Proyecto Fin de Grado
Grado en Ingeniería de las Tecnologías de Telecomunicación

Control remoto usando Bluetooth

Autor: Jorge Santamaría Velázquez
Tutor: Juan José Murillo Fuentes

Dep. Teoría de la Señal y Comunicaciones
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2015

Proyecto Fin de Grado
Grado en Ingeniería de las Tecnologías de Telecomunicación

Control remoto usando Bluetooth

Autor:
Jorge Santamaría Velázquez

Tutor:
Juan José Murillo Fuentes
Profesor titular

Dep. de Teoría de la Señal y Comunicaciones
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2015

Proyecto Fin de grado: Control remoto usando Bluetooth

Autor: Jorge Santamaría Velázquez

Tutor: Juan José Murillo Fuentes

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2015

El Secretario del Tribunal

A mi familia

A mis amig@s

A mis maestros

AGRADECIMIENTOS

Este trabajo marca un punto de inflexión en mi vida que, tras siete largos años de trabajo, puede considerarse como el fin de una de las etapas más importantes que he podido vivir hasta el momento, etapa que considero pilar fundamental para afrontar los nuevos retos que surjan.

Agradezco de forma incuantificable a mi familia, sobre todo a mis padres, que me hayan dado la oportunidad de poder vivir esta experiencia a partir de la que espero poder labrar un futuro propio. Hacer mención especial a mi hermana que, por la inocencia de su edad, ha hecho que pudiera quitarle importancia a muchos momentos difíciles pudiendo ver las cosas desde un punto de vista más optimista.

También agradecer a todos mis amigos los momentos compartidos en esta etapa, de estudio o de evasión, pero sobre todo a aquellas que se mostraban dudosas ante la posibilidad de que pudiera finalizar esta carrera por el gran esfuerzo que requiere, son las que han alimentado la perseverancia con la que lo he conseguido. En este punto hago mención especial a mi amiga Carmen que, desde los primeros días de esta etapa, ha sido quien más me ha aportado como compañera de estudios y, sobre todo, como amiga. También agradecer a mi novio Valentín, haberme dado el que ha sido el último empujón para finalizar mis estudios con su apoyo y cariño, ayudándome a levantar rápidamente cada vez que he tropezado.

Por último, agradecer a los profesores de la Escuela Técnica Superior de Ingenieros todo lo que me han aportado de forma docente y a nivel personal, tanto a aquellos que más difícil me lo han puesto y gracias a los que he aprendido a enfrentarme a situaciones especiales, como a los que se han mostrado cercanos y han hecho sentirme capaz de poder conseguir lo que me propusiera, especialmente a los profesores del departamento de Teoría de la Señal y Comunicaciones como Juan José Murillo Fuentes, Tutor de este proyecto, que ha hecho posible la realización del mismo.

Jorge Santamaría Velázquez

Grado en Ingeniería de las Tecnologías de Telecomunicación

Sevilla, 2015

RESUMEN

Bluetooth es una de las formas de tecnología inalámbrica para el intercambio de datos en distancias cortas mediante dispositivos fijos o móviles diseñado para el consumo de baja potencia.

Hasta la aparición de dicho protocolo de comunicación, las transmisiones de datos entre dispositivos requerían conexiones físicas para hacerlas posible o, si la transmisión era por el aire, requería que los dispositivos tuvieran una trayectoria clara y directa.

Como ventajas del Bluetooth se pueden destacar su bajo consumo de potencia a la hora de la transmisión y su existencia en la gran mayoría de dispositivos actuales, haciendo posible el intercambio de datos sin ningún tipo de coste entre la gran mayoría de personas que tienen un dispositivo móvil.

El contenido principal de este proyecto es proponer soluciones ante hábitos diarios en nuestras vidas que nos permitan una mayor simplicidad y comodidad, mediante el uso de la tecnología inalámbrica Bluetooth implantada en dispositivos cada día más cotidianos, desde un teléfono móvil hasta una Tablet. Previamente se analizarán las tecnologías inalámbricas existentes, comparándolas con Bluetooth. A continuación, se explicará la arquitectura y procedimientos empleados, seguido de las herramientas utilizadas para llevar a cabo el objetivo del proyecto así como sus implementaciones.

La funcionalidad del sistema que se va a desarrollar será la siguiente: el usuario del servicio, mediante un una aplicación en un dispositivo móvil, se identificará en el sistema de forma automática a través del IMEI de dicho dispositivo. Dependiendo del tiempo que lleve haciendo uso del servicio, la identificación tendrá lugar directamente en el controlador mediante el IMEI que previamente el dispositivo móvil le haya enviado vía Bluetooth o, tras haber hecho una consulta a una base de datos que permita al dispositivo móvil verificar que dicho usuario puede hacer uso del servicio y pueda llevarse a cabo la identificación en el controlador. Después del envío del IMEI al controlador, éste comprueba si el IMEI recibido es de un usuario y si todo es correcto procede a poner en funcionamiento el sistema de apertura de la puerta.

ÍNDICE

ÍNDICE DE TABLAS

Índice de Figuras

1 INTRODUCCIÓN

La motivación de este trabajo no es otra que facilitar, mediante la tecnología Bluetooth, la realización de tareas habituales para gran parte de la población.

El objetivo principal de este proyecto es controlar de forma remota, mediante una aplicación compatible con el sistema operativo del transmisor de señales y con el receptor (ambos programados de forma previa y capacitados para trabajar con la tecnología Bluetooth), para que actúen respecto a una serie de órdenes según la necesidad en un momento dado.

El proyecto está orientado sobre todo, aunque las aplicaciones podrían implantarse en otros escenarios con características similares, a prestar facilidades para acceder a diversos lugares donde sea posible aplicar la tecnología Bluetooth para automatizar la apertura de puertas en dichos lugares. Las especificaciones a seguir han sido las siguientes:

- Identificación automática y personal.
- Evitar que la conexión con el sistema receptor esté ocupada demasiado tiempo impidiendo a otro dispositivo hacer uso de ella.
- Permitir hacer uso del servicio en cuanto queramos empezar a usarlo.

La metodología ha sido, sencillamente, plantear un problema y buscar una solución en la que estuviera involucrada el sistema de transmisión vía Bluetooth, sin perder de vista el objetivo principal del proyecto, dar facilidad al usuario ante una acción habitual y, a poder ser, ahorrando costes.
Se ha elegido dicho protocolo de comunicación por estar implantado en un gran número de dispositivos móviles, haciendo que el uso del sistema propuesto en los siguientes capítulos pueda hacerse a la mayor escala posible en cuanto a usuarios se refiere.

Para resolver dicho problema de una forma sencilla para el usuario y sin perder de vista el objetivo mencionado, se ha realizado una aplicación para que dispositivos móviles con sistema operativo Android actúen emitiendo una señal si y solo si se cumplen ciertos requisitos detallados más adelante. Se ha procedido con este sistema operativo en particular por las numerosas facilidades existentes a la hora de programar una aplicación con el fin que requiera su usuario, como por ejemplo, poder realizar dicha aplicación sin coste alguno, o que dicho sistema operativo esté implantado en la mayoría de dispositivos móviles.

Como sistema receptor de la señal anteriormente explicada, se ha utilizado una plataforma de hardware libre, conocida como Arduino, detallada más adelante.

Además de la aplicación mencionada, para que el sistema pueda funcionar, se necesita un servidor web que aloje una base de datos MYSQL (my structured query language), en la que se almacenará la identificación del dispositivo móvil de cada usuario, y un servicio web programado en PHP (hypertext preprocessor) encargado de intercambiar datos entre la aplicación Android y la base de datos.

En este documento, en primer lugar se explicarán las diferentes tecnologías de corto alcance; tras esto, se desarrolla en profundidad el tema principal del proyecto, la tecnología Bluetooth: orígenes, forma de funcionamiento, seguridad, protocolos de nivel respecto al modelo OSI (open system interconection).

A continuación se detallan las especificaciones a cumplir por el sistema, así como los objetivos y su arquitectura, detallando los procesos que tendrán lugar en cada parte del sistema.

Una vez definidas las especificaciones y las partes del sistema, así como los procesos por los que puede pasar en cada momento, se verán las características de las herramientas utilizadas, tanto físicas como lógicas, siguiendo con la explicación del desarrollo del sistema y su funcionamiento.

Finalmente se han propuesto diferentes mejoras del sistema así como otros escenarios donde se podría implantar.

Como conclusión se puede adelantar que, si se implantara el sistema por control remoto propuesto más adelante, ahorraríamos costes en la producción de objetos físicos (como el plástico e impresión de tarjetas de identificación y acceso a lugares restringidos), aumentaría la seguridad ante el uso fraudulento de otros sistemas que no requieren una identificación personal y nos permitiría despreocuparnos de depender de los objetos necesarios (tarjetas, mandos, llaves magnéticas...) para poder acceder a lugares donde la única opción que lo permitiera fuera mediante el uso de identificación con el sistema anteriormente mencionado.

En los dos siguientes capítulos se lleva a cabo la definición y enumeración de características de diferentes tecnologías inalámbricas, así como de las de Bluetooth.

2 TECNOLOGÍAS DE CORTO ALCANCE

Desde sus inicios, al igual que el de cualquier otra tecnología inalámbrica, el objetivo principal del Bluetooth es evitar la conexión de periféricos mediante un medio físico como los cables, haciendo posible una movilidad más amplia y cómoda al usar dichos periféricos.

A continuación se revisan algunas soluciones alternativas a Bluetooth para comunicación de corto alcance entre dispositivos.

2.1. Infrarrojos

Una forma de comunicación de corto alcance de forma inalámbrica entre los dispositivos involucrados es mediante una conexión óptica y en particular la conocida como conexión infrarroja (IRDA).

Este tipo de comunicación tiene sus ventajas y desventajas respecto a la comunicación vía radio. Como clara desventaja, podemos señalar que este tipo de comunicación exige una trayectoria clara, directa y cercana entre ambos dispositivos, aparte de obligar a que el tipo de transferencia sea punto a punto. Por el contrario, tecnologías radio como Bluetooth, permiten que el intercambio de información no requiera una visión directa, que la distancia entre los dispositivos sea hasta de 10 metros y que un mismo receptor pueda soportar hasta 10 canales diferentes, es decir, 10 dispositivos conectados simultáneamente a él.

Como ventaja del sistema de transferencia mediante luz infrarroja (longitud de onda igual a 850 nanómetros), podemos resaltar aquellos casos en los que necesitemos un intercambio de datos rápido y simple, como podría ser la lectura de un código de barras de uno o varios productos (cada uno tratado de forma individual) en unos grandes almacenes o la información de los asistentes a una reunión, es decir, la identificación de algo de forma rápida y puntual.

En las mismas situaciones expuestas en el párrafo anterior, Bluetooth no tendría un comportamiento tan efectivo como empleando infrarrojos, aunque el objetivo cumplido fuera el mismo, hacerlo mediante Bluetooth conllevaría una inversión de tiempo mayor ya que antes de obtener la información deseada, tendríamos que identificar al objeto que nos proporcionará dicha información y, una vez identificado, realizar la toma de datos.

Uno de los casos que se asemeja al anterior mencionado en los que Bluetooth presentaría ventajas respecto a la tecnología infrarroja, sería el caso en el que en vez de querer extraer información puntual de un objeto, quisiéramos extraerla de varios a la vez, ya que en este caso la inversión de tiempo realizada en el reconocimiento de los dispositivos que nos proporcionaran la información y posterior envío de datos, sería menor que el tiempo invertido al realizar la misma adquisición de información uno a uno.

Otras de las ventajas que presenta Bluetooth respecto a la tecnología infrarroja es la posibilidad de penetrar en

objetos sólidos, sin necesidad de que entre ambos dispositivos haya una trayectoria con visión clara entre ambos, como la posibilidad de un intercambio automático de información tras haberse realizado anteriormente otro intercambio entre esos mismos dispositivos, reconociéndolo como fiable y ahorrándonos de nuevo el proceso de emparejamiento (definido más adelante) y permisos para dicha transferencia.

Como resumen, a continuación se adjunta una tabla con las características de funcionamiento de la tecnología de transmisión infrarroja. Para Bluetooth se hará más adelante atendiendo y analizando los aspectos fundamentales de éste respecto para cada versión del estándar.

Tabla 2-1 Características de funcionamiento de los infrarrojos [1]

Característica/función	Funcionamiento
Tipo de conexión	Infrarrojos, haz estrecho (ángulo de 30 grados o menos)
Espectro	Óptico, 850 nanómetros (nm)
Potencia de transmisión	100 milivatios (mW)
Velocidad de transferencia de datos	Hasta 16Mbps usando VFIR
Alcance	Hasta 1 metro
Dispositivos soportados	Dos (2)
Canales de voz	Uno (1)
Seguridad de los datos	El corto alcance y estrecho ángulo del haz de infrarrojos ofrecen una forma simple de seguridad.
Direccionamiento	Cada dispositivo tiene un identificador físico de 32 bits que se utiliza para establecer una conexión con otro dispositivo

2.2.**Redes de área local inalámbricas**

Las redes de área local inalámbricas (LAN), descritas en el estándar 802.11 establecido por el IEEE, son diferentes a las redes basadas en la especificación Bluetooth, presentando como mayor diferencia que uno o más puntos de estas redes se conectan a un concentrador ethernet el cual realiza la conexión a la red mediante cableado, proporcionando la interfaz entre la red de cable y la inalámbrica

Tabla 2-2. Características de funcionamiento de las LAN inalámbricas 802.11 [1]

Característica/función	Funcionamiento
Espectro	Banda ISM de 2.4 GHz y 5 GHz
Potencia de transmisión	100 milivatios (mW)
Alcance	Hasta 100 metros desde el punto de acceso, aproximadamente
Tipos de prestaciones	Múltiples dispositivos por punto de acceso; múltiples puntos de acceso por red
Direccionamiento	Cada dispositivo con una dirección MAC de 48 bits

2.3.**Redes vía Home-RF**

Home-RF es una tecnología inalámbrica cuya principal característica, gracias a que utiliza expansión en frecuencia de espectro de saltos de frecuencia con una velocidad más alta que Bluetooth, es la mínima posibilidad de interferencia entre productos que utilicen la misma tecnología en la misma zona. Este tipo de red inalámbrica puede dar cabida hasta a 127 nodos (cada uno de ellos de 4 tipos diferentes, que no se definirán ya que no es el objetivo de este documento).

Este tipo de transmisión está pensada para su implementación en tarjetas PC-Card.

Algunas de las diferencias respecto a la tecnología Bluetooth que podemos mencionar son: el alcance de transmisión, 50 metros en Home-RF y 10 metros en Bluetooth; la potencia necesaria, 100 milivatios en Home-RF y 1 milivatio en Bluetooth o el coste, que será mucho menor en dispositivos Bluetooth.

Cabe destacar también que, aunque quisiéramos combinar ambas tecnologías descritas, no son compatibles, siendo la única opción un sistema que conmute entre una y otra.

Tabla 2-3 Características de funcionamiento de Home-RF [1]

Característica/función	Funcionamiento
Tipo de conexión	Expansión de espectro (saltos de frecuencia)
Espectro	Banda ISM de 2.4 GHz
Potencia de transmisión	100 milivatios (mW)
Velocidad de transferencia de datos	1 Mbps o 2Mbps dependiendo de la modulación de saltos de frecuencia
Alcance	Cobertura media de un hogar
Estaciones soportadas	Hasta 127 dispositivos por red
Canales de voz	Hasta 6
Seguridad de los datos	Algoritmo cifrado de Blowfish
Direccionamiento	Cada dispositivo con una dirección MAC de 48 bits

2.4.**ZigBee**

ZigBee es un conjunto de protocolos para la comunicación inalámbrica para radiodifusión digital de bajo consumo. Basado en las redes WPAN, tiene como objetivo aquellas aplicaciones en las que son necesarias comunicaciones seguras con baja tasa de envío y maximización de la vida útil de sus baterías [2].

ZigBee es también conocida como HomeRF Lite.

2.1.1 **Categorías de los dispositivos ZigBee**

Hay tres tipos de categorías respecto a lo que nodos ZigBee se refiere, dependiendo de su función:

- Coordinador ZigBee (ZC, ZigBee Coordinator): es el nodo más completo, y se encarga de controlar toda la red y los caminos para su comunicación. Debe existir al menos uno por red.

-Router ZigBee (ZR, ZigBee router): nodo que interconecta a los demás nodos para poder ejecutar el código del usuario, o lo que es lo mismo, ofrece un nivel de aplicación dentro de la torre de protocolos.

-Dispositivo final ZigBee (ZED, ZigBee end device): nodo que solo recibe información y se comunica solo con el nodo padre. Este dispositivo tiene la ventaja de poder ahorrar energía permaneciendo 'dormido' y 'despertarse' cuando reciba información del nodo anteriormente mencionado. Con dos pilas AA, un nodo ZigBee de esta categoría puede funcionar desde 6 meses hasta 2 años.

Otra clasificación que podría plantearse sería respecto a la funcionalidad del nodo en la red, pudiendo distinguir:

-Dispositivo de funcionalidad completa (FFD, full-function device) o nodo activo: puede funcionar como ZC o ZR.

-Dispositivo de funcionalidad reducida (RFD, reduced-function device) o nodo pasivo: capacidad y funcionalidad limitadas con el objetivo de conseguir un bajo coste y gran simplicidad. Se les suele llamar sensores/actuadores.

Tanto los nodos activos como pasivos pueden permanecer dormidos mientras no se requiera su uso, pudiendo despertar en unos 15 milisegundos aproximadamente [2].

2.1.2 **Características de la tecnología ZigBee**

La tecnología ZigBee utiliza la banda ISM de 2.4GHz (mundialmente) además de la de 868MHz (en Europa) y la de 915MHz (en EEUU), para comunicarse con el resto de dispositivos, en 16 canales cada uno con un ancho de banda de 5 MHz. Para evitar la colisión de transmisiones entre diferentes canales, se utiliza CSMA/CA.

Las velocidades de transmisión en este tipo de tecnología se comprenden entre los 20kbps y 250kbps, con rasgos de alcance entre 10 metros y 75 metros.

El número de nodos ZigBee en una misma subred puede ser de hasta 255, pudiendo formar una red de hasta 65535 nodos, siendo la mayoría de ellos de tipo dispositivo final, definido anteriormente.

Como principales ventajas, podemos resaltar su fácil integración, bajo consumo, topología de red mallada, baja tasa para redes de transferencia de datos, fragmentación de mensajes largos, recolección centralizada de datos, reducción del ciclo de trabajo para proporcionar larga duración a la batería, bajo coste y sencillez en su construcción.

También se puede resaltar la agilidad de frecuencia con la que cuenta esta tecnología, es decir, la rapidez con la que las redes cambian los canales en forma dinámica en caso que ocurran interferencias.

Por el contrario, las desventajas que presenta este tipo de tecnología son: baja tasa de transferencia de datos en comparación con otras tecnologías, manipulación de textos pequeños, menor cobertura (al pertenecer al tipo de redes WPAN).

Otra de las características que presenta es tecnología, es que si la comparamos con la de Bluetooth, ZigBee necesita un código más amplio, ya que necesita alrededor de un 50% del código que se utiliza en Bluetooth [2].

2.1.3 **Topologías en la tecnología ZigBee**

Aunque la topología más usada es en malla, este tipo de tecnología puede ser en estrella, árbol y punto o punto.

-Topología en estrella: uno de los dispositivos FFD asume el rol de coordinador de la red, siendo el responsable de inicializar y mantener los dispositivos en la red. Los demás dispositivos serán de tipo final.

-Topología en malla: el coordinador ZigBee es el responsable de inicializar la red y elegir los parámetros de ésta. La ventaja de esta topología es que permite, mediante los dispositivos funcionando como routers, que la red pueda ser ampliada. También cabe destacar que este tipo de topología permite que en caso de caída de un camino hacia un nodo, tengamos otra manera de llegar a éste.

-Topología punto a punto: existe un solo dispositivo FFD y, a diferencia de la topología en estrella, cualquier dispositivo puede comunicarse con otro siempre que esté en el mismo rango de alcance circundante. Este tipo de topología provee confiabilidad en el enrutamiento de datos.

-Topología en árbol: es un caso especial de la topología punto a punto cuya diferencia es que muchos dispositivos son FFDs y los que actúan como RFD pueden conectarse como un único nodo al final de la red.

A continuación se muestra una tabla comparativa de las aplicaciones típicas de las tecnologías inalámbricas, teniendo todas en común el objetivo de eliminación de cables.

Tabla 2-4 Características de funcionamiento de las distintas soluciones

Tecnología	Aplicaciones típicas
Bluetooth	Comunicación entre dispositivos para voz y datos, redes PAN, control remoto de dispositivos, comercio electrónico móvil
ZigBee	Monitoreo y control de procesos industriales, redes de sensores inalámbricos...

Infrarrojos	Transferencia de archivos a alta velocidad entre dispositivos, control local de dispositivos
Home-RF	Comunicación de datos entre computadoras y entre computadoras y periféricos
LAN inalámbrica 802.11	Comunicación de datos entre computadoras y entre computadoras y periféricos

2.5. **Conclusiones**

Para finalizar este capítulo, se destacan las razones por las que se ha utilizado Bluetooth como tecnología inalámbrica para dar solución al problema planteado:

- En primer lugar, cabe destacar la característica que hace más interesante a la utilización de Bluetooth que es la existencia de dicho protocolo de comunicación en un gran número de los dispositivos móviles existentes, haciendo posible que un gran número de personas puedan hacer uso del sistema que se llevará a cabo en los siguientes capítulos.
- Otro de los motivos son las características que definen al Bluetooth, como el alcance o el bajo consumo, haciendo que de entre los diferentes protocolos de comunicación inalámbrica éste sea el más eficiente para cumplir el objetivo planteado.

3 CARACTERÍSTICAS DE LA TECNOLOGÍA BLUETOOTH

3.1. Los orígenes de Bluetooth

En 1994, la compañía global de telecomunicaciones Ericsson Mobile Communications investigó la viabilidad de una interfaz de radio de baja potencia y bajo coste entre dispositivos móviles y sus accesorios, llegando a la conclusión de que dicho tipo de conexión debería ser un enlace de radio de corto alcance.

Esta investigación atrajo la atención de otras grandes compañías como IBM, Intel, Nokia y Toshiba, creando el SIG Bluetooth, consorcio inalámbrico que creció exponencialmente tras pocos años. Estas compañías desarrollaron la primera especificación Bluetooth, conocida como Bluetooth 1.0 en 1999 la cual será, junto a las demás existentes hasta la actualidad, analizada más adelante.

3.1.1 Origen del nombre

El nombre fue dado con motivo de honrar a un rey vikingo danés del siglo X. Haral Bluetooth, el cuál reinó desde el año 940 hasta el 985 y al que se le atribuye la unificación del mencionado país y la adopción del cristianismo.

La relación entre el nombre y la tecnología Bluetooth es que lo que se pretende con ésta última es la unificación y armonía para permitir a diferentes dispositivos que se comuniquen a través de un estándar aceptado para la comunicación inalámbrica [1].

3.2. Bluetooth: red PAN

El grupo de trabajo 802.15 del IEEE tiene como objetivo la creación de estándares que proporcionen una base para un amplio rango de dispositivos de consumo interoperables, coexistiendo con otros tipos de tecnologías inalámbricas como las definidas en el 802.11.

Los dispositivos que utilicen redes PAN inalámbricas y la tecnología Bluetooth podrán ser usados en cualquier país.

3.3. La tecnología Bluetooth

El conjunto de especificaciones Bluetooth desarrolladas por Ericsson y otras compañías responde a las necesidades de conectividad inalámbrica de corto alcance para redes ad hoc.

Una red ad hoc es un tipo de red inalámbrica descentralizada, es decir, no depende de una infraestructura pre-existente.

El protocolo de banda base de Bluetooth es una combinación de conmutación de circuitos y de paquetes, haciéndola idónea para transmitir tanto voz como para datos.

La tecnología Bluetooth se implementa en transceptores de corto alcance, de pequeño tamaño y bajo coste en dispositivos móviles, integrada de forma directa o mediante dispositivos adaptadores.

La tecnología inalámbrica utiliza la banda de radio ISM mundialmente disponible, y que no requiere licencia, de 2.4 GHz. Éstas bandas incluyen los rangos de frecuencia entre 902-928 MHz y 2.4-2.484 GHz que no requieren licencia de operador otorgada por las autoridades reguladoras de telecomunicaciones [1].

3.3.1 Tipos de enlaces

En la especificación Bluetooth se han definido dos tipos de enlaces que la especificación permite implementar al mismo tiempo:

3.3.1.1 Enlace asíncrono sin conexión (ACL)

En los enlaces ACL, la información transmitida pueden ser datos de usuario o control y soportan tráfico de datos sin garantía de entrega. Las conexiones en este tipo de enlace son simétricas o asimétricas, de conmutación de paquetes y punto a multipunto (éstas últimas suelen ser utilizadas para datos). En las conexiones simétricas, la transferencia de datos máxima es de hasta 433.9 Kbps en ambas direcciones (envío y recepción).

3.3.1.2 Enlace síncrono orientado a conexión (SCO)

En los enlaces SCO, la información transmitida puede ser voz en tiempo real y tráfico multimedia, utilizando un ancho de banda reservado. Estos tipos de enlaces ofrecen conexiones simétricas por conmutación de circuitos, punto a punto, usualmente utilizadas para voz, para la que hay disponibles tres canales síncronos de 64Kb cada uno. Estos canales son creados utilizando modulación por impulsos codificados (PCM) o modulación diferencial de pendiente continuamente variable.

PCM es el estándar para codificar voz en forma analógica a formato digital de unos y ceros para su transmisión por la red telefónica. También es preciso mencionar el estándar CVSD, el cual ofrece más inmunidad a las interferencias, haciéndolo más adecuado que PCM para comunicaciones de voz sobre enlace inalámbrico.

