Universidad de Sevilla

Departamento de Ingeniería Electrónica

TESIS DOCTORAL

INTEROPERABILIDAD DINÁMICA DE SERVICIOS MEDIANTE COMUNICACIONES INALÁMBRICAS BLUETOOTH EN ENTORNOS URBANOS Y SISTEMAS INTELIGENTES DE TRANSPORTE

Autor: Francisco Javier Cortés Martínez

Directores: Federico Barrero García
Sergio Toral Marín

Sevilla, 3 de Marzo de 2011

A mis padres y a mi hermana

Resumen

La evolución en la capacidad de los sistemas empotrados y, en particular, de aquellos sistemas embarcados en terminales móviles, hacen posible que hoy en día puedan ofrecer servicios avanzados que antes sólo se concebían para redes y sistemas de mayor envergadura tales como PCs conectados a través de Internet. En el nuevo escenario de los servicios en movilidad las tecnologías inalámbricas juegan un papel fundamental, y se complementan con tecnologías procedentes de otros ámbitos, como el middleware, para dar soporte a los servicios. El uso de estas tecnologías en entornos urbanos y sistemas de transporte inteligente conlleva una serie de ventajas que se manifiestan, por ejemplo, en la posibilidad de ofrecer servicios dependientes del contexto o en la interoperabilidad de servicios.

Entre esas tecnologías destacan Bluetooth, como base para las comunicaciones inalámbricas y CORBA, como plataforma middleware. Si bien, las características de ambas tecnologías son bastante adecuadas al tipo de entornos mencionado, es necesario acomenter ciertas adaptaciones que tengan en cuenta las particularidades y situaciones que se dan en estos entornos y que garanticen un correcto acceso a los servicios y una gestión inteligente de las conexiones.

Sobre la base de Wireless CORBA, una de las iniciativas que une Bluetooth y CORBA, en esta Tesis se plantean modificaciones y se definen nuevos elementos que permiten su utilización en entornos urbanos y sistemas inteligentes de transporte. La definición del sistema se acompaña de estudios y recomendaciones que ayudan a optimizar el rendimiento en función del tipo de servicio y las condiciones del entorno. El sistema planteado supone una base común para el desarrollo de nuevos servicios, y prueba de ello lo constituyen las aplicaciones implementadas, que demuestran la viabilidad práctica del mismo.

Agradecimientos

A mis directores de tesis, Federico Barrero García y Sergio Toral Marín, por la inestimable ayuda y ánimo que me han dado para la eleboración de esta tesis. A ellos les agradezco la confianza que han depositado en mi para embarcarnos en diversos proyectos y su gran apoyo en momentos muy importantes para el desarrollo de mi carrera profesional.

A Sergio Gallardo Vázquez, por su ayuda desinteresada y sus sabios consejos.

A los miembros del grupo de investigación ACE-TI, y en especial a José María Hinojo y Esteban Marsal por las horas que pasaron ayudándome a realizar pruebas para la tesis.

A mis padres, por su esfuerzo silencioso y abnegado que, en ocasiones, los hijos no alcanzamos a ver y a valorar como se merece.

Índice de contenidos

Índice de figuras

Índice de tablas

1 INTRODUCCIÓN

Introducción

Los servicios en movilidad han cobrado gran importancia durante los últimos años como consecuencia del avance de la tecnologías de la información y las comunicaciones. La fuente habitual de información para estos servicios está en Internet, aunque cada vez es más fácil encontrar estas fuentes en entornos más cercanos al contexto en el que se mueve el usuario. Estas fuentes generan una gran cantidad de información con un gran potencial para ofrecer nuevos servicios. Un ejemplo de esta nueva situación lo constituyen las modernas infraestructuras públicas que existen en las ciudades, compuestas por cientos de cámaras de vigilancia, reguladores de semáforos, paneles informativos y equipos para la estimación de parámetros de tráfico. Estos equipos están normalmente conectados a una red urbana de datos, diseminando información de diversos tipos (datos en tiempo real, imágenes, vídeo, etc.) [1]. Esta información ayuda a las administraciones de gestión del tráfico a tomar decisiones, llevando a cabo las actuaciones apropiadas para reducir la congestión y mejorar el rendimiento global de la red de tráfico [2]. Pero toda esta información también puede ser útil si se traslada a otros ámbitos como los entornos urbanos inteligentes, donde la actividad humana tiene lugar.

El desarrollo de los sistemas para entornos urbanos inteligentes constituye un campo multidisciplinar que conjuga tecnología de sensores, visión artificial, redes inalámbricas, middleware, multimedia y muchos otros campos de investigación. Algunas de estas tecnologías, aunque no han sido diseñadas específicamente para su uso en estos entornos, convenientemente adaptadas, pueden mejorar el rendimiento y la funcionalidad de éstos.

En la actualidad existen varios sistemas e iniciativas en el ámbito de los entornos urbanos y sistemas de transporte inteligentes aunque se detecta una notable heterogeneidad tanto en las tecnologías y modelos de comunicación empleados como en la funcionalidad a la que están destinados. En este sentido, estas iniciativas no siempre resultan lo suficientemente flexibles como para servir de plataforma que dé soporte al desarrollo de una amplia gama de servicios. Esta heterogeneidad también se manifiesta en los sistemas electrónicos que están involucrados en estos sistemas y en los que potencialmente pudieran estarlo, siendo más evidente en el caso de los equipos terminales que en los equipos de infraestructura de red.

Objetivos

La investigación que se lleva a cabo en esta Tesis centra su estudio en dos pilares básicos: el middleware y las tecnologías inalámbricas (concretamente Bluetooth). Partiendo de estas dos tecnologías y de una de las plataformas existentes que combina ambas, se definen los siguientes objetivos para esta Tesis:

- Identificación de los puntos de unión entre ambas tecnologías.
- Modificación de la plataforma middleware para su adaptación a las particularidades de los entornos urbanos.
- Incorporación de nuevos elementos que cubran las funcionalidades no implementadas por la plataforma y requeridas por el tipo de entorno.
- Identificación y aplicación de las características relevantes de Bluetooth para mejorar la eficiencia del sistema.
- Implementación real del sistema y pruebas mediante aplicaciones de ejemplo.

El objetivo final consiste en conseguir un sistema que permita la interoperabilidad de servicios en entornos urbanos y sistemas inteligentes de transporte basándose en las tecnologías anteriormente citadas. La heterogeneidad de sistemas a la que se hacía mención en relación a las iniciativas existentes es más notable en el caso particular de los entornos urbanos inteligentes, lo que implica un menor grado de estandarización pero también un mayor espacio para la aportación de nuevas ideas. Por este motivo, el sistema que se plantea en esta Tesis está enfocado principalmente a este tipo de entornos con las características y restricciones que ello conlleva.

Organización de la Tesis

La estructura de esta Tesis comienza en el capítulo 2 con un estudio del estado del arte sobre las inciativas existentes en la actualidad en el ámbito de los entornos urbanos y los sistemas inteligentes de tranporte. En el capítulo 3 se realiza un estudio de las tecnologías inalámbricas prestando especial atención a aquellas características relevantes para el tipo de sistemas sobre el que trata la Tesis. En este capítulo se hace un especial tratamiento de la tecnología Bluetooth. En el capítulo 4 se describen las contribuciones y cómo estas influyen sobre la arquitectura del sistema completo, terminando con los resultados experimentales obtenidos de las aplicaciones de ejemplo. En el

capítulo 5 se presentan las conclusiones y comentarios sobre futuras líneas de investigación basadas en este trabajo. Finalmente, en el anexo A se detallan las características del hardware diseñado para la implementación de las aplicaciones y consideraciones sobre métodos de medición empleados en las pruebas.

2 ESTADO DEL ARTE

Sistemas empotrados

La fuerte evolución que la microelectrónica analógica y digital ha tenido en los últimos años ha propiciado el desarrollo de una amplia y potente gama de microprocesadores, reduciendo a la vez su tamaño y consumo. Uno de los campos que más se ha beneficiado de estas mejoras es el de los sistemas empotrados, que ahora pueden disponer de procesadores capaces de afrontar aplicaciones antes impensables para este tipo de sistemas como, por ejemplo, el procesamiento en tiempo real de señales de vídeo. Este aumento de las capacidades básicas (velocidad de procesamiento, memoria, etc.) ha ido acompañado, además, de la incorporación de un mayor número y diversidad de periféricos de conectividad, lo que ha fomentado la cooperación en red de estos equipos y su integración con servicios de mayor envergadura ofrecidos a través de Internet.

Existen varios tipos de procesadores que pueden constituir el núcleo de un sistema empotrado, de los cuales pueden destacarse los siguientes:

- **Procesadores digitales de señal (DSPs)**. Se trata de procesadores con un juego de instrucciones complejo (CISC) que pueden llegar a funcionar a tasas elevadas de instrucciones por segundo. Disponen de instrucciones específicas que aceleran las operaciones básicas utilizadas en el tratamiento digital de señales. Los DSPs actuales han evolucionado internamente hacia arquitecturas complejas que mejoran el rendimiento en cada ciclo de reloj y aumentan el grado de paralelismo. Un ejemplo interesante de esta característica es la arquitectura VLIW (Very Long Instruction Word), utilizada en algunos DSPs de Texas Instruments®, como en el caso de la familia C6000, que empaqueta las instrucciones en grupos de hasta ocho instrucciones que se ejecutan en paralelo gracias a que las estructuras que componen la unidad aritmético-lógica son replicadas. De esta forma, un DSP funcionando, por ejemplo, a 600 MHz, podría alcanzar hasta los 4800 MMACS (millones de instrucciones de multiplicación/acumulación por segundo)[#]. Los entornos y herramientas de desarrollo para DSPs también han evolucionado significativamente en los últimos años, mejorando en varios aspectos

La velocidad efectiva es variable y depende de la optimización que el compilador pueda hacer sobre el código.

como la calidad de las herramientas de compilación[*], capacidad de depuración o incluso la disponibilidad de código reutilizable en forma de librerías y sistemas operativos muy básicos.

- **Arquitecturas procesadoras System-on-a-Chip (SoC)**. Se trata de procesadores con un juego de instrucciones reducido (RISC) que incorporan funcionalidades concretas a través de sus periféricos. A lo largo del tiempo, estos procesadores han ido ganando en prestaciones y periféricos dando lugar a una amplia gama de dispositivos entre los que destacan los procesadores ARM[§] por su gran difusión. A pesar de su simplicidad, este tipo de procesadores poseen otras ventajas que han propiciado su expansión, especialmente en el segmento de los sistemas empotrados para terminales móviles. Entre estas ventajas cabe destacar las siguientes:
  - Bajo consumo. Esencial en sistemas alimentados por baterías.
  - Diversidad de periféricos. Muchos de ellos enfocados a la conectividad en redes de comunicaciones.
  - Compatibilidad con núcleos anteriores y de otros fabricantes. Facilita la reutilización de código.
  - Arquitectura soportada en los principales sistemas operativos. En el caso de GNU/Linux, la arquitectura ARM ha sido integrada en la línea de desarrollo principal del kernel (*mainline*) desde hace tiempo.
  - Amplio soporte. Existen comunidades de programadores dedicadas a estos procesadores que continuamente aportan drivers y mejoras, especialmente en el ámbito de las distribuciones GNU/Linux.
  - Bajo coste.
- **Lógica programable**. Los dispositivos lógicos programables y, en concreto, las FPGAs (Field Programmable Gate Array) constituyen la solución más flexible desde el punto de vista del hardware. Las FPGAs también han sufrido una gran evolución en los últimos años en varios aspectos:
  - Han incrementado la densidad de bloques lógicos, pudiendo albergar en su interior incluso implementaciones de microprocesadores.
  - Han aumentado la velociad de funcionamiento.

---

* Muchas de estas herramientas abstraen al programador de la complejidad de programar en lenguaje ensamblador con un juego tan extenso de instrucciones, permitiendo la compilación de código en C/C++

§ ARM no fabrica procesadores propiamente sino que comercializa el diseño del núcleo de estos procesadores a fabricantes de semiconductores, que les añaden periféricos y son los que fabrican los dispositivos comerciales.

- Incluyen bloques especializados en funciones demandadas como, por ejemplo, memorias RAM.
- Las herramientas de desarrollo han mejorado notablemente, facilitando el desarrollo, la depuración y la reutilización de diseños.

Como se puede apreciar, la evolución en el ámbito de los sistemas empotrados abre la puerta al desarrollo de nuevas aplicaciones que, anteriormente y por su complejidad, eran inviables para sistemas de estas características. En este sentido, los procesadores ARM, gracias en parte a su soporte en los principales sistemas operativos, están propiciando que algunas de las aplicaciones y servicios, que habitualmente se ejecutaban sobre ordenadores, se estén adaptando para su ejecución en sistemas empotrados. Es especialmente reseñable el éxito de los sistemas operativos libres en el ámbito de los sistemas empotrados, debido, entre otras cosas, a la flexibilidad y al control sobre el desarrollo que éstos ofrecen. Concretamente GNU/Linux es, por una amplia mayoría, el sistema operativo más utilizado en sistemas empotrados por las empresas españolas [3].

El apoyo que se está dando desde algunas instituciones públicas constituye una prueba de la importancia del desarrollo de los sistemas empotrados. Mencionar, en este sentido, las plataformas para la promoción de los sistemas empotrados denominadas PROMETEO [4], a nivel nacional y ARTEMIS [5], a nivel europeo. El objetivo de estas iniciativas es extender el desarrollo y uso de los sistemas empotrados en ámbitos de aplicación muy diversos. En el caso particular de las aplicaciones para entornos urbanos inteligentes, estos sistemas constituyen la base hardware para dar soporte a los servicios.

Sistemas y aplicaciones para entornos urbanos inteligentes

A la hora de estudiar las propuestas de sistemas y aplicaciones para entornos urbanos inteligentes que se están realizando a nivel de investigación se aprecia una separación en dos grandes líneas. Por un lado, existen iniciativas orientadas a aplicaciones para sistemas de transporte inteligentes (ITS) y, por otro lado, aplicaciones de inteligencia ambiental enfocadas a proporcionar servicios a usuarios de a pie, con un ámbito de actuación más reducido. En este apartado se describirán las principales iniciativas en cada una de estas líneas.

Sistemas inteligentes de transporte

Dentro de las iniciativas tecnológicas para sistemas inteligentes de transporte, la mayor parte del esfuerzo de investigación se centra en el tráfico por carretera, aunque existen también importantes iniciativas para otros medios de transporte, como el sistema ERTMS (European Rail Traffic Management System) [6], que se apoya en GSM-R, una variante de la tecnología asociada a la telefonía móvil que pretende establecer una serie de estándares a nivel europeo que garanticen la interoperabilidad del transporte ferroviario.

Sobre el transporte por carretera, por su naturaleza, influyen muchos más factores, lo que complica su caracterización y correcta gestión. No obstante, en los últimos años se han desarrollado nuevos sistemas capaces de obtener mayor información sobre los parámetros de tráfico, muchos de ellos basados en técnicas de visión artificial. Atendiendo a su funcionalidad, estos equipos se pueden clasificar en dos grandes grupos:

- Sistemas de detección y clasificación
- Sistemas de detección de infracciones (enforcement)

Los sistemas de detección y clasificación se emplean tanto en vías urbanas como interurbanas. Los datos se obtienen por cada carril, siempre que hayan sido definidas correctamente las regiones de detección y tienen un carácter macroscópico. Entre las variables que pueden ser estudiadas están:

- Intensidad
- Velocidad
- Distancia entre vehículos
- Longitud de los vehículos
- Nivel de servicio
- Contaje de vehículos
- Longitud de colas

Si bien este tipo de información se podía obtener con sistemas de detección convencionales (basados en espiras, detectores piezoeléctricos, etc.), la introducción de equipos basados en visión artificial ha permitido que hoy en día se pueda obtener información valiosa sobre situaciones que puedan afectar al tráfico, pudiendo prevenir accidentes. Esto ha dado lugar a los sistemas de detección de incidentes, que emiten avisos ante situaciones como las que se

enumeran a continuación:

- Vehículo parado
- Conducción en sentido contrario
- Presencia de peatones
- Pérdida de la carga
- Fuego y humo
- Colas
- Descenso de velocidad

Las aplicaciones de los sistemas de detección de infracciones son bastante variadas y están teniendo bastante éxito en los últimos años. Para estos equipos, las técnicas de reconocimiento óptico de caracteres (OCR) son fundamentales, ya que les permiten reconocer las matrículas de los vehículos. Una vez obtenida esta información, los distintos usos que se pueden hacer de ella dan lugar a las distintas aplicaciones. Así, por ejemplo, se puede emplear el dato de la matrícula de un vehículo que haya rebasado un semáforo en rojo para emitir un informe a un centro de control que pueda, a su vez, derivar en una sanción. La información de las matrículas también se puede utilizar para restringir el acceso a ciertas zonas como, por ejemplo, un parking o el centro de una ciudad. En este caso, los equipos contrastan la matrícula obtenida con una base de datos para permitir o denegar el acceso.

La información de la matrícula puede servir incluso para detectar excesos de velocidad. A diferencia de los equipos basados en Radar, que detectan la velocidad instantánea en un punto, los equipos basados en visión artificial pueden medir velocidades medias tomando referencias temporales para un mismo vehículo (una misma matrícula) en dos puntos. Para que el sistema funcione se debe conocer de antemano la distancia entre los puntos de medición y mantener una estricta sincronización entre los relojes de ambos equipos. Estos dispositivos, denominados radares de tramo, están empezando a adquirir cierta notoriedad, instalándose en las principales carreteras españolas.

Los equipos que llevan a cabo todas estas aplicaciones son, en definitiva, sistemas empotrados y, como se mencionó anteriormente, sus capacidades de conexión a redes han ido creciendo en los últimos años. En el caso de los equipos de tráfico, la comunicación se produce fundamentalmente entre el equipo y un centro de control, aunque la valiosa información que estos equipos producen puede ser aprovechada no sólo por las administraciones, sino también por los usuarios. Una información en tiempo real sobre la formación de atascos, por ejemplo, puede permitir a los conductores tomar itinerarios

alternativos a tiempo.

Las tecnologías de comunicación inalámbricas, por sus características, se postulan como la solución más adecuada para esa comunicación en tiempo real con los usuarios. Así lo han entendido diversos organismos internacionales con competencias en la materia, que tratan de establecer una serie de estándares para este tipo de comunicaciones a varios niveles. Una de las instituciones involucradas es el IEEE, que ha aprobado recientemente el estándar IEEE 802.11p. Este estándar supone una corrección al estándar IEEE 802.11 sobre redes inalámbricas de área local para dar soporte a las comunicaciones entre vehículos (V2V) y entre los vehículos y la infraestructura (V2I). Por este motivo, al estándar IEEE 802.11p también se le conoce como WAVE (Wireless Access in Vehicular Environment). Las principales modificaciones propuestas en IEEE 802.11p se describen en el capítulo 3. El proyecto de definición del estándar se origina a raíz de la decisión de la Comisión Federal de Comunicaciones de Estados Unidos (FCC) de reservar una banda en torno a los 5,9GHz (de los 5,850GHz a los 5,925GHz) para las comunicaciones dedicadas de corto alcance (DSRC). La banda DSRC fue reservada en Estados Unidos en 1999 para uso exclusivo de comunicaciones V2V y V2I [7]. Su objetivo es habilitar aplicaciones de seguridad públicas que permitan evitar accidentes de tráfico y mejorar el flujo de vehículos. En Europa, la Conferencia Europea de Administraciones de Correos y Telecomunicaciones (CEPT) reservó en 2008 una banda similar (desde los 5,875GHz a los 5,905GHz) con el mismo propósito.

Pero el estándar 802.11p no es la única iniciativa del IEEE en este ámbito. De hecho, 802.11p es sólo una parte de un conjunto de estándares recogidos en el grupo IEEE 1609, que definen una serie de protocolos a varios niveles para comunicaciones DSRC. IEEE 802.11p está limitado por el alcance de IEEE 802.11, que consta solamente de la capa de enlace (MAC) y la capa física (PHY). Sin embargo, estándares como IEEE 1609.3 se encargan de tareas de niveles superiores como el establecimiento y gestión de las conexiones. En la figura 1 se muestra la estructura de capas de IEEE 1609 y su relación con WAVE.

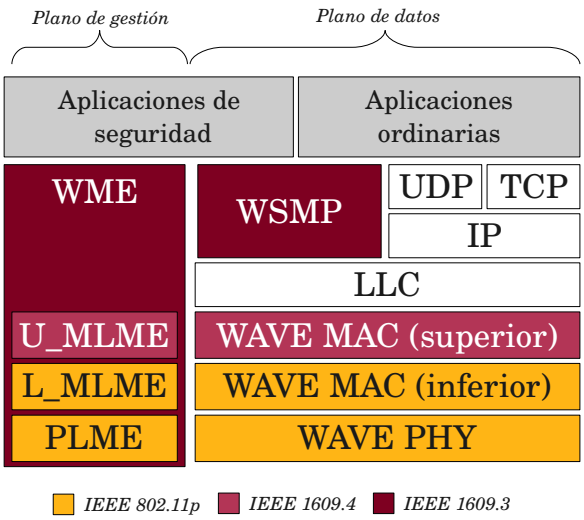

Figura 1: Estructura de capas de IEEE 1609

Según la arquitectura propuesta en IEEE 1609, las aplicaciones ordinarias pueden hacer uso de protocolos habituales como TCP/IP, aunque aparecen nuevos protocolos como WSMP (WAVE Short Message Protocol) para el intercambio rápido y fiable de mensajes, más adecuado para aplicaciones de seguridad donde un tiempo de latencia bajo es esencial.

Por otra parte, existe un grupo de trabajo del ISO (International Organization for Standardization) denominado TC204 WG16 encargado del desarrollo de una familia de estándares conocidos como CALM (Communications Access for Land Mobiles) [8]. Esta familia de estándares especifican una arquitectura común, protocolos de red e interfaces para las comunicaciones V2V y V2I. El concepto de CALM es más amplio que el de IEEE 802.11p o IEEE 1609. De hecho, CALM admite IEEE 802.11p como una de las tecnologías inalámbricas de comunicación, aunque se prevé que soporte muchas otras como:

- GPRS (2G)
- UMTS (3G)
- WiFi
- Wimax
- DAB
- GPS

y también algunas definidas específicamente para CALM como:

- M5
- Infrarrojos
- Millimetre

Dentro de CALM, el estándar ISO 21217 define la arquitectura del sistema, que se muestra en la figura 2.

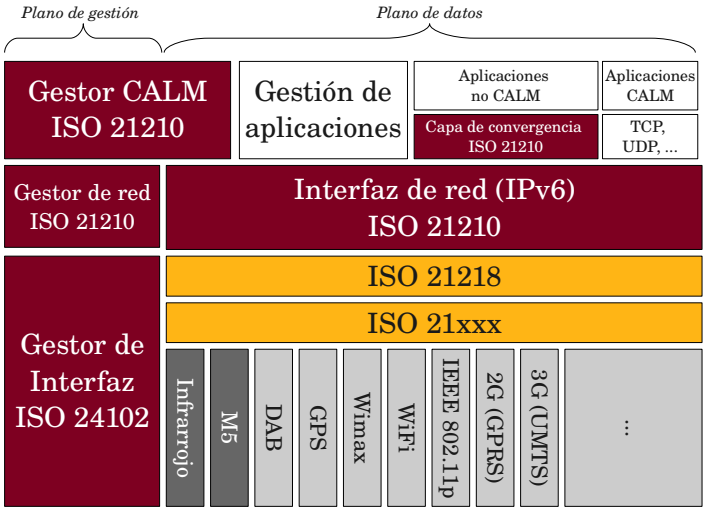

Figura 2: Estructura de capas de CALM

CALM define diversas capas de adaptación al medio de transmisión, pero mantiene otros elementos en común, principalmente a partir del nivel de red, que está basado fundamentalmente en IPv6. De esta manera, CALM es compatible con los servicios de Internet y no sufre las restricciones de direccionamiento de IPv4. Para las aplicaciones de seguridad donde el tiempo es un factor crítico, CALM dispone de modos de operación rápidos para mensajes cortos que no necesitan usar IPv6.

Por otra parte, existen iniciativas más concretas a nivel europeo financiadas por los programas marco de la Comisión Europea. Entre estas iniciativas cabe destacar tres:

- SAFESPOT [9]. Está enfocado fundamentalmente a la comunicación V2V, creando redes cooperativas dinámicas para la comunicación entre vehículos y la infraestructura. La seguridad es uno de los

objetivos de este proyecto, desarrollando aplicaciones para evitar accidentes.

- CVIS [10]. Este proyecto involucra tanto comunicación V2V como V2I prestando atención también a otros aspectos, como a los sistemas autónomos o a la integración con los sistemas internos del vehículo.
- COOPERS [11]. Está enfocado principalmente a la comunicación V2I. Su objetivo es la gestión cooperativa del tráfico mediante el desarrollo de aplicaciones telemáticas innovadoras. Al igual que en otros proyectos, la seguridad es clave en el desarrollo de las aplicaciones.

Sistemas para entornos inteligentes

En el caso de los sistemas para entornos inteligentes, las iniciativas existentes no tienen un nivel de estandarización comparable al caso de las aplicaciones ITS. Por el contrario, hay cierta dispersión en cuanto a la finalidad de la aplicación, las tecnologías empleadas y la integración con otros sistemas. Algunas de estas iniciativas están orientadas a ofrecer información dependiente del contexto [12], y su uso está dirigido a varios colectivos como personas con discapacidad visual, personas con dificultades de comprensión o simplemente turistas. La información que ofrecen también puede variar en función del tipo de usuario. Así, por ejemplo, los sistemas de información turísticos ofrecen información concreta sobre el lugar en el que se encuentra el visitante, mientras que la información para invidentes suele ser de orientación y localización.

En líneas generales, la mayoría de los sistemas existentes se basan en una arquitectura Baliza – Terminal móvil, aunque las tecnologías empleadas varían de uno a otro, siendo Bluetooth y WiFi las tecnologías más empleadas. Algunos sistemas, como Talking-Points [13][14] o InfoSign [15] combinan Bluetooth, para la detección de los terminales y la transmisión de un identificador, con accesos a Internet a través de la infraestructura de telefonía móvil, usando GPRS o UMTS. De este modo, el terminal puede consultar en Internet la información asociada a dicho identificador. InfoSign, además, plantea otras alternativas a la ubicación de la información, como, por ejemplo, el almacenamiento local en el terminal o la transmisión de la información directamente por Bluetooth. En cuanto al tipo de usuario, tanto Talking-Points como InfoSign están enfocados a personas, invidentes o no, que necesitan información de interés sobre el punto en el que se encuentran, ya sea de tipo turístico, educativo o comercial. Sin embargo, existen otros sistemas como URNA [16][17], más enfocado a personas invidentes y especializado en orientación en exteriores. Concretamente, las aplicaciones de

URNA están relacionadas con el tráfico y la orientación, ofreciendo instrucciones a los usuarios para cruzar la calle cuando el semáforo está en verde, por ejemplo. URNA también emplea Bluetooth como tecnología de comunicación.

Como se mencionó anteriormente, la información de orientación y localización puede resultar especialmente útil para el colectivo de invidentes, y son precisamente los sistemas de localización y posicionamiento una de las áreas que mayor interés ha despertado entre los investigadores. Son numerosos los estudios e iniciativas al respecto. Aunque la tecnología GPS es la más adecuada para estas aplicaciones, no siempre es posible recibir la señal de los satélites como, por ejemplo, en el interior de edificios. En estos casos se recurre a distintas tecnologías como infrarrojos [18], ultrasonidos [19][20][21], Bluetooth [22][23] o una combinación de ellas [24]. Dentro de los sistemas de localización por Bluetooth la mayoría de los autores recurre al parámetro RSSI (Received Signal Strength Indicator) como base para la estimación de la posición mediante técnicas de triangulación [25][26]. El problema asociado al RSSI, como se verá más adelante, es su fuerte no-linealidad, lo que hace que la correspondencia con la potencia recibida no sea sencilla de determinar. Además, la variabilidad del RSSI es bastante amplia, como se muestra en [27]. En [28] se propone la utilización de filtros de Kalman para el procesamiento del RSSI, pero aún así, los resultados siguen teniendo un error significativo. Otros estudios, como [29] y [30], sugieren usar el parámetro "Link Quality" en lugar del RSSI por tener una mejor correlación con la potencia recibida. El parámetro que mayor precisión ofrece para determinar la posición es la potencia recibida, aunque esta información no suele estar disponible en la mayoría de los dispositivos Bluetooth. Por este motivo, en [31] se sugiere la utilización de un circuito integrado específico para realizar una correlación y determinar la distancia con mayor precisión. En cualquier caso, la potencia recibida está afectada por múltiples factores en el canal principalmente derivados de los obstáculos existentes en los entornos urbanos (paredes, techos, elementos metálicos, etc.), lo que impide crear un modelo válido que relacione la distancia con la potencia recibida en todos los entornos.

Middleware y soporte a los servicios

La diversificación de los servicios en entornos urbanos inteligentes obliga a realizar un cuidadoso planteamiento en la forma de ofrecerlos, de modo que sobre una misma plataforma puedan acomodarse el mayor número de tipos de servicio. No tiene mucho sentido implantar múltiples sistemas con funcionalidades específicas por los inconvenientes que plantean:

- Incompatibilidad de sistemas
- Dificultades de uso para el usuario
- Exceso de equipamiento

Este problema no es nuevo ni específico de los entornos urbanos inteligentes. Desde el ámbito informático, hace años que se vienen empleando soluciones que pasan por establecer una base común lo suficientemente genérica para dar soporte a esos servicios. Esta base común da lugar al concepto de middleware. Hoy en día existen diversas plataformas middleware que pueden ser adaptadas a las aplicaciones para entornos urbanos inteligentes y, de hecho, algunos de los sistemas comentados anteriormente las utilizan. Entre las soluciones middleware empleadas destaca el uso de CORBA, aunque también se utiliza la invocación de métodos remotos de Java o los servicios web.

Arquitecturas de red

La organización de la red y el modo de intercambio de información influyen en el tipo de middleware utilizado. Atendiendo a este criterio se observan tres grandes grupos de sistemas:

- Los basados en puentes
- Los sistemas de intercambio entre pares (P2P)
- Los basados en redes MANET (Mobile Ad-hoc NETworks)

Arquitectura basada en puentes

Por su origen, el escenario habitual de las plataformas middleware son las redes TCP/IP, a menudo interconectadas a través de una gran infraestructura como Internet, donde las transacciones las ejecutan PCs y servidores. En una primera aproximación, la adaptación a los entornos urbanos inteligentes consistiría en conectar los terminales móviles a estas redes de modo que pudieran hacer uso de los servicios como si de un PC se tratase. Esto no siempre es posible o, al menos, no directamente. Por este motivo, algunos sistemas introducen el concepto de "puente". Un puente actúa como una pasarela entre la red de la infraestructura fija, normalmente cableada, y la red inalámbrica que se utilice para conectar con los terminales móviles.

Como se muestra en la figura 3, desde el punto de vista de los equipos,

uno de los extremos del puente sería el propio terminal móvil, mientras que el otro extremo lo constituye normalmente una baliza o punto de acceso que, a su vez, está conectada a la infraestructura fija. Esta baliza debe realizar las siguientes funciones:

- Actuar de intermediario entre el terminal móvil y las máquinas de la infraestructura donde residen los servicios, de modo que tanto uno como otro vean al otro extremo del servicio como un elemento local a su red.
- Resolver los problemas de direccionamiento dinámico en ambas redes.
- Gestionar el establecimiento y liberación de conexiones, especialmente en el lado de la infraestructura inalámbrica.
- Realizar las adaptaciones pertinentes sobre las capas de transporte para cumplir los requisitos técnicos del middleware.
- Gestionar la transferencia del punto de acceso de un terminal a otra baliza si se dan las condiciones necesarias (opcional).

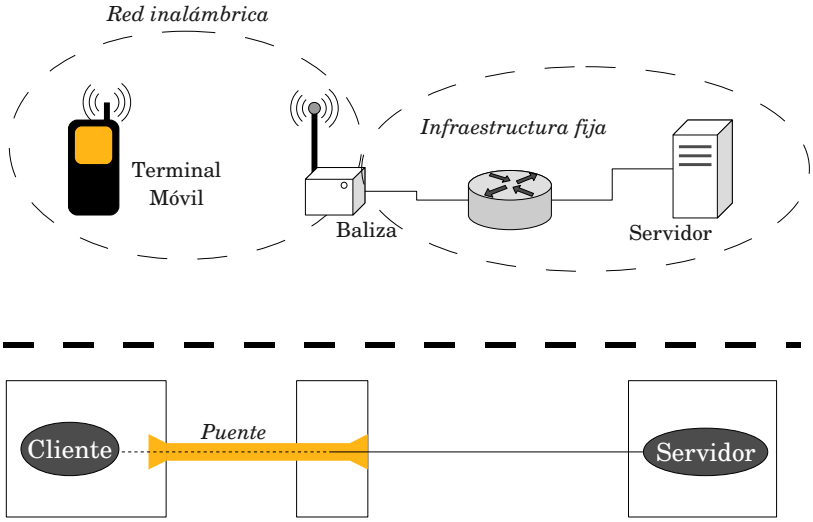

Figura 3: Arquitectura de red basada en puentes

Una vez establecido el puente, la comunicación se desarrolla siguiendo el modelo cliente-servidor, habitual en estas plataformas, donde los clientes suelen ubicarse normalmente en el lado del terminal móvil y los servicios en el lado de la infraestructura.

Dentro de los sistemas que utilizan puentes se pueden encontrar dos grandes iniciativas:

- Wireless CORBA
- Esperanto

La primera iniciativa surge de una especificación del organismo que coordina el desarrollo de CORBA, una plataforma middleware bastante popular y con amplia difusión. La especificación de Wireless CORBA* permite la utilización de servicios CORBA por parte de terminales con acceso inalámbrico sin que implique cambios en la programación de los servicios. Como consecuencia de la especificación aparecen dos elementos adicionales encargados de la formación y liberación de los puentes: TerminalBridge en el lado del terminal móvil y AccessBridge en el lado de la baliza. Además de implementar las funciones básicas del puente citadas anteriormente, estos elementos permiten el traspaso de conexiones activas con un terminal entre diferentes balizas. Es lo que se conoce como "handoff" [32].

La única implementación práctica de Wireless CORBA conocida hasta la fecha se ha realizado sobre MICO, una plataforma CORBA de código abierto. Dicha implementación ya incluye las adaptaciones necesarias para desplegar CORBA sobre Bluetooth, aunque en [33] se propone una capa alternativa a nivel de transporte que tiene en cuenta aspectos relacionados con la calidad del servicio (QoS) a la hora de establecer una conexión a un servicio, verificando previamente la capacidad para soportarlo. Otros estudios, como [34], sugieren el uso de registros adicionales de terminales en AccessBridge para mejorar los tiempos de acceso a medio plazo, o incluso variantes de TerminalBridge para permitir el establecimiento de puentes con otros terminales en lugar de hacerlo con balizas, como se describe en [35].

También existen iniciativas muy similares a Wireless CORBA, aunque sin seguir la especificación y enfocada a la tecnología inalámbrica WiFi, como ALICE [36]. Esta iniciativa comparte con Wireless CORBA algunas estrategias para resolver ciertos problemas, como la intervención en los identificadores de servicio CORBA (IOR), aunque difiere en otros aspectos, como la utilización de DHCP para la obtención de una dirección IP en la red (Wireless CORBA sobre Bluetooth no necesita hacer esto). La implementación de ALICE se ha realizado sobre ORBacus, otra plataforma CORBA.

Por otra parte, ESPERANTO [37] plantea una estructura interna diferente, aunque permite el uso de CORBA en la parte de la infraestructura fija. Una de las características más destacables de ESPERANTO es el uso de

* Las características técnicas de Wireless CORBA se detallan en el capítulo 3.

unos "sockets" especiales denominados NCSOCKS (Nomadic Computing SOCKetS), que abstraen de la tecnología inalámbrica utilizada (WiFi, Bluetooth o IrDA). La arquitectura se basa en el uso de "mediadores" de manera que un terminal se conecta a un mediador a través de una red inalámbrica, y éste se encarga de establecer comunicación con otros mediadores remotos a través de procedimientos remotos, CORBA u otras plataformas middleware. Las invocaciones de métodos se encapsulan en "tuplas", un sistema que aparece en iniciativas anteriores como LIME [38]. ESPERANTO también define un conjunto de primitivas básicas como "read", "write" o "take" para llevar a cabo las operaciones de los servicios. Otro detalle interesante es el uso de E-IDL, una extensión del lenguaje IDL empleado en CORBA para la descripción de servicios. El descubrimiento de servicios puede realizarse tanto de forma proactiva como reactiva [39].

Aunque ESPERANTO se plantea como una alternativa a Wireless CORBA, los estudios comparativos demuestran que los tiempos de latencia son mayores en ESPERANTO debido, en parte, a que las interacciones entre mediadores son complejas [40].

Arquitectura P2P

Los sistemas basados en arquitectura P2P (Peer to Peer) permiten la conexión espontánea entre dispositivos del mismo nivel jerárquico para el intercambio de información, funcionando de manera similar a como lo hacen las redes P2P convencionales a través de Internet [41]. Este hecho hace que los sistemas con arquitectura P2P estén más enfocados a ofrecer servicios que comparten ficheros. Según la arquitectura P2P, el tipo de conexión predominante se da directamente entre terminales, aunque también se prevé la conexión con la infraestructura. En estos casos la infraestructura juega el papel de fuente de contenidos, y las conexiones entre terminales ayudan a diseminarlos [42]. Se pueden utilizar distintas tecnologías inalámbricas, como Bluetooth o WiFi, aunque en todos los casos los terminales deben tener activados los mecanismos de descubrimiento de dispositivos. En [43] se da un paso más en el modo de difusión de los contenidos y se plantea el concepto de "nubes de usuarios", de manera que los contenidos que circulan por una nube pueden propagarse a otra nube si alguno de sus miembros se traslada. Mecanismos basados en contexto actúan de filtro para la propagación de esta información, seleccionando a través de qué terminales debe propagarse cada información.

Redes MANET

Las redes MANET (Mobile Ad hoc NETworks) se forman mediante conexiones inalámbricas entre terminales sin necesidad de que medie ninguna infraestructura fija. En cierto modo, algunas de las soluciones comentadas pueden encuadrarse dentro del concepto de MANET. Uno de los principales problemas que afrontan este tipo de redes es el enrutamiento de la información, lo que ha dado lugar a numerosos estudios al respecto. En una red MANET la comunicación no se limita al intercambio punto a punto habitual que ofrecen las tecnologías inalámbricas en modo ad hoc, sino que se pretende permitir la comunicación simultánea con otros terminales dentro de la red. Para lograr esto, los terminales deben actuar de tres formas diferentes: como fuente de la información, como destino de la información o como encaminadores. A diferencia de las redes TCP/IP convencionales, la dificultad del encaminamiento radica en la gran variabilidad de la red, donde constantemente entran y salen dispositivos que no se conocen a priori. La solución pasa por el desarrollo de protocolos de encaminamiento dinámico, y en esta tarea están involucrados, además de investigadores, organismos internacionales como el IETF, a través de su grupo de trabajo MANET. Actualmente existen diversas iniciativas basadas en protocolos como LUNAR [44] o DSR [45], aunque también existen otras basadas en protocolos más conocidos como Mobile IP [46]. Algunos de estos esquemas de funcionamiento también han sido combinados con la arquitectura basada en puentes, permitiendo la comunicación ad hoc entre terminales usando Wireless CORBA, como se propone en [35].

Calidad de servicio

Como se ha visto anteriormente, parte de las líneas de investigación en sistemas para entornos urbanos inteligentes se han centrado en implantar plataformas middleware sobre las arquitecturas de red descritas. En algunos casos se han transportado plataformas que originariamente fueron diseñadas para transacciones entre ordenadores. La adaptación a este nuevo entorno requiere un estudio cuidadoso de las diferencias, tanto en las características de las redes como en las características de plataformas hardware que ejecutan las transacciones. Diversos factores como la potencia de señal de los radioenlaces, la tasa de errores (BER) o incluso la capacidad gráfica del terminal afectan a la calidad del servicio. Por este motivo, los aspectos de calidad del servicio (QoS) han dado lugar a una serie de estudios en los que se trata sobre la implantación de mecanismos QoS en las adaptaciones de las plataformas middleware. En algunos casos, las propias tecnologías

inalámbricas ya disponen de mecanismos de QoS, por lo que puede ser conveniente delegar algunas de estas funciones en la propia torre de protocolos del enlace inalámbrico mientras que otras funciones deben derivarse a las capas middleware, tal y como se indica en [47]. En el caso particular de las redes Bluetooth, existen diversos mecanismos QoS que afectan a la formación de redes de área personal (PAN) e incluso a la formación de redes más extensas a través de la unión de múltiples redes de este tipo. En [48] se realiza un amplio estudio en este sentido. Otro de los factores a tener en cuenta al usar Bluetooth es la interferencia con otros tipos de redes que transmiten en la misma banda. Por ello, en [49] se analizan las restricciones sobre la calidad del servicio derivadas del efecto de dichas interferencias sobre el througput y los tiempos de conexión a la red (del orden de 2,5 segundos).

Para solucionar los problemas relacionados con la heterogeneidad derivados de las capacidades de procesamiento y representación de la información de los terminales, se plantea el concepto de "middleware reflectivo" [50]. El "middleware reflectivo" analiza su propio estado y sus capacidades y las toma en consideración para adaptar su comportamiento. De este modo, por ejemplo, un servicio de distribución de imágenes podría convertir éstas a blanco y negro de antemano si el ancho de banda disponible es escaso. En [51] se sugiere recurrir incluso a "funciones de utilidad" en los casos en los que puedan aparecer conflictos entre los requisitos de los servicios a la hora de aplicar este tipo de middleware.

Consideraciones

De la observación de las iniciativas de investigación en el ámbito de los entornos inteligentes se desprende que el desarrollo de estos sistemas gira en torno a tres elementos clave: los sistemas empotrados, las redes inalámbricas y el middleware. Estos tres elementos, convenientemente combinados, constituyen la base para la armonización de los servicios.

Se aprecia también un notable esfuerzo de estandarización en el ámbito concreto de los sistemas inteligentes de transporte, especialmente en el transporte por carretera, a diferencia de los sistemas para entornos urbanos inteligentes, donde existe una mayor dispersión y especificidad. Entre las soluciones más genéricas destaca el uso de plataformas middleware y, en particular, CORBA.

Por el tipo de servicios que se pueden ofrecer en entornos urbanos

inteligentes, la arquitectura que mejor se adapta es la basada en puentes. La arquitectura P2P ofrece mejores resultados en el intercambio de contenidos estáticos, pero no es eficiente para un intercambio de información dinámica. Por otra parte, las características que ofrecen las redes MANET en cuanto a las posibilidades de creación y extensión de redes ad hoc no aportan una ventaja sustancial a los servicios habituales en entornos urbanos inteligentes, que pueden resolverse en la gran mayoría de las situaciones mediante una arquitectura Baliza – Terminal donde los servidores están ubicados en el lado de la infraestructura y los clientes en el lado del terminal. En cualquier caso, la comunicación directa entre terminales también se puede hacer indirectamente en niveles superiores* con la arquitectura Baliza – Terminal.

