	*79*	*1-15*	*1-10 ; 16*	*14*
Ancho de banda de canal	*1 MHz*	*500 MHz-7.5 GHz*	*0.3/0.6 MHz; 2 MHz*	*22 MHz*
Modulación	*GFSK*	*BPSK/QPSK*	*BPSK(+ASK), O-QPSK*	*BPSK, QPSK, COFDM, CCK, M-QAM*
Mecanismo de coexistencia	*Saltos de frecuencia*	*Saltos de frecuencia*	*Selección dinámica de frecuencia*	*Selección dinámica de frecuencia y control de potencia de TX*
Tipo de celda básica	*Piconet*	*Piconet*	*Estrella*	*Subsistema de estación base (BSS)*
Extensión en la pila básica	*Scatternet*	*Peer-peer*	*Malla en árbol*	*Conjunto de servicio extendido (ESS)*
Máximo número de nodos	*8*	*8*	*Hasta 6500*	*Según Red*
Seguridad y protección de datos	*16 bit CRC*	*32 bit CRC*	*16 bit CRC*	*32 bit CRC*

Tabla 2-1. Comparativa de tecnologías de RF.

Como se observa en la Tabla 2-1, en lo referente a potencia de transmisión de señal, *Bluetooth* y ZigBee están principalmente pensados para dispositivos portátiles, alcances relativamente cortos y bajo consumo. Por otro lado, UWB está diseñado principalmente para tasas elevadas de transmisión de datos a corto alcance, lo que supone un mayor consumo energético. En el caso de WiFi, su principal uso implica una conexión y transmisión de datos en el tiempo, por lo que los dispositivos que soportan este estándar están provistos de importantes fuentes de alimentación.

A continuación, la Figura 2-7 [2] muestra una comparativa de consumo energético referido a la tasa de bit (1 Mb/s):

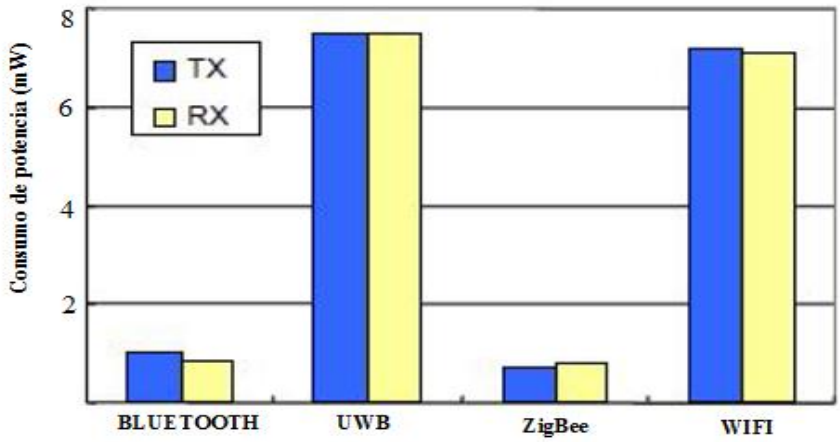


Figura 2-7. Gráfica de consumo de potencia de tecnologías RF

Finalmente, en la Figura 2-8 se expone una comparación de tecnologías en lo referente al coste de dispositivos y complejidad a la hora de implementar una red WPAN, necesaria para aplicar las tecnologías citadas a la localización en interiores.

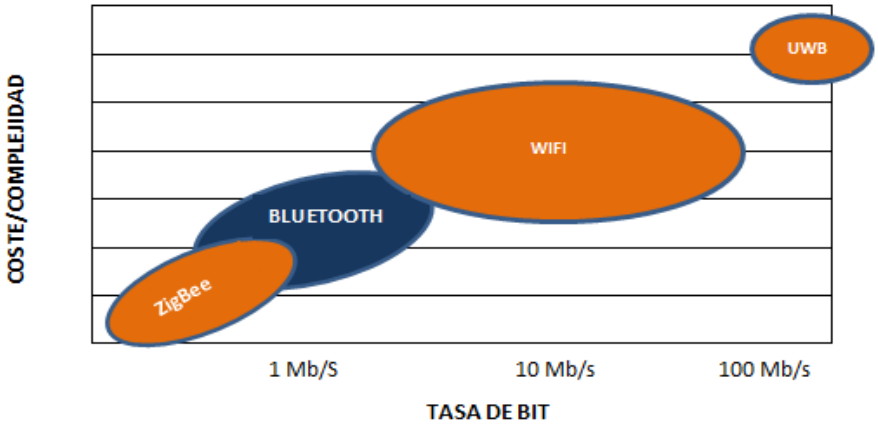


Figura 2-8. Gráfica coste /complejidad según tasa de bit para tecnologías de RF.

2.3 Métodos de cálculo para el posicionamiento

En lo referente al cálculo de la posición de un dispositivo en interiores, independientemente de la tecnología y de la arquitectura utilizada, existen dos principales métodos utilizados en la actualidad:

2.3.1 *Método de división del espacio*

Este método está basado en la división del espacio en celdas de igual tamaño a las cuales se les asigna un punto de acceso. Cada punto de acceso posee la misma cobertura, por tanto, cuando una baliza emite una señal, ésta será localizada en el área de responsabilidad de un determinado punto de acceso.

2.3.2 *Método de triangulación de magnitudes*

Este método está basado en el cálculo matemático de la geometría de triángulos. A partir de magnitudes conocidas entre emisor-receptor y conocida la localización de dos puntos de referencia, se

puede estimar la posición de un dispositivo en el plano. La localización precisa y sin errores se consigue con tres puntos de referencia conocidos.

Las magnitudes medidas y analizadas por los receptores fijos son:

- Ángulo de llegada de la señal del emisor (*AoA*).
- Tiempo de llegada de la señal del emisor (*ToA*).
- Nivel de señal recibido (*RSS*).

Estos métodos de cálculo son utilizados conjuntamente con diferentes técnicas de reconocimiento de patrones como: *k- Nearest Neighbors* (kNN), métodos bayesianos, *Support Vector Machines* o técnicas de redes neuronales [1].

En lo referente a refinamiento de la localización para mejorar la precisión, se utilizan diferentes herramientas como: filtros de partículas, filtros de Kalman, algoritmos de Montecarlo y modelos ocultos de Markov [1].

2.4 Proyectos en la actualidad

La localización en interiores está siendo un ámbito de estudio y desarrollo en la actualidad debido a sus numerosas aplicaciones: localización de objetos, control y cuidado de personas con necesidades especiales, control de personal en empresas, guía de maquinaria en planta, guía en rutas de interés en museos, localización de asientos en teatros y estadios, etc. Si, además de la localización, se consigue la interacción entre el usuario y el entorno inteligente en el que se encuentra, el rango de aplicaciones de este tipo de proyectos se hace casi infinito.

En el presente apartado se exponen los proyectos desarrollados o en vía de desarrollo por diferentes organismos en la actualidad, incluyendo algunos en el ámbito de control de personal a bordo de buques.

2.4.1 *Sistema de localización por redes neuronales. Universidad de Trento*

Este sistema desarrollado por la Universidad de Trento utiliza la lectura de potencia recibida (RSS) para establecer la localización.

Utiliza tarjetas de red WiFi del mismo fabricante y modelo para los emisores (ordenadores portátiles), los cuales emiten una señal utilizada para calibrar los puntos de acceso encargados de la localización. La calibración previa permite levantar mapas de señales para diferentes posiciones de los emisores en presencia de distintas cantidades de personal a localizar.

En la fase experimental de este proyecto, se utilizaron solamente tres puntos de acceso para una planta de 625 m^2 colocados en los bordes exteriores del edificio de pruebas, como se muestra en la Figura 2-9.

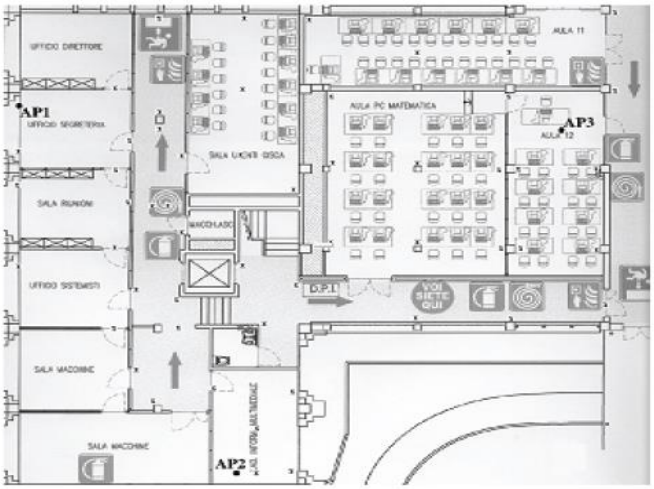


Figura 2-9. Esquema de situación de APs en planta.

Presentó, en su fase de experimentación, un error en la precisión de localización de 1,5 m, utilizando un sistema de referencia en coordenadas cartesianas. Esta precisión se ve empeorada en el caso de que los emisores operen con tarjetas de red WiFi diferentes, o que los puntos de acceso no se encuentren configurados de igual forma según su localización en la planta, siendo necesaria una continua calibración del sistema para cada una de estas variaciones.

Este sistema puede llegar a ser una buena opción para la localización en edificios o salas de tamaño considerable, donde el objetivo sea localizar maquinaria, mercancía u objetos que lleven implantados el *hardware* y *software* necesario para gestionar las tarjetas de red, quedando en un segundo plano para la localización de personal.

2.4.2 *Cricket*

Cricket es un sistema de localización en interiores para entornos inteligentes basados en sensores. Desarrollado por la empresa Oxygen y el MIT (*Massachusetts Institute of Technology*) [7], Cricket ofrece información de localización, identificadores de espacio y orientación, pudiéndose ejecutar sus aplicaciones en dispositivos de mano, portátiles y nodos sensores. Los componentes de este sistema se muestran en la Figura 2-10.

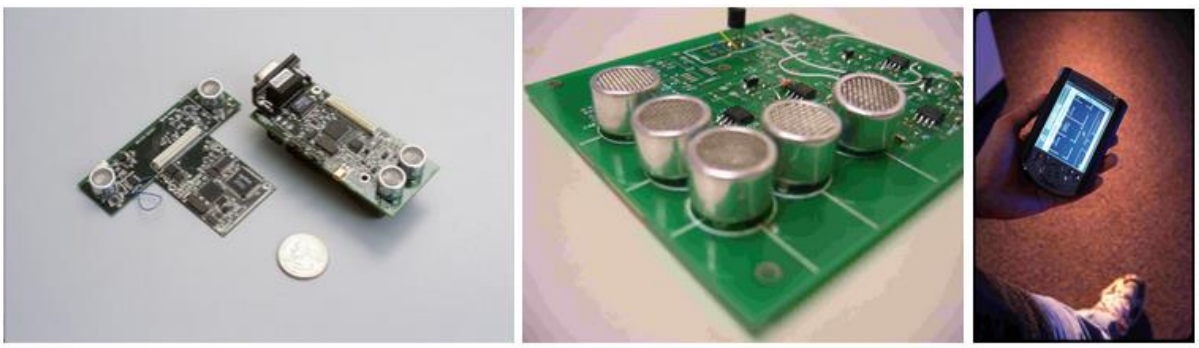


Figura 2-10. Componentes del sistema de localización Cricket.

Cricket utiliza una combinación de tecnologías de radiofrecuencia y ultrasonido para proporcionar información sobre la ubicación de los dispositivos host conectados. Las balizas son montadas en el techo o pared y emiten la información en un canal de RF. Con cada emisión de RF, la baliza transmite un pulso ultrasónico concurrente. Los receptores obtienen una estimación de la distancia de la baliza correspondiente mediante el aprovechamiento de la diferencia en las velocidades

de propagación entre la RF (velocidad de la luz) y el sonido. Posteriormente, el receptor ejecuta algoritmos de cálculo y correlaciona muestras de RF y de ultrasonidos. Además de determinar la distancia y la estimación de las coordenadas de posición, Cricket proporciona una capacidad de orientación en interiores a través de la brújula Cricket.

Cricket proporciona buena precisión (hasta 25 cm de error) y operatividad aceptable en entornos con considerables interferencias.

2.4.3 *AeroScout*

El sistema AeroScout es un sistema de localización tanto para exteriores como para interiores. Está basado en las tecnologías WiFi y RFID implementadas con técnicas de localización de tiempo de llegada (*ToA*) y de nivel de señal recibida (*RSS)* [3].

El funcionamiento del sistema se basa en la emisión periódica de una señal por parte de la etiqueta RFID instalada en el emisor. Esta señal es recibida en los puntos de acceso y se procesa en un servidor central encargado de ejecutar las técnicas ToA y RSS permitiendo obtener la localización. Por último, esta localización es presentada de manera gráfica en monitores.

En la Figura 2-11 se muestra un esquema básico del sistema.

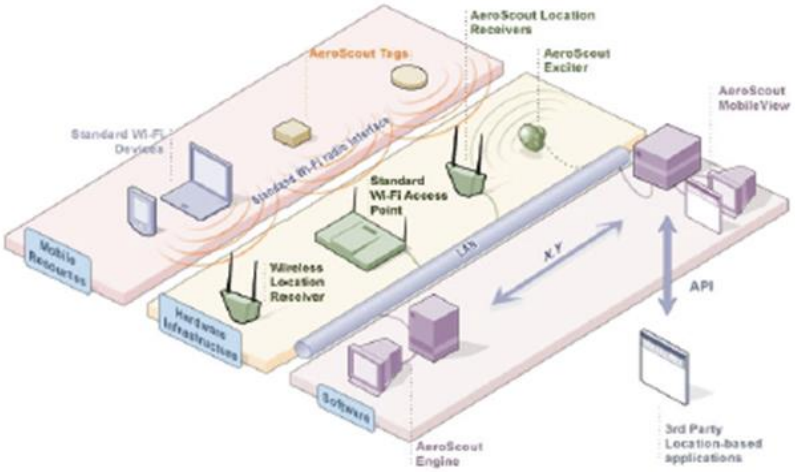


Figura 2-11. Esquema básico del sistema de localización AeroScout.

Este sistema muestra grandes ventajas al operar con dos tecnologías complementarias, permitiéndole ser una opción factible para un gran número de aplicaciones.

2.4.4 *Localización en entornos WiFi. Telefónica I+D*

El departamento de I+D de Telefónica ha desarrollado un proyecto basado en la localización WiFi en base a mediciones de nivel de potencia de señal recibida (RSS).

Implementa un método de análisis basado en el algoritmo de Monte Carlo, que tiene en cuenta la RSS, la arquitectura del entorno en el que se encuentra y el perfil del usuario. Además, este algoritmo considera datos estáticos y dinámicos del usuario y un sistema de filtrado para mejorar la precisión. La arquitectura del entorno donde se encuentra implementado el sistema debe ser

introducida como parámetro de análisis mediante formato BMP, al que se le añade información complementaria y limita el área de interés para la localización.

La calibración del sistema es realizada empíricamente mediante evaluación utilizando el software Matlab, dividiendo el entorno en celdas de un metro cuadrado y levantando el mapa de señal correspondiente a cada una de ellas.

El sistema permite la localización centralizada por parte de los puntos de acceso o la localización por parte de los propios terminales que portan los usuarios.

2.4.5 *Proyecto SHERLOC (Sistema Híbrido de Estrés Remoto y Localización)*

Este proyecto ha sido desarrollado por INDRA [8] y financiado por el Ministerio de Economía y Competitividad con el fin de crear un sistema de localización de personas que monitoriza las constantes vitales.

El sistema permite monitorizar la posición, situación de estrés y constantes vitales de agentes de policía, bomberos y servicios de protección civil durante las operaciones que llevan a cabo. También aumenta la seguridad de profesionales que realizan labores críticas, como por ejemplo, el personal que trabaja en plataformas petrolíferas, buques, la industria pesada, aeropuertos y hospitales.

Al usuario del sistema se le incorpora una pulsera o tobillera (Figura 2-12) con localizador y sensores que analizan los parámetros biométricos de la persona (tasa cardiaca y conductividad de la piel). La pulsera también dispone de un pulsador con el que el portador puede generar una alarma en caso de encontrarse en peligro. Desde un centro de control, se recibe la señal y se presenta sobre una aplicación cartográfica. Este centro puede llegar a gestionar miles de señales y centralizar la gestión de varios edificios y espacios.

Figura 2-12. Pulsera de usuario del sistema SHERLOC.

A nivel técnico, el sistema SHERLOC incorpora los últimos avances en sistema de localización en interiores (RSSI en RF, TOA en RF y RFID) y sistemas de posicionamiento global (GPS, GLONASS), lo que garantiza una señal de máxima precisión ininterrumpida.

Frente a los sistemas disponibles en el mercado, SHERLOC se comercializa a un coste razonable sin necesidad de calibrarse para poder localizar personas en entornos cerrados.

2.4.6 *Cosero*

El proyecto Cosero [9], llevado a cabo por la universidad alemana de Rheinische Friedrich-Wilhelms, trata de un robot inteligente capaz de desplazarse por entornos cerrados realizando diferentes labores que incluyen la localización de objetos y llevar a cabo de acciones sobre estos.

Figura 2-13. Robot inteligente COSERO.

Para esta labor de localización e interacción con objetos, implanta un sistema basado en *Bluetooth Low Energy* (BLE), debido principalmente a tres factores cruciales para la viabilidad del proyecto:

i. Baja potencia de transmisión, que se traduce en una mayor autonomía de los sensores *Bluetooth* implementados.
ii. Fácil estimación de la posición, debido al uso de técnicas de nivel de potencia de señal recibida (RSS).
iii. Bajo coste y tamaño de los elementos necesarios para montar los dispositivos localizadores.

Los componentes utilizados para desarrollar los dispositivos localizadores son los siguientes: placa de control Raspberry Pi, receptor *Bluetooth* USB y un adaptador WiFi USB para acciones secundarias gestionado por el Raspberry Pi [9].

El funcionamiento general del robot comienza por la petición de búsqueda de un objeto por parte de un usuario. El objeto, que debe llevar adosada una etiqueta *Bluetooth*, es localizado primeramente por los dispositivos localizadores fijos en el entorno (Raspberry Pi y adaptador *Bluetooth* USB). Tras la localización positiva del objeto, los dispositivos localizadores fijos envían la información al robot mediante tecnología *Bluetooth* y éste aplica las acciones correspondientes.

Este proyecto fue presentado y probado con éxito en el RoboCup German Open 2014.

2.4.7 *iBeacons Apple*

El sistema de posicionamiento en interiores *iBeacon* desarrollado por *Apple Inc*. está basado en una nueva clase de transmisores de bajo coste y bajo consumo capaces de localizar a dispositivos iOS7 por proximidad, permitiendo el envío de notificaciones *push* entre dispositivos.

El sistema se basa en la tecnología *Bluetooth* de baja energía (BLE) que se puede encontrar en dispositivos *Bluetooth* 4.0 que soporten modo dual. Trabaja como un sensor inalámbrico de posición/contexto en un entorno inteligente, como podría ser una tienda de un centro comercial (Figura 2-14), permitiendo obtener la localización del dispositivo. Emplea *Bluetooth Low Energy Proximity Sensing* [11] para transmitir un identificador único universal, recogido por un aplicación compatible o sistema operativo y que puede ser convertido en una localización física o generar una acción en el dispositivo de interés para el usuario (ofertas de productos, guía dentro de la tienda, etc.)

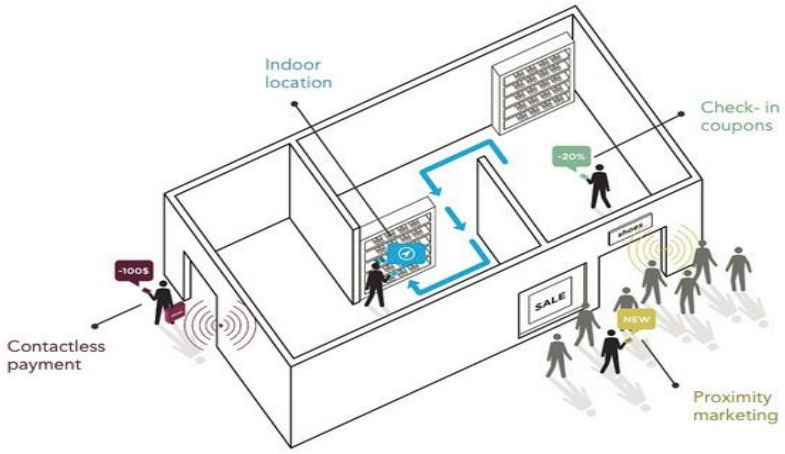


Figura 2-14. Sistema *iBeacon* implantado en comercio.

Las balizas o *beacons* pueden ser implementadas en diferentes formatos (dispositivos alimentados por pilas de botón, USB *sticks)* y versiones de software. Algunos ejemplos se muestran en la Figura 2-15.

Figura 2-15. Diferentes tipos de *iBeacon* tipo Estimote.

2.4.8 *Naval Automated Personnel Tracking (NAPT)*

Este proyecto se enmarca dentro del ámbito naval militar y tiene como principal objetivo la identificación de qué personal se encuentra embarcado o no, si se encuentra en cubiertas exteriores o si accede a compartimentos que presenten una amenaza.

El sistema ha sido desarrollado por la empresa australiana Blue Glue con el apoyo del programa de desarrollo tecnológico de Defensa australiano [12]. Utiliza tecnología RFID junto con una red Ethernet de sensores láser y de infrarrojo gestionada por un servidor central.

El sistema RFID realiza las funciones de confirmación de la identidad y la ubicación a bordo con una precisión de unos pocos metros. Las etiquetas del NAPT (portadas por los usuarios y mostradas en la Figura 2-16) tienen un alcance de hasta 50 metros y están equipadas con una batería

con autonomía de cinco años. Estas etiquetas están programadas para emitir pulsos automáticamente cada 1,5 segundos con los datos de identificación de cada usuario.

Figura 2-16. Etiqueta RFID portable sistema NAPT.

Por otro lado, los sensores infrarrojos y láser se activan con la intrusión de personal en un área peligrosa, informando cuando se atraviesa un haz de luz infrarroja o láser. Estos sistemas de detección de entrada pueden operar efectivamente sobre alcances de hasta 1,8 metros para rayos infrarrojos y 10 metros para láser.

Por último, el sistema es capaz de discriminar cuando un usuario abandona las cubiertas exteriores para entrar en la superestructura del buque y pasa a estar en un entorno seguro, mientras que, si la señal se pierde sin darse esta situación, determina que se ha producido una situación de "hombre al agua" activando las alarmas correspondientes.

2.4.9 *Sistema de Localización de Personal. Sistemas FABA*

Este sistema ha sido desarrollado por Sistemas FABA e implantado en los Buques de Acción Marítima (BAM). Utiliza la tecnología RFID y permite el control de personal a bordo.

El Sistema de Localización de Personal (SLP) permite la localización de los miembros de la dotación que acceden a zonas controladas por lectores de tarjetas RFID activas. Estas tarjetas tienen un identificador único y deben ser portadas por todo el personal embarcado. Los lectores se encuentran ubicados en zonas de acceso, cubiertas y locales, enviando la información recibida a *switches* que concentran y distribuyen las señales a las subestaciones del SLP más cercanas. En la Figura 2-17 se muestra un diagrama de conexiones del SLP.

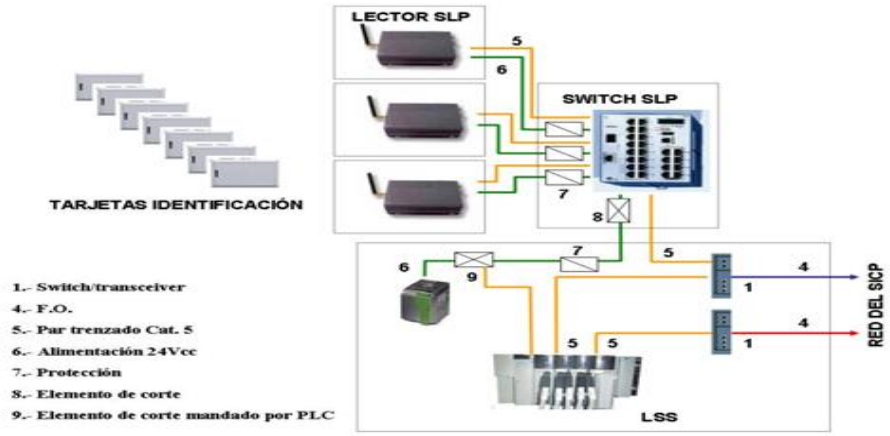


Figura 2-17. Infraestructura del Sistema de Localización de Personal.

El SLP se integra dentro del Sistema Integrado de Control de Plataforma (SICP) permitiendo el control de acceso a zonas restringidas, el control de embarco, desembarco y la localización de personal en tiempo real por compartimentos o zonas.

Como principales inconvenientes, el SLP necesita realizar reconfiguración de los lectores (ganancias de antenas) con cambios en la estructura del buque. Por otro lado, SLP integrado en SICP causa gran cantidad de falsas alarmas si los criterios de éstas no están correctamente definidos. Por último, el principal inconveniente de este sistema es la autonomía de las tarjetas RFID activas. Estas últimas no permiten el intercambio de baterías una vez agotadas debido a su estructura estanca.

2.5 Conclusión

Tras llevar a cabo una revisión técnica de las diferentes tecnologías estudiadas y desarrolladas hasta el momento, así como los diferentes métodos de cálculo de localización aplicados a dichas tecnologías, se puede considerar el estándar *Bluetooth* como una tecnología con tasa de bit, alcance, potencias de operación, arquitectura y relación coste/complejidad totalmente aceptables y factibles para la ejecución del presente proyecto.

Todo lo anterior queda reflejado en la variedad de proyectos desarrollados por importantes empresas del sector tecnológico, situando a *Bluetooth* como un estándar a utilizar en la localización en entornos cerrados en los que se requiera interactuar con el usuario.

En la Tabla 2-2 se muestra un cuadro resumen de los diferentes proyectos analizados.

PROYECTO	DESARROLLADOR	TECNOLOGÍA USADA	APLICACIONES
Sistema de localización por redes neuronales	*Universidad de Trento*	*WiFi*	*Localización en interior de edificios*
Cricket	*Oxygen y MIT*	*RF y ultrasonidos*	*Localización de personal en entornos cerrados*
AeroScout	*AeroScout Industrial*	*WiFi y RFID*	*Localización en interior y exterior*
Telefónica	*Departamento I+D Telefónica*	*WiFi*	*Localización y monitorización de personal en edificios*
Proyecto SHERLOC	*INDRA y Ministerio de Economía y Competitividad*	*RFID, GPS, GLONASS*	*Localización de personas y monitorización de constantes vitales*
Cosero	*Universidad Rheinische Friedrich-Wilhelms*	*Bluetooth LE, WiFi*	*Robot Inteligente*
iBeacon Apple	*Apple Inc*	*Bluetooth LE*	*Localización por proximidad e interactuación entorno/usuario*
Naval Automated Personnel Tracking (NAPT)	*Blue Glue*	*RFID, IR, Láser*	*Control y seguridad de personal en buques militares*
Sistema de Localización de Personal (SLP)	*Sistemas FABA*	*RFID*	*Localización de personal en buques de la Armada*

Tabla 2-2. Comparativa de proyectos en la actualidad.

3 ESTUDIO DE LA TECNOLOGÍA *BLUETOOTH*

Como se ha expuesto en la introducción, el presente proyecto tiene como objetivo la implementación de un sistema de balizas que permita el control de personal e intercambio de información en un entorno cerrado. Tras el análisis comparativo realizado en el Capítulo 2, la tecnología *Bluetooth* es la que mejor se ajusta a estas necesidades y objetivos siendo elegida para el desarrollo del sistema de balizas.

3.1 Introducción

La necesidad de empresas informáticas y de telecomunicaciones de desarrollar una interfaz abierta y de bajo coste para facilitar la comunicación entre dispositivos sin la utilización de cables, aprovechando la movilidad de los dispositivos inalámbricos, dio como resultado una tecnología cuyo nombre clave fue "*Bluetooth*".

El nombre procede del rey danés y noruego Harald Blåtand, cuya traducción al inglés es Harald *Bluetooth*, conocido por unificar las tribus noruegas, suecas-danesas y facilitar la comunicación entre ellos. La idea de este nombre fue propuesta por Jim Kardach que desarrolló un sistema que permite a dispositivos móviles comunicarse con ordenadores y unificar la interoperabilidad de los sistemas digitales. El logo de *Bluetooth* son las iniciales del nombre y apellido del rey Harald. La ᚼ (Hagall) (ᚼ) y la ᛒ (Berkana) (ᛒ) [6].

Figura 3-1. Logo tecnología *Bluetooth*.

3.1.1 *Historia*

Para entender el funcionamiento y las capacidades de esta tecnología es importante conocer los antecedentes de creación de *Bluetooth*. En 1994, la compañía de telecomunicaciones Ericsson comenzó un estudio para investigar la viabilidad de una interfaz de radio de baja potencia y bajo coste entre teléfonos móviles y sus accesorios. El objetivo era eliminar el cableado entre los teléfonos móviles y tarjetas de PC, dispositivos de audio, periféricos, etc.

El estudio fue parte de otro gran proyecto de investigación que tenía como objetivo la comunicación múltiple de varios dispositivos formando una red de teléfonos móviles. El último enlace en dicha conexión debería ser un radio enlace de corto alcance. A medida que el proyecto progresaba, se llegó a la conclusión de que las posibilidades que mostraba dicho enlace de corto alcance serían ilimitadas.

A comienzos de 1997, Ericsson se asocia con otros fabricantes de dispositivos portátiles para incrementar el interés en esta tecnología. El motivo era simple: para que el sistema fuera exitoso y verdaderamente útil, una cantidad importante de dispositivos portátiles deberían utilizar la misma tecnología de radioenlaces de corto alcance. En febrero de 1998, cinco importantes compañías compuestas por: Ericsson, Nokia, IBM, Toshiba e Intel, forman un *Special Interest Group* (SIG). Dicho grupo contenía la configuración ideal en lo referente al área de negocios: dos líderes del mercado en telefonía móvil, dos líderes del mercado en ordenadores portátiles y un líder del mercado en tecnología de procesamiento de señales digitales. La meta era establecer la creación de una especificación global para conectividad sin hilos de corto alcance.

El 20 y el 21 de mayo de 1998, el consorcio de *Bluetooth* se anunció al público general en Londres (Reino Unido), San José (California, EE.UU.) y Tokio (Japón), lo que provocó la adopción de la tecnología por varias compañías de interés en la época que comenzaron el desarrollo de nuevos dispositivos bajo la tecnología *Bluetooth*, estrenándose en el mercado en el año 2000.

3.1.2 *Special Interest Group (SIG)*

El *Special Interest Group* (SIG) en *Bluetooth* es una asociación privada sin ánimo de lucro con sede en Bellevue, Washington. A fecha de septiembre de 2014, el SIG está formado por más de 9000 compañías de telecomunicaciones, informática, automovilismo, música, textil, automatización industrial y tecnologías de red con pequeños grupos de personal en Hong Kong, Suecia y Estados Unidos. Los miembros del SIG dirigen el desarrollo de la tecnología inalámbrica *Bluetooth*, además de implementar y comercializar la tecnología en sus productos. Sin embargo, el *Bluetooth* SIG por sí mismo no fabrica ni vende dispositivos *Bluetooth*.

El SIG ha desarrollado varias versiones de estándares *Bluetooth* a lo largo de su historia, caracterizados principalmente por su retro compatibilidad, permitiendo al último estándar cubrir las versiones anteriores. Los estándares desarrollados hasta el momento son:

- ***Bluetooth* v1.0 y v1.0b (2001):** Las versiones 1.0 y 1.0b incluyen en el hardware de forma obligatoria la dirección del dispositivo *Bluetooth* (BD_ADDR) con el que se va a establecer la comunicación. Este hecho limitaba las capacidades de interoperabilidad en entornos *Bluetooth* de más de un dispositivo.

- ***Bluetooth* v1.1 (2002):** Fue la primera versión ratificada como estándar IEEE 802.15.1 (2002). Se corrigieron muchos errores en las especificaciones de la versión 1.0b, añadiéndose soporte para canales no cifrados y el indicador de señal recibida (RSSI).

- ***Bluetooth* v1.2 (2003):** Fue la primera versión compatible con la versión 1.1 presentando importantes mejoras:

  i. Una conexión más rápida y capacidad *Discovery* (detección de otros dispositivos *Bluetooth*).

  ii. *Adaptive Frequency Hopping* (AFH) que mejora la resistencia a las interferencias de radio frecuencia, evitando el uso de las frecuencias ya utilizadas en la secuencia de saltos.

iii. Mayor velocidad de transmisión (721 Kbit/s) que en versiones anteriores.

iv. *Extended Synchronous Connections* (ESCO), que mejoran la calidad de la voz de los enlaces de audio al permitir la retransmisión de paquetes corruptos, y, opcionalmente, puede aumentar la latencia de audio para proporcionar un mejor soporte para la transferencia de datos simultánea.

v. *Host Controller Interface* (HCI), apoyo a la comunicación en serie de tres hilos UART (*Universal Asynchronous Receiver-Transmitter).*

vi. Introdujo el control de flujo y los modos de retransmisión de L2CAP (*Logical Link Control and Adaptation Protocol*).

- ***Bluetooth* v2.0 + EDR (2004):** Esta versión de la especificación *Bluetooth* fue lanzada en 2004 y es compatible con la versión anterior 1.2. La principal diferencia está en la introducción de una velocidad de datos mejorada EDR (*Enhanced Data Rate*). La tasa nominal de EDR es de 3 Mbit/s, aunque la tasa de transferencia de datos práctica desciende a 2,1 Mbit/s. EDR utiliza una combinación de GFSK (*Gaussian Frequency Shift-Keying*) y modulación PSK (*Phase Shift Keying*). Además, puede proporcionar un menor consumo de energía a través de un ciclo de trabajo reducido.

- ***Bluetooth* v2.1 + EDR (2007):** La versión 2.1 de la especificación *Bluetooth* + EDR es totalmente compatible con la versión 1.2, y fue adoptada por el *Bluetooth* SIG el 26 de julio de 2007. Introdujo la mejora en el emparejamiento de dispositivos *Bluetooth* con *Secure Simple Pairing* (SSP) aumentando la seguridad en el enlace. Incluye un modo *inquiry* de bajo consumo que proporciona información ampliada en la búsqueda de dispositivos.

- ***Bluetooth* v3.0 + HS (*High Speed)* (2009):** La versión 3.0 + HS de la especificación *Bluetooth* fue aprobada por el *Bluetooth* SIG el 21 de abril de 2009. El estándar *Bluetooth* 3.0+HS soporta velocidades teóricas de transferencia de datos de hasta 24 Mbit/s entre sí, aunque no a través del enlace *Bluetooth* propiamente dicho. La conexión *Bluetooth* nativa se utiliza para la negociación y el establecimiento mientras que el tráfico de datos de alta velocidad se realiza mediante un enlace 802.11. Su principal novedad es AMP (*Alternate* MAC/PHY), que añade el estándar 802.11 como transporte de alta velocidad. Inicialmente, estaban previstas dos tecnologías para incorporar en AMP: 802.11 y UWB (*Ultra Wide Band*), pero finalmente UWB no se encuentra en la especificación. Además, esta nueva versión introdujo la transmisión *unicast* de datos sin establecer conexión entre dispositivos.

- ***Bluetooth* v4.0 (2010):** El SIG de *Bluetooth* completó la especificación del Núcleo de *Bluetooth* en su versión 4.0 en 2010, que incluye al *Bluetooth* clásico, el *Bluetooth* de alta velocidad y los protocolos *Bluetooth* de bajo consumo. El *Bluetooth* de alta velocidad se basa en 802.11 y el *Bluetooth* clásico consta de protocolos *Bluetooth* preexistentes. El *Bluetooth* de baja energía (*Bluetooth Low Energy* o BLE) es un subconjunto de *Bluetooth* v4.0 con una pila de protocolos completamente nueva para desarrollar rápidamente enlaces sencillos. Como alternativa a los protocolos estándar de *Bluetooth* que se introdujeron en *Bluetooth* v1.0 a v4.0, está dirigido a aplicaciones de muy baja potencia con dispositivos alimentados con pilas botón. Los diseños de los chips permiten dos tipos de implementación, de modo dual y de modo único. El 17 de diciembre de 2010, el *Bluetooth* SIG adoptó la tecnología *Bluetooth* de bajo consumo como el rasgo distintivo de la versión 4.0 que llegó a conocerse como Wibree. Los nombres provisionales Wibree y *Bluetooth* ULP (*Ultra Low Power*) fueron abandonados y el nombre BLE comenzó a ser utilizado ampliamente. A finales de 2011, se presentaron los nuevos logotipos *Smart Bluetooth Ready* para los anfitriones y *Smart Bluetooth* para los sensores de menor tamaño basados en la tecnología BLE.

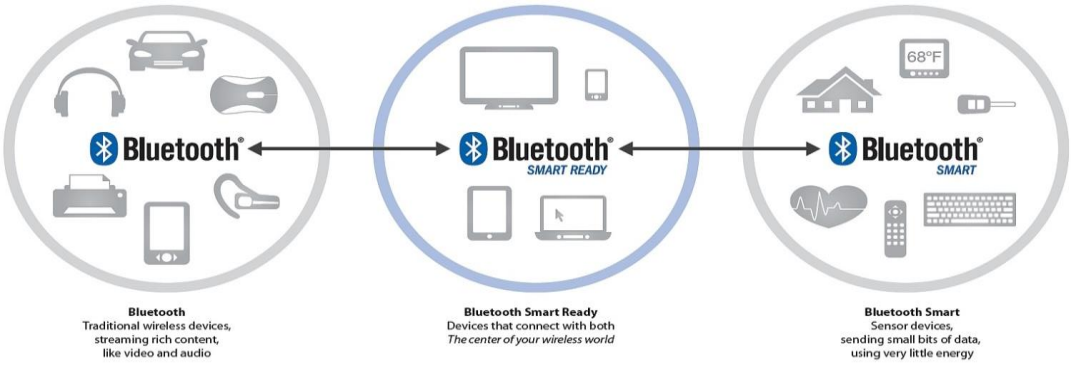


Figura 3-2. Esquema de capacidades de *Bluetooth* 4.0.

- ***Bluetooth* v4.1 (2014):** Esta versión presentada en 2014 presenta como novedades la optimización en la conexión y reconexión de dispositivos sin la necesidad de que el usuario deba interactuar con los dispositivos. Además permite que un dispositivo funcione tanto como periférico y *hub* de datos a la par, mejorando el soporte para convivir con otros protocolos como *Long Term Evolution* (LTE) (estándar de comunicación móvil).

- ***Bluetooth* v4.2 (Actualidad):** En el presente se encuentra en desarrollo la versión 4.2, que según apunta el SIG, incluirá la capacidad de operación con IPv6 mejorando el uso del Internet de las cosas y manteniendo los principios de bajo coste y bajo consumo energético.

Dado que *Buetooth* es la tecnología adoptada en este trabajo de fin de grado, es necesario un conocimiento de esta tecnología *Bluetooth* al completo. Por tanto, en los siguientes apartados se realizará un estudio en profundidad de *Bluetooth Clásico* y *Bluetooth Low Energy* con el fin de explotar en lo posible sus capacidades y funcionalidades en la implementación del sistema *iBeacon*.