De los anteriores tipos de enlaces definidos, en el sistema que se desarrollará en este proyecto se utilizará un enlace ACL ya que la información transmitida serán caracteres, o lo que es lo mismo, datos de usuario.

3.3.2 Enlace vía radio

Como característica especial del enlace vía radio se puede destacar su robustez, cuya razón es la consecuencia directa de utilizar la tecnología de expansión de espectro con saltos de frecuencia para hacer que los efectos de las interferencias y desvanecimientos sean mínimos.

La técnica de expansión del espectro consiste en "expandir" la señal mediante el aumento del número de bits transmitidos en la operación de codificación. Esta expansión hace que la señal sea prácticamente ruido para un oyente que no sea el receptor de la señal original, combatiendo las interferencias procedentes de otros dispositivos que también trabajan en la banda ISM ya definida anteriormente. Esto es una característica que hace bastante especial a la tecnología Bluetooth, ya que hay que tener en cuenta que objetos cotidianos como los microondas, redes locales inalámbricas u otros usados frecuentemente, se mueven en el mismo rango de frecuencias. Para que el receptor interprete la señal tal y como es realmente, éste tiene el mismo código que

utilizó el transmisor para ensanchar la señal que, mediante la operación de correlación, contrae la señal expandida y le devuelve su forma original. Al expandir la señal, otra consecuencia es que la potencia de ésta se expande utilizando una mayor banda de frecuencias, lo que la hace más robusta ante ruidos electromagnéticos y otras fuentes de interferencias. Con el añadido de saltos en frecuencia (las señales saltan de una frecuencia a otra), las transmisiones inalámbricas se hacen aún más seguras contra escuchas. Cada dispositivo de expansión salta 1600 veces por segundo entre 79 frecuencias distintas, saltos con una secuencia que el dispositivo que inicia la conexión le dice al otro.

En los casos en los que hay demasiadas interferencias, aunque la transmisión se pierde solo durante un milisegundo, se puede aumentar la fiabilidad enviando cada bits de datos por triplicado.

Como resumen a las características mencionadas anteriormente, a continuación se adjunta una tabla con las características más importantes del funcionamiento de los productos Bluetooth:

Tabla 3-1 Características de funcionamiento de los productos Bluetooth [1]

Característica/función	**Funcionamiento**
Tipo de conexión	Expansión de espectro (saltos de frecuencia)
Espectro	Banda ISM de 2.4 GHz
Potencia de transmisión	1 milivatio (mW)
Velocidad de datos total	1 Mbps
Alcance	Hasta 10 metros
Estaciones soportadas	Hasta ocho dispositivos por picorred
Canales de voz	Hasta tres
Seguridad de los datos	Autentificación: clave de 128 bits; cifrado: tamaño de clave configurable, entre 8 y 128 bits
Direccionamiento	Cada dispositivo con una dirección MAC de 48 bits

3.4. **Topología Bluetooth**

Una picorred (piconet), se define como un conjunto de unidades que comparte un canal común. Pueden soportarse hasta ocho dispositivos interconectados en una misma picorred, pudiendo haber un maestro y siete esclavos: un dispositivo que juega el papel de maestro, es aquél cuyo reloj y secuencia de saltos se utilizan para sincronizar a todos los demás dispositivos (los esclavos) de esa misma picorred a la que pertenece. Los esclavos son unidades de la picorred que se sincronizan con el maestro mediante su reloj y frecuencia de saltos.

Por lo anteriormente comentado, la topología Bluetooth se puede describir como un conjunto de picorredes interconectadas entre sí, ya que como la especificación Bluetooth soporta conexiones punto a punto y punto a multipunto, podemos enlazar varias picorredes formando una topología comúnmente conocida como red dispersa (scatternet), en las cuales cada picorred que la conforman se identifica por la frecuencia de saltos que tengan.

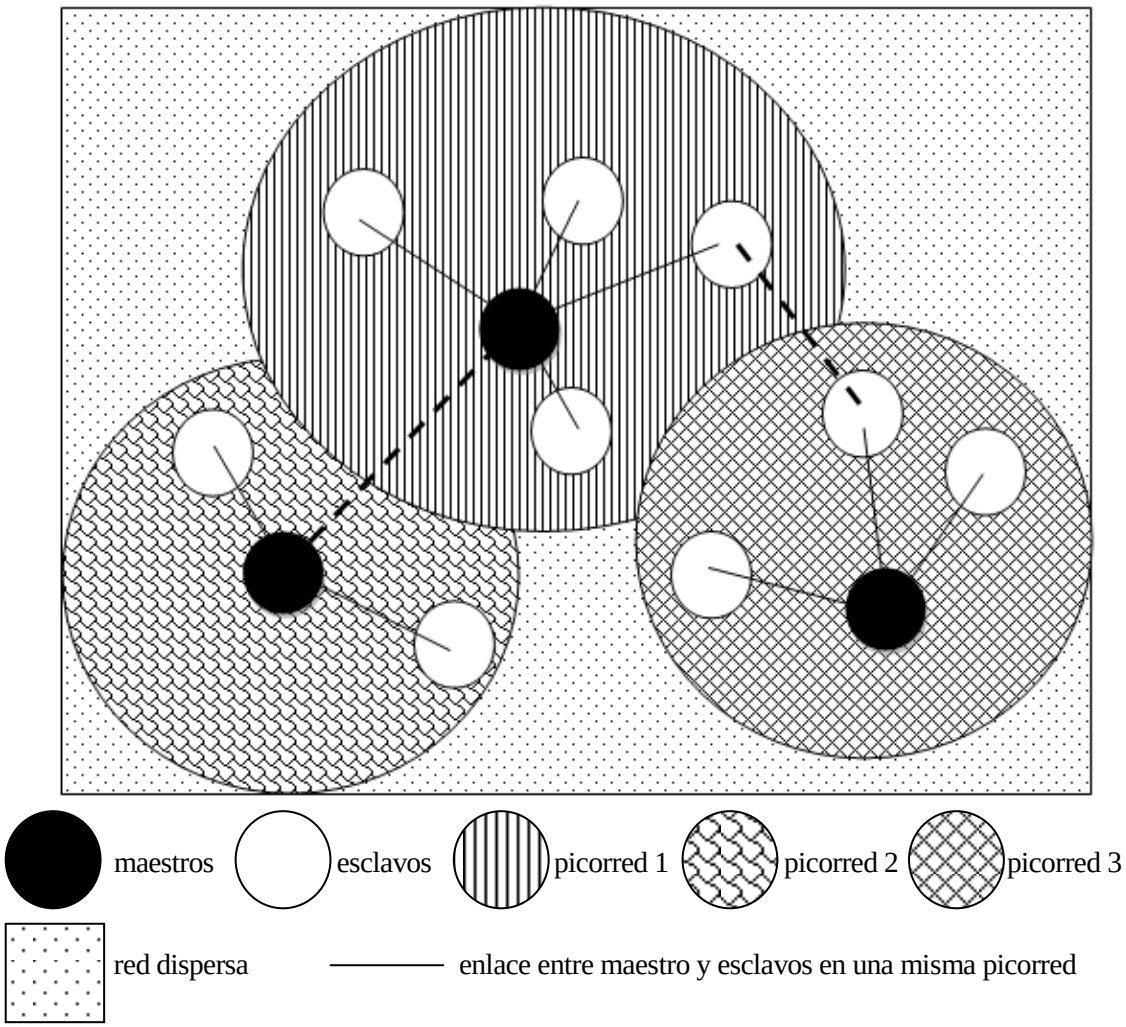


Figura 3-1. Interconexión entre varias picorredes formando una red dispersa

3.5. **Seguridad**

A parte de la seguridad proporcionada por el alcance limitado y la utilización de saltos de frecuencia, la especificación Bluetooth emplea otras funciones en otros niveles del modelo OSI que la hacen más segura aún.

A nivel de enlace, emplea funciones como autentificación y cifrado, definidas a continuación [1].

3.5.1 **Gestión del enlace**

Cuando dos dispositivos Bluetooth se encuentran dentro del radio de acción del otro, la entidad gestora de enlace de cada dispositivo, utiliza el mecanismo de descubrimiento para encontrar al otro dispositivo mediante una comunicación que intercambia mensajes a través del protocolo LMP. Estos mensajes realizan el establecimiento del enlace, incluyendo la autentificación y el cifrado que comprende: generación, intercambio y comprobación de las claves de cifrado y de enlace.

3.5.1.1 **Autenticación**

El método de autenticación tiene como finalidad prevenir la suplantación y acceso no deseado a datos o funciones, y se basa en un esquema de desafío-respuesta donde, tanto el maestro como el esclavo pueden actuar como verificadores.

El verificador envía una PDU (packet data unit) con un número aleatorio al solicitante, el cual calcula una respuesta en función de la dirección de dispositivo Bluetooth del solicitante, y de una clave secreta. La respuesta se envía de vuelta al verificador, que comprueba si es correcta o no. Para que el cálculo de la respuesta de autentificación sea correcta, se requiere que los dos dispositivos compartan la clave secreta.

Cuando falla la autenticación, y para prevenir que un intruso pruebe muchas claves diferentes hasta dar con la correcta, antes de que se pueda hacer un nuevo intento de autenticación, debe transcurrir un cierto intervalo de tiempo de espera. Para cada error de autentificación, el intervalo de espera para poder introducir una nueva clave, será mayor, aumentando de forma exponencial. El intervalo de espera entre errores de autenticación también puede disminuir, hecho que tendrá lugar cuando no se hacen intentos fallidos de autenticación durante un cierto periodo de tiempo.

3.5.1.2 **Emparejamiento**

Inicialmente, es decir, cuando dos dispositivos no se conocen, éstos no tendrán una clave común, por lo que se debe crear una clave de inicialización basada en un número de identificación personal y en un número aleatorio.
La clave de inicialización se crea tras hacerse una autenticación basada en la clave de inicialización, la cual, si es exitosa, se crea la clave de enlace.

La clave de enlace creada en el procedimiento de emparejamiento puede ser una clave de combinación o una clave de unidad de una de las dos unidades participantes, según el intercambio de PDU intercambiadas en el proceso de emparejamiento.

La clave combinada es el tipo más seguro de clave de enlace, derivada de los datos proporcionados por los dispositivos involucrados.

Cabe notar también, que la clave de enlace puede ser cambiada según el tipo de clave que sea. Si la clave de enlace ha sido calculada a partir de claves de combinación, se puede cambiar de forma manual, mientras que, si la clave de enlace ha sido calculada a partir de la clave de unidad de una de las dos unidades participantes, para cambiar la clave de enlace, las unidades deben realizar de nuevo el procedimiento de emparejamiento.

Para la primera conexión, el emparejamiento requiere que el usuario introduzca un código de seguridad Bluetooth, habitualmente denominada PIN, aunque no es exactamente un código PIN, ya que solo hay que introducirlo una vez.

Una vez que un dispositivo ha sido autentificado y se le ha permitido que tenga acceso basándose en su clave de nivel de enlace, pasa a ser un dispositivo confiable.

3.5.1.3 **Cifrado**

El cifrado es opcional, siendo su objetivo el de codificar los datos durante la transmisión para evitar escuchas y mantener la privacidad del enlace, protegiendo la confidencialidad, siendo el único requisito haber realizado al menos una autenticación.

Si el maestro quiere que todos los esclavos de la picorred utilicen los mismos parámetros de cifrado, tiene que generar una clave temporal y hacer que dicha clave sea la clave de enlace actual para todos los esclavos de la picorred antes de que comience el cifrado.

En primer lugar, el maestro y el esclavo deben ponerse de acuerdo en si utilizar cifrado o no. Otro requisito es que el cifrado solo se debe aplicar a los paquetes punto a punto o a los de difusión.

En segundo lugar, se determina el tamaño de la clave de cifrado mediante PDU's para negociar dicho tamaño.

Después comienza el cifrado, donde el maestro genera el número aleatorio y calcula la clave de cifrado. Éste número aleatorio será el mismo para todos los esclavos si la picorred soporta difusión cifrada. El maestro enviará un mensaje requiriendo el comienzo de la encriptación, mensaje donde estará incluido el número aleatorio. El esclavo, al recibirlo, calculará la clave de enlace actual y enviará de vuelta al maestro un mensaje de confirmación. En ambos lados se utilizan la clave de enlace actual y el número aleatorio como

entrada para el algoritmo cifrado.

Antes de comenzar el cifrado, se detiene el tráfico de datos de niveles superiores para evitar la recepción de datos corruptos.

Etapas en el inicio de cifrado:

1. Se configura el maestro para transmitir paquetes sin cifrar, y para recibir paquetes cifrados.
2. Se configura el esclavo para transmitir y recibir paquetes cifrados.
3. Se configura el maestro para transmitir y recibir paquetes cifrados.

Mediante el proceso de autenticación descrito anteriormente, la especificación Bluetooth solo autentica dispositivos, no a usuarios, es decir, que un dispositivo confiable que sea robado o prestado puede seguir siendo utilizado por otra persona.

Si hay necesidad de autenticación del usuario, se deben emplear métodos de seguridad suplementarios de nivel de aplicación, como la introducción de un nombre de usuario y contraseña, método que será necesario para la parte de aplicación práctica de este documento o como por ejemplo, donde está implementada la autenticación OBEX.

El dispositivo Bluetooth utilizado en el desarrollo del sistema especificado más adelante, cuenta con los 3 métodos de seguridad respecto a la gestión del enlace que se acaban de definir.

3.5.2 **Modos de seguridad**

Debido a que la demanda de seguridad será diferente atendiendo al lugar donde queramos implantarla respecto a la acción que queramos llevar a cabo, se definen 3 modos de seguridad que cubren la funcionalidad y la aplicación de los dispositivos [1].

3.5.2.1 **Modo 1**

En este modo hay ausencia de seguridad, y es utilizado cuando los dispositivos no tienen aplicaciones críticas, modo en el que los dispositivos desactivan las funciones de seguridad de nivel de enlace.

3.5.2.2 **Modo 2**

Este modo proporciona seguridad de nivel de servicio, especialmente para aplicaciones que se ejecuten en modo paralelo. Este modo nos permite definir niveles de seguridad para dispositivos y servicios.

Un dispositivo puede mantener dos niveles de confianza:

- Un dispositivo es de confianza cuando disfruta de acceso sin restricciones a todos los servicios. En los dispositivos que requieran autorización y autenticación, el acceso automático estará garantizado para dispositivos de confianza, y los demás dispositivos necesitaran un acceso manual.
- Un dispositivo no confiable es aquel que tendrá accesos restringidos a los servicios.

3.5.2.3 **Modo 3**

Este modo proporciona seguridad de nivel de enlace, donde el gestor de enlace (LM) impone una seguridad de nivel común para todas las aplicaciones en el momento de la configuración de la conexión.

Es de destacar la baja flexibilidad de este modo, aunque obliga a un nivel de seguridad común y es más fácil de implementar que el modo 2.

3.5.3 **Transmisión serie frente a paralelo**

La tecnología inalámbrica Bluetooth, al igual que otras tecnologías de comunicación, utiliza las comunicaciones serie para transmitir datos en forma binaria (unos y ceros). Este tipo de comunicación supone una transferencia de datos de forma secuencial (unos y ceros unos de tras de otro), modo de transferencia aplicado tanto a comunicaciones de datos síncronas como asíncronas.

3.5.3.1 **Transmisión serie**

Método de transmisión de datos (envío y recepción) bit a bit de forma secuencial. Este tipo de transmisión no solo se utiliza para conectar dispositivos locales entre sí, sino que son la base de todas las comunicaciones de datos en redes de área local (LAN) y área extensa (WAN), y en el medio físico existente en ellas.

3.5.3.2 **Transmisión paralelo**

Método de transmisión de datos (envío y recepción) que implica recibir un byte completo cada vez, donde cada uno de los ocho bits que componen dicho byte, se transmite por un canal diferente. Este tipo de transmisión sólo es buena para distancias cortas.

Como conclusión a este apartado, se puede destacar que, en la tecnología Bluetooth, los datos transmitidos a otros dispositivos se realiza en serie, de forma síncrona o asíncrona, y los datos transmitidos en los dispositivos internos de Bluetooth son transmitidos combinando la transmisión serie y paralelo.

Para finalizar el apartado, se puede poner como ejemplo dos casos de los diferentes tipos de transmisión anteriormente descritas: en el cifrado de datos para la transmisión en serie, se emplea un proceso de cifrado de flujo que carga cuatro entradas en paralelo a un generador de claves de datos, información que es utilizada para formular una clave que será enviada en serie al codificador, donde sería cifrada justo antes de ser enviada por el enlace inalámbrico.

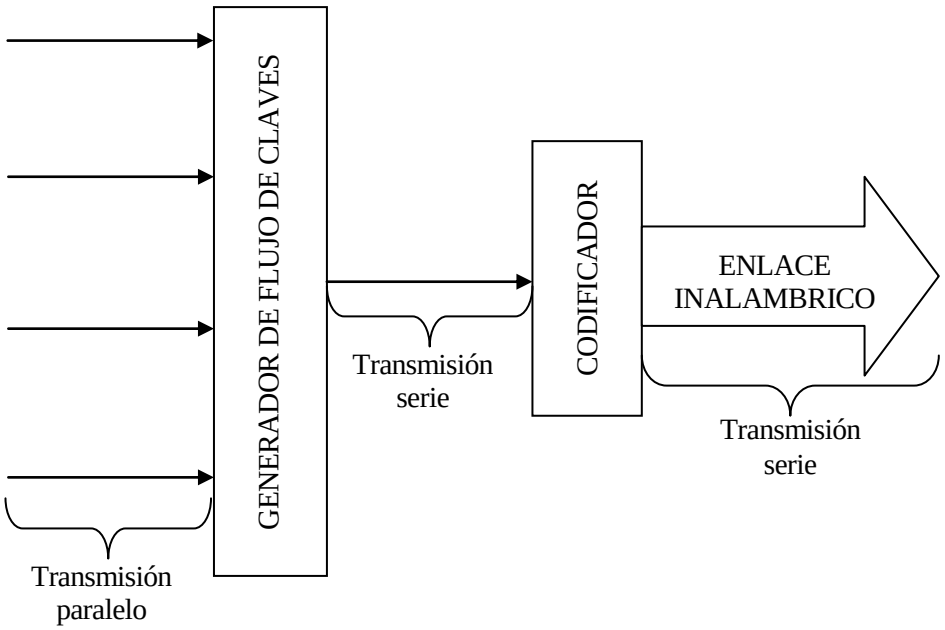


Figura 3-2. Comparación de la transmisión serie y paralelo

De los tipos de transmisión anteriormente definidos, el dispositivo utilizado en el desarrollo del proyecto utiliza una transmisión tipo serie.

3.6. Enlace asíncrono frente a síncrono

La transmisión de datos en serie presenta el siguiente problema:¿cómo sabemos cuando comienza cada nuevo carácter en un flujo continuo de bits?

Para resolver dicho problema, necesitamos de algún tipo de sincronización dada entre el emisor y el receptor.

Hay dos tipos de sincronización, ambas soportadas por la especificación Bluetooth, que se explicarán a

continuación [1]:

3.6.1 **Asíncrono**

En una transmisión asíncrona, la sincronización es resuelta acotando cada conjunto de ocho bits con un bit de arranque y otro de parada, así, cuando el receptor recibe dicho bit de arranque, instantáneamente sabe que se le está enviando un carácter, el cual el emisor enviará justamente después de enviar el mencionado bit de arranque. Una vez enviado el carácter, seguido de éste, se enviará un bit de parada, el cuál le indica al receptor que se ha completado la transmisión del carácter. El proceso descrito se repite durante toda la sesión.

En este tipo de transmisión, los bits en el flujo de datos serie no están supeditados a un reloj específico en el extremo del receptor, presentando como ventaja que cada carácter sea individual e independiente, o lo que es lo mismo, si tenemos un error en algún carácter, solo habrá que retransmitir dicho carácter, sin afectar ni al anterior ni al posterior.

Cabe también mencionar que, a veces , se inserta entre el último bit de un carácter y el primer bit de parada, un bit llamado *bit de paridad*, el cual tiene como función comprobar la integridad de los datos, o lo que es lo mismo, como un medio sencillo de comprobación de errores.

3.6.2 **Síncrono**

En una transmisión síncrona, los datos se transmiten por el enlace como un flujo continuo de bits (sin utilizar los bits de arranque y parada anteriormente mencionados en la transmisión asíncrona). El funcionamiento de este tipo de transmisión se lleva a cabo sabiendo el instante de tiempo donde deben empezar y terminar los flujos de datos de caracteres. Para que esto sea posible, como se puede intuir, los dispositivos de cara extremo tendrán que utilizar un reloj común, lo que se consigue mediante unos caracteres de sincronización enviados antes de enviar cualquier dato del usuario obligando a que exista la sincronización mencionada entre los extremos antes de que se envíen los datos de usuario.

La especificación de tecnología inalámbrica Bluetooth soporta un enlace síncrono orientado a conexión, enlace que ofrece conexiones punto a punto con conmutación de circuitos que suelen ser utilizados para tráfico de voz, datos y/o multimedia en un ancho de banda reservado, el cual será igual en la dirección de envío como en la de recepción.

4 ARQUITECTURA DEL PROTOCOLO BLUETOOTH

4.1. Protocolos fundamentales de Bluetooth

Los protocolos son la forma consensuada en la que los dispositivos intercambian información, y los fundamentales son aquellos específicos de la tecnología inalámbrica Bluetooth, desarrollados por el SIG Bluetooth, y que son requeridos por la mayoría de los dispositivos que utilizan dicha tecnología [1].

La especificación Bluetooth, incluye los protocolos que permiten el desarrollo de servicios y aplicaciones interactivas que funcionan sobre enlaces inalámbricos establecidos mediante módulos de radio interoperables.

Bluetooth tiene su propia pila de protocolos de cuatro niveles, aunque reutiliza los protocolos existentes en los niveles superiores del modelo OSI, siendo su objetivo, permitir que las aplicaciones escritas de acuerdo con la especificación puedan interoperar entre sí.

Cada aplicación utilizará una pila de protocolos diferente, excepto para los niveles de enlace de datos y físico, que serán comunes.

A continuación se muestra una tabla resumen de los protocolos y niveles de Bluetooth, de los que se explicarán aquellos que sean más relevantes en el objetivo práctico de este proyecto.

Tabla 4-1. Resumen de protocolos y niveles de la pila de protocolos de Bluetooth [1]

Nivel de protocolo Bluetooth	Componentes de la pila de protocolos
Protocolos fundamentales de Bluetooth	·Banda base
	·LMP
	·L2CAP
	·SDP
Protocolo de sustitución de cable	·RFCOMM
Protocolo de control de telefonía	·TCS BIN
	·Comandos AT
Protocolos adoptados	·PPP
	·UDP/TCP/IP
	·OBEX
	·WAP
	·vCard
	·IrMC
	·WAE

4.1.1 **Banda base**

Permite el enlace físico de RF entre unidades Bluetooth dentro de una picorred. Este nivel utiliza procedimientos de averiguación y localización para sincronizar la frecuencia de saltos de transmisión y los relojes de los diferentes dispositivos Bluetooth.

Este nivel proporciona los dos tipos de enlaces físicos (SCO y ACL) ya definidos en el capítulo 4.

4.1.2 **LMP (protocolo gestor de enlace)**

Es el responsable de la configuración y control del enlace entre dispositivos Bluetooth. También se utiliza para la seguridad (en la autenticación y el cifrado) , controla los modos de administración de energía y los ciclos de trabajo y los estados de conexión de una unidad Bluetooth dentro de una picorred.

4.1.3 **L2CAP (protocolo de adaptación y del enlace lógico)**

Soporta la multiplexación de los protocolos de nivel superior, la segmentación y re ensamblado de paquetes, y los mecanismos de calidad de servicio.

4.1.4 **SDP (protocolo de descubrimiento de servicios)**

Permite consultar la información de los dispositivos, los servicios que ofrecen y las características de dichos servicios.

4.2.**Protocolos de sustitución del cable**

Son aquellos protocolos que suministran señalización de control a través de enlaces inalámbricos.

4.3.**Protocolo de control de telefonía**

Protocolo orientado a bit que define la señalización de control de llamada para establecer llamadas de voz y datos entre dispositivos Bluetooth y procedimientos de gestión de movilidad para manejar grupos de

dispositivos TCS (Telephony Control Specification) Bluetooth.

4.4. Protocolos adoptados

Protocolos ya existentes reutilizados para propósitos de niveles superiores, lo que permite que las aplicaciones más antiguas funcionen con la tecnología Bluetooth, asegurando un correcto funcionamiento e interoperabilidad con las aplicaciones más actuales.

4.5. Versiones Bluetooth

Las versiones de la tecnología Bluetooth presentan diferencias entre ellas, entre las que la velocidad de transferencia de datos soportada en cada una es la más importante ya que el aumento de ésta es uno de los objetivos principales por la que se desarrollan nuevas versiones [3],[4],[5],[6].

Los dispositivos Bluetooth también pueden clasificarse por su capacidad de canal como se muestra en la siguiente tabla:

Tabla 4-2 Capacidad del canal según la versión de Bluetooth

Versión	Ancho de banda
Versión 1.2	1Mbps
Versión 2.0 + EDR	3Mbps
Versión 3.0 + HS	24Mbps
Versión 4.0	24Mbps

4.5.1 Bluetooth v1.0 y v1.0B

Estas versiones tuvieron muchos problemas, como el de interoperabilidad entre productos de diferentes fabricantes. A nivel hardware, estas versiones incluyen la dirección del dispositivo Bluetooth en la transmisión, haciendo imposible su anonimato a nivel de protocolo.