Dentro de las iniciativas con arquitectura basada en puentes, Wireless CORBA ofrece un mayor rendimiento y menor complejidad que ESPERANTO. El hecho de estar basado en una plataforma middleware bien conocida como es CORBA facilita el desarrollo de nuevos servicios y la integración con otros ya existentes. Por todos estos motivos, en esta Tesis se ha establecido Wireless CORBA como punto de partida.

No obstante, Wireless CORBA no resuelve todos los problemas que se plantean a la hora de ofrecer servicios en entornos urbanos inteligentes. Existen ciertos factores característicos de las redes en entornos urbanos inteligentes que hay que tener en cuenta:

- No se conoce de antemano el número de terminales que se conectarán a la red.
- Los terminales tampoco conocen de antemano los servicios que ofrece la red a la que se conectan.
- Los terminales viajan entrando y saliendo constantemente de las áreas de cobertura de las balizas. Además, los terminales pueden pasar de una red a otra perteneciente a un dominio completamente distinto.

Considerando estos factores surge la necesidad de disponer de una entidad que se encargue de gestionar el establecimiento y liberación de los puentes automáticamente, atendiendo a ciertos criterios. De esta forma se facilita la selección y acceso a los servicios en función del contexto y/o las preferencias del usuario. Otra de las cuestiones que Wireless CORBA no resuelve es la detección automática de dispositivos. En este sentido, la nueva

* Aplicable siempre que la topología de la red inalámbrica sea punto a punto entre los esclavos y el maestro, como ocurre en algunas tecnologías como Bluetooth.

entidad debe asumir las funciones de detección teniendo en cuenta las particularidades de la tecnología inalámbrica empleada y los efectos que pueda tener sobre el rendimiento de los enlaces o el consumo energético, máxime cuando se trata de terminales alimentados por baterías.

Tampoco hay que olvidar los mecanismos de calidad del servicio (QoS), que permiten introducir mejoras en aspectos como el uso eficiente de los recursos o una correcta gestión de los traspasos entre balizas (handoff). Wireless CORBA no implementa estos mecanismos aunque, como se adelantó anteriormente en este capítulo, es posible aprovechar algunos de los mecanismos que ya ofrecen las propias tecnologías inalámbricas.

Las contribuciones de esta Tesis van dirigidas precisamente a la creación de esa nueva entidad en forma de módulos adicionales y una serie de recomendaciones que hacen posible el uso de Wireless CORBA como base para ofrecer servicios en entornos urbanos inteligentes.

3 GESTIÓN DE COMUNICACIONES INALÁMBRICAS MEDIANTE SISTEMAS EMPOTRADOS

Tecnologías inalámbricas

El desarrollo de las tecnologías inalámbricas ha contribuido de una forma importante a la integración de sistemas empotrados. La incorporación de dispositivos de comunicación inalámbrica en los sistemas empotrados ha favorecido las operaciones de gestión sobre los equipos y ha mejorado notablemente las posibilidades de interacción entre ellos. Los sistemas empotrados utilizados en aplicaciones para entornos urbanos inteligentes no han permanecido ajenos a estas mejoras y, de hecho, las tecnologías inalámbricas constituyen un elemento fundamental sobre el que desarrollan su actividad la gran mayoría de ellos. En la actualidad existe una amplia gama de tecnologías inalámbricas, cada una con unas características determinadas. En los entornos urbanos inteligentes pueden darse muchas situaciones, dependiendo de las cuales, una tecnología inalámbrica puede resultar más o menos adecuada [52]. A lo largo de este apartado se hará un recorrido por distintas tecnologías inalámbricas aplicables al ámbito de los entornos urbanos inteligentes analizando sus principales ventajas e inconvenientes.

WiFi

El IEEE (Institute of Electrical and Electronics Engineers) aprobó la norma 802 en 1990, que normalizaba el funcionamiento de las redes de área local y metropolitanas, y de esta manera se definía el estándar necesario para que los productos de los diferentes fabricantes del mercado fueran compatibles entre sí. Esta primera norma se fue dividiendo sucesivamente en diferentes grupos de trabajo, y en la actualidad se pueden encontrar más de 20. Uno de estos grupos trabajo es el IEEE 802.11, que es específico para las redes locales inalámbricas creadas con la tecnología WiFi. Este grupo fue creado en 1999 por la asociación WECA. El principal problema que resolvió la normalización 802.11 fue la incompatibilidad que existía entre los dispositivos inalámbricos de distintos fabricantes. Dentro del grupo de trabajo IEEE 802.11 se pueden encontrar diversas versiones y complementos, tal y como se muestra en la tabla 1:

Versión	Características
802.11a	WLAN a 54 Mbps en la banda de 5GHz
802.11b	WLAN a 11 Mbps en la banda de 2,4GHz
802.11c	Cruce sin cables
802.11d	"Modo Mundial" adaptación de los requerimientos regionales
802.11e	QoS y extensiones que fluyen a través de 802.11 a/g/h
802.11f	Tránsito para 802.11 a/g/h
802.11g	WLAN a 54 Mbps en la banda de 2,4GHz
802.11h	802.11a con DFS y TCP (Europa)
802.11i	Autentificación y cifrado AES
802.11j	802.11a con canales adicionales por encima de 4-9GHz (Japón)
802.11k	Intercambio de información de capacidad entre clientes y puntos de acceso
802.11m	Mantenimiento, publicación de actualizaciones estándar
802.11n	Nueva generación WLAN hasta 500Mbps en la banda de 2,4GHz
802.11p	Acceso inalámbrico en entornos vehiculares en la banda de 5,9GHz

Tabla 1: Apartados del estándar IEEE 802.11

Entre las variantes del estándar IEEE 802.11 se pueden destacar las siguientes:

- **802.11a**. Trabaja en la banda de 5GHz y utiliza la técnica de transmisión conocida como OFDM (Orthogonal Frequency Division Multiplexing). La gran ventaja es que consigue velocidades de 54Mbps, llegando a alcanzar hasta 108Mbps.
- **802.11b**. Fue introducida en 1999 y su velocidad de transmisión es de 11Mbps. A pesar de su baja velocidad y de operar en la banda de 2,4GHz es muy sensible a la interferencia con otras tecnologías inalámbricas, como por ejemplo Bluetooth.
- **802.11g**. Surgió en 2003 como evolución del estándar 802.11b. Esta norma ofrece velocidades de 54Mbps en la banda de 2,4GHz (no licenciada) y es compatible hacia atrás con los equipos 802.11b, por lo

que ha tenido una gran acogida.

- **802.11n**. Es la versión más reciente y trabaja con velocidades de hasta 500Mbps gracias a la tecnología MIMO (Multiple Input-Multiple Output) que permite incrementar el rendimiento en función del número de antenas que utiliza.
- **802.11p**. Se trata de una reciente extensión de la norma para proporcionar acceso inalámbrico en aplicaciones de sistemas de transporte inteligente (ITS), lo que incluye las comunicaciones V2V y V2I. Utiliza la banda de 5,9GHz y puede alcanzar tasas entre 3 y 27 Mbps [53].

La torre de protocolos que gobierna las comunicaciones WiFi es muy parecida a la de otras tecnologías de redes LAN, tanto cableadas como inalámbricas, puesto que deriva de la norma IEEE 802. En concreto, se mantienen todas las capas a partir del nivel de red y solamente cambian la capa física y la subcapa MAC (Control de Acceso al Medio), perteneciente a la capa de enlace. La siguiente figura ilustra la torre de protocolos a estos niveles:

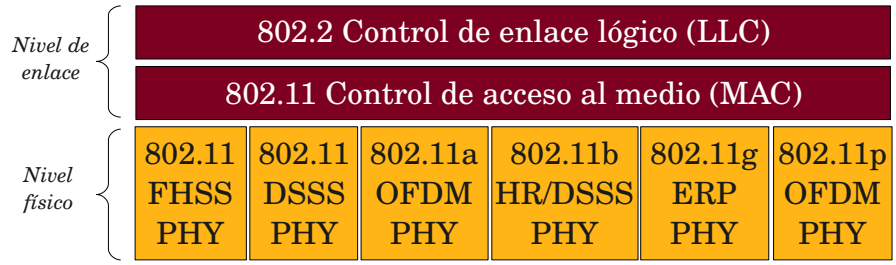

Figura 4: Niveles inferiores de la torre OSI en WiFi

El alcance de las redes WiFi convencionales se sitúa en torno a los 100m, aunque se pueden llegar a conseguir rangos de hasta 50Km con antenas parabólicas apropiadas. La relativa amplitud del alcance de las redes WiFi se debe a su orientación como red de área local, aunque esto también conlleva una mayor potencia radiada para lograr el citado alcance. Concretamente, la potencia nominal de transmisión en equipos WiFi oscila entre los 15 y los 20 dBm. La tecnología WiFi permite formar redes de hasta 2007 equipos para un solo punto de acceso y hace uso de diversos mecanismos tanto de comprobación de los datos (CRC-32) como de encriptación, pudiendo utilizar claves WEP (Wired Equivalent Privacy) o WPA (WiFi Protected Access). Entre estos dos mecanismos de encriptación, WEP ha mostrado ciertas debilidades, pudiendo ser descifrada la clave en cuestión de minutos

mediante la escucha de paquetes de la red. Como alternativa, surgió WPA, recogida en el estándar 802.11i.

Acceso a redes WiFi

Las redes WiFi pueden funcionar en dos modos: infraestructura y ad-hoc, siendo el primero el más habitual. En el modo ad-hoc se establece una conexión punto a punto entre dos equipos con conectividad WiFi. En el modo infraestructura existe uno o varios puntos de acceso y varios terminales acceden a la red mediante conexiones a estos puntos de acceso. En el caso de los estándares 802.11b y 802.11g, estos puntos de acceso transmiten en alguno de los canales en los que se subdivide la banda ISM en 2,4GHz. En Estados Unidos, se permiten hasta 11 canales, mientras que en Europa hay 13 canales disponibles en esta banda. Las interfaces de red inalámbricas sólo escuchan en uno de estos canales en un instante dado. Por tanto, si un terminal móvil quiere detectar todos los puntos de acceso a su alcance, tiene que sintonizar su interfaz en cada uno de estos canales y realizar una búsqueda o escaneo. El estándar IEEE 802.11 define dos técnicas de escaneo: Escaneo activo y pasivo.

En el caso del escaneo activo, un terminal móvil lleva a cabo el siguiente procedimiento por cada canal: En primer lugar, sintoniza la interfaz de red a un canal en particular. El terminal móvil espera la llegada de tramas generadas por otros dispositivos durante un tiempo máximo definido por un temporizador. A continuación, utiliza el procedimiento de acceso al medio definido en 802.11 para acceder al canal y enviar tramas de tipo “Probe Request”. Los puntos de acceso que reciben este tipo de tramas deben responder con tramas de tipo “Probe Response”. El terminal móvil espera durante un cierto tiempo y si no se reciben estas tramas, realiza el mismo procedimiento con el siguiente canal. Si se recibe alguna trama de tipo “Probe Response”, el dispositivo móvil la procesa para su uso posterior. Una trama “Probe Response” contiene información sobre las tasas de datos soportadas, el nombre de la red y la dirección MAC del punto de acceso. Examinando estas tramas un terminal móvil es capaz de detectar los puntos de acceso vecinos y la fuerza de sus señales. La norma IEEE 802.11 no define tiempos por defectos para los temporizadores citados anteriormente. Por tanto, el tiempo exacto requerido para llevar a cabo un escaneo activo puede variar significativamente dependiendo del número de puntos de acceso y de las capacidades del hardware empleado. Como referencia, cabe citar una medición empírica que sitúa este tiempo en torno a los 260 milisegundos [54]. En principio, el envío de las tramas de tipo “Probe Request” no es continuo y depende de la configuración del hardware utilizado. De hecho, como se

muestra en [55], algunos dispositivos envían ráfagas de estas tramas y, a continuación, dejan de enviarlas por un periodo de tiempo denominado "idle interval", que puede ser del orden de varios segundos. Concretamente, en [55] se citan valores que sitúan este intervalo en 8s – 10s y 18s – 20s para distintos dispositivos. Afortunadamente, este parámetro puede modificarse en algunos dispositivos hasta conseguir tiempos tan bajos como 0,3 segundos [54].

El escaneo pasivo fue introducido para reducir la carga de trabajo en los dispositivos móviles y, en consecuencia, ahorrar energía. En un escaneo pasivo el dispositivo escucha cada canal y espera durante un cierto periodo de tiempo. Si a un punto de acceso se le asigna un determinado canal, el terminal móvil debería recibir una trama de tipo "beacon". Todos los puntos de acceso difunden tramas de tipo "beacon" de forma regular para mantener la red. Estas tramas normalmente contienen la misma información que una trama de tipo "Probe Response", es decir, tasas de datos, nombre de la red y dirección MAC del punto de acceso. Examinando las tramas de tipo "beacon" un terminal móvil puede ser capaz de detectar los puntos de acceso vecinos y la fuerza de sus señales. Los puntos de acceso normalmente transmiten una trama de tipo "beacon" cada 100 milisegundos, lo que significa que un terminal móvil debería permanecer escuchando un canal concreto durante, al menos, ese periodo de tiempo para no perder la trama de un punto de acceso desconocido. En total, un escaneo pasivo requiere, al menos, 1,3 segundos para completarse, lo que supone casi cinco veces el tiempo requerido para un escaneo activo. Además, una desventaja añadida al escaneo pasivo es la interrupción de las comunicaciones en curso durante el tiempo que dura el escaneo [54].

El proceso completo de acceso a una red WiFi involucra varias fases, entre las que se incluye el escaneo, la autentificación y la asociación, que se muestran en la figura 5. De las tres primeras fases, la de escaneo es, con diferencia, la que más tiempo consume [55]. Si además, la dirección IP no se conoce de antemano, hay que añadir una fase para obtener una dirección IP válida dentro de la subred. De este proceso se encarga el protocolo DHCP (Dynamic Host Configuration Protocol). Por tanto, el tiempo total de acceso a una red WiFi en estos casos se puede expresar como:

$$t_{acceso} = t_{escaneo} + t_{autentificación} + t_{asociación} + t_{DHCP}$$

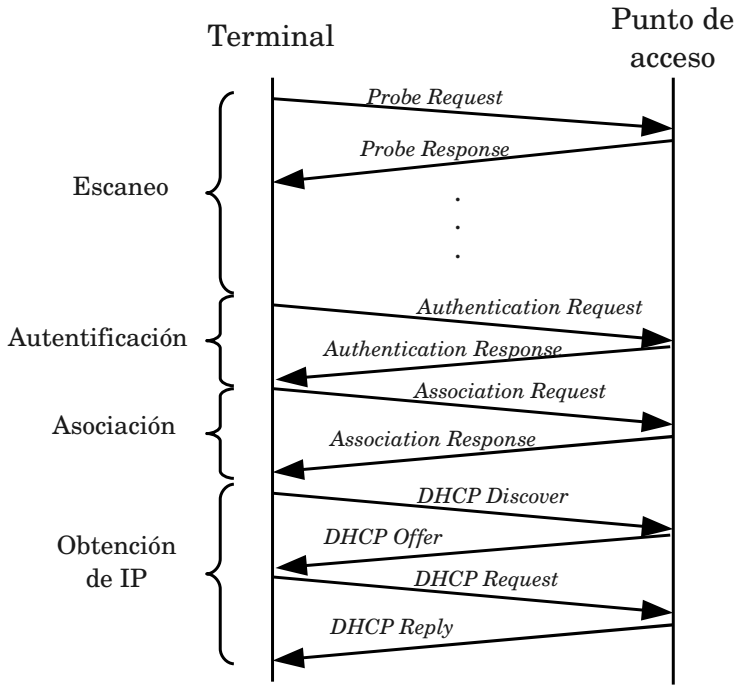

Figura 5: Fases de acceso a una red WiFi

DHCP proporciona direcciones IP a los terminales cuando éstos acceden a una red. Dentro de la red debe haber un servidor DHCP que se encargue de mantener un conjunto de direcciones IP libres y cederlas a los clientes durante un determinado tiempo. El proceso para obtener una dirección IP implica el intercambio de cuatro mensajes. El mensaje "DHCP Discover" es el primero de estos mensajes y se envía por difusión desde el cliente. Cuando el mensaje es recibido, el servidor comprueba su banco de direcciones IP disponibles e informa de la posibilidad de cesión de la dirección IP al cliente mediante un mensaje "DHCP offer". El cliente acepta la oferta enviando de vuelta un mensaje "DHCP Request". Si la oferta todavía es válida, el servidor envía un mensaje "DHCP Reply" confirmando el proceso. En todo este proceso, los mensajes enviados por el cliente son de difusión, mientras que los enviados por el servidor son dirigidos al propio cliente (unicast). A partir de este momento, el cliente puede utilizar la dirección IP asignada mientras durante el tiempo de cesión que le haya asignado el servidor DHCP. Cuando un cliente pretende abandonar la red, envía un mensaje "DHCP Release" al servidor para que éste pueda borrar la información referente a la cesión y devuelva la dirección IP al banco de direcciones disponibles. De acuerdo con [56], el proceso de obtención de una dirección IP mediante DHCP tiene una duración

media de 2,5 segundos.

IEEE 802.11p (WAVE)

Como se adelantó anteriormente, IEEE 802.11p constituye una corrección al estándar IEEE 802.11 para adaptarlo a las particularidades de las comunicaciones V2V y V2I. Su capa física es parecida a la de IEEE 802.11a aunque utiliza un ancho de banda de 10MHz frente a los 20MHz de 802.11a. La banda utilizada, en torno a los 5,9GHz, está dividida en 7 canales, de los cuales 6 son de servicio y uno de control. A diferencia del comportamiento original de IEEE 802.11, las modificaciones introducidas permiten la participación de los dispositivos en múltiples canales mediante multiplexación por división en el tiempo. Tambien se introduce un mecanismo de prioridades con cuatro niveles.

Uno de los principales objetivos de IEEE 802.11p es el de reducir el tiempo de acceso. Para ello se hace uso de las tramas "beacon" pero sin necesidad de transmitirlas periódicamente, sino que se transmiten bajo demanda. En dichas tramas se incluye toda la información necesaria para que otro dispositivo pueda decidir si unirse o no a la red. En caso de querer establecer una conexión, no son necesarios los procesos de asociación y autenticación, reduciendo notablemente los tiempos de acceso.

Por otra parte, las particularidades de estas conexiones, especialmente las de tipo V2V, hacen que los modos de funcionamiento habituales (infraestructura y ad-hoc) no resulten adecuados. Por este motivo, IEEE 802.11p introduce el "modo WAVE", en el que los dipositivos escuchan al mismo canal y pueden utilizar un identificador de red (BSSID) común sin necesidad de pertenecer a una red a priori. De esta manera, dos vehículos podrían comunicarse directamente sin una sobrecarga adicional de mensajes de establecimiento de conexión. Sobre estas cuestiones se han realizado algunos estudios como [53] y [57], donde se obtienen tiempos de transmisión del orden de decenas de milisegundos, pero debido al hecho de que se trata de un estándar muy reciente, todos los resultados están basados en simulaciones.

Mobile IP

Si bien, DHCP es la opción más inmediata para solucionar el problema de obtener una dirección IP en una red desconocida a priori por un terminal, cabe pensar también en otras soluciones. Mobile IP es una tecnología que permite mantener una dirección IP incluso cuando se accede a través de otras

redes diferentes de la propia. Mobile IP está definido por el IETF en sus recomendaciones RFC 3775 [58] y RFC 3344 [59], para IPv6 e IPv4, respectivamente. Para entender Mobile IP hay que definir algunos conceptos:

- Red propia (Home network): Es la red habitual de un terminal.
- Dirección propia (Home address): Es la dirección utilizada dentro de la red propia.
- Red visitada (Foreign network): Es la red desde la que un terminal opera cuando está lejos de su red propia.
- Dirección como visitante (Care-of address): Es una dirección contenida en el ámbito de la red visitada que se le asigna a un terminal cuando está en una red distinta a la propia.
- Agente local (Home agent): Se trata de un router en la red propia del terminal que se encarga de encaminar a través de un túnel los paquetes que deben ser distribuidos a un terminal perteneciente a la red cuando éste se encuentra lejos de la red propia. Mantiene información sobre la dirección IP actual del terminal.
- Agente visitado (Foreign agent): Se trata de un router que almacena información acerca de los terminales que visitan su red. Este agente informa sobre las direcciones como visitantes que deben adoptar dichos terminales.

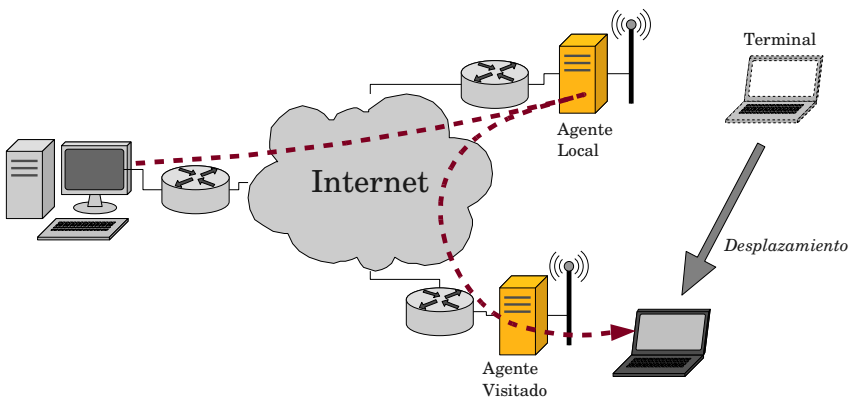

Figura 6: Elementos involucrados en Mobile IP

Las operaciones básicas en Mobile IP incluyen tres pasos [60]: Descubrimiento del agente, registro y establecimiento del túnel. El descubrimiento del agente es el método por el cual un terminal móvil comprueba los anuncios del agente para detectar su posición actual en la red

propia o en una red visitada. Estos mensajes están basados en una extensión del protocolo ICMP. Cuando el terminal se desplaza hacia una red visitada, puede obtener una dirección como visitante del agente visitado y crear una petición de registro a su agente local. El agente local envía una respuesta de registro al terminal móvil a través del agente visitado. Tras la operación de registro, el agente local guarda información relativa al terminal móvil en la red visitada. Todos los datagramas destinados al terminal serán retransmitidos por el agente local a través de un túnel hacia la dirección como visitante del terminal.

ZigBee

ZigBee es una tecnología de comunicación clasificada dentro de las redes inalámbricas de área personal (WPAN) que surge entre los años 2003 y 2004 a raíz del estándar IEEE 802.15.4 y la especificación 1.0 de ZigBee. La primera de las normas pertenece al conjunto de estándares del IEEE sobre redes WPAN y se encarga de definir las capas física y de enlace. La especificación ZigBee, por su parte, fue creada por la ZigBee Alliance [61], formada por amplio grupo de más de 70 empresas asociadas con el fin de crear y promocionar dicho estándar. Esta especificación se encarga de definir la capa de red y algunas capas de nivel superior.

Las características más importantes de esta tecnología son la flexibilidad de la red, el bajo coste y el bajo consumo de energía. ZigBee es diferente de los otros estándares inalámbricos, ha sido diseñado para soportar un diverso mercado de aplicaciones que requieren bajo coste y bajo consumo, con una conectividad más sofisticada que los anteriores sistemas inalámbricos. El estándar ZigBee enfoca a un segmento del mercado no atendido por los estándares existentes, con baja transmisión de datos, bajo ciclo de servicio de conectividad. La razón de promover un nuevo protocolo como un estándar es permitir la interoperabilidad entre dispositivos fabricados por compañías diferentes.

Las soluciones sobre el estándar ZigBee, en conexión de redes, se centran en mercados y aplicaciones específicas. El estándar ZigBee se ha hecho a medida para la monitorización y para aplicaciones de control mediante redes que no necesitan mucho ancho de banda y en las que el factor del consumo energético es importante. Por lo tanto, los mercados como la automatización de edificios y hogares, la atención sanitaria, control industrial, control de alumbrado y control comercial, son los principales campos de aplicación. Algunos ejemplos de aplicaciones son:

- Automatización de edificios y hogares
  - Seguridad
  - Alarmas
  - Control del aire acondicionado
  - Lectura de contadores de agua, gas, electricidad
  - Control de iluminación
  - Control de accesos
  - Control de riego
- Atención sanitaria
  - Monitorización de pacientes y equipos para la salud
- Control industrial
  - Control de procesos
  - Sensores
  - Instrumentación remota

Además, debido a su baja velocidad de transmisión de datos y su naturaleza de bajo consumo, también entra en el mercado del control remoto para la electrónica de consumo y en el de los denominados Dispositivos para la Interfaz Humana (HID), como teclados, ratones y joysticks.

La tecnología ZigBee define, como se mencionó anteriormente, desde la capa física hasta algunas partes por encima de la capa de red, según el modelo OSI. La figura 4 muestra la torre de protocolos usados en ZigBee:

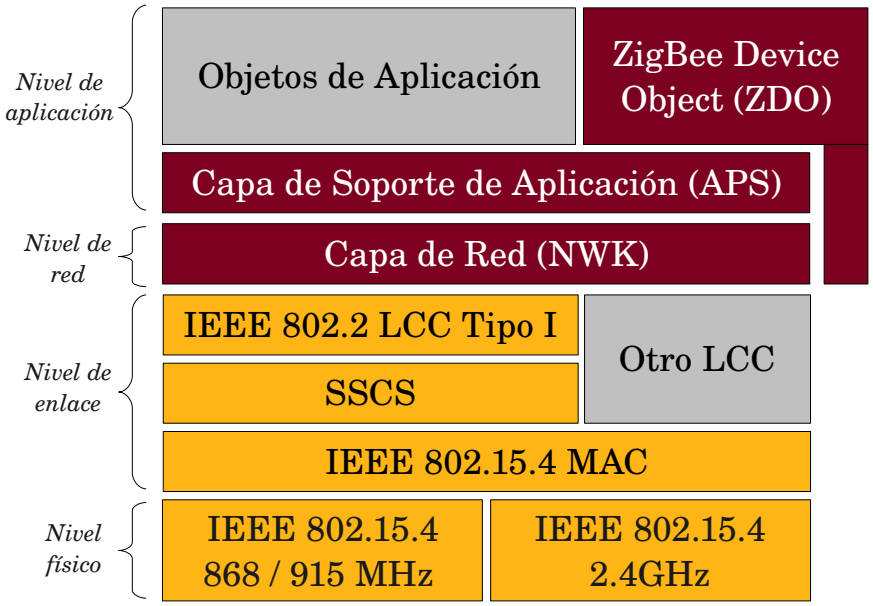

Figura 7: Torre de protocolos ZigBee

Capa física

IEEE 802.15.4 ofrece dos opciones de capa física (PHY) que combinan con el MAC para permitir un amplio rango de aplicaciones en redes. Ambas capas físicas se basan en métodos de Secuencia Directa de Espectro Extendido (DSSS) que resultan en bajos costos de implementación digital, compartiendo la misma estructura básica de paquetes de low-duty-cycle con operaciones de bajo consumo de energía. La principal diferencia entre ambas capas físicas radica en la banda de frecuencias. La capa física a 2.4 GHz, especifica la operación en la banda Industrial, Médica y Científica (ISM), que prácticamente está disponible en todo el mundo, mientras que la capa física a 868/915 MHz, especifica la operación en la banda de 865 MHz en Europa y 915 MHz en la banda ISM en Estados Unidos. La segunda distinción en las características de la capa física es la velocidad de transmisión. La capa física a 2.4 GHz permite una transmisión de 250 kb/s, mientras que la capa física a 868/915 MHz ofrece velocidades de transmisión de 20 kb/s y 40 kb/s, respectivamente. Esta velocidad superior de transmisión en la capa física a 2.4 GHz se atribuye principalmente a un orden mayor en la modulación, en la cual cada símbolo representa múltiples bits. Las diferentes velocidades de

transmisión se pueden explotar para lograr una variedad de objetivos o aplicaciones. Por ejemplo, la baja densidad de datos en la capa física a 868/915 MHz se puede ocupar para lograr mayor sensibilidad y mayores áreas de cobertura, con lo que se reduce el número de nodos requeridos para cubrir una área geográfica, mientras que la velocidad superior de transmisión en la capa física a 2.4 GHz se puede utilizar para conseguir salidas superiores y de poca latencia. Se espera que en cada capa física se encuentren aplicaciones adecuadas a cada una de ellas y a sus rangos de transmisión.

IEEE 802.15.4 define 27 canales de frecuencia entre las tres bandas. La capa física a 868/915 MHz soporta un solo canal entre los 868 y los 868.6 MHz, y diez canales entre los 902.0 y 928.0 MHz. Debido al soporte regional de esas dos bandas de frecuencias, es muy improbable que una sola red utilice los 11 canales. Sin embargo, las dos bandas se consideran lo suficientemente cercanas en frecuencia para que se pueda utilizar el mismo hardware en ambas, reduciendo los costes de producción. La capa física a 2.4 GHz soporta 16 canales entre los 2.4 y los 2.4835 GHz con un amplio espacio entre canales (5 MHz), con el objetivo de facilitar los requerimientos de filtrado en la transmisión y en la recepción.

Número de canal	Frecuencia central (MHz)
k=0	868,3
k=1 ... 10	906 + 2(k-1)
k=11 ... 26	2405 + 5(k-11)

Tabla 2: Frecuencias de los canales en ZigBee

Dado que en ciertos ambientes puede haber múltiples redes inalámbricas trabajando en las mismas bandas de frecuencias, así como una interferencia no intencionada de las diferentes aplicaciones, la capacidad de re-localización dentro del espectro es un factor importante. El estándar fue diseñado para implementar una selección dinámica de canales, a través de una selección específica de algoritmos la cual es responsabilidad de la capa de red. La capa MAC incluye funciones de búsqueda sobre una lista de canales permitidos en busca de una señal de guía, mientras que la capa física contiene varias funciones de bajo nivel, tales como la detección de los niveles de energía recibidos, indicadores de calidad en el enlace, así como de conmutación de canales, lo que permite asignación de canales y agilidad en la selección de frecuencias. Esas funciones son utilizadas por la red para establecer su canal inicial de operación y para cambiar los canales en respuesta a una pausa muy

prolongada.

La capa física en los 868/915 MHz utiliza una aproximación simple DSSS en la cual cada bit transmitido se representa por un chip-15 de máxima longitud de secuencia (secuencia m). Los datos binarios son codificados al multiplicar cada secuencia m por +1 o -1, y la secuencia de chip que resulta se modula dentro de la portadora utilizando BPSK (Binary Phase Shift Keying). Antes de la modulación se utiliza una codificación de datos diferencial para permitir una recepción diferencial coherente de baja complejidad.

Banda	**Parámetros de los datos**			**Parámetros del chip**	
	Velocidad de bits (kb/s)	Velocidad de símbolos (kbaud)	Modulación	Velocidad de chip (kchips/s)	Modulación
868 MHz	20	20	BPSK	300	BPSK
915 MHz	40	40	BPSK	600	BPSK
2,4 GHz	250	62,5	Ortogonal 16	2000	O-QPSK

Tabla 3: Parámetros de modulación en ZigBee

La capa física a 2.4 GHz emplea una técnica de modulación semi-ortogonal basada en métodos DSSS (con propiedades similares). Los datos binarios están agrupados en símbolos de 4 bits, y cada símbolo especifica una de las 16 secuencias de transmisión semi-ortogonales de código de pseudo-ruido (PN). Las secuencias PN son concatenadas y la secuencia resultante es modulada utilizando MSK (Minimum Shift Keying). En términos de eficiencia (energía requerida por bit), la señalización ortogonal mejora su funcionamiento en 2 dB frente a BPSK diferencial. Sin embargo, en términos de sensibilidad de recepción, la capa física 868/915 MHz tiene una ventaja de 6-8 dB debido a que tiene velocidades de transmisión más bajas. En ambos casos, las pérdidas de implementación debidas a factores como la sincronización, forma del pulso o simplificaciones en el detector generan desviaciones en las curvas óptimas de detección.

Las especificaciones actuales de sensibilidad del estándar IEEE 802.15.4 establecen un mínimo de –85 dBm para la capa física a 2,4 GHz y de -92 dBm para la capa de física a 868-915 MHz. Dichos valores incluyen suficiente margen para las tolerancias que se requieren debido a las

imperfecciones en la fabricación, de la misma manera que permite implementar aplicaciones de bajo coste. En cada caso, los mejores equipos deben ser del orden de 10 dB mejor que las especificaciones. Naturalmente, el rango deseado estará en función de la sensibilidad del receptor, así como de la potencia del transmisor. El estándar IEEE 802.15.4 especifica que cada dispositivo debe de ser capaz de transmitir al menos a 1 mW, pero dependiendo de las necesidades de la aplicación, la potencia de transmisión puede ser mayor o menor para aprovechar la energía. Los dispositivos típicos (1mW) pueden cubrir un rango de entre 10-20 m; sin embargo, con una buena sensibilidad y un incremento moderado en la potencia de transmisión, una red con topología tipo estrella puede proporcionar una cobertura total para toda una casa. Para aplicaciones que requieran mayor tiempo de latencia, la topología tipo "mesh" ofrece una alternativa atractiva con buenas coberturas, dado que cada dispositivo sólo necesita la energía suficiente como para comunicarse con su vecino más cercano.

Los dispositivos que operan en la banda de 2,4 GHz pueden recibir interferencias causadas por otros servicios que operan en dicha banda. Esta situación es aceptable en las aplicaciones que utilizan el estándar IEEE 802.15.4, las cuales requieren una baja calidad de servicio (QoS), no requieren comunicación asíncrona, y se espera que realice varios intentos para completar la transmisión de paquetes. Por el contrario, un requerimiento primario de las aplicaciones del IEEE 802.15.4 es una larga duración en las baterías, lo que consigue poca energía de transmisión y pocos ciclos de servicio. Debido a que el uso geográfico libre es un objetivo, la banda de 2,4 GHz es la banda preferida debido a su asignación internacional. Hay muchas bandas sin licencia en frecuencias superiores e inferiores. Las bandas de 2,4 GHz y 868/915 MHz fueron escogidas por el estándar IEEE 802.15.4 debido a sus características de propagación. Las frecuencias 868/915 MHz y 2,4 GHz tienen buena penetración tanto a través de paredes como de techos, pero tienen un rango limitado. La limitación de rango es realmente deseable para reducir las interferencias.

La coexistencia con otras tecnologías como Bluetooth o WiFi es posible. Las redes ZigBee y 802.15.4 están en las mejores condiciones atribuibles a su baja velocidad de transmisión y bajo ciclo de servicio. Los dispositivos ZigBee funcionan típicamente en ciclos de servicio de 0,1 al 1 %, permitiendo que el CSMA de la portadora tenga resultados robustos. Los dispositivos ZigBee o 802.15.4 observan que un canal esté libre antes de transmitir. El algoritmo CSMA es parte del software de 802.15.4, por lo que el usuario no tiene que crear ningún esquema adicional para evitar colisiones.

Capa de enlace

IEEE 802 divide a la Capa de enlace en dos subcapas, la subcapa de control de acceso al medio (MAC) y la subcapa de control de enlaces lógicos (LLC). El LLC es común a todos estándares 802, mientras que la subcapa MAC depende del hardware y varía respecto a la implementación física de esta capa. La estructura de estas capas, de acuerdo con el modelo OSI, se muestra en la figura 4.

Entre las funciones de la subcapa MAC IEEE 802.15.4 están la asociación/disociación, el reconocimientos de entrega de trama (ACK), los mecanismos de acceso al canal, la validación de trama, el control de garantía de ranuras de tiempo (Slot Time), el control de guías (Beacon) y el barrido de canal. La subcapa MAC proporciona dos tipos de servicios hacia las capas superiores, que se acceden a través de dos puntos de acceso a servicios (SAPs). A los servicios de datos MAC se accede por medio de la parte común de la subcapa (MCPS-SAP), mientras que el manejo de servicios MAC se hace por medio de la capa MAC de manejo de identidades (MLME-SAP). Esos dos servicios proporcionan una interfaz entre las subcapas de convergencia de servicios específicos (SSCS) u otro LLC, y las capas físicas. La subcapa MAC se caracteriza por una baja complejidad; el administrador de servicios MAC tiene 26 primitivas, frente a las cerca de 131 primitivas en 32 eventos que tiene Bluetooth, haciéndolo muy versátil para las aplicaciones hacia las que fue orientado. Por contra, se paga el coste de tener un instrumento con características inferiores a las del 802.15.1 (por ejemplo, IEEE 802.15.4 no soporta enlaces sincronizados de voz). La subcapa MAC 802.15.4 permite formar una red con hasta 264 dispositivos.

El formato general de las tramas MAC se diseñó para ser muy flexible y para que se ajustara a las necesidades de las diferentes aplicaciones con diversas topologías de red, al mismo tiempo que se mantenía un protocolo simple. A continuación se enumeran los tipos de tramas MAC:

- Data Frame. Usada para todas las transferencias de datos.
- Acknowledgement Frame. Usada para confirmar la recepción exitosa de una trama.
- MAC Command Frame. Usada para manejar todo el control de la entidad MAC.
- Beacon Frame. Usada por un coordinador para transmitir tramas guía o “beacons”.

Entre el conjunto de tramas, cabe resaltar las tramas "beacon", que añaden un nuevo nivel de funcionalidad a la red. Los dispositivos de los nodos pueden despertarse solamente cuando es transmitida una señal de guía o "beacon", escuchar su dirección y si no la escucha, volver al estado dormido con el consiguiente ahorro de energía. Las tramas "beacon" son importantes en las redes "mesh" y "cluster tree" para mantener todos los nodos sincronizados sin requerir que éstos tengan que escuchar durante largos periodos de tiempo y, por tanto, que tengan que consumir energía de la batería.

Algunas aplicaciones requieren anchos de banda dedicados para lograr grandes estados latentes, para un muy bajo consumo de energía. En estos casos, el estándar IEEE 802.15.4 puede operar en un modo opcional denominado "Superframes" o supertrama. En una supertrama, un coordinador de red transmite señales de guía o "beacons" a intervalos definidos. Estos intervalos pueden ser tan cortos como 15 ms o tan largos como 245 s. El tiempo entre cada uno de ellos se divide en 16 ranuras de tiempo (time slots), independientes a la duración de cada supertrama. Un equipo puede transmitir en cualquier instante durante una ranura de tiempo, pero debe terminar su transmisión antes de la siguiente supertrama de guía. De todas las ranuras de tiempo que forman la supertrama, el coordinador puede reservar un número de ellas para la comunicación con ciertos equipos, garantizando un ancho de banda determinado para dichas conexiones. El conjunto de ranuras reservadas se conoce como "Contention Free Period", mientras que el resto de ranuras forman el "Contention Access Period".

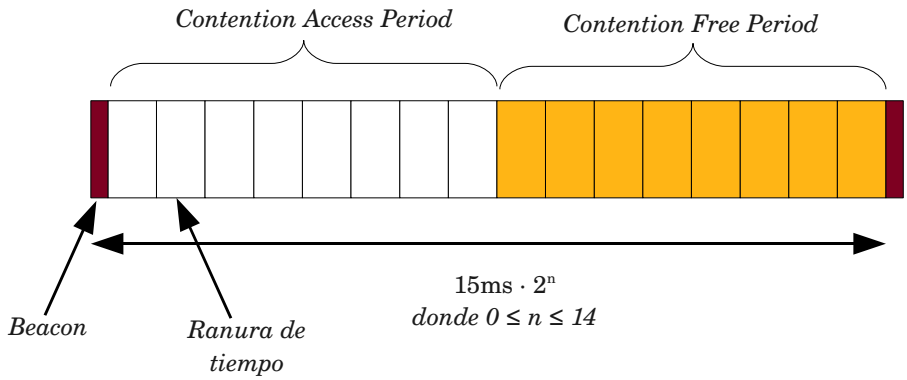

Figura 8: Estructura de una supertrama en Zigbee

En una red con supertramas, una portadora fragmentada en varios intervalos de tiempo permite múltiples accesos mediante el uso de CSMA-CA

(Carrier Sense Multiple Access – Collision Avoidance). Cualquier dispositivo que desee transmitir durante el "Contention Access Period" debe esperar a que comience la siguiente ranura de tiempo y después determinar si algún otro dispositivo se encuentra transmitiendo en la misma ranura. Si es así, el dispositivo se inhibe y espera un número aleatorio de ranuras antes de volver a intentarlo.

Capa de red

El estándar IEEE 802.15.4 soporta múltiples topologías para su conexión en red. La topología a escoger es una elección de diseño y va a estar determinada por la aplicación a la que se desee orientar. Las topologías disponibles en ZigBee son:

- Topología en estrella (star)
- Topología en árbol (cluster tree)
- Topología mallada (mesh)

Algunas aplicaciones, como periféricos e interfaces de PC, requieren de conexiones de baja potencia de tipo estrella, mientras que otros, como los perímetros de seguridad, requieren de una mayor área de cobertura, por lo que es necesario implementar una red mallada.

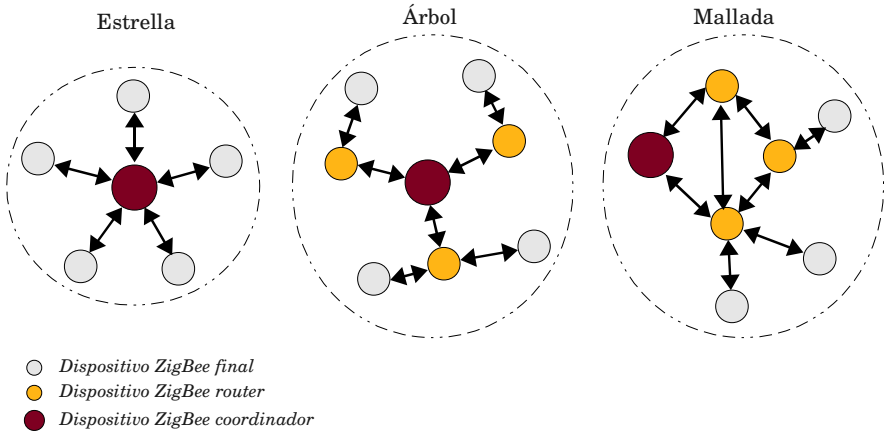

Figura 9: Topologías de red en Zigbee

Los dispositivos Zigbee pueden clasificarse atendiendo a dos criterios. Según su papel dentro de la red, existen tres categorías de dispositivos:

- Coordinador (ZC). El tipo de dispositivo más completo. Debe existir uno por red. Sus funciones son las de controlar la red y los caminos que deben seguir los dispositivos para conectarse entre ellos.
- Router (ZR). Interconecta dispositivos separados en la topología de la red, además de ofrecer un nivel de aplicación para la ejecución de código de usuario.
- Dispositivo Final (ZED). Posee la funcionalidad necesaria para comunicarse con su nodo padre (el coordinador o un router), pero no puede transmitir información destinada a otros dispositivos. De esta forma, este tipo de nodo puede estar dormido la mayor parte del tiempo, aumentando la vida media de sus baterías. Un ZED tiene requerimientos mínimos de memoria y es, por tanto, significativamente más barato.