3.2 *Bluetooth* clásico

La tecnología *Bluetooth* clásica está principalmente inspirada en el estándar *IEEE 802.11* [22], utilizando el modelo de red *ad-hoc* dentro de este estándar.

El modo *ad-hoc* hace referencia a una red simple donde la comunicación es establecida entre varias estaciones dentro de un área de cobertura sin la necesidad de configurar puntos de acceso comunes. Emplea métodos de solicitud arbitraria de utilización de los medios de la red, con el fin de maximizar el rendimiento de todos los usuarios en un conjunto de servicios base de ésta.

Estas redes *ad-hoc* son utilizadas para establecer *Wireless Personal Area Network* (WPAN) que, basadas en la tecnología *Bluetooth*, definen otro estándar conocido como *IEEE 802.15.1*. Las conexiones entre redes WPAN *Bluetooth* presentan dos tipos de topologías diferentes:

- **Picorredes WPAN *Bluetooth*:** Una picorred está formada por dispositivos *Bluetooth* que actúan como maestros en la picorred y por uno o más dispositivos que actúan como esclavos. Los dispositivos esclavos solo se comunican con el maestro en una estructura punto a punto bajo el control del mismo. Por otro lado, las transmisiones del maestro pueden ser punto a punto (*unicast*) o bien, transmisiones de punto a multipunto (*multicast*). Durante una sesión de comunicación, un elemento esclavo puede comportarse como maestro en un momento dado o viceversa. En la Figura 3-3 se muestra un ejemplo de tipos de picorredes.

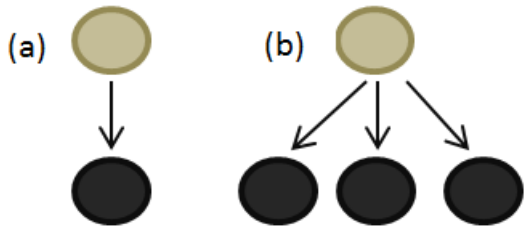


Figura 3-3. Diferentes tipos de picorredes: (a) *Unicast* y (b) *Multicast.*

- **Redes dispersas WPAN *Bluetooth*:** Una red dispersa está compuesta de varias picorredes que se solapan en tiempo y espacio. Son conocidas como *scatternet* y permiten que un aparato pueda ser esclavo en varias picorredes a la vez que es maestro en una sola de ellas. En la figura 3-4 se muestra un ejemplo de red dispersa.

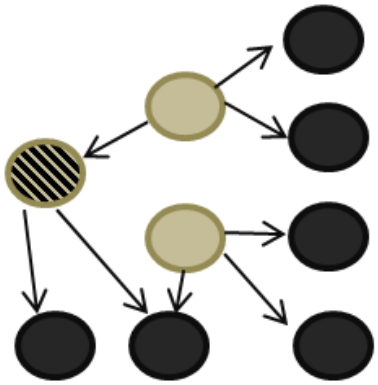


Figura 3-4. Red WPAN *Bluetooth* dispersa (*scatternet*).

La comunicación entre dispositivos *Bluetooth* dentro de una WPAN sigue unos patrones definidos por el estándar 802.15.1 donde la frecuencia de operación de los enlaces se establece en la banda ISM (*Industrial, Scientific and Medical*) de 2.4 GHz. Se utilizan transceptores de saltos de frecuencia (1600 saltos/s) para evitar interferencias y caídas de señal en combinación con la modulación binaria GFSK (*Gaussian Frequency Shift Keying*) para transmitir símbolos con una tasa de 1 símbolo/segundo. Además se utiliza la técnica de *time slot* en la que los canales de comunicación están divididos en ranuras de 625 µs por las que se envían paquetes de información con una frecuencia diferente siguiendo una secuencia esperada. Esto permite la comunicación full-dúplex entre dispositivos. En la Figura 3-5 se muestra una secuencia básica de transmisión de paquetes entre un maestro y un esclavo en una red WAN *Bluetooth*.

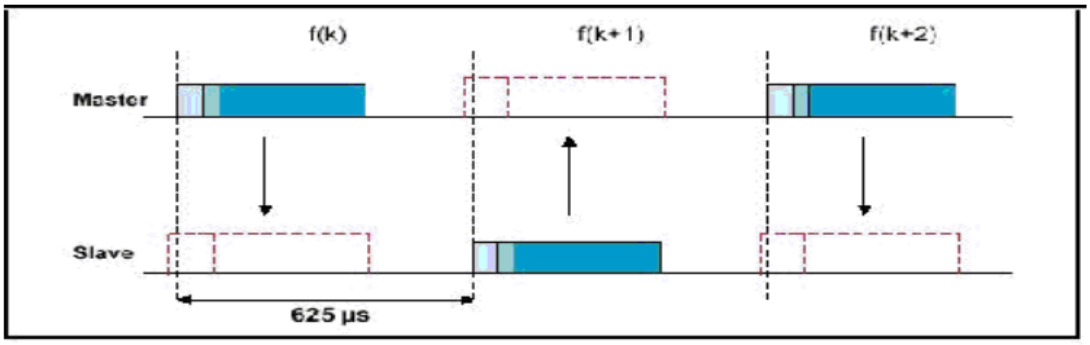


Figura 3-5. Técnica *time slot* y salto de frecuencia (*frequency hopping*) entre maestro y esclavo.

Este patrón común de comunicación entre diferentes dispositivos no es más que un protocolo definido por el estándar IEEE 802.15.1 a implementar en los dispositivos que utilicen esta tecnología.

3.2.1 *Pila de protocolos de Bluetooth clásico*

El mecanismo de comunicación *Bluetooth* está formado por diferentes capas que apiladas dan como resultado la pila de protocolos *Bluetooth*. Esta pila está formada por dos grupos de capas diferentes:

- **Capas específicas del protocolo *Bluetooth*:** Capas obligatorias y necesarias para implementar una comunicación *Bluetooth* entre dispositivos. Incluyen: *Logical Link Control and Adaptation Protocol* (L2CAP), *Host Control Interface* (HCI), *Link Manager Protocol* (LMP), Banda Base, *Link controller/audio* y capa física (Radio).

- **Capas no específicas del protocolo *Bluetooth*:** Capas no obligatorias y adicionales que pueden ser implementadas para permitir la interoperabilidad con la capa de aplicación. Incluyen: *Object Exchange Protocol* (OBEX), *Wireless Application Protocol* (WAP), capa RFCOMM (*Radio Frequency Communications)*, el *Telephony Control Service* (TCS) y la capa *Service Discovery Protocol* (SDP).

En la Figura 3-6 se muestra la estructura de una pila de protocolos *Bluetooth*.

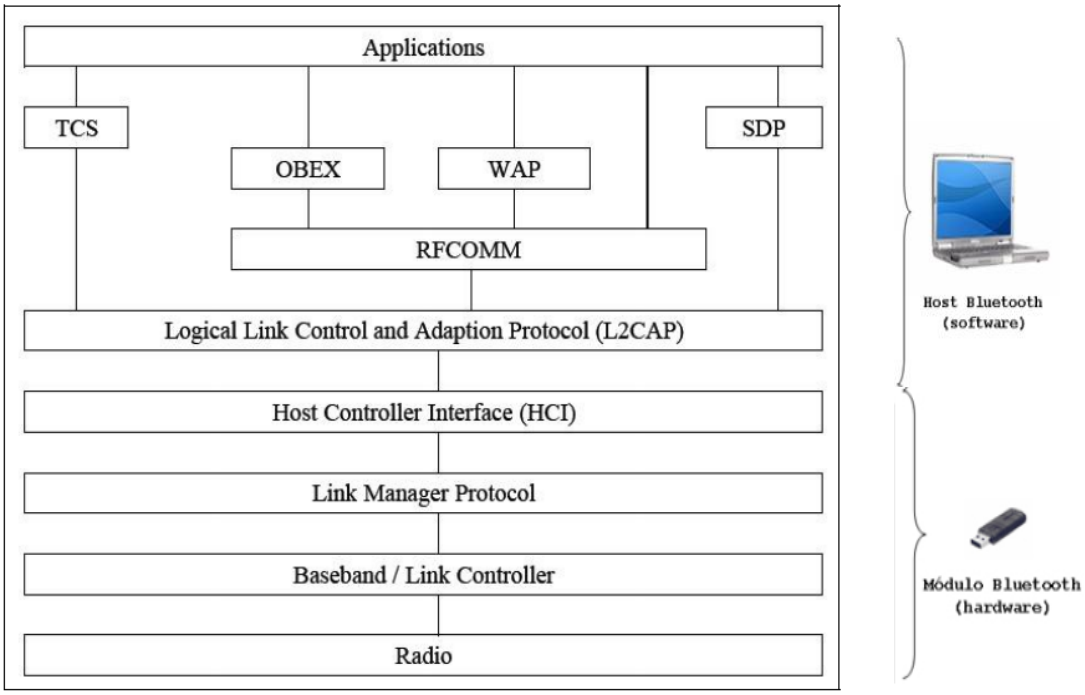


Figura 3-6. Pila de protocolo *Bluetooth* clásico.

La parte inferior de la pila compuesta por Radio, Banda Base, *Link Controller* y *Link Manager Protocol* hace referencia al controlador *Bluetooth* definido a nivel *hardware*, mientras que las capas superiores incluyen al *host Bluetooth* y se definen a nivel software. Controlador y *host* se comunican entre sí mediante la interfaz HCI.

3.2.2 *Controlador Bluetooth*

En este apartado se analizan las capas inferiores de la pila de protocolos *Bluetooth* gestionadas por el controlador.

3.2.2.1 Radio

La capa radio *Bluetooth* opera en la banda ISM 2,4 GHz y define las diferentes necesidades de los dispositivos *Bluetooth* en su comunicación.

En esta capa se divide el espectro de frecuencias en 79 o 23 canales en los que se realizan saltos de frecuencia durante la transmisión, incluyendo una banda de guarda en cada canal con el fin de evitar las transmisiones fuera de banda. En la Tabla 3-1 se muestra un ejemplo de la división del espectro de frecuencias *Bluetooth* en diferentes países.

País	Rango de Frecuencia	Canales Disponibles
Europa & EEUU	2400 - 2483.5 MHz	2402 to 2480 MHz (79 canales)
Japón	2471 - 2479 MHz	2473 to 2495 MHz (23 canales)
España	2445 - 2475 MHz	2449 to 2471 MHz (23 canales)
Francia	2446.5 - 2483.5 MHz	2454 to 2476 MHz (23 canales)

Tabla 3-1. Frecuencias *Bluetooth* por países.

En lo referente a la potencia de transmisión y recepción de los diferentes dispositivos *Bluetooth*, se pueden encontrar tres clases diferentes como se puede observar en la Tabla 3-2.

Clase	**Potencia máxima permitida**		**Potencia mínima permitida**		**Distancia**
	mW	**dBm**	**mW**	**dBm**	
Clase 1	100 mW	20 dBm	1 mW	0 dBm	100 metros
Clase 2	2'5 mW	4 dBm	0'25 mW	-6 dBm	10 metros
Clase 3	1 mW	0 dBm	N/A	N/A	1 metros

Tabla 3-2. Clases de dispositivos *Bluetooth*.

Existen dispositivos con la capacidad de optimizar la salida de energía en una conexión *Bluetooth*, midiendo el RSSI (*Receiver Signal Strength Indicator*) y comparándolo con el valor nominal de referencia de 0 dBm y aumentando o disminuyendo la potencia de transmisión si es necesario.

La señal *Bluetooth* presenta una modulación del tipo GFSK con un índice de modulación entre 0.28 y 0.35, donde un 1 binario se representa con una desviación positiva de frecuencia, y un 0 binario se representa con una desviación negativa. En la Figura 3-7 se muestra un esquema del bloque modulador GFSK.

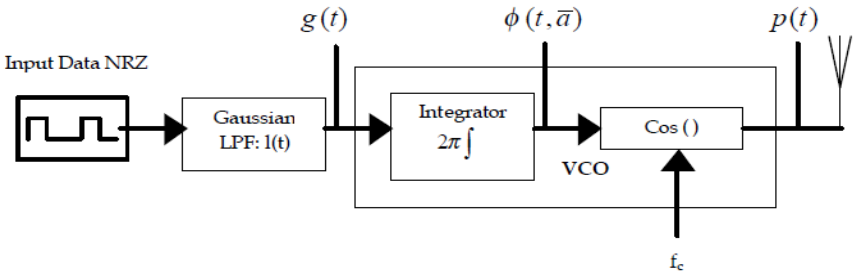


Figura 3-7. Transmisor con modulación GFSK.

Esta modulación GFSK (Figura 3-8) limita el error al cruce por cero a un valor inferior a 1/8 para un periodo de símbolos. Además, la desviación mínima de frecuencia correspondiente a una secuencia 1010 debe ser menor a un 80% de la desviación en frecuencia que presenta la secuencia 00001111 sin estar nunca por debajo de los 115 kHz. Debido a estas limitaciones, la tasa de transmisión de datos para una conexión *Bluetooth* se limita a 1Mbps.

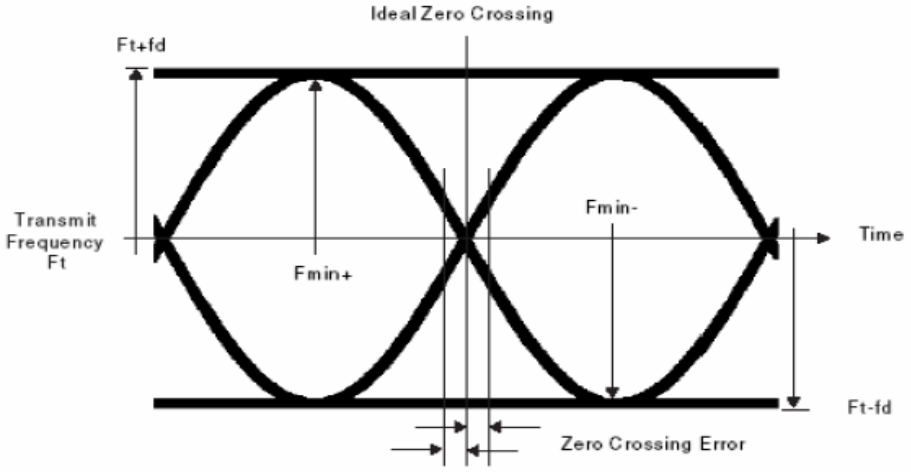


Figura 3-8. Modulación GFSK.

3.2.2.2 Banda Base

La capa superior o banda base controla las operaciones sobre bits y paquetes de datos, realiza detección y corrección de errores y cifrado de datos. También es la capa encargada de emitir peticiones y confirmaciones de repeticiones de las transmisiones en recepción.

Cada canal *Bluetooth* está representado por una secuencia de saltos seudo aleatorios a través de los canales de radiofrecuencia que utiliza la tecnología. Dos o más dispositivos que usan el mismo canal o *slot* forman una picorred, que, como se explicó anteriormente, está compuesta por un maestro y esclavos. El maestro es el responsable de la sincronización entre los dispositivos de la piconet, su reloj y saltos de frecuencia controlan al resto de los dispositivos, encargándose además del procedimiento de búsqueda y establecimiento de las conexiones. Los esclavos se sincronizan con el maestro y siguen la misma secuencia de saltos impuesta por el maestro. Cabe destacar que entre redes *scatternet* no existe este tipo de sincronización, por tanto, los esclavos son capaces de pertenecer a varias piconet haciendo uso de la multiplexación por división del tiempo (TDD) [23], permaneciendo activo en solo una piconet a la vez.

A la hora de gestionar los enlaces físicos entre dispositivos, la banda base maneja dos tipos principales:

- ***Synchronous Connection-Oriented* (SCO):** Enlace simétrico punto a punto entre maestro y un solo esclavo con canales o slots reservados e intervalos regulares preestablecidos. Son utilizados generalmente para enlaces con transmisión de voz.

- ***Asynchronous Connection-Less* (ACL):** Enlace punto a multipunto entre maestro y varios esclavos sobre canales o slots no reservados para SCO. Es un enlace no orientado a la conexión entre dispositivos que busca la transferencia de datos sin requerimientos temporales, por tanto, presenta posibilidad de pérdida de paquetes que obliga a implementar retransmisiones.

Además de los enlaces físicos, existen tres canales lógicos utilizados para transporte de información: UA para transporte de información asíncrona, UI para la isócrona y US para la síncrona.

Por otro lado, existen dos canales digitales utilizados para el control y gestión a nivel enlace: LCC (*Link Control Channel*) y LM (*Link Manager*). Estos canales lógicos son utilizados por la capa superior que se explica a continuación.

3.2.2.3 Gestión de enlaces: Link Manager Protocol

La tercera capa del protocolo *Bluetooth* es la encargada de la gestión en el establecimiento y finalización de las conexiones, la autenticación en caso necesario, el control de tráfico, la planificación en la conexión, la gestión del consumo y la total supervisión del enlace.

El *Link Manager Protocol* filtra e interpreta los diferentes mensajes bajo este protocolo que maestro y esclavo comparten en la conexión con información de: seguridad, energía, ciclos de trabajo y estado de la conexión dentro de la piconet. Cada uno de los dispositivos *Bluetooth* dentro de una piconet debe tener su propio LMP que realiza su función de proveedor de servicios en conjunto con el controlador de enlace (*Link Controller*) de la capa inferior. El controlador de enlace gestionado por el LMP es el encargado de ejecutar el procedimiento de búsqueda y establecimiento inicial de la conexión (Figura 3-9), que será de dos tipos según la información que el maestro tenga o no del esclavo:

- **Pregunta (*Inquiry*):** Este procedimiento permite al maestro descubrir qué dispositivos se encuentran en su zona de cobertura, determinando el reloj de todos aquellos esclavos que respondan a su mensaje de búsqueda. El maestro podrá indicar en su mensaje de búsqueda el tipo de esclavo con el que quiere establecer la conexión, para ello se utiliza el código de acceso de pregunta o *Generic Inquiry Access Code* (GIAC) y los códigos de acceso dedicados o *Dedicated Inquiry Access Code* (DIAC). Estos códigos son emitidos en diferentes saltos de frecuencia y, una vez recibidos por los esclavos, estos últimos entrarán en modo *inquiry response*, enviando al maestro un paquete con los parámetros del dispositivo incluyendo el paquete FHS (*Frequency Hopping Synchronization*) necesario para establecer la conexión.

- **Búsqueda (*Paging*):** Este procedimiento pregunta por la dirección de un dispositivo *Bluetooth* con el que se quiere establecer la conexión. Este identificador del dispositivo puede obtenerse de la respuesta de un *inquiry*, puede ser introducido por un usuario o pre-programado por el fabricante del dispositivo. El maestro envía un código de acceso o *Dedicated Access Code* (DAC) con la identificación del esclavo en diferentes canales y conmuta a escucha hasta recibir una copia del DAC enviada por el esclavo. Tras esto, el maestro envía el paquete FHS y conmuta a escucha hasta recibir el ACK del esclavo, siendo este último el que ajusta su reloj y establece la conexión con el maestro.

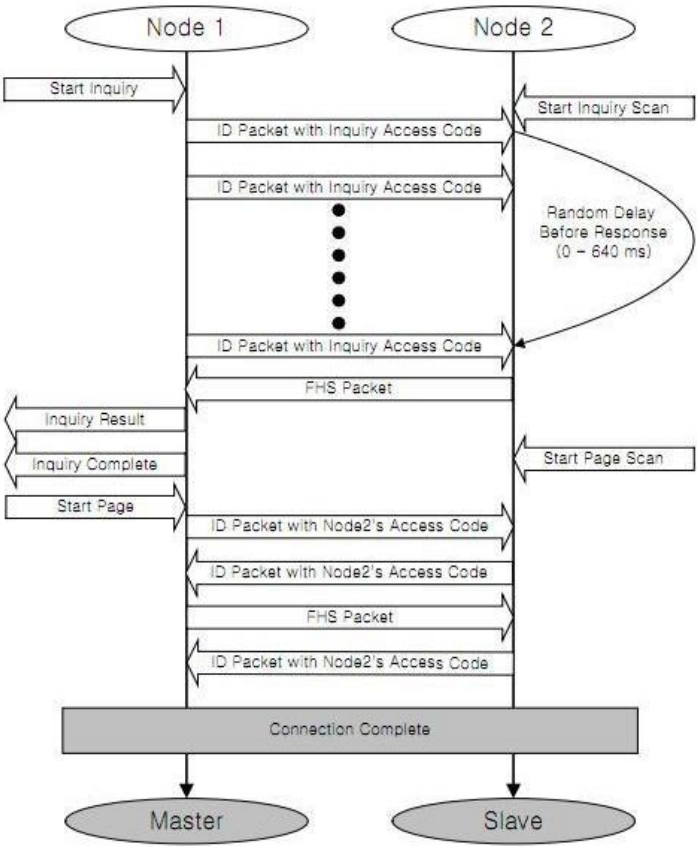


Figura 3-9. Procedimientos *inquiry* y *paging*.

Los mensajes que se intercambian entre los gestores de enlace (LM) tienen formato de unidades de datos de protocolo PDU (*Protocol Data Unit*) con una prioridad más alta que los datos que se quieran transmitir de capas superiores (L2CAP), con el fin de dar fiabilidad a la gestión del enlace entre dos dispositivos *Bluetooth*.

3.2.2.4 Host Controller Interface

El controlador de interfaz del host (HCI) permite el acceso mediante línea de comandos a la capa de banda base y LMP por parte de las capas superiores como se muestra en la Figura 3-10. Se encarga de controlar y recibir información acerca del estado del hardware *Bluetooth* y los registros de control de conexiones. Está compuesto por tres partes principales:

- **El *firmware* HCI:** Programa oficial del fabricante que forma parte del hardware *Bluetooth*.
- **El controlador HCI:** *Driver* para control que se encuentra en el software del dispositivo *Bluetooth*.
- **El *Host Controller Transport Layer:*** Encargado de la conexión entre *firmware* y *driver*.

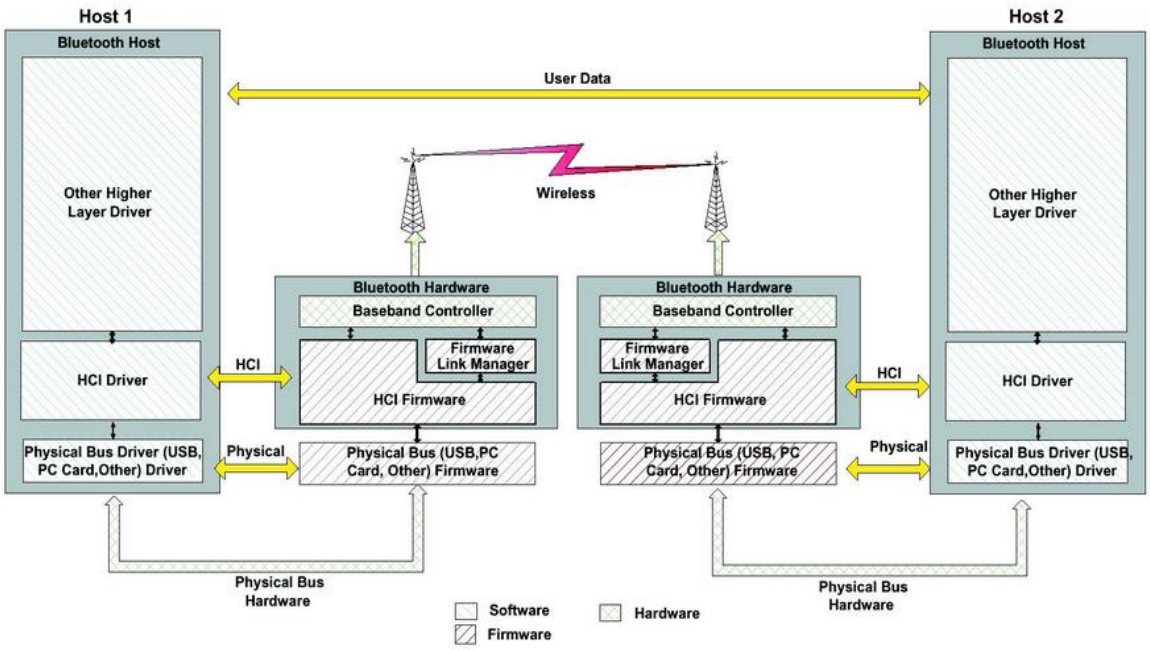


Figura 3-10. HCI dentro del controlador *Bluetooth*.

3.2.3 *Host Bluetooth*

3.2.3.1 Capa L2CAP

La capa *Logical Control and Adaptation Protocol* (L2CAP) es la encargada de proporcionar tanto los servicios de datos orientados a conexión como los no orientados a conexión de los protocolos de las capas superiores, junto con funcionalidades de multiplexación, segmentación y reensamblaje.

L2CAP acepta paquetes de hasta 64 KB provenientes de las capas superiores y los divide en tramas para transmitirlos. Las tramas se reensamblan nuevamente en paquetes en el otro extremo. En la Figura 3-11 se muestra el formato de una trama *Bluetooth*.

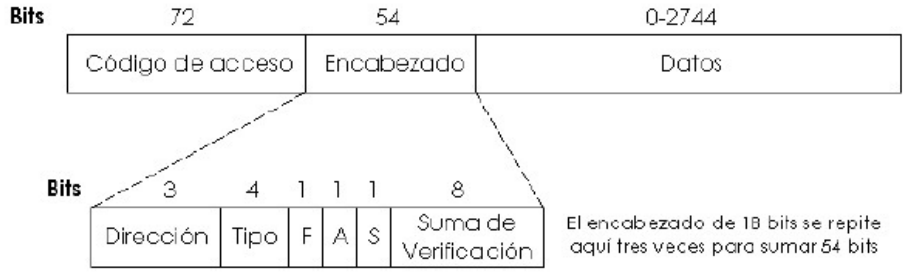


Figura 3-11. Trama de datos *Bluetooth* [24].

Esta capa se encarga además de la calidad de servicio, tanto al establecer los enlaces como durante el modo normal de operación. Para ello, durante el establecimiento de los enlaces se negocia el tamaño máximo de carga útil permitido, para evitar que un dispositivo que envíe paquetes grandes sature a uno que recibe paquetes pequeños.

3.2.3.2 RFCOM

El protocolo RFCOM es utilizado para establecer comunicaciones series punto a punto, emulando el protocolo RS-232 (intercambio de datos binarios entre terminales de datos) a través de RF y suministrando capacidades de transporte a los niveles superiores que utilizan puerto serie como mecanismo de transporte (OBEX y WAP) [25].

3.2.3.3 Service Discovery Protocol (SDP)

Este protocolo define como una aplicación cliente puede descubrir los servicios, atributos y propiedades que posee un servidor de aplicaciones. En la Figura 3-12 se muestra el proceso de petición y respuesta entre un cliente y un servidor bajo protocolo SDP.

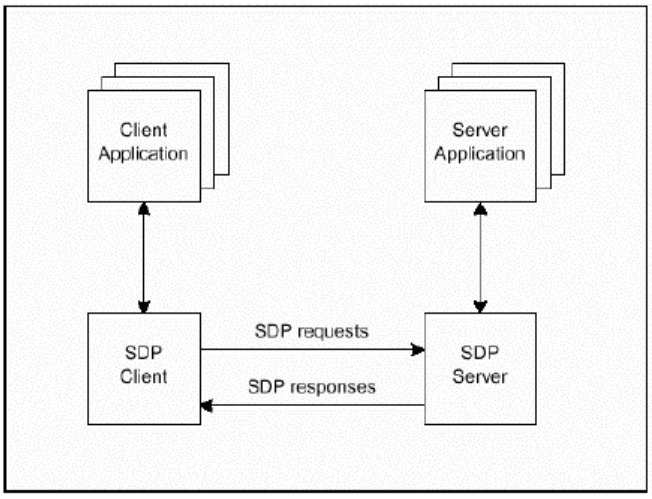


Figura 3-12. ***Service Discovery Protocol.***

SDP proporciona un mecanismo para el descubrimiento de servicios y sus atributos asociados, pero no proporciona ningún mecanismo ni protocolo para la utilización de dichos servicios.

3.2.3.4 Protocolo TCP

El *Telephony Control Protocol* (TCP) define la señalización para el control de llamadas de voz y datos entre dispositivos *Bluetooth*.

El protocolo TCP se basa en el estándar Q.931, que es un protocolo de control de telefonía desarrollado para el control de llamadas en la red telefónica pública (ISDN: *Integrated Services Digital Network*). El protocolo de control de llamada Q.931 define los mensajes y formatos de mensajes de control que son creados por los dispositivos de comunicación. Algunos de los tipos comunes de información contenidos en los mensajes Q.931 incluyen el establecimiento de llamada y mensajería, control de llamada a grupos y el control de mensajes de señalización.

Debido a que el protocolo Q.931 se ha utilizado de forma fiable en el sistema telefónico durante muchos años, el sistema *Bluetooth* ha adaptado gran parte de los mensajes de protocolo Q.931 para el procesamiento de llamadas.

3.2.4 Seguridad Bluetooth

Como en cualquier método de comunicación vía RF, la seguridad es un aspecto muy a tener en cuenta.

La tecnología *Bluetooth* establece tres diferentes modos de seguridad en el acceso genérico entre dispositivos *Bluetooth*:

- **Modo seguridad 1:** En este modo no se iniciará ningún proceso de seguridad.

- **Modo seguridad 2:** Se inicia el procedimiento de seguridad tras haberse establecido el canal de comunicación entre dispositivos.

- **Modo seguridad 3:** Se inicia el procedimiento de seguridad antes de que el canal de comunicación haya sido establecido.

El procedimiento de seguridad se lleva a cabo a nivel enlace estableciendo cuatro parámetros esenciales para mantener la seguridad en la conexión:

- **Dirección del dispositivo *Bluetooth* (BD_ADDR):** La dirección es asignada por el IEEE y tiene una longitud fija de 48 bits y es única para cada dispositivo *Bluetooth*.

- **Clave privada de autenticación de usuario:** Clave de 128 bits utilizada en el proceso de autenticación.

- **Clave privada de cifrado:** Clave de entre 8 y 128 bits obtenida de la clave de autenticación tras el proceso de autenticado.

- **Número aleatorio (RAND):** Número en constante cambio de valor que es derivado de un proceso aleatorio en la unidad *Bluetooth*.

En el proceso de establecimiento de conexión entre dispositivos se produce una transferencia de los anteriores parámetros entre maestro y esclavos, que permite la autenticación de los dispositivos y el establecimiento de la conexión bajo un enlace seguro.

3.3 *Bluetooth* 4.0: *Bluetooth Low Energy*

La tecnología *Bluetooth Low Energy* (BLE) o también conocida como *Bluetooth Smart* fue desarrollada por el SIG de *Bluetooth* con el fin de establecer enlaces de comunicación sencillos entre dispositivos portátiles. Es una tecnología de bajo consumo energético y de bajo coste implementada en dispositivos con necesidad de transferir una baja cantidad de datos a una distancia cercana de hasta 30 metros.

BLE reduce notablemente la potencia de pico a 15 mW y la potencia media a 5µW del *Bluetooth* clásico. Además realiza una gestión del consumo de energía en modo espera, utiliza menos canales RF para la comunicación entre dispositivos e implementa protocolos de comunicación simples, alcanzando eficiencias energéticas que pueden llegar a ser 20 veces mayores que las de *Bluetooth* clásico. En la Tabla 3-3 se muestra una comparativa entre *Bluetooth* Clásico y BLE.

TECNOLOGÍA / CARACTERISTÍCA	*Bluetooth Low Energy*	*Bluetooth Clásico*
Frecuencia RF	2,4 GHz	2,4 GHz
Distancia	Hasta 30 metros	Hasta 100 metros
Tasa de datos	1 Mbps	3 Mbps
Nodos / Esclavos activos	Ilimitados	7 / 16777184
Seguridad	Cifrado de bloques 128b	Cifrado de bloques 64b/128b
Tiempo de envío de datos	3 ms	100 ms
Capacidad de voz	No	Sí
Topología de red	Estrella - bus	*Scatternet*
Pico de potencia	<15 mW	<30 mW
Servicio de detección	Sí	Sí
Usos primarios	Balizas informativas, red de sensores, industria , etc.	Telefonía móvil, periféricos, audio *streaming*, etc.

Tabla 3-3. Comparativa entre BLE y *Bluetooth* clásico.

La tecnología BLE no está pensada para el *streaming* de datos. Con una tasa de transferencia de datos de 1 Mbps está diseñada para el envío de pequeñas tramas de datos. Estas pequeñas tramas de datos pueden ser leídas por un cliente BLE en cualquier momento y contienen información de diferentes balizas, sensores o dispositivos que implementen BLE. Por ejemplo, termómetros, podómetros, grupos fecha-hora, sensores de proximidad, etc, como se muestra en la Figura 3-13.

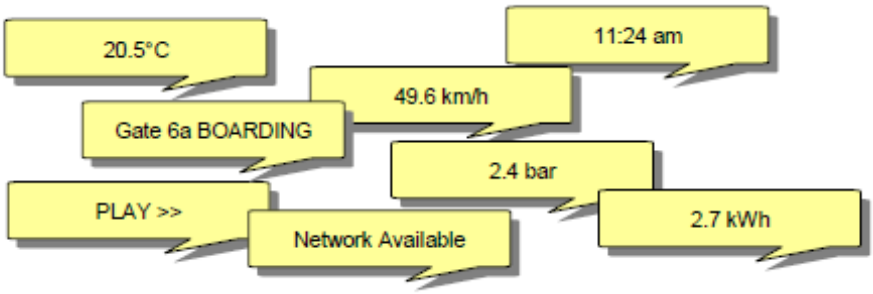


Figura 3-13. Datos de eventos locales con dispositivos BLE.

Los dispositivos BLE operan en dos modos principales según la tecnología que utilicen:

- Dispositivos operando en *modo dual*, que utilizan tanto *Bluetooth* clásico como BLE y que llevan implementados los correspondientes protocolos de cada tecnología.

- Dispositivos operando en *modo simple*, que sólo se comunican a través de protocolo BLE.

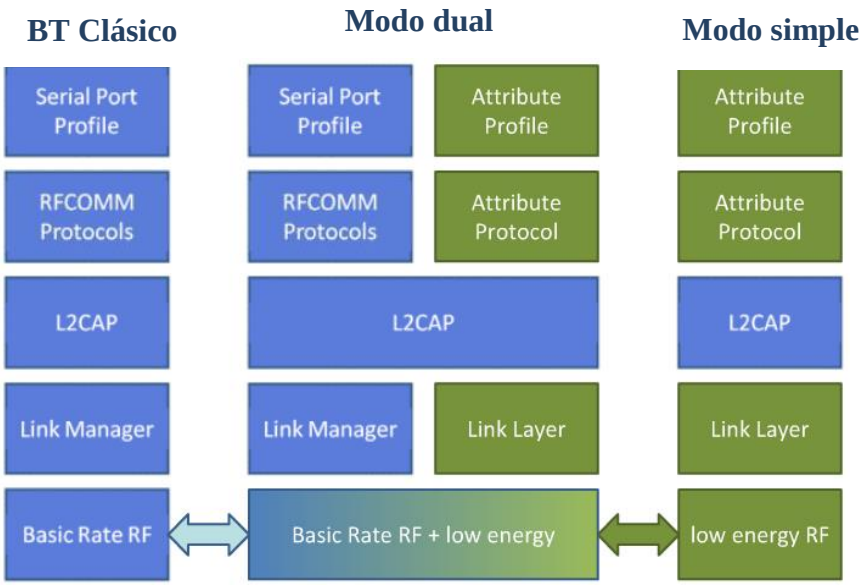


Figura 3-14. Comunicación entre protocolos de dispositivos modo dual y modo simple.

En la Figura 3-14 se muestra la estructura de comunicación entre las diferentes capas para los dos modos de operación explicados anteriormente. Los dispositivos en modo dual y modo simple introducen una nueva pila de protocolos específica de BLE que será estudiada en el siguiente apartado.

3.3.1 *Pila de protocolos BLE*

La tecnología BLE implementa una nueva pila de protocolos ajustada a los servicios y capacidades que esta tecnología ofrece. La pila de protocolos está dividida en dos partes principales al igual que *Bluetooth* clásico: *Controlador* y *host.*

El *controlador* opera en las capas inferiores de la pila (capa de enlace, capa física y HCI) y es el encargado de la captura de los paquetes de datos y de la gestión de las transmisiones en RF de los dispositivos BLE. Las capas superiores (aplicación, protocolo de atributos y L2CAP) son operadas por el *host*, mientras que la comunicación entre *controlador* y *host* es posible gracias al HCI.

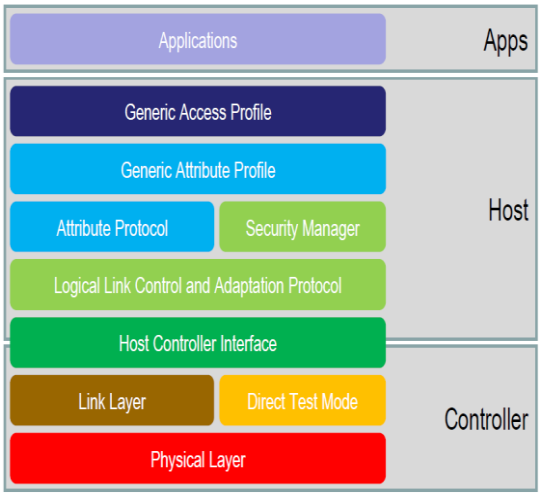


Figura 3-15. Pila de protocolos BLE.

En la Figura 3-15 se muestran las diferentes capas del protocolo específico de BLE que serán explicadas a continuación.

3.3.2 *Controlador BLE*

3.3.2.1 Capa física (Physical Layer)

Bluetooth LE opera en la banda ISM de 2,4 GHz con 40 canales separados 2 MHz entre ellos. Es capaz de transmitir a una velocidad de 1 Mbit/s utilizando la modulación GFSK al igual que *Bluetooth* clásico. Utiliza saltos de frecuencia adaptable. Utiliza 3 de los 40 canales para el modo *Low Energy Advertising* (LEAd), que permiten la detección de dispositivos, los 37 canales restantes son utilizados para la transmisión de datos una vez descubiertos y establecida la comunicación entre ellos. En la Figura 3-16 se muestran los 40 canales BLE dentro del espectro de frecuencias reservado para la tecnología *Bluetooth*.