Como resultado, la tecnología no era apta para ciertos servicios que esperaban poder contar con esta versión.

4.5.2 Bluetooth v1.1

Está ratificado como estándar IEEE 802.15.1-2002. En esta versión, se corrigieron muchos errores respecto a las versiones anteriormente mencionadas.

En la seguridad se añadió soporte para canales no encriptados. También se añadió el indicador de potencia recibida (RSSI).

Esta versión fue lanzada con el objetivo de que la tecnología Bluetooth se convirtiera en una tecnología simple y en un estándar de fábrica en las comunicaciones inalámbricas.

En esta versión ya se incluye el uso de la banda ISM partida en 79 frecuencias espaciadas unas de otras 1MHz. También se utiliza la técnica de salto en frecuencia para minimizar el espionaje e interferencias de otras redes que usen la banda ISM.

4.5.3 **Bluetooth v1.2**

Esta versión está ratificada como estándar IEEE 802.15.1-2005, siendo las nuevas características adoptadas en esta nueva versión:

- Conexión más rápida.
- AFH (adaptative frequency hopping): frecuencia de salto adaptativo de espectro ampliable.
- Mejora de los enlaces utilizados para transporte lógico.
- Mejoras en la detección de errores y control de flujo.
- Mejora en la capacidad de sincronización.
- Mayor velocidad de transmisión (hasta 721 kbps).
- Mejora de calidad de voz en los enlaces de audio.

4.5.4 **Bluetooth v2.0 + EDR**

Esta versión de la especificación es la principal del estándar Bluetooth, y fue lanzada en 2004.

Además de ser compatible con las versiones anteriores, la mejora más importante que se introduce es una velocidad de datos mejorada (EDR, Enhanced Data Rate) utilizando una combinación de las modulaciones GFSK y PSK con dos variantes $\pi/4$-DQPSK 8DPSK, que permite acelerar la transferencia de datos. La tasa nominal es de 3Mbps, aunque en la práctica se consiguen hasta 2.1 Mbps.

Otra de las ventajas introducidas es la reducción del consumo de energía gracias al uso de un ciclo de trabajo reducido.

4.5.5 **Bluetooth v2.1 + EDR**

La principal mejora que incorpora esta versión es el SSP (Secure Simple Pairing), lo que conlleva a la mejora del emparejamiento entre dispositivos Bluetooth permitiendo que se busquen entre sí y se conecten automáticamente. También incluye el EIR (Extended Inquiry Response), o lo que es lo mismo, la respuesta de consulta extendida, que proporciona más información sobre los dispositivos en el proceso de descubrimiento, mejorando el filtro. Esto reduciría el consumo de energía hasta 5 veces respecto a las versiones anteriores.

4.5.6 **Bluetooth v3.0 + HS**

Esta versión aparece a mediados del año 2009 cuya característica más relevante es que incorpora el soporte de velocidades de transferencia de datos hasta 24 Mbps. Dicha transferencia no es a través del enlace Bluetooth, éste solo se emplea para establecer la conexión.

Si la transferencia de datos es alta se utilizará AMP (alternate MAC/PHY), es decir, empleando wifi, permitiendo el uso de las características Bluetooth consiguiendo un mayor rendimiento en la velocidad de transferencia.

En esta versión también cabe destacar la optimización del consumo de energía, ya que se usa radio de alta velocidad solo cuando es necesario. También, ya que AMP permite descubrir otros dispositivos de alta velocidad solo cuando existe alguna transmisión entre ellos, hay una mejora en la seguridad.

Otras características:

- Mejoras en el L2CAP (protocolo de control y adaptación al enlace lógico).
- Mejoras HCI para AMP.
- PAL (protocol adaption layer).
- Mejora en el control de potencia.

4.5.7 **Bluetooth v4.0**

Esta versión fue adoptada en Junio de 2010 donde aparte de incluir el Bluetooth clásico, la alta velocidad (HS) y los protocolos de bajo consumo, cabe destacar la incorporación del Bluetooth Low Energy (BLE) que se define como un conjunto de protocolos nuevo para conseguir una tecnología de bajo consumo. Como clara desventaja, se puede señalar que el protocolo empleado en BLE no es compatible con los dispositivos que usen el Bluetooth clásico [7].

Un dispositivo que pueda funcionar en el modo dual, puede operar con BLE como con el Bluetooth clásico (de forma simultánea no).

Las características de esta versión son:

- Tres posibles modos de funcionamiento respecto al consumo de energía: pico ultra-bajo, promedio e inactivo.
- Bajo coste.
- Interoperabilidad entre diferentes fabricantes.
- Rango mejorado.

Con esta versión se establecen dos formas para el sistema Bluetooth, detalladas a continuación:

Basic Rate: se ofrecen conexiones síncronas y asíncronas con una tasa de datos desde 721.2 Kbps hasta 2.1 Mbps con el modo HS utilizando AMP.

Low Energy (LE): para aquellos productos que requieren un menor consumo de corriente, coste, tasa de transferencia de datos y ciclo de trabajo, donde se puede poner como ejemplo el sistema en el que se implementará el control remoto a través de Bluetooth más adelante.

4.5.8 **Bluetooth v4.1**

Entre las mejoras que se introducen en esta versión, cabe destacar aquella que permite mejor coexistencia con la nueva conexión móvil 4G LTE ya que la banda de frecuencias Bluetooth se sitúa entre las bandas 40 y 41 de dicha conexión móvil lo que puede ocasionar interferencias permitiendo la corrección con ésta versión [8].

Otras de las características son:

- Reconexiones automáticas cuando un dispositivo sale y vuelve a entrar dentro del campo de alcance de una conexión Bluetooth.
- Mejor transferencia de datos por lotes, orientado a dispositivos wearables.
- Conexión simultánea del dispositivo como receptor y transmisor.
- Soporte extendido de IPv6

Esta actualización permite ser implementada sin apenas tocar el hardware, lo que hace posible que se pueda usar en dispositivos Bluetooth v4.0 mediante actualización del firmware.

Para finalizar este apartado, se resumirá en una tabla comparativa las versiones del estándar Bluetooth anteriormente mencionadas:

Tabla 4-3. Resumen de las versiones del estándar Bluetooth existentes actualmente

Versión	**Principales características**
v1.0	Problemas de interoperabilidad entre dispositivos de diferentes compañías. No permitía el anonimato.
v1.1	Incorporación de la tecnología de salto en frecuencia.

	Redes piconet y scatternet.
v1.2	Conexión más rápida entre dispositivos.
	Incorporación de la tecnología de salto en frecuencia adaptativa.
	Mejora en la detección de errores y capacidad de sincronización.
v2.0 + EDR	Se agrega la tecnología EDR proporcionando una mayor capacidad de transferencia respecto a las versiones anteriores.
v2.1 + EDR	Automatización del emparejamiento entre dispositivos.
	Reporte de datos erróneos, EIR y SSP.
v3.0 + HS	Uso del AMP.
	Mejora de la velocidad.
	Mejora en el control de potencia.
v4.0	Tres modos de consumo de energía.
	Bajo coste.
	Emparejamiento mediante eventos de anuncio.
	Conexión y transmisión mediante eventos de conexión.
v4.1	Conexión y reconexión de dispositivos automáticas.
	Corrección de errores ante interferencias con dispositivos 4G.

Antes de finalizar este apartado, cabe destacar que el dispositivo móvil tiene una versión de Bluetooth v2.1 + EDR y el receptor Bluetooth la versión v2.0 + EDR.

4.6. Otras características de la gestión del enlace

A parte del cifrado y autenticación, mencionados y definidos anteriormente, en la gestión del enlace se dan otras funciones, de las que destacaremos algunas más.

- Modo de retención (Hold): el enlace entre dos dispositivos Bluetooth se puede poner en modo de retención durante un tiempo especificado, en el cual no se podrán transmitir paquetes. Normalmente se entra en este modo cuando no hay necesidad de enviar datos durante un período de tiempo relativamente largo, por lo que se apagará el transceptor para ahorrar energía. Tanto el maestro como el esclavo pueden solicitar entrar en modo de retención.
- Modo de escucha selectiva (Sniff): es otro modo de funcionamiento que permite ahorrar energía. Para entrar en este modo, el maestro y el esclavo negocian un intervalo de escucha selectiva y un desplazamiento de escucha selectiva donde se especifican las franjas de escucha selectiva.
- Modo de aparcamiento (Park): modo en el que se pone un esclavo cuando no es necesario que participe en el canal pero debe mantenerse sincronizado con los saltos de frecuencia.
- Control de potencia: Se utiliza para limitar la potencia de transmisión del dispositivo Bluetooth con el fin de optimizar el consumo de potencia y el nivel global de interferencias. Un dispositivo Bluetooth mide la intensidad de la señal recibida, e informa de si se debe aumentar o disminuir la potencia.
- Ajuste de la velocidad de datos en función de la calidad del canal: los dispositivos Bluetooth pueden configurarse para utilizar paquetes de velocidad media (DM), donde los datos útiles del paquete están protegidos mediante un código de corrección de errores (FEC), para utilizar paquetes de velocidad alta (DH), donde los datos útiles de los paquetes no están protegidos, o para que se ajusten de forma automática.

5 ESPECIFICACIONES Y DISEÑO DEL SISTEMA

Antes de la explicación de cada parte del sistema, es conveniente plantear las especificaciones a cumplir por el sistema que se llevará a cabo, su arquitectura y las fases por las que pasa la parte lógica del sistema (dependiendo de los parámetros y circunstancias en cada momento) mediante diagramas de flujo.

5.1. Especificaciones del sistema

Las especificaciones que tiene que cumplir el sistema que se realizará en los siguientes capítulos son los siguientes:

1. Apertura de puertas de forma cómoda y restringida solo para usuarios del sistema.
2. Identificación personal de cada usuario de manera automática y privada.
3. Envío de datos automatizado, sin tener que introducir ningún tipo de identificación para que el proceso sea lo más rápido y sencillo posible.
4. Conexión inalámbrica con el sistema receptor ocupada el menor tiempo posible.
5. Uso del servicio como nuevo usuario o cancelación del mismo de forma inmediata a su solicitud.
6. Mínimo coste de elementos (físicos y lógicos) utilizados.
7. Servicio sencillo e intuitivo.

5.2. Diseño del sistema

Una vez que se saben los objetivos que tiene que cumplir el sistema, se especifican las diferentes soluciones adoptadas para cumplir dichos objetivos:

5.2.1 Soluciones adoptadas respecto a las especificaciones de ámbito lógico

Para cumplir la especificación 1 se ha propuesto que la identificación del usuario se haga mediante el uso del IMEI (international mobile equipment identify) del dispositivo móvil desde donde se haga uso de una aplicación móvil que permita la apertura de puertas de forma sencilla e intuitiva (especificación 7), ya que es un identificador único de dispositivo asignado por el proveedor con el fin de identificar unívocamente al terminal (especificación 2). No se ha utilizado el número de teléfono por diversos motivos entre los que se puede destacar que ya que algunas compañías no registran el número de teléfono en la tarjeta SIM, no podemos acceder a éste mediante programación en lenguaje Java.

Las especificaciones 3, 4, 5 también son resueltas mediante la aplicación móvil, programándola de forma que cumpla dichos objetivos al igual que se ha hecho con la especificación 1.

Para cumplir con la especificación 6, se han buscado aquellos elementos que permitieran el uso de una red inalámbrica sin coste adicional y que esté disponible para el mayor número de usuarios posible como es Bluetooth, implantado en la gran mayoría de dispositivos móviles permitiendo la transmisión de datos sin contratar algún servicio como sería el caso del intercambio de datos vía internet (se necesitaría conexión GPS, la cual tiene un coste).

5.2.2 Soluciones adoptadas respecto a las especificaciones de ámbito físico

En cuanto a los elementos físicos utilizados se ha procedido de la misma forma, buscando aquellos elementos de menor coste y que cumplan las especificaciones planteadas.

5.3. Arquitectura del sistema

A continuación, de forma previa a la explicación cada parte de la arquitectura del sistema, se muestra una figura donde se pueden observar los 5 elementos del sistema, que serán agrupados en emisor, receptor y base de datos, que harán posible el cumplimiento de las especificaciones planteadas; así como el sentido del intercambio de información entre ellos mediante flechas:

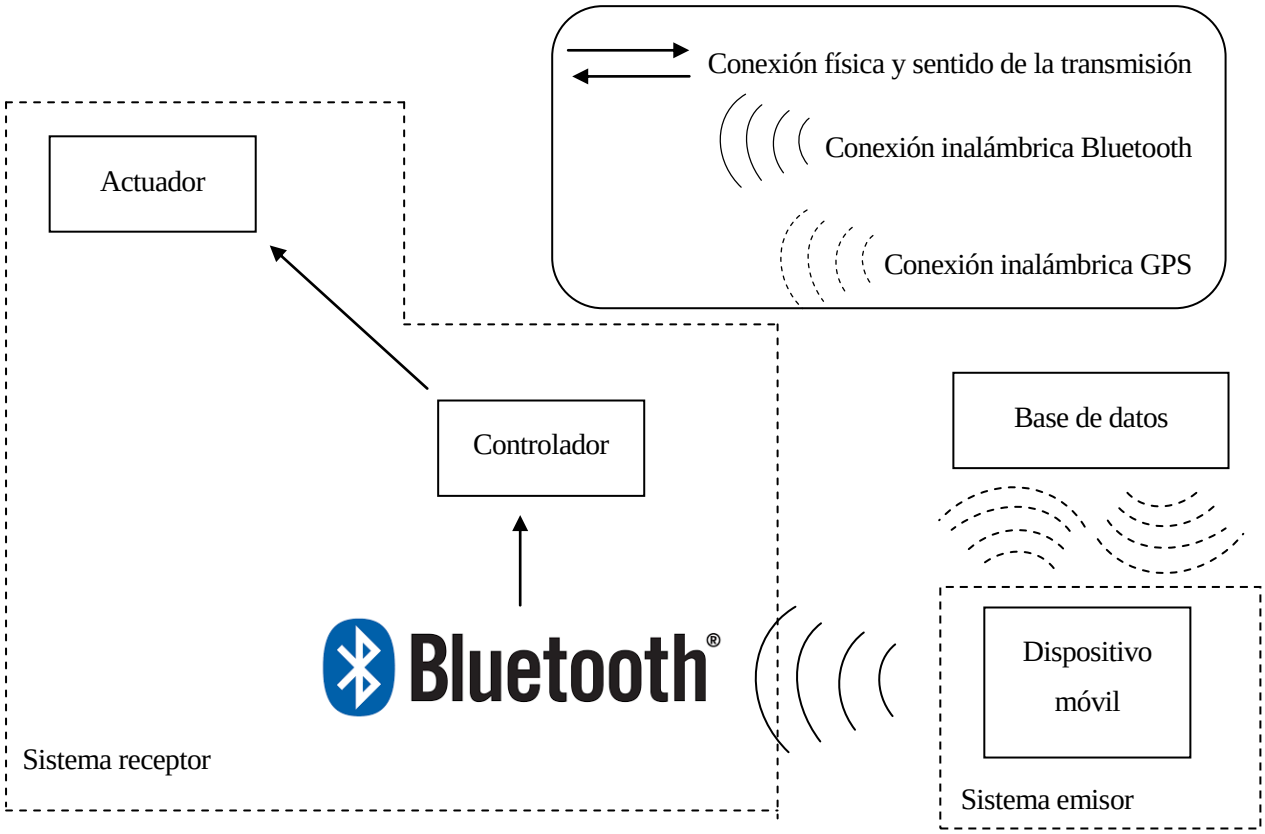


Figura 5-1. Arquitectura general del sistema.

5.3.1 Sistema emisor

Esta parte del sistema está formada por un dispositivo móvil Android, el cual se encargará de generar la interfaz de iteración para el uso del servicio así como de generar, mediante los parámetros de dicho dispositivo, la identificación personal del usuario y enviarla mediante Bluetooth o vía internet según sea el caso en el que se encuentre el usuario. Éstos diferentes casos serán explicados en éste mismo capítulo en el apartado de funcionamiento de los procesos mediante diagramas de flujo.

5.3.2 Sistema receptor

Esta parte del sistema está formada por dos elementos:

5.3.2.1 Receptor de datos vía Bluetooth procedentes del dispositivo móvil

Es la parte del sistema es la encargada de recibir los datos enviados por el dispositivo móvil de forma inalámbrica mediante el protocolo de comunicación Bluetooth, así como de su almacenamiento y posterior envío al controlador definido a continuación.

5.3.2.2 Controlador

Este elemento recibe los datos recogidos y almacenados por el Bluetooth, para actuar de una forma u otra según su programación, es decir, para enviar la orden de apertura de la puerta o seguir en reposo.

5.3.3 Base de datos

Esta parte del sistema se encarga de dar permiso o no, al dispositivo móvil que le realice una consulta pasándole como parámetro su identificación personal. Si dicho parámetro coincide con alguno de los que tiene almacenados, generará una respuesta, sino, generará otra. Esta respuesta será analizada en el dispositivo móvil Android para comportarse de una forma u otra respecto a dicha respuesta.

A continuación, para explicar de forma más detallada el comportamiento de cada elemento, se muestran 3 diagramas de flujo correspondientes a cada uno de los sistemas descritos anteriormente

5.4. Funcionamiento de los procesos

5.4.1 Procesos en el sistema emisor

El siguiente diagrama de flujo muestra los diferentes pasos por los que puede pasar el sistema emisor así como el envío de información a los otros elementos del sistema, o lo que es lo mismo, las diferentes fases por las que puede pasar la aplicación del dispositivo móvil Android respecto a la iteración del usuario y a los demás elementos del sistema:

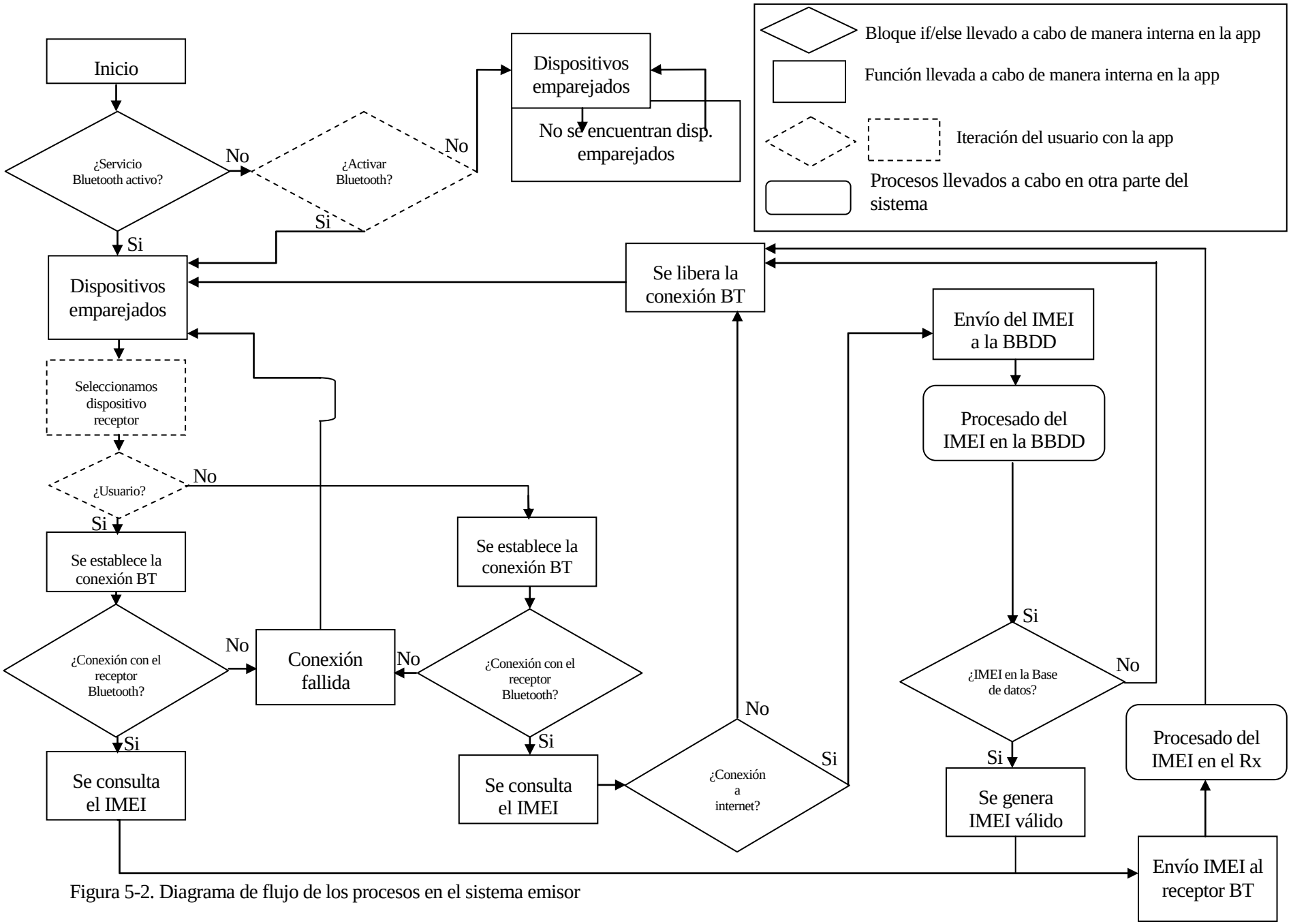


Figura 5-2. Diagrama de flujo de los procesos en el sistema emisor

Se ha decidido actuar de esta forma ya que el sistema receptor no dispone de conexión a internet, por lo que no podrá actualizar los números de identificación de usuario que tenga almacenados de forma diaria, teniendo que proceder de forma manual para llevar a cabo dicha actualización. Esto requiere el coste que conlleva que un técnico actualice el sistema receptor con los números de identificación de usuario de forma periódica, coste que se pretende optimizar haciendo que dicha actualización sea hecha en intervalos de tiempo largos. Por lo que para cumplir la especificación de poder utilizar el servicio de forma inmediata a su contratación, a aquellos usuarios con una antigüedad menor al intervalo de tiempo mencionado, se les identificará como "No usuario" y el sistema, aunque el servicio facilitado será el mismo que para un usuario, actuará de forma diferente, como se detalla más adelante.

A continuación se explican los diferentes caminos que puede tomar el anterior diagrama de flujo:

5.4.1.1 **Usuarios del servicio**

Partiendo de que el dispositivo móvil tiene activado el servicio de Bluetooth y que anteriormente se ha emparejado el dispositivo móvil al Bluetooth del sistema receptor desde el menú de ajustes, la aplicación móvil muestra una lista de los dispositivos Bluetooth emparejados, entre los que el usuario deberá seleccionar el del receptor del sistema, tras lo que se la aplicación móvil procede automáticamente a la conexión con el sistema receptor, como requería una de las especificaciones, comprobando si la conexión está disponible.

Si la conexión con el Bluetooth del sistema receptor está disponible, la aplicación móvil consulta el IMEI y lo envía al sistema receptor donde será procesado, tras lo que libera la conexión Bluetooth y vuelve al estado de reposo donde se muestra la pantalla que permite seleccionar el Bluetooth del receptor del sistema.

Respecto al diagrama anteriormente mostrado, el camino que se ha seguido en este caso ha sido el mostrado a continuación:

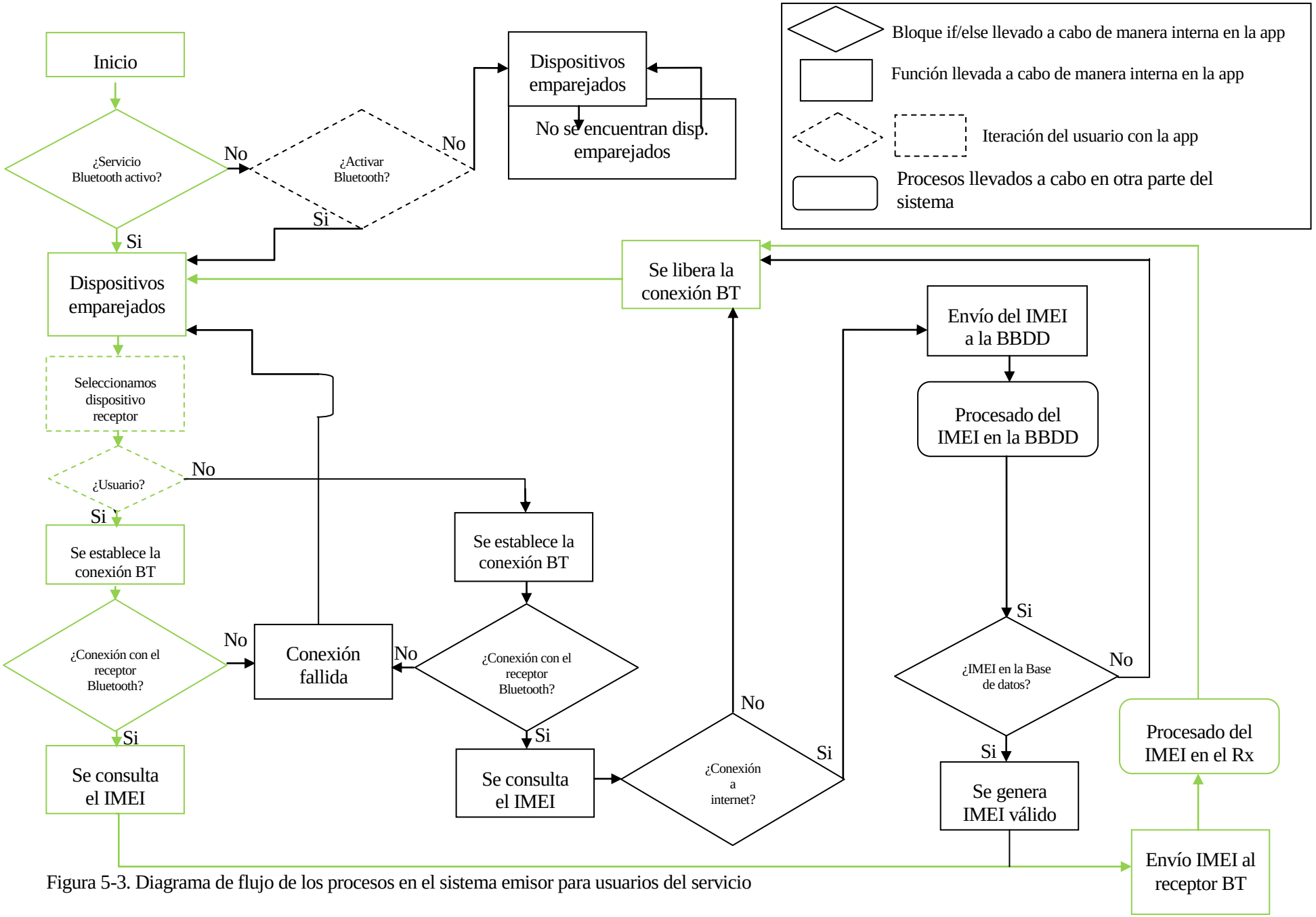


Figura 5-3. Diagrama de flujo de los procesos en el sistema emisor para usuarios del servicio

5.4.1.2 **No Usuarios del servicio**

Suponiendo de nuevo que el dispositivo móvil tiene activado el servicio de Bluetooth y que anteriormente se ha emparejado el dispositivo móvil al dispositivo Bluetooth del sistema receptor desde el menú de ajustes, la aplicación móvil muestra una lista de los dispositivos Bluetooth emparejados, entre los que el usuario deberá seleccionar el del receptor del sistema, tras lo que se la aplicación móvil procede automáticamente a la conexión con el sistema receptor, comprobando si la conexión está disponible.