Por otra parte, atendiendo a su funcionalidad, los dispositivos ZigBee pueden clasificarse en:

- Dispositivo de funcionalidad completa (FFD). También conocido como nodo activo. Es capaz de recibir mensajes en formato 802.15.4. Gracias a la memoria adicional y a la capacidad de procesamiento, puede funcionar como coordinador o router ZigBee o puede ser usado en dispositivos de red que actúen de interfaz con los usuarios.
- Dispositivo de funcionalidad reducida (RFD): También conocido como nodo pasivo. Tiene capacidad y funcionalidad limitadas (especificada en el estándar) con el objetivo de conseguir un bajo coste y una gran simplicidad. Básicamente, son los sensores/actuadores de la red.

La capa de red proporciona una serie de servicios genéricos a las capas superiores, entre los que se encuentran los siguientes:

- Escaneo de redes. Se utiliza para detectar canales activos dentro del rango de comunicaciones. Este rango es llamado a menudo POS (Personal Operating Space) en la conexión de redes de área personal.
- Creación y unión a redes PAN. Estas funciones permiten construir una red sobre un canal sin usar en el POS y que otros dispositivos puedan asociarse y participar en dichas redes.
- Descubrimiento de dispositivos. Permite identificar los dispositivos sobre los canales activos en la PAN.
- Descubrimiento de servicios. Permite determinar qué características o servicios son soportados sobre los dispositivos dentro de una red.
- “Binding”. Permite comunicarse, a nivel de aplicación, con otros dispositivos de la red.

Acceso a redes ZigBee

Para que un dispositivo pueda unirse a una red PAN existente, lo primero que debe hacer es usar las funciones de escaneo que proporciona la capa de red. Estas funciones ofrecen dos tipos de escaneo:

- Escaneo de detección de energía
- Escaneo activo

El primer tipo de escaneo es utilizado para determinar qué canales están siendo usados y su nivel de uso. Esto se hace en función de la energía recibida en cada uno de los canales. El escaneo activo, por su parte, envía tramas de tipo "beacon request" sobre un canal y se usa para determinar qué identificadores de redes PAN (PAN IDs) están en uso en ese canal dentro del rango de comunicaciones. Dado que el escaneo activo es capaz de detectar otras redes ZigBee, es utilizado por dispositivos que quieren unirse a una red. Si hay redes formadas al alcance del dispositivo, éste se dará cuenta tras la recepción de tramas de tipo "beacon". La duración del escaneo está definida por la especificación IEEE 802.15.4, y se representa como un número entero entre 0 y 14. El tiempo en milisegundos equivalente a cada uno de estos números enteros está relacionado con la duración de las supertramas [62], y se muestra en la tabla 4. Dependiendo de la banda utilizada y de las características particulares de la red, habría que repetir el procedimiento por cada canal que deba ser considerado a efectos de detección de la red o redes a las que se quiere conectar el dispositivo.

Valor	Duración (ms)	Valor	Duración (ms)
0	31	8	3948
1	46	9	7880
2	77	10	15744
3	138	11	31473
4	261	12	62930
5	507	13	125844
6	998	14	251674
7	1981		

Tabla 4: Duración del escaneo activo

Dentro de un determinado canal, cada red tiene un identificador de 16 bits distinto al que se denomina PAN ID. El estándar IEEE 802.15.4 especifica que el conjunto de identificadores disponibles está en el rango 0x0000 a 0xFFFE, aunque la especificación ZigBee lo restringe al rango 0x0000 a 0x3FFF. Normalmente este rango es suficiente para que no haya conflictos entre redes, pudiendo coexistir un gran número de ellas en el mismo espacio.

Dependiendo del tipo de aplicación, puede ser deseable que un dispositivo sólo se una a redes que tengan un perfil de aplicación particular. Una forma sencilla de resolver esto sería usar un PAN ID concreto para este tipo de redes y hacer que los dispositivos busquen redes con ese identificador. Esto puede ser válido en algunos casos, pero puede plantear problemas a medida que se expande el ámbito geográfico de la red ya que aumenta la probabilidad de que en alguno de los lugares dicho identificador puede estar siendo utilizado por otra red que tenga una finalidad distinta. Para solucionar este problema ZigBee define el concepto de PAN ID extendido. Estos nuevos identificadores tienen una longitud de 64 bits y su numeración no guarda ninguna relación con los PAN ID convencionales. Su finalidad es precisamente crear redes corporativas, por lo que es habitual formar estos identificadores con los OUI* asignados a empresas u organismos en los bits más significativos. El uso de identificadores extendidos no exime del uso de un PAN ID convencional para la red, pero añade la ventaja de que puede asignarse a cada ubicación un PAN ID que no esté siendo utilizado en esa zona.

En las tramas "beacon" siempre figura el PAN ID, pero cuando estas tramas se producen en respuesta a una trama "beacon request" previa, incorporan también el PAN ID extendido de la red, facilitando de esta manera la identificación de redes corporativas.

Bluetooth

Bluetooth† es un estándar de comunicaciones inalámbricas para cortas distancias que se enmarca dentro de las redes PAN. La historia de Bluetooth comienza en 1994, cuando Ericsson Mobile Communications inició un estudio

* Los OUI (Organizationally Unique Identifiers) son asignados por el IEEE para identificar empresas y organizaciones. Constan de 24 bits y se utilizan también para reservar bloques de direcciones MAC.

† El nombre Bluetooth procede de la adaptación al inglés del nombre del rey danés y noruego Harald Blåtand, conocido por unificar las tribus noruegas, suecas y danesas. Por ese carácter unificador, pero en el campo de las comunicaciones inalámbricas, se le dio ese nombre.

para investigar la posibilidad de una interfaz de radio de baja potencia y bajo costo entre teléfonos móviles y sus accesorios. El objetivo era eliminar los cables entre los teléfonos móviles y tarjetas de PCs, headsets, dispositivos desktop, etc. El estudio fue parte de otro gran proyecto de investigación que involucraba multicomunicadores conectados a la red celular por medio de los teléfonos móviles. El ultimo enlace en dicha conexión debería ser un radioenlace de corto rango. A medida que el proyecto progresaba, se vio claro que las aplicaciones que envuelven dicho enlace de corto rango serían ilimitadas. A comienzos de 1997, Ericsson se aproxima a otros fabricantes de dispositivos portátiles para incrementar el interés en esta tecnología. El motivo era simple, para que el sistema fuera exitoso y verdaderamente utilizable, una cantidad crítica de dispositivos portátiles deberían utilizar la misma tecnología de radioenlaces de corto alcance. En Febrero de 1998, cinco compañías, Ericsson, Nokia, IBM, Toshiba e Intel, forman un Grupo de Interés Especial (SIG) con el objetivo de crear una especificación global de la tecnología y gestionar su desarrollo. A fecha de escritura de esta Tesis, el SIG Bluetooth está formado por más 13000 empresas y la especificación Bluetooth [63] va ya por su versión 4.0. Además, Bluetooth fue ratificado como el estándar 802.15.1 por el IEEE desde la versión 1.1 de la especificación.

Desde su creación, Bluetooth ha experimentado un fuerte crecimiento en su implantación y, a día de hoy, está presente en prácticamente cualquier teléfono móvil u ordenador portátil. Algunas de las claves del éxito de Bluetooth están en su bajo coste, un consumo energético razonable y un ancho de banda suficiente para las aplicaciones en las que es utilizado. Esto ha motivado que Bluetooth sea empleado en una amplia variedad de aplicaciones como, por ejemplo:

- Dispositivos manos-libres
- Periféricos para PC sin cables (ratones, teclados, impresoras, etc.)
- Transferencia de ficheros entre terminales móviles
- Reemplazo de las tradicionales comunicaciones serie por cable en equipos de test, receptores GPS, lectores de códigos de barras, etc.
- Márketing de proximidad
- Equipamiento médico
- Pasarelas hacia redes TCP/IP

Bluetooth hace una distinción de los dispositivos en función del rol que adoptan cuando se establece una comunicación. De esta manera, cualquier equipo puede ser maestro o esclavo. El que adopta el papel de maestro se encarga de coordinar la comunicación en la red. Si bien el rol se determina durante el establecimiento de la conexión, éste puede ser cambiado

posteriormente mediante las órdenes adecuadas. A diferencia de otras tecnologías inalámbricas, el estándar Bluetooth está más enfocado al establecimiento de conexiones punto a punto entre un maestro y varios esclavos, aunque en ocasiones se pueden establecer comunicaciones punto a multipunto. Esta diferencia en la concepción de las redes Bluetooth hace que carezca de sentido la definición de un nivel de red, ya que no se permite, al menos directamente, la comunicación entre dos esclavos. La torre de protocolos de Bluetooth, según el modelo OSI, se muestra en la figura 7. Es importante reseñar que en la figura solamente aparecen algunos de los perfiles más relevantes, aunque existen muchos más.

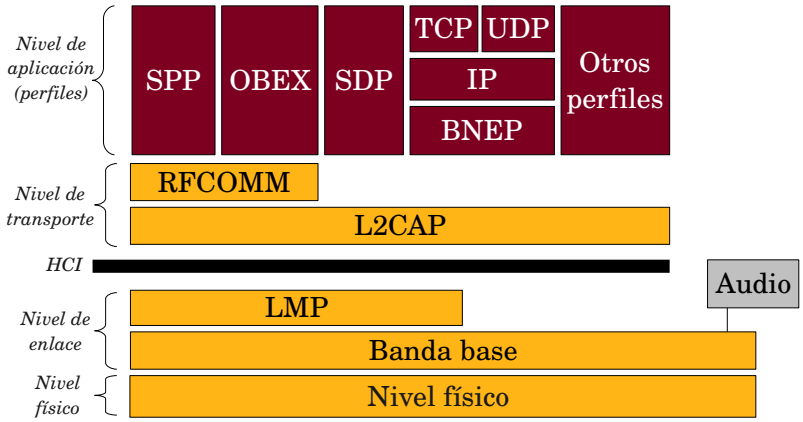

Figura 10: Torre de protocolos en Bluetooth

Nivel físico

Los dispositivos Bluetooth operan en la banda sin licencia ISM (Industrial Scientific Medical) a 2,4GHz. La banda se divide en 79 canales con una separación de 1MHz entre ellos de acuerdo con la siguiente expresión:

$$f = 2402 + k \text{ MHz, donde } k=0,...,78$$

Para evitar interferencias, la emisión en estos canales debe cumplir ciertos requisitos de atenuación mínima en las zonas del espectro adyacentes al canal. De esta manera, las emisiones en la zona comprendida entre 500KHz y 2MHz desde la frecuencia de la portadora, deben tener una atenuación de, al menos, 20dB respecto a ésta. A medida que nos alejamos de la portadora, las restricciones pasan a -20dBm para la zona comprendida entre 2MHz y 3MHz

desde la portadora y -40dBm para el resto de la banda.

Los dispositivos Bluetooth se dividen en tres clases según su potencia de emisión de acuerdo con los siguientes límites:

Clase	Potencia máxima	Potencia mínima
1	100mW (20dBm)	1mW (0dBm)
2	2,5mW (4dBm)	0,25mW (~6dBm)
3	1mW (0dBm)	--

Tabla 5: Clases de dispositivos Bluetooth y potencias de emisión

La modulación utilizada en Bluetooth es GFSK (Gaussian Frequency Shift Keying) con un producto de ancho de banda por tiempo de bit de 0,5. El índice de modulación está entre 0,28 y 0,35. Un '1' binario es representado por una desviación positiva de frecuencia, mientras que un '0' binario es representado por una desviación negativa de frecuencia. La desviación mínima de frecuencia no debe ser menor de 115KHz. Con estos parámetros se consigue una tasa de datos de 1Mbps, aunque desde la versión 2.0 de la especificación se introdujeron mejoras para conseguir tasas de datos mejoradas (EDR), alcanzándose los 3Mbps. Cuando se usa EDR, se cambia el tipo de modulación durante la transmisión de los paquetes. Una parte del paquete se transmite usando la tasa de datos básica de 1Mbps con modulación GFSK mientras que el resto del paquete se transmite usando una modulación PSK (Phase Shift Keying). Esta nueva modulación puede ser de tipo π/4-DQPSK (PSK diferencial cuaternaria rotada π/4), con la que se consigue una tasa de 2Mbps, o de tipo 8-DPSK (PSK diferencial de 8 niveles), con la que se alcanzan 3Mbps.

Desde el punto de vista del receptor, el aspecto más importante es el nivel de sensibilidad. Para una tasa de error o BER (Bit Error Rate) del 0.1% se define el umbral de sensibilidad de un receptor en –70dBm. La sensibilidad de cualquier dispositivo compatible con Bluetooth debe ser igual o inferior a dicho umbral.

Nivel de enlace

Como se mencionó anteriormente, Bluetooth proporciona conexiones punto a punto o punto a multipunto. Cuando dos o más dispositivos comparten un mismo canal físico, se forma una “piconet”. Un dispositivo

Bluetooth actúa como maestro de la piconet, mientras que el resto de dispositivos actúan como esclavos. Puede haber hasta siete esclavos activos en una piconet. De forma adicional, se pueden tener más esclavos conectados pero en un estado de inactividad que se conoce como estado "parked". Estos esclavos que están en estado "parked" no participan activamente en el canal pero se mantienen sincronizados con el maestro, pudiendo convertirse en nodos activos sin necesidad de usar el procedimiento de establecimiento de conexión ordinario. En cualquier caso, el acceso al canal por parte de esclavos activos o en estado "parked" está controlado por el maestro.

Cada piconet tiene un único maestro y uno o varios esclavos. Estos esclavos pueden pertenecer a más de una piconet según un esquema de multiplexación por división en tiempo. Es decir, que un esclavo divide su tiempo para atender a una piconet en cada momento. De igual modo se permite que el maestro de una piconet pueda pertenecer a otra piconet con la restricción de que solamente puede participar como esclavo en las otras piconets. En estas situaciones en las que algunos dispositivos pertenecen a varias piconets simultáneamente se forma lo que se conoce como "scatternet". Las piconets que componen una scatternet no pueden estar sincronizadas, ya que cada una debe transmitir en canales diferentes al resto para no colisionar. No debe entenderse una scatternet como un modo de crear redes jerarquizadas al estilo de las redes TCP/IP, ya que estas redes no están pensadas para establecer comunicaciones entre dos dispositivos arbitrarios de la scatternet (al menos, no directamente). En la figura 8.a se muestra el ejemplo más básico de piconet, en el que solamente intervienen un maestro y un esclavo. La figura 8.b muestra el esquema típico de las piconets con un maestro y varios esclavos, y la figura 8.c muestra una scatternet en la que el maestro de la piconet 1 es, a su vez, esclavo en la piconet 2 y un esclavo es compartido por las piconets 1 y 3.

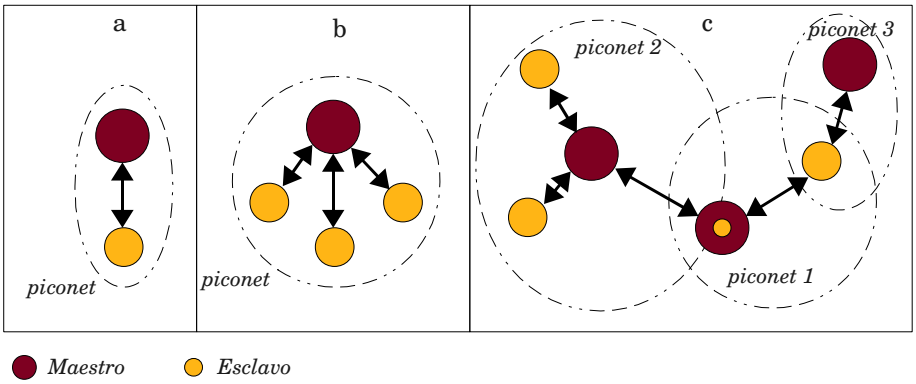

Figura 11: Ejemplos de piconets

Cada dispositivo Bluetooth dispone de un reloj nativo derivado del reloj del sistema, que corre libremente y puede estar inicializado a cualquier valor (no tiene nada que ver con la hora del día). Para la sincronización con otros dispositivos se utilizan diferencias respecto al reloj u "offsets". El reloj nativo está formado por un contador de 28 bits que se incrementa cada 312,5µs (la mitad de la duración de una ranura de tiempo). Este reloj se utiliza para determinar periodos críticos y disparar ciertos eventos en el dispositivo.

Los canales físicos en Bluetooth

La capa estructural más baja en Bluetooth la constituyen los canales físicos. Estos canales utilizan las modulaciones descritas para el nivel físico en combinación con secuencias pseudo-aleatorias de salto en frecuencia que se generan en función del reloj nativo. Estos saltos periódicos en la frecuencia de transmisión ayudan a reducir los efectos de la interferencia y el "fading", a la vez que satisfacen los requerimientos de las regulaciones locales en materia de uso de la banda de frecuencias. Un dispositivo Bluetooth sólo puede atender a un solo canal en cada instante, por lo que necesita repartir el tiempo entre varios canales cuando tiene que realizar operaciones de forma concurrente sobre varios canales. Se admite la posibilidad de que existan dispositivos avanzados capaces de atender simultáneamente a más de un canal, pero la especificación no entra a definir el comportamiento en estos dispositivos que, por otra parte, son muy infrecuentes. Bluetooth define cuatro canales físicos, cada uno de los cuales está optimizado para un propósito diferente:

- Basic Piconet Channel
- Adapted Piconet Channel
- Page Scan Channel
- Inquiry Scan Channel

Basic Piconet Channel

Es el canal utilizado por defecto tras establecer una conexión. El maestro se encarga de controlar el tráfico en el canal mediante un esquema de sondeo a los diferentes esclavos. Por definición, el dispositivo que inicia la conexión es el que adopta el papel de maestro, aunque esto puede cambiarse posteriormente. Este canal sigue una secuencia pseudo-aleatoria de frecuencias que abarca los 79 canales y cambia 1600 veces por segundo, es decir, cada 625μs. Esta secuencia está determinada por el reloj nativo y la dirección del maestro. La información de este reloj es comunicada a los esclavos cuando se establece una comunicación de modo que éstos puedan sincronizarse y conocer la secuencia de saltos en frecuencia utilizada. En el dominio del tiempo, el canal está dividido en ranuras de tiempo de 625μs cada una que son usadas de forma alternativa para las transmisiones maestro-esclavo y esclavo-maestro. El maestro selecciona, a través de la información de los paquetes, qué esclavo debe contestar. El comienzo de cada paquete está alineado con el comienzo de la ranura, y su duración puede ser de 1, 3 ó 5 ranuras completas. Como la duración de las ranuras y el tiempo de cambio de frecuencia son iguales, en cada ranura se utilizará una frecuencia diferente según la secuencia de salto, aunque durante la transmisión de paquetes que ocupan más de una ranura se mantiene la misma frecuencia, reanudándose posteriormente la secuencia de saltos por donde corresponda. La figura 9 representa un esquema típico de comunicación en una piconet con paquetes sencillos y multirranura. En cada caso, se indica la frecuencia, dentro de la secuencia, a la que se transmite cada paquete.

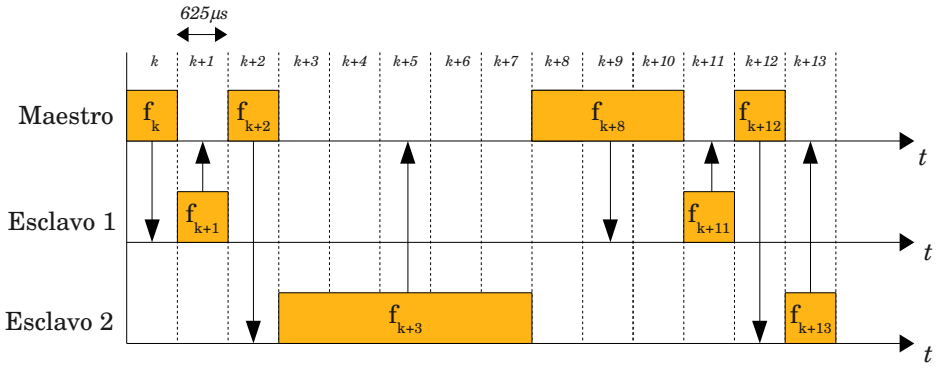

Figura 12: Esquema de comunicación en el canal básico de la piconet

Adapted Piconet Channel

Este canal tiene bastantes similitudes con el "Basic Piconet Channel", ya que también se utiliza para la transmisión de información tras el establecimiento de la conexión pero, a diferencia del anterior, este canal es usado por dispositivos que implementan el salto adaptativo de frecuencia (AFH). Este mecanismo puede usar menos de las 79 frecuencias disponibles y tiene la particularidad de que los esclavos utilizan, para responder, la misma frecuencia que usa el maestro para enviar el paquete precedente, en lugar de usar la que correspondería según la secuencia de salto. Esto es conocido como "mecanismo del mismo canal".

Page Scan Channel

Este canal se utiliza para realizar el proceso de llamada o "page", como paso necesario para establecer una conexión. Aunque los papeles de maestro y esclavo no están definidos antes del establecimiento de la conexión, se utiliza el término maestro para el dispositivo que realiza la operación de llamada, mientras que se asigna el término esclavo al dispositivo que escanea en busca de llamadas ("page scan"). Estos roles son finalmente adoptados cuando los dispositivos alcanzan el estado de conexión. Los saltos en frecuencia del canal de "page scan" siguen un patrón bastante más lento que el del canal básico de la piconet y pueden seguir varias secuencias dependiendo de la situación en el proceso de llamada. En concreto, para los cambios en frecuencia se tiene en cuenta a partir del bit 12 del reloj nativo, lo que provoca que el salto se produzca cada 1,28 segundos. En cuanto a la organización temporal del canal, al igual que en los canales definidos anteriormente, también está dividida en ranuras de tiempo de 625μs y también se alternan los paquetes en sentido

maestro-esclavo con los paquetes en sentido contrario. La excepción está en que algunos paquetes de corta longitud (ID) pueden alinearse con la mitad de la trama, pudiéndose transmitir dos de estos paquetes en una misma ranura.

Inquiry Scan Channel

Este canal se utiliza para realizar el proceso de descubrimiento de dispositivos o "inquiry", y tiene muchas similitudes con el "Page Scan Channel". Utiliza el mismo tratamiento de maestro/esclavo según se trate del dispositivo que busca o del que quiere ser descubierto, respectivamente. La secuencia de salto también es lenta, produciéndose un salto cada 1,28 segundos, y su ordenación se deriva del reloj nativo y de la dirección, pudiendo variar entre un conjunto de 32 frecuencias repartidas en dos juegos, también denominados "trenes", que no son fijos, sino que se intercambian frecuencias entre sí a medida que avanza el reloj. En cuanto a la organización temporal del canal se aplica lo mismo que al "Page Scan Channel".

Tipos de paquetes

Entre un maestro y sus esclavos se pueden establecer diferentes tipos de transportes lógicos. La especificación Bluetooth define cinco:

- Synchronous Connection-Oriented (SCO)
- Extended Synchronous Connection-Oriented (eSCO)
- Asynchronous Connection-Oriented (ACL)
- Active Slave Broadcast (ASB)
- Parked Slave Broadcast (PSB)

Los transportes lógicos síncronos (SCO, eSCO) crean enlaces punto a punto entre un maestro y un esclavo para la transmisión de datos con una regularidad acordada previamente. Por este motivo, estos transportes se utilizan a menudo para la transmisión de voz. Para llevar esto a cabo, el maestro reserva ranuras de tiempo a intervalos regulares dedicadas a la transmisión de paquetes SCO/eSCO. En el caso de los transportes eSCO, puede haber una ventana de retransmisión después de las ranuras reservadas.

Los transportes lógicos asíncronos (ACL) crean también enlaces punto a punto entre un maestro y un esclavo, y hacen uso de las ranuras que no están reservadas para tráfico síncrono. La organización de este tráfico se realiza bajo demanda y depende de la implementación del controlador de banda base,

que se encarga de multiplexar los flujos asíncronos de las capas superiores. Este tipo de transporte es el más habitual. Dentro de un transporte lógico ACL se pueden encontrar dos tipos de enlaces lógicos: uno denominado ACL-U (utilizado para transportar los datos de usuario, es decir, los datos procedentes de capas superiores) y otro denominado ACL-C (empleado por el gestor de enlaces para controlar el funcionamiento de dicho enlace asíncrono). Dentro del conjunto de enlaces lógicos existen prioridades. Así, un enlace ACL-C tendrá más prioridad que uno de tipo ACL-U o incluso que uno de tipo SCO-S/eSCO-S. Sin embargo, los enlaces SCO tendrán más prioridad que los ACL-U.

Los transportes de difusión (ASB, PSB) permiten a un maestro comunicarse con varios esclavos. En concreto, los transportes ASB van dirigidos a todos los esclavos activos en la piconet, mientras que los transportes PSB están destinados a la comunicación con los esclavos que están en estado "parked".

Existen varias formas de referenciar a un dispositivo Bluetooth, y cada una de ellas tiene sentido dentro de un contexto. En primer lugar, está la dirección Bluetooth (BD_ADDR), que está formada por 48 bits y es única a nivel mundial. Dicha dirección es obtenida de la Autoridad Registradora del IEEE (que asigna también las direciones MAC para redes LAN). Ésta es la dirección principal con la que aparece cualquier dispositivo cuando es descubierto y es utilizada a la hora de establecer conexiones con otros dispositivos. La dirección se descompone en una parte que identifica al fabricante del dispositivo y un número único para cada dispositivo asignado por el fabricante, denominado LAP (Lower Address Part). La primera parte ocupa los 24 bits más significativos, mientras que el LAP ocupa los 24 bits menos significativos. Dentro del conjunto de valores del LAP, el rango que va de 0x9E8B00 a 0x9E8B3F está reservado, y se utiliza en las operaciones de descubrimiento de dispositivos. Dentro de este rango, el valor 0x9E8B33 está asociado al código de acceso general para operaciones de inquiry, conocido como GIAC (General Inquiry Access Code), mientras que el resto de valores están asociados a códigos de acceso dedicados denominados DIAC (Dedicated Inquiry Access Code).

Una vez establecida la conexión, los esclavos son referenciados dentro de la piconet por el maestro a través de unas direcciones de transporte lógico (LT_ADDR) que constan únicamente de 3 bits*. Cuando un esclavo pasa a

* Esta longitud resulta suficiente para direccionar de una forma simplificada los 7 esclavos activos que, como mucho, puede haber en una piconet.

estado “parked”, se le asigna una dirección de tipo PM_ADDR que se compone de 8 bits. De esta manera, se pueden tener hasta 255 esclavos en estado “parked” en una piconet. Esta dirección sólo tiene validez mientras el esclavo permanece en estado “parked”. En cualquier caso, un dispositivo en este estado puede ser referenciado también por su dirección Bluetooth (BD_ADDR).

Por los transportes lógicos circulan paquetes de diversos tipos según el transporte utilizado, el canal físico y la función. Los paquetes siguen el esquema general representado en la figura 10.

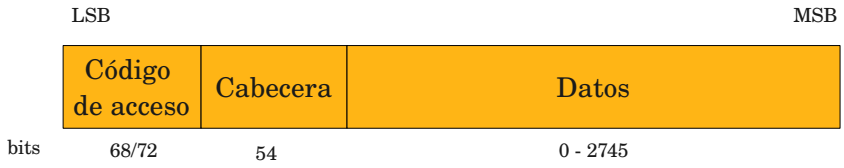

Figura 13: Estructura básica de un paquete en Bluetooth

El código de acceso puede tener una longitud de 72 ó 68 bits y la cabecera tiene una longitud de 54 bits. La carga de datos, por su parte, puede variar desde 0 hasta 2745 bits. Basándose en esta estructura, se pueden encontrar tres tipos de paquetes:

- Los compuestos únicamente por el código de acceso corto
- Los compuesto por el código de acceso y la cabecera
- Los que incluyen todos los campos

Cuando se usa EDR, la estructura cambia ligeramente. Se introduce un tiempo de guarda que oscila entre 4,75µs y 5,25µs, durante el cual el dispositivo cambia de tipo de modulación. El resto de la trama, que incluye la carga de datos, se transmite utilizando la modulación DPSK, permitiendo un mayor ancho de banda. La figura 11 ilustra esta variante del formato de los paquetes.

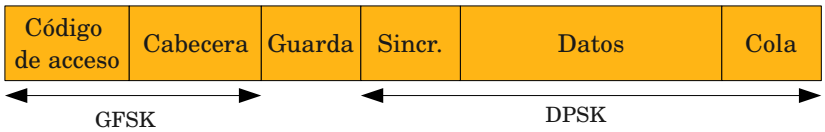

Figura 14: Estructura de un paquete con EDR

Cada paquete empieza con un código de acceso compuesto por un preámbulo, una palabra de sincronización y una cola. El preámbulo consta de 4 bits y sigue una secuencia de 1 y 0 alternándose. Si el bit menos significativo de la palabra de sincronización es 1, la secuencia del preámbulo será "1010" y en caso contrario será "0101". Este preámbulo ayuda al receptor a realizar una compensación de offset correcta y a identificar el comienzo de los paquetes. La palabra de sincronización es un código de 64 bits derivado de la parte LAP de la dirección Bluetooth. La cola de este código de acceso ocupa 4 bits y sólo se incluye si a continuación se transmite la cabecera. Al igual que el preámbulo, se trata de una secuencia de unos y ceros alternándose. Si el bit más significativo de la palabra de sincronización acaba en 0, la secuencia será "1010" y "0101" en caso contrario.

La cabecera contiene información de control del enlace y se compone de 6 campos, tal y como se detalla en la figura 12.

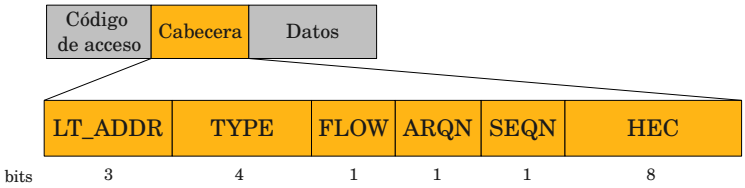

Figura 15: Estructura de la cabecera

- LT_ADDR. Referencia al esclavo destinatario del mensaje o al esclavo emisor del mensaje, según el sentido de la comunicación.
- TYPE. Indica el tipo concreto de paquete.
- FLOW. Se utiliza para control de flujo.
- ARQN. Se trata de un bit de asentimiento.
- SEQN. Se utiliza para mantener la secuencia de paquetes en orden.
- HEC. Es un código de redundancia utilizado para detectar errores en la cabecera.

Con esta estructura básica se pueden construir distintos tipos de paquetes. A continuación se detallan algunos de los paquetes más importantes:

Paquete ID

Estos paquetes constan únicamente del código de acceso de 68 bits, que puede tratarse del código de acceso del dispositivo (DAC) o del código de

acceso para inquiry (IAC). Este paquete identifica al dispositivo que lo emite.

Paquete FHS

Se trata de un paquete de control especial que contiene, entre otras cosas, la dirección Bluetooth del dispositivo y el reloj del remitente y la clase de dispositivo. Este paquete se utiliza como respuesta en los procedimientos de inquiry y llamada (page), acelerando este último gracias a la información que transporta. Los paquetes FHS se utilizan para la sincronización con el salto de frecuencias antes de que se establezca el canal de la piconet.

Paquetes para enlaces SCO

Para los transportes lógicos SCO se utilizan paquetes de tipo HV y DV. Los paquetes HV no incluyen un código de redundancia cíclica y no son retransmitidos en caso de error. Los paquetes DV sí incluyen un código de redundancia cíclica en la sección de datos y pueden ser retransmitidos si es necesario. Estos paquetes son utilizados habitualmente para transmisión de voz a 64kbps, aunque pueden ser usados para otros tipos de datos síncronos. Entre los paquetes HV encontramos tres variantes: HV1, HV2 y HV3. Las diferencias entre ellos están en el reparto entre la carga de datos y el código FEC (Forward Error Correction) para la corrección de errores. Las características concretas de estos paquetes se detalla en la tabla 6.

Paquetes para enlaces eSCO

Estos paquetes también son usados en transportes lógicos síncronos. Incluyen un código de redundancia cíclico y admiten retransmisiones si es necesario. Se definen tres tipos funadmentales de paquetes: EV3, EV4 y EV5, junto con cuatro tipos adicionales cuando se utiliza EDR, que son 2-EV3, 3-EV3, 2-EV5 y 3-EV5. Los paquetes EV4 y EV5 pueden cubrir hasta 3 ranuras de tiempo, mientras que los paquetes EV3 sólo pueden ocupar una. También existen diferencias en cuanto al reparto entre la carga de datos y el código FEC. Las variantes para EDR de estos paquetes son similares, aunque contienen más cantidad de información.

Paquetes para enlaces ACL

A diferencia de los paquetes anteriores, éstos se usan en enlaces asíncronos, y se clasifican en 7 tipos básicos más 6 tipos adicionales para EDR. Los tipos básicos son: DM1, DH1, DM3, DH3, DM5, DH5 y AUX1. En

estos casos, el número contenido en el nombre del tipo de paquete indica el número de ranuras de tiempo que ocupa. Así, un paquete DM1 ocupa sólo una ranura, mientras que un paquete DH5 ocupa hasta 5 ranuras. La diferencia fundamental entre paquetes DMx y DHx reside en que los primeros sí incorporan código FEC y los otros no. Al igual que en el caso de los paquetes para SCO, existen variantes específicas para EDR: 2-DH1, 3-DH1, 2-DH3, 3-DH3, 2-DH5 y 3-DH5. Las características concretas de estos paquetes se detalla en la tabla 6.

Tipo	**Carga de datos (bytes)**	**FEC**	**CRC**	**Tasa de datos máxima (kbps)**		
				Simétrica	*Asimétrica*	
					Directa	*Inversa*
ID	--	--	--	--	--	--
FHS	18	2/3	Sí	--	--	--
DM1	0 - 17	2/3	Sí	108,8	108,8	108,8
DH1	0 - 27	No	Sí	172,8	172,8	172,8
DM3	0 - 121	2/3	Sí	258,1	387,2	54,4
DH3	0 - 183	No	Sí	390,4	585,6	86,4
DM5	0 - 224	2/3	Sí	286,7	477,8	36,3
DH5	0 - 339	No	Sí	433,9	723,2	57,6
AUX1	0 - 29	No	No	185,6	185,6	185,6
2-DH1	0 - 54	No	Sí	345,6	345,6	345,6
2-DH3	0 - 367	No	Sí	782,9	1174,4	172,8
2-DH5	0 - 679	No	Sí	869,1	1448,5	115,2
3-DH1	0 - 83	No	Sí	531,2	531,2	531,2
3-DH3	0 - 552	No	Sí	1177,6	1766,4	235,6
3-DH5	0 - 1021	No	Sí	1306,9	2178,1	177,1
HV1	10	1/3	No	64	--	--
HV2	20	2/3	No	64	--	--
HV3	30	No	No	64	--	--

Tipo	Carga de datos (bytes)	FEC	CRC	Tasa de datos máxima (kbps)		
				Simétrica	*Asimétrica*	
					Directa	*Inversa*
DV	10 + (0 - 9)	2/3	Sí	64 + 57,6	--	--
EV3	1 - 30	No	Sí	96	--	--
EV4	1 - 120	2/3	Sí	192	--	--
EV5	1 - 180	No	Sí	288	--	--
2-EV3	1 - 60	No	Sí	192	--	--
2-EV5	1 - 360	No	Sí	576	--	--
3-EV3	1 - 90	No	Sí	288	--	--
3-EV5	1 - 540	No	Sí	864	--	--

Tabla 6: Tipos de paquetes en Bluetooth

Estados del controlador de enlace

Un controlador de enlace Bluetooth puede estar en tres estados principales:

- Stand-by
- Connection
- Parked

Adicionalmente se definen siete subestados de carácter transitorio:

- Page
- Page Scan
- Master Response
- Slave Response
- Inquiry
- Inquiry Scan
- Inquiry Response

Un controlador de enlace se encuentra por defecto en estado “stand-by”, es decir, en espera. En esta situación, el consumo es muy bajo ya que sólo corre el reloj nativo. El controlador puede abandonar este estado para realizar

procesos de descubrimiento o llamada o para atender a estos procesos cuando son iniciados por otros dispositivos. La relación entre los diferentes estados y subestados se muestra en la figura 13. El estado "connection" se alcanza cuando se ha establecido una conexión, pero para llegar a él es necesario utilizar el proceso de "page" o llamada y pasar por los subestados asociados. Para este propósito puede ser útil hacer uso previamente de los subestados de inquiry, que permiten descubrir dispositivos cercanos. Una vez establecida la conexión, ésta se puede mantener inactiva por un tiempo indefinido. Esto es lo que se conoce como estado "parked". Obsérvese que dentro del propio estado "connection" también se permite desactivar temporalmente y de forma total o parcial una conexión existente mediante los modos "sniff" y "hold", de los que se hablará más adelante.

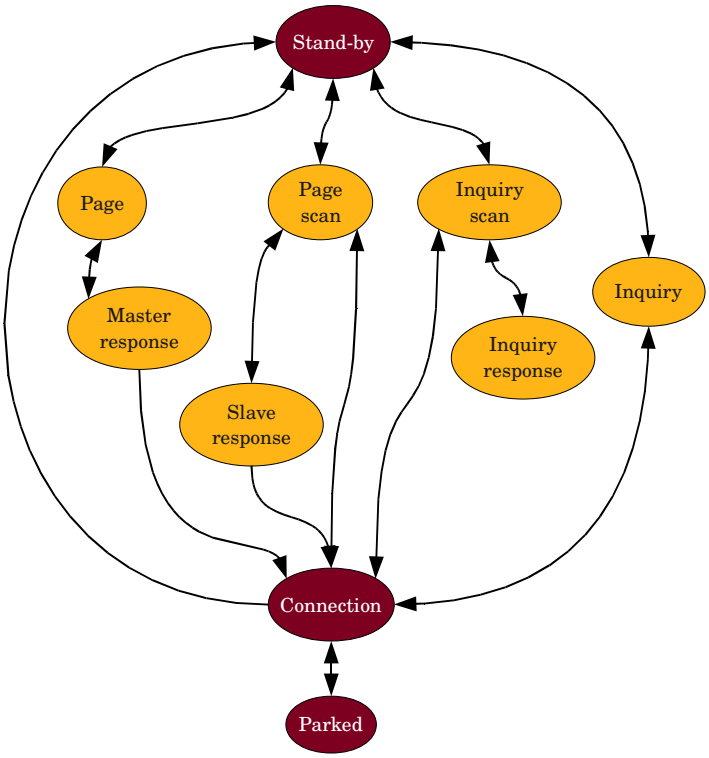

Figura 16: Relación entre los estados del controlador de enlace

Como se señaló anteriormente, cuando un esclavo pasa a estado "parked", éste cede su dirección LT_ADDR para que pueda ser utilizada por una nueva conexión activa y recibe dos direcciones: PM_ADDR (Parked Member Address) y AR_ADDR (Access Request Address), ambas de 8 bits. La

dirección PM_ADDR se utiliza para distinguir a un esclavo aparcado de otro cuando un maestro intenta sacar del estado "parked" a alguno de ellos. La dirección AR_ADDR la utiliza el propio esclavo aparcado cuando solicita salir del estado "parked". Como se puede apreciar, aunque el maestro es el que tiene capacidad para gestionar una piconet y determinar qué esclavos pasan al estado "parked", la iniciativa para volver a restaurar la conexión pueden tenerla tanto el maestro como el esclavo afectado.

Para dar soporte a los esclavos en estado "parked", el maestro transmite de forma periódica lo que se denominan "beacons" en ciertas ranuras de tiempo cuando hay uno o más esclavos en este estado. Además de estas tramas "beacon", se definen ventanas de acceso a través de las cuales los esclavos en estado "parked" pueden enviar peticiones para salir de dicho estado. Los esclavos aparcados despiertan a intervalos regulares y escuchan el canal para resincronizarse y comprobar si hay mensajes de difusión. Todos los mensajes enviados a los esclavos aparcados son transportados por paquetes de difusión. Dado que los periodos de transmisión de tramas "beacon" son configurables, esta información se le pasa a los esclavos en el momento de pasar al estado "parked". De este modo, los esclavos saben en qué momento deben despertarse.

Modo Sniff

El modo "Sniff" es aplicable a enlaces en estado "connection". En este modo, la actividad de un esclavo en una piconet puede reducirse, ya que éste no escuchará todas las ranuras. Según la reducción aplicada, el maestro sólo transmitirá paquetes al esclavo en determinadas ranuras que el esclavo escuchará. Dichas ranuras están separadas regularmente a intervalos definidos por el parámetro "Sniff interval" (T_{sniff}). Al comienzo de este intervalo, el esclavo escucha la transmisión del maestro en la ranura y usa las siguientes reglas para determinar si debe continuar escuchando en las siguientes ranuras:

- Si han transcurrido menos ranuras en sentido maestro-esclavo del expresado en el parámetro $N_{sniff\_attempt}$ desde el comienzo del intervalo.
- Si el esclavo ha recibido algún paquete destinado a él en las últimas $N_{sniff\_timeout}$ ranuras en sentido maestro-esclavo.
- Si el esclavo ha transmitido algún paquete ACL en las últimas $N_{sniff\_timeout}$ ranuras en sentido esclavo-maestro.
- Un esclavo puede infringir las reglas anteriores y dejar de escuchar antes de lo expresado si tiene actividad en otra piconet.

La reducción del tráfico del enlace, aunque está determinada en parte por los parámetros T_{sniff}, $N_{sniff\_attempt}$ y $N_{sniff\_timeout}$, depende también de las necesidades de comunicación entre el maestro y el esclavo. De esta forma, la restricción aplicada puede relajarse en los momentos de mayor demanda de tráfico. El modo "sniff" sólo es aplicable a transportes ACL y la entrada y salida de dicho modo está controlada por funciones API que pueden ser invocadas desde las capas superiores.