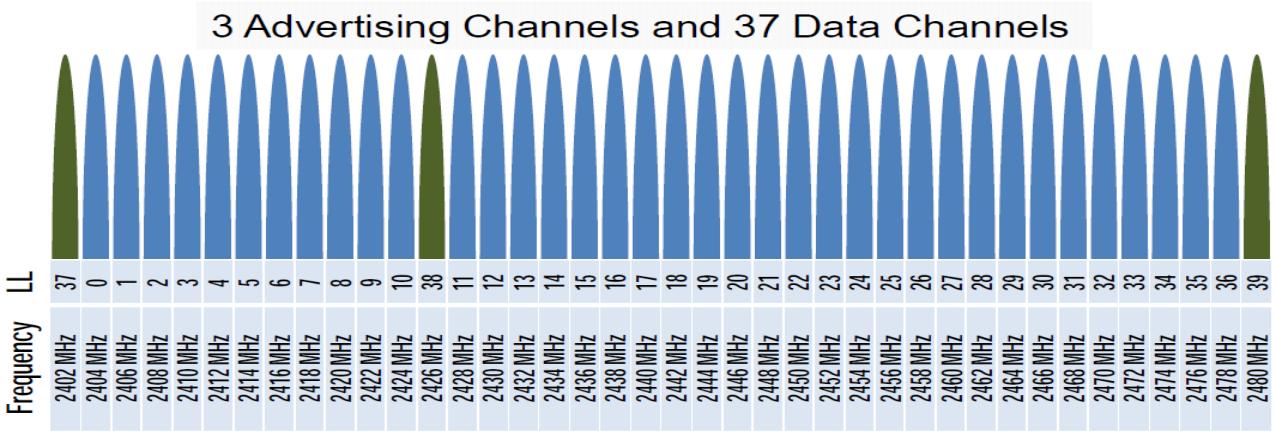


Figura 3-16. Canales BLE.

La utilización de tres canales de entre los 40 totales para el modo LEAd implica una drástica reducción en el tiempo de escaneo para la búsqueda de nuevos dispositivos con los que establecer conexión. En *Bluetooth* clásico se invierten 22.5 ms de media en el proceso de escaneo frente a los valores de entre 0.2 y 1.6 ms que utiliza BLE para completar este proceso, lo que se traduce en una notable disminución de consumo energético en los dispositivos.

3.3.2.2 Capa de detección de modos

Existen cinco modos básicos en los que un dispositivo BLE puede operar:

- **Modo stand-by:** El dispositivo no transmite ni escanea ningún paquete de datos.

- **Modo escaneo:** El dispositivo se encuentra a la escucha de paquetes en los tres canales reservados para el modo LEAd. Un dispositivo en este modo toma el nombre de *escáner*.

- **Modo LEAd:** El dispositivo se encuentra transmitiendo un paquete de datos con formato LEAd en los 3 canales reservados para este modo. Al mismo tiempo, se encuentra a la escucha de peticiones para ampliación de información o establecimiento de conexión.

- **Modo *Initiating*:** El dispositivo se encuentra a la escucha de paquetes LEAd de un dispositivo específico al que se quiere conectar, respondiendo con un paquete de datos iniciador de la conexión.

- **Modo conexión:** El dispositivo se encuentra conectado a otro en 37 canales reservados para la transmisión de datos. En este modo, los dispositivos conectados asumen dos roles diferentes. El dispositivo que se encontraba en modo *initiating* asume el rol de maestro, que es el encargado del establecimiento y la sincronización de la conexión con uno o varios dispositivos. El que se encontraba en modo LEAd asume el rol de esclavo y sólo se comunicará con el maestro. Cabe destacar que un maestro puede recibir información de hasta 2^{31} esclavos, pero solo establecer conexión con 7 esclavos dentro de una piconet.

La capa de detección de modos es la encargada de detectar el modo de trabajo de los dispositivos emisores de los paquetes de datos recibidos. Una vez identificado dicho modo, esta capa informa a la capa de enlace, que es la encargada de gestionar los diferentes modos en los que opera el dispositivo BLE receptor.

3.3.2.3 Capa de enlace (Link Layer)

La capa de enlace es la encargada de gestionar los diferentes procedimientos que se llevan a cabo para cada uno de los modos descritos anteriormente.

La tecnología BLE subdivide el canal físico en unidades de tiempo conocidas como eventos. De esta manera, los dispositivos transmiten datos por medio de paquetes que son posicionados en estos eventos de tiempo, existiendo dos tipos de eventos gestionados por la capa de enlace: los eventos *advertising* y los eventos de conexión y transmisión de datos.

Los paquetes BLE posicionados en cada evento siguen una estructura similar tanto en un evento *advertising* como en un evento de conexión y transmisión de datos (Figura 3-17). Este paquete de datos está compuesto por:

- **Preámbulo (1 byte):** Contiene datos para la sincronización entre dispositivos.

- **Dirección de acceso (4 bytes):** Contiene los códigos de accesos para el enlace físico entre dispositivos (direcciones MAC, indicador de modo de operación, etc.).

- ***Payload Data Unit* (PDU) (Hasta 39 bytes):** Contiene una cabecera con los identificadores para el enlace y transporte lógico entre dispositivos. Además, contiene un *payload* de datos a transmitir que varía en longitud dependiendo del tipo de evento: *advertising* o conexión y transmisión de datos.

- ***Cyclic Redundancy Check* (CRC):** Código de detección de errores de 3 bytes.

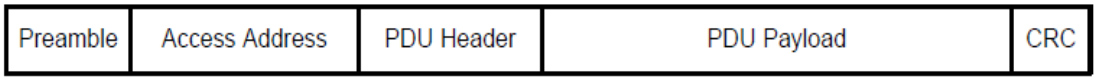


Figura 3-17. Paquete de datos BLE.

A continuación, se van a definir las diferentes PDU y procedimientos llevados a cabo tanto en los eventos *advertising* como en los eventos transmisión de datos:

i. Evento *advertising:*

La PDU del evento *advertising* viene definida por el *Bluetooth* SIG [25] con una cabecera de 16 bits y un *payload* que depende del tipo de *advertising* PDU entre 7 diferentes:

1. **ADV_IND**: evento *advertising* no dirigido a un dispositivo en concreto y con capacidad de establecer conexión. Longitud de 6 a 37 bytes.
2. **ADV_DIRECT_IND**: evento *advertising* dirigido a un dispositivo concreto y con capacidad de conexión. Longitud de 12 bytes.
3. **ADV_NONCONN_IND**: evento *advertising* no dirigido a un dispositivo en concreto y sin capacidad de establecer conexión. Longitud de 6 a 37 bytes.
4. **SCAN_REQ**: enviado por la capa de enlace en modo escaneo y recibido por una capa de enlace en modo LEAd. Longitud de 12 bytes.
5. **SCAN_RSP**: enviado por la capa de enlace en modo LEAd y recibido por una capa de enlace en modo escaneo. Longitud de 6 a 37 bytes.
6. **CONNECT_REQ**: enviado por la capa de enlace en modo *initiating* y recibido por una capa de enlace en modo LEAd. Longitud de 34 bytes.
7. **ADV_SCAN_IND**: evento *advertising* no dirigido a un dispositivo en concreto con ampliación de información ante un escaneo. Longitud de 6 a 37 bytes.

El evento *advertising* se realiza de forma secuencial en todos los canales disponibles con intervalos de repetición de una PDU comprendido entre 20 ms y 10.24 segundos dependiendo de la configuración. Un dispositivo en modo escaneo en el otro lado debe estar configurado con una ventana de análisis (tiempo de escaneo) y un intervalo de exploración (con qué frecuencia se realiza el escaneo) igual o inferior a 10.24 seg.

Para que un evento *advertising* pueda realizar una conexión entre dispositivos, el dispositivo en modo escaneo debe suministrar al dispositivo en modo LEAd dos parámetros:

1. **Intervalos de conexión:** determinan el instante de inicio de los eventos de conexión y la secuencia de intercambio de datos.
2. **Latencia del esclavo:** es la cantidad de intervalos de conexión que un esclavo puede ignorar.

Con el análisis de esos parámetros los emisores o receptores de un evento *advertising* establecen los intervalos en los que deben operar en el modo correcto, pasando a estar en modo *stand-by* en aquellos intervalos en los que no deban realizar ninguna acción. Esto tiene como resultado una disminución considerable del consumo energético.

En la Figura 3-18 se muestran, de forma gráfica, un ejemplo de eventos *advertising* de PDU ADV_IND y SCAN_REQ SCAN_RSP.

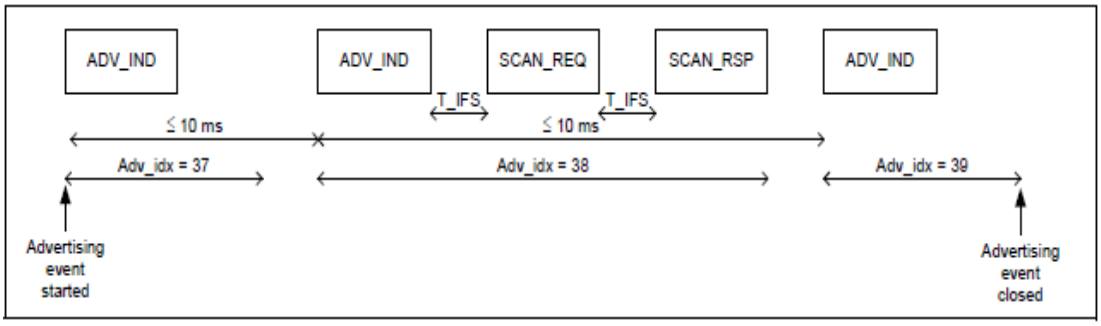


Figura 3-18. Evento *advertising* con PDU ADV_IND y SCAN_REQ SCAN_RSP.

ii. Evento de conexión y transmisión de datos:

La PDU que el *Bluetooth* SIG define para el evento de conexión es: CONNECT_REQ de 34 bytes. Esta PDU es enviada por un iniciador a un dispositivo en modo LEAd y contiene información

de los diferentes parámetros establecidos entre las capas de enlace que van a realizar la conexión (*Link Layer Data*) [25]. Estos parámetros son:

1. *AA*: Define la dirección de acceso de la capa de enlace del iniciador.

2. *CRCInit*: Valor inicial aleatorio generado por la capa de enlace que permite el posterior cálculo de CRC una vez establecida la conexión.

3. *WinSize*: Indica la *transmitWindowSize* o tamaño de la ventana de escucha que el esclavo de la conexión debe configurar para recibir la PDU completa.

4. *WinOffset*: Indica la *transmitWindowOffset* o momento de inicio de la ventana de escucha que el esclavo debe configurar para recibir la PDU completa.

5. *Interval*: Indica el intervalo entre *transmitWindowOffset.*

6. *Latency*: Indica al esclavo el número máximo de CONNECT_REQ consecutivos en los que debe permanecer a la escucha. Una vez superado ese valor, el esclavo puede pasar a modo *stand-by* hasta un nuevo evento de conexión.

7. *Timeout:* Indica al esclavo el tiempo máximo que puede pasar entre la recepción de dos PDU de datos consecutivos.

8. *ChM:* Indica el mapa de canales que se van a utilizar durante la transmisión de datos una vez establecida la conexión.

9. *Hop:* Indica el algoritmo que se va a seguir para realizar los saltos de frecuencias en los canales elegidos para la conexión y transmisión de datos.

10. *SCA:* Indicador utilizado para informar al esclavo de la precisión del reloj utilizado por el maestro en la conexión.

En la Figura 3-19 se muestra, de forma gráfica, un ejemplo de evento de conexión PDU CONNECT_REQ con los parámetros explicados anteriormente.

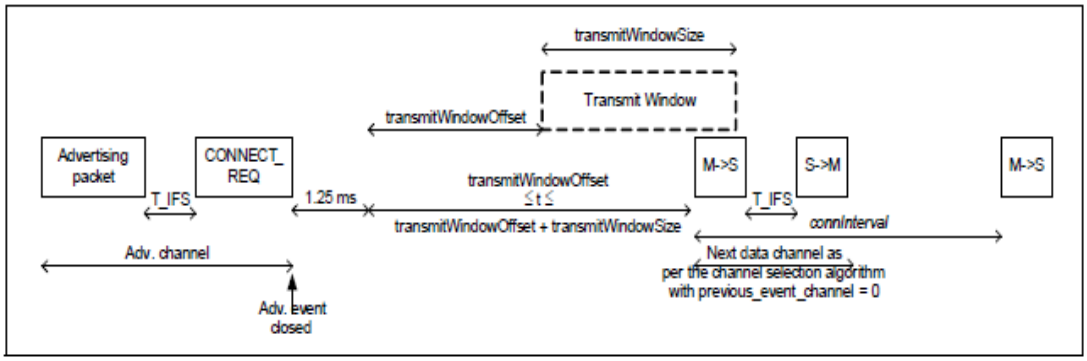


Figura 3-19. Evento de conexión PDU CONNECT_REQ.

Después de que la conexión se ha establecido con los saltos de frecuencia correspondientes y en los 37 canales reservados por BLE para transmisión de datos, aparece el evento de transmisión de datos entre maestro (M) y esclavo (S) con una PDU que *Bluetooth* define como: *Data Channel PDU.*

Esta PDU está compuesta por:

1. Cabecera: se encarga de informar al esclavo del tipo de PDU y la longitud de ésta. Longitud de 2 bytes.

2. *Payload:* puede ser LL Data PDU para la transmisión de datos de la capa L2CAP o puede ser LL Control PDU para el control de la conexión en la capa de enlace (actualizarla, resetearla, pausarla, finalizarla, etc.). Longitud de 0 a 22 bytes.

3. *Message Integrity Check* (MIC): se utiliza para asegurar la integridad y autenticidad de los datos transmitidos. Longitud de 4 bytes.

La conexión entre dos dispositivos termina cuando la capa de enlace del maestro o del esclavo envía una LL_TERMINATE_IND que inicia el proceso voluntario de desconexión.

3.3.2.4 HCI

Al igual que en *Bluetooth clásico,* el HCI se encarga de la comunicación por comandos entre el *host* y el controlador BLE. En el caso de la tecnología *Low Energy* se añaden comandos para los modos escáner y el modo *LEAd.*

3.3.3 *Host BLE*

3.3.3.1 Logical Link Control and Adaptation Layer (L2CAP)

El L2CAP utilizado en BLE es un protocolo optimizado y simplificado basado en el L2CAP de *Bluetooth clásico*. En BLE, el objetivo principal de esta capa es multiplexar los datos de los tres protocolos de las capas superiores: ATT, GATT y SMP.

Los datos de estos servicios son manejados por L2CAP en un enfoque de *best effort* sin el uso de retransmisiones ni mecanismos de control de flujo como los utilizados en otras versiones de *Bluetooth*.

L2CAP en BLE no utiliza ni la segmentación ni el reensamblado, ya que los protocolos de la capa superior proporcionan paquetes de datos que encajan en el tamaño máximo de la carga útil L2CAP que es igual a 23 bytes.

3.3.3.2 Security Manager Protocol (SMP)

Este protocolo se encarga de la autenticación y el cifrado de las conexiones. Utiliza el algoritmo de encriptación AES-128 [26] para generar el cifrado.

Es ejecutado en el dispositivo maestro, que es el encargado de establecer la conexión y distribuir la clave de cifrado, permitiendo al esclavo establecer una conexión segura.

3.3.3.3 Attribute Protocol (ATT)

El protocolo de atributos de BLE permite a un dispositivo exponer e intercambiar ciertos módulos de información conocidos como atributos.

Esta interacción entre dispositivos para la gestión de atributos da lugar a los roles ATT de cliente y servidor. El cliente inicia la petición de información acerca de un atributo, mientras que el servidor es quien responde con la información solicitada. Tanto maestro como esclavo (roles definidos a nivel enlace físico) pueden asumir el rol de cliente o el rol de servidor.

Los atributos son valores almacenados en registros internos de los dispositivos y que pueden tener una longitud variable de entre 1 y 512 bytes. A cada atributo se le asigna una dirección ATT

llamada *handle* a través de la cual la información se hace accesible a un cliente. Esta información contenida en un atributo toma el nombre de característica.

A los atributos también se les asocia un identificador único según el tipo de información que contengan, llamado *Universal Unique Identifier* (UUID). Si el atributo es definido por el *Bluetooth* SIG, el UUID tiene una longitud de 16 bits, mientras que si es definido por un fabricante fuera del SIG, el UUID tendrá una longitud máxima de 128 bits.

Tanto el *handle* como el UUID permiten al cliente acceder a la información del servidor.

ATT define los procedimientos para la comunicación entre un servidor ATT y un cliente. Los tres principales procedimientos de comunicación son:

- ***Read Characteristic Value:*** El cliente pide una característica al servidor utilizando la dirección *handle* de dicha característica.

- ***Read Using Characteristic:*** El cliente pide todos los valores de una característica utilizando su UUID.

- ***Write Characteristic Value:*** El cliente escribe un valor en el servidor mediante la dirección *handle* de dicho valor y siempre que éste tenga permiso de escritura.

Cada atributo contiene, además, datos sobre los permisos de escritura y/o lectura que le asigne el dispositivo servidor. En la Figura 3-20 se muestran algunas características definidas en *Bluetooth* SIG.

Aerobic Heart Rate Lower Limit	org.bluetooth.characteristic.aerobic_heart_rate_lower_limit	0x2A7E
Aerobic Heart Rate Upper Limit	org.bluetooth.characteristic.aerobic_heart_rate_upper_limit	0x2A84
Aerobic Threshold	org.bluetooth.characteristic.aerobic_threshold	0x2A7F
Age	org.bluetooth.characteristic.age	0x2A80
Alert Category ID	org.bluetooth.characteristic.alert_category_id	0x2A43
Alert Category ID Bit Mask	org.bluetooth.characteristic.alert_category_id_bit_mask	0x2A42
Alert Level	org.bluetooth.characteristic.alert_level	0x2A06
Alert Notification Control Point	org.bluetooth.characteristic.alert_notification_control_point	0x2A44
Alert Status	org.bluetooth.characteristic.alert_status	0x2A3F

Figura 3-20. Características definidas por *Bluetooth* SIG con su dirección *handle* (Derecha).

3.3.3.4 Generic Attribute Profile (GATT)

GATT se sitúa sobre el protocolo ATT y es el encargado de establecer operaciones comunes y una estructura para los datos almacenados por el protocolo de atributos. GATT gestiona los roles de servidor y cliente, no siendo una tarea exclusiva de esta capa, ya que la capa superior (*Generic Access Profile*), también puede asumir esta tarea. GATT y ATT no son capas de transportes exclusivas de BLE, por lo que pueden ser implementadas por *Bluetooth clásico.* Sin embargo, sí son obligatorias en los dispositivos que usen tecnología BLE.

El GATT del dispositivo que asume el rol de servidor almacena los datos transportados desde el ATT y acepta las peticiones, comandos y confirmaciones del GATT del dispositivo con el rol de cliente. El servidor envía respuestas a las peticiones cuando está configurado para ello, envía además

indicaciones y notificaciones de forma asíncrona al cliente cuando ocurren eventos en el servidor. GATT también se encarga de especificar el formato de los datos contenidos en el servidor.

El GATT establece la estructura que deben tomar los atributos para ser transportados sobre el enlace lógico entre los GATT de servidor y cliente. Esta estructura es generada en dos pasos:

1. En primer lugar, GATT formatea los atributos en servicios. Un servicio es una colección de características y comportamientos asociados a estas características que permiten a un dispositivo llevar a cabo sus funciones. Existen dos tipos de servicios: los primarios y secundarios [27], que están directamente relacionados con las funciones primarias o secundarias del dispositivo.

2. En segundo lugar, GATT encapsula los diferentes servicios en lo que se conoce como perfil. Un perfil está compuesto por uno o varios servicios y definen el conjunto de aplicaciones y usos que un dispositivo BLE puede ofrecer en función de sus servicios, características y las relaciones que se pueden establecer entre ellos en las capas inferiores.

A continuación, en la Figura 3-21 se muestra la estructura de un perfil GATT, mientras que en la Figura 3-22 se muestra un ejemplo de servicio ofrecido por un dispositivo BLE.

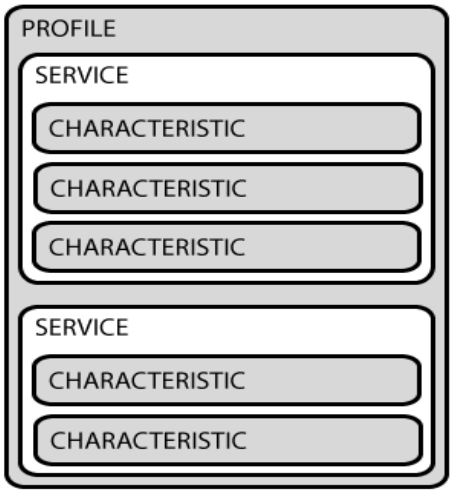


Figura 3-21. Estructura de un perfil GATT.

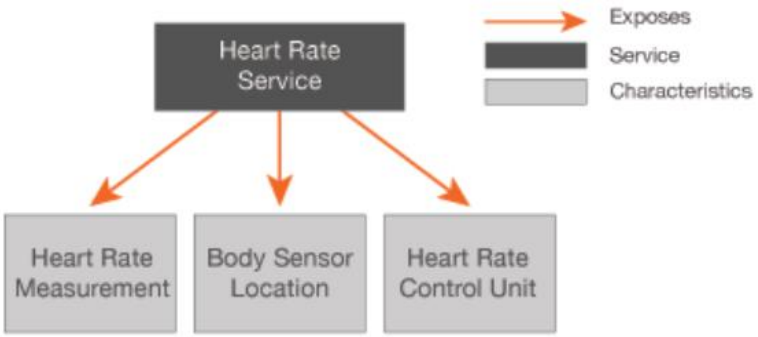


Figura 3-22. Ejemplo de servicio BLE con sus correspondientes características.

3.3.3.5 Generic Access Profile (GAP)

GAP se comunica directamente con la capa de aplicación ofreciéndole los diferentes perfiles que el dispositivo BLE utiliza para llevar a cabo sus funcionalidades. Según lo ordenado por la capa de aplicación, GAP se encargará de gestionar las capas inferiores para definir los diferentes roles que asume el dispositivo a la hora de comunicarse con su entorno. Estos roles pueden definirse sobre tres niveles:

1. Rol según perfil:

i. *Broadcaster:* Dispositivo funcionando exclusivamente en modo LEAd y llevando a cabo un evento *advertising.*

ii. *Observer:* Dispositivo en escucha y recepción de un evento *advertising.*

iii. *Peripheral:* Dispositivo con capacidad de transmitir y recibir eventos *advertising* pero que solamente puede recibir y no transmitir eventos de conexión. Asume el rol de esclavo tras establecer la conexión.

iv. *Central:* Dispositivo con capacidad de transmitir y recibir eventos *advertising* y de transmitir eventos de conexión. Asume el rol de maestro tras establecer la conexión.

2. Rol según estado:

i. Reconocible u oculto.
ii. Conectable o no conectable.

3. **Rol según privacidad:**

i. Muestra la dirección privada del dispositivo.
ii. No muestra la dirección privada del dispositivo.

Según el rol que defina el GAP, un dispositivo BLE podrá interactuar o no interactuar con el resto de dispositivos a su alrededor.

3.3.4 *Usos de BLE*

Debido a su bajo consumo, robustez, simplicidad y capacidades, BLE se está convirtiendo en la tecnología de referencia en un gran número de dispositivos en la actualidad. Como se ha visto en apartados anteriores, BLE es una tecnología abierta que permite el desarrollo de perfiles y características por aquellas empresas que lo deseen. Esto se traduce en un crecimiento constante y en la aparición continua de nuevas aplicaciones.

A continuación, se nombran algunas de las aplicaciones de BLE en la actualidad:

- **Deporte y actividad física:** Controladores de ritmo cardiaco, podómetros, relojes, auriculares sin cables, etc.
- **Sensores:** temperatura, humedad, alarmas, etc.
- **Medicina:** Medidores de presión sanguínea, medidores de glucosa en sangre, etc.
- **Entretenimiento:** controles remoto, periféricos de ordenadores y consolas, etc.
- **Automatización:** robots industriales, monitorización de parámetros de maquinaria, control de luces en hogares, etc.

- **Seguridad:** llaves inteligentes, sensores de proximidad, etc.
- **Publicidad:** puntos de información, anuncios, mapas interiores, etc.
- **Posicionamiento en interiores:** *iBeacon*.

Analizadas la tecnología *Bluetooth* clásica y la tecnología *Bluetooth Low Energy* y sus capacidades, en los siguientes apartados se expone el desarrollo del sistema *iBeacon* para el control de personal e intercambio de información en interiores.

4 DESARROLLO DEL SISTEMA *iBEACON*

En este capítulo de la memoria se describe el sistema *iBeacon* y se enumeran los pasos seguidos en la configuración del hardware y software (Apartado 1.4) necesarios para desarrollar el sistema de localización *iBeacon* basado en *Bluetooth Low Energy*.

4.1 Concepto *iBeacon*

iBeacon se define como un sistema de posicionamiento en interiores, compuesto por transmisores de bajo consumo y bajo coste, que permite la interacción mediante notificaciones entre un entorno inteligente y los usuarios que se encuentran en él.

4.1.1 *Arquitectura*

El funcionamiento de *iBeacon* está basado en la tecnología de transmisión de datos a corta distancia utilizando *Bluetooth Low Energy* (BLE). La comunicación de un *iBeacon* usando BLE se realiza mediante la transmisión periódica de una trama de datos o PDU con la estructura indicada en la Figura 4-1.

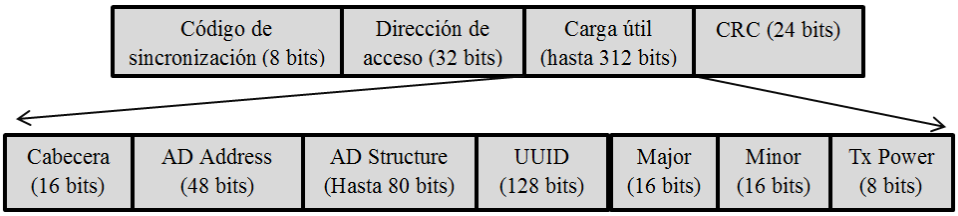


Figura 4-1. Estructura de una trama de un *iBeacon* sobre BLE.

El código de sincronización proporciona los datos de tiempo que utiliza el *iBeacon* para transmitir, utilizados por el dispositivo receptor para referenciar la recepción de señal.

La dirección de acceso contiene dos tipos de datos: por una parte, datos fijos y preestablecidos, que identifican el canal estándar utilizado para comunicaciones en modo *Low Energy* y, por otra parte, datos variables del canal utilizado por el *iBeacon* para realizar conexiones.

La carga útil es la parte de la trama que incluye todo lo necesario para que los dispositivos receptores puedan realizar las diferentes acciones: lectura de características de la baliza, localización e identificación del *iBeacon*. Para ello se utilizan diferentes campos que son explicados a continuación [21]:

- **Cabecera:** Indica que se trata de una transmisión en modo broadcast y además informa de la cantidad de datos que se transmitirán posteriormente.

- **AD *Address*:** Indica la dirección MAC del *iBeacon* y datos específicos del generador de la notificación (empresa, organismo…) [13].

- **AD *Structure*:** Está subdividido en *AD type*, que indica el tipo de datos a transmitir, AD *length*, que indica la longitud de los datos transmitidos y *AD data*, que codifica la notificación que se quiere transmitir.

- **UUID:** Se trata de un identificador estándar (*norma ISO / IEC11578* [14]) único y universal utilizado en entornos de computación distribuida (no necesita coordinación central). Aparece generalmente representado por 32 dígitos hexadecimales de la siguiente forma [8-4-4-4-12]. Por ejemplo: 2150d9400-f21b-51e4-b256-416354530000. Su función en un sistema *iBeacon* es identificar a todas las balizas pertenecientes a un mismo grupo.

- ***Major*:** Especifica un subgrupo de balizas dentro de un grupo principal.

- ***Minor*:** Identifica a una baliza específica dentro de un subgrupo.

- ***Tx Power*:** Indicador de potencia de transmisión del *iBeacon* (teórica) a un metro de distancia medida en dBm.

Por último, la trama de datos es verificada con un código de detección de errores basado en CRC (*Cyclic Redundancy Check*) de 24 bits.

4.1.2 *Posicionamiento con iBeacon. RSSI*

El posicionamiento en interiores con un sistema *iBeacon* se define como posicionamiento por proximidad. Este concepto está basado en el cálculo de la distancia a un *iBeacon* mediante medición de la potencia de señal recibida (RSS).

En el apartado 4.1.1 se introdujo el concepto de *Tx Power*: valor de potencia de señal teórico a un metro de distancia medida en dBm. La estimación de la distancia proporcionada por una baliza se basa en la relación de la potencia de la señal *iBeacon* recibida (RSS) sobre la potencia calibrada del transmisor (*Tx Power*). Esta relación queda definida por el RSSI (*Received Signal Strength Indicator)* que toma la siguiente expresión:

$$RSSI\ [dBm] = -(10 * n)\log_{10}(d) - A \qquad (1)$$

Siendo:

- 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 *d* de la expresión (1), se obtiene la distancia relativa entre emisor y receptor. Por tanto, la precisión en la posición dependerá de la correcta calibración previa del *iBeacon* para tener un ajustado valor de *Tx power*, la calidad de medición del hardware receptor (RSSI) y la correcta ejecución de los cálculos, que son llevados a cabo por aplicaciones con soporte móvil o hardware configurado como escáner de señales *iBeacon* [15] como se muestra en la Figura 4-2.

Figura 4-2. Captura de pantalla de *iOS iBeacon Locator App*.

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 a dicho valor: la fluctuación de RSSI. Esta fluctuación puede producirse por diferentes motivos [16]:

- Reflexión.
- Refracción.
- Difracción.
- *Scattering*.
- Multitrayecto.
- Orientación relativa entre emisor/receptor.

Los anteriores factores no se tienen en cuenta en los cálculos de distancia con la expresión (1), por lo que se ven directamente reflejados en el valor *d*. Por este motivo, el posicionamiento *iBeacon* por proximidad utiliza un sistema de referencia relativo y centrado en el receptor en el que la situación se establece en tres rangos (*Immediate*, *near* y *far*), que son función directa de los valores en los que oscila *d*, como se muestra en la Figura 4-3.

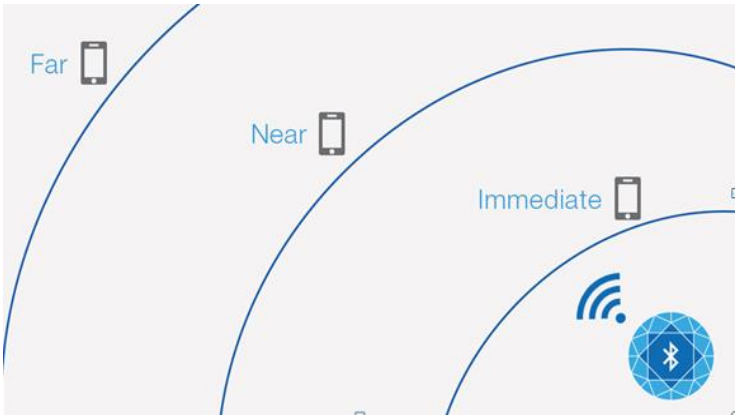


Figura 4-3. Rangos de proximidad a un *iBeacon*.

Donde:

- ***Immediate*:** Rango comprendido en el intervalo $0.1 < d < 1$ (metros).
- ***Near*:** Rango comprendido en el intervalo $1 \leq d < 10$ (metros).
- ***Far*:** $d \geq 10$ (metros).

El rango teórico máximo de distancia para un *iBeacon* utilizando BLE es de 100 metros en condiciones ideales de propagación.

A continuación, en la Figura 4-4 se muestran una serie de capturas de pantalla de la aplicación *Estimote* de Apple que muestra la posición relativa entre el receptor (soporte móvil de la aplicación) y un *iBeacon* desarrollado por Apple (punto).

Figura 4-4. Captura de pantalla de *iOS Estimote App*.

4.1.3 *Modo LEAd con iBeacon*

Otra de las funcionalidades que posee un sistema *iBeacon* es la de interactuar con el entorno en el que se encuentra. Para ello, la baliza envía un paquete de datos (Apartado 4.1.1) que contiene toda la información que se quiere compartir. Esta transmisión de información se realiza en modo *Bluetooth Low Energy Advertising* (LEAd), y consiste en la difusión del paquete de datos periódicamente.

El campo que precede al de carga útil (Dirección de Acceso) incluye los canales de comunicación y especificaciones del tipo de intercambio de datos que se establece con los dispositivos receptores en el entorno. Para el caso de *iBeacons* en modo LEAd, los datos se difunden en todas direcciones sin establecer conexión entre el sistema *iBeacon* y el receptor (*broadcast*).

La carga útil es decodificada en los receptores por aplicaciones compatibles permitiendo identificar la baliza. Tras la identificación, el dispositivo receptor analiza la información relacionada con el *iBeacon* en una base de datos (servidor, nube, etc.) y presenta dicha información de forma intuitiva al usuario.

A continuación, en la Figura 4-5 se muestra lo que sería un esquema básico de funcionamiento de *iBeacon* en modo *LEAd*.

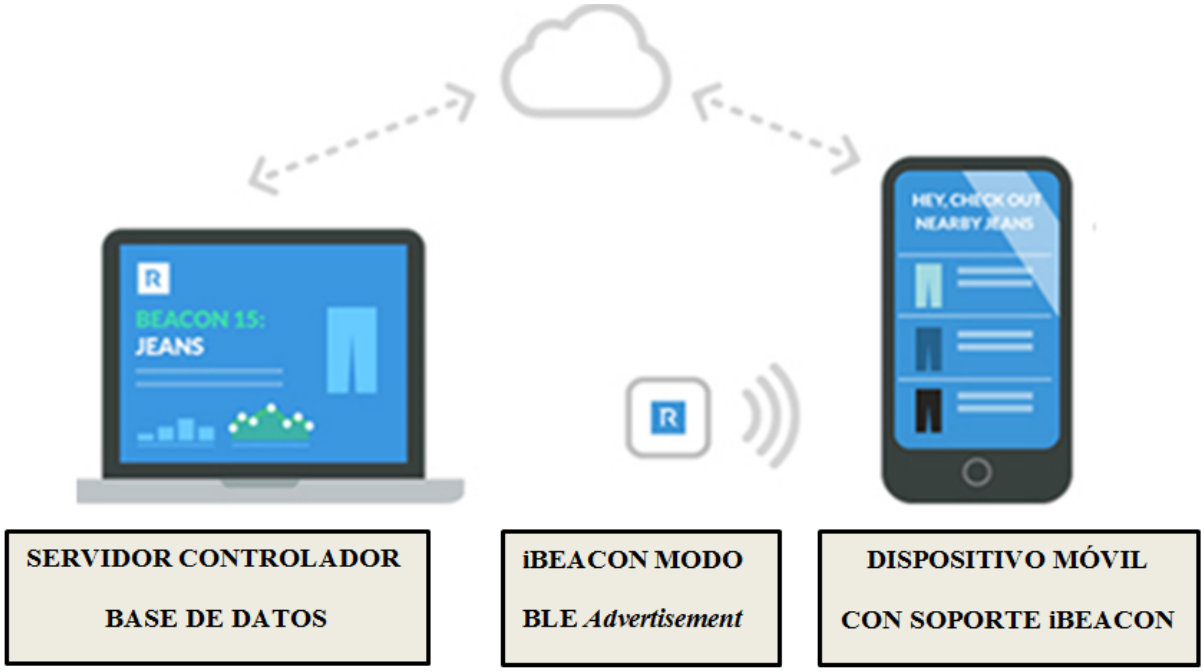


Figura 4-5. Esquema básico de funcionamiento de *iBeacon* en modo LEAd.

Una vez analizada la tecnología BLE y los principios teóricos de funcionamiento del sistema, el siguiente paso es la configuración de un sistema *iBeacon* basado en un ordenador de placa reducida Raspberry Pi versión B+ (Apartado 1.4.1) y que se realizará en dos etapas.

4.2 Configuración de RPi: Operación con BLE

En la primera etapa, el objetivo es la configuración de RPi para que pueda operar con la tecnología BLE. Todos los pasos necesarios para esta configuración se encuentran secuencialmente detallados en el Anexo I: "Manual de configuración de RPi: Operación con *Bluetooth Low Energy*".

A continuación, se describe el procedimiento de forma general y se explica el motivo de cada paso seguido:

4.2.1 *Instalación y configuración de Raspbian Debian Wheezy*

En primer lugar, se realizó la conexión de todos los periféricos necesarios para trabajar con el RPi, que incluyen: adaptador USB de BLE, adaptador USB de WiFi, teclado y pantalla con entrada HDMI.

A continuación, se instaló la distribución de Linux: Raspbian Debian Wheezy versión 3.12 como sistema operativo y se realizaron las configuraciones iniciales del RPi, quedando preparado para trabajar con él mediante la interfaz de línea de comandos.

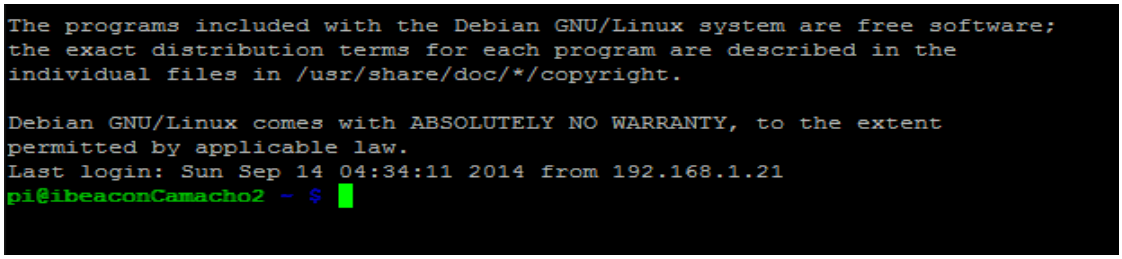


Figura 4-6. Interfaz de línea de comandos de Raspbian Debian Wheezy.

Debido a la necesidad de actualización del RPi, la instalación de librerías para el control de los adaptadores BLE vía USB y de los drivers *Bluetooth* para Raspbian, es necesario que nuestro RPi tenga conexión a Internet. Esta conexión puede realizarse de dos formas: por cable Ethernet o mediante módulo WiFi conectado a puerto USB. En el presente proyecto se llevó a cabo mediante módulo WiFi y se utilizó la conexión inalámbrica del Cuartel Marqués de la Victoria para las descargas necesarias.

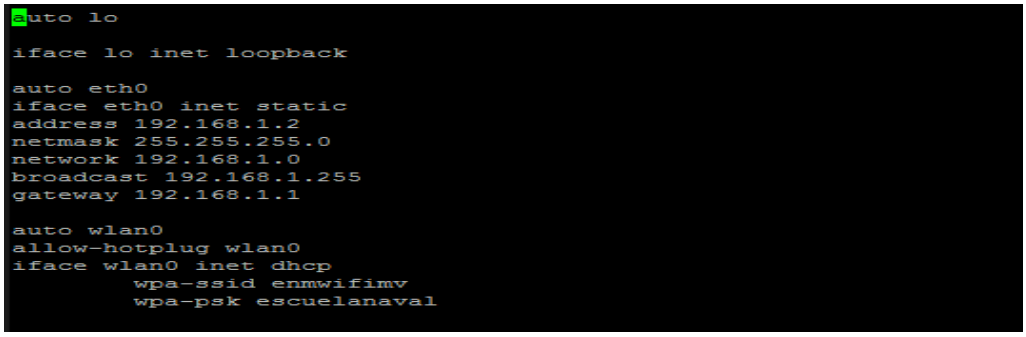


Figura 4-7. Captura de configuración de conexión inalámbrica del módulo WiFi.