Si la conexión con el Bluetooth del sistema receptor está disponible y la conexión a internet también, la aplicación móvil consulta el IMEI y lo envía a la base de datos sistema donde se procesará dicho IMEI. Tras esto la base de datos devolverá una respuesta que informe a la aplicación móvil de si dicho IMEI se encuentra registrado en ella. Esta respuesta puede ser de dos tipos, analizado cada uno a continuación:

5.4.1.2.1 No Usuarios del servicio registrados en la base de datos

En este caso la aplicación móvil recibe de la base de datos una respuesta de confirmación de registro, generando un IMEI válido y enviándolo vía Bluetooth al sistema receptor. A continuación se libera la conexión Bluetooth y la aplicación móvil vuelve al estado de reposo donde se muestra la pantalla que permite seleccionar el Bluetooth del receptor del sistema.

Respecto al diagrama principal del sistema, el camino que se ha seguido en este caso es el mostrado a continuación:

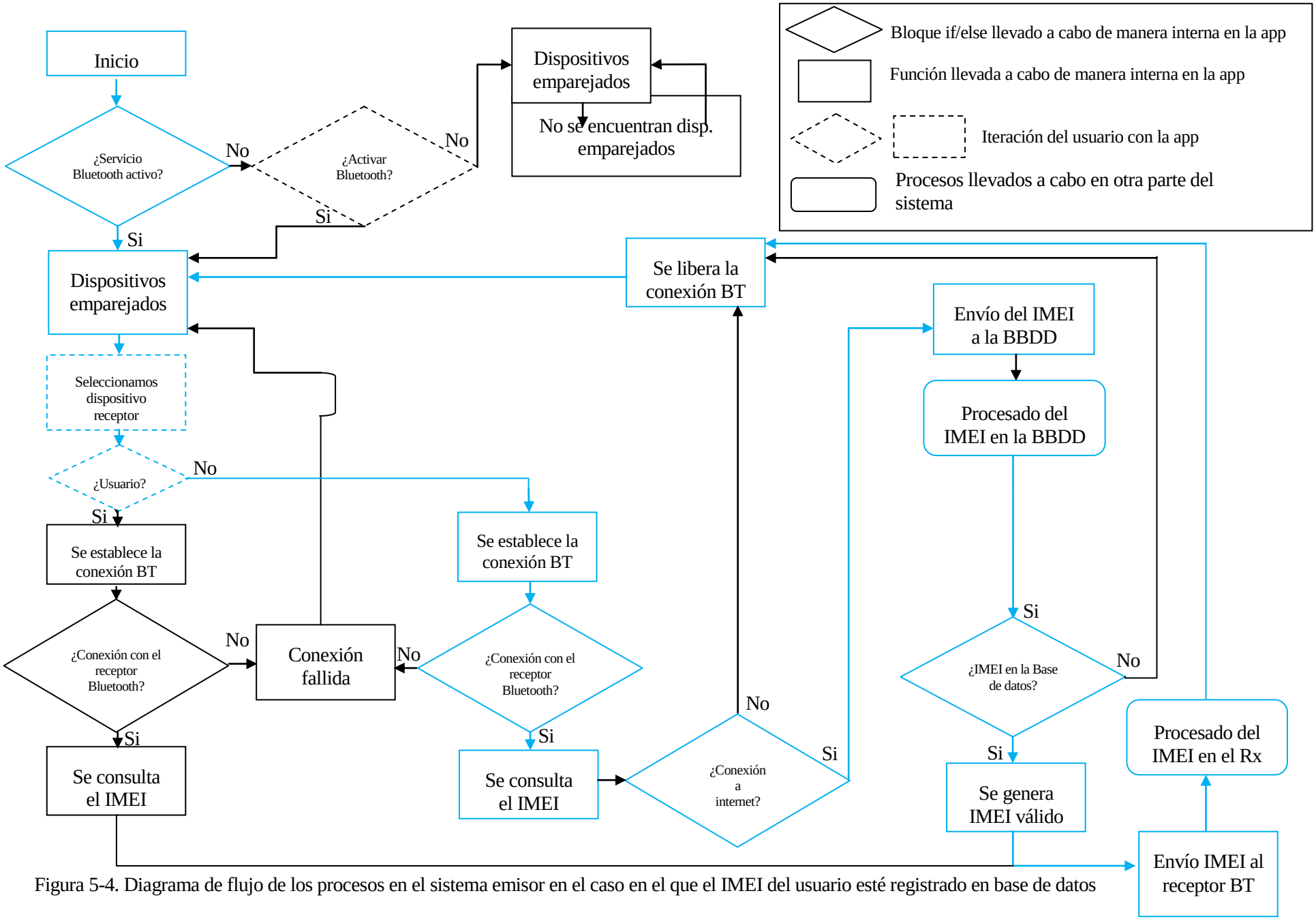


Figura 5-4. Diagrama de flujo de los procesos en el sistema emisor en el caso en el que el IMEI del usuario esté registrado en base de datos

5.4.1.2.2 No Usuarios del servicio no registrados en la base de datos

En este caso la aplicación móvil recibe de la base de datos una respuesta de negación de registro, por lo que la aplicación móvil procede a liberación de la conexión Bluetooth y vuelve al estado de reposo donde se muestra por pantalla la lista que permite seleccionar el Bluetooth del receptor del sistema.

Respecto al diagrama principal del sistema, el camino que se ha seguido en este caso es el mostrado a continuación:

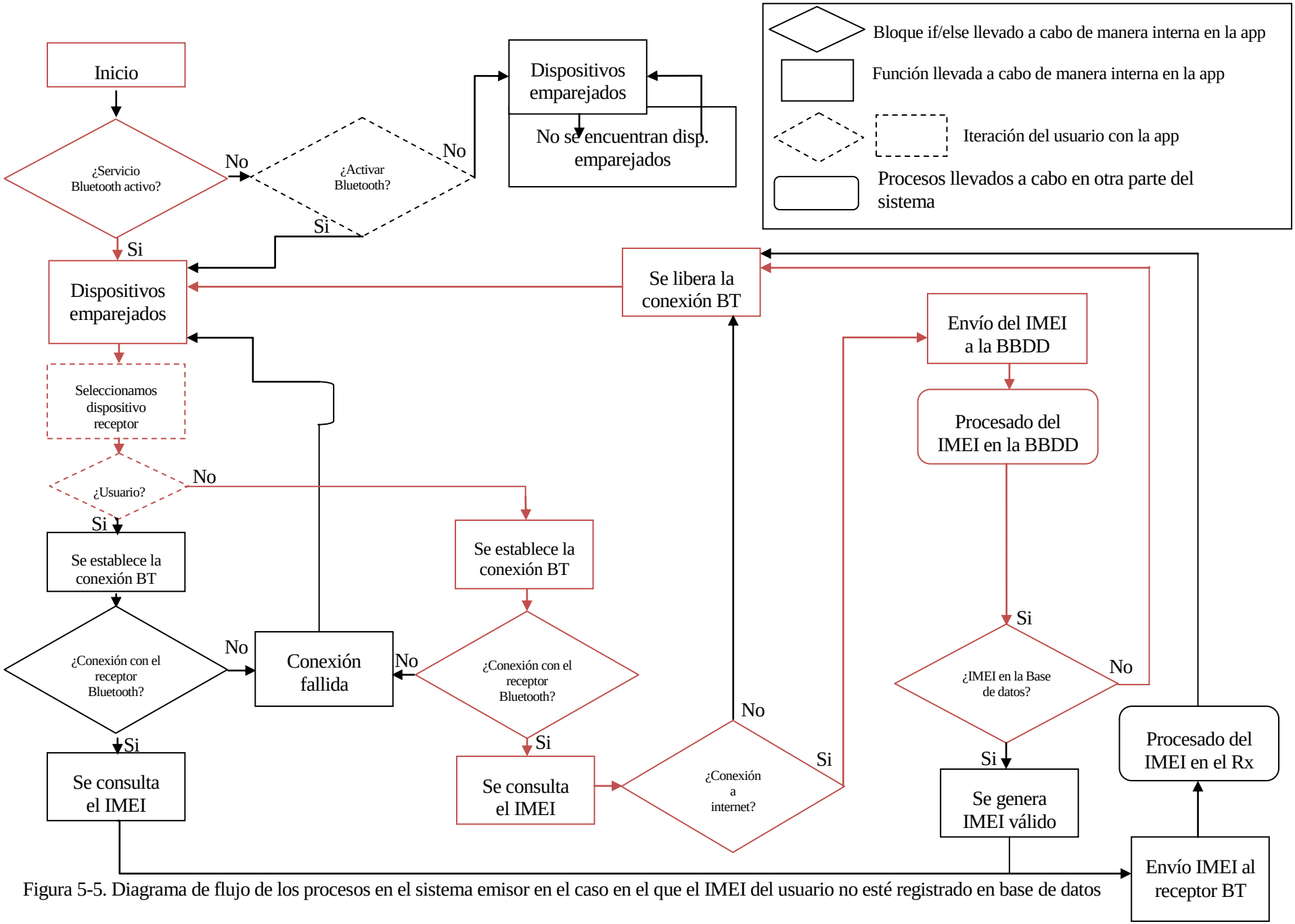


Figura 5-5. Diagrama de flujo de los procesos en el sistema emisor en el caso en el que el IMEI del usuario no esté registrado en base de datos

5.4.1.3 **Conexión a internet no disponible**

En caso de no poder hacer la consulta a la base de datos por no disponer de conexión a internet, se libera la conexión Bluetooth y la aplicación móvil vuelve al estado de reposo donde se muestra la pantalla que permite seleccionar el Bluetooth del receptor del sistema.

Respecto al diagrama principal del sistema, el camino que se ha seguido en este caso es el mostrado a continuación:

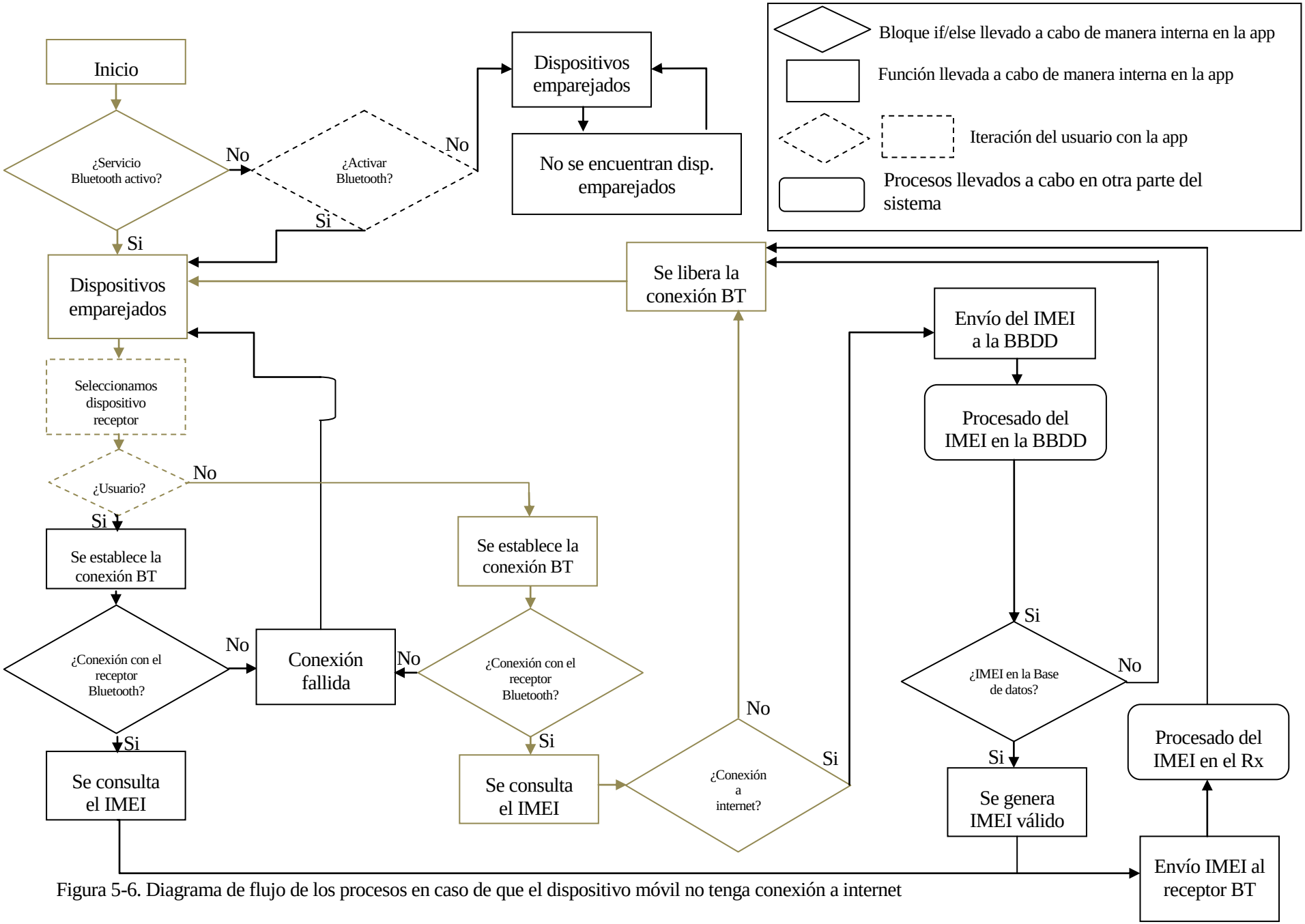


Figura 5-6. Diagrama de flujo de los procesos en caso de que el dispositivo móvil no tenga conexión a internet

5.4.1.4 **Conexión con el receptor Bluetooth fallida**

En cualquiera de los casos anteriores, si la conexión con el receptor Bluetooth resultara fallida, la aplicación móvil no pasaría a consultar el IMEI del dispositivo, sino que directamente volvería a mostrar por pantalla el menú de selección de dispositivos emparejados.

Respecto al diagrama principal del sistema, el camino que se ha seguido en este caso es el mostrado a continuación:

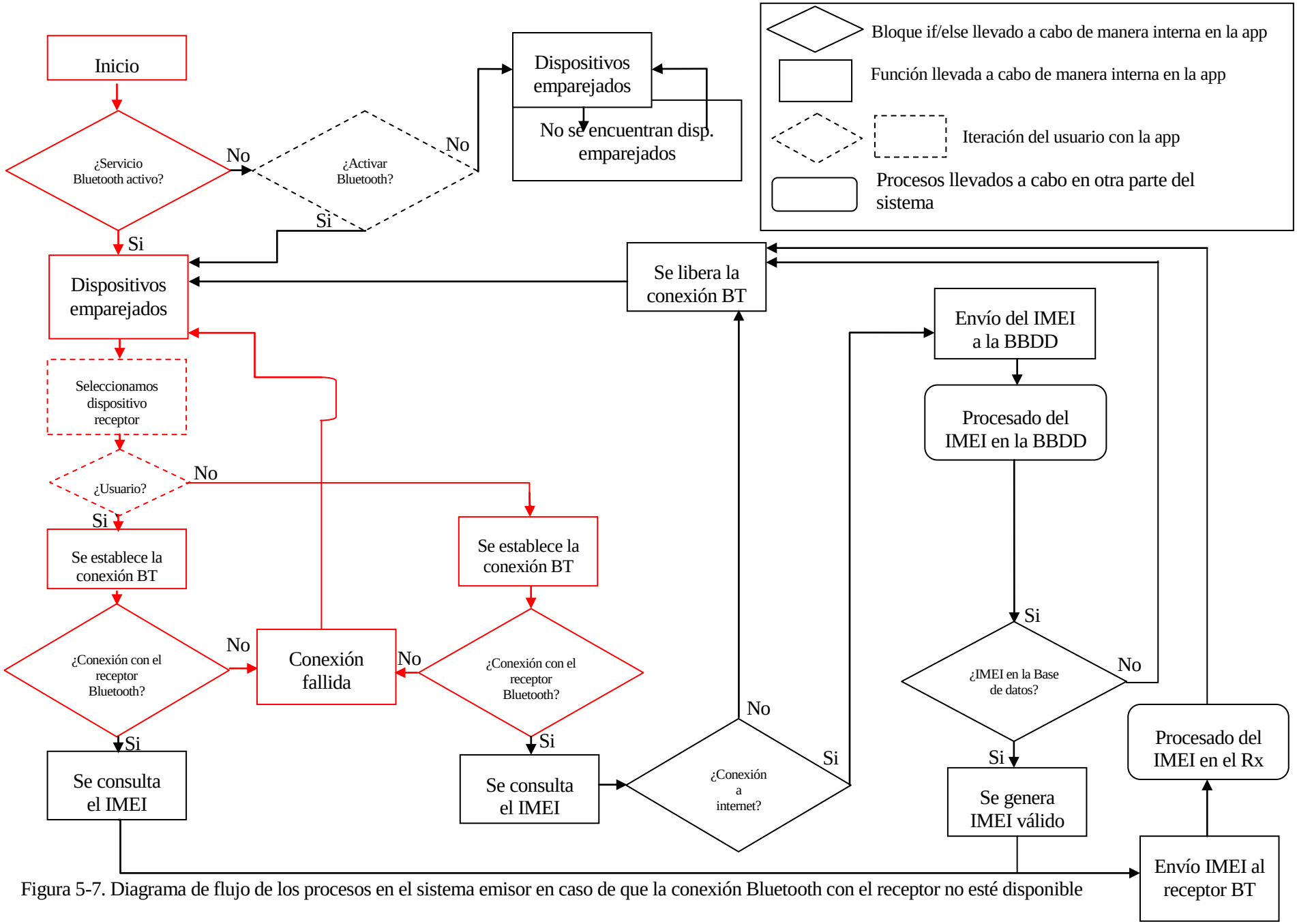


Figura 5-7. Diagrama de flujo de los procesos en el sistema emisor en caso de que la conexión Bluetooth con el receptor no esté disponible

5.4.1.5 **Servicio Bluetooth desactivado**

En caso de no tener el servicio de Bluetooth del dispositivo móvil activado, la aplicación móvil mostraría por pantalla la opción de activarlo.

En caso de confirmar la activación de dicha petición, el comportamiento de la aplicación seguiría uno de los caminos anteriormente detallados.

De lo contrario, si se niega dicha petición, la aplicación móvil también mostraría por pantalla el menú para seleccionar algún dispositivo emparejado, pero dicha lista aparecerá vacía por no tener activado el servicio Bluetooth.

Respecto al diagrama principal del sistema, el camino que se ha seguido en este caso es el mostrado a continuación:

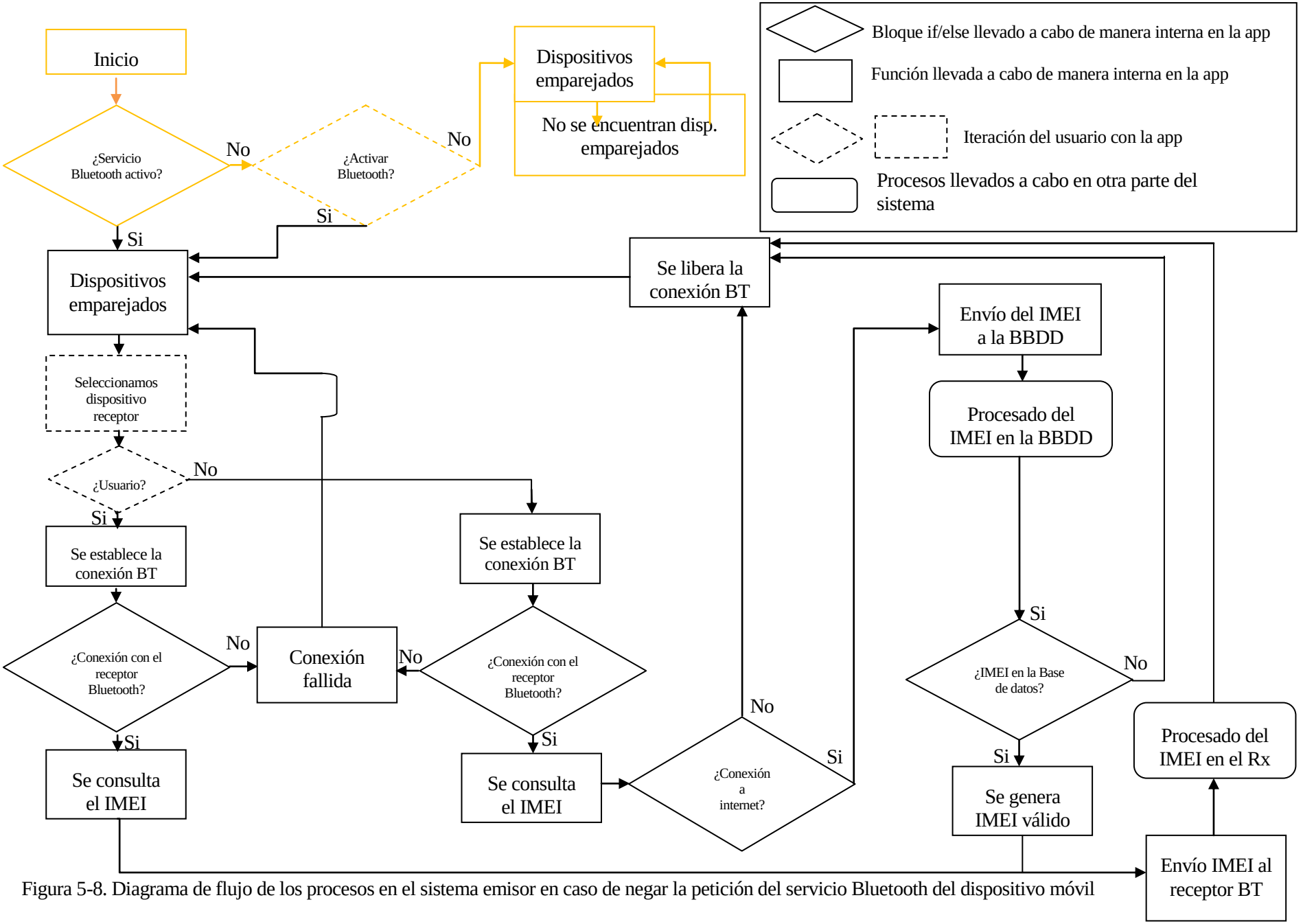


Figura 5-8. Diagrama de flujo de los procesos en el sistema emisor en caso de negar la petición del servicio Bluetooth del dispositivo móvil

En los casos en los que se envía el IMEI al receptor, el procedimiento que sigue éste será explicado en el apartado detallado a continuación:

5.4.2 **Procesos en el sistema receptor**

Los procesos descritos en este apartado son los que tienen lugar en el receptor de datos procedentes de la comunicación vía Bluetooth con el dispositivo móvil, como los que tienen lugar en el controlador.

El dispositivo receptor de datos vía Bluetooth se encuentra conectado al controlador, mediante el que ha sido programado para tener el siguiente comportamiento: en estado de reposo se encuentra en modo de escucha, esto es en espera a que algún dispositivo con el que se ha emparejado anteriormente empiece a transmitirle datos. Una vez que comienza dicha transmisión, el dispositivo receptor deja de estar en modo de escucha y empieza a recibir datos que transfiere al controlador y que éste va almacenando. Una vez que la recepción de datos ha finalizado, el controlador compara todos los dígitos recibidos como una sola cadena de caracteres con otras cadenas de caracteres que tiene almacenadas, actuando de una forma u otra respecto a la existencia de coincidencia entre los dígitos recibidos y los almacenados. Estos casos serán analizados a continuación.

La lista de cadenas con las que se hacen las comparaciones han sido introducidas en el controlador de forma manual y física es decir, por cable, ya que como anteriormente se ha mencionado no puede realizarse de forma remota al no disponer dicho controlador de conexión a internet.