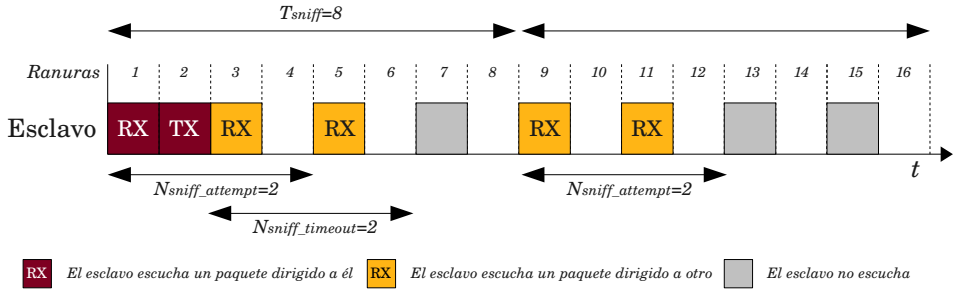

Figura 17: Ejemplo de funcionamiento del modo sniff

Modo hold

Durante el estado "connection", un transporte lógico hacia un esclavo puede ser puesto en modo "hold". Cuando un esclavo está en este modo, interrumpe totalmente el tráfico de paquetes ACL en el canal. Sin embargo, los paquetes SCO/eSCO continúan transmitiéndose. Con el modo "hold" se puede liberar capacidad para hacer otras cosas como, por ejemplo, buscar dispositivos, cambiar a otra piconet o incluso entrar en un modo de bajo consumo energético. Durante el modo "hold", el esclavo mantiene su dirección de transporte lógico (LT_ADDR) aunque no participe en la piconet. Antes de entrar en modo "hold", el maestro y el esclavo acuerdan el tiempo que este último permanecerá en modo "hold" mediante el parámetro "holdTO". Cuando este temporizador expira, el esclavo se despierta y se sincroniza con el tráfico de la piconet para esperar posteriormente paquetes procedentes del maestro.

Hay que tener en cuenta que, independientemente de que un esclavo entre en modo "hold", existe un temporizador de supervisión del enlace (Link Supervision Timeout) que sigue corriendo. Si un dispositivo no recibe ningún paquete válido dirigido a él durante el periodo del temporizador, el enlace se da por roto. Este mecanismo se utiliza para detectar la pérdida de conexión y, en situaciones normales, no interrumpe una conexión aunque las capas

superiores no estén enviado datos gracias a que el maestro periódicamente envía paquetes de sondeo a sus esclavos, lo que reinicia dicho temporizador. Sin embargo, actúa correctamente en los casos en los que se pierde la señal. Si bien el periodo por defecto de este temporizador se sitúa en el orden de los 20 segundos, éste puede configurarse a otros valores inferiores. Si la duración del modo "hold" es superior al periodo de dicho temporizador, cuando el dispositivo despierte, el enlace se habrá dado por roto aunque siga en la zona de cobertura de la piconet y pueda recibir la señal correctamente.

Descubrimiento de dispositivos. El proceso de inquiry

Un dispositivo Bluetooth entra en el subestado de inquiry cuando está buscando a otros dispositivos para formar una piconet, asumiendo, de forma provisional, el papel de maestro. Mientras está en este subestado, el dispositivo transmite dos paquetes ID en cada ranura de tiempo asignada a la transmisión del maestro. Uno de estos paquetes se transmite al comienzo de la ranura y el otro a la mitad, utilizando distintas frecuencias según la secuencia de salto prevista. Estos paquetes pueden contener el código de acceso general (GIAC) o uno de los códigos dedicados (DIAC), según se pretenda descubrir a todos los dispositivos Bluetooth al alcance o restringir la búsqueda sólo a aquellos dispositivos que utilicen el mismo DIAC, respectivamente.

Por su parte, los dispositivos que quieren ser descubiertos entran periódicamente en el subestado de inquiry scan y adoptan el papel de esclavos para este proceso. Mientras están en este subestado, los dispositivos escuchan en el "Inquiy Scan Channel" en busca de paquetes ID. Cuando detectan uno de estos paquetes contestan con un paquete FHS, que contiene información básica sobre el dispositivo.

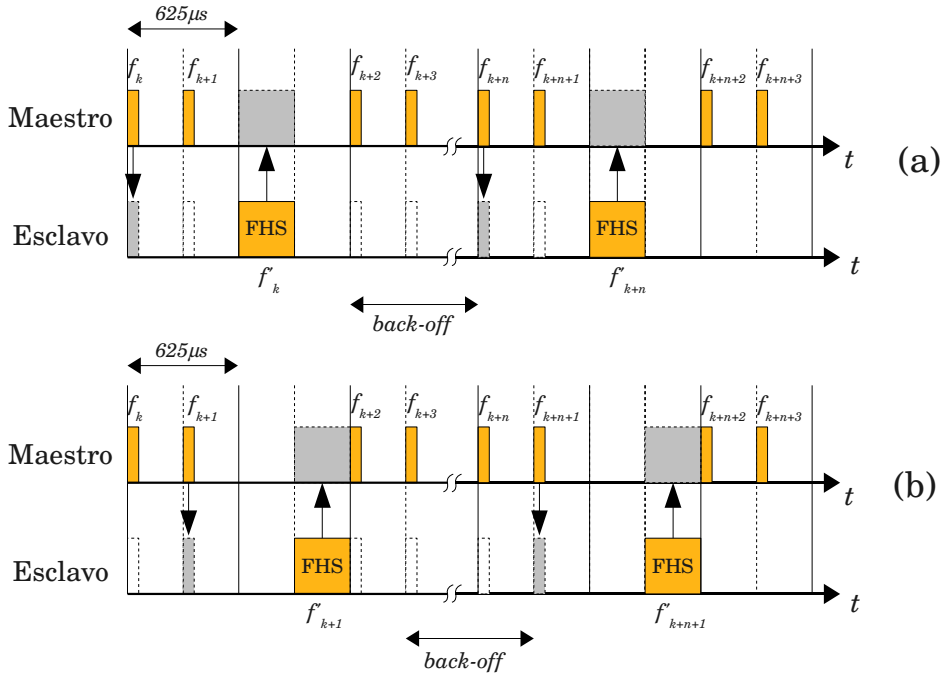

Figura 18: Intercambio de paquetes en el proceso de inquiry

La organización temporal del proceso, mostrada en la figura 15, prevé la alternancia de ranuras de tiempo en sentido maestro-esclavo con ranuras en sentido esclavo-maestro. A diferencia del canal básico de la piconet, en el canal físico utilizado para este proceso, el conjunto de frecuencias utilizables se reduce a 32, repartidas en dos juegos o "trenes" de 16 frecuencias cada uno. Las frecuencias que componen dichos trenes no son siempre las mismas, ya que se van intercambiando de forma periódica. Cada 1,28 segundos una frecuencia del tren A pasa a pertenecer al tren B y viceversa. La figura 16 muestra este intercambio a lo largo del tiempo hasta que ambos trenes intercambian todas sus frecuencias respecto a la situación de partida, al cabo de 20,48 segundos.

Tiempo	Primera Frecuencia																Segunda Frecuencia															
0 s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
1,28s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
2,56s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
3,84s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
5,12s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
6,40s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
7,68s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
8,96s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
10,24s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
11,52s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
12,80s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
14,08s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
15,36s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
16,64s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
17,92s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
19,20s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5
20,48s	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	0	1	2	3	4	5

Tren A Tren B

Figura 19: Evolución de los trenes de frecuencias para inquiry

En cada ranura de transmisión, el dispositivo maestro transmite dos paquetes ID cada uno a una frecuencia distinta dentro del mismo tren. De este modo, cuando transcurren 16 ranuras se han recorrido todas las frecuencias que componen el tren, esto es, al cabo de 10ms. Este proceso se repite hasta alcanzar las 256 iteraciones o, lo que es lo mismo, los 2,56 segundos, momento en el que se cambia de tren de frecuencias durante otros 2,56 segundos. Este comportamiento se repite hasta que el dispositivo maestro salga del subestado de inquiry. La duración de este proceso es configurable, pero siempre debe hacerse por múltiplos de 1,28 segundos.

Los dispositivos que quieren ser descubiertos entran periódicamente en el subestado de inquiry scan. Dicha periodicidad es configurable aunque, por defecto, ésta se sitúa en los 1,28 segundos. Cada vez que un dispositivo entra en este subestado se dice que abre la ventana de escaneo, permaneciendo en él durante un tiempo configurable que, por defecto, es de 11,25ms. Durante todo ese tiempo, el dispositivo escucha en una de las frecuencias de uno de los trenes de frecuencia. Los dispositivos esclavos también cambian su frecuencia de recepción siguiendo una secuencia similar a la descrita anteriormente para el dispositivo que hace la búsqueda, con la salvedad de que lo hacen mucho más despacio. Concretamente, estos dispositivos cambian de frecuencia cada 1,28 segundos. De este modo, durante el tiempo que la ventana permanece abierta, el dispositivo maestro ha emitido paquetes a todas las frecuencias que componen uno de los trenes. Si el dispositivo esclavo estaba escuchando en una de las frecuencias que componen el tren utilizado por el maestro, algo bastante probable (~50%), recibirá un paquete en algún instante de la ventana de escaneo.

A partir de este punto, el comportamiento puede ser diferente según la versión de la especificación Bluetooth que se considere. En las versiones antiguas, cuando un esclavo recibía un paquete ID, abandonaba el subestado de inquiry scan por un periodo de tiempo aleatorio conocido como "back-off" y volvía al subestado de inquiry scan una vez concluído ese tiempo. En esta nueva ocasión, si el dispositivo volvía a recibir un paquete ID contestaba en la ranura siguiente con un paquete FHS alineado con el instante en el que recibió el paquete ID (Ver figuras 15a y 15b). El tiempo de "back-off" representa un número aleatorio de ranuras que el dispositivo debe esperar y está uniformemente distribuído entre 0 y 1023. Por lo tanto, el tiempo de "back-off" puede variar entre 0 y 639,375ms. Gracias al carácter aleatorio del retraso introducido por el "back-off", se minimizan las colisiones entre múltiples dispositivos que intentan contestar al mismo paquete ID, ya que cada dispositivo generará un tiempo de "back-off" diferente.

En las especificaciones más recientes (desde la versión 1.2 en adelante), el comportamiento cambia ligeramente ya que los dispositivos que reciben un paquete ID responden con un paquete FHS en la siguiente ranura, independientemente de la aplicación del tiempo de "back-off". Cuando este tiempo termina, el dispositivo vuelve al subestado de inquiry scan y contestará de nuevo con un paquete FHS si recibe un paquete ID. Este nuevo comportamiento ayuda a mejorar los tiempos de detección, a la vez que mantiene la robustez aportada por el tiempo de "back-off".

Cuando el maestro recibe un paquete FHS, extrae cierta información contenida en el mismo y la almacena en una lista junto con la información de otros dispositivos que hayan contestado durante el proceso de inquiry. De la información contenida en un paquete FHS se pueden destacar los siguientes campos:

- LAP+UAP+NAP. La unión de estos tres campos forma la dirección Bluetooth del dispositivo. Este dato es esencial para identificar unívocamente a dicho dispositivo.
- Reloj nativo. Se transmiten los bits más significativos del registro que actúa como reloj nativo del dispositivo tal y como estaban al comienzo de la transmisión del paquete. Esta información le es bastante útil al maestro para la posterior sincronización durante el procedimiento de llamada.
- Clase del dispositivo. Se trata de un campo de 24 bits que codifica el tipo de dispositivo y las funcionalidades asociadas. Sus valores están determinados por la especificación Bluetooth y siguen el esquema

mostrado a continuación:

Clases de servicio	Major	Minor	00
11 bits	5 bits	6 bits	2 bits

Tabla 7: Campos dentro de la clase de dispositivo

El campo de clases de servicio está compuesto por una serie de bits que indican, a grandes rasgos, las funcionalidades básicas que admite el equipo que dispone de conexión Bluetooth. Estas funcionalidades tienen un carácter amplio y no tienen por qué estar relacionadas con la tecnología Bluetooth propiamente. En la siguiente tabla se enumeran estas funcionalidades, que no son mutuamente excluyentes:

bit	Funcionalidad
23	Información
22	Telefonía
21	Audio
20	Transferencia de objetos
19	Captura
18	Rendering
17	Funciones de red
16	Posicionamiento
13	Modo de descubrimiento limitado

Tabla 8: Funcionalidades admitidas en las clases de servicio

El campo “major” concreta más, mediante una serie de códigos, el tipo de equipo. Los códigos admitidos son los siguientes:

Código (binario)	Tipo
00000	Misceláneo
00001	Ordenador (de sobremesa, portátil, PDA, etc.)

Código (binario)	Tipo
00010	Teléfono (móvil, inalámbrico, módem, etc.)
00011	Punto de acceso a una red LAN
00100	Audio/Vídeo (auricular, altavoz, display, etc.)
00101	Periférico (ratón, teclado, joystick, etc.)
00110	Imagen (impresora, escáner, cámara, etc.)
00111	Usable
01000	Juguete
11111	Sin categoría
resto	Reservado

Tabla 9: Tipos de servicio

El significado del campo "minor" varía en función del major seleccionado. De esta manera, se obtiene un amplio número de combinaciones cuyos valores más interesantes se detallan en la tabla 10:

Major	Minor	Descripción
Ordenador	000001	Estación de trabajo de sobremesa
	000010	Servidor
	000011	Portátil
	000100	PC/PDA de mano
	000101	PC/PDA de pequeño tamaño
Teléfono	000001	Teléfono móvil
	000010	Teléfono inalámbrico
	000011	Teléfono inteligente
	000100	Módem cableado o pasarela de voz
	000101	Acceso común a RDSI
Periférico	01xxxx	Teclado
	10xxxx	Dispositivo para apuntar (ratón)

Major	Minor	Descripción
	11xxxx	Teclado + Dispositivo para apuntar
	xx0001	Joystick
	xx0010	Gamepad
	xx0011	Control remoto
	xx0100	Dispositivo táctil
	xx0101	Tableta digitalizadora
Audio / Video	000001	Headset
	000010	Dispositivo *Manos-libres*
	000100	Micrófono
	000101	Altavoz
	000110	Auriculares
	001000	Car audio
	001100	Videocámara
	001110	Video monitor
	010000	Equipo de videoconferencia
Imagen	xxx1xx	Display
	xx1xxx	Cámara
	x1xxxx	Escáner
	1xxxxx	Impresora

Tabla 10: Campos dentro de la clase de dispositivo

Existe una variante de este proceso que añade información extra a las respuestas de los esclavos. Si está habilitada la opción de respuesta extendida al inquiry, además del paquete FHS que contiene la información esencial, el esclavo puede transmitir un paquete con la información adicional 1250μs después del comienzo del paquete FHS, esto es, dos ranuras después. Este paquete adicional puede ser de alguno de estos tipos: DM1, DM3, DM5, DH1, DH3 ó DH5. El paquete de respuesta extendida puede contener la siguiente información:

- Nombre del dispositivo
- identificadores de clase de servicio (UUID)
- Potencia de transmisión
- Valores específicos del fabricante

El controlador de banda base puede facilitar datos adicionales a los contenidos en los paquetes descritos. Concretamente, se puede facilitar el valor de RSSI medido durante la recepción de dichos paquetes, que estaría relacionado con la potencia en recepción. En posteriores apartados se describirá dicha relación.

Modo entrelazado

El modo entrelazado para el proceso de inquiry se introdujo a partir de la versión 1.2 de la especificación y consiste en concatenar dos ventanas de escaneo seguidas empleando frecuencias de trenes distintos. De este modo, si, por ejemplo, el maestro está transmitiendo paquetes en las frecuencias del tren B, éstos no serán escuchados en la ventana que utilice una frecuencia del tren A, pero sí en algún punto de la ventana que escucha en una frecuencia del tren B. Como resultado, cada vez que se abre el par de ventanas de escaneo, la probabilidad de recibir un paquete ID en alguna de ellas es casi del 100%. En modo entrelazado, la permanencia en el subestado de inquiry scan es el doble (22,5ms cada 1,28s). Esto significa que el dispositivo no está disponible para otros usos durante el 1,75% del tiempo, frente al 0,87% cuando no se usa el modo entrelazado. Esta pequeña desventaja se compensa con creces con la notable reducción sobre el tiempo de detección.

El proceso de llamada

El proceso de llamada o “page” tiene muchas similitudes con el proceso de inquiry aunque difiere, fundamentalmente, en el tipo de paquetes y su secuencia de intercambio. De forma análoga a como se definieron los roles provisionales para el proceso de inquiry, en este proceso el que tiene la iniciativa de establecer la conexión es el que adopta el papel de maestro, siendo el esclavo el destinatario de dicha conexión. El subestado de “page” es utilizado por el maestro para activar y conectar a un esclavo en el subestado page scan. El maestro intenta coincidir con la actividad de exploración del esclavo, transmitiendo repetidamente los mensajes de paging consistentes en el código de acceso del dispositivo esclavo en diferentes canales. Puesto que el reloj del maestro y del esclavo no están sincronizados, el maestro no conoce exactamente cuando el esclavo despertará ni en qué salto de frecuencia. Por

tanto, el maestro transmite un tren de mensajes idénticos de page scan en diferentes frecuencias, escuchando entre los intervalos de transmisión hasta que recibe una respuesta del esclavo. El maestro comunica la dirección Bluetooth del esclavo al controlador. Ésta será usada por el maestro para determinar la secuencia de saltos del subestado page. Para la fase de la secuencia, el maestro empleará una estimación del reloj del esclavo. Aunque el maestro y el esclavo utilicen la misma secuencia de saltos, la fase puede ser diferente, de forma que el esclavo y él no se sincronicen. Por este motivo, el maestro transmite varios mensajes page durante un intervalo corto de tiempo en un conjunto de frecuencias activas. Este desconocimiento de la fase del reloj del esclavo no siempre ocurre, ya que en los casos en los que se haya realizado previamente una búsqueda a través del proceso de inquiry se habrá obtenido el reloj del esclavo. En este caso, el proceso de page se ve acelerado ya que el maestro puede estimar con mayor precisión la fase del reloj del esclavo.

Durante cada intervalo de transmisión, el maestro transmitirá secuencialmente en dos frecuencias diferentes. En el siguiente intervalo de recepción, el receptor escuchará secuencialmente en dos frecuencias diferentes.

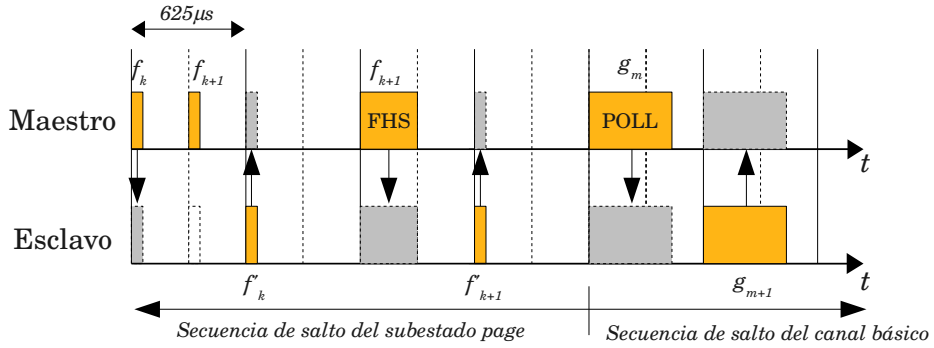

Figura 20: Intercambio de paquetes en el proceso de llamada

Por su parte, el esclavo debe entrar en algún momento en el subestado page scan, donde podrá ser configurado para usar un procedimiento de exploración estándar o entrelazado con el fin de establecer una conexión. Durante una exploración normal, el dispositivo escucha durante una ventana de tiempo (11,25ms por defecto), mientras que la exploración entrelazada se mejora con dos exploraciones, una tras otra, de la duración que se hubiere especificado para una exploración normal. Por este motivo, si el intervalo de

exploración no es al menos dos veces la ventana de exploración, el modo entrelazado no se podrá usar. Durante esta ventana, el dispositivo escuchará en una única frecuencia, mientras su "correlador" espera encontrar su código de acceso de dispositivo (DAC). La ventana de exploración deberá ser lo suficientemente larga como para que dé tiempo a completar la exploración de 16 frecuencias. Cuando el dispositivo entra en este estado, seleccionará la frecuencia de exploración de acuerdo a la secuencia de saltos determinada por la dirección Bluetooth del dispositivo. La fase de la secuencia se determinará por los bits del 12 al 16 del reloj nativo del dispositivo. Cada 1.28 segundos se seleccionara una frecuencia diferente. Este estado puede ser accesible desde los estados Standby y Connection.

El maestro entra en el subestado "Master Response" cuando recibe un mensaje de respuesta al paquete "page" por parte del esclavo. En esta situación, se tomará una muestra en el maestro del valor actual del reloj que se usa como entrada en el esquema de selección de salto para el subestado page. Este valor, se transmitirá en un paquete FHS con destino al esclavo. En él, se halla toda la información necesaria para construir el código de acceso al canal (CAC) sin requerir operaciones matemáticas para derivar la dirección de acceso del dispositivo Bluetooth maestro. El paquete FHS se transmitirá en el comienzo de la siguiente ranura de tiempo entre el maestro y el esclavo tras la respuesta del esclavo. Después de que se haya enviado, se esperará durante un segundo un mensaje de respuesta por parte del esclavo como asentimiento de la correcta recepción del paquete FHS. Esta respuesta contendrá el DAC del dispositivo esclavo. Si no se recibe respuesta, el maestro retransmitirá el paquete FHS con un valor actualizado del reloj y usando aún los parámetros del esclavo cada vez, hasta que se reciba un segundo mensaje de respuesta o el temporizador asociado al tiempo de espera de una respuesta se cumpla. En este último caso, el maestro volverá al subestado page y enviará un error al gestor de recursos de la banda base.

Por su parte, si la respuesta del esclavo se recibe, el maestro usará sus parámetros, es decir, pasará a utilizar su CAC y su reloj. Finalmente, el maestro entrará en el estado Connection. Una vez establecida la conexión, el primer paquete que el maestro enviará al esclavo será de tipo "poll" y el esclavo responderá con cualquier otro tipo de paquete válido, comenzando de este modo la transmisión de información.

RSSI

Una vez establecida una conexión se pueden monitorizar algunos

parámetros sobre la misma. Uno de ellos es el indicador de la fuerza de la señal recibida (RSSI). Este parámetro ofrece una estimación aproximada de la potencia recibida aunque con un matiz importante. En primer lugar, es conveniente indicar que la precisión de dicha medida depende del hardware utilizado. El parámetro RSSI se mide en decibelios y su lectura es 0dB cuando la potencia recibida se encuentra dentro del intervalo denominado "Golden Receive Power Range". Esto implica que cuando la potencia recibida se encuentra en dicho rango, un incremento o decremento ligero de la potencia no es detectable. El "Golden Receive Power Range" tiene dos umbrales: el inferior se corresponde a una potencia recibida entre los -56dBm y 6dB por encima de la sensibilidad real del receptor. El umbral superior está 20dB por encima del nivel inferior, con una precisión de ±6dB.

La definición de RSSI y la rápida variabilidad del parámetro* hacen prácticamente inviable el establecimiento de una relación sencilla entre el mismo y la potencia recibida. Este último parámetro, que puede ser relevante para ciertas aplicaciones como el posicionamiento, no siempre puede ser obtenido con precisión. Sin embargo, para otras aplicaciones, esta información puede ser suficientemente indicativa de la calidad del enlace.

La interfaz HCI

Las siglas HCI responden a "Host Controller Interface" e identifican a una interfaz que proporciona una serie de comandos para manejar algunos aspectos del controlador de la banda base. También permite el envío y recepción de datos desde o hacia las capas superiores, acceder al gestor de enlaces, al estado del hardware y a los registros de control. En esencia, esta interfaz ofrece un método homogéneo de acceso a las capacidades de la bandabase de Bluetooth y, a la vez, permite delimitar las funciones realizadas por el hardware de las funciones de los protocolos de niveles superiores, implementadas por software.

Desde el punto de vista del hardware, la interfaz permite independizar el equipo que controla la transmisión de las señales Bluetooth (como, por ejemplo, un dongle USB) del equipo que controla el intercambio de datos (como sería el caso de un ordenador). La especificación de Bluetooth ofrece algunas alternativas para la interconexión de ambos equipos como, por ejemplo, USB, UART o SDIO. De todas ellas, la tecnología USB es la más utilizada en la interfaz HCI.

* La experiencia práctica con dispositivos Bluetooth ha mostrado que el valor de RSSI fluctúa con mucha rapidez, lo que dificulta la obtención de medidas representativas.

Desde el punto de vista del protocolo, la especificación define claramente todas las posibilidades de interacción mediante los correspondientes comandos y eventos, disponibles de manera detallada en la parte E de la especificación Bluetooth [63].

La capa L2CAP

Dentro de la especificación Bluetooth, el protocolo L2CAP (Logical Link Control Adaptation Protocol, Protocolo de Adaptación y Control de Enlaces Lógicos) se encarga de la multiplexación de datos de los protocolos de niveles superiores y de las tareas de segmentación y recomposición de paquetes. También se encarga de transportar información de calidad de servicio entre un dispositivo y otro. L2CAP se halla en un nivel superior a la bandabase y puede englobarse en el nivel de transporte del modelo de referencia OSI.

El protocolo L2CAP está definido únicamente para enlaces asíncronos de tipo ACL. Esto se debe a que L2CAP depende de las comprobaciones de integridad en el nivel de bandabase para proteger la información transmitida, siendo los enlaces ACL los únicos capaces de satisfacer dichas exigencias a través de un esquema ARQN/SEQN de un bit. Entre las características de L2CAP están la sencillez y una baja sobrecarga del enlace, lo que le hace apropiado para su implementación en dispositivos con recursos de cálculo y memoria limitados como teléfonos móviles, sistemas empotrados o dispositivos PDA. Estas características le permiten alcanzar una elevada eficiencia de ancho de banda sin consumir demasiada energía, de acuerdo con los objetivos de eficiencia energética de la comunicación establecidos en la interfaz radio Bluetooth.

L2CAP realiza diversas operaciones para su funcionamiento, algunas de las cuales se detallan a continuación:

- **Multiplexación de protocolos y/o canales**. L2CAP es capaz de multiplexar flujos de las capas superiores para ser transmitidos a los esclavos.
- **Segmentación y reensamblado**. Si L2CAP tiene el control sobre la longitud de las tramas transmitidas, se favorece el funcionamiento de muchas aplicaciones multiplexadas sobre L2CAP.
- **Control de flujo por canal L2CAP**. Se implementa un control de flujo diferente al utilizado en la banda base, ya que este último puede no ser suficiente para la mayoría de las aplicaciones.

- **Control de errores y retransmisiones**. Algunas aplicaciones requieren una tasa de error residual mucho más pequeña que la suministrada por la bandabase. L2CAP incluye comprobaciones de errores adicionales y retransmisiones de tramas L2CAP.
- **Fragmentación y recombinación**. Las capas inferiores tienen limitadas las capacidades de transmisión, por lo que podrían requerir el uso de fragmentos de tamaños diferentes a los creados por la subcapa de segmentación de L2CAP. Por tanto, las capas por debajo de L2CAP pueden fragmentar y recombinar PDUs de L2CAP para crear fragmentos que se ajusten a las capacidades de la capa.
- **Calidad de servicio**. Durante el proceso de establecimiento de una conexión L2CAP se permite el intercambio de información atendiendo a la calidad de servicio esperada entre los dos dispositivos Bluetooth.

La capa L2CAP está basada en torno al concepto de canales. A cada una de las terminaciones de un canal L2CAP se le denomina identificador de canal (Channel Identifier, CID). De esta forma es como se conoce al nombre que representa a una terminación de un canal lógico en un dispositivo. Su alcance es local, es decir, únicamente afecta al dispositivo en cuestión.

Capas superiores. Los perfiles

Aunque L2CAP puede ser un punto de partida básico para el envío de datos entre dispositivos, en muchas ocasiones, para que un dispositivo pueda utilizar la tecnología inalámbrica Bluetooth, debe saber interpretar los perfiles que describen las distintas aplicaciones posibles. Estos perfiles son guías que indican los procedimientos por los que los dispositivos equipados con tecnología Bluetooth se comunican entre sí. Existe un amplio abanico de perfiles que detallan los diferentes tipos de uso y aplicaciones de la tecnología inalámbrica Bluetooth. Al seguir las directrices proporcionadas en las especificaciones Bluetooth, los desarrolladores pueden crear aplicaciones compatibles con otros dispositivos que se ajusten a este estándar. Cada perfil incluye, como mínimo, información sobre las siguientes cuestiones:

- Dependencia de otros perfiles.
- Propuestas de formato de interfaz de usuario.
- Características concretas de la pila de protocolos Bluetooth utilizada por el perfil. Para realizar su función, cada perfil se sirve de ciertas opciones y parámetros en cada capa de la pila. También se puede incluir un breve resumen de los servicios requeridos si resulta necesario.

En la tabla 11, se recogen algunos de los perfiles más importantes definidos por la especificación Bluetooth :

Perfil	Función
A2DP – Perfil de distribución de audio avanzado	Describe cómo transferir sonido estéreo de alta calidad de una fuente de sonido a un dispositivo receptor.
BPP – Perfil básico de impresión	Permite enviar mensajes de texto, de correo electrónico, tarjetas de visita electrónicas e imágenes, entre otras cosas, a las impresoras disponibles dependiendo de las tareas de impresión.
CTP – Perfil de telefonía inalámbrica	Describe la implementación de un teléfono inalámbrico a través de un enlace Bluetooth.
FAX – Perfil de fax	Describe cómo un dispositivo terminal puede utilizar a otro como puerta de enlace para la transmisión de faxes.
FTP – Perfil de transferencia de archivos	Establece los procedimientos de exploración de carpetas y archivos de un servidor a través de un cliente.
GOEP – Perfil genérico para el intercambio de objetos	Se utiliza para transferir objetos de un dispositivo a otro.
HFP – Perfil manos libres	Describe cómo un dispositivo que actúa de puerta de enlace puede utilizarse para realizar y recibir llamadas a través de un dispositivo manos libres.
HID – Perfil de dispositivo de interfaz humana	Recoge los protocolos, procedimientos y características empleados por las diferentes interfaces de usuario tales como teclados, ratones, consolas, etc.
OBEX – Protocolo de intercambio de objetos	Define los objetos de datos y los protocolos que deben usarse para el intercambio de objetos.
PAN – Perfil de	Describe cómo dos o más dispositivos con tecnología

Perfil	Función
redes de área perssonal	Bluetooth pueden formar una red ad hoc y cómo ese mismo mecanismo permite acceder a la red de forma remota a través de un punto de acceso.
SPP – Perfil de puerto serie	Describe cómo configurar puertos serie y conectar dos dispositivos a través de Bluetooth.
BNEP – Perfil de encapsulado de red	Se utiliza para encapsular y transportar protocolos de red comunes como TCP/IP a través de Bluetooth. Comúnmente se usa para crear pasarelas hacia ese tipo de redes.

Tabla 11: Principales perfiles en Bluetooth

Comparativa entre tecnologías inalámbricas

Con el objetivo de estudiar la idoneidad de las tecnologías inalámbricas presentadas para su uso en entornos urbanos inteligentes, se han comparado las principales características de cada una de ellas. El resultado obtenido se resume en la tabla 12:

Característica	Bluetooth	ZigBee	WiFi
Banda de frecuencia	2,4GHz	868/915MHz; 2,4GHz	2,4GHz; 5GHz
Velocidad máxima	3Mbps (EDR)	250Kbps	54Mbps (~500Mbps con 802.11n)
Alcance	1m / 10m / 100m	10m / 100m	Hasta 100m
Potencia de transmisión	0 a 20 dBm	-25 a 0 dBm	15 a 20 dBm
Tipo de modulación	GFSK, DPSK	BPSK, O-QPSK	BPSK, QPSK, COFDM, M-QAM
Espectro expandido	FHSS	DSSS	DSSS, OFDM
Mecanismo de coexistencia	Salto en frecuencia adaptativo	Selección dinámica de frecuencia	Selección dinámica de frecuencia

Característica	Bluetooth	ZigBee	WiFi
Topologías	Piconet, Scatternet	Estrella, "cluster tree", mallada	Punto de acceso, ad-hoc
Máximo número de nodos por celda	8	> 65000	2007
Encriptación	E0 stream cipher	AES	RC4, AES

Tabla 12: Comparativa técnica entre tecnologías inalámbricas

Todas las tecnologías analizadas utilizan técnicas de espectro expandido y utilizan preferentemente la banda libre de 2,4GHz. En los casos de ZigBee y WiFi también pueden usar otras bandas, pero éstas también son bandas libres para uso industrial, científico y médico. Dado que estas tecnologías pueden colisionar en la citada banda de 2,4GHz, todas ellas incorporan mecanismos para mejorar la coexistencia y minimizar las interferencias. Desde el punto de vista del número de nodos por celda, la limitación de 8 nodos de Bluetooth puede ser un inconveniente en aplicaciones de entornos urbanos inteligentes ya que sería factible que hubiera más de 7 usuarios en un determinado momento (se excluye el nodo que da acceso a los servicios).

La velocidad de transmisión es uno de los puntos críticos de la tecnología ZigBee. Esto es así porque está enfocada principalmente a aplicaciones tales como redes de sensores, donde no se necesita un gran ancho de banda. WiFi, por el contrario, se muestra como la opción con mayor ancho de banda, y Bluetooth se sitúa en un término medio. En principio, las aplicaciones orientadas a entornos urbanos inteligentes no tienen grandes requerimientos de ancho de banda, como podría ocurrir con la transmisión de vídeo. El ancho de banda ofrecido por ZigBee puede ser insuficiente para estas aplicaciones por lo que, atendiendo a este parámetro, sería más conveniente utilizar Bluetooth o WiFi. Evidentemente, mientras mayor sea el ancho de banda, antes se transmitirá la información, algo a tener en cuenta sobre todo en aplicaciones que involucre terminales embarcados en vehículos. En el tiempo de transmisión no sólo influye el ancho de banda, sino también el diseño de los protocolos. Atendiendo a este aspecto, la tecnología ZigBee, además de ser la más lenta, también tiene la menor eficiencia por la sobrecarga de cabeceras frente a Bluetooth y WiFi, que poseen un nivel de eficiencia similar [64]. Sin embargo, cuando la cantidad de información no es muy elevada, la diferencia entre un sistema u otro puede no ser demasiado importante. En esos casos, los mayores retrasos vienen determinados por los

tiempos de acceso, que en los casos de Bluetooth y WiFi pueden ser del orden de pocos segundos frente a los tiempos de ZigBee, que son del orden de las centenas de milisegundos. Estos tiempos suelen ser difíciles de estimar por varias razones:

- Influencia de aspectos configurables de cada tecnología.
- Carácter aleatorio de los mismos.
- Dependencia de factores externos como el entorno electromagnético, disposición de obstáculos, etc.
- División en varios procesos que, en muchos casos, pueden estar relacionados.
- Variabilidad entre dispositivos de distintos fabricantes o modelos.

El consumo de potencia es un factor muy importante en aplicaciones para entornos urbanos inteligentes, ya que muchos terminales pueden estar alimentados por baterías. En este sentido, Bluetooth y, sobre todo, ZigBee están pensados para su uso en este tipo de equipos. WiFi es la tecnología que mayor consumo energético tiene, lo que puede reducir notablemente la autonomía de estos terminales. Una comparativa de consumos entre las tecnologías involucradas en este apartado puede encontrarse en [64]. Obsérvese que el problema del consumo energético no sólo se limita a las operaciones de transmisión de datos sino también a las operaciones de descubrimiento de terminales. Si se aplica una topología basada en balizas y terminales, que es bastante utilizada en entornos urbanos inteligentes, se puede llegar a la conclusión de que es más conveniente que sean las balizas y no los terminales los que lleven a cabo las operaciones de descubrimiento que impliquen más consumo energético. Las balizas suelen formar parte de una infraestructura fija que, normalmente, tiene acceso a la red eléctrica u otras fuentes de energía no restrictivas, mientras que esto no suele ser posible en el caso de los terminales. En las tecnologías Bluetooth y WiFi los procesos de descubrimiento tienen asociados consumos energéticos no despreciables que deben ser tenidos en cuenta. En el caso particular de Bluetooth, existe la posibilidad de descargar esta responsabilidad en las balizas, pero para realizar un escaneo activo, que sería la actividad equivalente en tecnología WiFi, ésta debería llevarse a cabo por el terminal. Existe la posibilidad de utilizar el escaneo pasivo con WiFi pero, como contrapartida, aumenta el tiempo de acceso.

Otro factor a tener en cuenta es el coste de los dispositivos. Los dispositivos de las tecnologías Bluetooth y ZigBee tienen unos costes bastante bajos (en torno a los 2 ó 3 dólares), mientras que los de la tecnología WiFi son mayores [65].

El alcance del área de cobertura es un parámetro interesante, sobre todo en entornos urbanos inteligentes. A diferencia de lo que se pueda pensar, no siempre es conveniente que el alcance sea grande. En aplicaciones de guiado en interiores, por ejemplo, un tamaño limitado de cada celda puede ayudar a localizar con mayor precisión a un usuario. La granularidad es, por tanto, un factor decisivo para algunas aplicaciones. Si bien todas las tecnologías presentadas pueden alcanzar los 100m, no todas admiten un control más restrictivo del alcance. En el caso de WiFi, la granularidad es bastante grande, lo que constituye un impedimento para las aplicaciones mencionadas. Sin embargo, Bluetooth es la que ofrece una mayor variabilidad de la granularidad gracias a su clasificación por potencia de transmisión. De este modo, tenemos alcances del orden de 100m, 10m ó 1m, según se trate de un dispositivo de clase 1, 2 ó 3, respectivamente. Este hecho hace que Bluetooth sea la tecnología más interesante desde el punto de vista de la granularidad.

Otro aspecto importante a tener en cuenta en los entornos urbanos inteligentes es la identificación de las redes y de los servicios que ésta ofrece. Dada la amplia profusión de las tecnologías citadas, a la hora de realizar un despliegue de redes específicas para uno o varios servicios ha de tenerse en cuenta la implementación de algún mecanismo que discrimine las redes obtenidas en un proceso de descubrimiento. Un terminal no debería conectarse a cualquier red que esté a su alcance, sino sólo a aquellas redes que formen parte de una corporación concreta o que ofrezcan los servicios que el terminal espera obtener. La tecnología WiFi no dispone de medios específicos para realizar este filtrado, aunque se podría usar el propio identificador ESSID para codificar el tipo de red de acuerdo con alguna regla preestablecida. ZigBee resuelve este problema a través de los PAN ID extendidos, cuya finalidad es precisamente la identificación de redes específicas. Por su parte, Bluetooth dispone de varios mecanismos para llevar a cabo este filtrado:

- Usando códigos dedicados en el proceso de inquiry
- Utilizando la información de la clase de dispositivo
- Mediante los UUIDs contenidos en la respuesta extendida al inquiry

Característica	Bluetooth	ZigBee	WiFi
Granularidad	●●●	●●	●
Ancho de banda	●●	●	●●●
Eficiencia de los protocolos (carga útil)	●●●	●●	●●●
Rapidez en el acceso a la red	●●	●●●	●●
Menor consumo energético	●●	●●●	●
Menor precio	●●●	●●●	●●
Discriminación de redes	●●●	●●●	●

Tabla 13: Adecuación de las tecnologías inalámbricas a entornos urbanos inteligentes

Como resumen del estudio de las tecnologías inalámbricas, en la tabla 13 se muestra una comparativa sobre la adecuación de cada una de ellas a las características de interés en aplicaciones para entornos urbanos inteligentes. En la tabla se observa que la tecnología ZigBee obtiene muy buenos resultados en aspectos importantes como la rapidez en el acceso a la red. Sin embargo, su bajo ancho de banda supone una gran limitación, por lo que no resulta apropiada en muchos casos. Por otra parte, la tecnología Bluetooth muestra un buen comportamiento en la mayoría de las características estudiadas y destaca por su granularidad. Por último, la tecnología WiFi muestra algunas limitaciones fundamentalmente en aspectos tales como el consumo energético, la granularidad o los mecanismos que permiten la discriminación de redes. Por todos estos motivos se concluye que Bluetooth es la solución más apropiada para las aplicaciones en entornos urbanos inteligentes.

Comunicaciones en alto nivel. Middleware

El middleware es un software de conectividad que ofrece un conjunto de servicios que hacen posible el funcionamiento de aplicaciones distribuidas sobre plataformas heterogéneas. Funciona como una capa de abstracción de software distribuida, que se sitúa entre las capas de aplicaciones y las capas inferiores (sistema operativo y red). El middleware abstrae de la complejidad y heterogeneidad de las redes de comunicaciones subyacentes, así como de los sistemas operativos y lenguajes de programación, proporcionando medios para la fácil programación y manejo de aplicaciones distribuidas.

La diversidad de plataformas que se puede encontrar en muchos ámbitos de aplicación ha favorecido la introducción de las tecnologías middleware. Existen varios factores que han propiciado esta heterogeneidad, sobre todo en grandes sistemas:

- Sistemas heredados. Una organización podría aspirar a reemplazar sistemas antiguos con nuevas tecnologías. Sin embargo, esto suele ser en ocasiones demasiado caro y arriesgado, por lo que comúnmente se recurre a realizar una renovación progresiva, manteniendo sistemas antiguos que conviven con sistemas modernos.
- Nichos tecnológicos. Algunas plataformas hardware y software, por sus características específicas, escapan necesariamente a las normas corporativas de los sistemas, pero por su importancia estratégica deben mantenerse.
- Fusiones y adquisiciones. Como resultado de estos procesos, una organización puede verse en la necesidad de integrar plataformas y redes muy diferentes.

En la actualidad existen diversas soluciones middleware que varían en su estructura y complejidad. Algunas de las más conocidas son:

- RPC
- CORBA
- Servicios Web

El caso más básico y rudimentario lo constituyen las llamadas a procedimientos remotos (RPC). A lo largo de los años 80 se desarrolló esta tecnología y fue ampliamente utilizada. Para el código que las invoca (cliente) su uso es similar al de una llamada a una función local, aunque en realidad lo que hace es enviar a través de la red una referencia al tipo de función y la lista de parámetros. La implementación real de la función reside en otra máquina de la red (servidor) y, tras ejecutar la función, devuelve los resultados a través de la red. Estos procedimientos trabajan de forma síncrona, lo que provoca que el hilo del cliente quede bloqueado hasta que se complete la transacción. La simplicidad interna de las llamadas RPC hace que el nivel de abstracción sea bajo, lo que implica una menor facilidad a la hora de programar las aplicaciones frente a otras soluciones middleware.

El concepto de servicios Web abarca un conjunto de aplicaciones y tecnologías con capacidad para interoperar en la Web. Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web. En servicios Web las

aplicaciones se comunican mediante el protocolo SOAP (Simple Object Access Protocol) que, a su vez, se transporta sobre HTTP. Los datos que se intercambian usan el formato estándar XML. Esto resulta muy útil ya que permite la interpretación de la información por parte de aplicaciones que pueden estar realizadas con distintos lenguajes de programación. Sin embargo, el uso de XML y la orientación a formatos basados en texto reducen la eficiencia de los servicios Web frente a otras soluciones middleware debido a la sobrecarga de información y a los tiempos requeridos para la operación de la interfaz CGI (Common Gateway Interface). El protocolo HTTP, a pesar de ser ampliamente utilizado y conocido, tampoco destaca por su eficiencia y, además, supone un punto de vulnerabilidad a través del cual se pueden esquivar las medidas de seguridad basadas en cortafuegos (firewalls).