4.2.2 *Instalación de controladores de dispositivos USB*

Una vez establecida la conexión a Internet y actualizado Raspbian, se realizó la descarga e instalación de las librerías [17] que permiten el control de dispositivos conectados por puerto USB:

- ***libusb-dev:*** *Permite el acceso al dispositivo conectado a puerto USB.*
- ***libdbus-1-dev****: Sistema de mensajes bus para comunicación entre aplicaciones.*
- ***libglib2.0-dev****: Librería que implementa funciones del sistema.*
- ***libudev-dev****: Funciones que permiten acceder y realizar consultas en la base de datos de los dispositivos conectados al puerto USB.*
- ***libical-dev:*** *Implementa la librería de iCalendar para el intercambio de datos de calendario.*
- ***libreadline-dev:*** *Conjunto de funciones para el uso de aplicaciones que permiten editar líneas de comandos.*

En este momento, la placa RPi se encuentra configurada para comunicarse con el módulo *Bluetooth* por puerto USB, pero aún no está configurada para comunicaciones entre dispositivos utilizando el estándar BLE. Para ello, es necesaria la instalación de una pila de protocolo *Bluetooth* para Linux.

4.2.3 *Instalación de pila Bluetooth Low Energy: BlueZ*

En la actualidad existen dos pilas relevantes dentro del mundo de Linux: BlueZ y Affix (Nokia). BlueZ es una pila de protocolos de código abierto oficial de Linux con certificación por parte del Special Interest Group (SIG) de *Bluetooth*. BlueZ permite la implementación de BLE, por lo que ha sido elegida para el desarrollo del presente proyecto.

BlueZ da soporte a perfiles de *Bluetooth* clásico y BLE, incluyendo los siguientes [19]:

- ***Human Interface Device Profile (HIDP):*** Dispositivo de interfaz humana. Da soporte a dispositivos tales como ratones, joysticks y teclados.
- ***Advanced Audio Distribution Profile (A2DP):*** Define cómo se puede propagar un flujo de audio (mono o estéreo) entre dispositivos a través de una conexión *Bluetooth*.
- ***Hardcopy Cable Replacement Profile (HCRP):*** Reemplazo de cables para la comunicación por conexión cableada entre un dispositivo y una impresora**.**
- ***Service Discovery Application Profile (SDAP):*** Permite el descubrimiento de servicios.
- ***Generic Access Profile (GAP):*** Perfil de acceso genérico y base para todos los demás perfiles.
- ***Dial-up Networking Profile (DUN):*** Conexión a red por *dial-up* o línea conmutada.
- ***Fax Profile (FAX):*** Interfaz definida entre dispositivos con software FAX.
- ***Object Push Profile (OPP):*** Perfil básico para el envío de notificaciones genéricas

(fotos, textos,…).

- ***Synchronization Profile (SYNCH):*** Sincronización entre dispositivos.
- ***File Transfer Profile (FTP):*** Transferencia de archivos con acceso remoto a los sistemas de ficheros, listados de directorios y cambios, obtención, envío y borrado de ficheros.
- ***Personal Area Networking Profile (PAN):*** Permite el uso del protocolo de encapsulación de *Bluetooth* en protocolos de nivel de red sobre un enlace *Bluetooth*.
- ***Intercom Profile (ICP):*** Un perfil de control de teléfono para el transporte de llamadas entre dos dispositivos *Bluetooth* capaces de ello.

Cabe destacar que en los RPi se ha instalado la versión BlueZ 5.18 obtenida del distribuidor *Linux Kernel Organization* [20] y que contiene los perfiles anteriormente descritos. Sin embargo, se utilizará solamente los perfiles GAP y PAN para las configuraciones BLE.

4.3 Configuración del módulo *Bluetooth*: modo de operación *iBeacon*

En la segunda etapa, se llevó a cabo la configuración del módulo *Bluetooth* para la transmisión de datos en modo *iBeacon*.

4.3.1 *Generalidades*

Para llevar a cabo esta configuración es necesaria la comunicación con la capa de enlace (LL) y la capa física (PHY) de nuestro sistema *iBeacon* (ver apartado 3.2.1). La Figura 4-8 muestra un esquema de una pila de protocolos BLE.

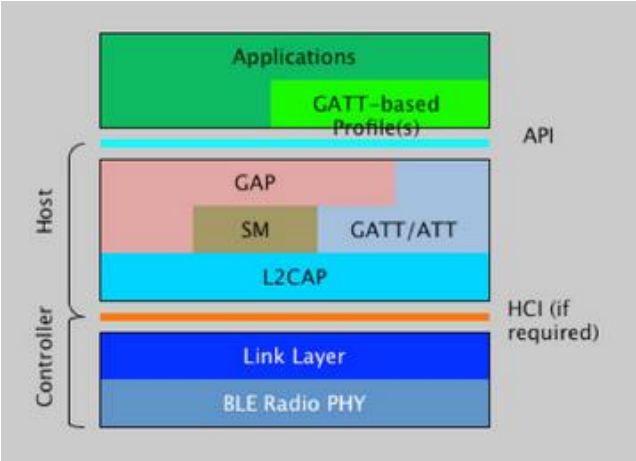


Figura 4-8. Pila de protocolos BLE.

El host, que es controlado por la capa de aplicación (interfaz con el usuario), es el encargado de establecer qué datos se envían, con qué tipo de conexión y los intervalos de emisión de estos. Esta configuración es enviada a las capas inferiores, formadas por la capa de enlace (LL) y la capa física (PHY), mediante la comunicación estandarizada con *Hardware Controler Interface* (HCI).

Aplicando lo visto anteriormente al sistema *iBeacon* del presente proyecto, el Host (RPi) se comunicará con el controlador (USB y módulo *Bluetooth*) mediante HCI (comandos de la pila BlueZ).

4.3.2 *Modo Low Energy Advertising (LEAd)*

La pila BLE permite dos tipos de comunicación principales: comunicación bidireccional o comunicación en un solo sentido. La primera proporciona conexión y transmisión de datos entre dos o más dispositivos, mientras que, en la segunda, el dispositivo *Bluetooth* lleva a cabo una transmisión periódica de datos sin establecer conexión con otros dispositivos. Este último tipo de comunicación es conocido como LEAd.

El modo LEAd es el principio de funcionamiento de los sistemas *iBeacon*, que como se explicó en el apartado 4.1, transmiten una trama de datos o *Payload Data Unit* (PDU) periódica que permite la localización e identificación de las balizas. Esta transmisión se realiza en los 3 canales de RF que BLE tiene predeterminados para *LEAd:*

- *CHANNEL 37*: 2402 MHz.
- *CHANNEL 38*: 2426 MHz.
- *CHANNEL 39*: 2480 MHz.

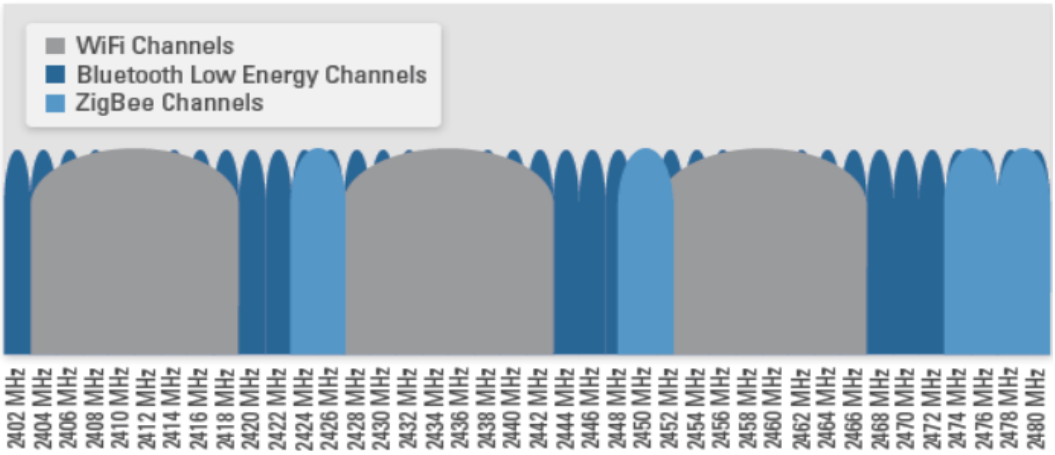


Figura 4-9. Plan de frecuencias de BLE con 37 canales de comunicación y 3 para notificaciones.

El periodo de transmisión de la PDU es otro parámetro configurable del modo LEAd, pudiendo tomar valores comprendidos entre 100 ms a 10.24 s (múltiplos de 0.625 ms), y con retardos aleatorios de 10 ms en cada pulso para evitar interferencias entre varios *iBeacon* situados en el mismo entorno.

4.3.3 *Activación del modo LEAd*

La activación del modo LEAd se lleva a cabo con la interfaz de control de hardware HCI mediante los perfiles de BlueZ. Para ello se utilizan tres comandos de HCI:

- El comando ***LE Set Advertising Data*** se utiliza para ajustar los datos de la carga útil del PDU. El tamaño máximo puede llegar hasta los 312 bits y debe seguir el formato de la especificación BLE (ver Apartado 4.1.1).

```
pi@ibeaconCamacho2 ~ $ sudo hcitool -i hci0 cmd 0x08 0x0008 1E 02 01 1A 1A FF 4C
 00 02 15 90 25 85 7E 8D 8E 4A 0F 9A 07 79 B2 22 7E 3C B9 00 01 00 02 CB 00
< HCI Command: ogf 0x08, ocf 0x0008, plen 32
  1E 02 01 1A 1A FF 4C 00 02 15 90 25 85 7E 8D 8E 4A 0F 9A 07
  79 B2 22 7E 3C B9 00 01 00 02 CB 00
> HCI Event: 0x0e plen 4
  01 08 20 00
```

Figura 4-10. Ejemplo de ejecución de comando *LE Set Advertising Data*.

En la Figura 4-10 se puede observar la configuración de un *iBeacon* del presente proyecto con su correspondiente *cabecera* <1E 02>, *AD Address y AD type* <01…15>, *UUID* <90 25…3C B9>, *Major* <00 01>, *minor* <00 02> e indicador de *Tx power* <CB> (consúltese Anexo II para más detalle).

- El comando ***LE Set Advertising Parameters*** se utiliza para configurar el intervalo de duración de transmisión del *payload* y, por tanto, el periodo con el que éste será repetido. Por defecto, el modo LEAd establece estos periodos en 1280 ms, pero pueden ser configurables como se indicó anteriormente. Además, este comando permite la selección del canal utilizado para la transmisión así como el modo de transmisión (conectable o no conectable).

```
pi@ibeaconCamacho3 ~/bluez/bluez-5.18 $ sudo hcitool -i hci0 cmd 0x08 0x0006 20
03 20 03 00 00 00 00 00 00 00 00 00 07 00
< HCI Command: ogf 0x08, ocf 0x0006, plen 15
  20 03 20 03 03 00 00 00 00 00 00 00 00 07 00
> HCI Event: 0x0e plen 4
  01 06 20 00
```

Figura 4-11. Ejemplo de ejecución de *LE Set Advertising Parameters*.

En la Figura 4-11 se ha configurado un intervalo fijo de 500 ms <20 03 20 03>, en modo conectable <00> y con el valor <07> que especifica el uso de los canales predeterminados para el modo LEAd (Ch 37, 38 y 39).

- El comando ***LE Set Advertising Enable*** es utilizado por el controlador para iniciar o interrumpir la transmisión del sistema *iBeacon* en modo LEAd.

```
pi@ibeaconCamacho3 ~/bluez/bluez-5.18 $ sudo hcitool -i hci0 cmd 0x08 0x000a 01
< HCI Command: ogf 0x08, ocf 0x000a, plen 1
  01
> HCI Event: 0x0e plen 4
  01 0A 20 00
```

Figura 4-12. Ejemplo de ejecución de *LE Set Advertising Enable*.

En la captura de la Figura 4-12 se ejecuta el comando *LE Set Advertising Enable* con valor <01>, que implica activación, mientras que <00> implicaría desactivarlo. (consúltese Anexo II para más detalle).

Tras los pasos anteriores, el sistema *iBeacon* queda configurado en modo *LE Advertising* conectable transmitiendo un *payload* con su correspondiente UUID, Major, minor y *Tx Power* que posibilita su identificación por un dispositivo receptor.

4.4 Configuración de RPi como escáner BLE

En el apartado 3.3.2 del Capítulo 3 se explicó que uno de los modos de funcionamiento de un dispositivo BLE es el modo escáner. En este modo, un receptor BLE es capaz de recibir e interpretar los paquetes de datos de otro dispositivo funcionando en modo *iBeacon*.

En el presente proyecto se han configurado tres RPi con sus correspondientes adaptadores *Bluetooth* en modo escáner asumiendo las funciones de un dispositivo *Bluetooth Smart Ready*. Para ello se ha utilizado el comando *Low Energy Scan* definido por la pila de protocolos BlueZ y que puede ser gestionado (activado o desactivado) por el comando *hci* utilizado para la comunicación entre *host* e *interfaz*. En la Figura 4-13 se muestran algunas de las acciones disponibles en los RPi tras ejecutar el comando *hcitool* sobre la pila de protocolos BlueZ.

```
      auth    Request authentication
      enc     Set connection encryption
      key     Change connection link key
      clkoff  Read clock offset
      clock   Read local or remote clock
      lescan  Start LE scan
      lewladd Add device to LE White List
      lewlrm  Remove device from LE White List
      lewlsz  Read size of LE White List
      lewlclr Clear LE White list
      lecc    Create a LE Connection
      ledc    Disconnect a LE Connection
      lecup   LE Connection Update
```

Figura 4-13. Acciones BLE disponibles en la pila de protocolos BlueZ.

Una vez se ejecuta el comando *hcitool lescan*, el RPi comienza a escanear recibiendo e interpretando los paquetes en modo LEAd que son recibidos. Cabe destacar que este modo de escaneo es activo por defecto, es decir, el escáner envía peticiones de ampliación de información al recibir paquetes LEAd. En el sistema *iBeacon* del presente proyecto, el escáner se configura en modo pasivo ya que no es necesaria más información que la contenida en una PDU del modo LEAd.

Para ver los datos contenidos en dichos paquetes es necesario ejecutar un segundo comando que se encarga de leer y presentar en pantalla los datos recibidos por el RPi configurado como escáner. El comando a ejecutar es *hcidump hci0 - -raw* y los datos presentados en pantalla tras un escaneo de paquete LEAd se muestran en la Figura 4-14.

```
pi@ibeaconCamacho2 ~ $ hcidump hci0 --raw
HCI sniffer - Bluetooth packet analyzer ver 5.18
device: hci0 snap_len: 1500 filter: 0xffffffff
> 04 0E 04 01 0B 20 0C
> 04 0E 04 01 03 0C 00
> 04 0E 0C 01 03 10 00 FF FF 8F FE DB FF 5B 87
> 04 0E 0C 01 01 10 00 06 BB 22 06 0A 00 BB 22
> 04 0E 0A 01 09 10 00 13 71 DA 7D 1A 00
> 04 0E 0B 01 05 10 00 36 01 40 0A 00 08 00
> 04 0E 07 01 23 0C 00 00 00 00
> 04 0E FC 01 14 0C 00 43 53 52 38 35 31 30 20 41 31 30 00 00
```

Figura 4-14. Captura de comando *hcidump* tras ejecutar un escaneo *Low Energy*.

Como se puede observar en la captura anterior, los datos son presentados en filas con formato hexadecimal y de forma poco intuitiva para el usuario. En dichas filas se encuentran datos de interés a la hora de escanear un *iBeacon* como pueden ser: *UUID*, *Major*, *minor* y valor de RSSI registrado por el escáner.

Con el fin de obtener una presentación intuitiva para el usuario que le permita la lectura de valores de interés directamente, se ha generado y configurado en los RPi´s un *script* ejecutable capaz de leer las líneas generadas por el comando *hcidump hci0 - -raw* y presentar en pantalla el valor de *UUID*, *Major*, *minor*, RSSI y el instante de recepción del paquete LEAd. En la Figura 4-15 se muestra una captura del terminal de comandos de RPi tras ejecutar el *script.*

```
pi@ibeaconCamacho2 ~ $ ./SCAN
UUID: 2F234454-CF6D-4A0F-ADF2-F4911BA9FFA6 MAJOR: 1 MINOR: 5 RSSI: -38
16:11:37
UUID: 2F234454-CF6D-4A0F-ADF2-F4911BA9FFA6 MAJOR: 1 MINOR: 5 RSSI: -39
16:11:39
UUID: 2F234454-CF6D-4A0F-ADF2-F4911BA9FFA6 MAJOR: 1 MINOR: 5 RSSI: -40
16:11:42
```

Figura 4-15. Captura del terminal de RPI tras ejecutar script de escaneo *Low Energy*.

A este *script* se le ha dado el nombre de *SCAN* y ha sido desarrollado utilizando como base un *script* similar creado por la empresa desarrolladora de *iBeacon Radius Networks*. Se han introducido algunas mejoras y modificaciones editando las líneas de código *bash* con el editor de texto de Debian *nano*. En el Anexo III del presente proyecto se encuentran las líneas de código del *script SCAN* así como el procedimiento necesario para configurar el RPi para su ejecución.

5 ESCENARIOS DE IMPLEMENTACIÓN DEL SISTEMA *IBEACON*

Tras el desarrollo del sistema *iBeacon* para la transmisión de información que permita su localización e identificación, así como la configuración de RPi como escáner BLE, se plantean en el presente capítulo los principales escenarios de aplicación en el control de personal a bordo.

En primer lugar, se planteará el escenario de localización de personal a bordo de buques. Tras una localización positiva, se plantea un sistema de envío de alarmas y envío de órdenes al personal embarcado. Por último, debido a la importancia de la temprana detección de la situación de hombre al agua, se plantea un sistema de transmisión de información acerca del nivel de batería del sistema *iBeacon* en tiempo real, con el fin de diferenciar la pérdida de recepción de señal *iBeacon* por caída al mar o por falta de energía en la baliza.

5.1 Escenario de localización a bordo

Como se ha explicado en el Capítulo 4, un *iBeacon* se caracteriza por la transmisión de un paquete de datos LEAd basado en la tecnología BLE. Este paquete de datos está caracterizado por un *UUID*, *Major*, *minor* y valor de *Tx Power* que hacen posible tanto la identificación del mismo, como el cálculo de proximidad a un dispositivo actuando como escáner BLE. Basado en estas dos posibilidades, se plantea un escenario de localización de personal a bordo.

En este escenario es necesaria la instalación de una infraestructura fija formada por escáneres BLE, conectados entre sí y formando una red de sensores capaz de detectar la presencia de *iBeacon* en sus correspondientes áreas de cobertura. Por otro lado, es necesario que estos escáneres registren los valores de *UUID*, *Major*, *Minor*, así como el análisis del valor de *RSSI* necesario para determinar la proximidad *iBeacon*-escáner.

El desarrollo de este sistema de localización se ha centrado en una serie de pruebas y experimentos con la finalidad de validar empíricamente su posible aplicación. Las pruebas se han llevado a cabo en tres tipos de escenarios con diferentes características físicas cada uno:

- **Hall del Cuartel Marqués de la Victoria:** Presenta un escenario libre de objetos, con una extensión de 80 m^2, recubrimiento de cemento y visión directa con los puntos de colocación de las balizas.
- **Seminarios del CUD:** Presentan un escenario compartimentado, con diferentes dimensiones en cada uno de los seminarios, recubrimiento no metálico y sin visión directa con los puntos de colocación de las balizas.

- **Patrullero "*Tabarca*":** Presenta un escenario compartimentado, con dimensiones variables y recubrimiento metálico, siendo éstas las características comunes de los buques en los que se pretende implementar el sistema *iBeacon*.

Los experimentos se han realizado en tres etapas principales y con diferentes objetivos cada una de ellas. La primera etapa se ha orientado a la configuración del sistema *iBeacon* con el fin de determinar los patrones de comportamiento estables de la señal *Bluetooth* y su relación con la distancia escáner-*iBeacon*. La segunda etapa se ha centrado en el análisis de RSSI en diferentes escenarios y situaciones con el fin de validar la posibilidad de implementación del sistema de localización basado en *iBeacon*.

Por otro lado, en la tercera etapa se ha simulado un entorno de localización en el patrullero "Tabarca" con el fin de analizar la viabilidad de implementación del sistema *iBeacon* como método de localización en interiores.

5.1.1 *Etapa 1: Análisis del comportamiento de la señal BLE*

5.1.1.1 Primer experimento

En las pruebas de esta primera etapa se ha utilizado un total de tres *iBeacon* para la emisión de las señales BLE mientras que la recepción y lectura de RSSI se ha realizado con un dispositivo iPhone 5C y la aplicación compatible con *iBeacon*: *Particle Detector*.

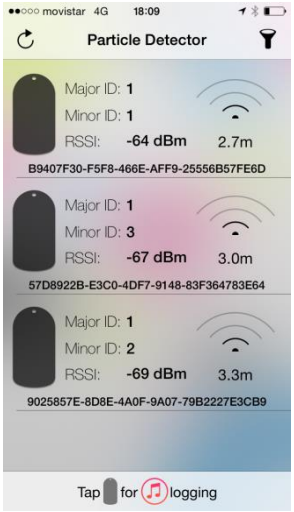


Figura 5-1. Captura de pantalla de *App Particle Detector*.

El primer experimento se llevó a cabo en los seminarios del CUD, colocando tres balizas configuradas en modo LEAd en tres seminarios diferentes, como se muestra en la Figura 5-2.

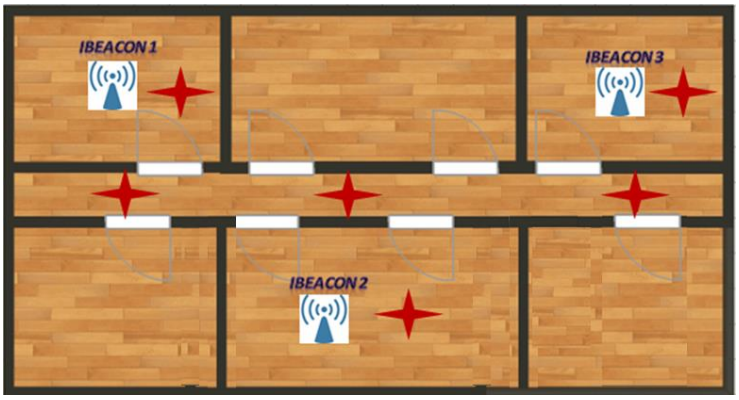


Figura 5-2. Esquema de situación de *iBeacons* y puntos de recogida de medidas (aspas).

Se realizaron medidas con el dispositivo iPhone 5C y con diferentes aplicaciones [18]: *Estimote*, *iBeacon Locator* y *Particle detector*, llegando a la conclusión de que esta última muestra los datos con mejor claridad y de forma más intuitiva. Se pudo comprobar que el valor de RSSI presentaba fluctuaciones en las tres balizas siendo complicada la determinación de la distancia. Además, se descubrió que variaciones en la posición de la antena de las balizas (Adaptador USB *Bluetooth*), afectaban al valor de RSSI.

Tras este experimento, se realizó un estudio de la directividad de los adaptadores BLE de las balizas. Para ello, se midió la RSSI a 1, 5 y 10 metros de distancia con cada baliza en diferentes posiciones relativas del receptor y la baliza. También se investigó acerca de la posibilidad de ajustar la potencia de transmisión y el intervalo de repetición de pulso del sistema *iBeacon*, con el fin de minimizar los efectos de fenómenos como: difracción, refracción y multitrayectos, causas principales de la fluctuación de la RSSI.

iBeacon en plano XY			
Distancia real	1 metro	5 metros	10 metros
Nº de medidas	30	30	30
RSSI media	55.76	63.42	67.92
Distancia medida	1.37	3.32	5.57
Dif.distancia	-0.37	1.68	4.43

iBeacon en plano -YX			
Distancia real	1 metro	5 metros	10 metros
Nº de medidas	30	30	30
RSSI media	56.41	62.89	68.45
Distancia medida	1.48	3.12	5.92
Dif.distancia	-0.48	1.88	4.08

iBeacon en plano XZ			
Distancia real	1 metro	5 metros	10 metros
Nº de medidas	30	30	30
RSSI media	54.36	65.73	69.87
Distancia medida	1.17	4.33	6.97
Dif.distancia	-0.17	0.67	3.03

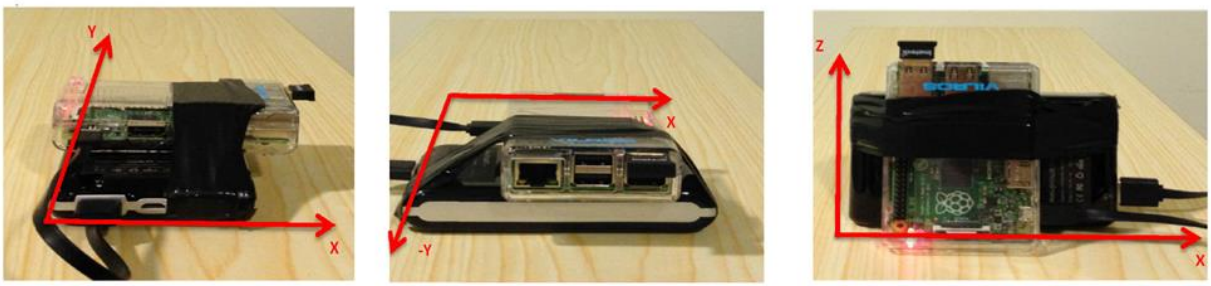


Figura 5-3. Resultados de estudio de directividad según plano de orientación.

Como se puede observar en la Figura 5-3, los resultados más exactos se obtienen orientando la baliza en el plano XZ. Por tanto, se decide orientar las tres balizas en dicho plano y continuar con los experimentos.

Tras la búsqueda de referencias e información acerca de la posibilidad de configurar la potencia de transmisión en este tipo de dispositivos, se llegó a la conclusión de que la única opción sería una modificación física del *hardware*, descartándose debido a su complejidad técnica. En lo referente a la configuración del intervalo de repetición de pulsos, la conclusión fue positiva, siendo este parámetro configurable como se explicó en el Capítulo 4.

5.1.1.2 Segundo experimento.

El segundo experimento se llevó a cabo en el hall del Cuartel Marqués de la Victoria y consistió en tres fases de mediciones. En las dos primeras se colocaron las tres *iBeacons* (en plano XZ) en un punto equidistante (primera fase a 5 metros y segunda fase a 10 metros) al punto de registro de medidas de RSSI como se muestra en la Figura 5-4.

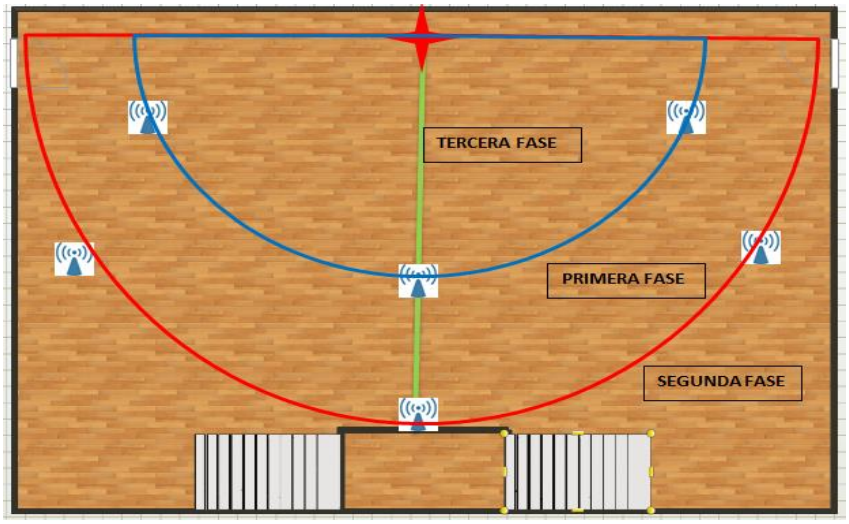


Figura 5-4. Disposición de *iBeacons*. Primera fase (azul), segunda fase (rojo) y tercera fase (verde).

En la tercera fase, se realizó el mismo experimento solamente con una de las balizas en funcionamiento, con el fin de analizar si existen o no interferencias que afecten al comportamiento del sistema.

El experimento consistió en la toma de 30 medidas conjuntas de RSSI de las tres balizas para intervalos de repetición de pulsos de 100 ms, 500 ms, 1000 ms, 1280 ms, 1500 ms y 2000 ms, llevando a cabo el análisis numérico y gráfico de los resultados con el fin de determinar la configuración para una señal estable, obteniéndose los datos de la Tabla 5-1 (consultar Anexo VI para más información).

		iBeacon1		iBeacon2		iBeacon3	
PRI	DISTANCIA REAL	5m	10m	5m	10m	5m	10m
100 ms	*RSSI Media(dBm)*	-65.93	-70.97	-66.87	-71.47	-67.37	-69
	Mediana	-65.5	-71	-67	-71	-66	-68
	Desv.Típica	3.12	2.99	2.7	3.16	3.03	2.67
	Distancia según RSSI(m)	4.43	7.91	4.94	8.38	5.23	6.31
500 ms	*RSSI Media(dBm)*	-64.27	-68.5	-64.03	-68.83	-64.13	-69.43
	Mediana	-64	-69	-64	-69	-64	-69.5
	Desv.Típica	1.06	3.04	0.87	3.8	0.88	3.49
	Distancia según RSSI(m)	3.66	5.96	3.56	6.19	3.6	6.63
1000 ms	*RSSI Media(dBm)*	-63.2	-62.23	-63.53	-64.1	-63.4	-62.3
	Mediana	-63	-63	-64	-63	-63	-61.5
	Desv.Típica	0.98	3.32	0.85	3.44	1.5	2.76
	Distancia según RSSI(m)	3.24	2.9	3.36	3.59	3.31	2.92
1280 ms	*RSSI Media(dBm)*	-64.47	-68.13	-65.33	-71.53	-62	-67
	Mediana	-64	-67	-61	-71	-61.5	-65.5
	Desv.Típica	2.11	3.71	2.66	3.51	3.29	3.29
	Distancia según RSSI(m)	3.74	5.71	4.14	8.45	2.82	5.01
1500 ms	*RSSI Media(dBm)*	-64.13	-71.23	-63.73	-70.03	-63.77	-70.17
	Mediana	-64	-70.5	-64	-70	-64	-69
	Desv.Típica	0.62	2.67	0.63	2.26	0.5	2.72
	Distancia según RSSI(m)	3.6	8.16	3.44	7.11	3.45	7.22
2000 ms	*RSSI Media(dBm)*	-68.83	-65.17	-68.5	-67.13	-68.63	-67.6
	Mediana	-69	-64.5	-68	-67	-68	-66
	Desv.Típica	1.86	2.77	1.48	3.68	1.83	4.2
	Distancia según RSSI(m)	6.19	4.06	5.96	5.09	6.05	5.37

Tabla 5-1. Mediciones de fluctuación de RSSI.

Las casillas marcadas en verde indican los valores de desviación típica sobre un valor central de RSSI así como las estimaciones de distancia que se adaptan con mejor exactitud a la distancia real de medida. Cabe destacar que el cálculo de dicha distancia se ha llevado a cabo con la fórmula (1) expuesta en el Capítulo 4, con un valor *Tx Power* configurado en las balizas de -53 dBm.

Como se puede observar en los resultados, para un intervalo de repetición de pulsos de 1500 ms, la señal *iBeacon* presenta una mayor estabilidad en la propagación, con unas estimaciones de distancias aceptables.

Con el fin de confirmar la viabilidad de esta configuración, se realizó la tercera fase del experimento con una única baliza, obteniéndose los siguientes resultados:

		iBeacon1	
PRI	DISTANCIA REAL	5m	10m
100 ms	*RSSI Media(dBm)*	-64.6	-69
	Mediana	-64	-69
	Desv.Típica	2.64	3.15
	Distancia según RSSI(m)	3.8	6.31
500 ms	*RSSI Media(dBm)*	-66.67	-68.6
	Mediana	-66	-68.5
	Desv.Típica	2.81	3.27
	Distancia según RSSI(m)	4.82	6.03
1000 ms	*RSSI Media(dBm)*	-64.3	-64.4
	Mediana	-64	-63
	Desv.Típica	2.53	4.43
	Distancia según RSSI(m)	3.67	3.72
1280 ms	*RSSI Media(dBm)*	-66.83	-68.6
	Mediana	66	-68.5
	Desv.Típica	2.81	3.83
	Distancia según RSSI(m)	4.92	6.03
1500 ms	*RSSI Media(dBm)*	-65.37	-69.77
	Mediana	-66	-70
	Desv.Típica	1.31	2.93
	Distancia según RSSI(m)	4.33	6.89
2000 ms	*RSSI Media(dBm)*	-69.23	-65.9
	Mediana	-68	-66
	Desv.Típica	5.06	3.05
	Distancia según RSSI(m)	6.48	4.42

Tabla 5-2. Mediciones de fluctuación de RSSI con una sola baliza.

Una vez más se cumple que para un intervalo de repetición de pulsos de 1500 ms se obtiene una mayor estabilidad de la señal *iBeacon* así como una estimación de distancia con exactitud aceptable. En la Figura 5-5 se muestran las fluctuaciones de RSSI obtenidas en el experimento para esta configuración (Valores de RSSI expresados en valor absoluto).

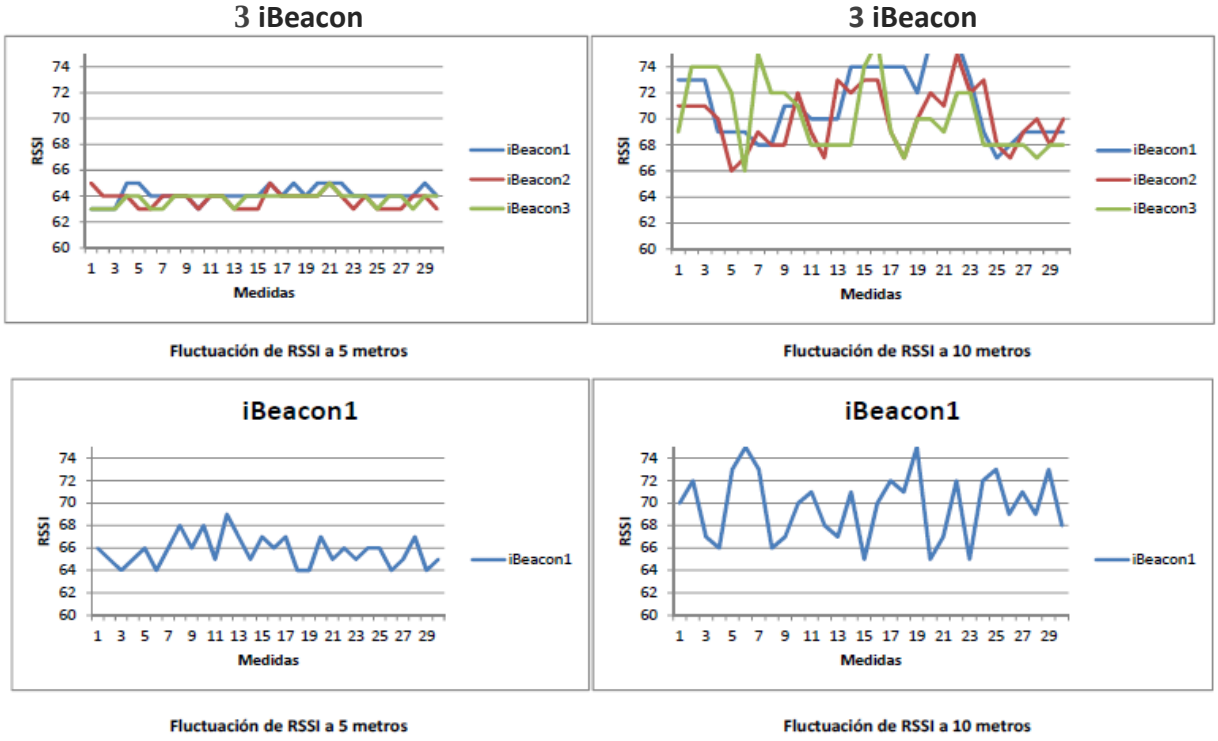


Figura 5-5. Representación gráfica de fluctuación de RSSI para un PRI de 1500 ms.

Tras finalizar esta etapa de experimentación, se llegó a la conclusión de que el sistema *iBeacon* es robusto frente a posibles interferencias de dispositivos *Bluetooth* con características similares. Se estableció el intervalo de repetición de pulsos (PRI) en 1500 ms y la posición de la baliza para obtener la configuración estable del sistema. Por último, se planteó el objetivo de la segunda etapa de experimentación: analizar el comportamiento de la RSSI con dicha configuración en el patrullero "Tabarca".

5.1.2 *Etapa 2: Análisis de RSSI en el patrullero "Tabarca"*

La segunda etapa de experimentación tuvo como principal objetivo definir el comportamiento del sistema *iBeacon* en un entorno compartimentado, con recubrimiento metálico y con todas las características en las que se pretende implementar el sistema de localización e intercambio de información.

El escenario de medida se centró en la cubierta baja donde se encuentra el sollado de dotación y la cámara de máquinas. En la Figura 5-6 se muestra con detalle la situación del experimento.

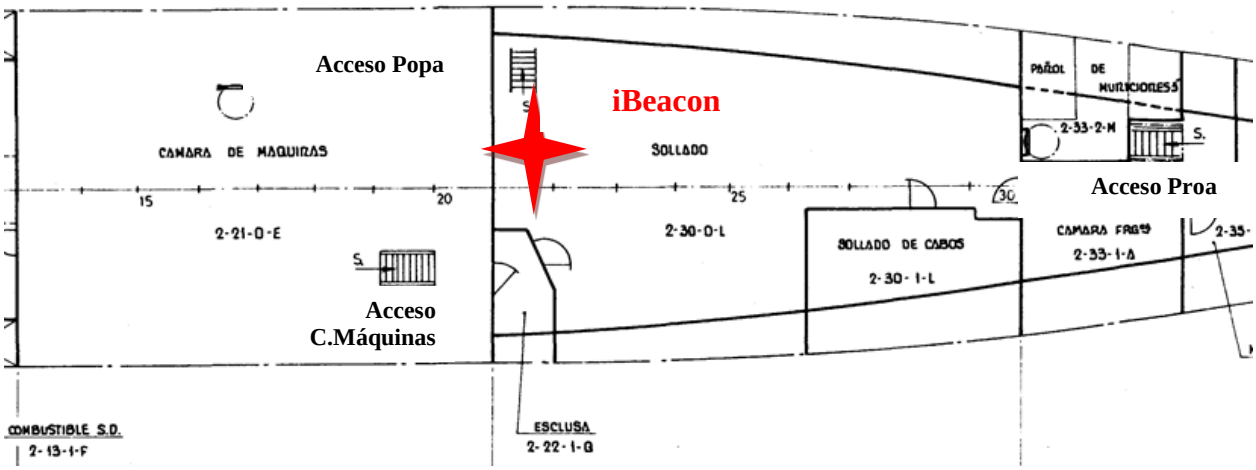


Figura 5-6. Cubierta baja del patrullero "Tabarca".

Para la realización de las pruebas se colocó la baliza en el falso techo del sollado de dotación, junto a la escala de acceso de popa y más cercana a la cámara de máquinas, como se puede ver en la Figura 5-7.