En el caso de que coincida alguna de las cadenas de caracteres almacenadas en el controlador con la que recibe el dispositivo Bluetooth, el controlador enviará al sistema de apertura una orden para que se ponga en funcionamiento. De lo contrario, el sistema no realizará ninguna acción.

Sea cual sea el resultado, tras realizar la comparación, se reinicializa la variable donde se almacenaron los dígitos recibidos para que otro usuario pueda hacer uso del sistema de forma inmediata.

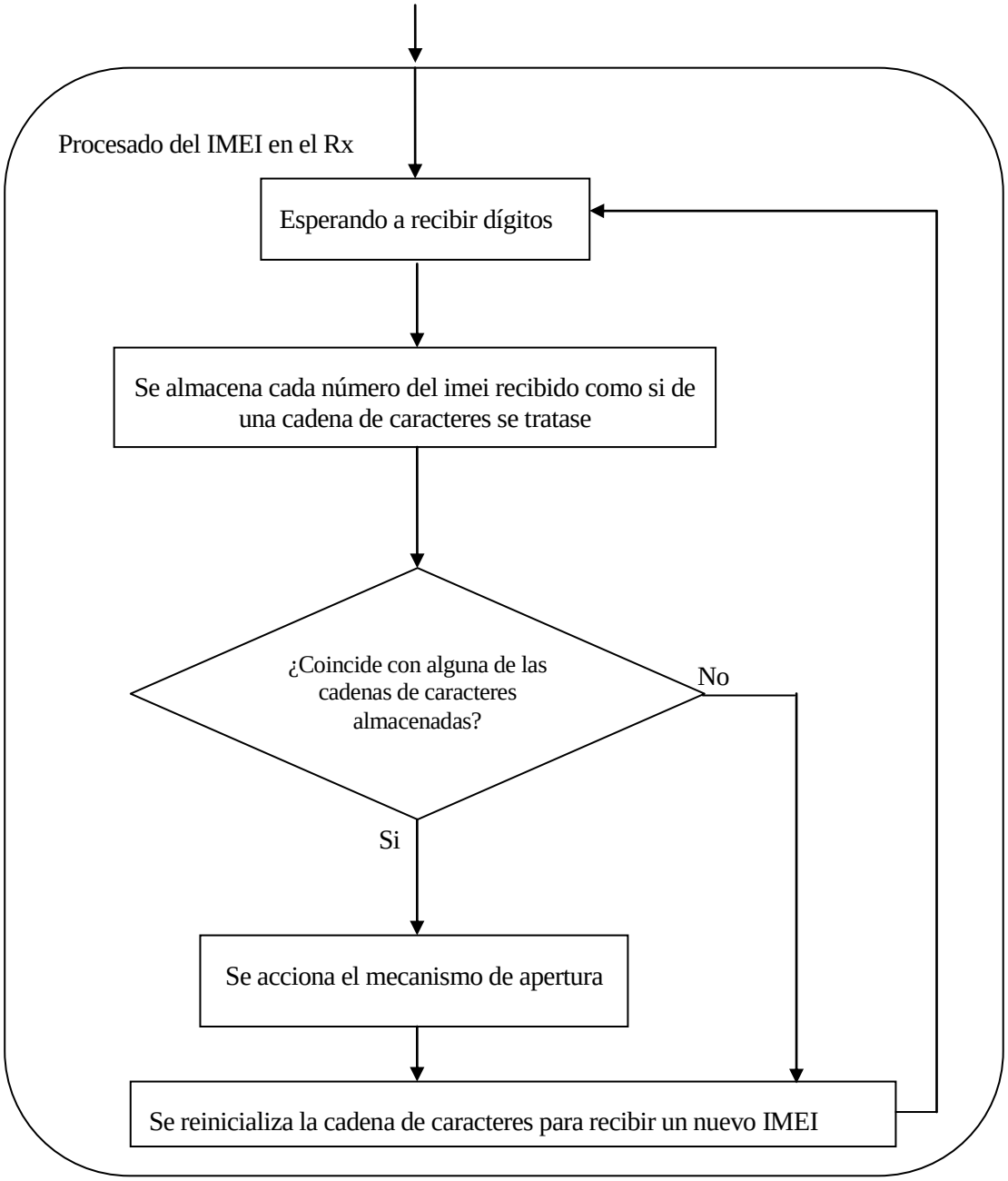


Figura 5-9. Diagrama de flujo los procesos en el sistema receptor

5.4.3 **Procesos en la Base de datos**

El proceso descrito en este apartado es el que tiene lugar tras darse la petición de consulta en el dispositivo móvil, el que le enviará a través de una conexión a internet el IMEI.

Una vez que el servidor recibe la variable con el IMEI, éste busca en una tabla almacenada en una base de datos si existe alguna cadena de caracteres idéntica a la recibida por el dispositivo móvil.

Si existiera dicha cadena de caracteres, se generará una respuesta que indicara dicha coincidencia. De lo contrario, si no existiera, se generará otra respuesta indicando este caso.

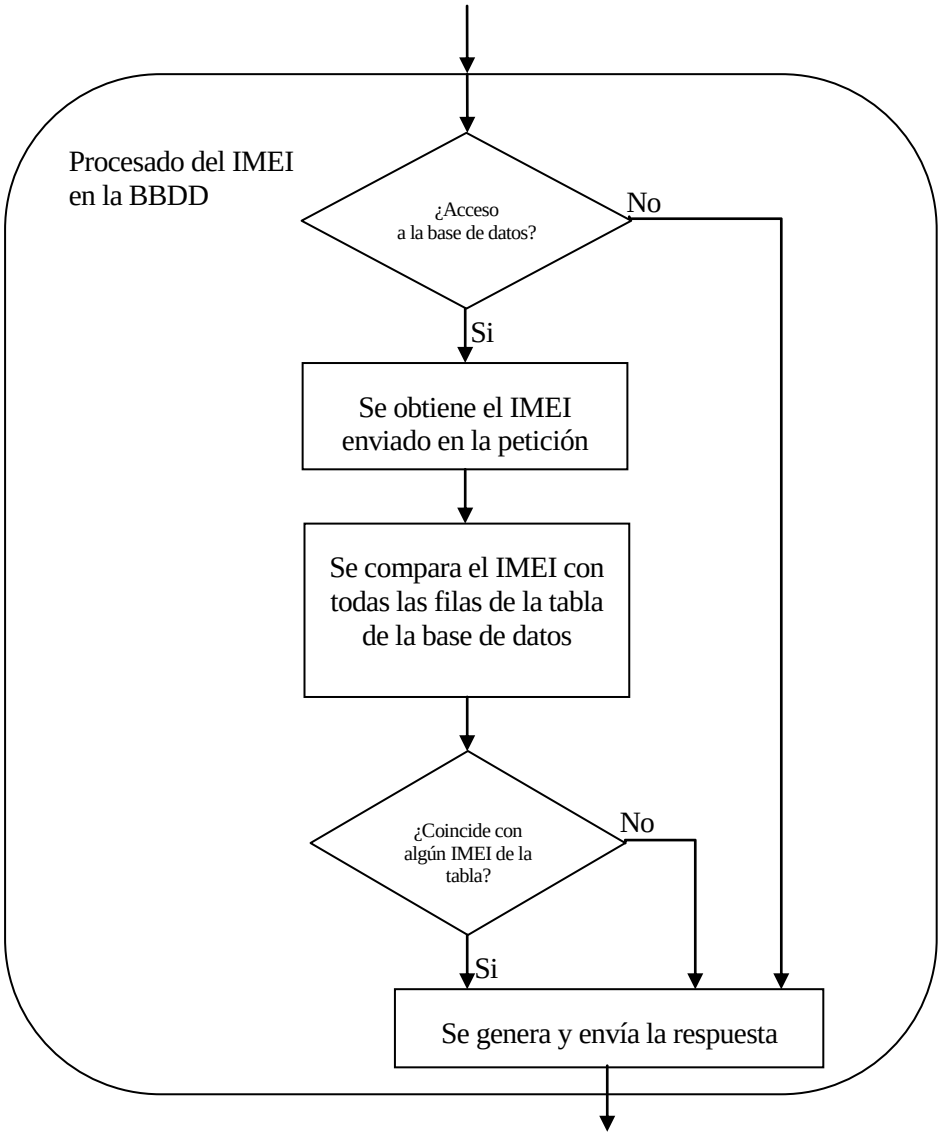


Figura 5-10. Diagrama de flujo de los procesos en la base de datos

La respuesta es devuelta al dispositivo móvil que realizó la consulta, analizando dicha respuesta y actuando en consecuencia a ella, como se ha explicado en el apartado PROCESOS EN EL SISTEMA EMISOR.

En el siguiente capítulo se describirán las herramientas utilizadas para la parte lógica del sistema así como los dispositivos físicos para su implantación.

6 HERRAMIENTAS Y DISPOSITIVOS UTILIZADOS

En este capítulo se describirán todas las herramientas que se han usado para la realización del sistema, tanto a nivel lógico como físico.

6.1. Nivel lógico

A nivel lógico, se especifican los lenguajes de programación utilizados para el desarrollo de la aplicación móvil y la configuración del acceso al servidor creado para realizar las consultas vía internet.

6.1.1 Aplicación móvil

Para la aplicación móvil se han utilizado:

- XML (eXtensible Markup Language): se trata de un metalenguaje, es decir, un lenguaje utilizado para decir algo de otro, o mejor dicho, un lenguaje que nos permite la organización y etiquetado de otros documentos, definiendo los lenguajes necesarios según las necesidades.
- Java: utilizado para el desarrollo de la aplicación (clases, paquetes...). Se define como el lenguaje de programación orientado a objetos que tiene las siguientes características:
    - Flexibilidad a la hora de reutilizar código.
    - Funcionamiento en cualquier plataforma (dado que el código es ejecutado por la máquina virtual y no por el sistema).
    - No se necesita licencia para programar con él.
    - Fuente abierta de sus librerías nativas.
    - Lenguaje expandible.

Figura 6-1. Logo Java.

Estos dos tipos de lenguajes serán utilizados en el desarrollo de la aplicación móvil Android, como se puede ver en el anexo A.1 (XML) y Anexo A.2 (Java).

- JSON (JavaScript Object Notation): este tipo de lenguaje se ha utilizado para el intercambio de datos entre la aplicación móvil Android y el servidor web. Se puede definir como un lenguaje de programación basado en la sintaxis de JavaScript que se utiliza para transmitir a través de internet una gran cantidad de bytes de información.

El uso de este lenguaje puede verse en el anexo A.3.

Figura 6-2. Logo JSON.

Para el intercambio de datos con la base de datos creada se utilizará un servicio web escrito en php (hypertext pre-processor) cuya definición podría ser la siguiente: lenguaje de código abierto adecuado para el desarrollo web con posibilidad de ser incrustado en HTML, de software libre. El código php se procesa directamente en el servidor o a través de un software que permita ejecutar comandos.

La característica más importante de PHP es que permite modificar dinámicamente el contenido de una página actualizando la base de datos, característica esencial para el sistema que se va realizar en el que se almacenará l la identificación personal de cada usuario (IMEI del dispositivo móvil) en la base de datos.

Figura 6-3. Logo php.

En el anexo B se puede observar el código utilizado en éste lenguaje.

Para el desarrollo de la aplicación móvil se ha utilizado el software gratuito "Android Studio 1.0", el cual permite la programación de aplicaciones para dispositivos Android mediante un entorno de desarrollo integrado (IDE). Sus principales características principales son:

- Soporte para programar aplicaciones para Android wear (sistema operativo para dispositivos corporales).
- Herramientas Lint (detecta códigos no compatibles entre arquitecturas o código confuso que no es capaz de controlar el compilador).

Figura 6-4. Logo del software Android Studio.

Más adelante, en el capítulo donde se explica el funcionamiento de la aplicación móvil, se pueden observar diferentes capturas de pantalla utilizando dicho software.

Se ha utilizado este software por la gran variedad de características que incluye, haciéndolo el mejor para el desarrollo de una aplicación móvil siendo el único requisito indispensable saber programar en Java. Para profundizar en dichas características, se puede visitar el siguiente enlace: http://academiaandroid.com/android-studio-v1-caracteristicas-comparativa-eclipse/

6.1.2 Sistema receptor de órdenes Bluetooth

Para el sistema receptor de las órdenes emitidas desde el dispositivo móvil, se utilizará un dispositivo Bluetooth conectado a una plataforma de hardware y software libre conocida como Arduino uno, definido más adelante, la cual estará programada de forma que según los datos recibidos en el Bluetooth, realice una acción u otra [10], [11], [12], [13], [14], [15].

El lenguaje en el que está programada dicha plataforma es C++ definido de la siguiente forma:

C++ es un lenguaje de programación diseñado como extensión del lenguaje C. Este lenguaje abarca la programación estructurada, la programación genérica y la programación orientada a objetos. Sus principales características son:

- Facilidades para la programación orientada a objetos.
- Uso de plantillas o programación genérica (templates).
- Posibilidad de redefinir operadores.
- Identificación de tipos en tiempo de ejecución.
- Permite trabajar a bajo y alto nivel.

En el anexo C se puede observar el código de programación ejecutado en el controlador Arduino uno en éste lenguaje.

6.1.3 Base de datos alojada en servidor web

Para la realización de la base de datos en internet, se ha seguido un tutorial de un blog de internet en el que se hacía una base de datos similar pero con el mismo objetivo.

Pinchando en el siguiente enlace se accede a dicho blog: http://cursoandroidstudio.blogspot.com.es/

Dicho blog utiliza un sitio web donde te permiten realizar un hosting web con php y MySQL (My Structured Query Langugage, Lenguaje de consulta estructurada) tan solo creando una cuenta gratuita. Para acceder a dicho sitio web pinchar en el siguiente enlace: http://www.hostinger.es/

Figura 6-5. Logo Hostinger.

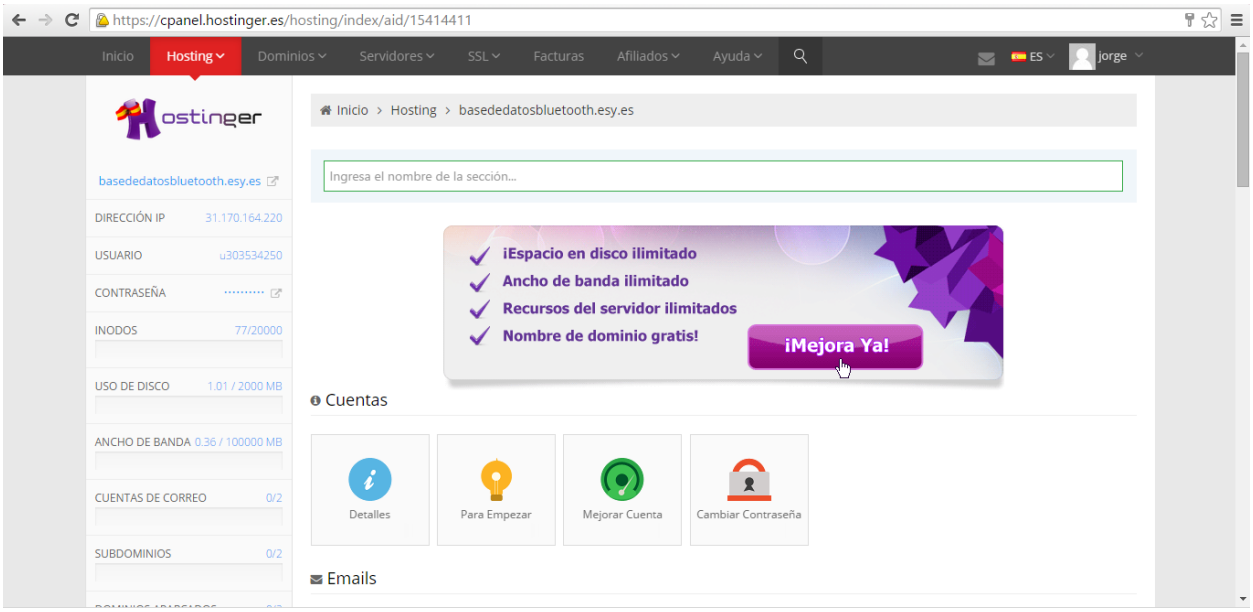


Figura 6-6. Vista previa del sitio web donde se ha realizado la base de datos

El motivo por el que se ha realizado la base de datos de esta forma online es porque no requiere la programación de la dicha base de datos, simplemente hay que seguir los pasos para especificar la forma que tiene que tener la tabla donde se van a almacenar los datos, sin tener que programar nada.

6.2. **Nivel físico**

A nivel físico, las herramientas utilizadas han sido:

- Dispositivo móvil (Samsung Galaxy Mini Plus GT-s5570i) proporcionado por el tutor del proyecto.

  Para ver las características del dispositivo móvil, pinchar en el siguiente enlace:

  http://www.movilcelular.es/movil/samsung-galaxy-mini-plus-gt-s5570i/406

Figura 6-7. Dispositivo móvil Samsung Galaxy Mini Plus GT-s5570i.

Se ha utilizado este dispositivo móvil ya que ha sido el facilitado por el tutor de este proyecto para poder hacer las pruebas detalladas en el siguiente capítulo. Aunque sea un dispositivo móvil del año 2011, cuenta con las características necesarias para actuar como emisor del sistema desarrollado en este proyecto. Al tener cuatro años de antigüedad, no dispone de las versiones Android o Bluetooth mejoradas que existen actualmente pero la aplicación funciona de la misma forma que en un dispositivo actual. Con esto se pretende decir que, siendo éste un dispositivo "antiguo" respecto a los actuales, el sistema funciona cumpliendo las especificaciones requeridas, por lo que en un dispositivo actual también funcionara de la misma forma o mejor por sus características mejoradas respecto al sistema operativo o a la versión Bluetooth que disponga.

- Plataforma de hardware y software libre (Arduino uno) proporcionada por el departamento con todas las demás herramientas para proceder con la conexión de diferentes elementos electrónicos a dicha plataforma: para ver sus características pinchar en el siguiente enlace:

  https://www.arduino.cc/en/Main/arduinoBoardUno

Figura 6-8. Componentes Arduino Starterkit.

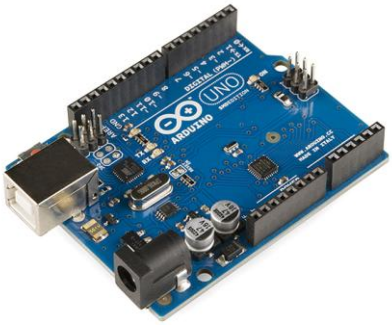


Figura 6-9. Plataforma Arduino uno.

Existen muchos tipos de plataformas con microcontroladores y se ha utilizado ésta, aparte de por ser la facilitada por el departamento, por tener un precio asequible (menos de 60€), por funcionar en los sistemas operativos Windows, Macintosh OSX y Linux o por su sencillo entorno de programación. Hay más características que hacen que esta plataforma sea de las mejores para cumplir las especificaciones de éste proyecto aunque solo se hacen referencia a las mencionadas como ejemplo.

- Módulo Bluetooth (HC-05 FC-114) obtenido a través de tienda de electrónica online, cuyas características principales son [16], [17]:
  - Chipset CSR BC417143
  - Bluetooth versión V2.0+EDR

- Tensión de alimentación: 3.3V
- Frecuencia: 2.4GHz banda ISM
- Modulación: GFSK (Gaussian Frequency Shift Keying)
- Seguridad: Autentificación y encriptación.
- Velocidad: Asíncrono: 2.1Mbps (Max) / 160 kbps ; Síncrono: 1Mbps/1Mbps
- Soporta comandos AT para configuración a través de un puerto serie.
- Configuración por defecto para el puerto COM: 9600, N, 8,1
- Temperatura de trabajo: -20 ºC a +75 ºC
- Dimensiones: 26.9mm x 13mm x 2.2 mm

Para obtener más información sobre dicho módulo, consultar el data sheet mediante el siguiente enlace: http://www.electronica60norte.com/mwfls/pdf/newBluetooth.pdf

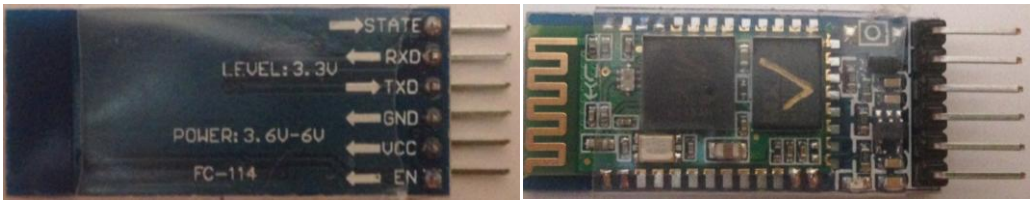


Figura 6-10. Módulo Bluetooth (HC-05 FC-114).

Se ha utilizado este dispositivo Bluetooth para la recepción de los datos provenientes del dispositivo móvil por ser uno de los más utilizados en proyectos con objetivos similares a éste, lo que lo hace especialmente interesante a la hora de buscar información para hacer uso de él ya que existen bastantes páginas de internet donde los usuarios explican todo tipo de detalles sobre sus características y funcionamiento. También cabe mencionar que otro de los motivos por los que se ha utilizado este dispositivo Bluetooth es por su bajo precio de compra.

7 DESARROLLO DEL SISTEMA

Una vez explicadas las especificaciones que debe cumplir el sistema, la arquitectura que tendrá y las herramientas utilizadas en dicha arquitectura para cumplir las especificaciones, en este capítulo se explica la puesta a punto del sistema creado, mostrando mediante imágenes las diferentes fases por las que pasará cada uno de los elementos de dicho sistema y que fueron explicadas en el capítulo 5.

Antes de explicar las posibles fases que se pueden dar en el sistema emisor, se explicarán otros aspectos previos al uso de la aplicación en el dispositivo, como la generación del archivo de instalación de la aplicación realizada en el software mencionado en el capítulo anterior y la instalación de dicho archivo en el dispositivo móvil, el cual nos permitirá hacer uso de dicha aplicación.

7.1. Programación y generación del archivo de instalación de la app

En la siguiente captura de pantalla se muestra una captura de dicho software en la cual se puede ver el "explorador de proyectos" del software de programación Android Studio con los diferentes archivos que forman la app creada.

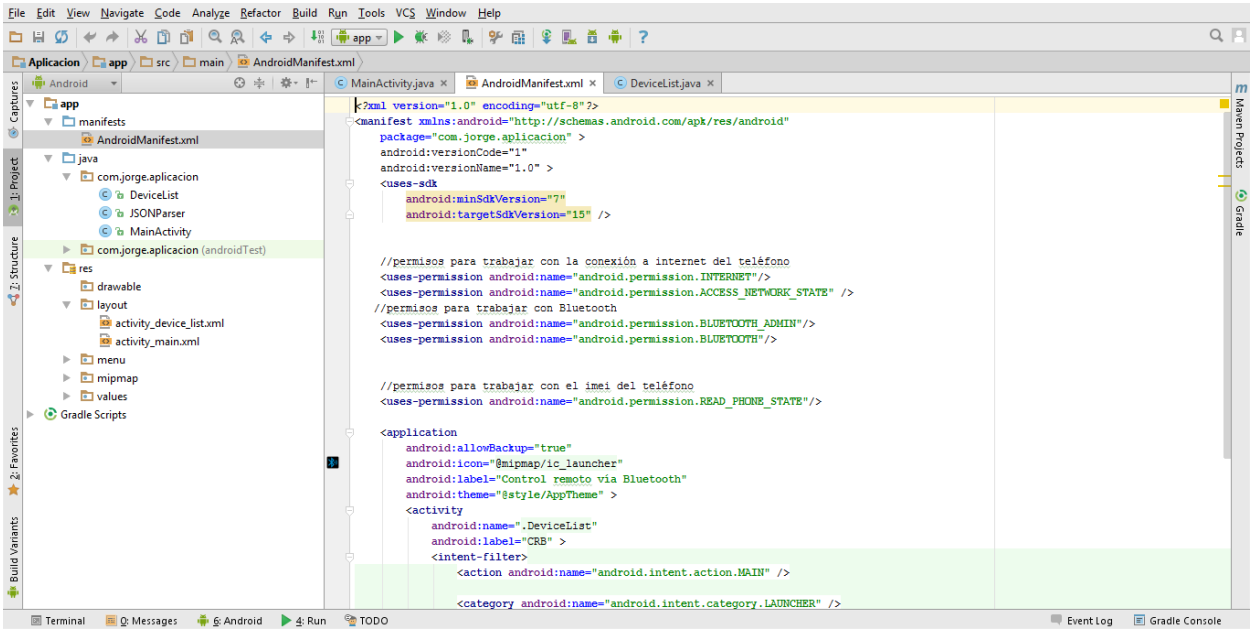
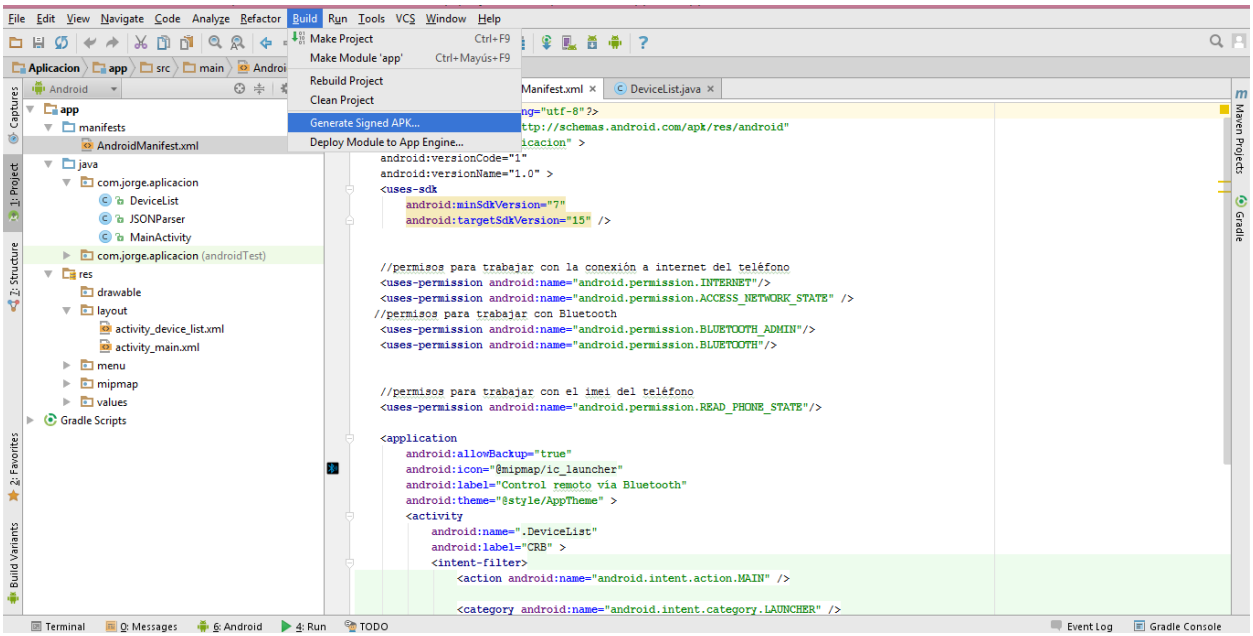


Figura 7-1. Captura de pantalla del software Android Studio.