Por su parte, CORBA (Common Object Request Broker Architecture) ofrece una solución más completa e integrada que mejora en prestaciones a las plataformas comentadas anteriormente. A lo largo de los siguientes apartados se describirá con detalle las principales características que hacen de CORBA una solución adecuada para dar soporte a los servicios ofrecidos en entornos urbanos inteligentes.

CORBA

La especificación de CORBA surge como resultado del consenso entre los miembros del OMG (Object Management Group), una organización sin ánimo de lucro creada en 1989 formada con la misión de crear un mercado de programación basada en componentes, impulsando la introducción de objetos de programación estandarizada. Su principal propósito es desarrollar una arquitectura única, utilizando la tecnología de objetos para la integración de aplicaciones distribuidas y garantizando la reutilización de componentes, la interoperatividad y la portabilidad basada en componentes de programación disponibles comercialmente.

El OMG es una organización de estandarización de carácter neutro e internacional, ampliamente reconocida. Hoy en día son miembros del OMG alrededor de mil distribuidores de software, desarrolladores y usuarios que trabajan en diferentes campos, incluyendo universidades e instituciones gubernamentales. Además, mantiene estrechas relaciones con otras organizaciones como ISO, W3C, etc. El OMG no produce implementaciones, sólo especificaciones de software que son fruto de la recopilación y elaboración de las ideas propuestas por los miembros del grupo OMG.

CORBA es una arquitectura estándar para el desarrollo de aplicaciones distribuídas basadas en objetos. Permite que las clases que forman parte de las aplicaciones puedan ser implementadas en distintos lenguajes y que se ejecuten en distintas plataformas. CORBA se articula en torno a tres ideas fundamentales:

- Separación entre interfaz e implementación. A través del lenguaje de definición de interfaces (IDL) se especifican todos los componentes CORBA. IDL es un lenguaje puramente declarativo con una sintaxis muy parecida a la de C++, pero sin estructuras programáticas. Es independiente del lenguaje utilizado en la implementación, existiendo traductores (mappings) para diversos lenguajes como C, C++, Java, etc.
- Independencia de localización. El núcleo y componente más importante de cualquier implementación CORBA es el ORB (Object Request Broker). Éste se encarga de hacer transparente la localización de los objetos, enrutando las peticiones de manera que un objeto pueda comunicarse con otros independientemente de si ambos objetos se ejecutan en la misma máquina o en otras, incluso a través de redes heterogéneas.
- Independencia del fabricante. El protocolo GIOP (General Inter-ORB Protocol), y su variante específica para Internet (IIOP), permiten a ORBs de distintos fabricantes comunicarse de una manera estándar.

El ORB permite a los objetos realizar de forma transparente las invocaciones y recibir respuestas de otros objetos. En este sentido actúa como un bus. Para que un objeto cliente pueda invocar operaciones en un objeto servidor, el objeto cliente tiene que obtener una referencia al objeto servidor. Una vez hecho esto, podrá realizar la invocación de los métodos y acceder a los atributos de este objeto, definidos a través del IDL de la clase a la que este último pertenece.

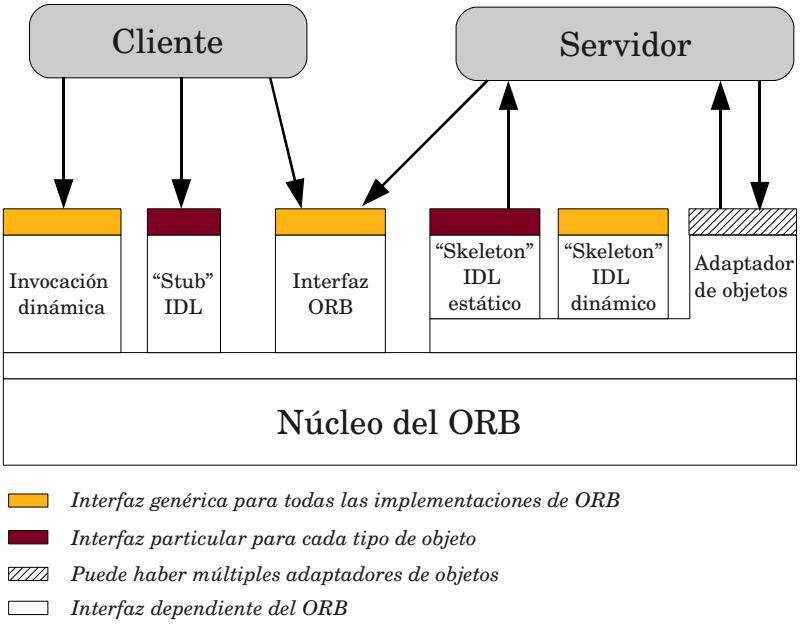

Figura 21: Estructura del ORB de CORBA

En la figura 18 se muestra la estructura del ORB de CORBA y las interfaces que el ORB ofrece a todos los objetos CORBA. El propio ORB ofrece un conjunto de servicios comunes a través de la interfaz ORB que pueden ser utilizados tanto por objetos cliente como por servidores. Algunas de estas operaciones básicas tienen que ver con la conversión de referencias a objetos. Las referencias a objetos son nombres complejos que identifican de manera unívoca al objeto en cuestión dentro del sistema. Tanto los clientes como los servidores necesitan unos adaptadores que transformen, en la parte cliente, una invocación local a una petición al ORB, y en la parte servidora, una invocación por parte del ORB en una invocación en el objeto servidor, que contiene la implementación real. En la parte cliente, estos trozos de código son denominados "stubs", mientras que en la parte servidora se denominan "skeletons". CORBA también proporciona interfaces para construir invocaciones de forma dinámica. La diferencia con las invocaciones estáticas está en que el cliente no necesita conocer los detalles del servidor ni su interfaz en tiempo de compilación, sino que se obtiene dicha información en tiempo de ejecución accediendo al repositorio de implementaciones (IR). En estos casos, los clientes hacen uso de la interfaz de invocación dinámica (DII), que intercambia datos con la interfaz de esqueleto dinámico (DSI) de forma

análoga a como lo hacen los "stubs" con los "skeletons" en las invocaciones estáticas. Las invocaciones dinámicas complican un poco el diseño y ralentizan las transacciones a cambio de obtener una mayor flexibilidad.

Finalmente, el adaptador de objetos se apoya en el repositorio de implementación para controlar el registro de servidores, activación y desactivación de las implementaciones, instanciación en base a varias políticas, etc. Existen varios adaptadores de objetos, como el BOA (Basic Object Adapter), aunque éste ha quedado ya obsoleto y actualmente el más utilizado es el POA (Portable Object Adapter).

El desarrollo de un objeto CORBA se divide en las siguientes etapas:

1. Diseño. Se determinan los servicios que debe proporcionar el objeto y la implementación de las interfaces IDL correspondientes.
2. Compilación de IDL. Mediante el compilador de IDL se procesan las definiciones de interfaces y se generan los skeletons y stubs correspondientes, en un lenguaje de programación determinado.
3. Implementación de "servants". Utilizando las clases generadas por el compilador de IDL, se crean los "servants" de la aplicación, que contienen la implementación de las operaciones del objeto CORBA.
4. Implementación del servidor. El objeto CORBA debe proporcionar la infraestructura base para que la comunicación con el ORB sea posible.
5. Compilación. Se procede a la compilación de los archivos de código fuente. Debe incluirse el código generado automáticamente por el compilador de IDL (la parte correspondiente a los skeletons).

CORBA permite la definición de servicios adaptados a las necesidades de prácticamente cualquier aplicación. Sin embargo, existen ciertos servicios básicos que, por su utilidad, tienen definiciones estandarizadas y están disponibles en la mayoría de los ORBs. A continuación se citan algunos de los más relevantes:

El servicio de nombres es utilizado por los objetos cliente para buscar referencias a determinados servicios que, previamente, han publicado los objetos servidores. Para localizar los servicios con mayor facilidad, estas referencias están asociadas a un nombre que identifica al servicio. El servicio de nombres constituye una forma estandarizada y estructurada de obtener referencias IOR. Adicionalmente, el servicio de nombres permite establecer diferentes contextos de nombrado y la creación de federaciones de servidores de nombres, extendiendo aún más las posibilidades de localización de servicios.

El servicio de eventos permite la notificación asíncrona de eventos acompañados de información adicional. La mayoría de los servicios CORBA se realizan de forma síncrona, es decir, que el invocante queda bloqueado hasta que termina la operación. En determinadas situaciones esto no es deseable, por lo que se recurre al servicio de eventos para lograr un comportamiento asíncrono. El servicio de eventos se basa en un modelo productor-consumidor a través de un canal de eventos. Los productores de eventos los inyectan en el canal y éstos pueden ser escuchados por varios consumidores simultáneamente. La suscripción a estos canales de eventos, tanto como productor como consumidor, admite dos modos de funcionamiento: "push" y "pull" ("empujar" y "tirar"). Dependiendo del modo y el papel del objeto en cuestión deberá implementar una interfaz servidora o cliente y deberá conectarse con un determinado tipo de objeto estandarizado perteneciente al servicio de eventos. La figura 19 ilustra las posibilidades de suscripción a un canal de eventos:

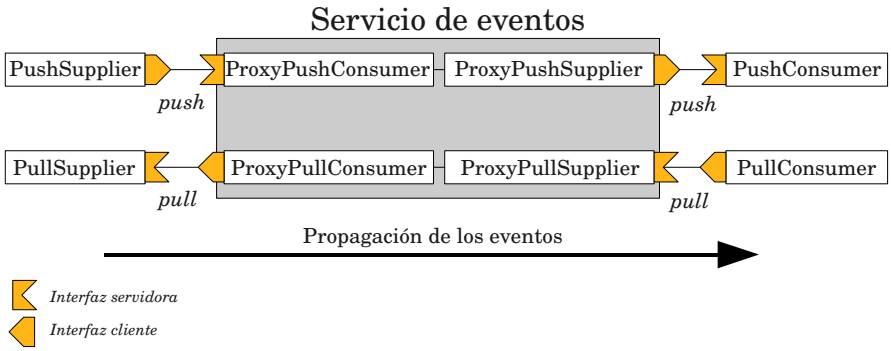

Figura 22: Flujo de eventos

El servicio de "trading" es, en cierta manera, similar al servicio de nombres, ya que su finalidad también es la localización y obtención de las referencias a los servicios. La diferencia con el servicio de nombres está en que el servicio de trading busca por tipo de servicio. El servicio de nombres podría ser comparable a las páginas blancas, donde los usuarios buscan números de teléfono por el nombre, mientras que el servicio de trading podría equipararse a las páginas amarillas, donde el usuario lo que busca es un tipo de servicio. Los objetos servidores pueden publicar información adicional que permita a los objetos cliente realizar búsquedas paramétricas. Al igual que el servicio de nombres, tiene un carácter estructurado y permite la federación.

Actualmente existen diversas implementaciones de CORBA, algunas de ellas libres como MICO, omniORB o TAO, y otras comerciales como ORBexpress, Orbacus o Visibroker. Cada una de estas plataformas tienen características y enfoques distintos, pero entre ellas cabe destacar MICO, por ser la única que implementa la especificación Wireless CORBA.

Wireless CORBA

La mayoría de los ORB recurre a TCP como protocolo del nivel de transporte. Sin embargo, TCP puede no ser la solución más adecuada cuando se trata de comunicaciones inalámbricas [66]. Conscientes de los problemas relacionados con la movilidad y las comunicaciones inalámbricas, el grupo "Telecommunications Domain Task Force" (DTF) del OMG solicitó una petición de información en Junio de 1998 para tratar de determinar qué cuestiones necesitaban ser resueltas y cómo éstas habían sido resueltas en otras tecnologías. Como resultado de la consulta, el grupo creó un borrador detallando las cuestiones relacionadas con la movilidad. Finalmente, en Mayo de 1999, el DTF solicitó una petición de propuestas. En el año 2000, el OMG recibió dos propuestas que finalmente fueron fusionadas en una sola. Entre las empresas que redactaron la propuesta estaba Nokia que, a partir de ese año, lanzaría el proyecto Vivian [67]. El objetivo de este proyecto fue el desarrollo de una plataforma adecuada para teléfonos inteligentes con servicios middleware que facilitaran la comunicación entre ellos y el desarrollo de aplicaciones para este tipo de dispositivos. El trabajo presentado como propuesta al OMG se consolidó y en Junio de 2001 el OMG publicó la especificación de Wireless CORBA [68]. Posteriores correcciones sobre la especificación se llevaron a cabo en 2002 para mejorar algunos aspectos.

La especificación de Wireless CORBA define tres dominios en los que divide el sistema:

- Dominio local. Es el dominio administrativo del terminal. Si el terminal pertenece a una red corporativa, en este dominio se encontrarán sus principales servicios. Uno de los elementos más representativos de este dominio es el "Home Location Agent" (HLA), sobre el que recae la responsabilidad de hacer un seguimiento de la ubicación de los terminales que pertenecen a su dominio. La existencia de un dominio local en un sistema no es obligatoria, por lo que puede haber terminales que no estén asociados a ningún dominio local. En este caso, los terminales reciben el calificativo de "nómadas".
- Dominio visitado. Es un dominio distinto del local al que los

terminales acceden para hacer uso de sus servicios.

- Dominio del terminal. Este dominio comprende únicamente el ámbito de las aplicaciones en el terminal móvil.

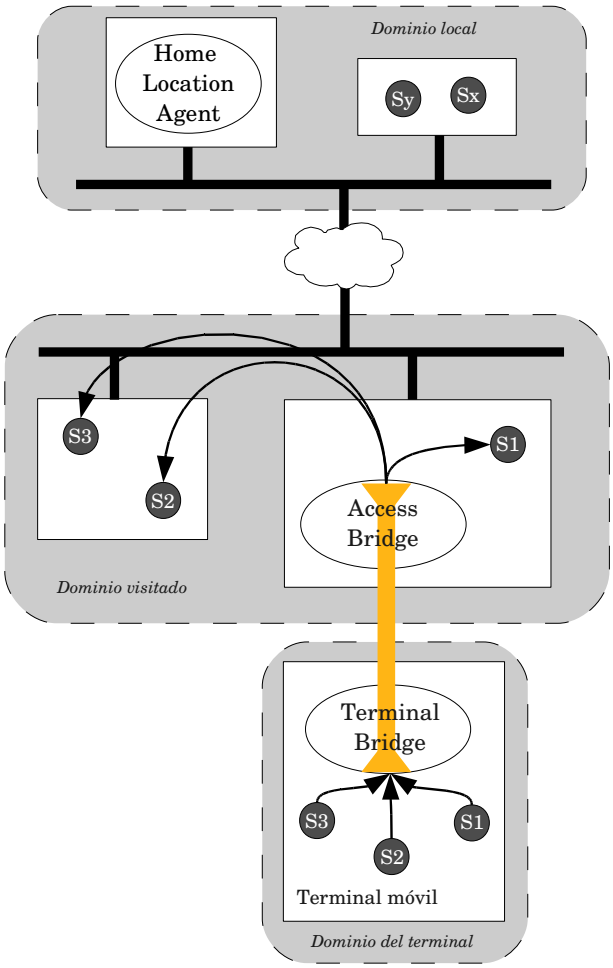

Figura 23: Dominios y elementos en Wireless CORBA

La especificación se centra fundamentalmente en la descripción de la interacción entre estos dominios a través de ciertos elementos. Además del HLA, descrito anteriormente, existen otros dos elementos relevantes: "Access Bridge" y "Terminal Bridge". Access Bridge es el elemento más representativo del dominio visitado y constituye la puerta de acceso a los servicios ofrecidos en una red para los terminales que lo visitan. Terminal Bridge, por su parte,

es un elemento esencial dentro del dominio del terminal ya que todas las aplicaciones CORBA que acceden a servicios del dominio visitado encaminan sus peticiones a través de él. Es Terminal Bridge el que tiene normalmente la iniciativa para establecer conexiones con los Access Bridge. En la figura 20 se muestra el conjunto de elementos y dominios que componen Wireless CORBA.

Habitualmente, las aplicaciones en los terminales actúan como clientes de los servicios CORBA, aunque también puede darse el caso contrario: que las aplicaciones servidoras estén en los terminales. En esta situación, el HLA juega un papel de intermediario capaz de redirigir las peticiones de los clientes al Access Bridge que en cada momento esté dando acceso al terminal que contiene la aplicación servidora. Esto es posible porque el HLA guarda un registro actualizado de los dominios visitados en los que se encuentra cada terminal perteneciente a su dominio. La información de rutado hacia el objeto servidor en las transacciones CORBA ordinarias va incluida en las referencias IOR (Interoperable Object Reference). Este sistema funciona correctamente cuando dichos objetos se ejecutan en una máquina concreta conectada a la red y perfectamente identificada por su dirección IP, pero este esquema no es aplicable en el caso de conexiones inalámbricas, ya que no se sabe a priori a qué Access Bridge está conectado el terminal que contiene el objeto servidor. Por este motivo, Wireless CORBA define el concepto de Mobile IOR, que amplía la información contenida en la referencia para lograr, junto con la intervención del HLA, un correcto encaminamiento en estos casos. La estructura de las referencias Mobile IOR está definida mediante el lenguaje IDL.

Cuando Access Bridge y Terminal Bridge establecen una conexión, crean un túnel mediante el uso del protocolo GTP (GIOP Tunneling Protocol) por el que circularán los mensajes GIOP que contienen las transacciones CORBA. GTP es un protocolo abstracto diseñado para ser utilizado con diversos protocolos del nivel de transporte. Los requisitos de fiabilidad de GTP son los mismos que para GIOP, por lo que habitualmente se recurre a la incorporación de capas de adaptación entre GTP y el protocolo del nivel de transporte para garantizar dichos requisitos. Un caso particular de capa de adaptación es LTP (L2CAP Tunneling Protocol), que adapta GTP para su transmisión sobre L2CAP. Esta capa permite la realización de transacciones CORBA a través de redes Bluetooth y es una de las contribuciones del proyecto Vivian. Como se vio anteriormente, entre los perfiles que define la especificación de Bluetooth se encuentra el perfil BNEP, que está pensado para establecer pasarelas hacia redes TCP/IP. En una primera aproximación se puede pensar que BNEP podría ser la solución idónea, ya que, de cara a las aplicaciones CORBA, proporcionaría una interfaz de red TCP/IP, con lo que se

podría usar directamente el protocolo IIOP y no sería necesario recurrir a GTP. Sin embargo, si se analiza un poco la estructura de protocolos del perfil se observa que existe una sobrecarga de cabeceras. En efecto, a las cabeceras aplicadas hasta el nivel de L2CAP hay que apilar las propias de BNEP y las correspondientes a TCP/IP, por lo que esta opción es claramente ineficiente. La alternativa propuesta en el proyecto Vivian de usar GTP con L2CAP ofrece una menor sobrecarga de cabeceras, a la vez que proporciona capacidad de multiplexación y demultiplexación de protocolos a las capas superiores [69]. L2CAP ofrece un servicio de datos orientado a conexión, con un canal fiable que mantiene el orden de la secuencia de mensajes. Uno de los puntos críticos es el límite en el tamaño de los paquetes L2CAP, pero la capa LTP se encarga de solucionar esto mediante la fragmentación y reensamblaje de los mensajes GTP.

Otra de las características llamativas de Wireless CORBA es la capacidad de los terminales para cambiar de un Access Bridge a otro adyacente que pertenezca al mismo dominio sin interrumpir las comunicaciones en curso. Este concepto se denomina "handoff" y es aplicado a otras tecnologías inalámbricas como, por ejemplo, las redes de telefonía móvil. Tanto Access Bridge como Terminal Bridge ofrecen funciones a través de sus interfaces CORBA para realizar estos cambios. Esta posibilidad es bastante interesante ya que permite a los usuarios desplazarse libremente por un dominio sin interrupciones en el servicio. Estos procedimientos, transparentes para las aplicaciones de usuario, están controlados por Access Bridge, Terminal Bridge y, opcionalmente, el HLA.

4 APORTACIONES PARA EL DESARROLLO DE WIRELESS CORBA Y APLICACIONES

Wireless CORBA en entornos inteligentes

En los estudios presentados en los capítulos anteriores se destacan dos tecnologías por su potencial para ser utilizadas en entornos urbanos inteligentes: Bluetooth en el ámbito de las comunicaciones inalámbricas y CORBA en el ámbito del middleware.

El punto de encuentro entre ambas tecnologías (CORBA y Bluetooth) se produce en la especificación de Wireless CORBA presentada por el OMG. En esta especificación se sientan las bases para dar soporte a CORBA sobre redes inalámbricas y, en particular, sobre Bluetooth. Si bien esta combinación se presenta como una solución efectiva al problema de las comunicaciones en entornos inteligentes, existen determinados conceptos y limitaciones en su diseño que no permiten un encaje directo en este tipo de entornos. A pesar de que Wireless CORBA posee algunos mecanismos interesantes como el de roaming, está principalmente pensado para ser usado en entornos donde los terminales se mueven por redes pertenecientes al mismo dominio [70].

No obstante, en los entornos urbanos inteligentes la situación es distinta; el número de terminales que se conectarán a la red no se conoce a priori. Por otra parte, los terminales tampoco conocen el número de servidores activos en la red ni los servicios que éstos ofrecen. En consecuencia, se debe considerar un escenario distinto donde unos terminales "nómadas" entran y salen del área de cobertura de la red, pudiendo incluso trasladarse hacia otras redes que pertenezcan a dominios distintos y ofrezcan otros servicios. Por este motivo surge la necesidad de disponer de gestores de las conexiones que se encarguen de las siguientes tareas:

- Establecer conexiones
- Liberar conexiones
- Facilitar a las aplicaciones del terminal el acceso a los servicios que ofrece la red
- Detectar dispositivos

Con este enfoque, a lo largo de este capítulo se describirá un sistema que, combinado con Wireless CORBA, resuelve los problemas planteados anteriormente y ciertos detalles de implementación para que pueda ser utilizado en entornos urbanos inteligentes. En el sistema propuesto, cada

dominio se compone de un conjunto de balizas interconectadas entre sí por medio de una red TCP/IP fija y dan servicio a un conjunto indeterminado de terminales a través de enlaces Bluetooth. Desde el punto de vista del software, la solución se divide en dos conjuntos de módulos, uno para las balizas y otro para los terminales. Cada uno de estos conjuntos sigue un esquema parecido, ya que combinan un gestor de conexiones con uno de los extremos del puente (AccessBridge en el caso de la baliza y TerminalBridge en el caso del terminal). La arquitectura interna de los gestores de conexión también es parecida ya que ambos se articulan en sendos objetos de política de red, que son los que tienen la responsabilidad de ordenar establecer y liberar conexiones. Los gestores se apoyan no sólo en las funcionalidades de AccessBridge y TerminalBridge, sino también en ciertos módulos especiales que intervienen en el envío y recepción de la información básica de red a través de un protocolo creado al efecto, al que se ha denominado NCC (Network Control Channel).

Dentro de las funciones de los gestores de conexión, la detección de dispositivos merece un tratamiento especial por su relevancia en el acceso a los servicios, máxime cuando la interfaz inalámbrica es compartida para las labores de detección y transmisión de datos de servicio. Por este motivo, en este capítulo se hará una caracterización de una de las componentes más importantes del tiempo de servicio en función de los parámetros sobre los que los objetos de política de red pueden intervenir. Como resultado del análisis de dicha caracterización se ofrece una serie de recomendaciones que pueden ayudar a mejorar los tiempos de servicio.

Mejoras en el tiempo de detección

Uno de los parámetros más importantes a tener en cuenta en una red inalámbrica en entornos urbanos es el tiempo de servicio. Un tiempo de servicio demasiado elevado puede provocar que el terminal salga de la zona de cobertura antes de completar la petición. En el tiempo de servicio intervienen muchos factores, por lo que es prácticamente imposible caracterizar este tiempo de forma rigurosa. Entre estos factores se pueden destacar los siguientes:

- Tiempo de llegada. Un terminal puede llegar a la zona de cobertura en cualquier momento.
- Tiempo de permanencia. El tiempo que un terminal permanece dentro del área de cobertura es aleatorio y puede depender en parte de la velocidad a la que se mueve el terminal.

- Consideraciones electromagnéticas del entorno. Dependiendo del entorno se pueden producir atenuaciones, interferencias por multitrayecto, etc. Esto afecta a la calidad del enlace, que puede variar en función del tiempo y la posición.
- Ancho de banda disponible. Depende del número de dispositivos que haya en la piconet y del reparto que se haga del ancho de banda.
- Frecuencia del proceso de búsqueda. Mientras más frecuente sea el proceso de búsqueda más rápido se detectarán los terminales, pero se reducirá el ancho de banda efectivo.
- Cantidad de datos a intercambiar. Esta característica es inherente al propio servicio y puede variar con el tipo particular de petición o los parámetros que se le indiquen.

Esta diversidad de factores, unida al carácter aleatorio en algunos casos y a las dependencias que pueden establecerse entre ellos, complica el estudio del tiempo de servicio e impide crear un modelo válido en todas las situaciones. Por otra parte, un estudio particular de cada caso es inviable. Sin embargo, se pueden estudiar ciertos factores por separado para establecer reglas que ayuden a reducir el tiempo de servicio. Para facilitar el análisis, el tiempo de servicio se puede descomponer como sigue:

$$t_{SRV} = t_{DET} + t_{EP} + t_{TRANS}$$

Donde:

- t_{DET} es el "tiempo de detección" y se define como el tiempo que transcurre desde que un terminal entra en el área de cobertura de una baliza hasta que la baliza tiene conocimiento de su presencia. Este último instante coincide con la finalización del proceso de inquiry dentro de un ciclo. El tiempo de detección puede contribuir de manera significativa al tiempo de servicio, motivo por el cual será objeto de estudio. Este tiempo tiene carácter aleatorio y depende, en gran medida, del inquiry cycle.
- t_{EP} es el "tiempo de establecimiento del puente" y se define como el tiempo que transcurre desde que un terminal es detectado hasta que queda establecida la comunicación entre TerminalBridge y AccessBridge. Este tiempo tiene carácter aleatorio.
- t_{TRANS} es el "tiempo de transacciones" y se define como el tiempo que transcurre desde que las aplicaciones son notificadas de la disponibilidad del puente hasta que se completa la última transacción CORBA. Esto incluye accesos a servidores de nombre o de "trading"

que ofrezcan referencias a los servicios concretos que el terminal quiera utilizar. Este tiempo tiene carácter aleatorio y depende, en gran medida, de la cantidad de datos a intercambiar.

A lo largo de este apartado se analizarán los puntos de mejora que permiten conseguir una reducción del tiempo de detección apoyándose en las particularidades técnicas de la tecnología Bluetooth.

Búsqueda selectiva

Como ya se mencionó en el capítulo 3, tras el proceso de inquiry se obtiene una lista de los dispositivos descubiertos que, además, incluye información útil sobre los dispositivos como, por ejemplo, la clase de dispositivo. En el caso concreto de los entornos urbanos, hacer uso de este parámetro puede mejorar la eficiencia. Hoy en día muchos dispositivos, como teléfonos móviles, ordenadores, etc., disponen de conectividad Bluetooth. Es bastante probable que estos dispositivos pudieran encontrarse inmersos en una red Bluetooth para entornos urbanos inteligentes como las que se describen en esta Tesis y que, sin embargo, no vayan a hacer uso de sus servicios. Tras el proceso de inquiry, una baliza detectará todos aquellos terminales que estén a su alcance y, a continuación, intentará establecer una conexión con ellos para trasladarles la información básica de red aunque fracasará si los terminales no están preparados para recibir este tipo de información. Esto implica que la eficiencia de la baliza se vería reducida por el establecimiento de conexiones inútiles a dispositivos que no están preparados para hacer uso de sus servicios. Por este motivo es más conveniente filtrar la lista en base a la clase de dispositivo y establecer conexiones únicamente a aquellos terminales que estén interesados en consultar los servicios que ofrece la baliza. Para realizar dicho filtrado se ha definido un servicio específico siguiendo el sistema de numeración descrito en el capítulo 3. El número que deberían usar los terminales para el identificador de la clase de dispositivo es el 0x820010. Con este número se indica que se soportan las clases de servicio de información y networking. En el campo “Major Device Class” se ha utilizado el valor para dispositivos misceláneos y dentro de esta clasificación se utiliza el valor binario particular 100000. De esta manera, todos los dispositivos que respondan a esta clase de dispositivo recibirán automáticamente la información básica de los servicios que ofrecen las balizas cuando estén al alcance de éstas.

Modo entrelazado

El modo entrelazado para el proceso de inquiry, descrito en el capítulo 3, mejora considerablemente los tiempos de descubrimiento tal y como se demuestra de forma analítica en [71], donde se caracteriza el tiempo que transcurre entre el inicio del proceso de inquiry y el instante en que el dispositivo es detectado. Los resultados experimentales corroboran este hecho y muestran una alta probabilidad de descubrimiento incluso para un tiempo de inquiry de tan sólo 1,28 segundos, tal y como se aprecia en la figura 24.

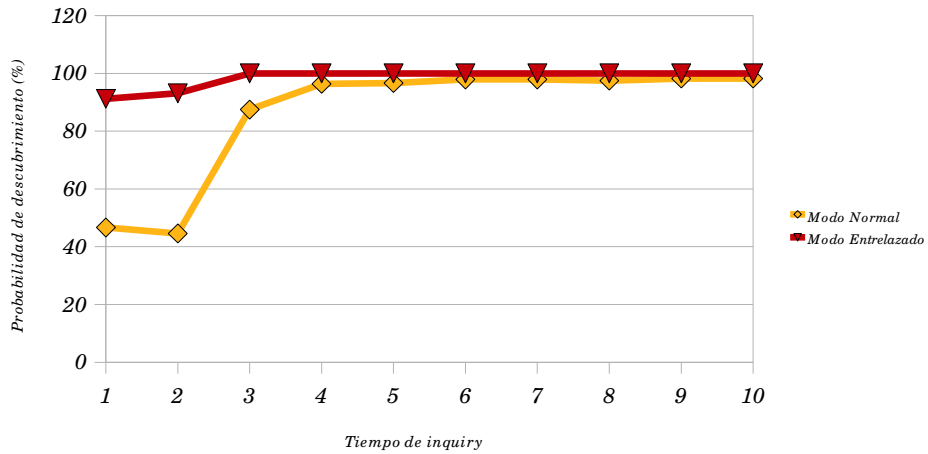

Figura 24: Probabilidad de descubrimiento frente a tiempo de inquiry

Las pruebas experimentales se realizaron con 15 dispositivos Bluetooth y se realizaron 30 medidas para cada uno de los tiempos de inquiry representados. El tiempo de inquiry está expresado en bloques de 1,28 segundos.

El uso del modo entrelazado tiene una pequeña repercusión sobre el throughput* visto desde el lado del terminal. A diferencia del modo normal, cada vez que el dispositivo entra en el estado de Inquiry Scan, se abren dos ventanas seguidas de 11,25ms cada una. Dado que, por defecto, esto ocurre cada 1,28 segundos, el enlace deja de estar disponible para la transmisión de paquetes ACL durante el 1,75% del tiempo frente al 0,87% en el caso del modo normal. Como se puede apreciar, la mejora en el tiempo de descubrimiento compensa con creces la pérdida en el throughput frente al modo normal. Para

* El throughput puede entenderse como una medida de la velocidad media de datos en una conexión.

aprovechar esta ventaja, se han introducido en la inicialización de NCCserver ciertas sentencias para cambiar a modo entrelazado. De esta manera, todos los terminales quedan configurados en este modo al iniciar las aplicaciones.

Throughput frente a Inquiry Cycle

El Inquiry Cycle determina la relación entre el tiempo que se dedica a la búsqueda de dispositivos por parte de la baliza y el que se dedica al intercambio de información. Este parámetro es sumamente importante no sólo porque afecta directamente al throughput sino también porque influye en los tiempos de detección de los terminales. En una primera aproximación se puede decir que mientras más frecuente y más largo sea el tiempo dedicado al inquiry más se facilitará la detección de dispositivos, pero menor será el throughput.

Por simplicidad, los tiempos de inquiry y de datos se considerarán en bloques de 1,28 segundos, que es el tiempo mínimo de inquiry. De esta forma, el inquiry cycle se expresará como relaciones (i:d), donde i es el número de bloques dedicados al inquiry y d el número de bloques dedicados a datos. La figura 25 muestra un ejemplo de reparto con dos bloques dedicados a inquiry y tres a datos.

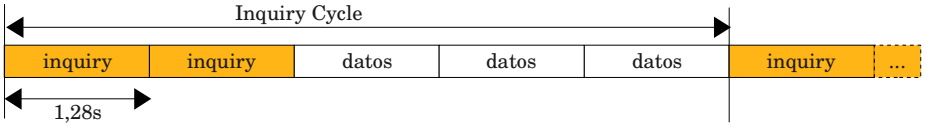

Figura 25: Ejemplo de inquiry cycle (2:3)

Influencia directa sobre el throughput

El throughput de una conexión concreta en bits/s depende de varios factores como el tipo de paquetes usados, la versión del controlador Bluetooth, el número de conexiones existentes en la piconet, la tasa de errores de bit (BER), etc. Por este motivo, no tiene sentido intentar calcular cifras exactas. En su lugar, se calculará la influencia sobre el throughput en términos de porcentaje sobre el throughput que se obtendría si se omite el proceso de inquiry. Este porcentaje estima el valor medio relativo y es fácil de obtener a través del propio reparto. La tabla 14 muestra estos porcentajes para algunas de las estrategias de reparto más sencillas.

		Inquiry		
		1	2	3
Datos	1	50%	33%	25%
	2	66%	50%	40%
	3	75%	60%	50%

Tabla 14: Afectación del troughput en función del inquiry cycle

Dependiendo del tráfico de datos que se necesite para llevar a cabo un servicio concreto, esta reducción del throughput puede ser o no un factor limitante. Afortunadamente, muchos servicios para entornos urbanos sólamente requieren el intercambio de datos sencillos, por lo que se puede sacrificar el throughput en favor de un menor tiempo de descubrimiento. Para servicios que requieran una mayor transferencia de datos puede ser más conveniente optar por repartos con menor peso del inquiry.

Influencia sobre el tiempo de detección

El tiempo de detección, tal y como se ha definido en esta Tesis, difiere del concepto de tiempo de descubrimiento usado en [71] y otros estudios relacionados como [72] y [73]. Por este motivo, en esta Tesis se ha optado por obtener la función de distribución de probabilidad para el tiempo de detección considerando diversos esquemas para el inquiry cycle y asumiendo que los terminales pueden entrar en cobertura en cualquier instante del ciclo de inquiry. Es precisamente ese instante desde donde empieza a contar el tiempo de detección, a diferencia de los artículos citados, donde se asume que los terminales ya están en la zona de cobertura cuando se inicia el proceso de inquiry. Otra diferencia es que el tiempo de detección termina necesariamente cuando finaliza una fase de inquiry en lugar de terminar tras la recepción del paquete FHS, como ocurre con el tiempo de descubrimiento.

La obtención empírica del tiempo de detección es realmente difícil puesto que los dispositivos Bluetooth no notifican el instante en el que “entran en cobertura”. En cualquier caso, este concepto es algo difuso, por lo que no se puede medir con exactitud. Por estas razones se optó por simular el experimento mediante un programa en C++ creado al efecto que emula el

comportamiento de los dispositivos Bluetooth. El simulador implementa únicamente aquellas partes del estándar* relacionadas con el proceso de inquiry que son relevantes de cara al experimento. Para el desarrollo del programa se ha asumido lo siguiente:

- El instante de entrada en cobertura se modela como una variable aleatoria de distribución uniforme que abarca todo el inquiry cycle.
- La duración de las fases de inquiry y datos se cuantifica en bloques de 1,28 segundos.
- Sólo hay un dispositivo esclavo que entra en el área de cobertura de una baliza, por lo que no se producen colisiones con otros dispositivos.
- La respuesta al inquiry no es extendida.
- El dispositivo esclavo utiliza el modo entrelazado para el rastreo de dispositivos abriendo dos ventanas seguidas de 11,25ms cada 1,28s.
- Todos los paquetes se transmiten sin errores.

Con estas condiciones se realizaron simulaciones para algunas de las estrategias más habituales de reparto del inquiry cycle. En cada simulación se obtuvieron del orden de las 20000 muestras para el tiempo de detección, y posteriormente se clasificaron para obtener una aproximación de la función de densidad de probabilidad. En la figura 26 se muestra la probabilidad para distintos tiempos de detección agrupados en intervalos de 0,1s cuando se utiliza un reparto 1:1.

* Para el desarrollo del simulador se ha tenido en cuenta la especificación Bluetooth en su versión 4.0, que es la más reciente hasta la fecha.

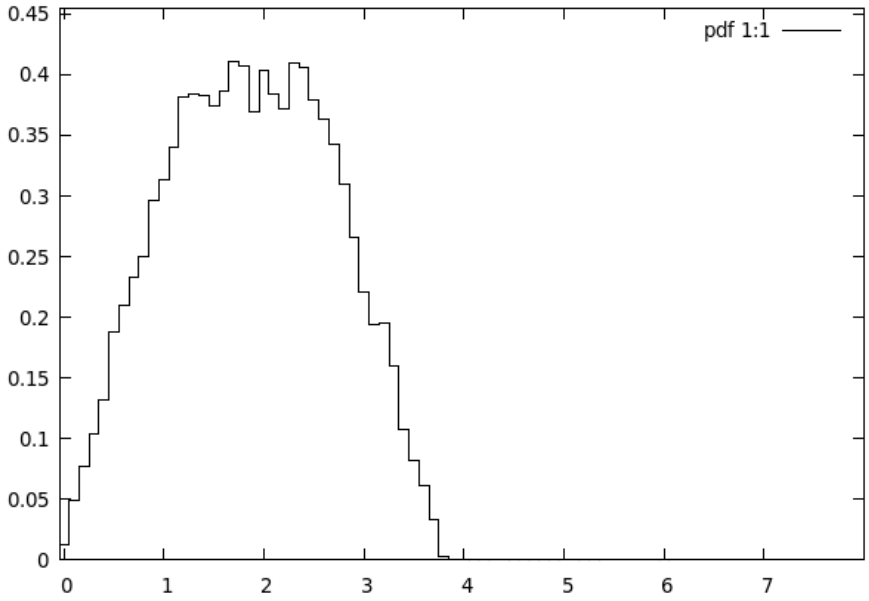

Figura 26: Densidad de probabilidad con un reparto 1:1

Se aprecia que la función de densidad de probabilidad tiene una forma predominantemente trapezoidal. La anchura de este trapecio es proporcional al tiempo total del inquiry cycle. Concretamente, la anchura de dicho trapecio es aproximadamente 1,28+Tcycle, donde Tcycle es el tiempo total que dura un inquiry cycle.

Las gráficas de las funciones de densidad de probababilidad para el resto de los repartos más habituales también siguen una forma trapezoidal y pueden encontrarse en el anexo A3. A través de las muestras y gráficas obtenidas se extraen los parámetros característicos para cada uno de los repartos. Los valores medios y de desviación típica para el tiempo de detección, ambos en segundos, se muestran en la tabla 15.

		Inquiry		
		1	**2**	**3**
Datos	**1**	μ=1,9188 *σ=0,8293*	μ=2,5593 *σ=1,1701*	μ=3,1929 *σ=1,5270*
	2	μ=2,5564 *σ=1,1741*	μ=3,1993 *σ=1,5315*	μ=3,8423 *σ=1,8788*
	3	μ=3,2003 *σ=1,5398*	μ=3,8524 *σ=1,8847*	μ=4,5033 *σ=2,2506*

Tabla 15: Valores medios y desviaciones típicas del tiempo de detección

Comprobación experimental del tiempo de detección

Por la propia definición del tiempo de detección, su comprobación experimental es bastante compleja. La principal dificultad radica en la imposibilidad de conocer el instante en el que un terminal entra en el área de cobertura ya que depende de varios factores, como la distancia a la baliza, los obstáculos, la potencia de transmisión de la baliza, la sensibilidad del receptor y otros factores concretos para cada escenario. Sin embargo, existe una alternativa para hacer que un terminal entre en cobertura de forma súbita. Si un terminal está dentro del área de cobertura de una baliza pero tiene desactivado el proceso de inquiry scan, es como si estuviera fuera de ella a efectos prácticos. La activación del proceso de inquiry scan en un momento determinado equivale a una entrada repentina del terminal en el área de cobertura pero con la salvedad de que dicho instante es conocido. El instante de finalización del tiempo de detección, por el contrario, se puede obtener de una manera sencilla, ya que coincide con el instante en el que termina el proceso de inquiry en la baliza.

Para la obtención experimental del tiempo de detección se han utilizado dos PCs con conectividad Bluetooth situados a corta distancia. Dado que los instantes de inicio y finalización del tiempo de detección se toman en distintas máquinas, se ha diseñado un programa que utilice una conexión Ethernet para la notificación y sincronización de ambas máquinas. De este modo, se obtienen marcas temporales sobre la misma referencia temporal. La transmisión de esta información de sincronización a través de una red Ethernet requiere un tiempo, aunque éste es despreciable frente al tiempo de detección, por lo que puede ignorarse de cara a los cálculos.

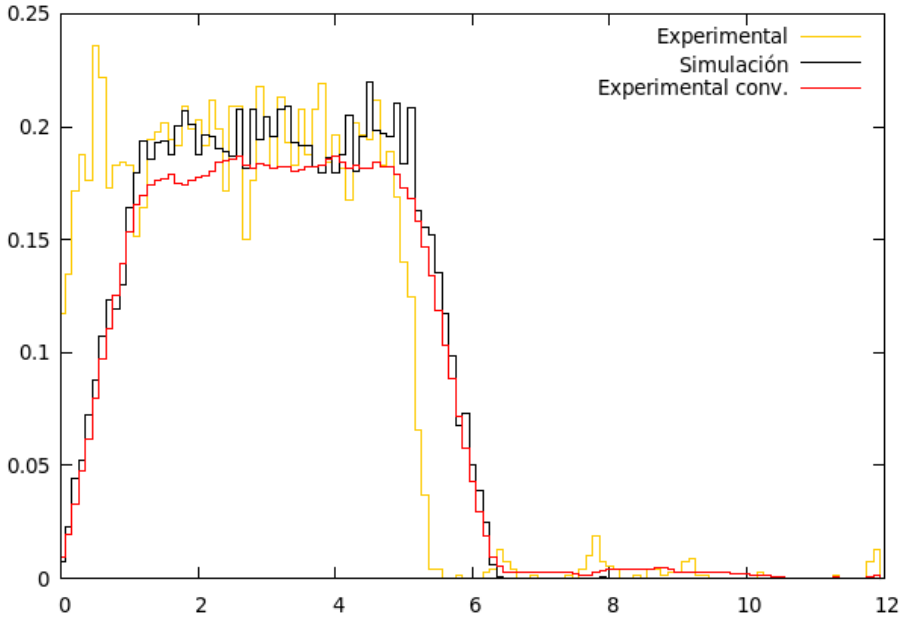

Figura 27: Función de densidad de probabilidad para un reparto 2:2

En la figura 27 se representa la función de densidad de probabilidad por intervalos correspondiente a un reparto 2:2. Con un trazo negro se representa la curva obtenida mediante simulación mientras que con un trazo amarillo se representa la curva obtenida de los datos experimentales. Se observa que la curva de datos experimentales es más abrupta en la subida y en la caída, pareciéndose más a un rectángulo. Se puede observar, que el final de la caída de la curva ocurre aproximadamente 1,28 segundos antes que en la curva obtenida por simulación. Esto se debe al hecho de que el dispositivo Bluetooth entra en el subestado inquiry scan justo tras la activación, algo que no siempre ocurre en las situaciones reales. Teniendo en cuenta que el dispositivo Bluetooth abre la ventana de inquiry scan durante 22,5ms cada 1,28 segundos, el instante de entrada en cobertura podría producirse mientras el dispositivo no tiene abierta la ventana, lo que alargaría ligeramente el tiempo de detección. Este tiempo adicional hasta que el dispositivo abre la ventana de inquiry scan puede caracterizarse como una variable aleatoria con distribución uniforme entre 0 y 1,28 segundos durante el cual el terminal no es descubierto por la baliza. Sin embargo, se han identificado dos casos en los que esta suposición puede dar lugar a un error:

1. Si el instante de activación del proceso de inquiry scan se produce en el intervalo de 1,28 segundos posterior a la finalización del proceso de

inquiry de la baliza y, además, se da la circunstancia de que el tiempo adicional fuera lo suficientemente grande como para que la ventana de inquiry scan pudiera haber estado abierta y se hubiera producido una detección previa al instante de detección real.