Figura 5-7. Situación de *iBeacon* en el sollado de dotación del patrullero "Tabarca".

La primera prueba realizada fue la medida de RSSI dentro del sollado, en un punto de medida situado a 5 metros de la baliza y en línea de visión directa. Se realizaron 30 medidas con cada configuración de intervalo de repetición de pulsos, obteniéndose los resultados de la Tabla 5-3.

PRI	DISTANCIA REAL = 5 m	iBeacon1
100 ms	RSSI Media(dBm)	-55.5
	Mediana	-54.5
	Desv.Típica	3.68
	Distancia según RSSI(m)	1.33
500 ms	RSSI Media(dBm)	-55.33
	Mediana	-56
	Desv.Típica	3.31
	Distancia según RSSI(m)	1.31
1000 ms	RSSI Media(dBm)	-50.97
	Mediana	-52
	Desv.Típica	4.52
	Distancia según RSSI(m)	0.79
1280 ms	RSSI Media(dBm)	-59.07
	Mediana	-59
	Desv.Típica	5.6
	Distancia según RSSI(m)	2.01
1500 ms	RSSI Media(dBm)	-58.77
	Mediana	-58
	Desv.Típica	2.94
	Distancia según RSSI(m)	1.94
2000 ms	RSSI Media(dBm)	-50.87
	Mediana	-51.5
	Desv.Típica	3.56
	Distancia según RSSI(m)	0.78

Tabla 5-3. Medidas de RSSI en sollado de dotación del patrullero "Tabarca".

Se puede observar en la anterior tabla que para un PRI de 1500 ms se obtiene el menor valor de desviación típica. Se aprecia que, en general, la estimación de distancia toma un valor inferior al registrado en experimentos anteriores, lo que se traduce como una recepción de potencia relativamente mayor en este tipo de escenarios metálicos.

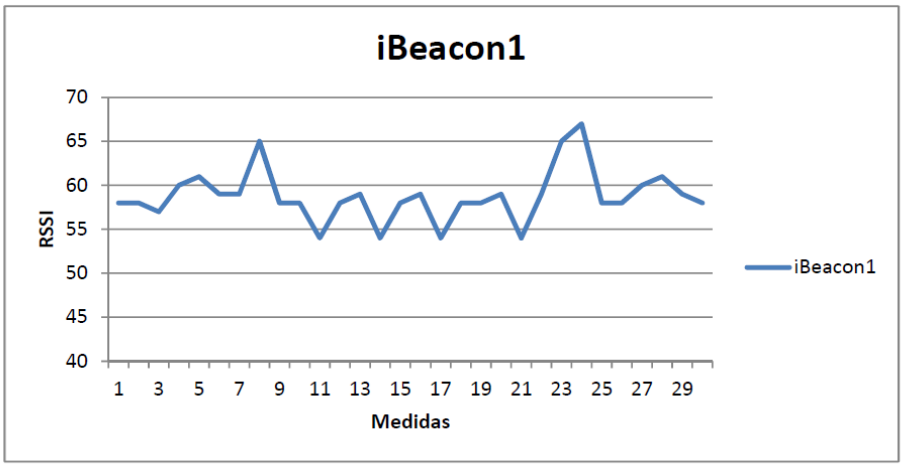


Figura 5-8. Fluctuación de RSSI para un PRI de 1500 ms en sollado del patrullero "Tabarca".

En la Figura 5-8 se puede observar la fluctuación de RSSI registrada en el sollado del patrullero para un PRI de 1500 ms. Esta fluctuación se encuentra comprendida entre valores de -55 y -65 dBm, que se traduce en un error de medida de ±1.4m. Con estos resultados, se llega a la conclusión de que el posicionamiento en interiores de buques no estará orientado a la exactitud en la posición concreta de un individuo dentro del compartimento, sino que se enfocará a determinar en qué compartimento o en qué área determinada se encuentra el mismo.

Tras esta conclusión, se llevó a cabo una segunda prueba a bordo del patrullero. Con la baliza colocada en la misma posición anteriormente citada, se recorrieron todos los compartimentos adyacentes al sollado de dotación con el fin de determinar el patrón de propagación de la señal en un entorno metálico y compartimentado.

A lo largo de toda la eslora y manga del sollado, la señal BLE fue recibida para los diferentes valores de PRI. En la cubierta superior se dieron dos casos claramente diferenciados:

1. En el primer caso, tras acceder a la cubierta principal por la escala de popa (más cercana al *iBeacon*), la señal fue recibida sin ningún tipo de interferencia y con un comportamiento similar al registrado en el sollado.

2. En el segundo caso, se accedió a la cubierta superior por la escala de proa (más lejana al *iBeacon*) donde la señal *Bluetooth* ya no fue recibida excepto para un PRI de 100 ms, donde en el transcurso de dos minutos se registraron solamente tres recepciones de -74, -69 y -78 dBm respectivamente. Recorriendo la cubierta superior de proa a popa se pudo observar que existía la recepción de señal sin un comportamiento o patrón estable (excesiva fluctuación y nulos).

A continuación, se accedió a la cámara de máquinas por el acceso de estribor (consultar Figura 5-6) y se comprobó el número de nulos de señal (intensidad indetectable en recepción) que se producían en 30 medidas para los diferentes valores PRI obteniéndose lo siguiente:

- **100 ms:** 19 nulos.
- **500 ms:** 11 nulos.
- **1000 ms:** 19 nulos.
- **1280 ms:** 19 nulos.
- **1500 ms:** 27 nulos.
- **2000 ms:** 30 nulos.

De los datos anteriores se llega a la conclusión de que es posible determinar un patrón de comportamiento de la señal *Bluetooth* en interiores, que queda resumido en lo siguiente:

1. PRI más adecuado de los evaluados debido a la estabilidad de sus resultados: 1500 ms.
2. En entornos compartimentados y relativamente estancos, la señal *Bluetooth* presenta límites de propagación que determinan zonas de cobertura de señal.
3. Existen zonas de incertidumbre en las que es difícil prever un patrón de comportamiento estable de la señal.

Con lo anterior se dio por concluida la segunda etapa de pruebas.

5.1.3 *Etapa 3: Prueba del sistema iBeacon de localización a bordo*

Tras el análisis del comportamiento de la señal BLE, se simuló un entorno de localización a bordo del patrullero "Tabarca" con el fin de analizar la viabilidad del sistema *iBeacon* para el posicionamiento en interiores. Para la ejecución de las pruebas se llevaron a cabo los siguientes pasos y configuraciones previas:

1. Configuración de 3 RPi en modo escáner: En este paso, se llevaron a cabo los procedimientos explicados en el capítulo 4 para la configuración de los RPi con el rol de escáner y poder registrar los datos de *Major*, *minor* y valor de RSSI.

2. Posicionamiento de los escáneres en el buque: En este paso, se colocaron los escáneres en tres puntos diferentes. El escáner *ES* se colocó en el sollado de dotación en la cubierta baja, el escáner *EC* se colocó en la cámara de máquinas en la cubierta baja y, por último, el escáner *ET* se colocó en la toldilla del patrullero (cubierta principal). En la Figura 5-9 se muestra la disposición de los escáneres.

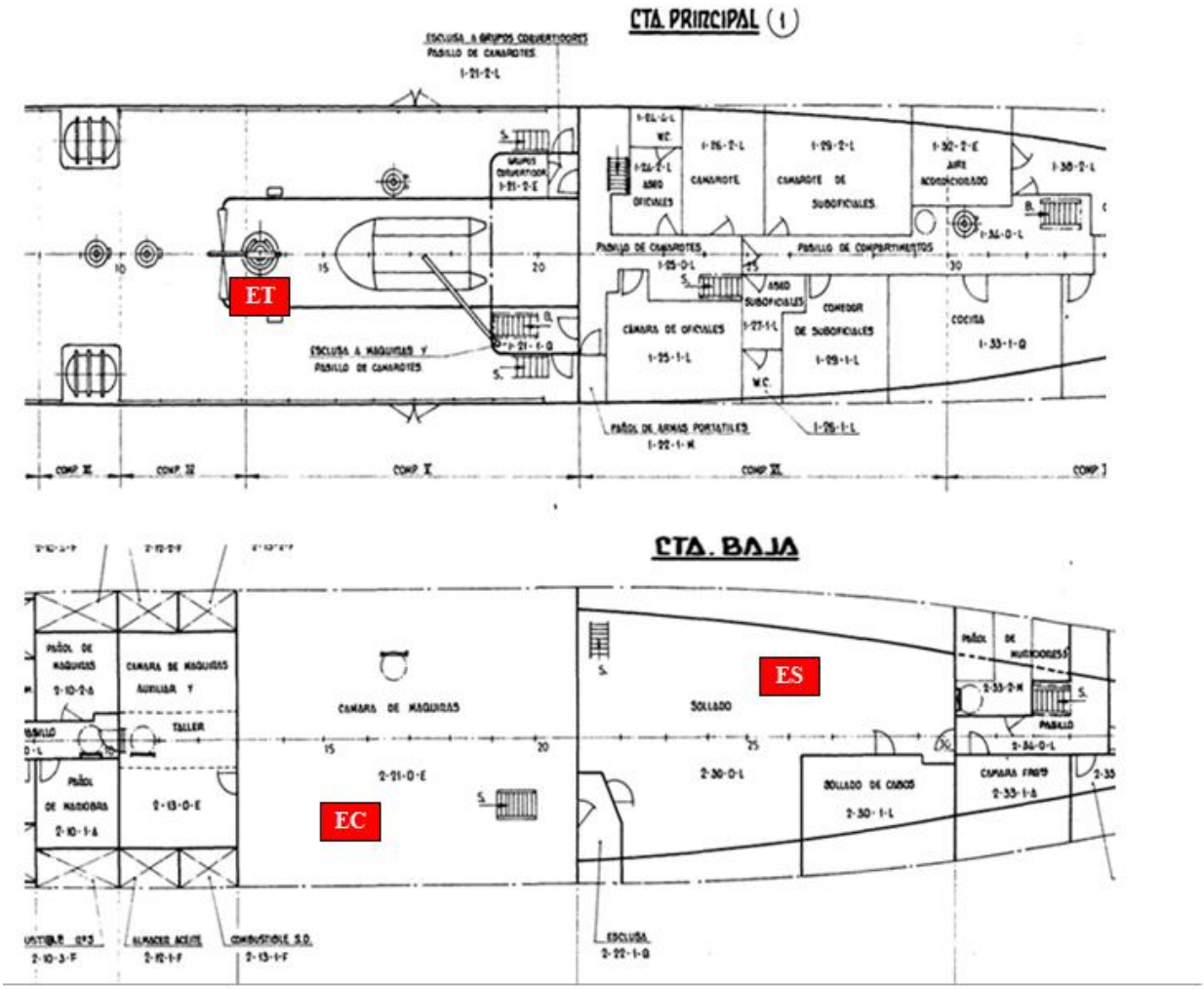


Figura 5-9. Esquema de situación de escáneres.

3. Creación de WLAN: En este paso, se realizó la configuración de una red WLAN con un router *Livebox* actuando como punto de acceso común y permitiendo la comunicación entre los RPi y el ordenador portátil utilizado para monitorizar los resultados.

4. Monitorización de resultados de escáner: En este paso, se ejecutó el *software putty* en el ordenador portátil, permitiendo la conexión remota *SSH* vía punto de acceso con los tres escáneres simultáneamente para monitorizar los resultados del escaneo.

5. Configuración de iPhone 5C en modo LEAd: Se utilizó la aplicación *iBeacon Locator* [18] de iOS sobre un iPhone 5C para simular un *iBeacon* operando en modo LEAd.

6. Sincronización de relojes: En este paso, se realizó la sincronización de relojes de los tres escáneres, del ordenador y del reloj de pulsera del portador del *iBeacon* con el fin de establecer una referencia común para el registro de medidas.

5.1.3.1 Desarrollo del experimento

Este experimento consistió en la monitorización de los datos escaneados por los tres RPi en modo escáner a medida que un individuo se desplazaba por el buque con el *iBeacon* situado en el bolsillo de su pantalón. Se siguió un itinerario registrando los tiempos en los que el *iBeacon* accedía a los compartimentos de interés y registrando el momento en el que se accedió a exteriores (toldilla). En la Figura 5-10 se muestra el itinerario real seguido.

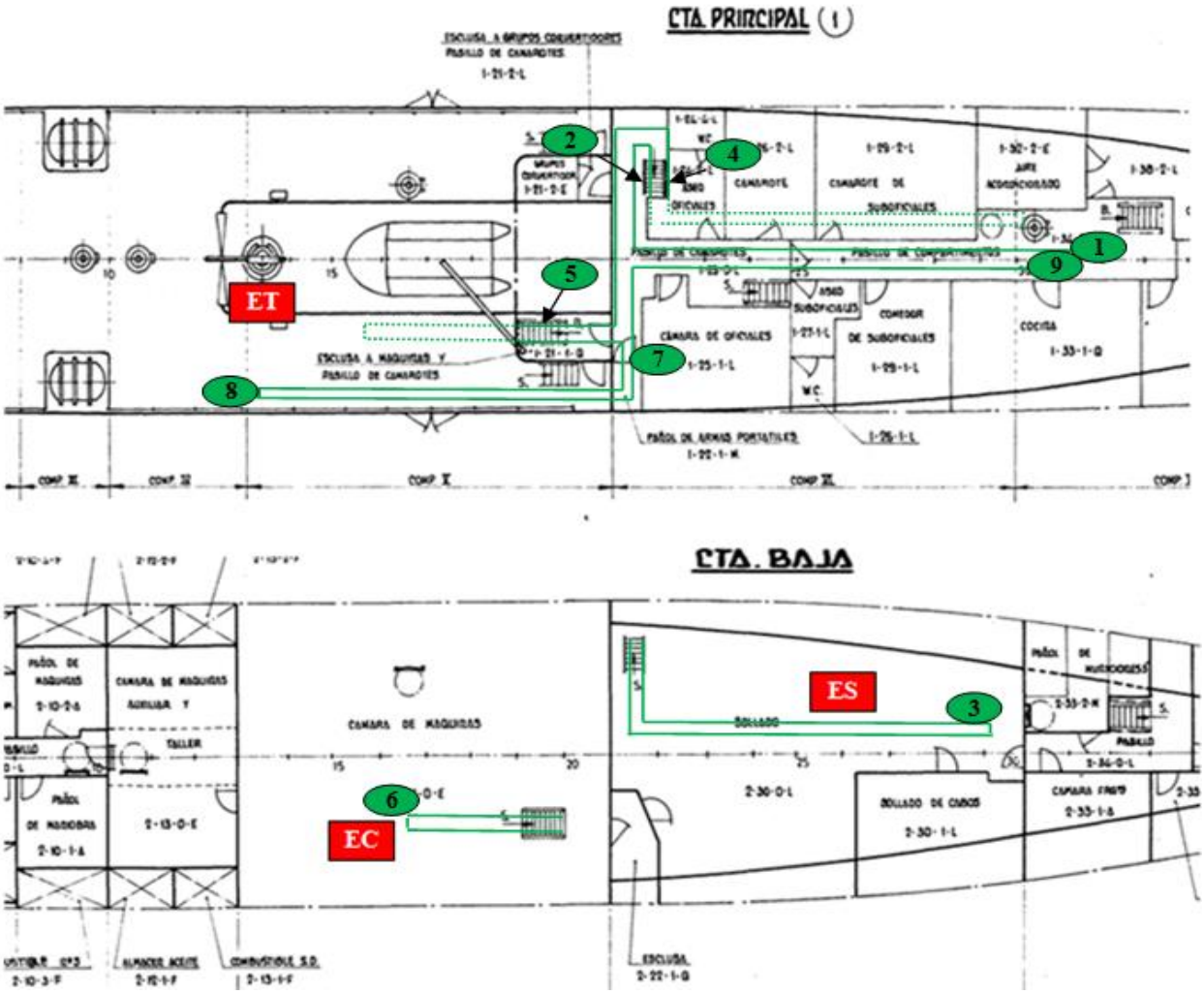


Figura 5-10. Itinerario de pruebas.

El itinerario comienza en el punto 1 en la cubierta principal en dirección al sollado de dotación (cubierta baja). Se accede al sollado en el punto 2, se recorre el compartimento pasando por el punto 3 y se accede de nuevo a la cubierta principal por el punto 4. El punto 5 indica el momento de acceso a la cámara de máquinas (cubierta baja). En su interior se realizó un registro de medidas en el punto 6 y se volvió a acceder a la cubierta principal. El punto 7 indica el momento en el que se accede a la toldilla (exteriores). Se avanza hasta el punto 8 donde se realiza un registro de medidas y, por último, se vuelve al punto de inicio de las pruebas (punto 9). Cabe destacar que el itinerario se realizó a ritmo constante, anotando 20 valores de RSSI en cada uno de los puntos de registro de medidas.

El objetivo del experimento es reproducir dicho itinerario analizando los valores de RSSI de cada uno de los escáneres, valores que permiten definir la proximidad del *iBeacon* a un escáner u otro. En la Tabla 5-4 se muestran los valores de RSSI y distancia calculada a partir de dicho valor en los instantes de tiempo de interés.

PUNTO DE MEDIDA	ES	EC	ET
1	83 dBm 22 m	NULO	NULO
2	66 dBm 3 m	84 dBm 25 m	NULO
3	50 dBm 0.5 m	86 dBm 31.5 m	NULO
4	72 dBm 6 m	83 dBm 22.4 m	NULO
5	85 dBm 28 m	65 dBm 2.8 m	83 dBm 22 m
6	87 dBm 35 m	53 dBm 0.7 m	NULO
7	83 dBm 22 m	70 dBm 5 m	NULO
8	NULO	89 dBm 44 m	52 dBm 0.6 m
9	88 dBm 39 m	NULO	NULO

Tabla 5-4. Valores de RSSI y distancia teórica en los puntos de medida.

La distancia se ha calculado con la fórmula: $RSSI\ [dBm] = -(10 * n)log_{10}(d) - A$, explicada en el capítulo 4. Esta expresión establece una relación logarítmica entre la distancia y el valor de RSSI, por lo que variaciones de bajo orden en el valor de RSSI implican variaciones importantes en el valor de la distancia. En la Tabla 5-4 se puede ver como, para un valor de RSSI de -83 dBm, la distancia teórica se corresponde con 22 metros, mientras que, para un valor de -72 dBm, la distancia toma un valor de 6 metros. Este hecho permite establecer rangos en los valores de RSSI que determinen de forma clara la proximidad a un determinado escáner.

En el Capítulo 4 se introdujo el concepto de proximidad y se dividió éste en tres rangos dependiendo de la distancia entre *iBeacon* y receptor LEAd, siendo estos rangos:

- **Immediate:** Rango comprendido en el intervalo $0.1 < d < 1$ (metros).
- **Near:** Rango comprendido en el intervalo $1 \leq d < 10$ (metros).
- **Far:** $d \geq 10$ (metros).

Se puede afirmar que, en el entorno interior definido por un buque, como es el escenario de pruebas a bordo del "Tabarca", las dimensiones de los compartimentos permiten establecer la zona de cobertura de los escáneres a un radio no superior a los 10 metros. Por tanto, el rango de proximidad se centrará entre *Immediate* y *Near*. Para situar un *iBeacon* dentro de este entorno interior, bastará con analizar la distancia de éste a cada uno de los escáneres situados en puntos de referencia conocidos identificando en qué rango de proximidad se encuentra. En la Tabla 5-5 se muestra una comparativa entre la proximidad real del *iBeacon* de pruebas y la proximidad obtenida con la medida de RSSI.

PUNTO DE MEDIDA	PROXIMIDAD ES		PROXIMIDAD ET		PROXIMIDAD EC	
	REAL	MEDIDA	REAL	MEDIDA	REAL	MEDIDA
1	*Far*	*Far*	*Far*	*NULO*	*Far*	*NULO*
2	*Near*	*Near*	*Far*	*NULO*	*Far*	*Far*
3	*Near*	*Immediate*	*Far*	*NULO*	*Far*	*Far*
4	*Near*	*Near*	*Far*	*NULO*	*Far*	*Far*
5	*Far*	*Far*	*Far*	*Far*	*Near*	*Near*
6	*Far*	*Far*	*Far*	*NULO*	*Near*	*Immediate*
7	*Far*	*Far*	*Far*	*NULO*	*Near*	*Near*
8	*Far*	*NULO*	*Near*	*Immediate*	*Far*	*Far*
9	*Far*	*Far*	*Far*	*NULO*	*Far*	*NULO*

Tabla 5-5. Comparativa de proximidad real-medida.

Con los datos de la tabla anterior puede llevarse a cabo una reproducción del itinerario seguido:

- **Punto 1:** Se puede afirmar que el *iBeacon* se encuentra fuera de la zona de cobertura del escáner de toldilla y del escáner de la cámara de máquinas. Sin embargo, se encuentra en la zona de cobertura del escáner del sollado, por lo que la situación del *iBeacon* se limita a las proximidades del mismo. Se puede afirmar que el *iBeacon* se encuentra en la cubierta superior al sollado o a proa del compartimento.

- **Punto 2:** Se puede afirmar que el *iBeacon* se encuentra en las proximidades del sollado de dotación y a popa del mismo, ya que se encuentra próximo al escáner del sollado y dentro de la cobertura del escáner de la cámara de máquinas.

- **Punto 3:** Se puede afirmar que el *iBeacon* se encuentra muy próximo al escáner del sollado, por lo que se sitúa a éste dentro del compartimento.

- **Punto 4:** Se puede afirmar que el *iBeacon* se encuentra próximo al sollado y a popa de éste, ya que está dentro de la cobertura del escáner de la cámara de máquinas.

- **Punto 5:** Se puede afirmar que el *iBeacon* se encuentra próximo a la cámara de máquinas y dentro de la cobertura del resto de escáneres. La única zona que cumple esto es el pasillo de acceso a la cámara de máquinas.

- **Punto 6:** Se puede afirmar que el *iBeacon* se encuentra muy próximo al escáner de la cámara de máquinas, por lo que se sitúa al mismo en dicho compartimento.

- **Punto 7:** Se puede afirmar que el *iBeacon* se encuentra próximo a la cámara de máquinas y dentro de la cobertura del escáner del sollado de dotación. Por tanto, se puede situar al mismo a proa de la cámara de máquinas (cubierta baja) o en el pasillo central que comunica el sollado y la cámara de máquinas (cubierta principal).

- **Punto 8:** Se puede afirmar que el *iBeacon* se encuentra muy próximo al escáner de toldilla, por lo que se sitúa al mismo a popa en exteriores (toldilla).

- **Punto 9:** Se repite el patrón del punto 1.

Con lo anterior puede reproducirse un itinerario aproximado al realmente realizado donde la localización del *iBeacon* se realiza por compartimentos o por áreas dentro del buque. Existen algunas zonas de incertidumbre (punto 1 y 9) donde el posicionamiento no está claramente definido. Bastaría con la colocación de otro escáner a proa del sollado en una zona de cobertura adyacente que permita definir la proximidad al sollado o a este nuevo escáner.

En el caso del punto 7, el posicionamiento es incorrecto ya que, en el itinerario real, este punto corresponde con el acceso a exteriores hacia la toldilla. El escáner situado en toldilla registra nulos en dicho punto, donde teóricamente se deberían registrar valores de RSSI que permitan establecer un rango de proximidad *Far* o incluso *Near*. La propagación de la señal en exteriores (sin rebotes ni medio metálico) y el perfil geométrico presentado por el acceso a exteriores pueden ser causantes de una reducción de la cobertura del escáner de toldilla o de una mayor cantidad de pérdidas en la señal emitida por el *iBeacon*. Para solucionar este error de posicionamiento bastaría con resituar el escáner en un punto que le permita cubrir el punto 7 de acceso a exteriores.

Con los resultados obtenidos en la fase de experimentación se puede concluir que el dato principal utilizado para determinar la proximidad de un *iBeacon* a un escáner es el rango de distancia existente entre los mismos, siendo las proximidades *Immediate* y *Near* las que aseguran una localización aceptable. Como contrapartida, una proximidad *Far* introduce un grado de ambigüedad que no permite una correcta localización. Por tanto, uno de los factores importantes a tener en cuenta para la implementación del sistema *iBeacon* de localización es la correcta colocación de los escáneres BLE dentro del escenario, permitiendo establecer en éste zonas de cobertura comprendidas solamente entre rangos *Immediate* y *Near*.

5.2 Escenario de envío de alarmas y órdenes

Una vez que es posible localizar un *iBeacon* dentro de un buque, se pretende implementar una mejora al escenario de localización que permita el envío de alarmas y órdenes al personal portador de un sistema *iBeacon* en las diferentes situaciones a bordo de un buque.

El sistema de alarma y órdenes debe estar integrado en las funcionalidades del sistema *iBeacon* permitiendo la transmisión de una PDU con el correspondiente *UUID*, *Major*, *minor* y *Tx Power* característicos de una baliza BLE actuando como esclavo. Además, debe ser capaz de realizar un escaneo continuo y ejecutar la señal de alarma u orden al recibir el comando proveniente de un nodo actuando como maestro. En la Figura 5-11 se muestra un esquema básico de funcionamiento.

.

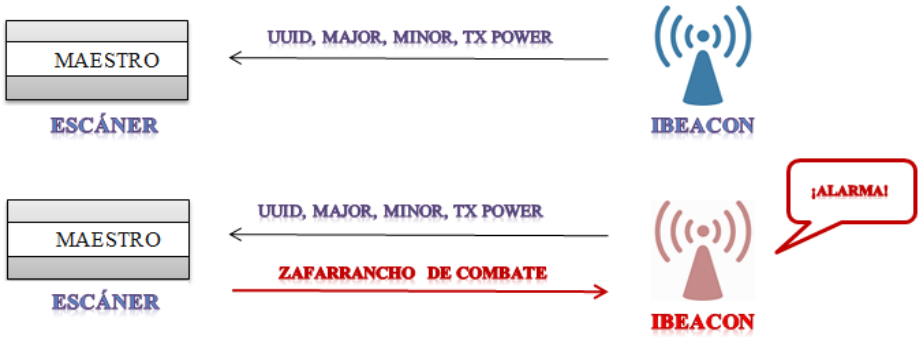


Figura 5-11. Esquema de sistema de alarma *iBeacon*.

En primer lugar, el escáner actuando como maestro, recibe el paquete LEAd de un *iBeacon* que se encuentre en su zona de cobertura. El *iBeacon* se encuentra transmitiendo en modo LEAd y, a la

vez, realiza un escaneo *Low Energy* pasivo que le permite identificar paquetes LEAd en caso de que el maestro los envíe.

Esta funcionalidad se implementa de dos formas diferentes: por un lado, el sistema de envío de alarmas está pensado para alertas generales que deben ser transmitidas a todo el personal del buque en una situación determinada (zafarrancho de combate, abandono de buque, etc.); por otro lado, el sistema de envío de órdenes está pensado para ser ejecutado entre un maestro (escáner) y un esclavo (*iBeacon*) en concreto dentro de su zona de cobertura, creando una línea de mando interna.

5.2.1 *Sistema de envío de alarmas*

En el momento que sea necesario activar el sistema de envío de alarmas, el maestro envía un paquete LEAd al *iBeacon* caracterizado por un valor de *UUID, Major y minor*. El valor de *UUID* enviado por el escáner coincidirá con el *UUID* configurado en el sistema *iBeacon* para la activación del sistema de alarmas. El valor de *Major* identifica el grupo de alarma al que pertenece la alerta que se quiere enviar. El valor de *minor* identifica la alerta en concreto que debe ser enviada.

Por ejemplo, si se desea enviar la alarma *Zafarrancho de combate* a todos los portadores de *iBeacon* a bordo de un buque, todos los escáneres instalados en el mismo deben llevar a cabo las siguientes acciones.

1. Transmitir el *UUID* pre-configurado en el sistema *iBeacon* para la activación del sistema de alarmas.
2. Transmitir un *Major* correspondiente al grupo de alertas que se quiere enviar.
3. Transmitir un *minor* que permita al *iBeacon* receptor identificar la alerta recibida.

Por parte del *iBeacon* receptor, se deben llevar a cabo las siguientes acciones:

1. Tras la recepción del *UUID* pre-configurado en el sistema como iniciador del sistema de alarmas, continuar el escaneo e identificación del valor de *Major.*
2. Tras la identificación del grupo de alarmas (*Major)* al que corresponde el paquete LEAd recibido, identificar el valor *minor*.
3. Tras la identificación de la alerta por su correspondiente valor de *minor,* emitirla en el dispositivo de forma gráfica y auditiva.

Para llevar a cabo esta funcionalidad es necesaria una pre-configuración y la creación de una base de datos en los *iBeacons* receptores. Esta base de datos debe contener el valor de *UUID* genérico utilizado por los escáneres para iniciar el sistema de alarmas. Por otro lado, un registro de grupos de alertas según el valor de *Major* (e.g: *Major*=1 Alarmas NBQ; *Major*=2 Alarmas de Seguridad Interior; *Major*=10 Alarmas generales). Por último, un registro de los valores de *minor* según la alerta concreta a emitir (e.g: *Major=10 minor=1* Zafarrancho de Combate dentro de alarmas generales).

En el presente proyecto se han configurado dos tipos de alertas agrupadas en el grupo de "alarmas generales" correspondiente a los *Major* de valor 10: Alerta de *Zafarrancho de Combate* (*minor=1)* y alerta de *Abandono de Buque* (*minor=2).*

En la Figura 5-12 se muestra un ejemplo de presentación en pantalla de la alarma de *Zafarrancho de Combate*.

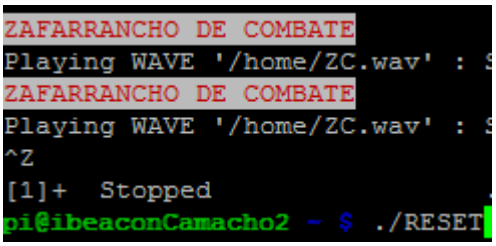


Figura 5-12. Captura de alerta de Zafarrancho de Combate en terminal Debian.

5.2.2 *Sistema de envío de órdenes*

En el momento que sea necesario activar el sistema de órdenes, el maestro envía un paquete LEAd al *iBeacon* caracterizado por un valor de *UUID, Major* y un valor *minor*. El valor de *UUID* enviado por el escáner coincidirá con el *UUID* del *iBeacon* al que se quiere enviar la orden. El valor de *Major* identificará el grupo al que pertenece esta orden enviada por el escáner. El valor de *minor* la identifica dentro de un grupo en concreto.

Por ejemplo, si se desea enviar la orden *Acuda a puente* a un *iBeacon* con *UUID*: *57d8922b-e3c0-4df7-9148-83f364783e64*, el escáner en la cobertura de dicho *iBeacon* debe realizar las siguientes acciones:

1. Transmitir un *UUID = 57d8922b-e3c0-4df7-9148-83f364783e64.*
2. Transmitir un *Major* correspondiente al grupo de órdenes que se desea enviar.
3. Transmitir un *minor* que permita al *iBeacon* receptor identificar la orden recibida.

Por parte del *iBeacon* receptor, se deben llevar a cabo las siguientes acciones:

1. Tras la recepción del *UUID* y reconocer que coincide con su propio *UUID*, activar el sistema de órdenes y escanear el valor de *Major.*
2. Tras la identificación del valor de *Major* y, por tanto, el grupo al que pertenece la orden recibida, realizar escaneo del valor de *minor.*
3. Tras la identificación de la orden por su correspondiente valor de *minor*, emitirla en el dispositivo de forma gráfica y auditiva.

En la Figura 5-13 se muestra un ejemplo de presentación en pantalla de la orden *Acuda a puente*.

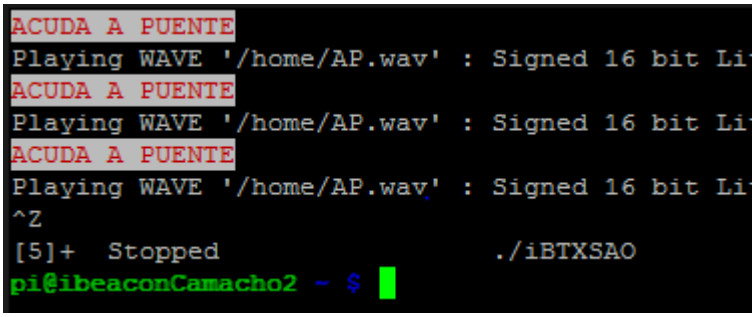


Figura 5-13. Captura de orden Acuda a puente.

La implementación de estos dos sistemas se ha llevado a cabo mediante la creación de un *script* que ejecuta los comandos necesarios en cada una de las acciones. A continuación, se enumeran en orden las acciones que debe ejecutar el *script*:

1. Transmisión continua de paquete LEAd en modo *iBeacon*.
2. Escaneo continuo de señales *Low Energy.*
3. Registro de *UUID, Major* y *minor.*

4. Identificación de sistema de alarma o envío de órdenes según *UUID*.
5. Identificación de alerta u orden, *Major y minor* (base de datos).
6. Presentación de alerta u orden gráficamente.
7. Reproducción de alerta u orden en audio.

La creación del *script* descrito anteriormente se encuentra explicada, paso a paso, en el Anexo IV del presente proyecto.

5.3 Escenario de detección de hombre al agua

Uno de los objetivos del sistema de control de personal desarrollado en el presente proyecto es la detección de la situación de hombre al agua. Esta situación puede llegar a determinarse analizando las siguientes hipótesis:

1) La situación de hombre al agua en un buque puede producirse solamente cuando una persona se encuentra en exteriores del buque, siendo imposible darse dicha situación si el individuo se encuentra en compartimentos interiores.

2) Si un *iBeacon* no es registrado por un escáner situado en exteriores, puede ser por tres motivos diferentes:

i. ***iBeacon* inoperativo:** Esta situación puede producirse por mal funcionamiento, falta de batería o avería.

ii. ***iBeacon* accede a interiores:** En esta situación, el *iBeacon* pasa de estar situado en la zona de cobertura del escáner de exteriores a la zona de cobertura de un escáner en interiores.

iii. ***iBeacon* cae por la borda:** En esta situación, el *iBeacon* sale de la zona de cobertura de un escáner en exteriores sin entrar en la zona de cobertura de ningún otro escáner, ya sea de interiores o de exteriores.

Presentadas las hipótesis, se plantean las alternativas que ofrece el sistema *iBeacon* para cada una de ellas.

En primer lugar, con lo expuesto en el apartado 5.1 del presente capítulo, se puede afirmar que en un escenario de localización a bordo es posible diferenciar cuando un *iBeacon* se encuentra en un compartimento interior o en una zona de exteriores. Por tanto, se descarta la situación de hombre al agua con una localización positiva en interiores. Además, se ha visto que es posible determinar el momento en el que un *iBeacon* accede o abandona exteriores realizando una correcta colocación de los escáneres BLE en los accesos. Por tanto, se resuelve lo planteado en la primera hipótesis y los puntos ii y iii de la segunda.

Por otro lado, si el *iBeacon* se encuentra inoperativo es imposible la recepción de la señal por un escáner en exteriores, siendo elevada la posibilidad de falsa situación de hombre al agua. La inoperatividad de un *iBeacon* puede ser consecuencia de avería, mal funcionamiento o falta de batería.

La situación de avería o mal funcionamiento de las balizas es completamente impredecible, presentando como única solución la realización periódica de revisiones y mantenimientos de los dispositivos. Por otro lado, la falta de batería es predecible haciendo uso de los indicadores visuales de carga remanente en las fuentes de alimentación, dependiendo completamente del factor humano.

Con el fin de aislar el factor humano de la gestión de carga de las fuentes de alimentación del sistema *iBeacon*, se plantea un sistema de aviso de nivel de batería que permita a éste informar el nivel de carga remanente a los escáneres próximos.

5.3.1 *Sistema de aviso de nivel de batería*

Con las configuraciones realizadas hasta ahora al sistema *iBeacon*, se han creado esclavos capaces de emitir una señal caracterizada por *UUID*, *Major*, *minor* y *Tx Power* al mismo tiempo que pueden realizar un escaneo de alarmas *iBeacon* enviadas por un maestro con el rol de escáner.

Como se explicó en el Capítulo 4, el valor de *Tx Power* indica la potencia teórica transmitida por un *iBeacon* a un metro de distancia, permitiendo realizar la estimación de proximidad con la fórmula (1) presentada en dicho capítulo. Sin embargo, en el presente proyecto la distancia no es calculada directamente por el escáner receptor, sino que debe realizarse con una aplicación externa (hoja Excel, etc.) donde el valor de *Tx Power* es introducido por defecto sin ser necesario utilizar el valor recibido por parte de un *iBeacon*. Por tanto, para el desarrollo del sistema de aviso de nivel de batería se utilizará el campo *Tx Power* como indicador configurable.

El *iBeacon* transmitirá un valor de *Tx Power* correspondiente al nivel de energía remanente en su fuente de alimentación:

- **Nivel de batería alto:** *Tx Power = -56 dBm*
- **Nivel de batería medio:** *Tx Power = -20 dBm*
- **Nivel de batería bajo:** *Tx Power = -1dBm*

El escáner, al recibir la información del valor de *Tx Power*, lo analizará y será capaz de determinar el nivel de batería del *iBeacon* que se encuentra dentro de su zona de cobertura.

Para la implementación de este sistema es necesario que el *iBeacon* conozca el nivel de batería de su fuente de alimentación en cada momento. La placa RPi y el SO *Raspbian* no proporcionan esta funcionalidad por defecto. Sin embargo, desarrolladores particulares [28] han creado una placa controladora de fuentes de alimentación conectable a RPi conocida con el nombre de *MoPi*. En la Figura 5-14 se muestra físicamente la placa.

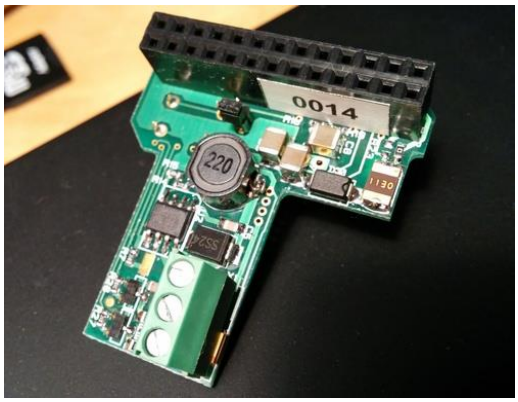


Figura 5-14. Placa *MoPi* [28].

La placa *MoPi* permite la alimentación de un RPi con varias fuentes de alimentación ya que posee dos entradas de baterías, recargables o no, con un voltaje mínimo de 6.2V y máximo de 20V. Al tener dos entradas, mantiene el RPi alimentado aun cuando una de las baterías no tenga carga suficiente evitando la interrupción no controlada del RPi. Cabe destacar que el RPi sólo necesita 5V para su funcionamiento, pero la instalación de *MoPi* eleva la alimentación total del sistema a 6.2V debido al consumo extra requerido por la placa para desarrollar sus funcionalidades (led indicador, componentes electrónicos, etc.).

MoPi se conecta a RPi a través de los puertos *GPIO* (*General Purpose Input Output*) utilizando el *bus I2C* (*Inter-Integrated Circuit)* para el intercambio de datos entre ambas placas. Una vez realizada la conexión física, es necesario instalar en RPi el controlador que permita configurar y gestionar las funcionalidades de *MoPi*. Para ello, se ejecuta el comando de instalación del paquete controlador: *sudo apt-get simbamond*.