Para generar el archivo de instalación de la apk, en el software Android Studio: Build>generate singed apk, y tras especificar algunos datos como la ubicación o la contraseña que tendrá, se genera dicho archivo como se muestra en las siguientes figuras:



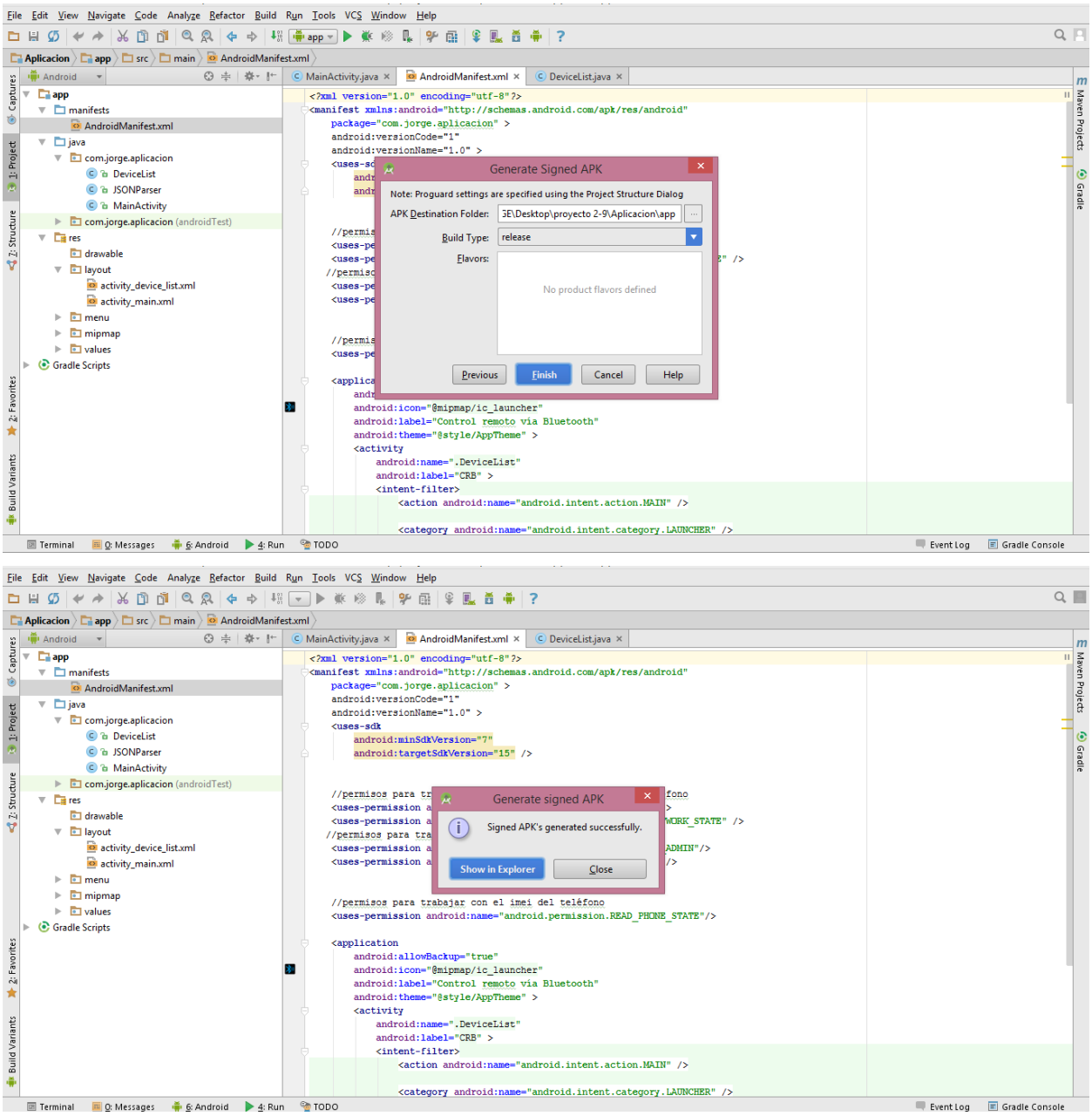


Figura 7-2. Proceso de generación del archivo de instalación de la aplicación en el dispositivo móvil.

Una vez obtenido el archivo de la app, se copia en el dispositivo móvil (conectando éste al pc donde se generado dicho archivo) mediante el cable para la transferencia de archivos del dispositivo móvil.

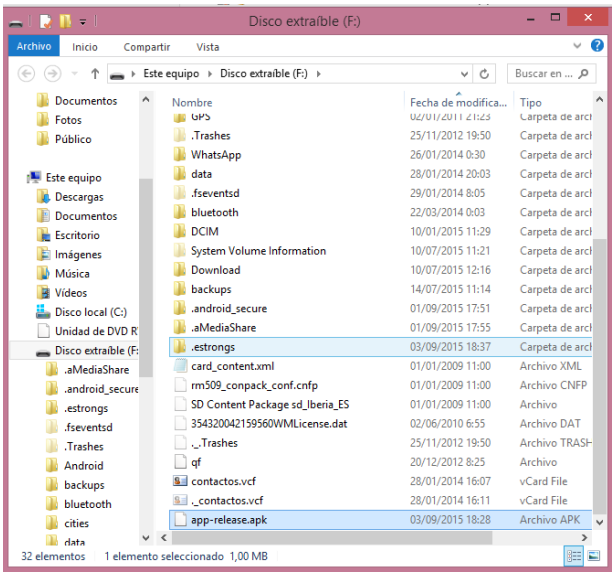
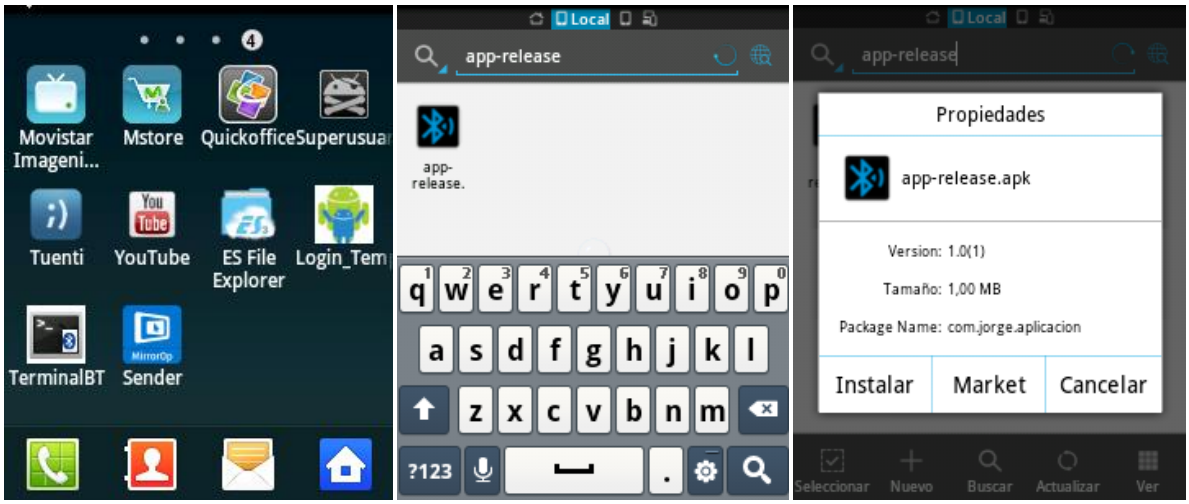


Figura 7-3. Transferencia del archivo de instalación de la app al dispositivo móvil.

7.2. **Instalación de la app en el dispositivo móvil**

Una vez se tenga el archivo de instalación de la aplicación en el dispositivo móvil en el que se llevará a cabo el uso del servicio propuesto en este proyecto, para agilizar el proceso, se puede buscar dicho archivo con una aplicación de gestor de archivos.

Hay gran variedad en cuanto a aplicaciones de gestor de archivos se refiere, en este proyecto se ha utilizado "ES File Explorer" como se puede ver en las siguiente figuras, así como la búsqueda del archivo de la app, su instalación y aspecto final en el dispositivo móvil:



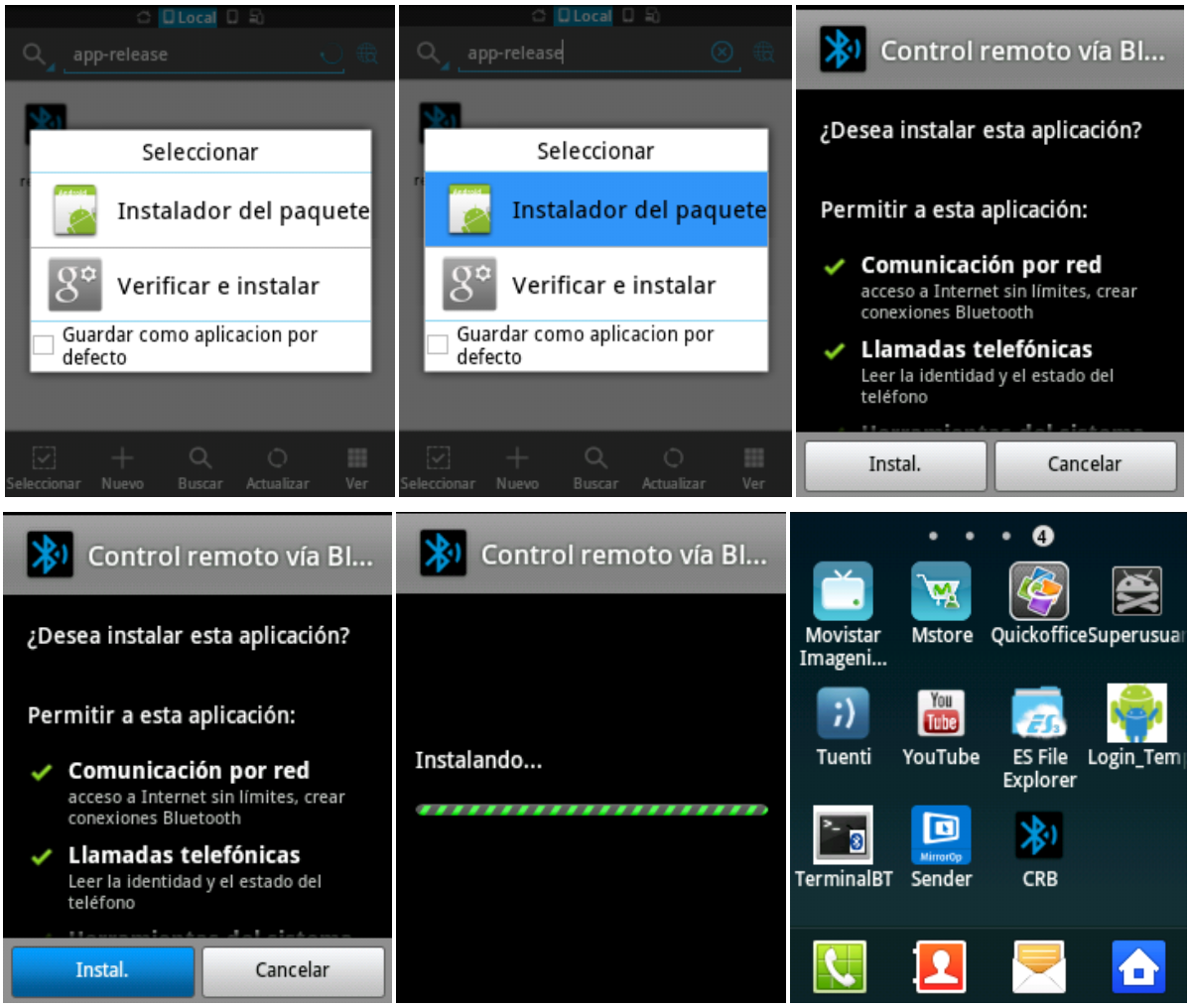


Figura 7-4. Búsqueda del archivo de instalación de la aplicación y dicha instalación en el dispositivo móvil.

Respecto al gestor de archivos mencionado, si se pudiera descargar el archivo de instalación directamente desde el buscador de aplicaciones "Play Store", lo que conlleva un coste económico, no se tendría que utilizar dicha aplicación, como cualquier caso de instalación de aplicación Android que se descarga desde el buscador "Play Store" existente en todos los dispositivos móviles Android. Esta mejora se mencionará en el CAPITULO 8 de este mismo documento.

7.3. Emparejamiento entre el dispositivo móvil y el dispositivo receptor Bluetooth del sistema

Antes de acceder a la aplicación móvil, para que ésta de la opción de seleccionar el dispositivo Bluetooth receptor del sistema de control remoto, se habrá tenido que emparejar el dispositivo móvil dónde tengamos la app con dicho sistema receptor.

A la hora de emparejar los dispositivos, tras seleccionar el dispositivo receptor (en nuestro caso el dispositivo se llama HC-05), se tendrá que introducir la clave que requiere el emparejamiento de dispositivos (en nuestro caso es 1234) para que el emparejamiento quede memorizado y los dispositivos se reconozcan.

Todo esto se observar en las siguientes imágenes:

Figura 7-5. Vista previa de la aplicación antes de vincular el dispositivo móvil con el receptor Bluetooth.

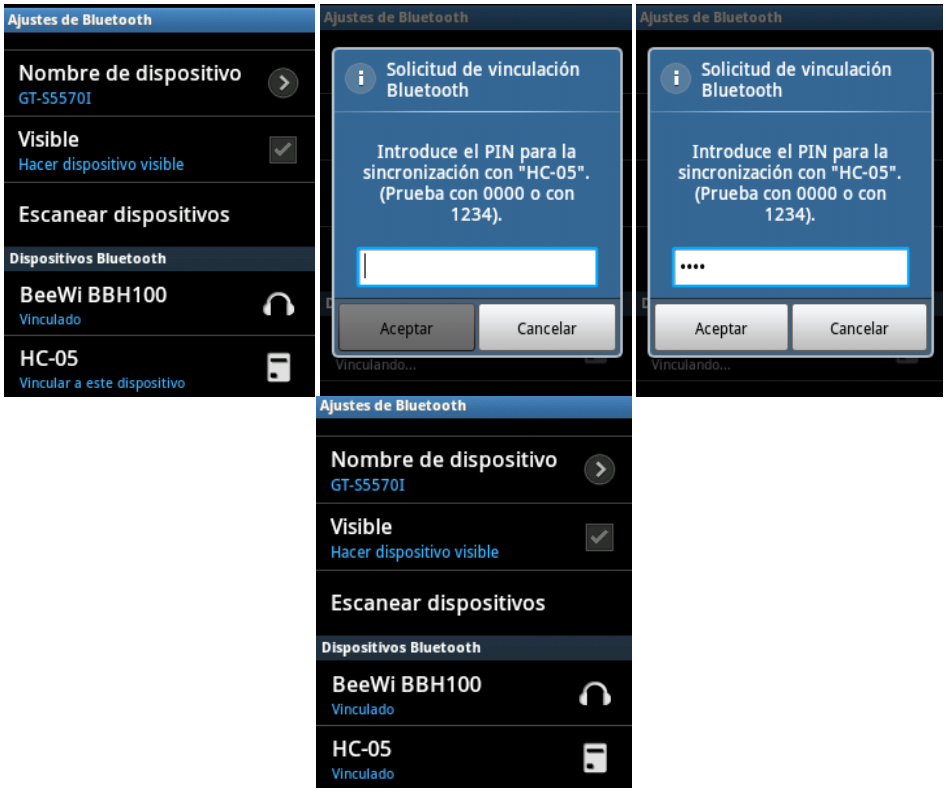


Figura 7-6.Vista previa del emparejamiento desde el dispositivo móvil con el receptor Bluetooth.

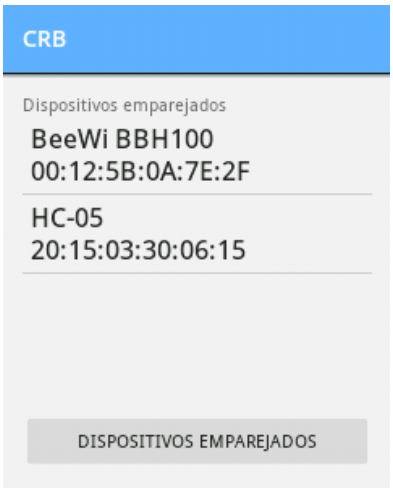


Figura 7-7. Vista previa de la aplicación tras vincular el dispositivo móvil con el receptor Bluetooth.

Tanto el nombre del dispositivo receptor como la clave de autenticación para el emparejamiento son las mencionadas anteriormente, las cuales venían por defecto en el dispositivo Bluetooth del receptor. Estos parámetros podrían ser modificados accediendo al dicho dispositivo desde el controlador Arduino mediante una serie de comandos AT.

En la siguiente figura, se puede ver el IMEI del dispositivo móvil desde el que se va trabajar. Para cualquier dispositivo, independientemente del sistema operativo utilizado, se puede saber el IMEI marcando en el teclado la siguiente secuencia: *#06#

Figura 7-8. Vista previa del IMEI del dispositivo móvil desde el que estamos trabajando.

Antes de la explicación del funcionamiento de la app, cabe destacar que aunque alguien supiera alguno de los números IMEI con los que se puede dar uso del servicio, no se podría suplantar la identidad ya que la app no da la opción de introducir ningún carácter, solo permite seleccionar diferentes opciones. El almacenamiento de caracteres se hace de forma automática sin posibilidad de enviar una cadena de caracteres diferente al IMEI del dispositivo móvil o a otra que asigne la aplicación móvil como se verá más adelante.

7.4. Funcionamiento del sistema emisor

En este apartado se explica cada una de las fases por las que puede pasar la aplicación instalada en el dispositivo móvil, igual que se hizo en el apartado 4 del capítulo 5, mostrando las respuestas de la aplicación móvil respecto la situación del momento.

7.4.1 Usuarios del servicio

Al igual que se hizo en el capítulo 5, para este caso se supondrá que el dispositivo móvil tiene activado el servicio de Bluetooth y que anteriormente se ha emparejado con el dispositivo Bluetooth del sistema receptor desde el menú de ajustes.

La aplicación móvil mostrará una lista de los dispositivos Bluetooth emparejados como la que se puede observar en la figura 7-8.

Una vez que el usuario seleccione el dispositivo Bluetooth receptor del sistema y la opción "Usuario" (porque el usuario tiene una antigüedad superior al tiempo establecido para la actualización del sistema receptor), la aplicación móvil procede automáticamente a la conexión con el sistema receptor, comprobando si dicha conexión está disponible. Si la conexión con el Bluetooth del sistema receptor está disponible, la aplicación móvil consulta el IMEI y lo envía al sistema receptor donde será procesado, liberando a continuación la conexión Bluetooth y volviendo al estado de reposo donde se muestra la pantalla que permite seleccionar el dispositivo Bluetooth receptor del sistema. El receptor Bluetooth se encargará de proceder de una manera u otra según dicho número, como se explicará más adelante. Todo esto puede verse a continuación:

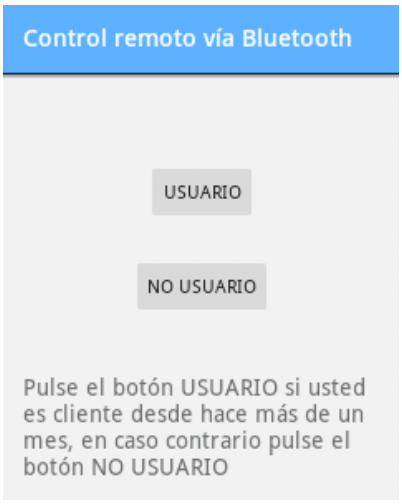


Figura 7-9. Menú mostrado tras seleccionar el dispositivo Bluetooth receptor del servicio de control remoto.

Figura 7-10. Mensaje de la aplicación cuando se está llevando a cabo la conexión Bluetooth y la obtención del IMEI.

7.4.2 No Usuarios del servicio

Al igual que en el apartado anterior, se supondrá que el dispositivo móvil tiene activado el servicio de Bluetooth y que anteriormente se ha emparejado con el dispositivo Bluetooth del sistema receptor desde el menú de ajustes. Además, se supondrá también que el dispositivo móvil cuenta con una conexión a internet

para realizar consultas a una base de datos alojada en un servidor web.

La aplicación móvil mostrará una lista de los dispositivos Bluetooth emparejados como la que se puede observar en la figura 7-8.

Una vez que el usuario seleccione el dispositivo Bluetooth receptor del sistema y la opción "No Usuario" (porque el usuario tiene una antigüedad inferior al tiempo establecido para la actualización del sistema receptor), la aplicación móvil procede automáticamente a la conexión con el sistema receptor, comprobando si dicha conexión está disponible. Si la conexión con el Bluetooth del sistema receptor está disponible, la aplicación móvil consulta el IMEI y lo envía a la base de datos, donde se procesará dicho IMEI y la base de datos devolverá una respuesta que informe a la aplicación móvil de si dicho IMEI se encuentra registrado en ella.

A continuación se muestra el aspecto de la aplicación móvil cuando está teniendo lugar la consulta a la base de datos:

Figura 7-11. Pantalla mostrada en la aplicación mientras se realiza la consulta a la base de datos.

La respuesta devuelta por la base de datos puede ser de dos tipos, analizado cada uno de ellos a continuación:

7.4.2.1 No Usuarios del servicio registrados en la base de datos

En este caso la aplicación móvil recibe de la base de datos una respuesta de confirmación de registro, generando un IMEI válido y enviándolo vía Bluetooth al sistema receptor. A continuación la aplicación móvil libera la conexión Bluetooth y vuelve al estado de reposo donde se muestra la pantalla que permite seleccionar el dispositivo Bluetooth del receptor del sistema.

Figura 7-12. Mensaje mostrado por la aplicación en caso de estar registrado en la base de datos.

7.4.2.2 No Usuarios del servicio no registrados en la base de datos

En este caso la aplicación móvil recibe de la base de datos una respuesta de negación de registro, por lo que la aplicación móvil procede a liberación de la conexión Bluetooth y vuelve al estado de reposo donde se muestra por pantalla la lista que permite seleccionar el Bluetooth del receptor del sistema como se muestra en la siguiente figura.

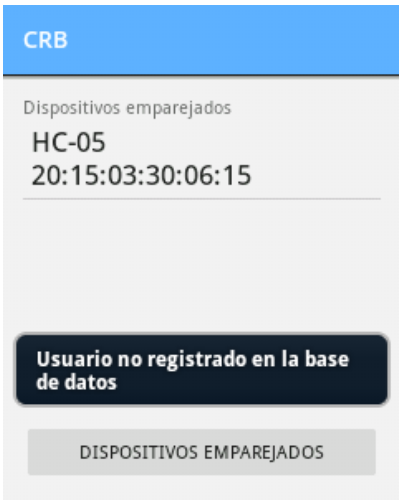


Figura 7-13. Mensaje mostrado por la aplicación en caso de no estar registrado en la base de datos.

7.4.3 Conexión a internet no disponible

En este caso no se puede proceder con la comprobación del IMEI en la base de datos, recibiendo por pantalla un mensaje que informe de tal evento tras lo que la aplicación móvil libera la conexión Bluetooth y vuelve al estado de reposo donde se muestra por pantalla la lista que permite seleccionar el Bluetooth del sistema como se muestra en la siguiente figura:

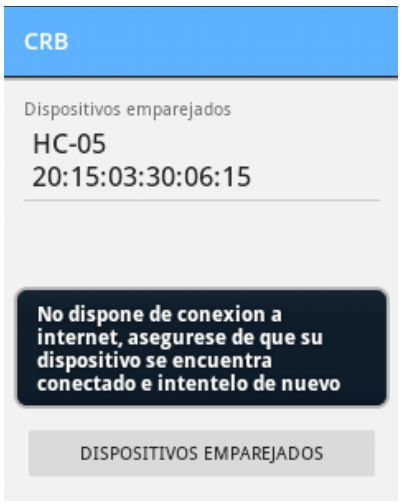


Figura 7-14. Mensaje mostrado por la aplicación en caso de no tener conexión a internet.