2. Si el instante de activación del proceso de inquiry scan se produce poco antes de la finalización de la fase de inquiry en la baliza (hasta 16 ranuras de tiempo antes) de modo que no se produzca la detección y, además, se da la circunstancia de que el tiempo adicional fuera lo suficientemente grande como para que la ventana de inquiry scan pudiera haber estado abierta y se hubiera producido una detección previa al instante de detección real.

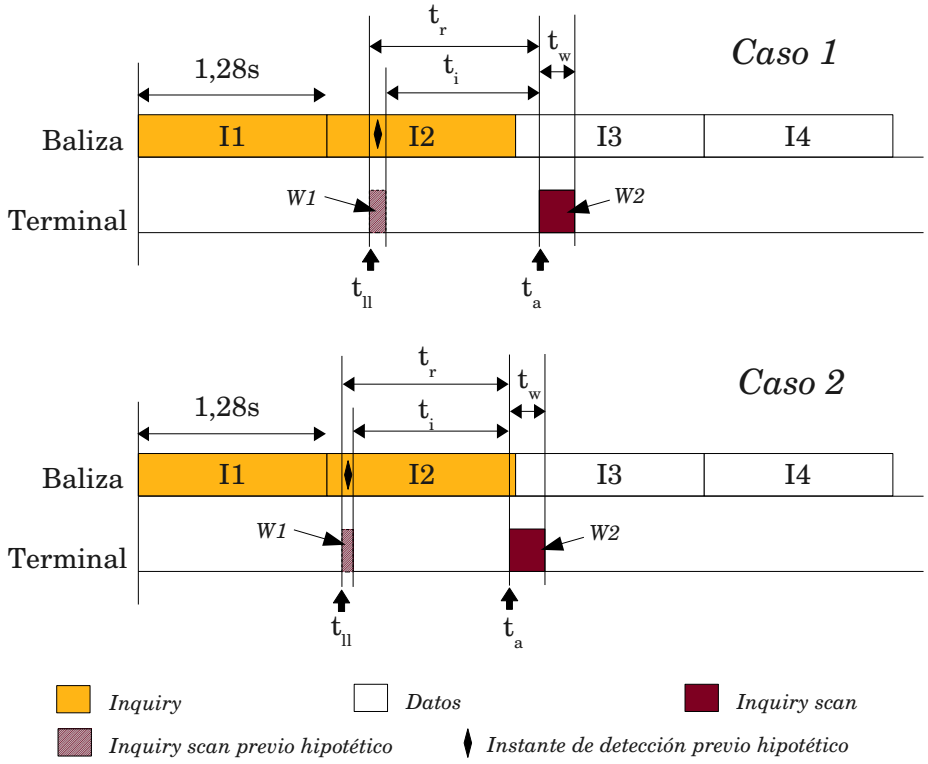

Figura 28: Casos especiales de entrada en cobertura para un reparto 2:2

Ambas situaciones se describen gráficamente en la figura 28 sobre un reparto 2:2, donde se asumen las siguientes definiciones:

- t_a: Instante de activación del proceso de inquiry scan en el receptor

- t_r: Tiempo adicional hasta que el receptor abre la ventana de inquiry scan
- t_w: Anchura de la ventana de inquiry scan
- t_{ll}: Tiempo estimado de entrada en cobertura
- t_i: Tiempo de inactividad de inquiry scan por bloque (equivale a 1,28-t_w)

Si se dan las situaciones expuestas en los casos 1 y 2, se habría producido una detección previa N·1,28 segundos antes que la detección real que se produce en el experimento, siendo N el número de bloques de 1,28 segundos que abarca el ciclo de inquiry utilizado. Afortunadamente, como se demuestra a continuación, la probabilidad de que esto ocurra es despreciable. Concretamente, se puede acotar esta probabilidad de error como:

$$P_e = P_{e\,CASO1} + P_{e\,CASO2} \quad (1)$$

Para el caso 1 se definen los siguientes sucesos:

$$\begin{gathered} A1 : (t_r > t_i) \cap (t_a \in I3) \\ A2 : Detección\ en\ W1 \\ A3 : t_{ll} \in I2 \end{gathered} \quad (2)$$

Tanto t_a como t_r pueden caracterizarse como variables aleatorias con distribución uniforme en los intervalos (0 , N·1,28) y (0 , 1,28), respectivamente. Utilizando las funciones de densidad de probalibidad correspondientes se obtiene que:

$$P(t_r > t_i) = \frac{t_w}{1{,}28}\ ;\ P(t_a \in I3) = \frac{1}{N} \quad (3)$$

Por la definición del suceso A1, y teniendo en cuenta (3) se obtiene:

$$P(A1) = \frac{t_w}{(1{,}28 \cdot N)} \quad (4)$$

Por otra parte, aplicando probabilidad condicional, y definiendo el suceso A13 como la ocurrencia simultánea de los sucesos A1 y A3, se puede expresar que:

$$P(A13) = P(A1 \cap A3) = P(A1) \cdot P(A3 | A1) \quad (5)$$

donde

$$P(A3|A1)=P((t_{ll}\in I2)|A1) \qquad (6)$$
$$t_{ll}=t_a-t_r$$

Teniendo en cuenta lo anterior, se recalculan las funciones de densidad de probabilidad para t_a y t_r condicionadas a la ocurrencia del suceso A1, obteniéndose las gráficas mostradas en la figura 29.

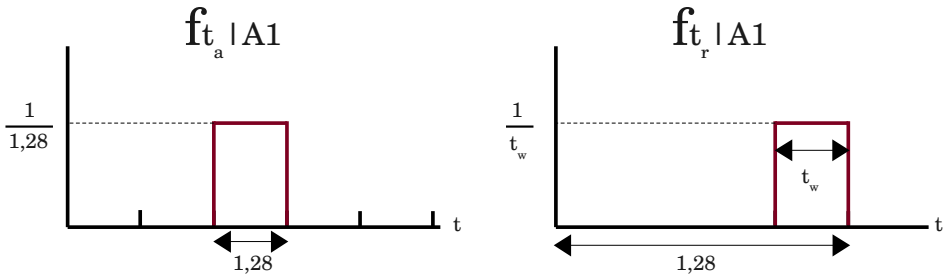

Figura 29: Funciones de densidad de probabilidad condicionada para t_a y t_r

Considerando la definición de t_{ll} y que t_a y t_r son independientes, la función de densidad de probabilidad de t_{ll} condicionada al suceso A1 se calcula como la convolución de las funciones de t_a y $-t_r$, cuyo resultado se muestra en la figura 30.

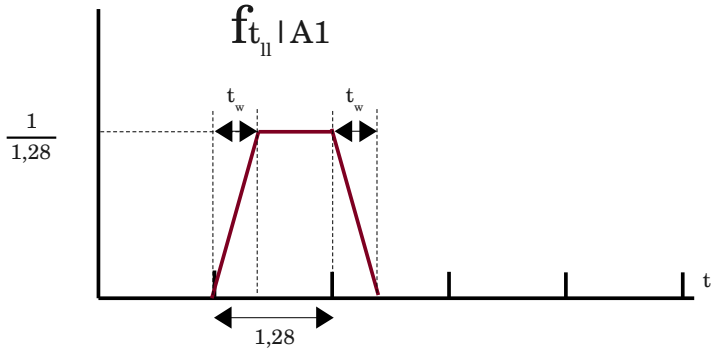

Figura 30: Funciones de densidad de probabilidad condicionada para t_{ll}

Integrando sobre la función anterior se obtiene:

$$P(A3|A1)=P((t_{ll}\in I2)|A1)=1-\frac{t_w}{(2\cdot 1{,}28)} \quad (7)$$

Sustituyendo (7) en (5) se obtiene la probabilidad del suceso conjunto A13:

$$P(A13)=\frac{\left(1-\frac{t_w}{2{,}56}\right)\cdot t_w}{(1{,}28\cdot N)} \quad (8)$$

Una vez obtenida esta probabilidad, se puede incorporar el suceso A2 aplicando de nuevo probabilidad condicional. De esta forma, se obtiene la probabilidad de error en el caso 1:

$$P_{eCASO1}=P(A2\cap A13)=P(A13)\cdot P(A2|A13) \quad (9)$$

Para calcular el último factor de la expresión (9) hay que tener en cuenta lo siguiente: Si se cumplen las condiciones del suceso A13, la probabilidad de ser detectado en la ventana de inquiry scan previa es del 100% salvo que el tiempo que transcurre entre la entrada en cobertura y el final de la ventana de inquiry scan o el final de la fase de inquiry, lo que antes suceda, sea demasiado pequeño (menor que 16 ranuras de tiempo). Por tanto, el segundo factor de la expresión (9) es un valor cercano a 1. Con objeto de acotar el error, se puede escribir que:

$$P_{eCASO1}\leq P(A13) \quad (10)$$

Respecto al caso 2, deben darse tres condiciones:

$$\begin{gathered} B1: 2\cdot 1{,}28-0{,}010<t_a<2\cdot 1{,}28 \\ B2: Detección\ en\ W1 \\ B3: t_r>t_i \end{gathered} \quad (11)$$

Integrando sobre las funciones de densidad de probabilidad de t_a y t_r se obtienen las probabilidades de los sucesos B1 y B3:

$$P(B1)=0.010/(1{,}28\cdot N) \quad (12)$$

$$P(B3)=t_w/1{,}28 \quad (13)$$

Al igual que en el caso 1, la probabilidad de que ocurra el suceso B2 no es sencilla de calcular pero se puede acotar el error para el caso 2 escribiendo:

$$P_{eCASO2} \leqslant P(B1) \cdot P(B3) \quad (14)$$

Sustituyendo los valores de t_w y N (0,0225 y 4, respectivamente) en las expresiones (10) y (14) se obtienen las probabilidades de error máximas en cada caso. Según (1), al sumarlas se obtiene una cota máxima para el error:

$$Pe \leqslant 0{,}004355 + 0{,}000002 = 0{,}004357 \quad (15)$$

Es decir, que la probabilidad de error es inferior al 0,43%. Por tanto, se puede asumir que el tiempo adicional se caracteriza como una variable aleatoria independiente con distribución uniforme entre 0 y 1,28 segundos. Este tiempo, junto con el resto del tiempo de detección, que está caracterizado experimentalmente por la curva amarilla de la figura 27, permite calcular el tiempo total de detección como la suma de dos variables aleatorias independientes. En esta situación, la función de densidad de probabilidad resultante equivale a la convolución de las funciones de densidad de probabilidad de las variables de partida. El resultado de la convolución se representa por el trazo rojo de la figura 27, que sí guarda un mayor parecido con la curva obtenida en simulación.

El experimento se ha repetido con otros repartos y se ha observado que también coinciden, por lo que se corroboran los resultados obtenidos en simulación.

Throughput residual durante el inquiry

Con frecuencia se da la circunstancia de que existen enlaces de datos activos entre la baliza y los terminales mientras se está realizando el proceso de inquiry. En esta situación, el controlador de la banda base del dispositivo Bluetooth de la baliza permite la transmisión y recepción de paquetes ACL aunque a un ritmo bastante menor que en los periodos en los que no se realiza inquiry. El hecho de que algunas ranuras sean ocupadas por paquetes ACL en lugar de los paquetes de identificación puede perturbar el descubrimiento de dispositivos, siendo esta perturbación más perniciosa cuanto mayor es la tasa de datos. Esta tasa de datos residual varía en función del modelo y fabricante del dispositivo bluetooth ya que va implícita en la política de gestión del ancho de banda interna del dispositivo. Se ha observado que durante el inquiry, la tasa de datos de un enlace puede bajar hasta el 10%, aunque en otros casos

puede ser mayor. La intervención en esta política por parte de las capas superiores es muy limitada ya que únicamente se podría recurrir a la especificación de parámetros de calidad de servicio (QoS). En cualquier caso, la intervención únicamente actuaría en el sentido de garantizar cierto ancho de banda para el enlace de datos, lo cual puede ser incluso más perjudicial para el descubrimiento de dispositivos.

Debido a la variabilidad de comportamiento entre distintos dispositivos Bluetooth y a la complejidad que implica introducir un nuevo factor aleatorio como es la inserción de paquetes ACL, parece más razonable abordar el estudio del problema mediante simulación. Para ello se ha utilizado el simulador descrito anteriormente con las modificaciones pertinentes para que introduzca un determinado número de paquetes de datos durante la fase de inquiry. Este parámetro se ha expresado en términos de tanto por ciento respecto al número total de ranuras. De esta manera se ha evaluado el efecto que dicho throughput residual tiene sobre la función de densidad de probabilidad del tiempo de detección. Para apreciar mejor los efectos se ha hecho una comparación entre dos esquemas de reparto parecidos: el reparto 1:2 y el 2:1, que poseen curvas muy similares para la densidad de probabilidad. En la figura 31 se muestra el efecto de diferentes tasas residuales sobre el reparto 1:2 mientras que en la figura 32 se muestra el mismo efecto sobre el reparto 2:1.

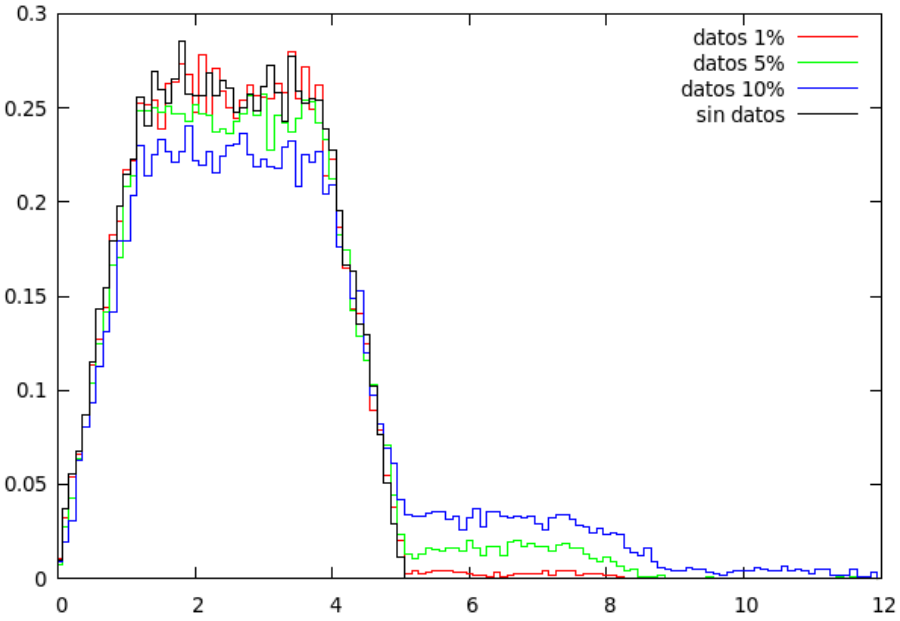

Figura 31: Densidad de probabilidad con un reparto 1:2 y throughput residual

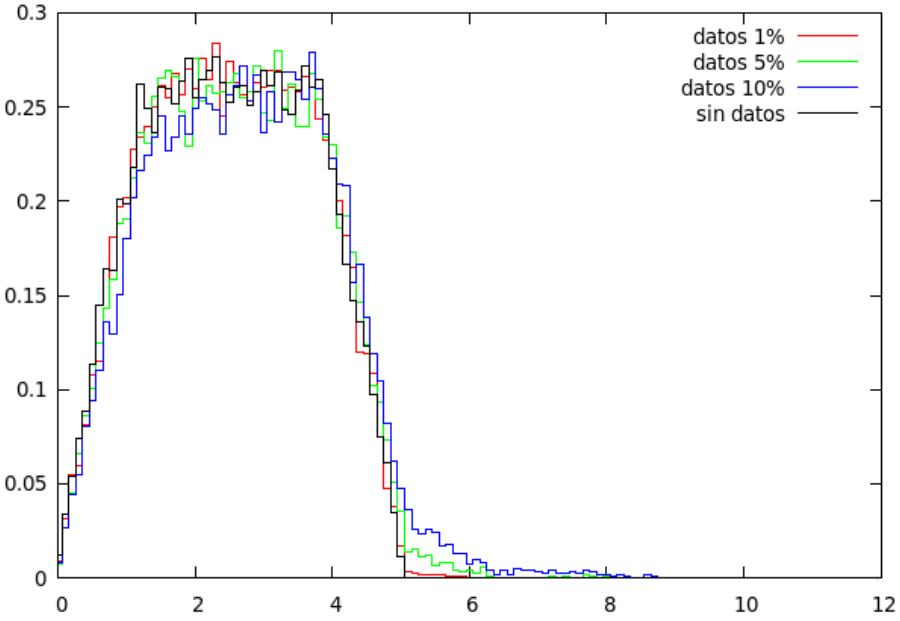

Figura 32: Densidad de probabilidad con un reparto 2:1 y throughput residual

A través de las figuras se observa que, a mayor tasa de datos más se expande la función de densidad de probabilidad. Pero hay otro fenómeno a destacar, y es que el efecto es más notable en el reparto 1:2 que en el 2:1. Efectivamente, el efecto del throughput residual es bastante más acusado en aquellos repartos que sólo dedican un bloque de 1,28 segundos al inquiry, por lo que los repartos que dedican más de un bloque al inquiry son más robustos. Esto se observa más claramente en la tabla 16, donde se muestra la media y la desviación típica de los resultados obtenidos.

	Tasa de datos residual			
Reparto	**0%**	**1%**	**5%**	**10%**
1:2	μ=2,5564 *σ=1,1741*	μ=2,5992 *σ=1,2303*	μ=2,7903 *σ=1,4781*	μ=3,1663 *σ=1,9636*
2:1	μ=2,5593 *σ=1,1701*	μ=2,5839 *σ=1,1655*	μ=2,6414 *σ=1,2143*	μ=2,7751 *σ=1,2947*

Tabla 16: Efecto del throughput residual en los repartos 1:2 y 2:1

Los errores en el canal RF, debidos a ruido, multitrayecto, obstáculos, etc., son difíciles de caracterizar y además varían en cada ubicación, pero su efecto sobre el tiempo de descubrimiento puede ser, en cierto modo, similar al del throughput residual. De nuevo, en ambientes donde la tasa de errores sea elevada, los repartos que dedican más de un bloque de 1,28 segundos al inquiry se muestran más robustos.

Desde el punto de vista del througput global en las conexiones, la existencia del throughput residual contribuye a su mejora en una pequeña cantidad. Sin embargo, esta leve mejora no compensa el aumento del tiempo de detección, máxime cuando existen varias conexiones entre la baliza y los terminales, ya que en este caso, la mejora del throughput global puede ser incluso menor mientras que el efecto sobre el tiempo de detección se agrava.

Soluciones al problema del throughput residual

Afortunadamente, los modos hold y sniff, que son aplicables al estado de conexión, pueden resolver el problema del throughput residual. Si una conexión se pone en modo hold, ésta queda interrumpida por un periodo de tiempo determinado y no se produce ninguna transmisión ni recepción de datos durante ese periodo. Un detalle que hay que tener en cuenta a la hora

de utilizar el modo hold es que el tiempo de supervisión del enlace debe ser mayor que el periodo de hold ya que, de lo contrario, los dispositivos darían por rota la comunicación antes de que terminara el periodo de hold. La estrategia a seguir para solucionar el problema haciendo uso del modo hold consiste en pasar a modo hold todas las conexiones justo antes de comenzar el inquiry y por un periodo igual al número de bloques de 1,28 segundos que se dediquen al inquiry. Si los tiempos de supervisión de los enlaces son menores, habrá que modificarlos, al menos mientras dure el inquiry, y restaurarlos después.

El modo sniff, a diferencia del anterior, sí permite el intercambio de paquetes ACL durante el inquiry, pero puede restringirlo fuertemente, de modo que intercambien un mínimo de información que evite que venza el temporizador de supervisión del enlace. Por tanto, esta solución no implica una modificación de los tiempos de supervisión de los enlaces. Como contrapartida está el hecho de que queda un throughput residual, aunque éste es muy reducido. La estrategia a seguir ahora consiste en poner todas las conexiones en modo sniff justo antes de comenzar el inquiry y sacarlas de dicho modo cuando termine.

Para la implementación del sistema presentado en esta Tesis se ha optado por utilizar el modo hold, ya que así se garantiza que no hay interferencias en el proceso de inquiry y se facilita una rápida detección de los dispositivos. Las operaciones relacionadas con el modo hold se llevan a cabo en ABCM dentro de la clase nccclient mediante los métodos públicos nccclient::ncc_hold_all() y nccclient::ncc_restore_all_lst(). El primero de los métodos se invoca antes del inquiry, pasa todas las conexiones activas a modo hold y modifica el tiempo de supervisión del enlace si es necesario. El segundo método se invoca después del inquiry y restablece los tiempos de supervisión si fuera necesario. Se deja a la responsabilidad del objeto de política de red el uso apropiado de estos métodos.

Inquiry cycle adaptativo

Como se ha comentado anteriormente, el uso de repartos con alta dedicación al inquiry aumenta la probabilidad de descubrimiento en un ciclo de inquiry a costa de reducir el throughput medio. Si bien, esto último puede ser considerado una desventaja, en ciertas situaciones puede no serlo. Cuando una baliza no tiene ninguna conexión activa, resulta poco eficiente priorizar el tiempo de transmisión de datos frente al de inquiry. En estos casos convendría cambiar el reparto por otro que favoreciera el inquiry sólo mientras no

hubiera conexiones activas. De esta forma, se reduciría el tiempo de detección al menos para los primeros terminales, lo cual resulta bastante interesante sobre todo si la ocupación media de la baliza es baja.

La gestión adaptativa del inquiry cycle es responsabilidad del objeto de política de red y puede llevarse a cabo mediante varios métodos en los que intervienen además las clases nccclient y la clase principal abdmd. En un primer momento, tras la creación de una instancia del objeto de política de red, se le indica a éste el reparto utilizado mediante el método abnpolicy::set_inqcycle(). Durante el funcionamiento normal de ABCM, si el objeto de política de red determina, a través de la lista de conexiones, que se debe cambiar el reparto, puede modificar sus variables locales y, mediante la referencia al objeto nccclient, cambiar el tiempo de inquiry. Dado que la temporización es responsabilidad de la clase abcmd, ésta puede consultar cambios en el reparto al finalizar cada inquiry cycle invocando el método abnpolicy::get_suggested_inqcycle(), que proporciona información actualizada sobre el reparto que sugiere el objeto de política de red. En función de la información obtenida, la clase abcmd puede cambiar la temporización.

Selección del inquiry cycle

Una correcta selección del inquiry cycle ayuda a minimizar el tiempo de servicio y, en particular, el tiempo de detección. Dependiendo de la situación particular puede interesar más optar por un reparto u otro. Así, por ejemplo, en entornos urbanos donde los terminales puedan estar en vehículos, sería más conveniente optar por repartos cortos como el 1:1 para poder ofrecer los servicios en el menor tiempo posible, ya que los vehículos pueden permanecer poco tiempo en la zona de cobertura. En estos casos, la reducción del 50% del throughput medio, tal y como se muestra en la tabla 14 puede no ser un inconveniente si el tráfico de datos del servicio no es grande.

En otras situaciones donde los terminales se muevan a velocidad inferior y la prioridad sea detectar el máximo número de terminales al alcance, se puede optar por otros repartos que prioricen el inquiry frente a los datos. En los casos en los que el entorno sea ruidoso, en términos electromagnéticos, o en los que la cobertura sea irregular, también será conveniente optar por repartos con más peso de inquiry o que al menos dediquen más de un bloque de 1,28 segundos al inquiry, ya que, como se ha visto anteriormente, estos repartos resultan ser más robustos ante los errores. Un ejemplo de aplicación donde este esquema sería útil es en los servicios de localización y guiado de personas en interiores.

En los casos en los que el servicio requiere un intercambio de datos grande y los terminales se mueven a poca velocidad, resulta más interesante emplear repartos que favorezcan la transmisión de datos frente al inquiry a costa de aumentar el tiempo de detección.

Coenlaces

Ante el problema del tiempo de detección, en la tecnología Bluetooth se han planetado soluciones como las descritas en [74] y [75], que se basan en el uso de un coenlace basado en otra tecnología inalámbrica. La utilidad de un coenlace en una red para entornos urbanos radica en la posibilidad de dar a conocer la existencia de una baliza a los terminales que estén en las proximidades de una forma más rápida, enviando la información precisa para que se pueda establecer la conexión a través de otras redes como Bluetooth. En el caso particular de Bluetooth esto no sólo ahorra gran parte del tiempo de detección a través del inquiry, sino que además se puede aprovechar todo el tiempo para la comunicación de datos en la piconet evitando los recortes del throughput medio indicados en la tabla 14. En los artículos anteriormente referenciados se cita concretamente a la tecnología IrDA (infrarrojos) como solución para la detección por su rapidez para establecer conexiones. El gran inconveniente de la tecnología IrDA es que debe haber visibilidad directa entre la baliza y el receptor y, además, este último debe apuntar a la baliza, ya que el ángulo de emisión es limitado. Otra de las tecnologías que se citan es RFID, que también permite establecer conexiones con rapidez, pero su principal inconveniente es su corto rango de alcance (en el caso de etiquetas RFID pasivas es del orden de centímetros). Otra alternativa, parecida en cierto modo a la anterior, es utilizar un radioenlace en alguna de las bandas libres (433/868MHz, por ejemplo). Esta tecnología presenta un mayor alcance que las etiquetas RFID. En cualquier caso, el uso de coenlaces complica y encarece el diseño, pero, en determinadas situaciones donde el tiempo de detección sea crítico, puede ser útil.

El sistema presentado en esta Tesis contempla en su diseño la posibilidad de sustituir ciertos módulos para poder dar soporte a estos coenlaces.

Modelo de comunicación

A la vista de las restricciones aplicables a las redes de dispositivos en entornos urbanos inteligentes, expuestas al principio del capítulo, se llega a la

conclusión de que, si bien Wireless CORBA se perfila como la solución que mejor encaja en este tipo de aplicaciones, no puede utilizarse directamente sin acometer importantes modificaciones tanto a alto como a bajo nivel. La única implementación conocida de Wireless CORBA en el momento de realizar este trabajo de investigación es la de MICO. Las modificaciones que se describen a continuación se aplican concretamente sobre la versión 2.3.13 de MICO y se ha procurado realizar las modificaciones a alto nivel para alterar el núcleo del ORB lo menos posible.

La capa LTP

Las modificaciones que llevaron a la incorporación de Wireless CORBA en MICO, al igual que el sistema propuesto en esta Tesis, supusieron cambios a varios niveles: algunas situaciones se resolvieron en forma de servicios CORBA y otras directamente sobre el núcleo de MICO. Uno de los cambios más evidentes fue la inclusión de una nueva capa de transporte sobre Bluetooth, que deriva del proyecto Vivian [67] y se denomina MICOLTP::LTPTransport. Esta capa tiene una estructura muy parecida a la de otras capas de transporte más habitulales. De hecho, la clase que implementa la capa hereda de la clase genérica CORBA::Transport. Su misión es establecer un túnel usando una conexión L2CAP e intercambiar a través de él los mensajes del protocolo GIOP. La importancia de esta clase radica en que es la que maneja la conexión L2CAP a bajo nivel, manteniendo el descriptor del socket y ofreciendo a las capas superiores las funciones básicas de lectura/ escritura de datos y establecimiento/liberación de la conexión. En la figura 33 se muestran las capas y protocolos aplicables a las conexiones CORBA ordinarias a través de redes TCP/IP (a) y a través de Bluetooth (b).

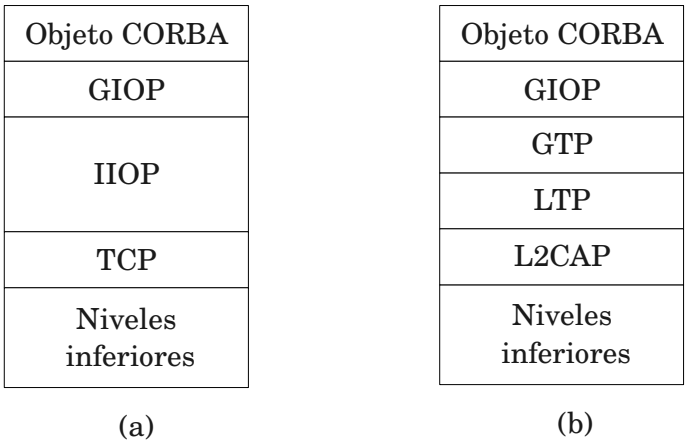

Figura 33: Torre de protocolos de CORBA y Wireless CORBA

En la capa de transporte, junto a MICOLTP::LTPTransport, se tiene una serie de clases necesarias entre las cuales se destacan las siguientes:

- MICOLTP::LPTAddress. Esta clase almacena las direcciones propias del protocolo LTP y deriva de la clase genérica CORBA::Address, por lo que su manejo por parte de las capas superiores es similar al de una dirección TCP/IP ordinaria.
- MICOLTP::LTPAddressParser. Esta clase deriva de CORBA::AddressParser e interpreta direcciones LTP.
- MICOLTP::LTPTransportServer. Esta clase deriva de CORBA::TransportServer y es utilizada fundamentalmente en el extremo que actúa como servidor atendiendo conexiones LTP entrantes.

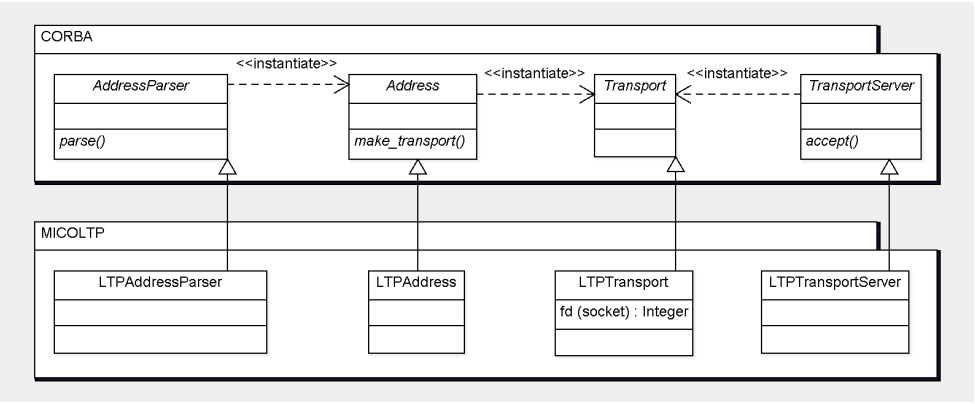

Figura 34: Clases involucradas en la capa de transporte LTP

Para describir el funcionamiento básico de estas clases hay que distinguir dos casos: cuando se actúa como servidor y cuando se actúa como cliente.

En el caso de actuar como servidor, existe una instancia de la clase MICOLTP::LTPTransportServer que es la encargada de realizar el ciclo básico de un servidor, que consiste en los siguientes pasos:

1. Asociar un socket a un puerto (bind). En el caso particular de las conexiones L2CAP, en lugar de puerto, se recurre al concepto de Multiplexor de Servicios y Protocolos, o PSM, como se le conoce por sus siglas en inglés. En cualquier caso, el funcionamiento del PSM es análogo al de los puertos TCP/UDP.
2. Esperar a que lleguen conexiones entrantes al socket (listen).
3. Aceptar las conexiones entrantes (accept). En este punto se crea un nuevo socket para gestionar la comunicación entrante. Tras haber aceptado la conexión, desde la clase MICOLTP::LTPTransportServer se crea una nueva instancia MICOLTP::LTPTransport que, a partir de ese momento, gestionará la comunicación a través del nuevo socket y servirá de transporte a los mensajes CORBA que se intercambien entre el cliente y el servidor. Por su parte, la instancia de MICOLTP::LTPTransportServer retorna al punto 2.

En el caso del funcionamiento como cliente, el procedimiento es más sencillo ya que basta con crear una nueva instancia de la clase MICOLTP::LTPTransport e invocar el método connect.

En el transcurso de los procesos descritos anteriormente intervienen las clases MICOLTP::LTPAddressParser y MICOLTP::LTPAddress. La primera, a través de su método parse(), interpreta la dirección pasada en forma de cadena de caracteres y si concuerda en formato con las direcciones LTP devuelve una instancia a un objeto MICOLTP::LTPAddress que almacenará la dirección desglosada internamente. Pero ésta no es la única finalidad de la clase MICOLTP::LTPAddress ya que ésta, a través de sus métodos make_transport() y make_transport_server(), es capaz de crear instancias de MICOLTP::LTPTransport y MICOLTP::LTPTransportServer, respectivamente. Por tanto, el ORB obtiene una instancia de MICOLTP::LTPAddress a través de la cual se crean a su vez las instancias que gestionan el transporte.

Una vez creadas, las instancias de MICOLTP::LTPTransport realizan

las operaciones de lectura y escritura, asbtrayendo a las capas superiores de los detalles particulares del manejo de la conexión Bluetooth. Para ello hacen uso de buffers y funciones de tipo "callback" externas a las que llaman para notificar un evento como, por ejemplo, la disponibilidad de datos para su lectura o la notificación de la escritura.

ELTP como solución a las limitaciones de la capa LTP

Si bien la capa LTP constituye una pieza clave en el funcionamiento de Wireless CORBA, en la práctica la implementación de esta capa muestra dos limitaciones importantes cuyo efecto es bastante pernicioso en el tipo de redes que se proponen en esta Tesis:

- Se ha observado que las clases de transporte sobre LTP no resuelven bien la pérdida de cobertura. En estos casos, los fallos en recepción son tratados como errores no dañinos y no se notifican a las capas superiores, lo que desemboca en reintentos continuos de las operaciones de lectura sobre una conexión que realmente se ha roto. Cuando esto ocurre, los puentes (AccessBridge/TerminalBridge) entran en una dinámica iterativa dejando de funcionar correctamente y disparando tanto la carga de la CPU como el consumo de la misma.
- La otra limitación está también ligada a la pérdida de cobertura y viene determinada por el tiempo que transcurre desde que realmente se rompe la conexión hasta que los sockets lo detectan. Por defecto, este tiempo es de 20 segundos, un tiempo demasiado grande para este tipo de entornos donde un terminal puede cambiar de una piconet a otra en cuestión de segundos.

Para solucionar estas limitaciones se propone la creación de la capa ELTP (Enhanced LTP). Para ello se ha partido de las propias clases de LTP creando una estructura paralela que incorpora las modificaciones necesarias no sólo para solucionar los inconvenientes mencionados anteriormente sino también para aportar nuevas funcionalidades.

En primer lugar, ELTP realiza un control más estricto de los mensajes de error que devuelven las operaciones sobre sockets L2CAP mediante mecanismos acumulativos de errores calificados como "no dañinos" y prestando especial atención al tipo de error ETIMEDOUT, que es utilizado para informar al socket del corte inesperado de la conexión. De esta forma, las capas superiores son informadas y dan por terminado el enlace en las situaciones de pérdida de cobertura, evitando la sucesión interminable de

reintentos. Si se necesita reestablecer la conexión, siempre se pueden realizar las peticiones oportunas a instancias superiores, como se verá más adelante.

En cuanto al tiempo de detección de interrupción de la conexión se ha averiguado que existe un parámetro en la capa HCI de la pila de protocolos de Bluetooth que controla precisamente este tiempo. Este parámetro se denomina LST (Link Supervision Timeout) y su ámbito de aplicación se circunscribe a cada una de las conexiones Bluetooth establecidas. Por este motivo, es necesario configurar el parámetro cada vez que se crea una nueva conexión. Teniendo en cuenta que el parámetro se configura por número de ranuras, y que cada ranura en Bluetooth tiene una duración de 625ms, se puede calcular a partir de este dato el valor correspondiente al tiempo en segundos que se requiere para el enlace. Para facilitar la configuración de este parámetro y permitir rebajar el tiempo de supervisión hasta límites aceptables se ha añadido el método público setlst() a la clase MICOLTP::ELTPTransport. De hecho, cada vez que la clase MICOLTP::ELTPTransportServer acepta una nueva conexión, crea una instacia de tipo MICOLTP::ELTPTransport y configura este parámetro.

La modificación del parámetro LST requiere permisos de superusuario y sólo puede hacerse desde el extremo que actúa como maestro de la comunicación Bluetooth. Como se verá más adelante, por razones de eficiencia, es el extremo que ejecuta el AccessBridge el que actúa como maestro. Por tanto, parece razonable que la invocación de la operación de cambio del LST tenga lugar en la clase MICOLTP::ELTPTransportServer, que es utilizada en AccessBridge para atender las peticiones de conexión. Adicionalmente, al tratarse de una operación a bajo nivel, se requiere una serie de parámetros tales como descriptores de la conexión HCI y manejadores (handlers) antes de poder hacer uso de la función de la capa HCI que lleva a cabo la operación (hci_write_link_supervision_timeout()). Dado que dichos descriptores están relacionados con cada conexión, el lugar más idóneo para obtenerlos es precisamente en la clase MICOLTP::ELTPTransport. A este efecto, se han incluido miembros privados para guardar estos descriptores y métodos para, de esta forma, obtenerlos a través de la información contenida en la propia clase. El hecho de disponer de los descriptores facilita también la obtención de otro tipo de información de utilidad como el nivel de RSSI o el rol adoptado en la comunicación (maestro o esclavo), que proporcionan los nuevos métodos getrssi() y getside(), respectivamente.

Con objeto de no afectar a la actual capa LTP, se establece como elemento diferenciador la sintaxis de las direcciones, de manera que las capas superiores puedan distinguir correctamente si se debe usar LTP o ELTP,

creando instancias de la clase adecuada. Concretamente, las direcciones de LTP seguían el esquema:

```
bt-l2cap:<direccion_bluetooth>#<PSM>
```

Mientras que las nuevas direcciones de ELTP siguen el esquema:

```
eltp:<direccion_bluetooth>#<PSM>
```

Medidas de gestión activa de conexiones

Algunos ORBs implementan mecanismos de Gestión Activa de Conexiones o ACM, por sus siglas en inglés. El estándar de CORBA no define ninguna terminología para definir esta capacidad, por lo que no todos los ORBs implementan estos mecanismos y los que lo hacen, en ocasiones lo denominan de formas diferentes. Como ejemplo, Orbacus sí lo implementa y lo denomina como ACM, mientras que OmniORB o TAO lo implementan con otros nombres.

Básicamente, el uso de ACM permite el cierre de conexiones ociosas para hacer un uso más eficiente de los recursos de red. Las conexiones pueden ser posteriormente restablecidas si se necesita transmitir algo. Aunque esto puede parecer poco relevante para las transacciones CORBA que se realizan a través de una red de área local o Internet, resulta de vital importancia cuando el transporte se realiza sobre un enlace Bluetooth. Teniendo en cuenta que en una piconet sólo puede haber hasta un máximo de 7 conexiones abiertas de forma simultánea, y que los puentes (AccessBridge/TerminalBridge) mantienen las conexiones abiertas por tiempo indefinido independientemente de su actividad, resulta imprescindible adoptar mecanismos de ACM que eviten situaciones en las que los terminales ociosos impidan el acceso a los servicios a los nuevos terminales que entren en el área de cobertura de una baliza por haber saturado el número de conexiones posibles. Desafortunadamente, MICO no implementa estos mecanismos.

Para solucionar este problema se ha añadido un mecanismo sencillo de ACM a MICO, tomando las debidas precauciones dado que las modificaciones se han realizado sobre el núcleo del ORB y había que asegurarse de que el cierre de la conexión se produjera sólo si el enlace estaba ocioso.

Como paso previo a la implantación del mecanismo se identificaron los puntos del código en los cuales las transacciones CORBA se dan por

terminadas. Esto ocurre en determinados puntos de los métodos CORBA::StaticRequest::invoke() y CORBA::Request::get_response(), aplicables a las invocaciones estáticas y dinámicas, respectivamente. En esos puntos se insertaron llamadas a un nuevo método del ORB creado expresamente para esta función que se denomina CORBA::ORB::ACM_EOT() y es invocado cuando finaliza una transacción. Éste, a su vez, tiene al alcance la referencia al objeto MICO::IIOPProxy, al que también se le ha añadido un método nuevo denominado MICO::IIOPProxy::force_close(), que fuerza el cierre de la conexión. Finalmente, esta instancia, que dispone de las referencias a todas las conexiones lógicas abiertas, puede llamar al método kill_conn() que acaba consumando el cierre de la conexión GIOP tras haber efectuado la transacción.

Este comportamiento debe activarse en los clientes CORBA que residen en el terminal. A tal efecto se ha creado una nueva opción del ORB que se añade como parámetro al lanzar el cliente CORBA y cuya sintaxis es la siguiente:

```
-ORBCloseAfterTrans <redirected | all>
```

La opción acepta dos argumentos posibles: redirected y all. El primero restringe el comportamiento descrito únicamente a aquellas conexiones que estén siendo redirigidas hacia el puente (TerminalBridge), mientras que el argumento "all" hace que se aplique el comportamiento a todas las conexiones.

El cierre de las conexiones GIOP no implica el cierre de la conexión Bluetooth que constituye el puente, pero permite mantener actualizado un registro con el estado de todas las conexiones. Este registro se almacena en los puentes, concretamente en el miembro protegido WATMBridge::GIOPBridge::_tstamps que, adicionalmente, incluye marcas de tiempo con el último instante en que la conexión estuvo activa. Esta lista de conexiones es manejada por dos métodos nuevos: WATMBridge::GIOPBridge::conn_ts_busy() y WATMBridge::GIOPBridge:: conn_ts_free(), que marcan la conexión como ocupada u ociosa, respectivamente. Las llamadas a estos métodos están situadas dentro de la clase en los puntos donde se inician y terminan las transacciones que son canalizadas por el puente. En este sentido, fueron de utilidad las funciones ref() y deref() de la propia clase WATMBridge::GIOPBridge, que suelen determinar estos eventos, por lo que en muchos casos los nuevos métodos acompañan a dichas funciones. La figura 35 ilustra la relación entre los métodos y las clases implicadas.