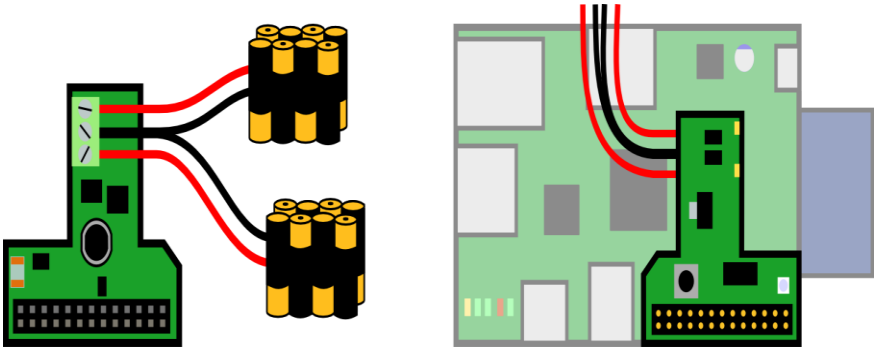


Figura 5-15. Alimentación y conexión de MoPi.

Una vez conectada la placa *MoPi*, como se muestra en la Figura 5-15, e instalado el controlador de ésta, se realiza la configuración del RPi para operar con baterías a través de *MoPi*. Es necesario definir los niveles de voltaje en los que van a operar las baterías clasificándolos en: voltaje máximo, voltaje bueno, voltaje bajo y voltaje crítico. Para ello, se ejecuta el comando *sudo mopicli -wc t mV mV mV mV*, donde *wc* indica la fuente a configurar (*-wc* fuentes 1 *y 2; -wc1* fuente 1; *-wc2* fuente 2), *t* indica el tipo de batería conectada (1-recargable; 2-no recargable) y las posiciones *mV* indican los niveles de voltaje, respectivamente. Para comprobar la configuración existente en el sistema *MoPi*, el controlador permite la ejecución del comando: *sudo mopicli –rc*, indicando los valores de voltaje almacenados en la placa. En la Figura 5-16 se muestra un ejemplo de la configuración establecida en el desarrollo del sistema de aviso de nivel de batería.

```
pi@iBeacon1 ~ $ sudo mopicli -rc
Combined conf:
  Source type: Non-rechargeable batteries
  Maximum voltage: 18000 mV
  Good voltage: 8500 mV
  Low voltage: 6500 mV
  Critical voltage: 6200 mV
```

Figura 5-16. Configuración de niveles de voltaje en *MoPi*.

Con *MoPi* conectada, instalada y configurada, se ejecuta el comando *sudo mopicli –v*. Este comando permite conocer en cada momento el valor de voltaje remanente de la fuente de alimentación conectada a RPi a través de *MoPi*. En el desarrollo de este sistema se utilizaron 2 baterías alcalinas no recargables de 9 V conectadas en serie al *MoPi*. Al ejecutar el comando anterior se obtuvo: *Current source voltage: 9360*, indicando que el voltaje remanente en la fuente de alimentación es 9.36 V en ese momento.

Utilizando este mecanismo es posible conocer el nivel de batería de nuestro sistema *iBeacon* en cada instante que se quiera. Por tanto, el siguiente paso a tomar en el desarrollo del sistema de aviso de nivel de batería es la creación de un *script* ejecutable que realice las siguientes funciones:

1. Transmisión de un paquete LEAd con *UUID, Major, minor* y *Tx Power*.
2. Comprobar el nivel de batería periódicamente.
3. Modificar el valor de *Tx Power* según el nivel de batería (valores de voltaje configurados según la batería utilizada, pero siempre comprendidos entre 6,2 V y 20 V) [28].

- **Nivel de batería alto:** voltaje > 12.5 V
- **Nivel de batería medio:** 8.5V > voltaje > 7.0 V
- **Nivel de batería bajo:** voltaje < 7.0 V

4. Transmisión de un paquete LEAd con *UUID, Major, minor* y nuevo *Tx Power* correspondiente al nivel de batería.

El desarrollo del *script* que permite la implementación del sistema de aviso de nivel de batería se encuentra detallado paso a paso en el Anexo V.

Por último, es necesaria la modificación del código del *script* creado en el Capítulo 4 para que el escáner analice el valor de *Tx Power* (*-56 dBm, -20 dBm y -1 dBm*), identifique el nivel de batería del *iBeacon* en su zona de cobertura y lo muestre por pantalla. Para ello, se incluyen diferentes líneas de código al *script SCAN* ya creado. El código final se incluye en el paso 5 del Anexo III.

En las siguientes figuras se muestran varias capturas realizadas al *iBeacon* transmitiendo con el sistema de aviso de nivel de batería activado durante un periodo de tiempo, así como las capturas paralelas de los resultados presentados en pantalla por un escáner configurado con el sistema de lectura de nivel de batería.

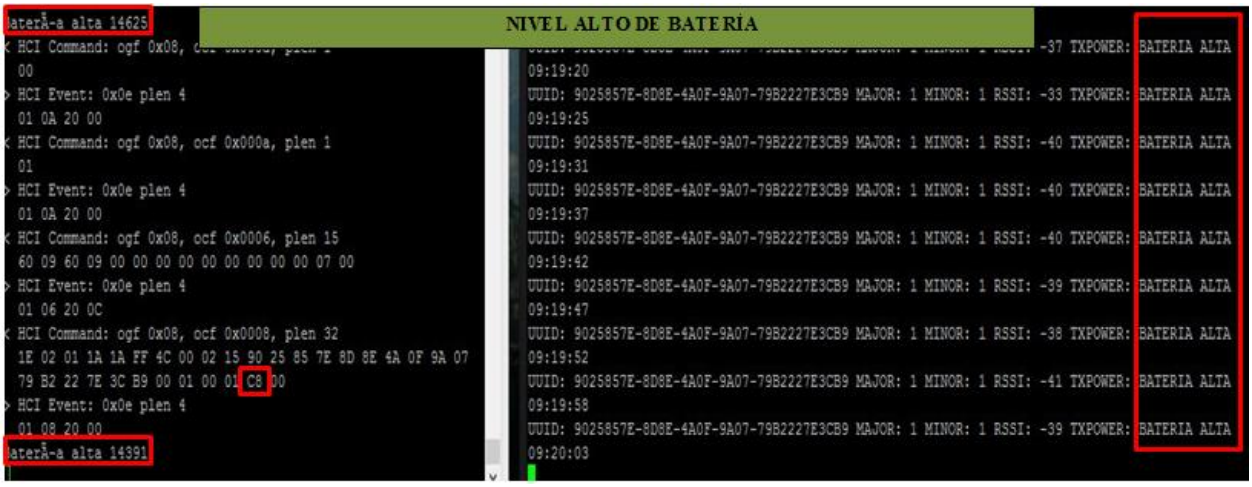


Figura 5-17. Sistema de aviso de nivel de batería: Nivel alto.

Figura 5-18. Sistema de aviso de nivel de batería: Nivel medio.

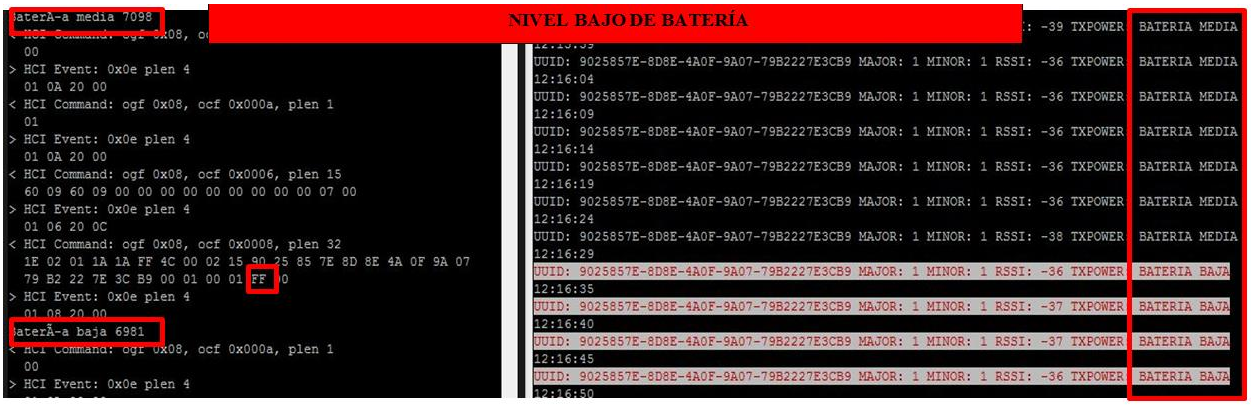


Figura 5-19. Sistema de aviso de nivel de batería: Nivel bajo.

A la izquierda (*iBeacon* en transmisión), se muestran los valores de batería que *MoPi* analiza. También se puede observar la configuración del valor de *Tx Power* (*C8: - 56 dBm; EC: -20 dBm; FF: -1 dBm*) que realiza el *script* del sistema de aviso de nivel de batería según el valor del voltaje remanente. A la derecha, se puede observar la lectura realizada por el escáner mostrando el diagnóstico del nivel de batería recibido por parte del *iBeacon*.

Una vez que los escáneres, dentro de un escenario de localización, son capaces de conocer el último valor del nivel de batería de un *iBeacon* del cual se pierde la señal en exteriores, se puede establecer un nivel de probabilidad de que la situación de hombre al agua sea real o no. En el caso de que el último valor registrado sea *batería baja*, la probabilidad de hombre al agua real es baja. Por otro lado, si el último registro coincide con un valor de *batería alta*, la probabilidad de un hombre al agua real es elevada.

Sin embargo, sigue siendo imposible descartar completamente la hipótesis de inoperatividad del *iBeacon* planteada al inicio de este apartado. Una posible solución es la redundancia de dispositivos, es decir, que el personal porte dos *iBeacon* con semejantes configuraciones, reduciendo la probabilidad de que se produzca la inoperatividad de estos en un mismo instante.

6 ENTORNO INTELIGENTE

Una vez desarrollado el sistema *iBeacon* con las funcionalidades de transmisor, escáner, sistema de alarma y sistema de aviso de nivel de batería y, tras realizar las pruebas de funcionamiento que validan su operatividad, en el presente capítulo se plantea el concepto de *Entorno Inteligente* o *Sistema ciberfísico* y la implementación teórica en un buque de la Armada.

El científico *Mark Weiser* definió *Entorno Inteligente* como: *"un mundo físico que está ricamente entretejido e invisible con sensores, actuadores, visualizadores y elementos computacionales, integrados a la perfección en los objetos cotidianos de nuestras vidas y conectados a través de una red continua"* [30]. Este concepto ha evolucionado en los últimos años con los importantes avances en componentes y sistemas de las tecnologías de la información y la comunicación, permitiendo la creación de *sistemas ciberfísicos* capaces de controlar entidades físicas haciendo uso de elementos computacionales [31].

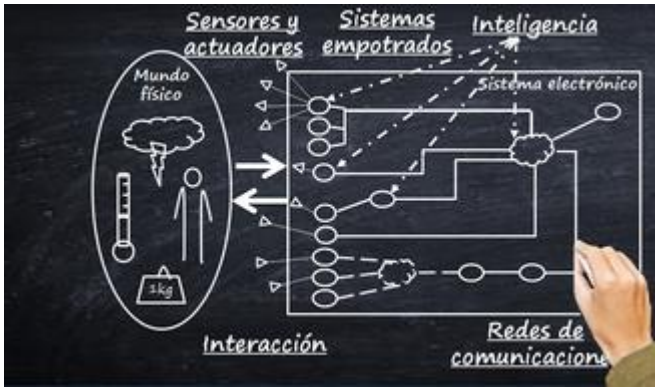


Figura 6-1. Esquema de entorno inteligente [30].

Las entradas y salidas del sistema son sensores y actuadores, que pueden ser simples transductores o dispositivos inteligentes que integran cierta capacidad de proceso y/o comunicación. La inteligencia puede estar distribuida en sistemas empotrados o centralizada en grandes servidores. Los dispositivos se organizan en redes de comunicaciones que pueden ser cableadas, inalámbricas o híbridas.

Si se aplica lo anterior al sistema *iBeacon* desarrollado en el presente proyecto y asumiendo que en un barco se puede crear un sistema que centralice la inteligencia en servidores, podremos afirmar que un buque de la Armada puede convertirse en un *Entorno Inteligente* (EI) que controle y sea capaz de intercambiar información con el personal a bordo.

6.1 Infraestructura del Entorno Inteligente

Para llevar a cabo la implementación de un EI, es necesaria la creación de una infraestructura que albergue un servidor capaz de gestionar la información presentada por el sistema *iBeacon* y actuar en consecuencia. Se trataría de la instalación de una red de escáneres conectados a un punto de acceso común y situados en los diferentes compartimentos del buque en los que se quiera controlar e interactuar con el personal embarcado. Esta red debe ser gestionada por un servidor y debe presentar la posibilidad de ser controlada por una aplicación intuitiva al usuario. En la Figura 6-2 se muestra el esquema básico de una posible red de tres escáneres *iBeacon* gestionada por un servidor común y controlada por una aplicación (usuario).

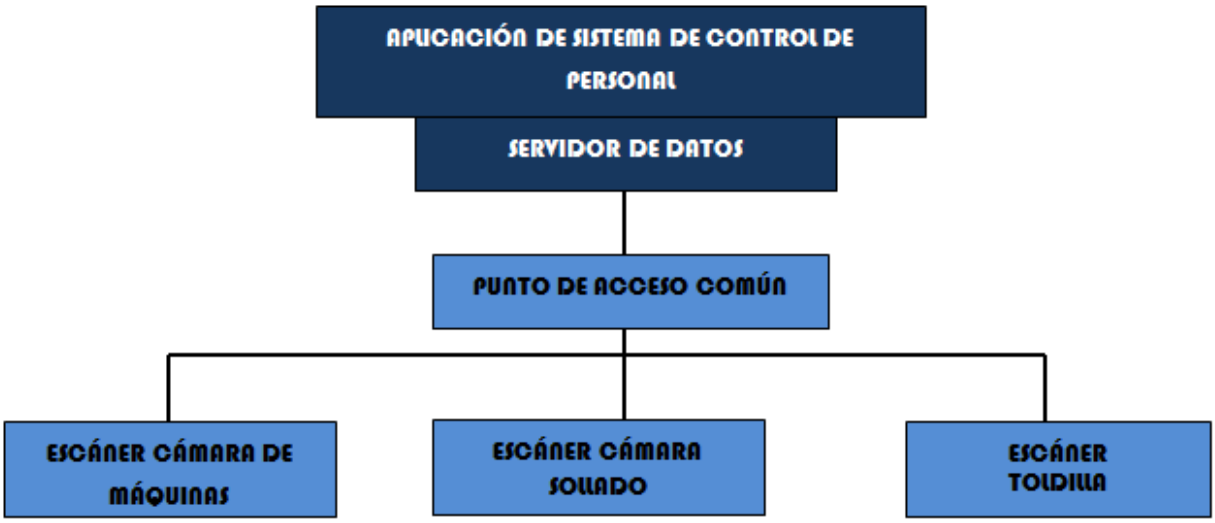


Figura 6-2. Esquema básico de entorno inteligente en buques.

Los escáneres situados en los diferentes compartimentos son los encargados de la localización de los *iBeacon* presentes en su zona de cobertura, recibiendo datos de identificación (*UUID, Major* y *minor)*, datos de interés como el nivel de batería y valor de *RSSI* para permitir definir la proximidad al escáner. Además, son los encargados de transmitir las señales de alerta en su zona de cobertura.

En un nivel superior se encuentra el punto de acceso común a los escáneres. Este punto de acceso es el encargado de centralizar la conexión cableada a los diferentes nodos, direccionando a estos los datos provenientes del servidor.

El servidor de datos contiene archivos y registros de la infraestructura del buque (relación de escáner y compartimento), registro de identidad de personal (*Major, minor*), base de datos de alarmas y órdenes a transmitir y es el encargado de volcar y almacenar la información recibida por los escáneres en la bases de datos.

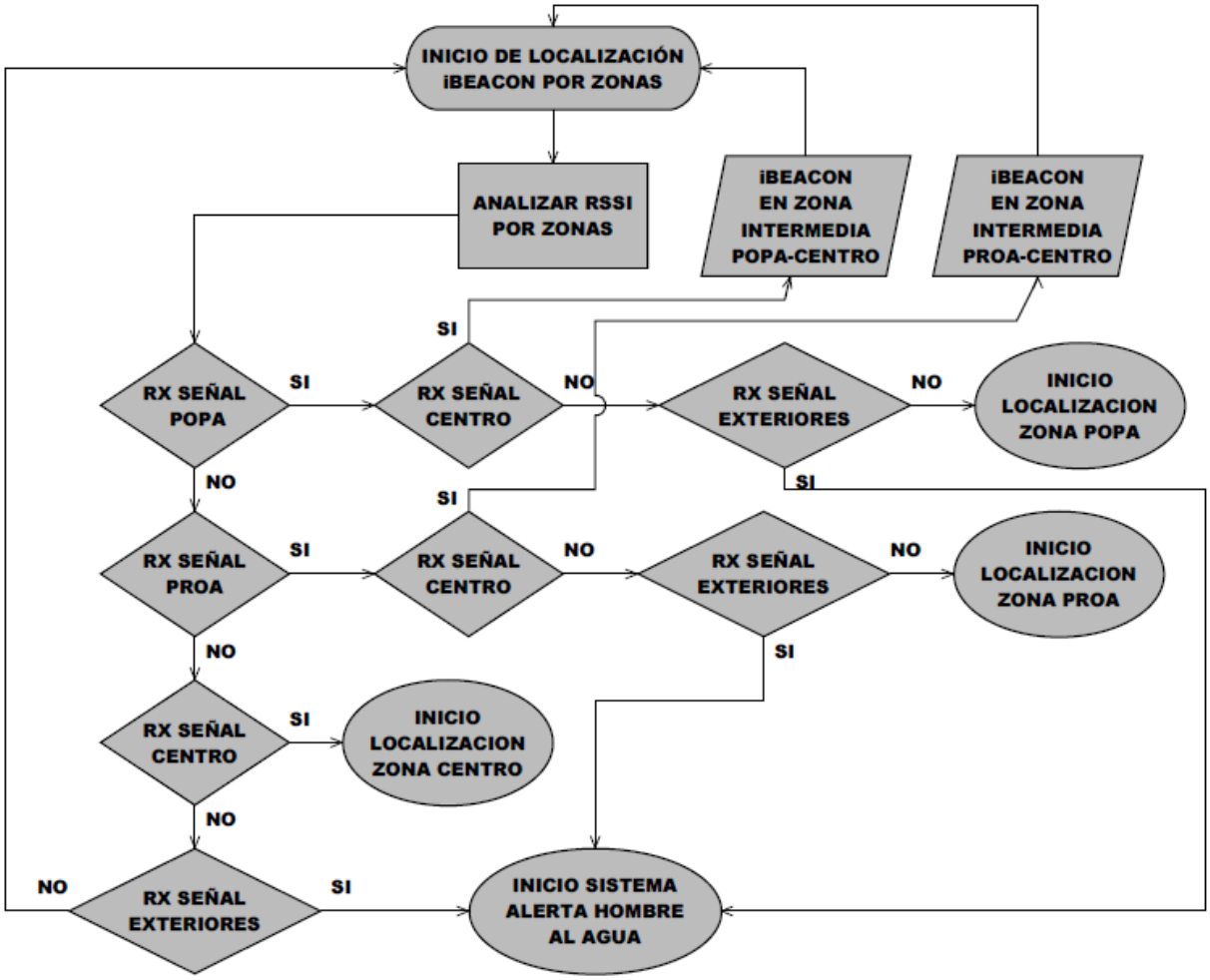
En la parte superior del esquema, se encuentra la aplicación del sistema de control de personal. Presenta una interfaz que permite al usuario el control de los datos contenidos en el servidor y la ejecución de las diferentes lógicas que permitan al sistema realizar las siguientes funciones:

- **Localización de personal por compartimentos o zonas.**
- **Envío de alarmas y señales de aviso.**
- **Sistema de alerta de hombre al agua.**

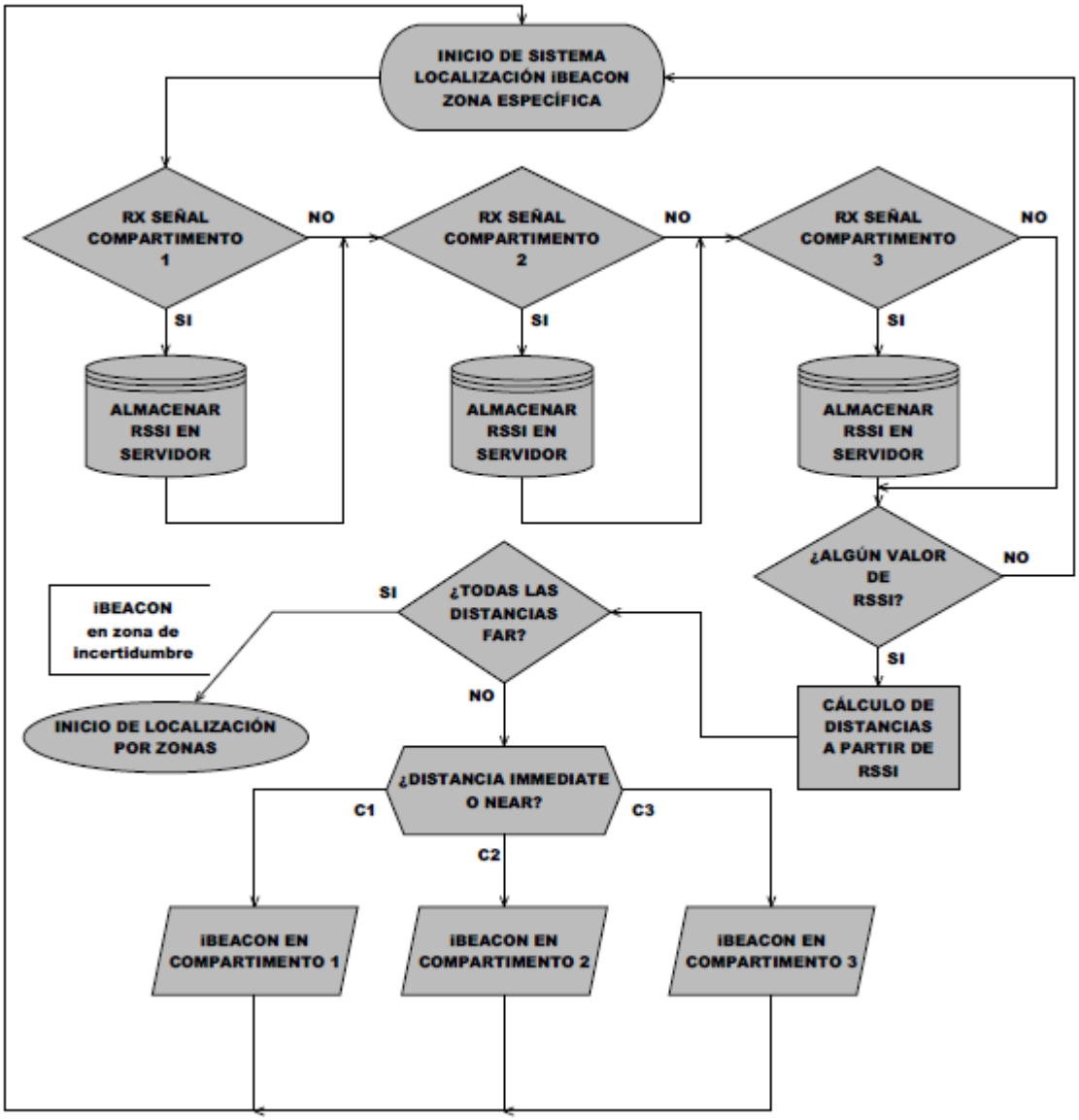
A continuación, se definen las diferentes lógicas y acciones que deben realizarse en los diferentes componentes de la infraestructura para llevar a cabo las anteriores funcionalidades.

6.2 Lógica del sistema de localización de personal por compartimentos o zonas

Para realizar la función de localización de personal, el EI debe ser capaz de desarrollar la lógica mostrada en el siguente diagrama:



El sistema comienza analizando la recepción de señal por zonas del buque. Estas zonas abarcan varios escáneres y están definidas de tal forma que se eviten en lo posible zonas de incertidumbre o no recepción de señales por escáneres de las zonas adyacentes. Se asume que un *iBeacon* en la zona de popa podrá ser detectado por los escáneres contenidos en esa zona, o, en algún caso, por la zona adyacente a ésta (exteriores o centro). Lo mismo ocurre con la zona de proa. Una vez que un *iBeacon* es localizado en una zona determinada, se ejecuta el proceso de localización propio de la zona. A continuación, se muestra la lógica utilizada para localización por zonas específicas:



En este proceso, se analiza la RSSI recibida por los escáneres contenidos en la zona específica. El valor de RSSI y los identificadores (*UUID, Major, minor y Tx Power)* son volcados y almacenados por el escáner en el servidor a través del punto de acceso. La aplicación que controla al servidor calcula la distancia según el valor de RSSI y determina en qué rango se encuentra: *Immediate, Near* o *Far.* Si el valor es *Near o Immediate* se asume que el *iBeacon* se encuentra en el compartimento que ha registrado ese valor. Si todos los registros se encuentran en el rango de distancias *Far,* se reinicia el proceso de localización por zonas.

En este proceso, puede plantearse el caso de detección de un rango de distancias *Immediate o Near* por parte de más de un escáner, lo que imposibilitaría la ejecución de la lógica correctamente. Para evitar esta situación, es importante una correcta colocación de los escáneres a bordo, con la suficiente separación entre ellos que imposibilite la existencia de un punto equidistante a dos escáneres dentro de un rango *Immediate* o *Near.*

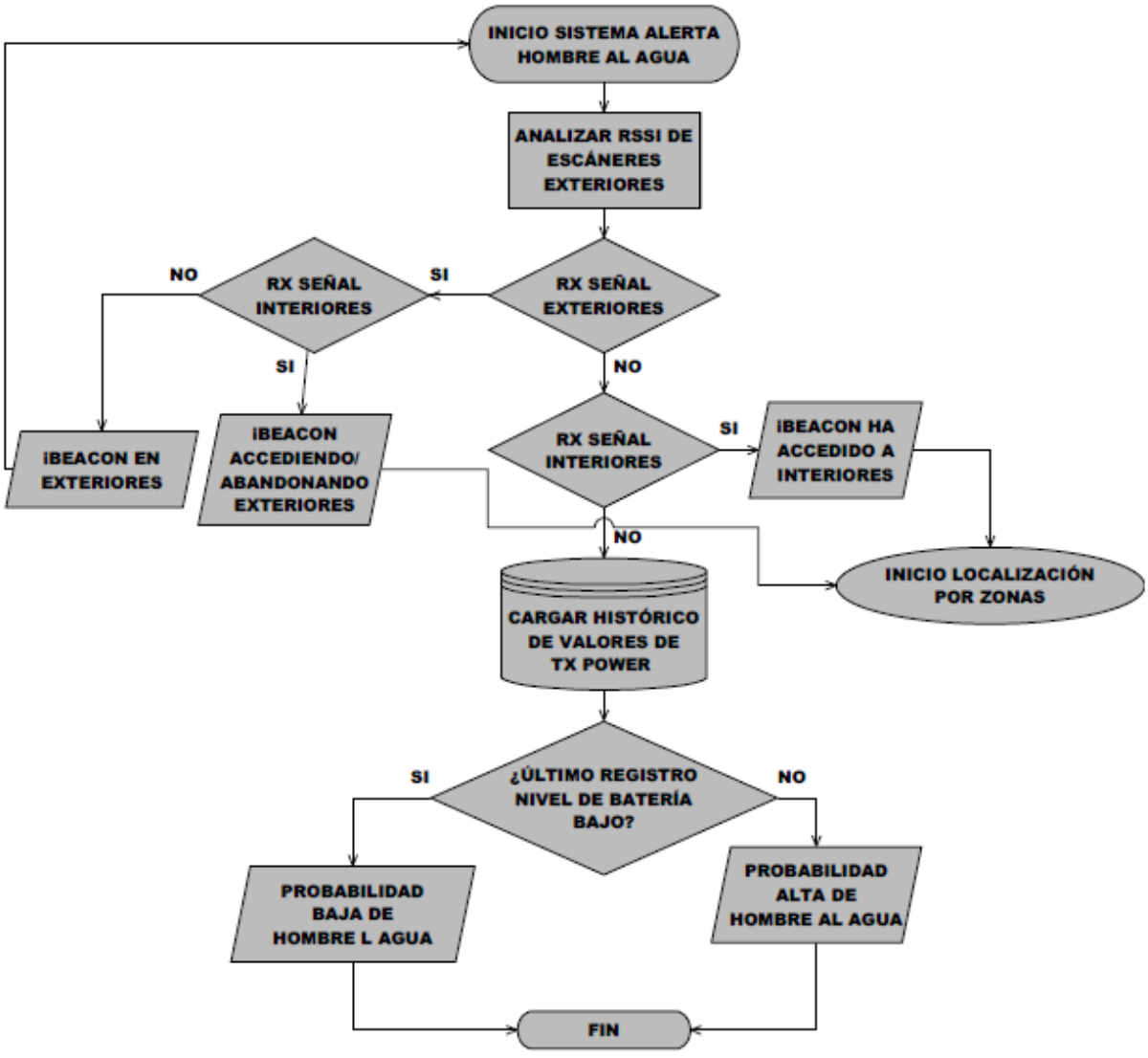
6.3 Lógica del sistema de envío de alarmas y órdenes

Para el desarrollo de la funcionalidad de envío de alarmas y órdenes bastará con que la aplicación configure el escáner para que envíe un paquete LEAd en el compartimento o zona que se requiera. Por

ejemplo: en el caso de zafarrancho de combate, la aplicación carga la alerta correspondiente del servidor de datos y realiza la configuración de todos los escáneres para que la emitan. En el caso de una incidencia en la cámara de máquinas, la aplicación carga la alerta de abandono de compartimento desde el servidor de datos y configura el escáner de dicho compartimento para que la emita.

6.4 Lógica del sistema de alerta de hombre al agua

En el caso de que un *iBeacon* se encuentre en un compartimento exterior, se ejecutará la siguiente lógica que permitirá determinar la situación de hombre al agua.



En la implementación de esta funcionalidad es esencial el sistema de aviso de nivel de batería. La lógica permite definir un grado razonable de probabilidad de detectar la situación de hombre al agua asumiendo que esta situación solo se dará cuando un individuo se encuentra en una zona exterior.

La no detección de señal por un escáner exterior puede suceder por diversos motivos como los expuestos en el Capítulo 5. La lógica propuesta permite determinar varios de los motivos anteriores. Sin embargo, un mal funcionamiento del sistema o avería de éste son sucesos poco predecibles. Por tanto, el sistema de alerta de hombre al agua define solamente el nivel de probabilidad de ocurrencia de esta situación, siendo necesaria la comprobación por otros métodos.

7 CONCLUSIONES Y LÍNEAS FUTURAS

Tras el estudio realizado de las diferentes tecnologías que se han aplicado hasta el momento para la localización en interiores, se puede aceptar la viabilidad del uso de la tecnología *Bluetooth Low Energy* para el desarrollo de un sistema de control de personal a bordo.

El bajo consumo energético, el bajo coste, la facilidad de implantación de dicha tecnología y el tamaño relativamente reducido de los componentes que la implementan, la convierten en la mejor opción para la configuración de un sistema *iBeacon* capaz de localizar personal a bordo de buques e intercambiar información con ellos. Este sistema *iBeacon* resuelve el problema del control de personal en entornos cerrados salvando algunas limitaciones: la localización se realiza por proximidad, lo que limita ésta a zonas que deben ser acotadas previamente; el sistema *iBeacon* debe ser implementado conjuntamente con un *software* dedicado que sea capaz de gestionar la información necesaria para llevar a cabo el control de personal (registro de valores de RSSI recibidos en los escáneres, ejecución de las lógicas del sistema de localización, el sistema de alerta de hombre al agua y el sistema de alarmas y órdenes).

Por otro lado, la utilización de una placa RPi para la construcción de los tres prototipos de pruebas deja en evidencia las limitaciones de los mismos. El tamaño de los prototipos, así como el consumo registrado por el sistema *iBeacon* implementado sobre esta placa, son factores que hacen poco recomendable la implementación de *iBeacon* sobre RPi para los dispositivos que transporta el personal a controlar. Además, para la completa implementación del sistema de alertas y órdenes sería necesaria la instalación de altavoces y una pantalla en los dispositivos, aumentando el consumo y el tamaño de los prototipos.

No es el caso de los escáneres utilizados como infraestructura fija, ya que estos, pueden ser alimentados por la instalación eléctrica del buque y estar conectados a una red cableada de datos que permita volcar la información en servidores que un usuario pueda gestionar.

Expuesto lo anterior, se plantean como futuras líneas de investigación y mejora del presente proyecto las siguientes:

1. Estudio de la implementación del sistema *iBeacon* portátil en dispositivos de tamaño y consumo reducido: se propone el empleo de teléfonos móviles con tecnología BLE, microchips especializados en BLE (Figura 7-1), que permitan llevar a cabo las funciones del sistema en todos los escenarios propuestos con un consumo energético aceptable.

Figura 7-1. Placa de desarrollo BLE. Serie: DB-CSR1010-10137-1A-ND [29].

2. **Desarrollo del *software* dedicado:** que permita gestionar las funcionalidades que el sistema *iBeacon* ofrece para el control de personal a bordo en los diferentes escenarios propuestos. Este *software* debe ser capaz de ejecutar las diferentes lógicas para cada uno de los escenarios propuestos en el presente proyecto. Además, debe ofrecer una interfaz al usuario que le permita, de forma intuitiva, acceder a los datos sobre localización de personal, el intercambio de información y la gestión de la base de datos que contiene los identificadores personales (*UUID, Major y minor)* así como las diferentes alarmas y órdenes a enviar.
3. **Levantamiento de mapas de cobertura de los RPi en modo escáner:** que permitan definir correctamente los puntos de instalación de los escáneres en la infraestructura fija a bordo de los buques. Con lo anterior, es posible limitar la detección de *iBeacons* a rangos de distancia *Near* o *Immediate*, descartando aquellas balizas en rangos *Far* que introducen incertidumbre al sistema de control de personal a bordo.
4. **Desarrollo de entorno inteligente:** que permita validar la viabilidad del sistema *iBeacon* de control de personal a bordo de los buques de la Armada una vez realizadas las mejoras anteriores. Este desarrollo implica: diseño de zonas a compartimentar del buque; la instalación de una infraestructura fija de escáneres en un buque; la creación de una red cableada que permita el intercambio de datos con un servidor común gestionado por el *software* dedicado; asignación de *iBeacons* a personal de dotación.

El desarrollo del entorno inteligente permitiría a su vez la realización de un mayor número de pruebas:

- **Pruebas de verificación de las lógicas propuesta para cada escenario.**
- **Pruebas de comprobación de capacidad del sistema de localización:** para determinar el número máximo de *iBeacons* detectables por un escáner en una zona determinada.
- **Pruebas de autonomía:** que permitan establecer los ciclos de mantenimiento y recarga de las balizas.

5. **Estudio de la versión 4.2 de Bluetooth:** como se expuso en el Capítulo 3, en la actualidad se encuentra en desarrollo la versión 4.2 de la tecnología *Bluetooth* que, entre algunas mejoras, incluye la capacidad de operar con IPv6 manteniendo los principios de bajo coste y bajo consumo energético. Debido a esto se plantea, como línea futura de investigación, el estudio de dicha versión y su posible utilización para el desarrollo del sistema *iBeacon* de control de personal a bordo con capacidad de comunicación *full-dúplex* (voz y datos).

8 BIBLIOGRAFÍA

[1] L. D. Ambrona, Sistemas de localización en interiores, PFC , Facultad de Informática, Universidad Politécnica de Madrid.

[2] F. S. Granja, Sistemas de localización en interiores basados en radiofrecuencia, Instituto de Automática Industrial (CSIC) [En línea]: www.car.upm-csic.es/lopsi/static/publicaciones [Último acceso: 9 enero 2015]

[3] AeroScout Indrustial, *Tracking solutions* [En línea]: http://www.aeroscout.com/solutions [Último acceso: 9 enero 2015].

[4] GrupoNeat, Localización indoor[En línea]: http://www.gruponeat.com/index.php?ids=906&lang=es [Último acceso: 9 enero 2015].

[5] S. Tadakamadla, *Indoor Local Positioning System for ZigBee, based on RSSI* [En línea]: http://www.researchgate.net/publication/242665667_Indoor_Local_Positioning_System_For_ZigBee_Based_On_RSSI [Último acceso: 9 enero 2015].

[6] «Intel press» [En línea]: http://www.intelfreepress.com/news/the-man-who-named-bluetooth/4078/ [Último acceso: 10 enero 2015].

[7] Massachusetts Institute of Technology, *The Cricket Indoor Location System* [En línea]: http://cricket.csail.mit.edu/#overview. [Último acceso: 23 diciembre 2014].

[8] IndraCompany, Proyecto Sherloc [En línea]: http://www.indracompany.com/sostenibilidad-e-innovacion/proyectos-innovacion/sherloc-sistema-hibrido-de-estres-remoto-y-localiza [Último acceso: 11 enero 2015].

[9] M. S. y. S. B. David Schwarz, *Cosero, find my Keys! Object Localization and Retrieval using Bluetooth Low Energy Tags*, Rheinische Friedrich-Wilhelms-Universitat [En línea]: http://www.robocup2014.org/?page_id=3267 [Último acceso: 23 diciembre de 2014].

[10] LiberaCompany,*Whitepaper* RFID [En línea]: http://www.libera.net/uploads/documents/whitepaper_rfid.pdf. [Último acceso: 27 diciembre 2014].

[11] Apple. Inc, *iBeacons for developers* [En línea]: https://developer.apple.com/ibeacon/. [Último acceso: 29 diciembre 2014].

[12] Blueglue Ltd, *Naval Automated Personnel Tracking* [En línea]: http://www.blueglue.com.au/products_2/personnel_tracking.html [Último acceso: 28 diciembre 2014].

[13] B. SIG,*Company Identifiers* [En línea]: https://www.bluetooth.org/en-us/specification/assigned-numbers/company-identifiers. [Último acceso: 14 enero 2015].

[14] *International Organization for Standardization*, ISO/IEC 11578:1996: *Remote Procedure Call* (RPC),» [En línea]: http://www.iso.org/iso/catalogue_detail.htm?csnumber=2229. [Último acceso: 15 enero 2015].

[15] GitHub, *Reliable Bluetooth LE (iBeacon) scanner for Raspberry Pi*,[En línea]: https://github.com/switchdoclabs/iBeacon-Scanner-. [Último acceso: 15 enero 2015].

[16] C. B. T. C. Avik Ghose, *BlueEye – A System for Proximity Detection Using Bluetooth on Mobile Phones*, TCS Innovation Labs [En línea]: http://www.researchgate.net/publication/262277611_BlueEye_a_system_for_proximity_detection_using_bluetooth_on_mobile_phones [Último acceso: 15 enero 2015].