7.4.4 Conexión con el receptor Bluetooth fallida

En cualquiera de los casos anteriores, si la conexión con el receptor Bluetooth resultara fallida, al pulsar el botón "Usuario" o "No Usuario" la aplicación móvil no pasaría a consultar el IMEI del dispositivo, sino que directamente volvería a mostrar por pantalla el menú de selección de dispositivos emparejados como se muestra en la siguiente figura:

Figura 7-15. Mensaje mostrado por la aplicación en caso de que la conexión Bluetooth no pueda establecerse.

7.4.5 **Servicio Bluetooth desactivado**

En caso de tener el servicio Bluetooth del dispositivo móvil desactivado, la pantalla inicial mostraría un mensaje con una petición de permisos para proceder a la activación de dicho servicio para poder hacer uso de él, como se muestra en la siguiente figura:

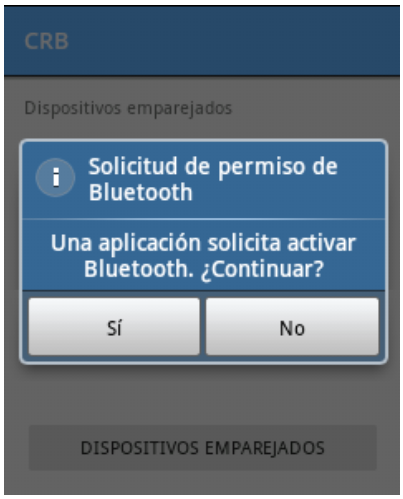


Figura 7-16. Solicitud de permiso para hacer uso del Bluetooth del dispositivo móvil.

Si se confirma dicha petición, se activará el Bluetooth y las pantallas mostradas serán las siguientes:

Figura 7-17. Mensaje mostrado al aceptar la petición del servicio Bluetooth y siguientes pantallas.

En caso de negar los permisos, la pantalla mostrada será la misma pero al intentar acceder a los dispositivos emparejados, nos aparecerá un mensaje por pantalla como el que observamos en dicha figura:

Figura 7-18. Mensaje mostrado al negar la petición del servicio Bluetooth y siguientes pantallas.

Como se puede observar, al intentar acceder a los dispositivos emparejados, no se encuentran muestra ninguno por tener el servicio Bluetooth del dispositivo móvil desactivado.

En caso de tener el servicio Bluetooth del dispositivo móvil activado, la pantalla mostrada en la aplicación será directamente aquella que nos da la opción de mostrar los dispositivos emparejados.

7.5. **Funcionamiento del sistema**

Una vez que el dispositivo Bluetooth del sistema receptor recibe el IMEI, se lo pasa al controlador Arduino, que tendrá una lista de los IMEI's pertenecientes a los usuarios con antigüedad mayor a la especificada y uno genérico para los casos de usuarios con antigüedad menor al tiempo de actualización de dicho controlador.

El IMEI recibido será comparado con los almacenados en el controlador Arduino mediante el código de programación que se puede observar en el anexo C. En caso de que coincida con alguno, el controlador Arduino mandará una orden al sistema que procede a la apertura de la puerta y volverá a reposo, mientras que si no existiera alguna coincidencia entre los IMEI's, el sistema seguiría en reposo a la espera de que se le pasara una nueva cadena de caracteres para comparar.

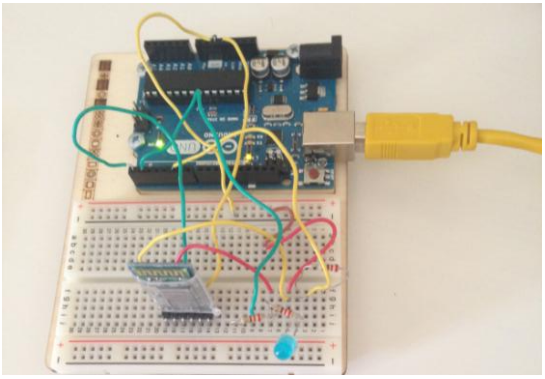


Figura 7-19. Sistema Arduino y Bluetooth para la recepción del IMEI del dispositivo móvil.

Para una visión más clara del sistema mostrado en la figura anterior, a continuación se muestra su protoboard, así como su esquemático y el pcb, realizados mediante el software "Fritzing":

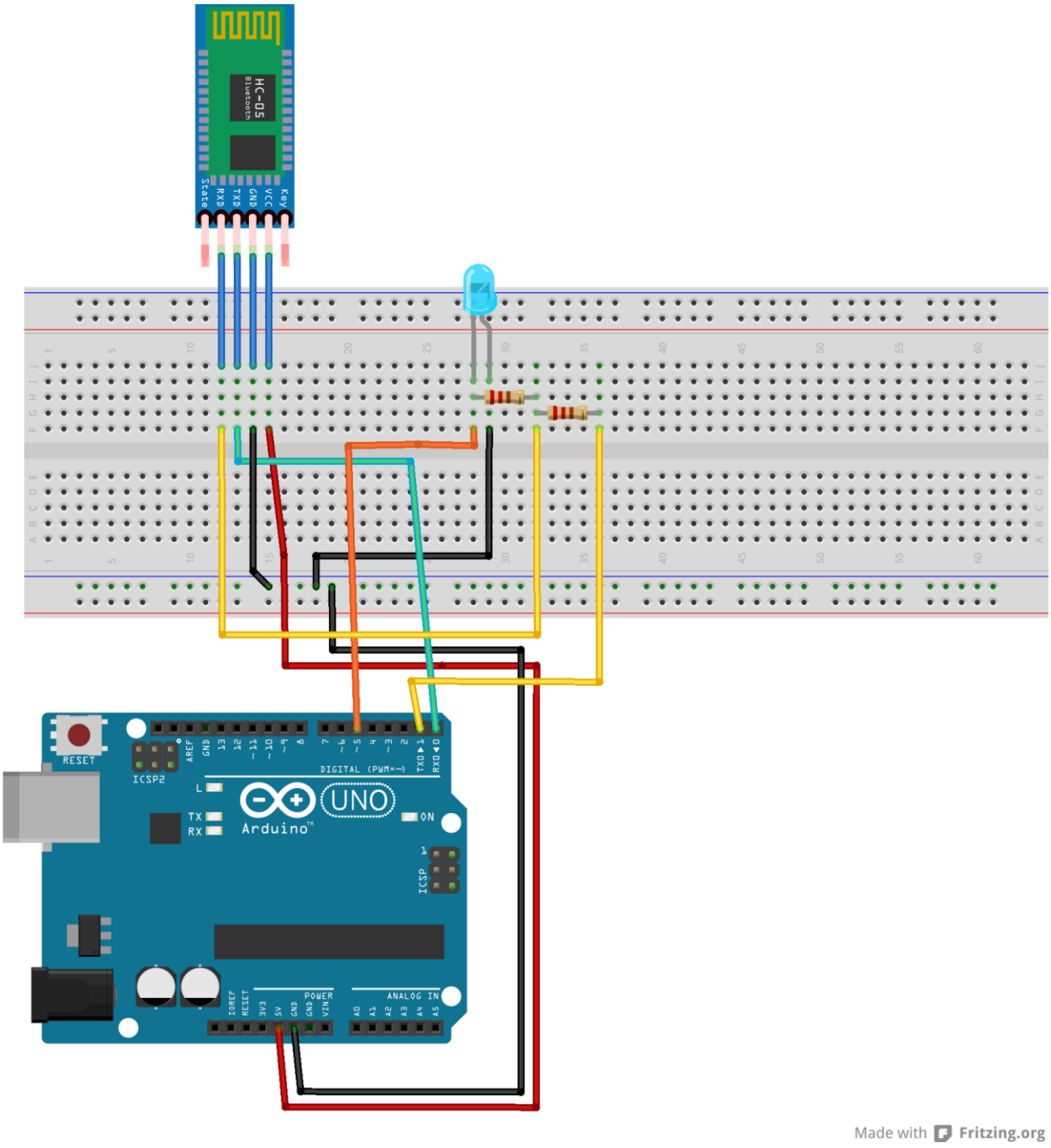


Figura 7-20. Protoboard del sistema receptor

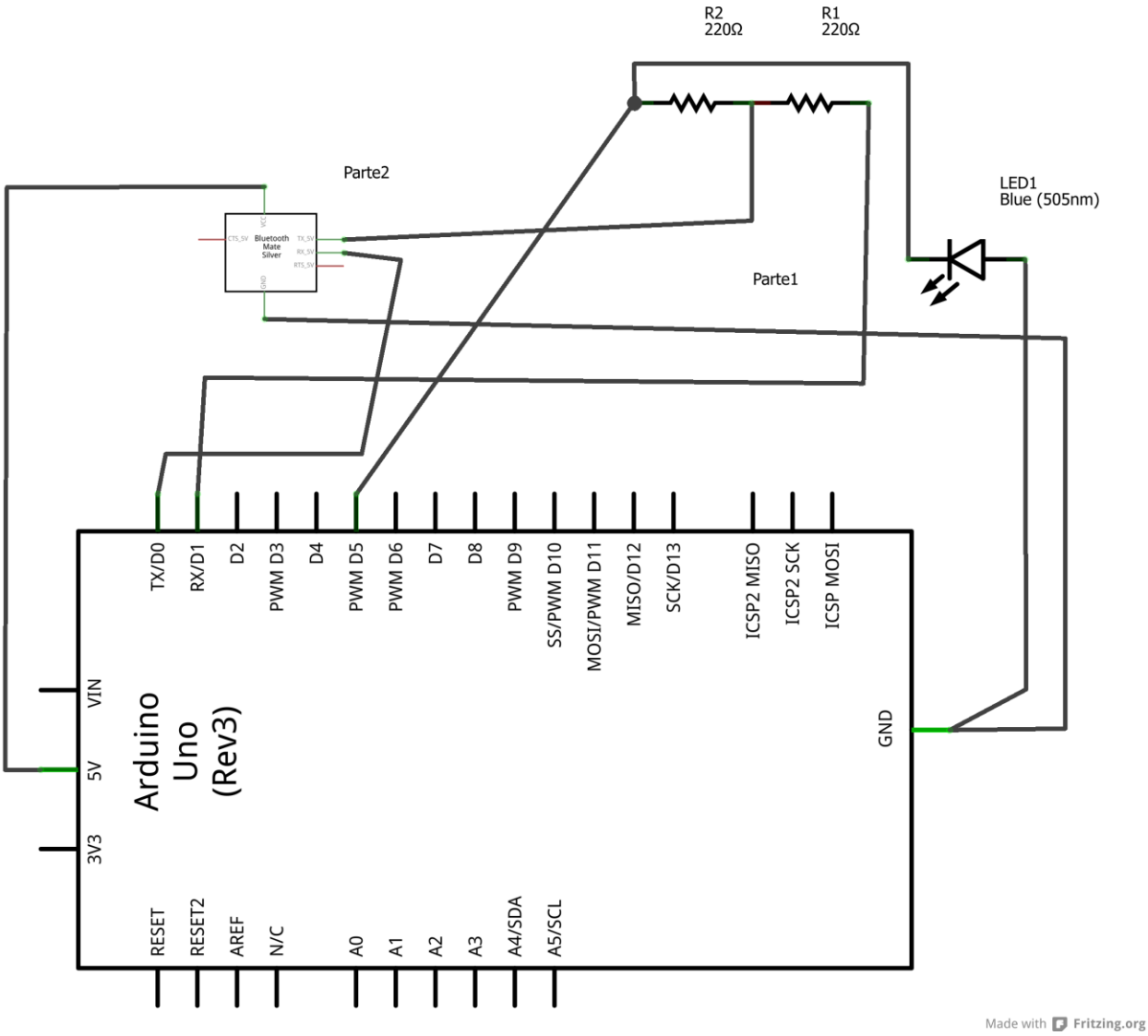


Figura 7-21. Esquemático del sistema receptor

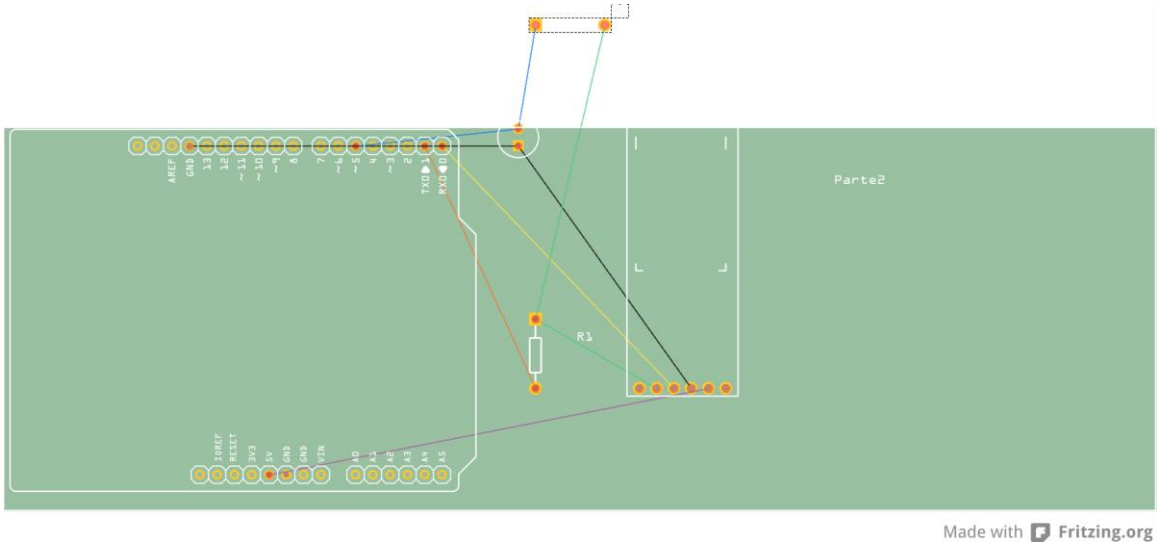


Figura 7-22. PCB del sistema receptor

7.6. Funcionamiento del servidor MYSQL y servicio web PHP

La base de datos MYSQL contendrá una tabla donde estarán almacenados los IMEI's de aquellos usuarios con antigüedad inferior a una específica donde, mediante el servicio web PHP programado (ver anexo B), el IMEI facilitado por la aplicación móvil desarrollada será comparado con cada una de las entradas de dicha tabla.

A continuación se pueden ver diversas pantallas de la página web mediante la que se ha realizado la base de datos almacenada en el servidor:

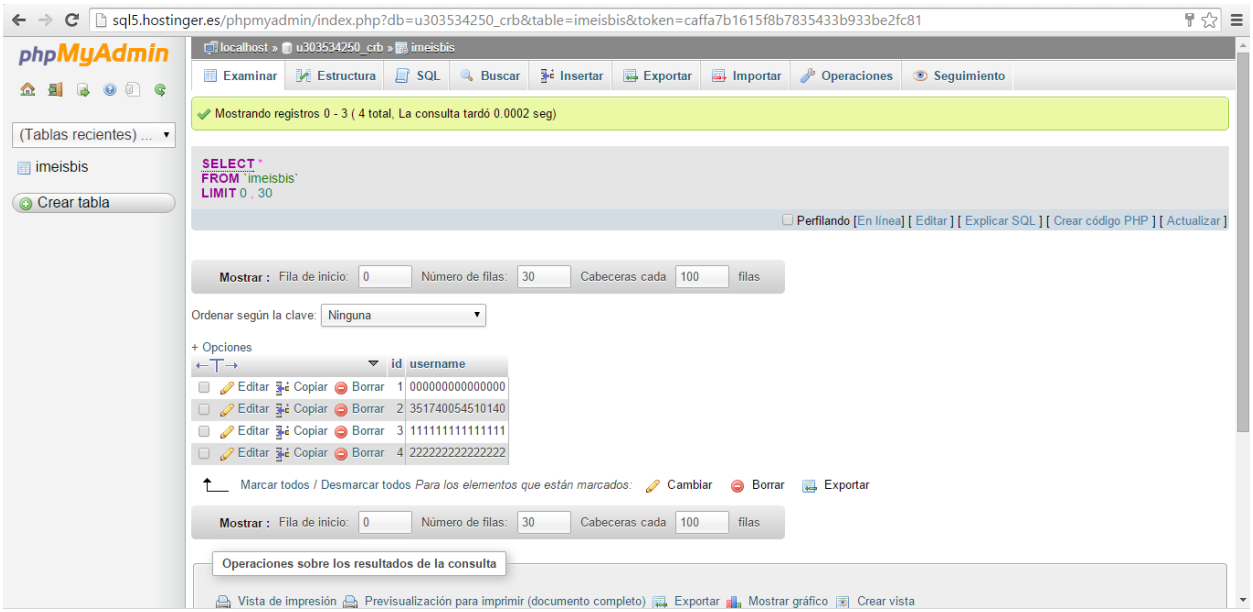


Figura 7-23. Vista previa de la creación de la tabla donde están almacenados los IMEI's (columna username).

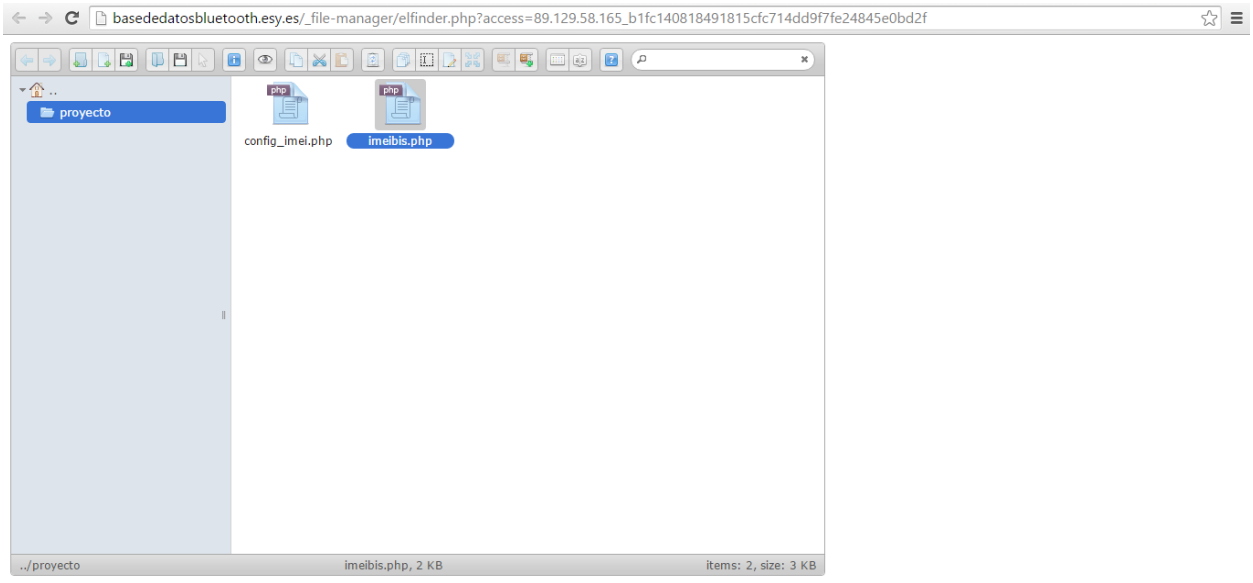


Figura 7-24. Vista previa de los ficheros PHP para el manejo de datos recibidos y la tabla de la base de datos.

8 CONCLUSIONES Y FUTURAS LÍNEAS DE TRABAJO

Con la realización de este proyecto se ha podido observar el potencial que tiene el sistema operativo Android y que, con las tecnologías o plataformas web, puede ser de gran utilidad para la realización de sistemas que permitan, a parte de cumplir con el objetivo de dar solución a un problema dado, ahorrar costes, dar facilidades en hábitos comunes y usuales...

Uno de los aspectos que podría mejorarse en la realización de este proyecto sería el de poder actualizar diariamente, en una franja horaria en la que el sistema se use menos, la lista de IMEI's contenida en el sistema receptor mediante una conexión GPS desde el controlador, de esta forma nos ahorraríamos el coste del técnico que lo hace cada cierto tiempo de forma manual y la aplicación móvil no necesitaría conexión a internet ahorrando dicho recurso y dotándola de mayor simplicidad ya que su interfaz mostraría un solo botón con la opción de envío del IMEI al receptor Bluetooth, sin necesidad de hacer consultas a la base de datos.

A parte de la simplicidad mencionada en la interfaz de la aplicación, a nivel lógico también supondría un ahorro de código de programación, por lo que la aplicación móvil podría ejecutarse más rápidamente en el dispositivo y ocuparía menos espacio en éste.

Llegados a este punto, se proponen otros posibles usos del sistema desarrollado anteriormente:

- Sustitución del método de identificación con tarjeta para usuarios de la comunidad universitaria.

  El objetivo sería sustituir la tarjeta de identificación tanto del personal docente como del alumnado por una aplicación para sus dispositivos móviles en los que se identifiquen con su uvus y contraseña y los lectores de tarjetas sean sustituídos por sistemas receptores formados por un controlador y un receptor Bluetooth como el detallado en el capítulo 5.

  De ésta forma no dependeríamos de dicha tarjeta además de proporcionar un grado de seguridad más alto ya que al perder la tarjeta cualquiera puede suplantar la identidad, mientras que con el sistema propuesto no.

- Identificación y control remoto en garajes individuales.

  En este caso, al ser un garaje individual dentro de nuestra propia casa, no sería necesario ninguna actualización de usuario ya que siempre sería el/los mismo/s dispositivos el/los que accionaría/n el control remoto.

  Este sistema permitiría al usuario agregar tantos números IMEI en la lista almacenada en el controlador como considerase oportuno desde un dispositivo almacenado anteriormente para evitar que cualquiera pudiese introducir cualquier número.

- Obtener el software directamente desde un buscador de aplicaciones Android como "Play Store", así como sus actualizaciones con mejoras o para su mantenimiento.

ANEXO A: CÓDIGOS DE PROGRAMACIÓN DE LA APLICACIÓN ANDROID

A.1 Códigos extensión .xml

Android Manifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.jorge.aplicacion" >
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk
        android:minSdkVersion="7"
        android:targetSdkVersion="15" />

    //permisos para trabajar con la conexión a internet del teléfono
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"
/>
   //permisos para trabajar con Bluetooth
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>
    <uses-permission android:name="android.permission.BLUETOOTH"/>

    //permisos para trabajar con el imei del teléfono
    <uses-permission android:name="android.permission.READ_PHONE_STATE"/>

    //Código que programa la pantalla que nos muestra una lista con los
posibles Receptores Bluetooth emparejados
    //con el dispositivo móvil
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Control remoto vía Bluetooth"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".DeviceList"
            android:label="CRB" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
```

```
            </intent-filter>
        </activity>
        <activity android:name=".MainActivity">
        </activity>
    </application>
</manifest>
```

menu_device_list,xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
tools:context="com.led.led.DeviceList">
    <item android:id="@+id/action_settings"
android:title="@string/action_settings"
        android:orderInCategory="100" app:showAsAction="never" />
</menu>
```

menu_main.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
tools:context=".MainActivity">
    <item android:id="@+id/action_settings"
android:title="@string/action_settings"
        android:orderInCategory="100" app:showAsAction="never" />
</menu>
```

activity_device_list.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
    android:layout_height="match_parent"
android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
tools:context=".DeviceList">

    //diseño del aspecto que tiene la lista de dispositivos Bluetooth
emparejados al teléfono móvil
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Dispositivos emparejados"
        android:id="@+id/textView"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Dispositivos emparejados"
        android:id="@+id/button"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
```

```
        android:layout_alignParentStart="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true" />

    <ListView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/listView"
        android:layout_centerHorizontal="true"
        android:layout_above="@+id/button"
        android:layout_below="@+id/textView" />
</RelativeLayout>
```

activity_main.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
    android:layout_height="match_parent"
android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
tools:context=".MainActivity">

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Usuario"
        android:id="@+id/button"
        android:layout_marginTop="56dp"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="No usuario"
        android:id="@+id/button2"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:text="Pulse el botón USUARIO si usted es cliente desde hace
más de un mes, en caso contrario pulse el botón NO USUARIO"
        android:id="@+id/textView2"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true" />

</RelativeLayout>
```

A.2 Códigos extensión .java

DeviceLList.java

```
package com.jorge.aplicacion;

import android.content.Intent;
import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.Set;
import java.util.Timer;
import java.util.TimerTask;


public class DeviceList extends ActionBarActivity
{
    //Declaración de la lista de dispositivos bluetooth emparejados con el
dispositivo móvil y el botón para que aparezca dicha lista
    Button btnPaired;
    ListView devicelist;
    //De claración variables BluetoothBluetooth
    private BluetoothAdapter myBluetooth = null;
    private Set<BluetoothDevice> pairedDevices;
    public static String EXTRA_ADDRESS = "device_address";

    //Función que define el comportamiento de la aplicación tras pulsar
alguno de los botones
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_device_list);

        //Llamadas a las funciones según el botón pulsado
        btnPaired = (Button)findViewById(R.id.button);
        devicelist = (ListView)findViewById(R.id.listView);

        //Configuración de conexión Bluetooth
        myBluetooth = BluetoothAdapter.getDefaultAdapter();

        //Si no tenemos conexión Bluetooth, se le notifica al usuario y
vuelve a la pantalla principal
        if(myBluetooth == null)
        {
            Toast.makeText(getApplicationContext(), "Conexion Bluetooth no
disponible", Toast.LENGTH_LONG).show();
            //finalizamos apk
```

```
            finish();
        }
        //sino, si tenemos Bluetooth pero no está activado, pedimos permisos
al usuario para conectarlo y hacer uso de él
        else if(!myBluetooth.isEnabled())
        {
            Intent turnBTon = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
            startActivityForResult(turnBTon,1);
        }

        //iteracción para emparejar con un dispositivo Bluetooth de la lista
        btnPaired.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                pairedDevicesList();
            }
        });
    }