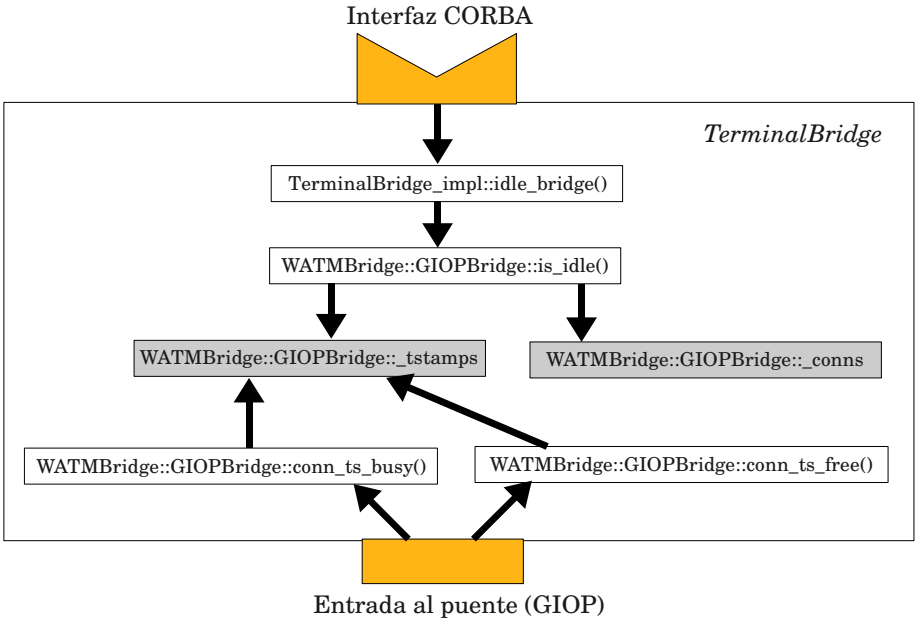

Figura 35: Métodos y clases implicados en el mecanismo ACM

Puesto que son los puentes los que llevan este registro de las conexiones, se ha hecho accesible esta información desde el exterior mediante la adición de un nuevo método, idle_bridge, a la clase TerminalBridge. Este método, disponible en el nuevo IDL, accede a dicha información y permite determinar si existen o no conexiones activas en curso a través del puente. Para llevar a cabo esta tarea, el método, implementado en la clase TerminalBridge_impl recurre a su vez al método WATMBridge::GIOPBridge::is_idle(). Es este último el que consulta directamente al miembro _tstamps que, como se comentó anteriormente, es quien guarda la información. El método diseñado es responsable de indicar si hay o no alguna conexión en uso cuyo tiempo de inactividad no supere el umbral establecido en el parámetro que se le pasa.

Al añadir este método a la interfaz CORBA de TerminalBridge, la información de la actividad de las conexiones puede ser accesible a otros objetos y procesos que, en función de la misma, desencadenen una acción como, por ejemplo, cerrar la conexión Bluetooth. De hecho, otros procesos que forman parte del sistema completo, como se verá más adelante, adoptarán este tipo de decisiones para evitar el problema del bloqueo de conexiones por parte de terminales ociosos.

Modelo de gestión de las conexiones

Como se indicó al principio del capítulo, Wireless CORBA no puede ser aplicado directamente en entornos urbanos debido, entre otras cosas, a que carece de elementos que gestionen de una manera efectiva y dinámica las conexiones a las distintas redes que un terminal puede encontrar a su alcance. Para solucionar el problema se propone una arquitectura basada en gestores de la conexión tanto en el lado del terminal como en el lado de la infraestructura, tratando siempre de minimizar el impacto sobre los extremos finales de los servicios, esto es, los servidores y los clientes. Mediante este modelo, se pueden construir redes de servicios donde exista un conjunto de balizas que se encarguen de proporcionar el acceso a los terminales que visiten la red. La figura 36 ilustra la topología de estas redes.

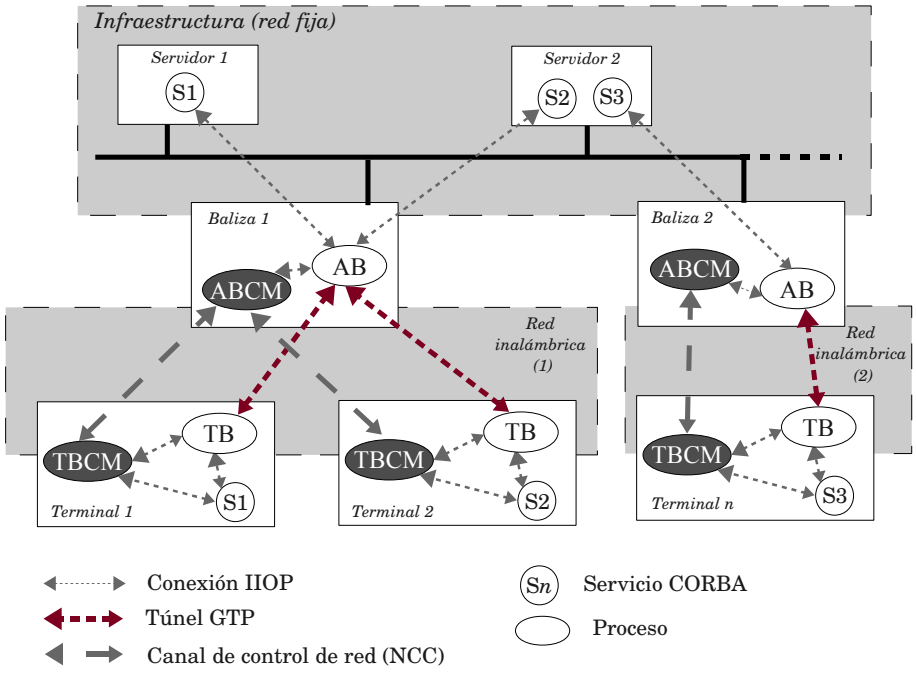

Figura 36: Topología de red para entornos urbanos inteligentes

Desde el punto de vista de la infraestructura fija, los servicios pueden estar alojados en cualquier máquina que esté conectada a la red, aunque estos servicios también pueden alojarse en las propias balizas. Esto es especialmente útil cuando un servicio depende fuertemente de la jerarquía de

contextos o de la ubicación*.

En un entorno urbano, los terminales que aparecen son "nómadas" y no conocen a priori ni la estructura interna ni los servicios que oferta una red. En estos casos, el concepto de "pertenencia" de un terminal a una red carece de sentido y, por tanto, no es necesaria la figura del Home Location Agent (HLA) que se desarrolla en el planteamiento original de Wireless CORBA. Si bien el HLA interviene en los casos en los que los servidores se ubican en los terminales y los clientes en la infraestructura, no se han identificado servicios para entornos urbanos inteligentes que requieran este escenario, por lo que en el modelo presentado no se considerará el uso del HLA.

Los gestores de las conexiones constituyen una de las piezas clave para la creación de este tipo de redes. Estos gestores de la conexión han sido implementados en forma de procesos que hacen uso de interfaces CORBA tanto para ofrecer información adicional como para interactuar con otros elementos como, por ejemplo, los puentes. En el lado de las balizas se implanta el denominado Access Bridge Connection Manager (ABCM), mientras que en los terminales se implanta su homólogo, el Terminal Bridge Connection Manager (TBCM). Entre los terminales y la infraestructura se establecen dos canales que hacen uso de la conexión inalámbrica. El primero de ellos es el propio enlace que se establece entre AccessBridge y TerminalBridge, por el que circulan las peticiones CORBA. Como ya se comentó anteriormente, este enlace es de tipo L2CAP. El otro enlace se ha creado específicamente para la comunicación de los gestores y también utiliza el protocolo L2CAP pero, a diferencia del caso anterior, sobre este protocolo se monta un nuevo protocolo específico al que se ha denominado Network Control Channel (NCC). Internamente, la interacción entre los distintos procesos se realiza fundamentalmente mediante interfaces CORBA de modo que se dota de una mayor independencia y modularidad al diseño. Adicionalmente, estos procesos hacen uso de algunos servicios básicos de CORBA como el servicio de nombres (nsd) y el servicio de eventos (eventd). La figura 37 ilustra las interacciones entre los distintos procesos del terminal y la baliza.

* La precisión de la ubicación depende de la granularidad de las redes inalámbricas utilizadas. En el caso particular de Bluetooth dependerá de la cobertura, la clase de los dispositivos, etc.

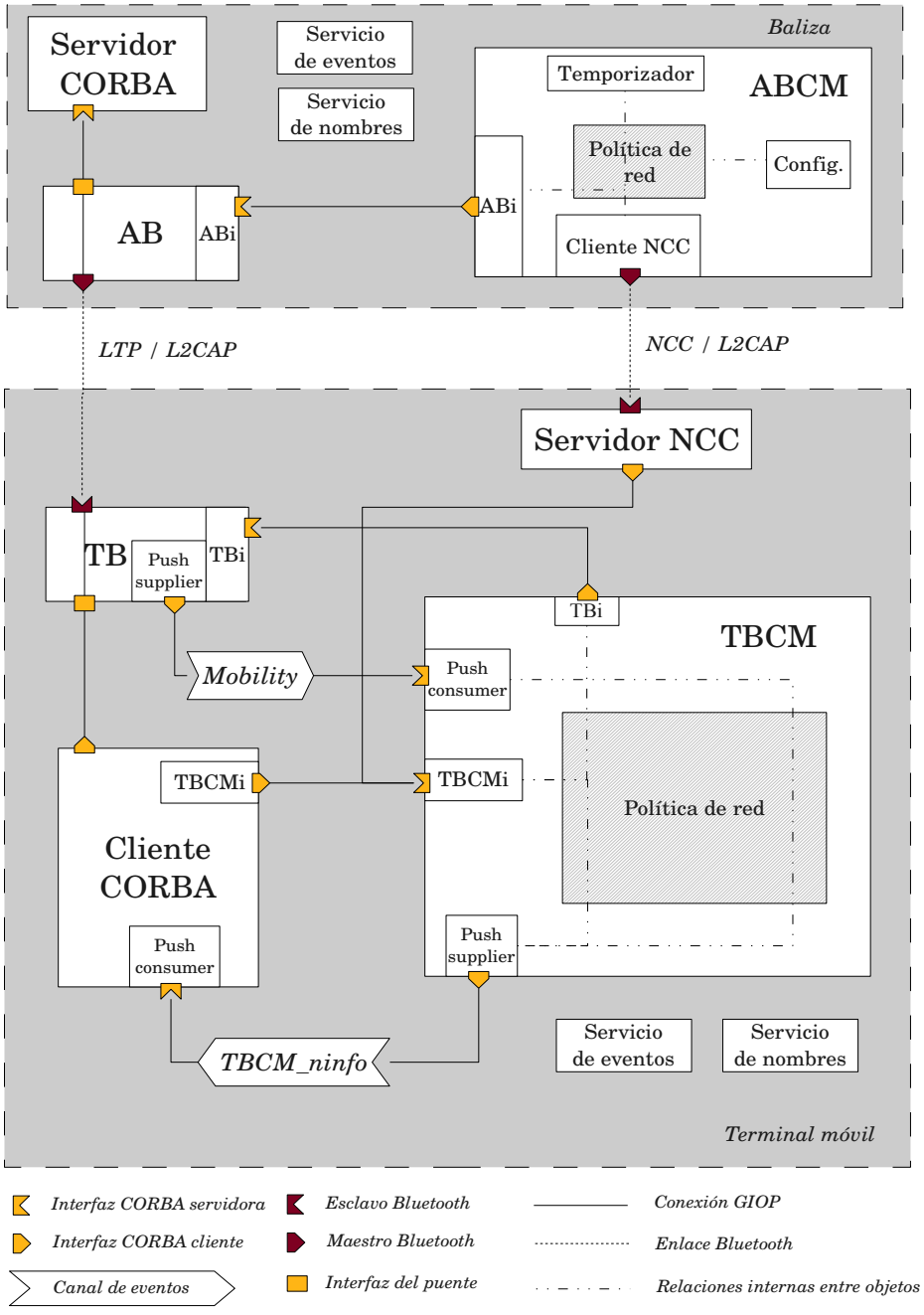

Figura 37: Relación entre los procesos de la baliza y el terminal

Terminal Bridge Connection Manager (TBCM)

El módulo TBCM es el responsable del establecimiento y liberación de conexiones inalámbricas en el terminal. Para ello utiliza fundamentalmente los métodos connect() y release() de la interfaz CORBA de TerminalBridge aunque potencialmente puede utilizar cualquier otro incluido en su descripción IDL, como los nuevos métodos idle_bridge() o get_strength(), que informan de la actividad del puente y de la calidad del enlace inalámbrico, respectivamente. A continuación se muestran las definiciones IDL de los nuevos métodos añadidos a la interfaz de TerminalBridge.

```
boolean idle_bridge( in unsigned long tout );
boolean get_strength( out long st );
```

La interfaz CORBA no es el único medio por el que TBCM obtiene información de TerminalBridge. El gestor de conexiones también se suscribe al canal de eventos de TerminalBridge en busca de mensajes que le ayuden a determinar cuándo está disponible el puente o cuándo se interrumpe la conexión. Por otra parte, TBCM también crea su propio canal de eventos mediante el cual notifica a las aplicaciones la disponibilidad de la conexión y la detección de nuevos puntos de acceso. Esta información se empaqueta en la estructura st_ninfo, descrita en la propia interfaz IDL de TBCM y tiene dos campos:

- addr: Indica la dirección de la baliza a la que hace referencia el estado, si procede.
- status: Indica el estado de la conexión. Puede tomar los siguientes valores:
  - NINFO_DISCONNECTED. Indica que se ha perdido la conexión.
  - NINFO_DETECTED. Indica que la dirección corresponde a una nueva baliza detectada.
  - NINFO_CONNECTED. Indica que se ha establecido conexión con la baliza indicada en addr.

TBCM también dispone de una interfaz CORBA propia cuyos métodos se pueden dividir en dos grupos: los que son invocados por servidores NCC y los que son invocados por las aplicaciones cliente ordinarias. La tabla 17 describe el uso de los métodos descritos en la interfaz IDL de TBCM, de la que se muestra un fragmento:

```
  typedef sequence < string > contextlist;

  struct st_ninfo {
      short status;
      string addr;
  };

  struct st_bior {
      string srvname;
      string srvior;
  };

  typedef sequence < st_bior > iorlist;

  interface TBCM {
      //Methods for ordinary client applications
      short connect(in string addr);
      string  get_bs_ior(in  string  addr,  in  string
servicename);
      contextlist get_context_list(in string addr);
      short get_strength(in string addr);
      //Methods for NCC servers
      void report_beacon(in short type, in string addr,
in contextlist cl, in iorlist il);
      void yield();
  };
```

Método	Descripción
connect	Se utiliza para solicitar a TBCM que se conecte a la baliza identificada por la dirección addr. El hecho de invocar este método no implica necesariamente que se efectúe la conexión, sino que simplemente se trata de una petición. La decisión final ha de tomarla el objeto de política de red.
get_bs_ior	Busca la referencia al servicio identificado por el parámetro servicename en la lista de servicios proporcionada por la baliza cuya dirección se indica en el parámetro addr. Si encuentra la referencia, devuelve el IOR en forma de cadena de caracteres. En caso contrario, devuelve una cadena vacía.
get_strength	Devuelve la fuerza de la señal recibida, expresada en porcentaje, para una conexión dada.
get_context_list	Devuelve la secuencia de cadenas de caracteres que indican el contexto jerarquizado de la baliza cuya dirección se indica en el parámetro addr. Si encuentra la referencia, devuelve el contexto. En caso contrario, devuelve una secuencia vacía.
report_beacon	Informa a TBCM de la presencia de una baliza, aportando su dirección, contexto y referencias básicas. Este método es usado por servidores NCC
yield	Solicita a TBCM que libere el puente AB↔TB.

Tabla 17: Métodos de la interfaz TBCM

Por último, el núcleo del gestor TBCM lo constituye el objeto de política de red, del que se hablará más adelante.

Desde el punto de vista de la implementación, TBCM es una aplicación desarrollada en C++ compuesta por 5 clases propias y algunas externas, tal y como se muestra en la figura 38.

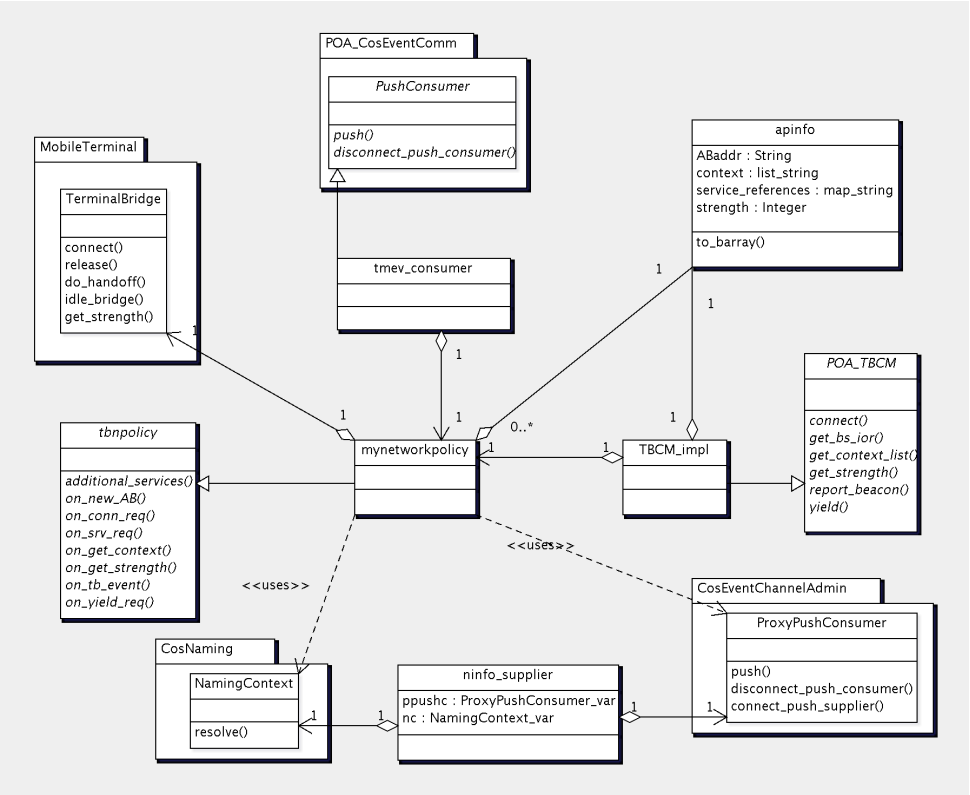

Figura 38: Diagrama de clases de TBCM

De acuerdo con el diagrama, la clase consumidora de eventos del terminal (tmev_consumer) es la encargada de suscribirse al canal de eventos de TerminalBridge y recibir la información. Para ello, la clase ha sido implementada en forma de push consumer [76]. La interfaz CORBA descrita para TBCM es implementada en la clase tbcm_impl. La clase información del punto de acceso (apinfo) se utiliza como apoyo para el almacenamiento y el intercambio de la información de red. Su estructura interna está diseñada para facilitar la transmisión y recepción de dicha información mediante el protocolo NCC. La clase de notificación de información de red (ninfo_supplier) abstrae del proceso de obtención de las referencias a los servicios de nombres y eventos que después serán utilizadas por las otras clases. Esta clase también se encarga de crear el canal de eventos particular de TBCM (TBCM_ninfo) a través del cual se emiten las notificaciones a las aplicaciones cliente. Por último, la clase principal (mynetworkpolicy, en el caso particular de la figura 38), situada en el centro del diagrama es la clase que implementa la política de red. Esta clase centraliza la información recibida del resto de

clases y actúa en consecuencia.

La forma de uso y las opciones particulares de TBCM se describen en el anexo A4.

Access Bridge Connection Manager (ABCM)

El módulo ABCM es el responsable de la detección de terminales y de la supervisión de los puentes que AccessBridge establece con los distintos terminales. Desde el punto de vista de la arquitectura interna, ABCM es más simple que su homólogo en el lado del terminal. ABCM, al igual que TBCM, está escrito en C++ y consta de 4 clases propias y algunas externas como se muestra en la figura 39.

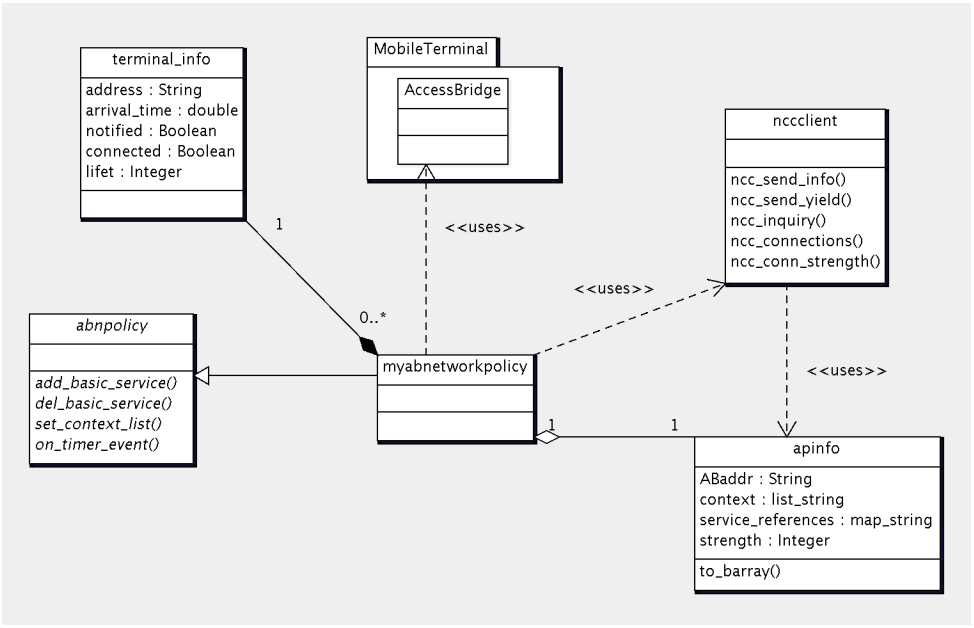

Figura 39: Diagrama de clases de ABCM

Como se observa en el diagrama, ABCM también utiliza la clase de información del punto de acceso (apinfo) por los mismos motivos expuestos anteriormente para TBCM. La clase del cliente NCC (nccclient) engloba todas las tareas referentes al canal NCC y al descubrimiento de terminales. ABCM también obtiene una referencia a la interfaz CORBA de AccessBridge para poder intervenir en el establecimiento y liberación de los puentes si es

necesario. Por último, el corazón de ABCM, al igual que ocurría en TBCM, es el objeto de política de red, que deriva de la clase abstracta abnpolicy. En el caso particular de la figura 39 el objeto de política de red se concreta en la clase myabnetworkpolicy. En la implementación concreta que se ha realizado, el objeto de política de red, además de implementar los métodos de abnpolicy y utilizar las clases descritas, hace uso particular de la clase de información de terminales (terminal_info). Esta clase sirve para llevar un control de los terminales descubiertos, lo que ayuda en una correcta gestión de la información que se envía a los terminales, evitando de esta forma envíos repetitivos e innecesarios de la misma información a un terminal. En cualquier caso, esta clase se ha creado específicamente para ser usada por parte de la implementación particular del objeto de política de red, por lo que no es un componente esencial de ABCM.

Al margen de las clases, existen otras tareas que se realizan fuera de las mismas y que son importantes para el correcto funcionamiento de ABCM:

- Configuración. El objeto de política de red debe ser configurado tras la creación de su instancia mediante las funciones que la clase abnpolicy define a tal efecto.
- Temporización. El control del tiempo entre operaciones o del tiempo asignado a la realización de ciertas operaciones, como el descubrimiento de terminales, debe ser regulado fuera de las clases. El código que realiza la temporización es responsable de llamar a ciertas funciones del objeto de política de red, siendo en gran medida el motor de las operaciones de ABCM.

Otra de las labores del código que envuelve a las clases es la obtención de la información básica que se transmitirá a los terminales y del resto de información de configuración. La descripción de la forma de uso y de las opciones especiales que deben darse a ABCM para que lleve a cabo esta labor se explican en el anexo A4.

Network Control Channel (NCC)

Existe cierta información básica que un terminal debe conocer para poder empezar a buscar los servicios de su interés. Dado el carácter nómada de los terminales que acceden a redes en entornos urbanos, éstos no pueden conocer a priori dicha información por lo que necesitan un método para obtenerla de forma dinámica. El protocolo NCC se encarga precisamente de solucionar este problema.

Información básica de red

Por cuestiones de agilidad es conveniente que el paquete de información básica de red sea lo más breve posible, pero a la vez lo suficientemente útil como para que un terminal pueda localizar el servicio deseado a partir de él. En este sentido, se ha seleccionado el siguiente conjunto de información para ser transmitida a través del canal NCC a los terminales que entran en el alcance de una baliza:

- Información de Contexto. Se trata de una descripción jerarquizada del contexto en el que se ubica la baliza y lo establece libremente el administrador de la infraestructura. Se concreta en una lista variable de cadenas de caracteres.
- Referencias básicas. Se trata de un conjunto de pares clave/valor donde la clave es el nombre de un servicio y el valor es la referencia IOR correspondiente.
- Comandos. Se trata de órdenes para la correcta gestión de la red.

Debido a la longitud de las referencias IOR, el conjunto de referencias básicas debería ser limitado. En la práctica se estima que bastaría únicamente con incluir la referencia a un servicio de trading que diera acceso al resto de los servicios. En cualquier caso, es necesario que haya consenso en la elección de la clave que identifique a estos servicios básicos para garantizar la interoperabilidad de los terminales en cualquier entorno. En este sentido se proponen las claves indicadas en la tabla 18.

Clave	Servicio
Trading	Servicio de Trading estándar de CORBA
Naming	Servicio de nombres estándar de CORBA

Tabla 18: Claves de servicios básicos

Protocolo NCC

El protocolo NCC se ha diseñado para transportar la información de red de una manera sencilla en el caso particular de las redes Bluetooth, aunque podrían crearse fácilmente variantes del mismo si las necesidades de la aplicación requirieran el uso de otras tecnologías inalámbricas. En el caso

concreto de la implementación presentada en esta Tesis, el protocolo es unidireccional (desde la baliza hacia el terminal), se apoya en la capa L2CAP y sobre ella se transfiere la información serializada en forma de cadena de caracteres delimitada por ciertas secuencias especiales de caracteres. Como delimitadores se han usado los caracteres '#' y '>'. Por simplicidad se asume la limitación en el uso de los mismos, restringiéndolos a la función de delimitación, por lo que no pueden ser usados dentro de los campos de información.

El primero de los caracteres ('#') es el delimitador de campos y también se usa para marcar el final de la trama. Este delimitador es seguido por un carácter que identifica el tipo de campo. Hasta el momento se han definido tres tipos de campos y un identificador especial de final de trama. El conjunto de identificadores se muestran en la tabla 19.

Identificador	Tipo de campo
'C'	Información de contexto
'R'	Referencias a servicios básicos
'Y'	Petición de liberación del puente (yield)
'X'	Final de trama

Tabla 19: Tipos de campos del protocolo NCC

El segundo delimitador ('>') se utiliza para separar subapartados dentro de un campo. Dado que el tipo de información que se transfiere puede ser considerado en la mayoría de los casos como un array de cadenas de caracteres, puede haber varios delimitadores de este tipo dentro de un campo. El campo de información de contexto está formado por un conjunto de cadenas de caracteres ordenadas jerárquicamente en orden decreciente, es decir, empezando por los contextos más amplios. El campo de referencias a servicios básicos es un array compuesto por parejas de cadenas de caracteres que alternativamente representan el nombre del servicio y su referencia IOR convertida en cadena de caracteres. Como se mencionó anteriormente, el conjunto de nombres de servicios está restringido a los que aparecen en la tabla 18. Tanto el campo de petición de liberación del puente como el de final de trama constan únicamente del identificador. El siguiente ejemplo ilustra la confección de una trama del protocolo NCC:

Una baliza instalada en el departamento de Ingeniería Electrónica de la Universidad de Sevilla que ofrece un servicio de trading enviaría una trama como la siguiente:

```
#C>Universidad de Sevilla>Escuela Superior de
Ingenieros>Departamento de Ingeniería
Electrónica>#R>Trading>IOR:010000001000000049444
c3a4163636f756e743a312e300002000000000000034000
000010102000a0000003132372e302e302e3100050d17000
00044656661756c742f4163636f756e742f4163636f756e7
400000000001000000240000000100000001000000010000
000140000000100000001000100000000009010100000000
000>#X
```

Integración de NCC en los gestores de conexión

Los módulos del protocolo NCC se han integrado de diferente manera en TBCM y en ABCM. En el caso de TBCM, el servidor NCC, que en este caso se dedica a escuchar la información proveniente de las balizas, se ha implementado como un proceso aparte y el punto de conexión con el gestor es a través de la propia interfaz CORBA de TBCM.

Esto aporta una mayor independencia al diseño y potencialmente permitiría a TBCM recibir información de varios módulos NCC que usasen coenlaces basados en distintas tecnologías inalámbricas para los procesos de descubrimiento y transferencia de información básica de la red, incluso, de forma simultánea. El hecho de poder acceder a un conjunto heterogéneo de redes puede llegar a ser un aspecto muy interesante para los terminales ya que en determinadas situaciones, las condiciones del entorno podrían aconsejar el uso de otras tecnologías inalámbricas.

En el caso de ABCM, la situación es ligeramente diferente. En este caso la necesidad de emplear diversas tecnologías inalámbricas en un mismo equipo no es tan fuerte. El sistema planteado en esta Tesis se centra en el empleo de Bluetooth, tecnología inalámbrica adecuada tanto para el intercambio de información como para el canal NCC, optándose por integrar el cliente NCC como una clase más del proceso ABCM.

Políticas de conexión

Existe un conjunto de factores como heterogeneidad de los terminales, el

tipo de aplicación o las situaciones particulares que pueden hacer variar las necesidades de terminales y balizas no sólo entre equipos distintos, sino también para un mismo equipo cuando cambia sustancialmente su entorno. Así, por ejemplo, en un terminal podría ser más conveniente que TBCM se conectara automáticamente a las redes cada vez que es detectado por las balizas y en otro terminal podría ser más conveniente conectarse sólo si alguna de las aplicaciones cliente lo pide. Esta flexibilidad requiere que el diseño de los gestores de conexión permita el cambio en la política de red de una manera sencilla. En el sistema propuesto en esta Tesis, esto se consigue creando una clase abstracta de la que pueden derivarse múltiples objetos de política de red que heredan unos métodos comunes. La diferenciación de las políticas de red está en la forma de implementar estos métodos. Durante el diseño y definición de los métodos se ha procurado independizar en la medida de lo posible a estos objetos respecto del resto del programa para que la implementación interna fuera lo más fácil posible. Se ha favorecido que estas subclases pudieran tener un comportamiento reactivo, evitando el uso de múltiples hilos* o temporizadores en su interior.

En el caso de TBCM, los objetos de política de red heredan de la clase abstracta tbnpolicy, que define los métodos indicados en la tabla 20.

* En el momento de redactarse este documento, el uso de wireless CORBA en MICO todavía es incompatible con el soporte para aplicaciones multihilo.

Método	Funcionalidad
additional_services	Es invocado tras la creación de la instancia y sirve para pasarle al objeto las referencias al servicio de nombres y al canal de eventos propio de TBCM.
on_new_AB	Es invocado cada vez que el servidor NCC es detectado por una baliza y pasa como parámetro la información de red que ésta le haya transferido.
on_conn_req	Es normalmente invocado por las aplicaciones cliente para solicitar a TBCM que establezca un puente con la baliza indicada. El objeto de política de red no tiene que forzar la conexión necesariamente cuando este método es invocado. La decisión final dependerá de la implementación interna del objeto.
on_srv_req	Devuelve la referencia IOR al servicio indicado aportada por la baliza referenciada por los parámetros.
on_get_context	Devuelve el contexto de la baliza referenciada por los parámetros.
on_get_strength	Devuelve la fuerza de la señal, expresada en porcentaje, correspondiente a la conexión identificada por la dirección de la baliza indicada como parámetro.
on_tb_event	Es invocado cada vez que TerminalBridge lanza un evento a través de su propio canal. La información que acompaña al evento se pasa como parámetro.
on_yield_req	Es invocado cada vez que el servidor NCC recibe el comando “yield”.

Tabla 20: Métodos abstractos para la política de red en TBCM

En el caso de ABCM, los objetos de política de red heredan de la clase abstracta de política de red en el lado de la baliza (abnpolicy), que define los métodos descritos en la tabla 21.

Método	Funcionalidad
add_basic_service	Añade un par nombre/referencia a un servicio básico. Estas referencias serán enviadas a los terminales.
del_basic_service	Borra una referencia a un servicio.
ncc_client_ref	Aporta al objeto una referencia a una instancia del cliente NCC.
ab_ref	Aporta al objeto una referencia al objeto CORBA que representa el AccessBridge.
set_context	Establece la información de contexto que será enviada a los terminales.
on_timer_event	Es invocado cada vez que vence el temporizador externo a la clase, para que ésta pueda realizar las tareas que se ejecutan de forma periódica.

Tabla 21: Métodos abstractos para la política de red en ABCM

En principio, podría diseñarse cualquier tipo de política de red tanto para las balizas como para los terminales, pero para la primera implementación del sistema se ha optado por una política de red sencilla. Esta política hace que se establezca automáticamente una conexión entre TB y AB cada vez que el terminal es descubierto por una baliza. Ante la petición de conexión a una baliza por parte de cualquier aplicación cliente, el objeto de política reacciona obedeciendo la orden, y libera el puente si está ocioso cada vez que recibe una petición de tipo “yield”. Con todo esto, en la figura 40 se muestra un diagrama con la secuencia básica de conexión y acceso a los servicios que, aunque particularizada para este caso, sería parecida a la que tendría lugar empleando otras políticas de red.

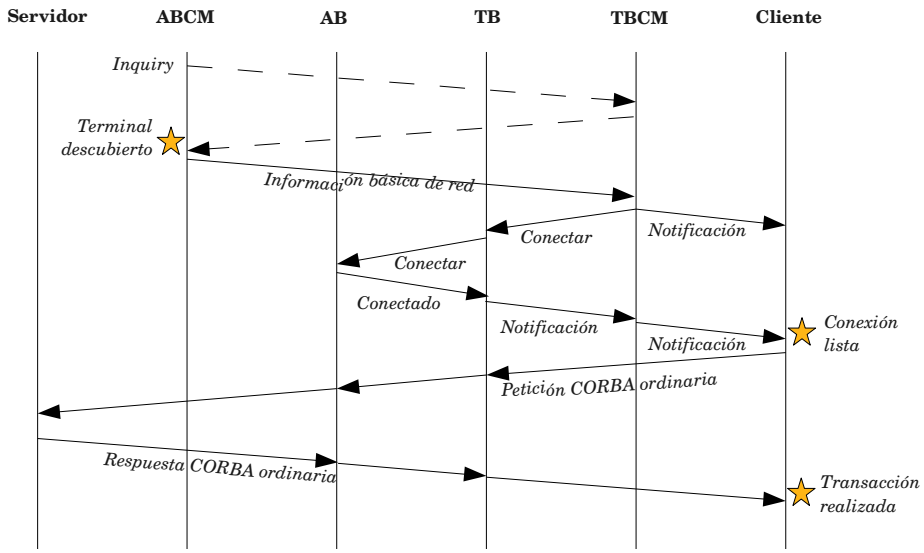

Figura 40: Secuencia básica de conexión y acceso a servicios

Eficiencia

Roles

Al analizar la tecnología Bluetooth se comentó que un dispositivo puede funcionar como maestro o como esclavo en una piconet. En determinadas situaciones, cuando tenemos múltiples dispositivos, éstos pueden comportarse como maestros de su propia piconet pero a la vez pueden participar como esclavos en otras piconets, formando lo que se denomina como scatternet. Sin embargo, esta "simultaneidad" se resuelve en la realidad mediante la multiplexación por división en el tiempo, esto es, que los dispositivos reparten su tiempo atendiendo a las diferentes piconets en las que participan, estando activos sólamente en una de ellas en cada instante. Este detalle es importante a la hora de establecer el modelo de comunicación, ya que el rendimiento y la velocidad de transferencia van a depender en buena medida del rol que se adopte para cada elemento del sistema.

Si se tiene en cuenta que en una piconet sólo puede haber hasta un máximo de 7 conexiones activas y que, en principio, un esclavo podría participar en un número indeterminado de piconets, en primera aproximación, se podría pensar que es más conveniente que las balizas adoptaran el rol de

esclavas y los terminales el de maestros ya que de esta forma se soslayaría la limitación de las 7 conexiones activas. Si se considera la forma en la que Bluetooth consigue esta "simultaneidad" se llega a la conclusión de que esto reduce la velocidad de transmisión efectiva, no sólo por el tiempo que conlleva la resincronización con la nueva piconet, sino también porque en cada piconet, la baliza debería compartir las ranuras con el resto de conexiones que ya hubiera en dicha piconet. El modelo de comunicación en el que la baliza adopta el rol de maestra y los terminales el de esclavos, a pesar de la limitación de conexiones activas, es más eficiente en cuanto al aprovechamiento del ancho de banda.

En principio, el dispositivo que inicia la comunicación es el que adopta el rol de maestro. Esto podría suponer un problema si en algún momento se hace por parte del terminal, ya que habría que incluir mecanismos de cambio de rol. Afortunadamente, la solución puede hacerse de una forma más sencilla puesto que la pila de protocolos BlueZ, a través de comandos HCI permite configurar el modo del enlace de manera que un dispositivo pueda aceptar conexiones entrantes pero adoptando automáticamente el rol de maestro. Esta configuración puede hacerse directamente sobre la consola de Linux con la orden "lm" del comando hciconfig, indicándole las opciones "MASTER" y "ACCEPT". Por tanto, bastaría con llamar a dicho comando durante la inicialización de la baliza.

Consumo energético

Un factor crítico que hay que tener en cuenta en los dispositivos utilizados en los entornos urbanos es el consumo energético. Muchos de los dispositivos terminales (PDAs, teléfonos móviles y otros equipos portátiles) están alimentados por baterías y la autonomía de los mismos puede ser crucial en el éxito de estos productos. En el caso particular de la tecnología Bluetooth, es necesario que uno de los dispositivos busque al otro mediante el proceso de inquiry. Cabría pensar que es más conveniente que el terminal sea quien realice la búsqueda de forma periódica mientras éste no esté intercambiando datos con una baliza. De esta forma, se hace un mejor aprovechamiento del ancho de banda de la piconet ya que la baliza no tendría que dedicar parte de su tiempo a buscar terminales. Ciertamente, este esquema conlleva ventajas en cuanto a la velocidad efectiva en la piconet, pero si tenemos en cuenta que, por los requerimientos de los entornos urbanos, los ciclos de búsqueda son bastante cortos llegaremos a la conclusión de que el terminal tendría que destinar una parte significativa de su tiempo al proceso de inquiry. Durante este proceso, el consumo energético del dispositivo

Bluetooth aumenta considerablemente, por lo que se corre el riesgo de agotar la batería rápidamente.

Por este motivo, se desaconseja que la búsqueda la realice el terminal. Es preferible dejar esta responsabilidad a la baliza que, normalmente, no tiene limitaciones tan estrictas en cuanto a consumo energético. Como contrapartida, se reduce la velocidad de transmisión efectiva en la piconet, ya que la baliza debe reservar parte de su tiempo para las tareas de inquiry.

Conexiones ociosas

Como se comentó anteriormente en este capítulo, se realizaron modificaciones sobre el ORB de MICO y sobre los puentes para dotarlos de mecanismos de detección de conexiones ociosas, por el perjuicio que ello representaba. La inclusión de estas medidas permite trasladar dicha información al objeto de política de red y descargar sobre él la toma de decisiones al respecto. Por un lado, el objeto de política de red puede recibir peticiones para que libere el puente por parte de la baliza a través de la orden "yield" contemplada en el protocolo NCC. La liberación del puente no es obligatoria, pero sí aconsejable cuando se recibe esta orden. Para la toma de la decisión, el objeto de política de red puede hacer uso del método idle_bridge que se añadió a la interfaz CORBA de TerminalBridge. Con esta información, el objeto podría, por ejemplo, acceder a liberar el puente sólo si no hay ninguna transacción en curso a través del puente con el objetivo de no perturbar a las aplicaciones cliente.

Otras consideraciones

Debido a que, según el esquema propuesto, las balizas se encargan del proceso de descubrimiento y transferencia de la información básica de red, éstas deben tener conocimiento de la existencia de otras balizas en las inmediaciones para no enviarles por error dicha información básica. Esto puede hacerse mediante una lista de balizas adyacentes aunque en el caso particular de Bluetooth puede hacerse una simplificación si asumimos que las balizas no necesitan ser descubiertas por otros dispositivos bluetooth. En ese caso, se puede hacer que las balizas sean "invisibles" al proceso de inquiry. Para ello, basta con llamar durante la inicialización de la baliza al comando hciconfig desde la consola de Linux con la orden "pscan" para habilitar únicamente el modo page scan y, en consecuencia, deshabilitar el modo inquiry scan, que es el que hace "visible" a un dispositivo Bluetooth.

Aplicaciones

Con objeto de demostrar la funcionalidad y versatilidad del sistema propuesto en esta Tesis se plantean en este capítulo dos aplicaciones en áreas importantes enmarcadas dentro de los entornos urbanos inteligentes.

La primera de ellas se engloba dentro de los servicios inmóticos y ofrece información de orientación en interiores de edificios. Esta aplicación se apoya parcialmente en algunos trabajos realizados en el marco del proyecto FENIX, en el que el autor de esta Tesis ha participado activamente junto con el grupo de investigación ACE-TI, de la Universidad de Sevilla, y la empresa Visión Sistemas de Localización, S.L. Este proyecto ha sido financiado por el programa Avanza I+D del Ministerio de Industria, Turismo y Comercio, y contempla varias actuaciones en el ámbito de las aplicaciones para la protección de víctimas de la violencia de género. Además del desarrollo de una comunidad virtual para mujeres maltratadas, el proyecto incluye el desarrollo de un sistema de localización permanente de los agresores. Este sistema de localización contempla la posibilidad de completar la información de localización con tecnologías diferentes a GPS cuando ésta no está disponible, situación que se da, por ejemplo, en el interior de edificios. En esos casos, el sistema prevé una localización basada en la tecnología Bluetooth mediante el uso de sistemas empotrados, que actúan como balizas, instalados en el edificio. La posibilidad de localización abre la puerta al desarrollo de otras aplicaciones adicionales sobre el sistema, como, por ejemplo, la localización de personas en caso de desastres [77]. Siguiendo en la línea de ampliar funcionalidades y explotar el potencial de estos sistemas, se ha continuado el desarrollo de estas balizas para dar lugar, con las adaptaciones pertinentes, a las balizas principales y de localización, descritas en el anexo A1 y que forman parte de un sistema más complejo desarrollado en esta Tesis [78]. En todo caso, la aplicación de la plataforma basada en Wireless CORBA, que se describió anteriormente, aporta a las balizas originales del proyecto FENIX una mayor versatilidad y capacidad para dar soporte a infinidad de servicios, evitando la proliferación de un ecosistema de soluciones específicas e incompatibles entre equipos, como se describió en el capítulo 2.