[17] Ubuntu packages, [En línea]: http://www.packages.ubuntu.com/. [Último acceso: 16 enero 2015].

[18] App Store, Utilidades [En línea]: https://itunes.apple.com/es/genre/ios-utilidades/id6002?mt=8 [Último acceso: 18 enero 2015].

[19] Bluetooth Developer portal, *Profiles overview*, [En línea]: https://developer.bluetooth.org/TechnologyOverview/Pages/Profiles.aspx. [Último acceso: 17 enero 2015].

[20] *The Linux Kernel Organization*, [En línea]: https://www.kernel.org/category/about.html. [Último acceso: 17 enero 2015].

[21] Bluetooth Org, *Core Specification - Bluetooth*, [En línea]: https://www.bluetooth.org/docman. [Último acceso: 7 febrero 2015].

[22] *Bluetooth* IEEE 802.15.1 [En línea]: http://www.ieee.org. [Último acceso: 02 febrero 2015].

[23] *TDD and FDD Explained* [En línea]: http://www.exaltcom.com/TDD-and-FDD-Explained.aspx. [Último acceso: 2 febrero 2015].

[24] Anexo A, Protocolo Bluetooth [En línea]: http://ants.dif.um.es/~felixgm/docencia/android [Último acceso: 4 febrero 2015].

[25] Bluetooth Org, *Specification of The Bluetooth System* [En línea]: www.bluetooth.org. [Último acceso: 6 febrero 2015].

[26] Wikipedia, *Advanced Encryption Standard* [En línea]: http://en.wikipedia.org/wiki/Advanced_Encryption_Standard. [Último acceso: 7 febrero 2015].

[27] *Bluetooth Developer, GATT Services* [En línea]: https://developer.bluetooth.org/gatt/services/Pages/ServicesHome.aspx. [Último acceso: 7 febrero 2015].

[28] MoPi, *Hot-Swap Mobile Power for the Pi* [En línea]: https://pi.gate.ac.uk/pages/mopi.html. [Último acceso: 17 febrero 2015].

[29] CSR Products, *Development Boards and Development Kits Evaluation* [En línea]: http://www.csr.com/products/csr101x-product-family. [Último acceso: 16 febrero 2015].

[30] Universidad de Málaga, Máster Oficial en Sistemas Electrónicos para Entornos Inteligentes [En línea]: http://www.masterseeiuma.es/entornos-inteligentes-1/ [Último acceso: 21 febrero 2015].

[31] *Cyber- Physical Sistems* [En línea]: http://cyberphysicalsystems.org/ [Último acceso: 27 febrero 2015].

ANEXO I. MANUAL DE CONFIGURACIÓN DE RPI: OPERACIÓN CON BLE

En el presente anexo se exponen los pasos a seguir para la configuración de RPi para operar con tecnología BLE.

Paso 0: Hardware necesario.

- Raspberry Pi model B+.
- MicroSD memory card con un mínimo de 2 GB de memoria.
- Módulo *Bluetooth* CSR 4.0 USB.
- Adaptador WiFi USB (opcional).
- Fuente de alimentación de 5V DC 1500-2000mA (Batería o alimentación de red).
- Cable HDMI.
- Pantalla con entrada HDMI.
- Teclado con conexión USB.
- Ordenador con conexión a Internet.

Paso 1: Instalación del SO Raspbian Wheezy en RPi.

1.1 Acceder a la página <http://www.raspberrypi.org> e ir a la sección “downloads”.

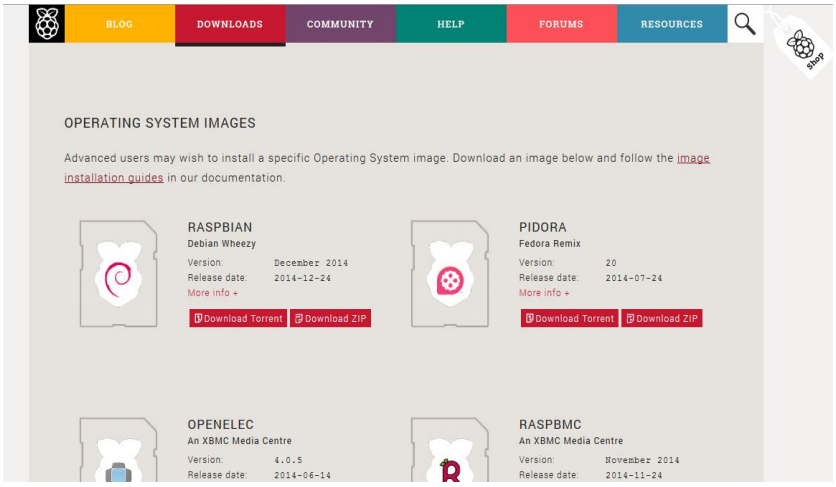


Figura AI-1. Página de inicio de Raspberrypi.org.

1.2 Descargar el archivo RASPBIAN en formato ZIP y descomprimirlo en un directorio conocido (escritorio).

1.3 Insertar la tarjeta microSD en su ordenador.

En este momento, procedemos a crear un ejecutable del SO en la microSD que permita a RPi cargarlo y ejecutarlo para la instalación de Raspbian. Para ello, se utiliza el *software Win32DiskImager* para montar la imagen en la microSD.

1.4 Descargar e instalar *Win32DiskImager*.
Link: http://sourceforge.net/projects/win32diskimager/

1.5 Abrir el *Win32DiskImager* como administrador. Donde pone *device*, seleccionar la microSD y, donde aparece el *icono* de carpeta, seleccionar el archivo del Raspbian descomprimido anteriormente.

Tras estos pasos, el SO Raspbian se encuentra preparado en la microSD para ser instalado.

1.6 Conectar el RPi a la pantalla vía cable HDMI, conectar el teclado, insertar la tarjeta micro SD y la fuente de alimentación. Aparecerá un led rojo fijo y un led verde parpadeando que indica que RPi está en funcionamiento.

1.7 En el primer arranque, aparecerá una ventana de configuración de *Raspi-config*, donde podrá cambiar parámetros como la hora local. Si no, se puede hacer más adelante. En esta ventana, seleccionar la opción *expand_rootfs* y decir "*y"* para resetear.

1.8 Una vez reseteado, pedirá usuario/contraseña para acceder al sistema. Por defecto son:

- **Usuario:** pi
- **Contraseña:** raspberry

1.9 Tras el acceso, la línea de comandos aparecerán como: *pi@raspberry ~ $*.

Paso 2: Conexión a Internet del RPi.

La conexión se puede realizar por dos métodos: mediante conexión por cable Ethernet o utilizando un adaptador USB WiFi. Para el primer caso, bastará con conectar el cable al RPi y reiniciarlo con el comando *sudo shutdown -r now* y ya estaría establecida la conexión. Para el segundo caso, se debe realizar el siguiente proceso:

2.1 Comprobar que Raspbian lee y puede operar con el adaptador USB WiFi. Se ejecuta el comando *lsusb* y debería aparecer algo parecido a lo siguiente:

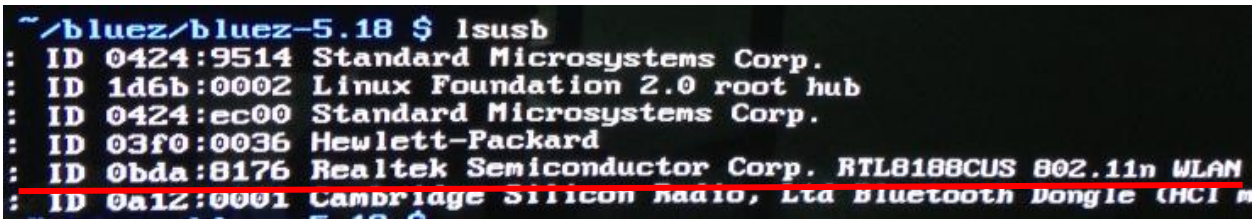


Figura AI-2. Comprobación de conexión de adaptador USB WiFi en Raspbian.

2.2 En caso de no aparecer lo anterior (Figura AI-2), es necesario instalar el driver del adaptador USB WiFi. Es necesario informarse de qué paquete compatible con Raspbian se debe instalar según el fabricante y modelo del adaptador. Un vez conocido el nombre del paquete, se instalará con el comando *sudo apt-get install nombre del paquete*.

2.3 Ejecutar, de nuevo, el paso 2.1 para comprobación.

Una vez comprobado que Raspbian puede operar con el adaptador WiFi USB, es necesario configurar la interfaz de red para establecer la conexión wireless:

2.4 Ejecutar comando *sudo nano /etc/network/interfaces* y editar el archivo para que quede lo siguiente:

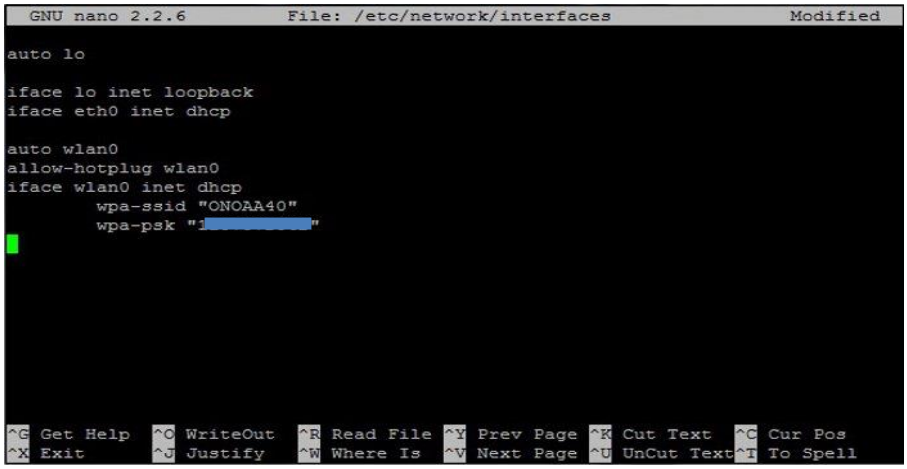


Figura AI-3. Edición de la interfaz de red con editor de texto GNU nano.

En *wpa-ssid* debe ir el nombre de la red a la que queremos conectar el RPi y en *wpa-psk* debe introducirse la contraseña de dicha red. Al finalizar la edición, guardar tecleando *ctrl+X* y reiniciar el RPi.

Paso 3: Instalación del módulo USB *Bluetooth* CSR 4.0.

En este paso se comprueba que Raspbian lee y puede operar con el módulo USB de *Bluetooth*. Para ello se procede de forma análoga al paso 2.

3.1 Comprobación. Comando *lsusb*:

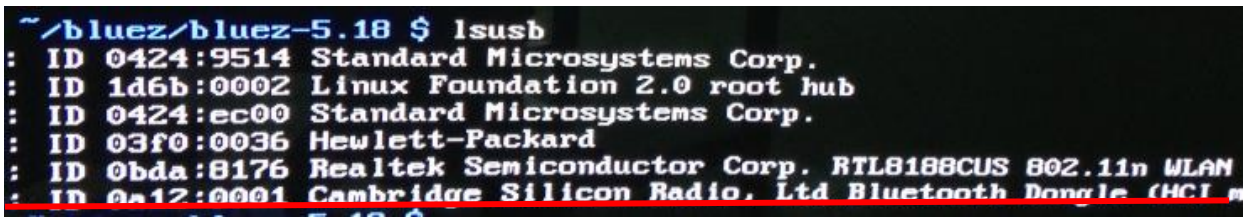


Figura AI-4. Comprobación de conexión del módulo *Bluetooth* USB en Raspbian.

3.2 En caso de no leerlo correctamente, ejecutar el paso 2.2 con el paquete de *drivers* correspondiente al módulo USB de *Bluetooth* 4.0.

Una vez instalado el SO, conectados y verificados los adaptadores WiFi/*Bluetooth* y establecida la conexión a Internet, se procede a instalar las librerías necesarias para configurar el RPi como transmisor *iBeacon*, así como las librerías BlueZ que dan soporte al protocolo *Bluetooth*.

Paso 4: Instalación de librerías para control del módulo USB de BLE.

Ejecutar los siguientes comandos en orden:

sudo apt-get update
sudo apt-get upgrade
sudo apt-get install libusb-dev
sudo apt-get install libdbus-1-dev
sudo apt-get install libglib2.0-dev --fix-missing
sudo apt-get install libudev-dev
sudo apt-get install libical-dev
sudo apt-get install libreadline-dev

Paso 5: Instalación de librerías BlueZ de soporte al protocolo *Bluetooth*.

5.1 Crear el directorio de la librería y descargar paquete *BlueZ* (desde el servidor de la *Linux Kernel Organization*). Ejecutar:

sudo mkdir bluez
cd bluez
sudo wget www.kernel.org/pub/linux/bluetooth/bluez-5.18.tar.gz
sudo gunzip bluez-5.18.tar.gz
sudo tar xvf bluez-5.18.tar

5.2 Instalación del paquete BlueZ. Ejecutar:

cd bluez-5.18
sudo ./configure --disable-systemd
sudo make
sudo make install

El comando *sudo make* puede tardar varios minutos en completarse.

5.3 Reiniciar el dispositivo. Ejecutar: *sudo reboot.*

Tras finalizar todos los pasos anteriores, el RPi está configurado y preparado para operar con la tecnología BLE.

ANEXO II. MANUAL DE CONFIGURACIÓN DE MÓDULO *BLUETOOTH*: OPERACIÓN EN MODO *IBEACON*

Tras la configuración de RPi y la instalación de los drivers Linux necesarios para operar con la tecnología BLE, se procede a configurar el módulo *Bluetooth* en modo operación *iBeacon*.

Paso 0. Conectar el módulo *Bluetooth* al puerto USB y encender RPi. Una vez en el terminal, dirigirse al directorio de la librería BlueZ instalada. Ejecutar:

cd bluez/bluez-5.18

Paso 1. Verificación de estado del módulo *Bluetooth* y sus controladores. Ejecutar:

hciconfig

```
pi@ibeaconCamacho3 ~/bluez/bluez-5.18 $ hciconfig
hci0:   Type: BR/EDR  Bus: USB
        BD Address: 00:1A:7D:DA:71:13  ACL MTU: 310:10  SCO MTU: 64:8
        DOWN
        RX bytes:1893 acl:0 sco:0 events:123 errors:0
        TX bytes:2159 acl:0 sco:0 commands:123 errors:0
```

Figura AII-1. Módulo *Bluetooth* del *iBeacon* 2 en estado DOWN.

En esta operación se pueden dar dos casos: que el servicio esté inactivo (DOWN) o en modo escáner (PSCAN). El modo correcto de trabajo debe ser UP RUNNING, para configurarlo se debe ejecutar:

sudo hciconfig hci0 up
sudo hciconfig hci0 noscan

Una vez ejecutados los comandos, se procede a verificar el estado del módulo de nuevo. Ejecutar:

hciconfig

```
pi@ibeaconCamacho3 ~/bluez/bluez-5.18 $ hciconfig
hci0:   Type: BR/EDR  Bus: USB
        BD Address: 00:1A:7D:DA:71:13  ACL MTU: 310:10  SCO MTU: 64:8
        UP RUNNING
        RX bytes:1893 acl:0 sco:0 events:123 errors:0
        TX bytes:2159 acl:0 sco:0 commands:123 errors:0
```

Figura AII-2. Módulo *Bluetooth* del *iBeacon* 2 en estado UP RUNNING (Activo).

En este momento, tanto el RPi como el módulo USB están preparados para transmitir una trama de datos *iBeacon*. A continuación, se procede a la configuración con la correspondiente clave UUID para ser detectado por la aplicación compatible que vayamos a utilizar.

Paso 3. Configuración de intervalo de transmisión (*LE_Set_Advertising_Parameters).*

El modo *LE Advertisement* permite la configuración del intervalo de tiempo entre transmisiones de datos. Para configurarlo, ejecutar:

sudo hcitool -i hci0 cmd 0x08 0x0006 XX XX YY YY ZZ 00 00 00 00 00 00 00 07 00

Dónde:

- XX XX es el intervalo mínimo en ms expresado en hexadecimal.
- YY YY es el intervalo máximo en ms expresado en hexadecimal.
- ZZ selecciona el tipo *de LE Advertisement* (03 activa el modo notificación no conectable , 00 activaría el modo conectable)
- 07 indica el canal en uso (no modificar).
- Los valores 00 permanecen igual.

Los intervalos de repetición de pulsos estarán comprendidos entre:

Tiempo	Valor de XX XX / YY YY
100 ms	A0 00
500 ms	20 03
1000 ms	40 06
1500 ms	60 09
2000 ms	80 0C
2500 ms	A0 0F
3000 ms	C0 12

Tabla AII-1. Tabla de conversión de intervalos de repetición.

Por tanto, si queremos configurar un intervalo comprendido entre 100 ms y 500 ms, los valores a sustituir serán:

XX XX ⟶ A0 00 **YY YY** ⟶ 20 03

El comando a ejecutar quedaría de la siguiente forma:

sudo hcitool -i hci0 cmd 0x08 0x0006 A0 00 20 03 00 00 00 00 00 00 00 00 00 07 00

Es recomendable configurar el intervalo de forma que permanezca fijo y no oscile entre dos valores. Para ello, bastará con definir el intervalo mínimo y el intervalo máximo con el mismo valor. Para un valor fijo de 500 ms, el comando quedaría de la siguiente forma:

sudo hcitool -i hci0 cmd 0x08 0x0006 20 03 20 03 00 00 00 00 00 00 00 00 00 07 00

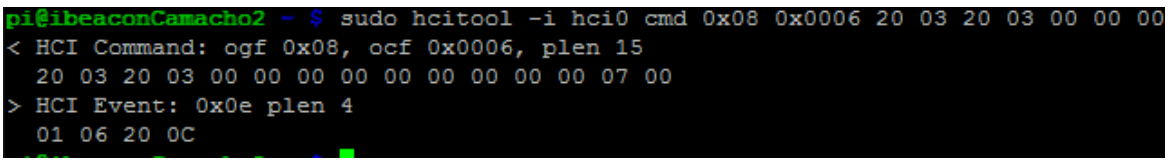


Figura AII-3. Ejemplo de configuración de intervalo de repetición de pulso fijo de 500 ms.

Paso 4. Activación del modo BLE Advertisement *(LE_Set_Advertise_Enable).* Ejecutar:

sudo hcitool -i hci0 cmd 0x08 0x000a XX

Sustituir XX por 01 (ON) o XX por 00 (OFF).

```
pi@ibeaconCamacho3 ~/bluez/bluez-5.18 $ sudo hcitool -i hci0 cmd 0x08 0x000a 01
< HCI Command: ogf 0x08, ocf 0x000a, plen 1
  01
> HCI Event: 0x0e plen 4
  01 0A 20 00
```

Figura AII-4. Ejemplo de activación de transmisión de módulo *Bluetooth*.

Paso 5. Configuración de UUID, Major y minor en la *Payload Data Unit* (*LE_Set_Advertising_Data*). Ejecutar:

5.1 Generar clave UUID. Para este proceso se recomienda utilizar un generador de clave abierto disponible Linux:

sudo uuidgen

5.2 Configuración del módulo con la clave UUID generada. Ejecutar el siguiente comando sustituyendo "X" por los dígitos de la clave en mayúsculas:

sudo hcitool -i hci0 cmd 0x08 0x0008 1E 02 01 1A 1A FF 4C 00 02 15 XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX MM MM mm mm CB 00

5.3 Configuración del *Major* y *minor*. Sustituir en MM MM y mm mm el valor de *Major* y *minor* respectivamente que queramos configurar en valor hexadecimal.

En la siguiente figura se puede observar un ejemplo de configuración con UUID "*57d8922b-e3c0-4df7-9148-83f364783e64*", *Major=1* y *minor=2.*

```
pi@ibeaconCamacho2 ~ $ sudo hcitool -i hci0 cmd 0x08 0x0008 1E 02 01 1A 1A FF 4C
 00 02 15 90 25 85 7E 8D 8E 4A 0F 9A 07 79 B2 22 7E 3C B9 00 01 00 02 CB 00
< HCI Command: ogf 0x08, ocf 0x0008, plen 32
  1E 02 01 1A 1A FF 4C 00 02 15 90 25 85 7E 8D 8E 4A 0F 9A 07
  79 B2 22 7E 3C B9 00 01 00 02 CB 00
> HCI Event: 0x0e plen 4
  01 08 20 00
```

Figura AII-5. Ejemplo de configuración de UUID, Major y minor.

Finalizados los anteriores pasos, el sistema *iBeacon* se encuentra transmitiendo como baliza con la UUID e intervalo de transmisión configurado.

NOTA 1: Es importante no alterar el orden de los pasos y, si se realizan modificaciones en aquellos parámetros configurables, es necesario volver a repetir los pasos posteriores para una correcta configuración.

NOTA 2: En el paso 2 existe la posibilidad de ejecutar el comando *sudo hciconfig hci0 leadv* que sirve para activar la transmisión del módulo *Bluetooth* con un intervalo de repetición de pulsos por defecto de 1280 ms, siendo innecesaria la ejecución de los pasos 3 y 4.

ANEXO III. MANUAL DE CONFIGURACIÓN DE *IBEACON*: MODO ESCÁNER BLE

En el presente anexo se explican los pasos a seguir para la configuración de RPi como escáner BLE.

Paso 1: Creación de *script*.

Para llevar a cabo este primer paso, se ejecuta desde el terminal de comandos de Debian el comando *sudo nano SCAN*, que lanzará el editor de texto *nano* y creará un nuevo archivo. Una vez abierta la ventana del editor, se introducirán las siguientes líneas de código *bash:*

```
#!/bin/bash
# ESCANER IBEACON DE RADIUS NETWORK MEJORADO POR IONE CAMACHO
if [[ $1 == "parse" ]]; then
  packet=""
  capturing=""
  count=0
  while read line
  do
    count=$[count + 1]
    if [ "$capturing" ]; then
      if [[ $line =~ ^[0-9a-fA-F]{2}\ [0-9a-fA-F] ]]; then
        packet="$packet $line"
      else #CAPTURA DE DATOS DEL HCIDUMP PARA OBTENER VALOR DE UUID MAJOR MINOR Y RSSI
        if [[ $packet =~ ^04\ 3E\ 2A\ 02\ 01\ .{26}\ 02\ 01\ .{14}\ 02\ 15 ]]; then
          UUID=`echo $packet | sed 's/^.\{69\}\(.\{47\}\).*$/\1/'`
          MAJOR=`echo $packet | sed 's/^.\{117\}\(.\{5\}\).*$/\1/'`
          MINOR=`echo $packet | sed 's/^.\{123\}\(.\{5\}\).*$/\1/'`
          RSSI=`echo $packet | sed 's/^.\{132\}\(.\{2\}\).*$/\1/'`
          UUID=`echo $UUID | sed -e 's/\ //g' -e
's/^\(.\{8\}\)\(.\{4\}\)\(.\{4\}\)\(.\{4\}\)\(.\{12\}\)$/\1-\2-\3-\4-\5/'`
          MAJOR=`echo $MAJOR | sed 's/\ //g'`
          MAJOR=`echo "ibase=16; $MAJOR" | bc`
          MINOR=`echo $MINOR | sed 's/\ //g'`
          MINOR=`echo "ibase=16; $MINOR" | bc`
          RSSI=`echo "ibase=16; $RSSI" | bc`
          RSSI=$[RSSI - 256]
          if [[ $2 == "-b" ]]; then
            echo "$UUID $MAJOR $MINOR $RSSI"
          else # MUESTRA DE RESULTADOS DEL ESCANEO
            echo "UUID: $UUID MAJOR: $MAJOR MINOR: $MINOR RSSI: $RSSI"
            date +"%T"
          fi
        fi
        capturing=""
        packet=""
        sleep 1
       sudo hcidump --raw  | ./$0 parse $1
      fi
    fi
    if [ ! "$capturing" ]; then
```

```
        if [[ $line =~ ^\> ]]; then
          packet=`echo $line | sed 's/^>.\(.*$\)/\1/'`
          capturing=1
        fi
      fi
    done
  else #INICIO DE ESCANEO Y OBTENCION DE DATOS CON HCIDUMP
    sudo hcitool lescan -- passive --duplicates 1>/dev/null &
    if [ "$(pidof hcitool)" ]; then
      sudo hcidump --raw | ./$0 parse $1
    fi
  fi
```

Una vez introducido el código, pulsar ctrl+O para guardar el código con el nombre *SCAN* y posteriormente pulsar ctrl+X para salir del editor.

Paso 2: Hacer el *script* ejecutable.

Creado el *script SCAN*, el siguiente paso es configurarlo con permisos para ejecución por el sistema. Para ello, se ejecutará el comando *sudo chmod +x SCAN* que es el encargado de establecer los permisos.

Para comprobar que el *script* es ejecutable, se ejecutará el comando *ls* y se comprobará que el nombre *SCAN* aparece en color verde.

Paso 3: Ejecución del *script*.

Para ejecutar el *script SCAN*, basta con introducir en la línea de comandos del terminal lo siguiente:

./SCAN

Cabe destacar que la ejecución del *script* implica la puesta en funcionamiento del RPi como escáner y la presentación en pantalla de los datos recibidos sin necesidad de ejecutar ninguna otra acción.

Paso 4: Modificación del *script* para la implementación del sistema de medición de nivel de batería.

En este paso, se modifican, con el editor de texto *nano*, las líneas de código del *script SCAN* para incluir nuevas líneas del sistema de medición de batería, quedando el código de la siguiente forma:

```
#!/bin/bash
# ESCANER IBEACON CON INDICADOR DE NIVEL DE BATERIA POR IONE CAMACHO
if [[ $1 == "parse" ]]; then
  packet=""
  capturing=""
  count=0
  while read line
  do
    count=$[count + 1]
    if [ "$capturing" ]; then
      if [[ $line =~ ^[0-9a-fA-F]{2}\ [0-9a-fA-F] ]]; then
        packet="$packet $line"
      else #CAPTURA DE DATOS DEL HCIDUMP PARA OBTENER VALOR DE UUID MAJOR MINOR RSSI Y TXPOWER
        if [[ $packet =~ ^04\ 3E\ 2A\ 02\ 01\ .{26}\ 02\ 01\ .{14}\ 02\ 15 ]]; then
          UUID=`echo $packet | sed 's/^.\{69\}\(.\{47\}\).*$/\1/'`
          MAJOR=`echo $packet | sed 's/^.\{117\}\(.\{5\}\).*$/\1/'`
          MINOR=`echo $packet | sed 's/^.\{123\}\(.\{5\}\).*$/\1/'`
          RSSI=`echo $packet | sed 's/^.\{132\}\(.\{2\}\).*$/\1/'`
          TXPOWER=`echo $packet | sed 's/^.\{129\}\(.\{2\}\).*$/\1/'`
          UUID=`echo $UUID | sed -e 's/\ //g' -e
's/^\(.\{8\}\)\(.\{4\}\)\(.\{4\}\)\(.\{4\}\)\(.\{12\}\)$/\1-\2-\3-\4-\5/'`
          MAJOR=`echo $MAJOR | sed 's/\ //g'`
          MAJOR=`echo "ibase=16; $MAJOR" | bc`
          MINOR=`echo $MINOR | sed 's/\ //g'`
          MINOR=`echo "ibase=16; $MINOR" | bc`
```

```
            RSSI=`echo "ibase=16; $RSSI" | bc`
            RSSI=$[RSSI - 256]
            TXPOWER=`echo "ibase=16; $TXPOWER" | bc`
            TXPOWER=$[TXPOWER - 256]
            if [[ $2 == "-b" ]]; then
              echo "$UUID $MAJOR $MINOR $RSSI $TXPOWER"
            else # COMPROBADOR DE ALERTAS DE NIVEL DE BATERIA
              if [[ $TXPOWER == -1 ]]; then
              echo "$(tput setaf 1)$(tput setab 7)UUID: $UUID MAJOR: $MAJOR MINOR: $MINOR RSSI: $RSSI
TXPOWER: BATERIA BAJA$(tput sgr 0)"
              date +"%T"
              fi
              if [[ $TXPOWER == -20 ]]; then
              echo "UUID: $UUID MAJOR: $MAJOR MINOR: $MINOR RSSI: $RSSI TXPOWER: BATERIA MEDIA"
              date +"%T"
              fi
              if [[ $TXPOWER == -56 ]]; then
              echo "UUID: $UUID MAJOR: $MAJOR MINOR: $MINOR RSSI: $RSSI TXPOWER: BATERIA ALTA"
              date +"%T"
              fi
            fi
          fi
          capturing=""
          packet=""
          sleep 2
          sudo hcidump --raw  | ./$0 parse $1
        fi
      fi
```

ANEXO IV. MANUAL DE CONFIGURACIÓN DE *iBEACON*: SISTEMA DE ENVÍO DE ALARMAS Y ÓRDENES

En el presente anexo, se explican los pasos a seguir para la configuración de *iBeacon* en modo envío de alarmar y órdenes.

Paso 1: Creación de *script*.

Para llevar a cabo este primer paso, se ejecuta desde el terminal de comandos de Debian el comando *sudo nano SAO*, que lanzará el editor de texto *nano* y creará un nuevo archivo. Una vez abierta la ventana del editor, se introducirán las siguientes líneas de código *bash:*

```
#!/bin/bash
# SISTEMA DE ALARMA Y ORDENES PARA IBEACON POR IONE CAMACHO

if [[ $1 == "parse" ]]; then
  packet=""
  capturing=""
  count=0
  while read line
  do
    count=$[count + 1]
    if [ "$capturing" ]; then
      if [[ $line =~ ^[0-9a-fA-F]{2}\ [0-9a-fA-F] ]]; then
        packet="$packet $line"
      else # CAPTURA DE DATOS DE HCIDUMP PARA OBTENER VALORES DE MAJOR Y MINOR
        if [[ $packet =~ ^04\ 3E\ 2A\ 02\ 01\ .{26}\ 02\ 01\ .{14}\ 02\ 15 ]]; then
          UUID=`echo $packet | sed 's/^.\{69\}\(.\{47\}\).*$/\1/'`
          MAJOR=`echo $packet | sed 's/^.\{117\}\(.\{5\}\).*$/\1/'`
          MINOR=`echo $packet | sed 's/^.\{123\}\(.\{5\}\).*$/\1/'`
          UUID=`echo $UUID | sed -e 's/\ //g' -e
's/^\(.\{8\}\)\(.\{4\}\)\(.\{4\}\)\(.\{4\}\)\(.\{12\}\)$/\1-\2-\3-\4-\5/'`
          MAJOR=`echo $MAJOR | sed 's/\ //g'`
          MAJOR=`echo "ibase=16; $MAJOR" | bc`
          MINOR=`echo $MINOR | sed 's/\ //g'`
          MINOR=`echo "ibase=16; $MINOR" | bc`
          if [[ $2 == "-b" ]]; then
            echo "$UUID $MAJOR $MINOR " # COMPROBACIÓN DE VALORES DE MAJOR Y MINOR
          else
           if [[ $UUID == 0308ED62-CAB8-440D-BF27-93643486EFC1 ]]; then # INICIADOR DEL SISTEMA DE
ALARMAS
            if [[ $MAJOR == 10 ]]; then # ACTIVADOR DE ALARMAS GENERALES(MAJOR 10)
             if [[ $MINOR == 1 ]]; then # ALERTA ZAFARRANCHO DE COMBATE(MINOR 1)

#CAMPO PARA INTRODUCIR BASE DE DATOS DE ALARMAS
                  echo "$(tput setaf 1)$(tput setab 7)ZAFARRANCHO DE COMBATE$(tput sgr 0)"
                  aplay /home/ZC.wav #REPRODUCCION DE ALARMA
                  clear
             fi
             if [[ $MINOR == 2 ]]; then #ID ALARMA ABANDONO DE BUQUE (MINOR 2)

                  echo "$(tput setaf 1)$(tput setab 7)ABANDONO DE BUQUE$(tput sgr 0)"
                  aplay /home/AB.wav #REPRODUCCIÓN DE ALARMA
```

```
                clear
              fi
             fi
            fi
            if [[ $UUID == 5AFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFFF ]]; then # INICIADOR DEL SISTEMA DE
ORDENES.EL UUID A INTRODUCIR EN ESTE CAMPO CORRESPONDE CON EL DEL TX DEL IBEACON
            #CAMPO PARA INTRODUCIR BASE DE DATOS DE ORDENES
             if [[ $MAJOR == 2 ]]; then # ACTIVADOR DE ORDENES GENERALES (MAJOR 2)
              if [[ $MINOR == 1 ]]; then # ID ALERTA ACUDA A PUENTE (MINOR 1)

                echo "$(tput setaf 1)$(tput setab 7)ACUDA A PUENTE$(tput sgr 0)"
                aplay /home/AP.wav #REPRODUCCION DE ORDEN ACUDA A PUENTE
                clear
              fi
             fi
            fi
            fi
        fi
        capturing=""
        packet=""
        sudo hcidump --raw  | ./$0 parse $1
      fi
    fi
    if [ ! "$capturing" ]; then
      if [[ $line =~ ^\> ]]; then
        packet=`echo $line | sed 's/^>.\(.*$\)/\1/'`
        capturing=1
      fi
    fi
  done
else #INICIO DE TX MODO IBEACON Y ESCANEO PASIVO DE SEÑALES DE ALARMA
  sudo hciconfig hci0 up
  sudo hcitool lescan --passive --duplicates 1>/dev/null &
  echo "Iniciando TX ..."
  sudo hcitool -i hci0 cmd 0x08 0x0006 60 09 60 09 00 00 00 00 00 00 00 00 00 07 00
  sudo hcitool -i hci0 cmd 0x08 0x000a 01
  sudo hcitool -i hci0 cmd 0x08 0x0008 1E 02 01 1A 1A FF 4C 00 02 15 54 FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF 00 01 00 02 C8 00
  echo "Transmitiendo Major 1 Minor 2"
  sleep 1
  sudo amixer cset numid=3 1 #CONFIGURACIÓN INICIAL DE AUDIO
  clear

  if [ "$(pidof hcitool)" ]; then
    sudo hcidump --raw | ./$0 parse $1
  fi
fi
```

Una vez introducido el código, pulsar ctrl+O para guardar el código con el nombre *SAO* y posteriormente pulsar ctrl+X para salir del editor.

Paso 2: Hacer el *script* ejecutable.

Creado el *script SAO*, el siguiente paso es configurarlo con permisos para ejecución por el sistema. Para ello, se ejecutará el comando *sudo chmod +x SAO* que es el encargado de establecer los permisos.

Para comprobar que el *script* es ejecutable, se ejecutará el comando *ls* y se comprobará que el nombre *SAO* aparece en color verde.

Paso 3: Configuración de RPi para reproducción de audio.

En este paso, se explica cómo descargar e instalar el controlador de sonido ALSA para Debian y cómo configurar RPi para reproducir audio.

El RPi B+ tiene incorporada una tarjeta de sonido, por lo que solo es necesario instalar un controlador que nos permita gestionarla. El controlador elegido, tras algunas pruebas y errores con *pulseaudio*, ha sido ALSA. Para descargar e instalar dicho controlador, ejecutar en el terminal lo siguiente:

sudo apt-get update
sudo apt-get dist-upgrade

sudo apt-get install esound esound-clients esound-common alsa-base alsa-tools alsa-utils alsa-oss linux-sound-base python-alsaaudio gnome-media libsdl1.2debian-alsa

Una vez instalado ALSA, ejecutar *sudo reboot* para reiniciar el RPi.

La salida de audio de RPi puede configurarse tanto por puerto HDMI como por el conector *jack* de 3.5 mm del que dispone el dispositivo. En el dispositivo de alarma se configurará la salida por el conector *jack* de 3.5 mm con el comando:

sudo amixer cset numid=3 1

No es necesario ejecutar este comando en este paso, ya que ha sido introducido en el *script* descrito en el paso 1.

Para la configuración de los parámetros de sonido de RPi, ejecutar el comando *sudo alsamixer* que abrirá una ventana de configuración donde se pueden seleccionar parámetros como: tarjeta de sonido, volumen, modo de reproducción, etc. A continuación, se muestra una captura del configurador *alsamixer*.

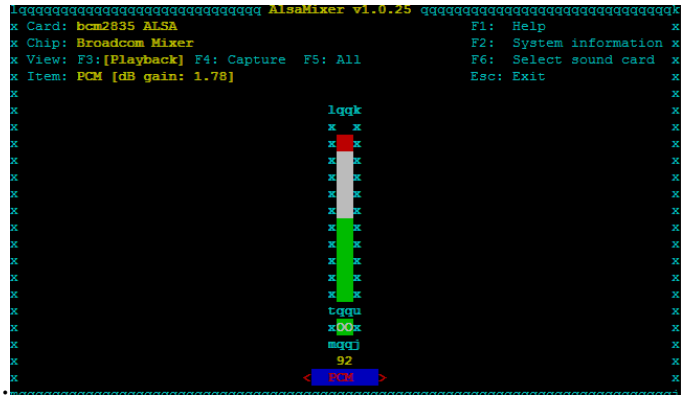


Figura AIV-1. Configuración de audio con interfaz de *alsamixer*.

La reproducción de audio es llevada a cabo por la aplicación *aplay* incluida en el paquete ALSA descargado. Este reproductor por defecto reproduce archivos en formato *wav* con 16 bits de resolución, una tasa de muestreo de 48000 Hz y en canal e*stéreo*. Es importante mantener este formato para la correcta reproducción de archivos de audio.

Una vez realizada la configuración, conectar un sistema de audio (altavoz portátil, auriculares, etc.) al conector *jack* y el dispositivo se encuentra preparado para ejecutar el *script* y emitir la señal sonora de alarma.

Paso 4: Ejecución del *script*.

Para ejecutar el *script SAO* basta con introducir en la línea de comandos del terminal lo siguiente:

./SAO

Tras su ejecución, el *iBeacon* se encuentra transmitiendo su identificador en modo LEAd y listo para ejecutar una alerta enviada por un dispositivo maestro.

ANEXO V: MANUAL DE CONFIGURACIÓN DE *iBEACON*: SISTEMA DE AVISO DE NIVEL DE BATERÍA

En el presente anexo, se explican los pasos a seguir para configurar el sistema *iBeacon* con la funcionalidad de aviso de nivel de batería.

Paso 0: Conexión e instalación de placa de control de voltaje *MoPi*.

Este paso se ha explicado en el Capítulo 5.

Paso 1: Creación de *script* ejecutable que permita las siguientes funcionalidades:

1. Configurar el *iBeacon* para transmitir paquete LEAd.
2. Verificación periódica de nivel de batería.
3. Modificar valor de *Tx Power* según voltaje remanente en la fuente de alimentación.