    //Función que nos muestra los dispositivos emparejados al teléfono
    private void pairedDevicesList()
    {
        pairedDevices = myBluetooth.getBondedDevices();
        ArrayList list = new ArrayList();

        //Si tenemos más de un dispositivo emparejado, los mostramos todos en
forma de lista
        if (pairedDevices.size()>0)
        {
            for(BluetoothDevice bt : pairedDevices)
            {
                list.add(bt.getName() + "\n" + bt.getAddress()); //Obtenemos
los nombres de los dispositivos emparejados y sus direcciones
            }
        }
        //si el dispositivo no tiene dispositivos emparejados, se notifica al
usuario
        else
        {
            Toast.makeText(getApplicationContext(), "No se han encontrado
dispositivos Bluetooth emparejados.", Toast.LENGTH_LONG).show();
        }

        final ArrayAdapter adapter = new
ArrayAdapter(this,android.R.layout.simple_list_item_1, list);
        devicelist.setAdapter(adapter);
        devicelist.setOnItemClickListener(myListClickListener); //Llamamos a
este método cuando seleccionamos un dispositivo emparejado

    }

    //función asociada al evento de selección de un dispositivo Bluetooth
emparejado
    private AdapterView.OnItemClickListener myListClickListener = new
AdapterView.OnItemClickListener()
    {
        public void onItemClick (AdapterView<?> av, View v, int arg2, long
arg3)
        {
            // Obtenemos la dirección MAC
            String info = ((TextView) v).getText().toString();
```

```
            String address = info.substring(info.length() - 17);

            // Iniciamos una nueva actividad
            Intent i = new Intent(DeviceList.this, MainActivity.class);
            i.putExtra(EXTRA_ADDRESS, address);
            startActivity(i);
        }
    };


    @Override
    public boolean onCreateOptionsMenu(Menu menu)
    {
        getMenuInflater().inflate(R.menu.menu_device_list, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
        }

        return super.onOptionsItemSelected(item);

    }
}
```

MainActivity.java

```
package com.jorge.aplicacion;

import android.annotation.TargetApi;
import android.content.Context;
import android.content.SharedPreferences;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.Build;
import android.preference.PreferenceManager;
import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.telephony.TelephonyManager;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;

import android.bluetooth.BluetoothSocket;
import android.content.Intent;
import android.view.View;
import android.widget.Button;
import android.widget.SeekBar;
import android.widget.TextView;
import android.widget.Toast;
import android.app.ProgressDialog;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.os.AsyncTask;

import java.io.IOException;
```

```
import java.util.ArrayList;
import java.util.List;
import java.util.Timer;
import java.util.TimerTask;
import java.util.UUID;


import org.apache.http.message.BasicNameValuePair;
import org.json.JSONException;
import org.json.JSONObject;

//Definición de funciones principales del programa
public class MainActivity extends ActionBarActivity {



    //Definición de los botones para iteraccionar
    Button btnUsuario, btnNoUsuario;

    //Definición de variables
    String address = null;
    private ProgressDialog progress;
    BluetoothAdapter myBluetooth = null;
    BluetoothSocket btSocket = null;
    private boolean isBtConnected = false;
    static final UUID myUUID = UUID.fromString("00001101-0000-1000-8000-
00805F9B34FB");
    // Clase JSONParser
    JSONParser jsonParser = new JSONParser();
    //definimos la dirección web donde se aloja la base de datos utilizada
para las consultas
    private static final String LOGIN_URL =
"http://basededatosbluetooth.esy.es/proyecto/imeibis.php";
    // La respuesta del JSON es
    private static final String TAG_SUCCESS = "success";
    private static final String TAG_MESSAGE = "message";
    private int Variable;

    //Función de la pantalla principal:
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        Intent newint = getIntent();
        address = newint.getStringExtra(DeviceList.EXTRA_ADDRESS);
//recibimos la dirección Bluetooth
        setContentView(R.layout.activity_main);


//Iteracciones (llamadas a funciones),según el botón pulsado
        btnUsuario = (Button)findViewById(R.id.button);
        btnNoUsuario = (Button)findViewById(R.id.button2);

//se activa el bluetooth automáticamente y se enlaza con el dispositivo
seleccionado manualmente

        //Iteracción con el botón usuario registrado en el sistema que
acciona la apertura
      btnUsuario.setOnClickListener(new View.OnClickListener() {
          @Override
          public void onClick(View v) {
              //se almacena el imei del teléfonoen una variable
```

```
//llamamos a la función "envioimei" pasándole como parámetro el imei
queacabamos de almacenar
              new ConnectBT().execute(); //Llamada a la clase que realiza la
conexión de forma automática
              }
      });

//Iteracción con el botón no usuario, que serán aquellos usuarios recientes,
alojados en la base de datos remota, pero no en el stma
     btnNoUsuario.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v) {

new ConnectBTT().execute();

         }


         });


    }


    //Funcion que nos desconecta del Bluetooth del stma
    private void desconexion()
    {
        if (btSocket!=null) //si el socket no está ocupado
        {
            try
            {
                btSocket.close(); //cierre de conexión
            }
            catch (IOException e)
            { msg("Error");}
        }
finish();
    }

    //Definición de la función que recibe el imei del teléfono de aquél
usuario que está registrado ya en el sistema
    private void envioimei(String s, String imei)
    {

       if (btSocket!=null) {
           try {

               //si el socket no está ocupado, enviamos el imei al stma
               btSocket.getOutputStream().write(imei.toString().getBytes());
               // inmediatamente después, pase lo que pase, se procede a la
desconexión para dejar libre el Bluetooth del stma

           } catch (IOException e) {
               msg("Error");
           }
           Variable = 0;
           while (Variable < 100000000)
           {
               Variable++;
       }
```

```
        }
        desconexion();
    }


    private void msg(String s)
    {
        Toast.makeText(getApplicationContext(), s, Toast.LENGTH_LONG).show();
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;



    }


    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();

        if (id == R.id.action_settings) {
            return true;
        }

        return super.onOptionsItemSelected(item);
    }

//definición de la clase que se encarga de la conexión con el Bluetooth del
stma
    private class ConnectBT extends AsyncTask<Void, Void, Void>
    {

        private boolean ConnectSuccess = true;

        @Override
        protected void onPreExecute()
        {
            progress = ProgressDialog.show(MainActivity.this,
"Conectando...", "Espere!!!");  //Diálogo del progreso de la conexión
        }

        @TargetApi(Build.VERSION_CODES.GINGERBREAD_MR1)
        @Override
        protected Void doInBackground(Void... devices) //Mientras se muestra
el diálogo del progreso de la conexión, ésta se realiza en segundo plano
        {
            try
            {
                if (btSocket == null || !isBtConnected)
                {
                    myBluetooth = BluetoothAdapter.getDefaultAdapter();
                    BluetoothDevice dispositivo =
myBluetooth.getRemoteDevice(address);//nos conectamos a la dirección si está
disponible el bluetooth
                    btSocket =
dispositivo.createInsecureRfcommSocketToServiceRecord(myUUID);//creamos
RFCOMM (SPP) para la conexión
                    BluetoothAdapter.getDefaultAdapter().cancelDiscovery();
```

```
                    btSocket.connect();//se conecta
                }
            }
            catch (IOException e)
            {
                ConnectSuccess = false;
            }

            return null;
        }


        @Override
        protected void onPostExecute(Void result) //después de todo lo que
pasa en segundo plano, si todo escorrecto:
        {
            super.onPostExecute(result);

            if (!ConnectSuccess)
            {
                progress.dismiss();
                msg("Conexion fallida. Pruebe de nuevo.");
                finish();
            }
            else
            {
                TelephonyManager tm = (TelephonyManager)

getSystemService(MainActivity.this.TELEPHONY_SERVICE);
                String imei = tm.getDeviceId();
                progress.dismiss();
                msg("Conectado al receptor Bluetooth, obteniendo imei del
usuario");
                isBtConnected = true;

                envioimei("imei", imei);
            }
        }
    }


    private class ConnectBTT extends AsyncTask<Void, Void, Void>
    {

        private boolean ConnectSuccess = true;

        @Override
        protected void onPreExecute()
        {
            progress = ProgressDialog.show(MainActivity.this,
"Conectando...", "Espere!!!");  //Diálogo del progreso de la conexión
        }

        @TargetApi(Build.VERSION_CODES.GINGERBREAD_MR1)
        @Override
        protected Void doInBackground(Void... devices) //Mientras se muestra
el diálogo del progreso de la conexión, ésta se realiza en segundo plano
        {
            try
            {
                if (btSocket == null || !isBtConnected)
                {
```

```
                    myBluetooth = BluetoothAdapter.getDefaultAdapter();
                    BluetoothDevice dispositivo =
myBluetooth.getRemoteDevice(address);//nos conectamos a la dirección si está
disponible el bluetooth
                    btSocket =
dispositivo.createInsecureRfcommSocketToServiceRecord(myUUID);//creamos
RFCOMM (SPP) para la conexión
                    BluetoothAdapter.getDefaultAdapter().cancelDiscovery();
                    btSocket.connect();//se conecta
                }
            }
            catch (IOException e)
            {
                ConnectSuccess = false;
            }

            return null;
        }


        @Override
        protected void onPostExecute(Void result) //después de todo lo que
pasa en segundo plano, si todo escorrecto:
        {
            super.onPostExecute(result);

            if (!ConnectSuccess)
            {
                progress.dismiss();
                msg("Conexion fallida. Pruebe de nuevo.");
                finish();
            }
            else
            {
                TelephonyManager tm = (TelephonyManager)

getSystemService(MainActivity.this.TELEPHONY_SERVICE);
                String imei = tm.getDeviceId();
                progress.dismiss();
                msg("Conectado al receptor Bluetooth, obteniendo imei del
usuario y enviando a la base de datos...");
                isBtConnected = true;
Variable=0;
                while (Variable<5000000){
                    Variable++;
                }
                ConnectivityManager connMgr =  (ConnectivityManager)
                        getSystemService ( Context. CONNECTIVITY_SERVICE );
                NetworkInfo networkInfo = connMgr . getActiveNetworkInfo();

                if  ( networkInfo !=  null  && networkInfo . isConnected()) {
                    new AttemptLogin().execute(); //Ejecutamos la función que
accede a la base de datos y realiza la consulta
                }
                else {

                    msg("No dispone de conexion a internet, asegurese de que
su dispositivo se encuentra conectado e intentelo de nuevo");
                    desconexion();

                }
            }
```

```
        }
    }

    //definición de la clase que se ocupa de la conexión, y consulta a la
base de datos en el servidor web
    class AttemptLogin extends AsyncTask<String, String, String> {

        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            progress = new ProgressDialog(MainActivity.this);
            progress.setIndeterminate(false);
            progress.setCancelable(true);
        }

        @Override
        protected String doInBackground(String... args) {


            int success;
            //obtenemos el imei del teléfono
            TelephonyManager tm = (TelephonyManager)
                    getSystemService(MainActivity.this.TELEPHONY_SERVICE);
            String imei = tm.getDeviceId();
            //definimos un imei general, de tal forma que sí el servidor nos
devuelve una respuesta de imei registrado, podamos acceder al stma
            String imei_general ="123456789";
            try {
                List params = new ArrayList();
                //pasamos como usuario y contraseña (a la base de datos), el
imei del teléfono
                params.add(new BasicNameValuePair("username", imei));
                Log.d("consultando!", "empezando");



                JSONObject json = jsonParser.makeHttpRequest(LOGIN_URL,
"POST",
                        params);

                Log.d("Identificando", json.toString());
                success = json.getInt(TAG_SUCCESS);
                if (success == 1) {
                    Log.d("Login Successful!", json.toString());
                    SharedPreferences sp = PreferenceManager
                            .getDefaultSharedPreferences(MainActivity.this);
                    SharedPreferences.Editor edit = sp.edit();

                    edit.putString("username", imei);
                    edit.commit();
                    //llegados aqui, todo es correcto, asociamos a la
variable imei, la cadena "imei general"
                    envioimei("imei", imei_general);
                    //inmediatamente después llamamos a la función
desconexión para liberar el Bluetooth del sistema

                    return json.getString(TAG_MESSAGE);
                } else {
                    //sino, el imei no se encuentra en la base de datos, no
asociamos el imei general, y directamente desconectamos
                    Log.d("Login Failure!", json.getString(TAG_MESSAGE));
                    desconexion();
```

```
                    return json.getString(TAG_MESSAGE);

                }
            } catch (JSONException e) {
                e.printStackTrace();
            }

            return null;

        }

        protected void onPostExecute(String file_url) {
            progress.dismiss();
            if (file_url != null) {
                Toast.makeText(MainActivity.this, file_url,
Toast.LENGTH_LONG).show();
            }
        }

    }


}
```

A.3 Códigos con extensión .java utilizando lenguaje JSON

JSONParser.java

```
package com.jorge.aplicacion;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.util.List;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.client.utils.URLEncodedUtils;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONException;
import org.json.JSONObject;
import android.R;

import android.util.Log;


//Clase Que define el comportamiento adecuado para hacer una consulta a una
base de datos en un servidor web
public class JSONParser {
```

```
    static InputStream is = null;
    static JSONObject jObj = null;
    static String json = "";

    // constructor
    public JSONParser() {

    }


    public JSONObject getJSONFromUrl(final String url) {

        // Obtenemos la respuesta HTTP
        try {
            // Cliente y respuesta HTTP
            DefaultHttpClient httpClient = new DefaultHttpClient();
            HttpPost httpPost = new HttpPost(url);

            HttpResponse httpResponse = httpClient.execute(httpPost);
            //Extracción de datos de la respuesta
            HttpEntity httpEntity = httpResponse.getEntity();
            is = httpEntity.getContent();

        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        } catch (ClientProtocolException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }

        try {
            BufferedReader reader = new BufferedReader(new InputStreamReader(
                    is, "iso-8859-1"), 8);
            StringBuilder sb = new StringBuilder();
            String line = null;

            while ((line = reader.readLine()) != null) {
                sb.append(line + "\n");
            }

            is.close();

            json = sb.toString();
        } catch (Exception e) {
            Log.e("Error de Buffer", "Error al convertir el resultado " +
e.toString());
        }

        try {
            jObj = new JSONObject(json);
        } catch (JSONException e) {
            Log.e("JSON Parser", "Error de datos " + e.toString());
        }
        return jObj;

    }


    public JSONObject makeHttpRequest(String url, String method,
                                      List params) {
```

```
        // Haciendo la Peticion HTTP
        try {

            // comprobamos la respuesta de cada método
            if(method == "POST"){
                DefaultHttpClient httpClient = new DefaultHttpClient();
                HttpPost httpPost = new HttpPost(url);
                httpPost.setEntity(new UrlEncodedFormEntity(params));

                HttpResponse httpResponse = httpClient.execute(httpPost);
                HttpEntity httpEntity = httpResponse.getEntity();
                is = httpEntity.getContent();

            }else if(method == "GET"){
                DefaultHttpClient httpClient = new DefaultHttpClient();
                String paramString = URLEncodedUtils.format(params, "utf-8");
                url += "?" + paramString;
                HttpGet httpGet = new HttpGet(url);

                HttpResponse httpResponse = httpClient.execute(httpGet);
                HttpEntity httpEntity = httpResponse.getEntity();
                is = httpEntity.getContent();
            }

        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        } catch (ClientProtocolException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }

        try {
            BufferedReader reader = new BufferedReader(new InputStreamReader(
                    is, "iso-8859-1"), 8);
            StringBuilder sb = new StringBuilder();
            String line = null;
            while ((line = reader.readLine()) != null) {
                sb.append(line + "\n");
            }
            is.close();
            json = sb.toString();
        } catch (Exception e) {
            Log.e("Error de Buffer", "Error de conversión " + e.toString());
        }
        try {
            jObj = new JSONObject(json);
        } catch (JSONException e) {
            Log.e("JSON Parser", "Error de datos " + e.toString());
        }
        return jObj;

    }
}
```

ANEXO B: CÓDIGOS DE PROGRAMACIÓN EN LENGUAJE PHP

Imeisbis.php

```
<?php

//carga y se conecta a la base de datos
require("config_imei.php");

if (!empty($_POST)) {
    //obtenemos los usuarios respecto a la usuario que llega por parámetro
    $query = "
            SELECT
                id,
                username
            FROM imeisbis
            WHERE
                username = :username
        ";

    $query_params = array(
        ':username' => $_POST['username']
    );

    try {
        $stmt   = $db->prepare($query);
        $result = $stmt->execute($query_params);
    }
    catch (PDOException $ex) {
        //para testear pueden utilizar lo de abajo
        //die("la consulta murio " . $ex->getMessage());

        $response["success"] = 0;
        $response["message"] = "Problema con la base de datos, vuelve a intetarlo";
        die(json_encode($response));
```

```
    }

    //la variable a continuación nos permitirá determinar
    //si es o no la información correcta
    //la inicializamos en "false"
    $validated_info = false;

    //vamos a buscar a todas las filas
    $row = $stmt->fetch();
    if ($row) {
        if ($_POST['username'] === $row['username']) {
            $login_ok = true;
        }
    }
     // Otherwise, we display a login failed message and show the login form again
    if ($login_ok) {
        $response["success"] = 1;
        $response["message"] = "Usuario registrado en la base de datos";
        die(json_encode($response));
    } else {
        $response["success"] = 0;
        $response["message"] = "Usuario no registrado en la base de datos";
        die(json_encode($response));
    }
} else {
?>
  <h1>Login</h1>
  <form action="imeibis.php" method="post">
      Username:<br />
      <input type="text" name="username" placeholder="username" />
      <br /><br />
      <input type="submit" value="Login" />
  </form>
 <?php
}

?>
```

config_imei.php

```
<?php

 //los atributos de abajo son los que indican los parámetros necesarios para acceder a la
//base de datos.


    $username = "u303534250_crb";
    $password = "bluetooth0";
    $host = "localhost";
    $dbname = "u303534250_crb";

    try
    {
        $db = new PDO("mysql:host={$host};dbname={$dbname};charset=utf8", $username,
$password, $options);
```

```
}
catch(PDOException $ex)
{
    die("Failed to connect to the database: " . $ex->getMessage());
}

$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
$db->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE, PDO::FETCH_ASSOC);

if(function_exists('get_magic_quotes_gpc') && get_magic_quotes_gpc())
{
    function undo_magic_quotes_gpc(&$array)
    {
        foreach($array as &$value)
        {
            if(is_array($value))
            {
                undo_magic_quotes_gpc($value);
            }
            else
            {
                $value = stripslashes($value);
            }
        }
    }

    undo_magic_quotes_gpc($_POST);
    undo_magic_quotes_gpc($_GET);
    undo_magic_quotes_gpc($_COOKIE);
}
header('Content-Type: text/html; charset=utf-8');
session_start();


?>
```

ANEXO C: CÓDIGOS DE PROGRAMACIÓN EN C++ PARA EL ARDUINO

Receptor_arduino.ino

```
char command;
String string;
#define led 5

 void setup()
 {
   Serial.begin(9600);
   pinMode(led, OUTPUT);
 }

 void loop()
 {
   if (Serial.available() > 0)

   {string = "";}

   while(Serial.available() > 0)
   {
     command = ((byte)Serial.read());

     if(command == ':')
     {
       break;
     }
      else
     {
       string += command;
     }

    }
```

```
    // original: 351740054510140

    delay(1000);
    if (string == "351740054510140"|| string=="123456789")
   {

    Serial.println("entra");
    Serial.println(string);
    delay(1000);
    analogWrite(led, 255);
    delay(1000);
    string="";
        analogWrite(led, 0);
   }
   else{
         Serial.println(string);
    Serial.println("no entra");
    delay(1000);
   }

 }
     void ledOff()
{
    digitalWrite(led, LOW);

}
 void ledOn()
  {
    digitalWrite(led, HIGH);

    ledOff();
   }
```

ACRÓNIMOS

IRDA	InfraRed Data Association	Asociación de datos por infrarrojos	3
VFIR	Very Fast InfraRed	Infrarrojos de muy alta velocidad	4
LAN	Local Area Network	Red de área local	4
IEEE	Institute of Electrical and Electronic Engineers	Instituto de Ingenieros Eléctricos y Electrónicos	4
Mbps	Megabits per second	Megabits por segundo	5
WPAN	Wireless Personal Area Network	Red inalámbriba de área personal	5
AA		Alcalinas	5
ISM	Industrial, Scientific and medical	Industriales, científicas y médicas	6
GHz	Gigahertz	Gigahercio	6
EEUU		Estados Unidos	6
CSMA/CA	Carrier Sense Multiple Acces	Acceso múltiple por detección de portadora con evasión de colisiones	6
PAN	Personal Area Network	Red de área personal	7
IBM	International Business Machines Corporation	Comercio internacional de máquinas	9
SIG	Special Interest Group	Grupo de interés especial	9
ACL	Asynchronous Connection-Oriented	Enlace asíncrono sin conexión	10
SCO	Syncronous Connection-Oriented	Enlace síncrono orientado a conexión	10
PCM	Pulse code modulation	Modulación por impulsos codificados	10
CVSD	Continuous Variable Slope Delta	Delta de pendiente contínuamente variable	10
mW	MilliWatt	Milivatio	11
MAC	Medium Access Control	Control de acceso al medio físico	11
OSI	Open Systems Interconnection	Interconexión de sistemas abiertos	12
LMP	Link Manager Protocol	Protocolo de gestión de enlace	12
PDU	Protocol Data Unit	Unidad de datos del protocolo	13
LM	Link Manager	Gestor de enlace	14
OBEX	Object Exchange protocol	Protocolo de intercambio de objetos	14
RF	Radio Frecuency	Radiofrecuencia	18
L2CAP	Logical Link Control and Adptation Protocol	Protocolo de adaptación y control de enlace lógico	18
SDP	Service Discovery protocol	Protocolo de descubrimiento de protocolos	18

RFCOMM	Radio Frecuency Communication	Comunicación por radiofrecuencia	18
TCS	Telephony Control Specification	Especificación de control de telefonía	18
AT	Attention	Atencion	18
PPP	Point-to-Point Protocol	Protocolo punto a punto	18
UDP	User Datagram Protocol	Protocolo de datagramas de usuario	18
TCP/IP	Trasnmission Control Protocol/ Internet Protocol	Protocolo de control de transmisión/ protocolo de internet	18
IrMC	Infrared Mobile Communications	Comunicaciones móviles por infrarrojos	18
WAE	WirelessApplication Environment	Entorno inalámbrico de aplicación	18
EDR	Enhanced Data Rate	Velocidad de datos mejorada	19
HS	High Speed	Alta Velocidad	19
RSSI	Received Signal Strength Indicator	Indicador de fuerza de señal	19
AFH	Adaptative Frequency Hopping	frecuencia de salto adaptativo de espectro ampliable	20
SSP	Secure Simple Paining	Emparejamiento seguro sencillo	20
EIR	Extended Inquiry Response	Respuesta de consulta extendida	20
GFSK	Gaussian Frequency Shift Keying	Frecuancia Gassiana con clave de cambio	20
PSK	Phase Shift Keying	Modulación por desplazamiento de fase	20
AMP	Alternate MAC/PHY	MAC/PHY alternadas	20
HCI	Host Controller Interface	Interfaz de controlador host	20
PAL	Phase Alternating Line	Línea de fase alternada	20

REFERENCIAS

[1] Nathan J.Miller, *Tecnología Bluetooth*, Madrid: McGraw-Hill Profesional, 2002.

[2] *ZIGBEE*, 2012, *http://sx-de-tx.wikispaces.com/ZIGBEE*

[3] *Bluetooth*, 2015, *https://es.wikipedia.org/wiki/Bluetooth*

[4] *La resurrección de Bluetooth*, 2013, *http://www.bbc.com/mundo/blogs/2013/12/131213_blog_un_mundo_feliz_resurreccion_del_bluetooth*

[5] *Nuevas tendencias en la tecnología Bluetooht*, 2013, *http://eie.ucr.ac.cr/uploads/file/proybach/pb2013/pb2013_031.pdf*

[6] *Redes y servicios Ubicuos para internet de las cosas en dispositivos llevables*, 2012, *http://oa.upm.es/13950/1/PFC_ALEXANDRA_CUERVA_GARCIA.pdf*

[7] *Bluetooth 4.0, que podemos esperar*, 2009, *http://www.xataka.com/otros/bluetooth-40-que-podemos-esperar*

[8] *Bluetooth 4.1, nueva versión con mejoras interesantes*, 2013, http://www.xatakahome.com/la-red-local/bluetooth-4-1-nueva-version-con-mejoras-interesantes

[9]*Bluetooth Basic*, *https://learn.sparkfun.com/tutorials/bluetooth-basics/common-versions*

[10] José Rafael Lajara Vizcaíno y José Pelegrí Sebastía, *Sistemas integrados con arduino*, Barcelona: marcombo, 2014.

[11] Eduardo Gallego del Pozo, *Empezando con arduino* UNO, Madrid: complubot, 2014.

[12] Scott Fitzgerald and Michael Shiloh, *The arduino projects book*, Italia, 2012.

[13] Miguel Pareja Aparicio, *Iniciación a arduino uno*, Barcelona: marcombo, 2014..

[14] *Tarjeta HC 05-Arduino*, 2014, *http://www.sigmaelectronica.net/manuals/HOJA%20REFERENCIA%20TARJETA%20HC-05%20ARD.pdf*

[15] *Comandos AT con Arduino uno: modulo Bluetooth HC-05*, 2013, *https://www.youtube.com/watch?v=xwu33aA89qQ*

[16]Modulo Bluetooth HC 05, 2015, *https://www.youtube.com/watch?v=XXtf1XtQC_0*

[17] *El módulo Bluetooth HC-05*, 2015, *http://www.prometec.net/bt-hc05/*