Para ello, ejecutar el comando *sudo nano iBeaconTXSANB*. En el editor de texto introducir las siguientes líneas de código.

```
#! /bin/bash
# SISTEMA DE AVISO DE NIVEL DE BATERIA POR IONE CAMACHO

# CONFIGURACIÃ“N INICIAL DE IBEACON
Iniciador=0
sudo hciconfig hci0 up
Iniciador=$[Iniciador + 1]
echo "Transmitiendo M 1 m 1"

# INICIO DEL BUCLE COMPROBADOR DEL NIVEL DE VOLTAJE CON PLACA MOPI
while [ $Iniciador -eq 1 ] ; do
        medida=`sudo mopicli -v | cut -d " " -f4`
         if [ $medida -gt 8500 ] ; then # ARRANQUE INICIADOR DE ALERTA DE BATERIA BAJA TX UUID MAJOR 1
MINOR 1 Y TX POWER -1(INDICADOR)
                sudo hcitool -i hci0 cmd 0x08 0x000a 00
                sudo hcitool -i hci0 cmd 0x08 0x000a 01
                sudo hcitool -i hci0 cmd 0x08 0x0006 60 09 60 09 00 00 00 00 00 00 00 00 00 07 00
                sudo hcitool -i hci0 cmd 0x08 0x0008 1E 02 01 1A 1A FF 4C 00 02 15 90 25 85 7E 8D 8E 4A
0F 9A 07 79 B2 22 7E 3C B9 00 01 00 01 C8 00
                echo "BaterÃa alta $medida"
                sleep 20
         fi
         if [ $medida -le 8500 ] ; then
              if [ $medida -le 7000 ] ; then # ARRANQUE INICIADOR DE ALERTA DE BATERIA BAJA TX UUID
MAJOR 1 MINOR 1 Y TX POWER -1(INDICADOR)
                sudo hcitool -i hci0 cmd 0x08 0x000a 00
                sudo hcitool -i hci0 cmd 0x08 0x000a 01
                sudo hcitool -i hci0 cmd 0x08 0x0006 60 09 60 09 00 00 00 00 00 00 00 00 00 07 00
```

```
                sudo hcitool -i hci0 cmd 0x08 0x0008 1E 02 01 1A 1A FF 4C 00 02 15 90 25 85 7E 8D 8E 4A
0F 9A 07 79 B2 22 7E 3C B9 00 01 00 01 FF 00
                echo "BaterÃa baja $medida"
                sleep 10
              fi
              if [ $medida -gt 7000 ] ; then # ARRANQUE INICIADOR DE ALERTA DE BATERIA MEDIA TX UUID
MAJOR 1 MINOR 1 Y TX POWER -1(INDICADOR)
                sudo hcitool -i hci0 cmd 0x08 0x000a 00
                sudo hcitool -i hci0 cmd 0x08 0x000a 01
                sudo hcitool -i hci0 cmd 0x08 0x0006 60 09 60 09 00 00 00 00 00 00 00 00 00 07 00
                sudo hcitool -i hci0 cmd 0x08 0x0008 1E 02 01 1A 1A FF 4C 00 02 15 90 25 85 7E 8D 8E 4A
0F 9A 07 79 B2 22 7E 3C B9 00 01 00 01 EC 00
                echo "BaterÃa media $medida"
                sleep 20
              fi
        fi
done
```

Una vez introducido el código, pulsar ctrl+O para guardar el código con el nombre *iBeaconTXSANB* y posteriormente pulsar ctrl+X para salir del editor.

Paso 2: Hacer el *script* ejecutable.

Creado el *script iBeaconTXSANB*, el siguiente paso es configurarlo con permisos para ejecución por el sistema. Para ello se ejecutará el comando *sudo chmod +x iBeaconTXSANB.*

Para comprobar que el *script* es ejecutable, se ejecutará el comando *ls* y se comprobará que el nombre *iBeaconTXSANB* aparece en color verde.

Paso 3: Ejecución del *script*.

Para ejecutar el *script iBeaconTXSANB*, basta con introducir en la línea de comandos del terminal lo siguiente:

./iBeaconTXSANB

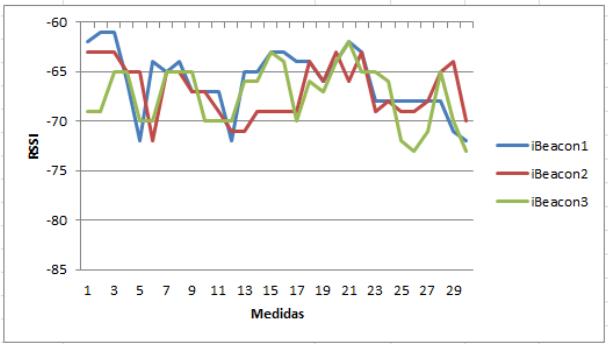
Tras su ejecución, el *iBeacon* se encuentra transmitiendo su identificador en modo LEAd con un valor de *Tx Power* configurado en función del nivel de batería remanente.

ANEXO VI: GRÁFICAS Y RESULTADOS DE MEDICIONES DE RSSI

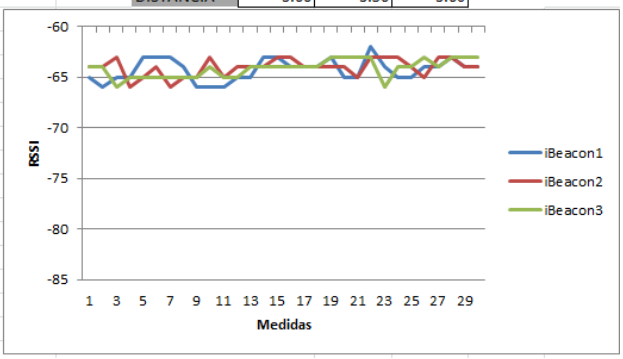
En el presente anexo, se muestran las mediciones y gráficas obtenidas tras la medición de RSSI para diferentes valores de PRI.

- **Pruebas 3 *iBeacon* a 5 metros en hall Cuartel Marqués de la Victoria.**

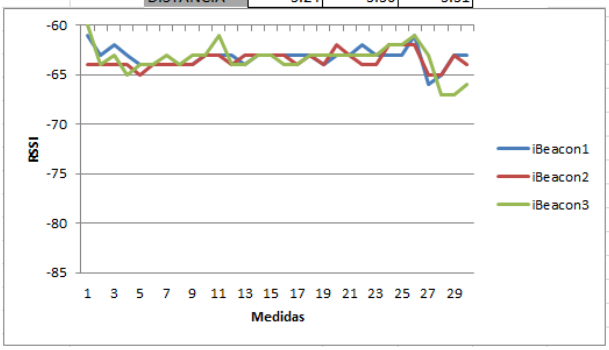
PRI 100 ms	iBeacon 1	iBeacon 2	iBeacon 3
MEDIA	-65.93	-66.87	-67.37
DES.TÍPICA	3.12	2.70	3.03
MEDIANA	-65.50	-67.00	-66.00
DISTANCIA	4.43	4.94	5.23



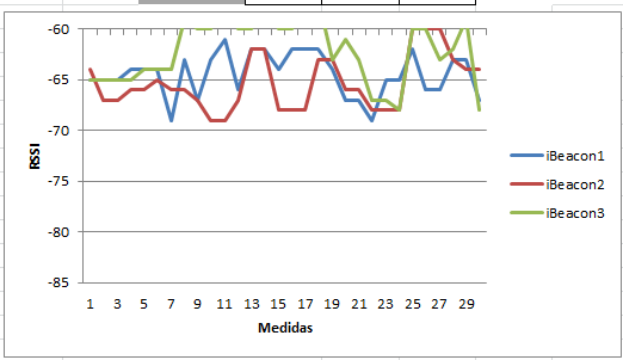
PRI 500 ms	iBeacon 1	iBeacon 2	iBeacon 3
MEDIA	-64.27	-64.03	-64.13
DES.TÍPICA	1.06	0.87	0.88
MEDIANA	-64.00	-64.00	-64.00
DISTANCIA	3.66	3.56	3.60



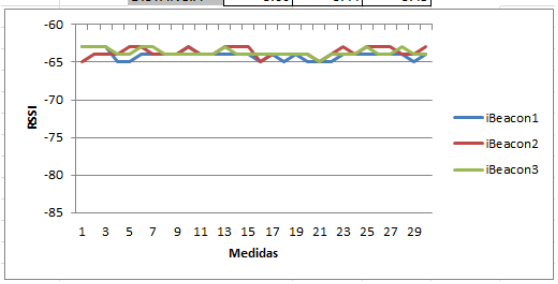
PRI 1000 ms	iBeacon 1	iBeacon 2	iBeacon 3
MEDIA	-63.20	-63.53	-63.40
DES.TÍPICA	0.98	0.85	1.50
MEDIANA	-63.00	-64.00	-63.00
DISTANCIA	3.24	3.36	3.31



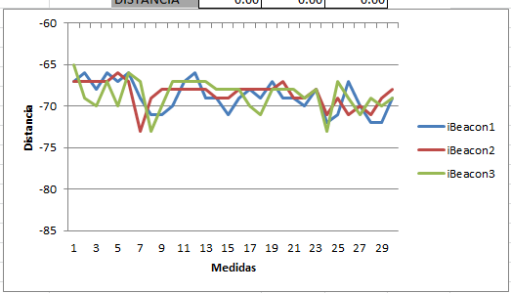
PRI 1280 ms	iBeacon 1	iBeacon 2	iBeacon 3
MEDIA	-64.47	-65.33	-62.00
DES.TÍPICA	2.11	2.66	3.29
MEDIANA	-64.00	-66.00	-61.50
DISTANCIA	3.74	4.14	2.82



PRI 1500 ms	iBeacon 1	iBeacon 2	iBeacon 3
MEDIA	-64.13	-63.73	-63.77
DES.TÍPICA	0.62	0.63	0.50
MEDIANA	-64.00	-64.00	-64.00
DISTANCIA	3.60	3.44	3.45

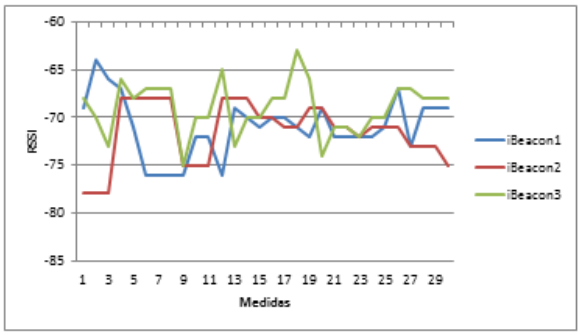


PRI 2000 ms	iBeacon 1	iBeacon 2	iBeacon 3
MEDIA	-68.83	-68.50	-68.63
DES.TÍPICA	1.86	1.48	1.83
MEDIANA	-69.00	-68.00	-68.00
DISTANCIA	0.00	0.00	0.00

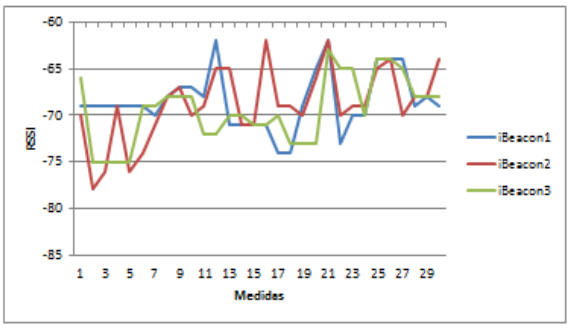


- **Pruebas 3 *iBeacon* a 10 metros en hall Cuartel Marqués de la Victoria.**

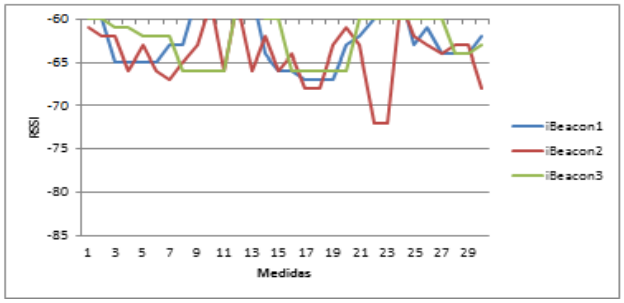
PRI 100 ms	iBeacon 1	iBeacon 2	iBeacon 3
MEDIA	-70.97	-71.47	-69.00
DES.TÍPICA	2.99	3.16	2.67
MEDIANA	-71.00	-71.00	-68.00
DISTANCIA	7.91	8.38	6.31



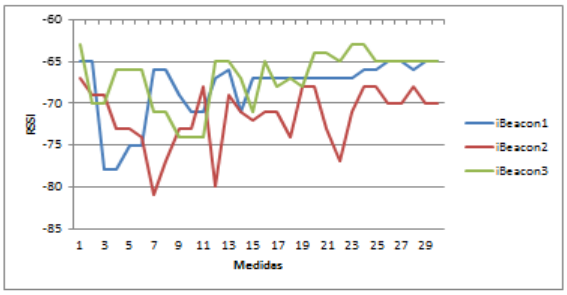
PRI 500 ms	iBeacon 1	iBeacon 2	iBeacon 3
MEDIA	-68.50	-68.83	-69.43
DES.TÍPICA	3.04	3.80	3.49
MEDIANA	-69.00	-69.00	-69.50
DISTANCIA	5.96	6.19	6.63



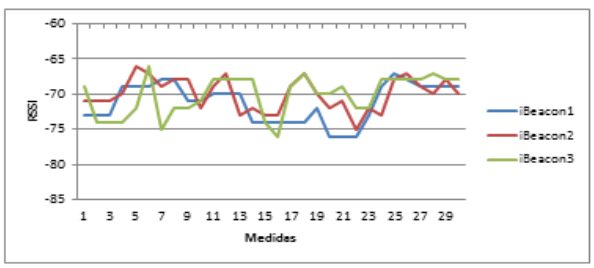
PRI 1000 ms	iBeacon 1	iBeacon 2	iBeacon 3
MEDIA	-62.23	-64.10	-62.30
DES.TÍPICA	3.32	3.44	2.76
MEDIANA	-63.00	-63.00	-61.50
DISTANCIA	2.90	3.59	2.92



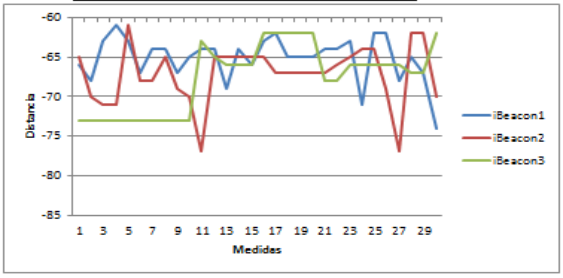
PRI 1280 ms	iBeacon 1	iBeacon 2	iBeacon 3
MEDIA	-68.13	-71.53	-67.00
DES.TÍPICA	3.71	3.51	3.29
MEDIANA	-67.00	-71.00	-65.50
DISTANCIA	0.00	0.00	0.00



PRI 1500 ms	iBeacon 1	iBeacon 2	iBeacon 3
MEDIA	-71.23	-70.03	-70.17
DES.TÍPICA	2.67	2.26	2.72
MEDIANA	-70.50	-70.00	-69.00
DISTANCIA	0.00	0.00	0.00

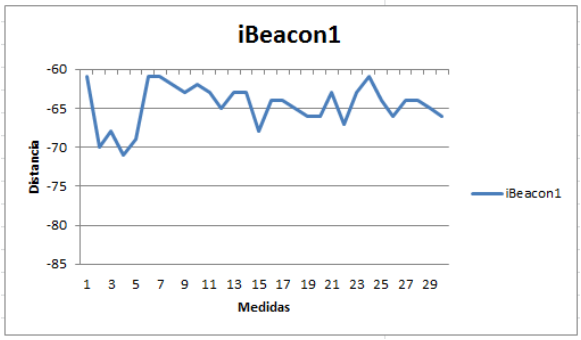


PRI 2000 ms	iBeacon 1	iBeacon 2	iBeacon 3
MEDIA	-65.17	-67.13	-67.60
DES.TÍPICA	2.77	3.68	4.20
MEDIANA	-64.50	-67.00	-66.00
DISTANCIA	4.06	5.09	5.37

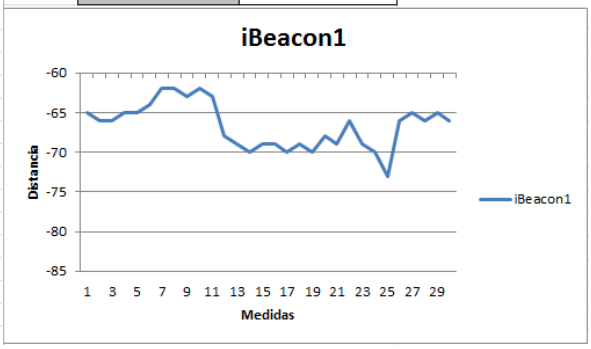


- **Pruebas 1 *iBeacon* a 5 metros en hall Cuartel Marqués de la Victoria.**

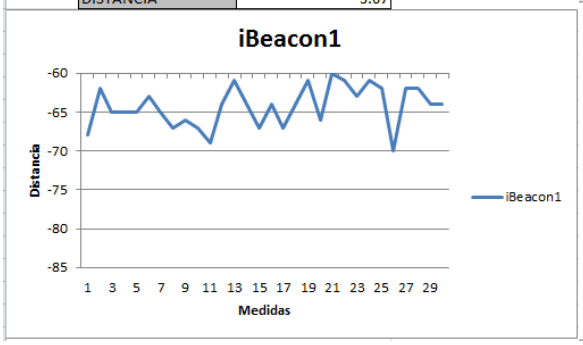
PRI 100 ms	iBeacon 1
MEDIA	-64.60
DES.TÍPICA	2.64
MEDIANA	-64.00
DISTANCIA	3.80



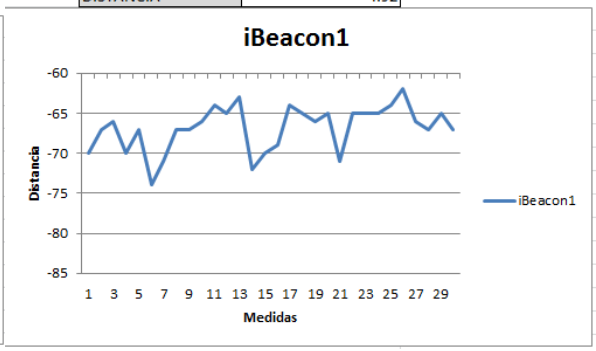
PRI 500 ms	iBeacon 1
MEDIA	-66.67
DES.TÍPICA	2.81
MEDIANA	-66.00
DISTANCIA	4.82



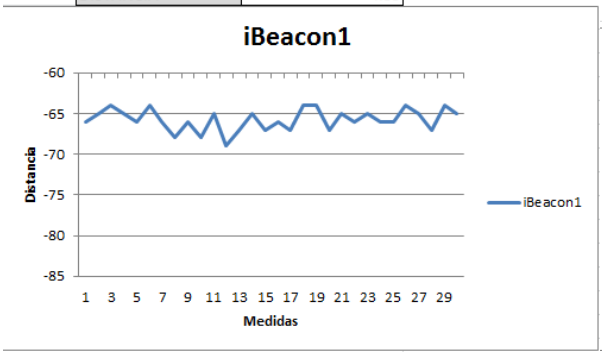
PRI 1000 ms	iBeacon 1
MEDIA	-64.30
DES.TÍPICA	2.53
MEDIANA	-64.00
DISTANCIA	3.67



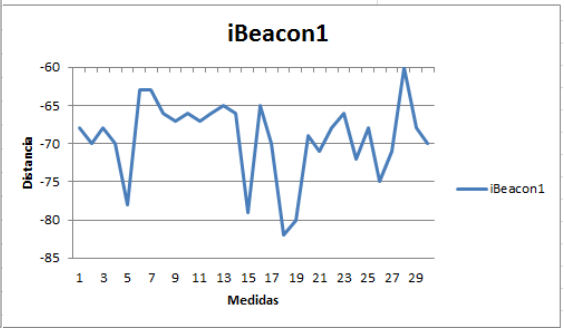
PRI 1280 ms	iBeacon 1
MEDIA	-66.83
DES.TÍPICA	2.81
MEDIANA	-66.00
DISTANCIA	4.92



PRI 1500 ms	iBeacon 1
MEDIA	-65.73
DES.TÍPICA	1.31
MEDIANA	-66.00
DISTANCIA	4.33

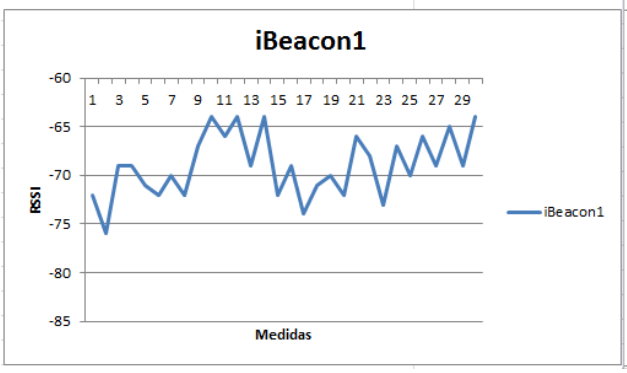


PRI 2000 ms	iBeacon 1
MEDIA	-69.23
DES.TÍPICA	5.06
MEDIANA	-68.00
DISTANCIA	6.48

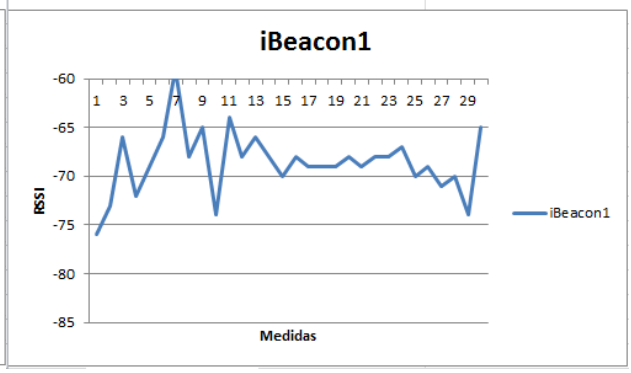


- **Pruebas 1 *iBeacon* a 10 metros en hall Cuartel Marqués de la Victoria.**

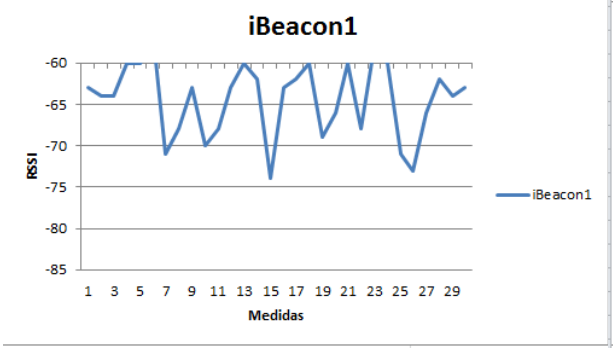
PRI 100 ms	iBeacon 1
MEDIA	-69.00
DES.TÍPICA	3.15
MEDIANA	-69.00
DISTANCIA	6.31



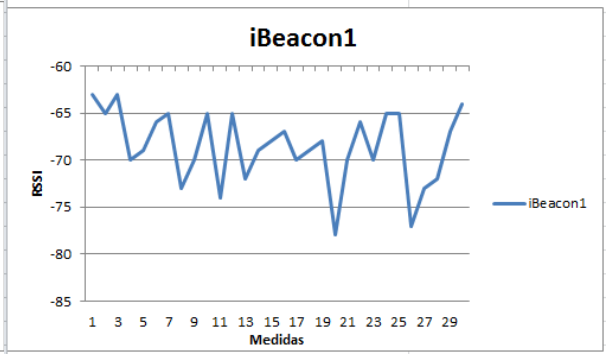
PRI 500 ms	iBeacon 1
MEDIA	-68.60
DES.TÍPICA	3.27
MEDIANA	-68.50
DISTANCIA	6.03



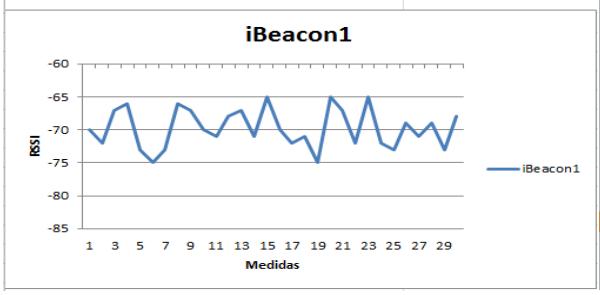
PRI 1000 ms	iBeacon 1
MEDIA	-64.40
DES.TÍPICA	4.43
MEDIANA	-63.00
DISTANCIA	3.72



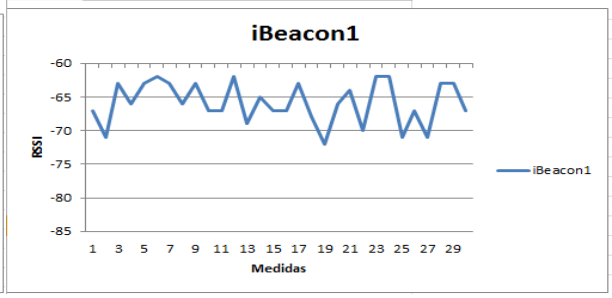
PRI 1280 ms	iBeacon 1
MEDIA	-68.60
DES.TÍPICA	3.83
MEDIANA	-68.50
DISTANCIA	6.03



PRI 1500 ms	iBeacon 1
MEDIA	-69.77
DES.TÍPICA	2.93
MEDIANA	-70.00
DISTANCIA	6.89

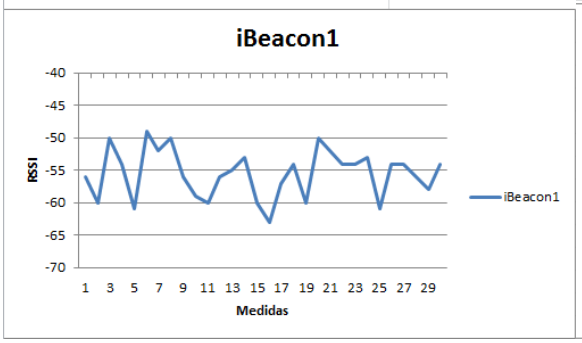


	iBeacon 1
MEDIA	-65.90
DES.TÍPICA	3.05
MEDIANA	-66.00
DISTANCIA	4.42

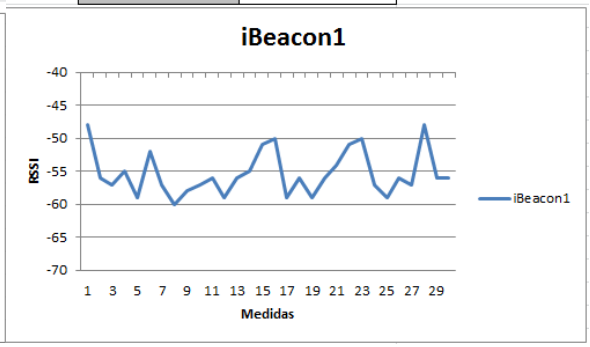


- **Análisis de RSSI en el patrullero "Tabarca".**

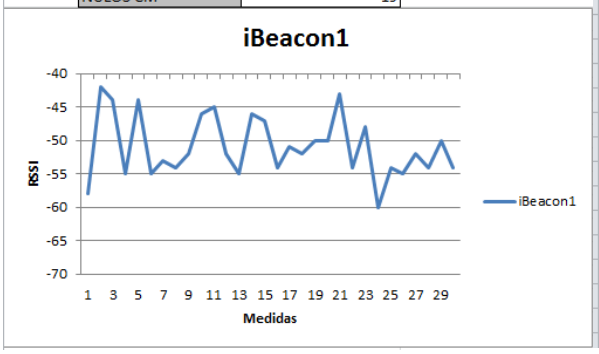
PRI 100 ms	iBeacon 1
MEDIA	-55.50
DES.TÍPICA	3.68
MEDIANA	-54.50
DISTANCIA	1.33
NULOS CM	19



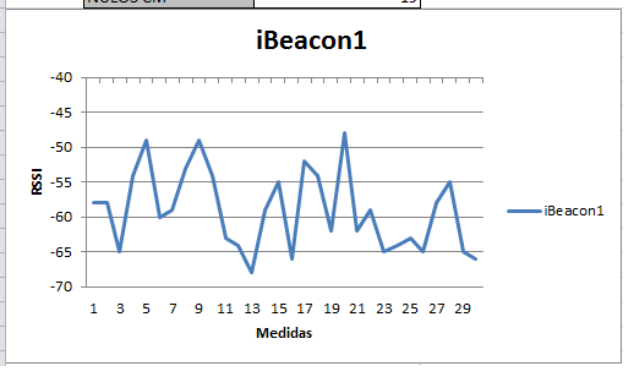
PRI 500 ms	iBeacon 1
MEDIA	-55.33
DES.TÍPICA	3.31
MEDIANA	-56.00
DISTANCIA	1.31
NULOS CM	11



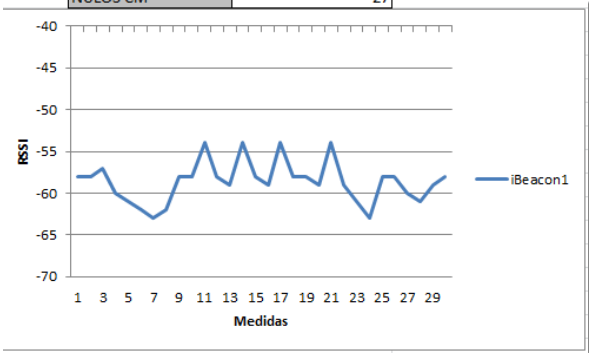
PRI 1000 ms	iBeacon 1
MEDIA	-50.97
DES.TÍPICA	4.52
MEDIANA	-52.00
DISTANCIA	0.79
NULOS CM	19



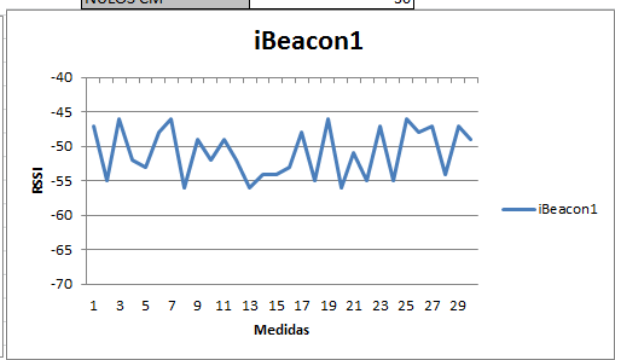
PRI 1280 ms	iBeacon 1
MEDIA	-59.07
DES.TÍPICA	5.60
MEDIANA	-59.00
DISTANCIA	2.01
NULOS CM	19



PRI 1500 ms	iBeacon 1
MEDIA	-58.63
DES.TÍPICA	2.41
MEDIANA	-58.00
DISTANCIA	1.91
NULOS CM	27



PRI 2000 ms	iBeacon 1
MEDIA	-50.87
DES.TÍPICA	3.56
MEDIANA	-51.50
DISTANCIA	0.78
NULOS CM	30



ANEXO VII: PRUEBAS DE CONSUMO ENERGÉTICO

En el presente anexo, se muestran los resultados de las pruebas de consumo energético realizadas al sistema *iBeacon* con el fin de determinar la autonomía del mismo en los diferentes modos de funcionamiento.

Para la obtención de los datos de consumo energético, se instaló en uno de los RPi una placa *MoPi 24/7* con las características explicadas en el Apartado 5.3.1. En la Figura AVII-1 se muestra uno de los RPi con la placa *MoPi* conectada y alimentada con dos baterías alcalinas no recargables de 9V.

Figura AVII-1. Placa MoPi alimentando a RPi.

Se realizó la configuración de la misma con los valores de voltaje que permiten operar con baterías alcalinas no recargables. Para ello, se ejecutó el comando *sudo mopicli –wc 2 18000 7000 6500 6200*, que establece el valor de voltaje máximo soportado por la *MoPi* en 18V, el valor normal de trabajo en 7V y un valor bajo de 6.5V en el que la placa indicará mediante *led* naranja que el nivel de batería es bajo. Por último, un valor crítico de 6.2V en el que la *MoPi* iniciará la secuencia de apagado del RPi de forma segura para evitar daños en el mismo. En la Figura AVII-2 se muestra la configuración de la *MoPi* establecida.

```
pi@iBeacon1 ~ $ sudo mopicli -rc1
Source #1 conf:
  Source type: Non-rechargeable batteries
  Maximum voltage: 18000 mV
  Good voltage: 7000 mV
  Low voltage: 6500 mV
  Critical voltage: 6200 mV
```

Figura AVII-2. Configuración de voltajes de la fuente de alimentación en MoPi.

Una vez realizada la instalación y configuración de la *MoPi*, se realizaron tres tipos de experimentos según el modo de operación del RPi:

- **Modo normal**: En este experimento, se analizó el consumo del RPi + *MoPi* sin realizar ningún tipo de acción con el fin de definir la energía consumida por el simple hecho de estar encendido.
- **Modo *iBeacon*:** En este experimento, se analizó el consumo del RPi + *MoPi* emitiendo un paquete de datos LEAd para localización.
- **Modo *iBeacon* + alarma*:*** En este experimento, se analizó el consumo del RPi + *MoPi* emitiendo paquetes LEAd y realizando escáner pasivo de alertas.

En cada uno de los experimentos se realizaron medidas del voltaje remanente en las fuentes de alimentación cada 5 minutos durante un periodo de 55 minutos (12 medidas). Para ello se ejecutó el comando *sudo mopicli –v* que presenta en pantalla el voltaje registrado a la entrada de la *MoPi*, obteniéndose los resultados de la Tabla AVII-1.

	Normal (mV)	iBeacon (mV)	Alarma (mV)
H	9360	9300	9320
H+ 5min	9009	8940	8890
H +10 min	8697	8609	8385
H +15 min	8380	8226	7878
H +20 min	7995	7860	7527
H +25 min	7605	7527	7293
H +30 min	7449	7332	6942
H +35 min	7254	7176	6825
H +40 min	7059	6903	6652
H +45 min	6903	6825	6496
H +50 min	6786	6647	6390
H +55 min	6651	6480	6280
VOLTAJE TOTAL	2709	2820	3040
DIFERENCIA MODO NORMAL	0	111	331

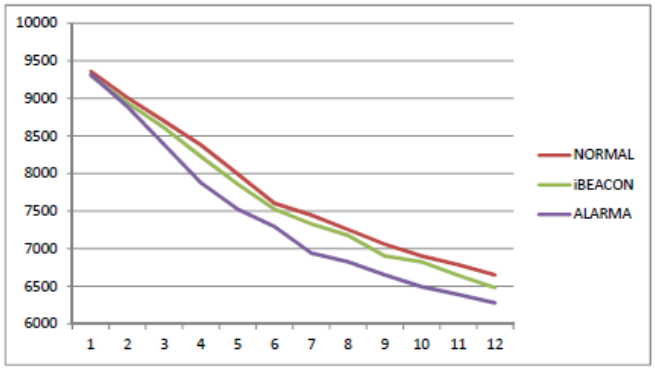


Tabla AVII-1. Mediciones de voltaje con *MoPi*.

En la tabla se puede apreciar la diferencia de voltaje total en los tres modos de funcionamiento para un tiempo total de 55 minutos, así como la diferencia entre el modo *iBeacon* y el modo alarma con respecto al funcionamiento normal del RPi. Para calcular el consumo eléctrico de los diferentes modos de funcionamiento, es necesario el cálculo de la potencia eléctrica desarrollada en los 55 minutos del experimento.

Cuando se trata de corriente continua, la potencia eléctrica desarrollada en un cierto instante por un dispositivo de dos terminales, es el producto de la diferencia de potencial entre dichos terminales y la intensidad de corriente que pasa a través del dispositivo. Por esta razón, la potencia es proporcional a la corriente y a la tensión. Esto es:

$$P = \frac{dw}{dt} = \frac{dw}{dq} \cdot \frac{dq}{dt} = V \cdot I \qquad (2)$$

Donde **I** es el valor instantáneo de la intensidad de corriente y **V** es el valor instantáneo del voltaje. Si **I** se expresa en amperios y **V** en voltios, **P** estará expresada en watts (vatios). Igual definición se aplica cuando se consideran valores promedio para **I**, **V** y **P**.

Aplicando lo anterior, y sabiendo que la capacidad ofrecida por las baterías alcalinas según el fabricante es de 680 mAh, se calcularon las potencias eléctricas desarrolladas por cada uno de los modos:

- **Modo normal:** Voltaje total = 2709 mV ; I = 680 mAh * (55min / 60 min) = 623.3 mA

  P = V*I = (2.7 V) * (0.623 A) = 1.68 W en 55 minutos, por tanto, tendremos:

  P_{1h} = 1.68* (60 min/55min) = **1.84 W**

- **Modo *iBeacon*:** Voltaje total = 2820 mV ; I = 680 mAh * (55min / 60 min) = 623.3 mA

  P = V*I = (2.82 V) * (0.623 A) = 1.75 W en 55 minutos, por tanto tendremos:

  P_{1h} = 1.75* (60 min/55min) = **1.92 W**

- **Modo alarma:** Voltaje total = 3040 mV ; I = 680 mAh * (55min / 60 min) = 623.3 mA

  P = V*I = (3.04 V) * (0.623 A) = 1.89 W en 55 minutos, por tanto tendremos:

  P_{1h} = 1.89* (60 min/55min) = **2.06 W**

Como se puede apreciar en los resultados, el consumo energético toma un valor relativamente elevado debido a la energía que el RPi y *MoPi* necesitan para funcionar como un ordenador de placa reducida. La implementación del modo *iBeacon* aumenta el consumo en 0.08 W, mientras que, en el modo alarma, este valor aumenta a 0.22 W.

En el presente proyecto se han utilizado baterías recargables con capacidad de 38.4 Wh. Este dato permite estimar la autonomía del sistema según el modo de funcionamiento, ya que:

$$Autonomía\ (h) = Capacidad\ batería\ / P_{1h}\ consumida \quad (3)$$

Por tanto, las autonomías para cada modo de funcionamiento con estas baterías serán:

- **Modo normal:** *Autonomía (h)* = 38.4 Wh / 1.84 W = **21,1 horas**
- **Modo *iBeacon*:** *Autonomía (h)* = 38.4 Wh / 1.92 W = **20 horas**
- **Modo *iBeacon* + alarma:** *Autonomía (h)* = 38.4 Wh / 2.06 W = **18.6 horas**

Como se comentó anteriormente, el mayor consumo energético viene determinado por el ordenador de placa reducida RPi con la placa *MoPi* instalada. A continuación, se realiza el cálculo de la autonomía que se obtendría aislando el funcionamiento del modo *iBeacon* y el modo alarma del consumo de la placa RPi, con el fin de hacer una comparativa con los datos anteriores, obteniéndose:

- **Modo *iBeacon* sin RPi:** *Autonomía (h)* = 38.4 Wh / 0.08 W = **20 días**
- **Modo *iBeacon* + alarma sin RPi:** *Autonomía (h)* = 38.4 Wh/ 0.22 W = **7 días y 16 horas**

Como conclusión a los datos analizados anteriormente, tanto el sistema *iBeacon* como el sistema *iBeacon* + alarmas presentan unas autonomías limitadas debido al uso de RPi como placa controladora. Una autonomía de menos de 24 horas no se considera factible para un sistema de control de personal a bordo debido a la constante necesidad de recargar el dispositivo. Sin embargo, se puede observar que si el sistema se implementa en una placa controladora que no requiera un consumo energético elevado, la autonomía puede aumentar a valores aceptables para la implementación del sistema.