La segunda aplicación se centra en el desarrollo de equipamiento público en entornos urbanos y aporta información selectiva sobre el estado del tráfico en determinados puntos de una ciudad. Para la obtención de la información de tráfico se plantea el uso de los equipos denominados VisioWay®. Estos equipos fueron desarrollados para Aeronaval de Construcciones e Instalaciones S.A. (ACISA) por el grupo de investigación

ACE-TI de la Universidad de Sevilla a través de AICIA*, en el marco del proyecto de cooperación industrial denominado "Visión artificial aplicada al tráfico y transporte", en el que participó el doctorando, que además formó parte del departamento de I+D de ACISA durante los años de implantación comercial del producto (2007 - 2008). El proyecto fue financiado por el Plan de Investigación y Modernización de Andalucía, promovido por la Junta de Andalucía. El objetivo del proyecto fue el diseño de una plataforma hardware y software para el desarrollo de productos basados en visión artificial destinados a aplicaciones de tráfico [79]. Este y otros proyectos, como el denominado "Sistemas Embebidos para la Seguridad Vial", realizado igualmente por el grupo de investigación ACE-TI para ACISA a través de FIUS§, han dado lugar a dos equipos comercializados por ACISA; un producto para reconocimiento automático de matrículas y otro para reguladores de tráfico.

Aplicación 1: Orientación en interiores

A partir del proyecto FENIX, el doctorando desarrolló bajo la tutela de sus directores de Tesis y con la colaboración de otros miembros del grupo de investigación ACE-TI, un sistema empotrado versátil con capacidad de integrar nuevos servicios TIC. Se pensó, en concreto, en el desarrollo de un servicio que pudiese ofrecer orientación en el ámbito de un edificio inteligente y a personas en movimiento. Para darle mayor interés, el servicio se enfoca fundamentalmente a personas que, por alguna limitación, necesiten ayuda a la hora de localizar un determinado lugar dentro del edificio. Entre estas limitaciones cabe citar las siguientes:

- Deficiencia visual
- Deficiencia auditiva o de expresión oral
- Dificultades por cuestiones de idioma

En estos casos, la posibilidad de disponer de un terminal adaptado que ofrezca las indicaciones oportunas para llegar a un lugar puede ser de gran utilidad. Por ello, en este apartado se presenta un ejemplo de aplicación de orientación en interiores basado en el sistema propuesto en esta Tesis.

* La Asociación de Investigación y Cooperación Industrial de Andalucía (AICIA) es una entidad vinculada a la Universidad de Sevilla a través de la cual se gestiona la relación Universidad – Empresa.

§ La Fundación para la Investigación de la Universidad de Sevilla (FIUS) es una entidad cuyo cometido es similar al de AICIA.

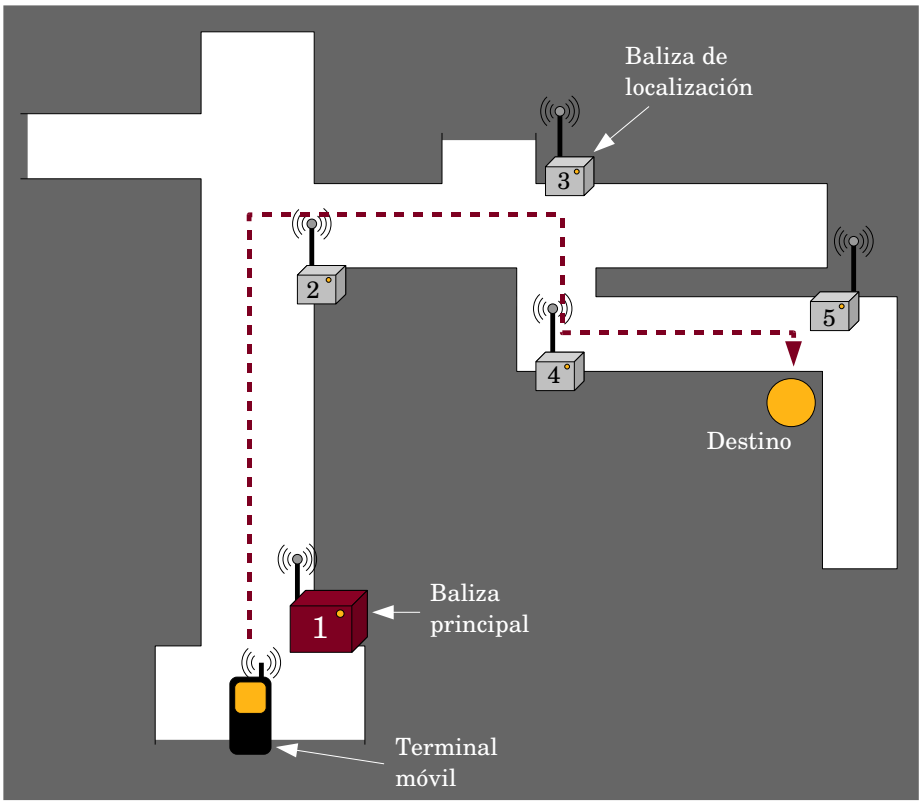

Figura 41: Sistema utilizado en la prueba del servicio de orientación

En la figura 41 se describe la estructura del sistema implementado para la prueba. El sistema consta de una serie de balizas distribuídas por el edificio y un terminal móvil que porta la persona que utiliza el servicio. El objetivo del servicio consiste en guiar a la persona desde el punto de entrada hasta el destino que haya seleccionado entre la lista que se le ofrece.

Desde el punto de vista del hardware, el sistema está formado por tres tipos de elementos distintos:

- Baliza Principal
- Baliza de Localización
- Terminal

La baliza principal y la de localización se describen en el anexo A1 y están basadas en el sistema desarrollado para el proyecto FENIX. El terminal

ha sido desarrollado por el doctorando a partir del dispositivo móvil denominado OpenMoko [80], creado con idea de expandir el uso de los sistemas operativos libres en los dispositivos electrónicos personales. Desde el punto de vista hardware, el sistema operativo de OpenMoko se ejecuta sobre un prototipo de teléfono móvil, el Neo FreeRunner, cuyo diseño también es libre y está accesible. La figura 42 muestra una imagen del terminal Neo FreeRunner. Esto facilita especialmente la creación de software de bajo nivel y drivers a los desarrolladores de la comunidad de OpenMoko, pudiendo aprovechar todas las capacidades del teléfono. La idoneidad de este teléfono para este tipo de aplicaciones queda demostrada por las propias características técnicas y algunas iniciativas similares basadas en este terminal ([81], [82]). Entre las características más importantes del terminal Neo FreeRunner se incluyen:

- Procesador con núcleo ARM920T @ 400MHz
- Memoria SDRAM de 128MB
- Memoria Flash de 256MB
- Pantalla táctil de alta resolución (480x640 píxels)
- Bluetooth clase 2 compatible con la versión 2.0 +EDR
- Conectividad WiFi 802.11 b/g
- 2 Acelerómetros de tres ejes
- GPS
- GSM/GPRS tribanda
- Ranura para tarjetas micro-SD
- Puerto USB

El sistema operativo de OpenMoko está basado en GNU/Linux, existiendo actualmente varias distribuciones soportadas, algunas de ellas derivadas de distribuciones más generalistas como Debian o Gentoo, y otras específicas para OpenMoko, como OM2009, FDOM o QtMoko. El hecho de disponer de GNU/Linux como sistema operativo permite que los desarrollos software realizados para OpenMoko sean transportables a otros terminales que utilicen este sistema operativo.

Figura 42: Teléfono Neo FreeRunner

El servicio de orientación en interiores se ha planteado como un par de aplicaciones (cliente/servidor) creadas en C++. La aplicación servidora se instala en una baliza principal, mientras que la aplicación cliente se instala en el terminal móvil. El intercambio de información entre dichas aplicaciones viene determinada por la descripción IDL del servicio:

```
struct or_node
{
    short id;
    string place;
    short anglep;
    string address;
};

struct or_path
{
    short id1;
    short angle1;
    short id2;
    short angle2;
    short distance;
};

typedef sequence < or_node > nodelist;
typedef sequence < or_path > pathlist;

interface orienta
{
    nodelist get_nodelist();
    pathlist get_pathlist();
};
```

Básicamente, la aplicación servidora ofrece dos métodos a través de los cuales transfiere al cliente un grafo para que la aplicación cliente pueda determinar el camino hacia un lugar concreto. El grafo está compuesto por dos listas, una de nodos (nodelist) y otra de caminos (pathlist). Cada nodo representa una baliza, ya sea de localización o principal. Por cada nodo de la lista se recoge una serie de parámetros de interés, como un identificador (id), una descripción del lugar (place), la dirección Bluetooth de la baliza (address) o el ángulo de orientación del frontal de la baliza respecto al norte geográfico. Este último parámetro sirve para establecer una alineación inicial; en el caso particular de orientación de personas con deficiencias visuales, se puede recurrir a la emisión de señales acústicas para que la persona se sitúe frente a la baliza y, a partir de ahí, darle las indicaciones oportunas, teniendo en cuenta que la persona estará situada formando un ángulo con el norte geográfico aproximadamente igual al opuesto del indicado para la baliza.

En la descripción de cada camino se hace referencia a los nodos que conecta mediante sus identificadores (id1 e id2) y se especifican los ángulos de salida de cada uno de los nodos por el camino en cuestión, siempre expresados respecto al norte geográfico. La figura 43 ilustra gráficamente la definición de estos parámetros.

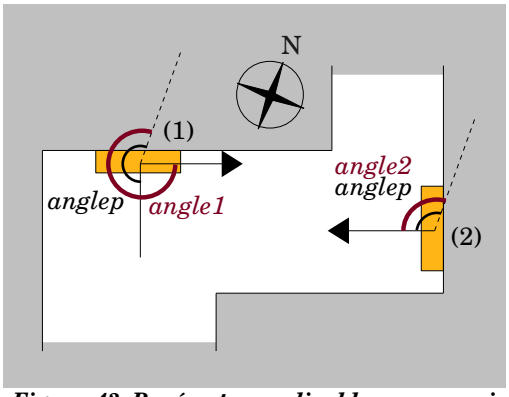

Figura 43: Parámetros aplicables a un camino

La primera vez que el terminal entra en el área de cobertura de una baliza principal que ofrece este servicio, éste solicita la lista de nodos y caminos. A continuación, la aplicación cliente ofrece al usuario todos los destinos posibles contemplados en el mapa para que elija. Una vez seleccionado el destino, la aplicación emite mensajes de orientación de forma automática sin necesidad de intervención del usuario. A medida que el

usuario pasa cerca de las balizas de localización, éstas lo detectan y le transmiten su información básica. La aplicación cliente del terminal es informada de estos eventos a través del canal de eventos de TBCM, al que se suscribe nada más arrancar. A través de estas notificaciones, la aplicación cliente extrae la dirección Bluetooth de la baliza que le ha detectado y puede ubicarla dentro del mapa. Dado que la aplicación cliente lleva un registro de las balizas por las que se ha pasado anteriormente, con estos datos y la información del mapa puede determinar los giros a realizar. Si el usuario toma un camino equivocado, la aplicación cliente puede darse cuenta si recibe una notificación de detección de una baliza contenida en el mapa y es capaz de reconducir al usuario por el camino correcto. Una vez alcanzado el destino, la aplicación lo notifica al usuario y vuelve al estado inicial.

La realización de la prueba se ha llevado a cabo en el edificio de la Escuela de Ingenieros. El recorrido realizado con el terminal móvil se corresponde con el esquema presentado en la figura 41 y está cubierto por cinco balizas, la primera de ellas es una baliza principal y el resto son balizas de localización. Todas las balizas utilizan dispositivos Bluetooth de clase 2. El recorrido se ha realizado varias veces por una persona caminando a una velocidad media estimada de 1,2 m/s. El objetivo es estimar la probabilidad de que el servicio se complete correctamente, recibiendo todas y cada una de las indicaciones a lo largo del recorrido. El experimento se ha repetido con distintos esquemas de reparto del inquiry cycle para determinar la idoneidad de cada uno de ellos en este tipo de aplicación.

Por simplicidad, la interfaz de usuario de la aplicación cliente está basada en consola, aunque podría adaptarse fácilmente a las necesidades del usuario. Así, por ejemplo, para personas con deficiencias visuales, los mensajes en pantalla podrían sustituirse por mensajes sonoros o incluso sonidos estéreo controlando el balance para emular el origen según el ángulo de giro. Para personas sin deficiencias visuales, los mensajes podrían sustituírse por indicadores gráficos tales como flechas con la orientación correspondiente.

En la tabla 22 se muestran los resultados obtenidos del experimento descrito anteriormente. Por cada uno de los experimentos se han realizado 10 repeticiones. En primer lugar, la tabla muestra las probabilidades de que el terminal sea detectado por un número determinado de balizas (NB). Dado que el montaje del experimento consta de 5 balizas, la columna P(NB=5) puede entenderse también como la probalibidad de completar el recorrido con éxito. En ningún experimento se dio el caso de que el parámetro NB fuera inferior a 3, razón por la cual no se han incluído en la tabla las columnas

correspondientes a dichos valores de NB. Por último, la columna P_B indica la probabilidad de ser detectado por una baliza cualquiera, que resulta de tener en cuenta los datos del experimento de forma global.

Reparto	Probabilidad de éxito			
	P(NB=5)	*P(NB=4)*	*P(NB=3)*	P_B
4:4	30%	60%	10%	83,9%
2:4	70%	20%	10%	92%
2:2	90%	10%	0%	97,9%
1:3	50%	40%	10%	88%

Tabla 22: Probabilidad de éxito en el servicio de orientación

En la tabla 22 se observa que el reparto más favorable para este tipo de aplicaciones es el 2:2. Este reparto implica ciclos de inquiry suficientemente cortos como para que dé tiempo a detectar y conectar con un terminal antes de que éste salga del área de cobertura. Como se comentó al principio del capítulo, los repartos que dedican más de un bloque de 1,28 segundos al proceso de inquiry suelen tener una menor desviación típica y un menor tiempo medio*. Esto se traduce en una disminución del tiempo de detección que es más acusada cuanto mayor es la tasa de errores en el canal. Por este motivo, el reparto 1:3, de igual duración que el 2:2, obtiene peores resultados incluso que el reparto 2:4 a pesar de que este último es más largo. En el caso particular del servicio descrito en este apartado, las necesidades de ancho de banda para la transmisión de datos no son muy altas, por lo que puede usarse el reparto 2:2 aunque suponga una reducción del 50% del throughput máximo. No obstante, si se necesitara un mayor throughput se pueden seleccionar los repartos 2:3 ó 2:4, con los que se obtendrían probabilidades de detección todavía aceptables.

Dada la estructura del edificio sobre el que se desarrollaron las pruebas, en aquellas ubicaciones donde las balizas estaban más próximas unas de otras (4m aproximadamente) se observó en un número pequeño de casos que la detección se producía en un orden incorrecto según la dirección del recorrido. Idealmente, las áreas de cobertura de las balizas de localización no deberían solaparse, pero en los casos en los que esto no sea posible se puede recurrir a reducir ligeramente la potencia de transmisión de dichas balizas o a utilizar

* La comparación se establece frente a repartos con un ciclo de inquiry similar pero con un solo bloque de 1,28 segundos para el proceso de inquiry.

antenas directivas siempre que el dispositivo Bluetooth permita la conexión de una antena externa adaptada específicamente para la ubicación.

Aplicación 2: Estado del tráfico

La segunda aplicación desarrollada se enmarca en el ámbito de los entornos urbanos inteligentes, y genera información actualizada del estado del tráfico en una ciudad. Este tipo de aplicaciones puede clasificarse dentro de los servicios de información de tráfico, que están entre los servicios ITS con mayor valor añadido [83]. A través de una plataforma como la descrita en esta Tesis, en combinación con un navegador GPS, se podrían realizar estimaciones más precisas de la duración de un recorrido programado y, si fuera necesario, se podrían calcular rutas alternativas donde el tráfico fuera más fluído. La ventaja fundamental de estos sistemas está en el ahorro de tiempo, aunque también existen otras ventajas no menos importantes como la posibilidad de informar a los conductores de incidencias puntuales tales como calles cortadas por obras, vías obstaculizadas por accidentes, etc. Además, desde el punto de vista de las administraciones locales, estos equipos pueden ayudar a sugerir a los conductores itinerarios con menor nivel de congestión de modo que, en términos globales, mejore el estado de circulación en la ciudad y se evite la formación de atascos y se mitigue la emisión de gases contaminantes.

Los actuales equipos ITS permiten la recolección de datos sobre el estado del tráfico en tiempo real [84] y, por tanto, la realización de una planificación dinámica de los itinerarios aconsejados. Los avances en el tratamiento digital de señales han propiciado que muchos de estos equipos estén basados en técnicas de visión artificial. Estos equipos pueden aportar gran cantidad de información a la vez que permiten ampliar las funcionalidades con algoritmos que extraigan nuevos parámetros sobre el estado del tráfico. Estos equipos utilizan diferentes técnicas para sus algoritmos dependiendo del tipo de información que se trate de extraer. Así, para los algoritmos de detección de vehículos se pueden usar modelos de fondo [85] para descartar zonas de la imagen irrelevantes y ayudar a distinguir los vehículos del fondo. En el caso de los algoritmos de reconocimiento automático de matrículas se puede recurrir a la detección de bordes verticales y a técnicas de crecimiento de regiones [86]. Además de una mayor abundancia y riqueza de información, los equipos basados en visión artificial son más fáciles de mantener, en contraposición con los antiguos sistemas basados en espiras, que en ocasiones se veían seriamente afectados por las frecuentes obras que tienen lugar en entornos urbanos. La instalación también resulta más sencilla que la

de las espiras, ya que no se necesita realizar ninguna obra sobre la propia calzada, evitando molestias a los usuarios.

Uno de los parámetros más interesantes de cara a la aplicación propuesta que se pueden obtener con estos equipos es la "ocupancia". Para entender este concepto es necesario aclarar brevemente el funcionamiento de estos equipos. Sobre la imagen capturada, se define una serie de polígonos conocidos como regiones de detección, que se activan cada vez que un vehículo los invade. Su funcionamiento es similar al de una espira aunque ofrece una mayor flexibilidad ya que se puede definir todo tipo de formas y tamaños. La ocupancia ofrece una estimación del porcentaje de tiempo que una región de detección está ocupada por un vehículo [87]. Para determinar el estado del tráfico en una vía se puede recurrir al número de vehículos que han atravesado una región de detección en un determinado tiempo, resultando en una estimación de tráfico más denso cuanto mayor sea el número de vehículos. Sin embargo, en ciertas situaciones en las que el tráfico circula con lentitud, como en un atasco, puede que haya pocos vehículos que atraviesen la región de detección, precisamente debido a su baja velocidad. En estos casos, el parámetro del número de vehículos induce a un error. La ocupancia, sin embargo, sí ofrece una relación más clara con la fluidez del tráfico. Por este motivo, en el servicio propuesto se ofrecerá como resultado precisamente el dato de ocupancia.

Para que esta aplicación pueda funcionar es necesario que exista una amplia red de balizas y equipos de visión artificial distribuidos por toda la ciudad, tal y como se muestra en la figura 44. Estos equipos, además, deben estar interconectados por algún tipo de red. Este último requisito en cuanto a infraestructuras, resulta bastante exigente y no todas las ciudades pueden cumplirlo. Sin embargo, las grandes ciudades habitualmente cuentan con redes de fibra óptica para la telegestión de diversos servicios de la administración local por lo que, en estos casos, sí se podría implantar la aplicación. Además, en estas situaciones existe la posibilidad de acceder a otros tipos de equipos ITS, con lo que las aplicaciones se enriquecen y pueden verse extendidas mediante el uso de CORBA como base para el intercambio de información entre equipos [88].

El objetivo del servicio consiste en ofrecer los datos de ocupancia de aquellas vías por las que el vehículo tenga previsto circular según la ruta calculada por el navegador GPS. El proceso se puede descomponer en dos tramos: el tramo "terminal–baliza" y el tramo "baliza–equipo de visión". En el primero de los tramos intervendrían los componentes de la plataforma planteada en esta Tesis (Wireless CORBA + Gestores de conexión) y la

comunicación se establecería a través de Bluetooth, mientras que en el segundo tramo, la baliza lanzaría peticiones CORBA ordinarias hacia los diferentes equipos de visión artificial involucrados en la ruta que se transportarían a través de una red TCP/IP, usando las capas de transporte habituales en CORBA. En definitiva, se trataría de una cadena de transacciones CORBA donde la baliza actuaría de intermediaria y daría soporte a la comunicación con el terminal.

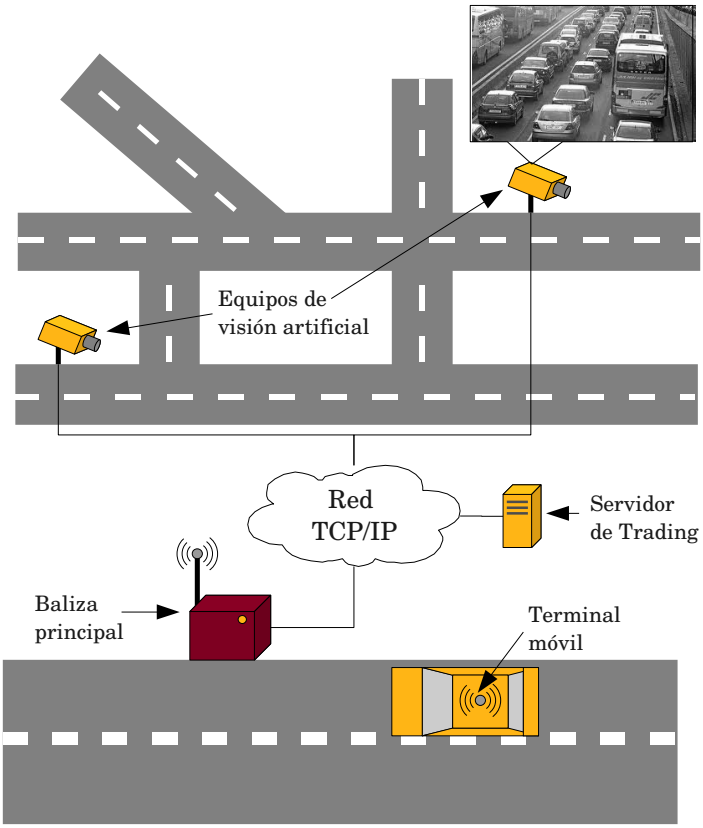

Figura 44: Esquema de una red de equipos para el servicio de estado del tráfico

Desde el punto de vista hardware, el sistema está formado por cuatro tipos de elementos distintos:

- Baliza Principal
- Equipo de Visión Artificial
- Terminal

- Servidor de Trading

La baliza principal es del mismo tipo que las utilizadas en la aplicación 1 con la salvedad de que en esta ocasión se utilizará un dispositivo Bluetooth de clase 1 por su mayor área de cobertura. El resto de las características técnicas se mantienen y pueden ser consultadas en el anexo A1.

El servidor de Trading se puede alojar en un PC convencional, por lo que sus características técnicas no resultan relevantes. Los otros dos elementos difieren de los utilizados en la aplicación 1 y serán descritos a continuación.

Como se comentó anteriormente, para esta aplicación se han utilizado equipos de visión artificial pertenecientes a la gama VisioWay. Concretamente se trata de equipos VisioWay OpenCounter, que extraen estadísticas de tráfico y entre los parámetros que obtienen está la ocupancia. Las características técnicas de estos equipos se describen en el anexo A1.

Para que la aplicación propuesta tuviera utilidad práctica, es necesario integrar en el sistema terminal que incorpore un navegador GPS con capacidad suficiente, tanto hardware como software, como para acceder al servicio mediante la plataforma propuesta, a la vez que realiza las operaciones propias de un navegador GPS. Por simplicidad, para las pruebas se utilizará un ordenador portátil a modo de terminal. Dado que la aplicación cliente que se ha diseñado para la prueba no aplica el resultado obtenido del servicio a un nuevo cálculo de ruta y no realiza representación gráfica alguna, las necesidades de la misma en cuanto a carga computacional son mínimas, por lo que la velocidad de procesamiento del ordenador portátil no es determinante. No obstante, sí es conveniente indicar las características técnicas del dispositivo Bluetooth integrado. En el caso particular del ordenador portatil utilizado, el dispositivo es de clase 1 compatible con la versión 2.1+EDR.

El ordenador portátil funciona con un sistema operativo GNU/Linux, por lo que tanto las aplicaciones como los elementos que rodean al ORB pueden ser fácilmente compilados para esta máquina.

El hecho de escoger dispositivos Bluetooth de clase 1 para esta aplicación se justifica por dos motivos:

- Los dispositivos de clase 1 ofrecen una mayor área de cobertura, lo que redunda en mayores tiempos de permanencia de los terminales en el

área de cobertura de las balizas. Este hecho es muy importante en aplicaciones en las que los terminales están embarcados en vehículos si se tienen en cuenta las velocidades a las que circulan éstos. En cualquier caso, la aplicación sólo contempla su uso en vías urbanas, donde las velocidades de los vehículos no son demasiado altas.

- A diferencia de las aplicaciones donde se utilizan terminales de mano alimentados por baterías, la restricción de consumo energético no es tan importante en vehículos, ya que tanto la potencia disponible como la autonomía son claramente superiores.

El servicio de información del estado del tráfico se ha planteado, por tanto, en dos tramos, como se ha comentado anteriormente. En el primer tramo, una aplicación servidora se instala en la baliza principal, mientras que la aplicación cliente se instala en el terminal móvil embarcado en el vehículo. La interacción entre el cliente y el servidor viene determinada por la descripción IDL del servicio, que se muestra a continuación:

```
interface infostreet
{
    short occupancy(in string streetname);
};
```

La interfaz prevé un único método a través del cual la aplicación cliente puede obtener el dato de ocupancia en tanto por ciento de la calle indicada en el parámetro "streetname", que identifica la calle por su nombre. Utilizando este parámetro, la baliza lo contrasta con una base de datos local que establece la correspondencia entre los nombres de las calles y un identificador numérico. Con este identificador, la aplicación servidora de la baliza principal recurre al servidor de trading para obtener una referencia al servicio de estado del tráfico ofrecido por el equipo de visión artificial que se halla en la calle indicada. Mediante la referencia, la aplicación de la baliza puede realizar una transacción CORBA, esta vez actuando como cliente, con la aplicación servidora alojada en el equipo de visión artificial. Por simplicidad, el tipo de servicio aplicable a este segundo tramo es idéntico al utilizado en el primer tramo y se utiliza la misma descripción IDL, aunque en este caso el parámetro "streetname" carece de sentido y no es tenido en cuenta. El dato así obtenido es finalmente trasladado por la baliza al terminal que inició la petición.

El uso del servidor de trading permite que cualquier baliza pueda localizar el servicio ofrecido por cualquier equipo de visión artificial de una manera dinámica. Al arrancar, los equipos de visión artificial exportan la

referencia a su servicio en el servidor de trading, indicando todos ellos el mismo tipo de servicio (Estado de tráfico) pero un valor distinto para la propiedad "Identificador de calle". Las balizas buscarán siempre en el servidor de trading servicios del tipo indicado pero estableciendo en cada caso un valor distinto para la propiedad "Identificador de calle".

En los equipos VisioWay OpenCounter, el dato de ocupancia para cada región de detección puede consultarse accediendo a un fichero concreto. Este fichero es actualizado periódicamente por la aplicación principal de procesamiento de vídeo (epi.out). La aplicación que implementa el servicio CORBA de estado del tráfico extrae de dicho archivo los últimos datos de ocupancia para las distintas regiones de detección, calcula la media aritmética de todos ellos y devuelve el resultado al cliente. La frecuencia de actualización de dichos datos está determinada por un parámetro de configuración de la aplicación principal.

Efectos del servicio middleware sobre la carga computacional

El importante tener en cuenta el efecto que la implantación de un servicio middleware tiene sobre un equipo de visión artificial. Los algoritmos que procesan las imágenes suelen hacer un uso intensivo de los recursos del sistema. De hecho, en los equipos VisioWay OpenCounter, el proceso principal (epi.out) acapara casi el 100% del tiempo de CPU en condiciones ordinarias. Dado que las imágenes capturadas llegan a una tasa determinada (25 imágenes por segundo), los algoritmos de visión artificial deben procesarlas muy rápido para no perder información. Por este motivo, la existencia de procesos adicionales puede reducir los recursos asignados al proceso principal, haciendo que éste no pueda procesar todas las imágnenes. Para estudiar el efecto del servicio descrito anteriormente sobre la aplicación de procesamiento de vídeo se han realizado pruebas de rendimiento realizando peticiones masivas al servicio. Las peticiones se han agrupado en ráfagas de 30 segundos con una densidad de peticiones variable para observar la degradación de la aplicación principal en función de la carga de peticiones. En la gráfica de la figura 34 se representan los porcentajes de tiempo de CPU asignados a la aplicación principal (epi.out) y al servicio CORBA de estado del tráfico en función del número de peticiones contenido en cada ráfaga.

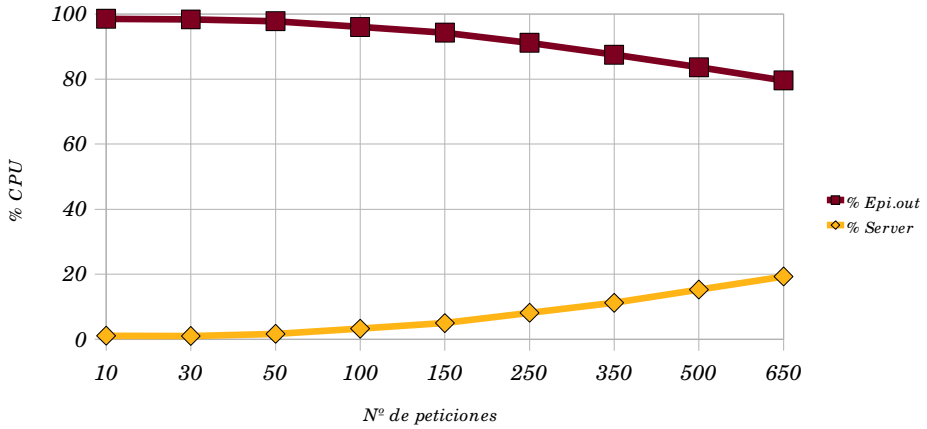

Figura 45: Efecto del servicio CORBA sobre la aplicación de visión artificial

De la gráfica se observa que el efecto sobre la aplicación de visión artificial empieza a ser apreciable con un número relativamente alto de peticiones (a partir de 100 peticiones en 30 segundos). Según estos datos se puede concluir que el servicio puede convivir con la aplicación principal sin que su efecto sea apreciable, a menos que se produzca un ataque deliberado por envío masivo de peticiones.

En cuanto al consumo de memoria RAM, la aplicación principal (epi.out) requiere 2732KB, mientras que el servicio CORBA requiere 4764KB, lo que suponen un 4,34% y un 7,56% de la memoria disponible, respectivamente.

El método de medición utilizado para estimar el tiempo de CPU consumido por cada proceso está basado en la información que el propio sistema operativo ofrece a través de ciertos ficheros especiales. Por las particularidades del sistema operativo y de las alteraciones que introduce el propio proceso de medida, se ha utilizado un método específico que tiene en cuenta estos factores para obtener una medida fidedigna. El método de medición se detalla en el anexo A2.

5 CONCLUSIONES

Contribuciones y resultados

Wireless CORBA se erige como una de las soluciones más interesantes para ofrecer servicios sobre redes inalámbricas, y en particular sobre Bluetooth. Sin embargo, su aplicabilidad a entornos urbanos y sistemas inteligentes de transporte no es directa. Por su diseño y enfoque, Wireless CORBA presenta algunas carencias que se manifiestan en situaciones habituales que se dan en los entornos urbanos inteligentes y que no están contempladas en su planteamiento original. Entre los principales problemas que se presentan en estos entornos cabe destacar los siguientes:

- Carácter nómada de los terminales. El planteamiento de pertenencia a un dominio carece de sentido.
- Ausencia de elementos de detección de terminales y gestión de las conexiones.
- Ausencia de mecanismos de presentación y acceso a los servicios.
- Desconocimiento previo entre terminales e infraestructura.

Precisamente una de las contribuciones de esta Tesis se centra en dar solución a estos problemas mediante la definición de nuevos módulos y modificaciones sobre los ya existentes. Como resultado se obtiene un nuevo sistema basado en gestores de conexión que añaden a Wireless CORBA las siguientes funcionalidades:

- Establecimiento y liberación automática de conexiones
- Medidas de gestión activa de las conexiones
- Información básica para el acceso a los servicios
- Adaptación a las particularidades técnicas de la tecnología inalámbrica

Una de las claves del sistema es su articulación en torno a objetos de política de conexión intercambiables, favorecida por un diseño modular. Esta modularidad también permite la sustitución de los elementos de detección, que dependen fuertemente de la tecnología inalámbrica utilizada.

Otra de las principales contribuciones de esta Tesis está relacionada con la tecnología utilizada (Bluetooth) y la forma en que se produce la detección de terminales. El estudio presentado plantea una caracterización

del tiempo de detección en función del esquema de reparto entre los tiempos de búsqueda y los de transmisión de datos y cómo éstos afectan al throughput global. Esta caracterización se ha corroborado mediante datos experimentales. Los resultados del estudio pueden ser utilizados para seleccionar el esquema de reparto más adecuado a cada situación considerando el tiempo de permanencia en cobertura de los terminales y la cantidad de datos intercambiados en las transacciones de los servicios.

La viabilidad técnica y práctica del sistema planteado queda demostrada por las aplicaciones presentadas al final del capítulo 4, donde se observa cómo es posible ofrecer servicios tales como la orientación en interiores compartiendo los recursos de los dispositivos Bluetooth tanto para la detección como para la transmisión de datos. Las aplicaciones también demuestran que es posible conseguir un sistema sencillo, económico y adaptado a una amplia gama de servicios muy diversos y de gran utilidad.

Futuras líneas de investigación

Dada la estructura del sistema presentado, una de las líneas de expansión natural se centra en el desarrollo de nuevos objetos de política de conexiones que, apoyados en la información que se aporta desde las capas de transporte y el módulo NCC, puedan establecer mecanismos más sofisticados y eficientes.

No obstante, podrían plantearse líneas de investigación bastante importantes desde el punto de vista de la interoperabilidad de servicios. El método de selección y acceso a los servicios planteado en esta Tesis puede verse mejorado si se introducen mecanismos avanzados de descripción de los servicios como, por ejemplo, una ontología común combinada con el uso de las invocaciones dinámicas de CORBA. Téngase en cuenta que dos servicios similares implementados por distintas entidades y ofrecidos, por tanto, a través de diferentes dominios pueden tener diferencias notables en su implementación. Para garantizar la interoperabilidad, un terminal debería estar preparado para reconocer el tipo de servicio y el tipo de parámetros adaptándose a una interfaz concreta y realizando las conversiones necesarias. Un ejemplo de esta situación sería un servicio de indicación del estado de tráfico en una ciudad. En una ciudad, el servicio podría estar implementado de forma que requiera el nombre de la calle como parámetro de entrada, mientras que en otra ciudad, un servicio similar podría requerir como parámetro de entrada una coordenada GPS. En estos casos, una ontología que describiera el tipo de parámetros podría hacer que el terminal aportara

directamente o derivara de otros datos el valor más conveniente en cada momento. Por su parte, el uso de invocaciones dinámicas resulta muy importante ya que el terminal normalmente no conoce la interfaz de los servicios que se ofertan en la red que visita.

Por las características técnicas de la tecnología inalámbrica utilizada, Bluetooth, el sistema empleado en esta Tesis está enfocado para su uso en entornos urbanos, entendiendo como tal el uso por parte de usuarios desplazándose a pie o en un vehículo a las velocidades permitidas en vías urbanas. En el caso particular de terminales embarcados en vehículos circulando por vías interurbanas, donde la velocidad es mayor, la tecnología Bluetooth puede verse muy limitada y es necesario optar por otro tipo de tecnología inalámbrica con un acceso rápido y una mayor área de cobertura. En este sentido, podría ser interesante adaptar el sistema para usar alguna de las tecnologías expuestas en el capítulo 2, como IEEE 802.11p.

En cualquier caso, la tecnología Bluetooth continúa evolucionando y muy recientemente ha aparecido una variante dentro de la especificación denominada "Bluetooth Low Energy". Entre las nuevas características que aporta "Bluetooth Low Energy" destacan su bajo consumo y la aparición de canales de anuncio (advertisement). Esto último puede resultar muy interesante sobre todo de cara a la detección de terminales, ya que se agiliza notablemente el tiempo de detección. Por otra parte, la característica de bajo consumo también puede contribuir a aumentar la autonomía de los terminales.

ANEXOS

A1. Desarrollos hardware

Baliza principal

La baliza principal es un sistema basado en una tarjeta comercial PC/104 cuyas principales características técnicas se detallan a continuación:

- Procesador AMD® Geode LX800 a 500MHz
- 512 MB de memoria RAM DDR a 333MHz
- Tarjeta Compact Flash de 4GB
- Puerto Ethernet 10/100Base-T
- 4 puertos USB 2.0 Host
- Códec de audio tipo AC97

Esta tarjeta PC/104 está montada sobre una tarjeta base de diseño propio que le proporciona las tensiones de alimentación a través de los correspondientes circuitos de conversión de tensión. Además, la tarjeta base alberga los conectores para la comunicación a través de Ethernet y USB, líneas de entrada/salida de propósito general, e incluye un circuito de amplificación de audio conectado a un altavoz para permitir la emisión de señales acústicas. Conectado a uno de los puertos USB se encuentra un "dongle" Bluetooth que proporciona conectividad con los terminales. El hecho de utilizar una interfaz USB y "dongles" ofrece una mayor flexibilidad a las aplicaciones, ya que se puede utilizar prácticamente cualquier dispositivo Bluetooth sean cuales sean sus características. Esto permite, por ejemplo, limitar el alcance de la baliza usando "dongles" de clase 1, 2 ó 3, según el caso. En la figura 46 se muestra una imagen de este tipo de balizas y en la figura 47 el diagrama de bloques correspondiente.

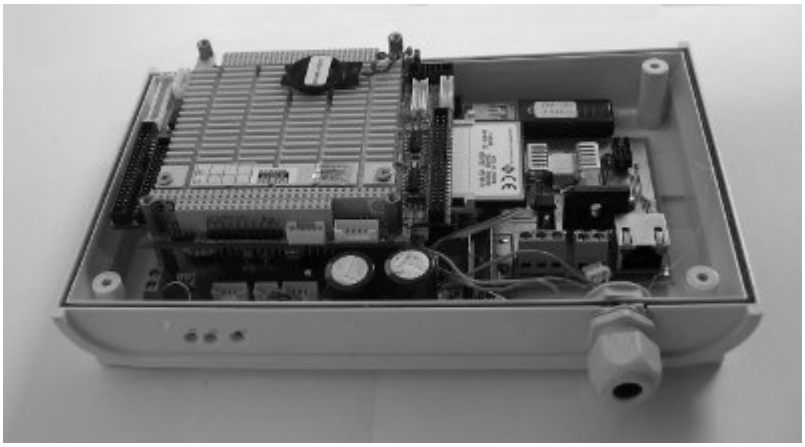

Figura 46: Imagen de la baliza principal

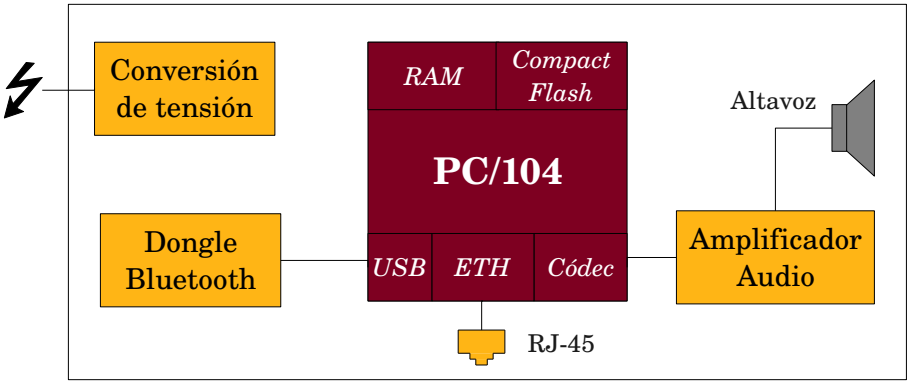

Figura 47: Diagrama de bloques de la baliza principal

Baliza de localización

La baliza de localización es un sistema que, aunque comparte algunos objetivos con la baliza principal, tiene una funcionalidad muy reducida. Como su propio nombre indica, estas balizas se utilizan para localizar ubicaciones. La baliza de localización se ha desarrollado como un diseño propio para el tipo de sistemas planteado en esta Tesis. Por su sencillez, la baliza de localización está gestionada por un microcontrolador Atmega8 de Atmel®. Las características fundamentalmentes de este microcontrolador se detallan a continuación:

- Núcleo AVR de 8 bits a 16MHz
- 1KB de memoria SRAM interna
- 8KB de memoria Flash para programas
- 512B de memoria EEPROM para datos
- Puerto serie (USART)
- Bajo consumo (aprox. 3.6mA @ 5V, 4MHz)

Como elemento de conectividad a redes Bluetooth, la baliza de localización utiliza un módulo WT12 de Bluegiga®. Se trata de un módulo Bluetooth de clase 2 acorde a la versión 2.1 (+EDR) de la especificación. El módulo dispone de dos interfaces para su control: USB y serie. Aparte de las funciones propias de un dongle Bluetooth, este módulo dispone de una implementación propia de la pila de protocolos a alto nivel (por encima de la capa HCI), lo que permite el establecimiento de conexiones L2CAP mediante comandos específicos sobre la interfaz serie. Esto evita la necesidad de implementar determinados niveles de la torre de protocolos en el software del microcontrolador Atmega, algo muy importante si se tienen en cuenta los limitados recursos de dicho microcontrolador.

El resto de los elementos que componen la baliza de localización están asociados al circuito de conversión de tensión, conformando un sistema totalmente autónomo. En la figura 48 se muestra una imagen de la baliza de localización y en la figura 49 el diagrama de bloques correspondiente.

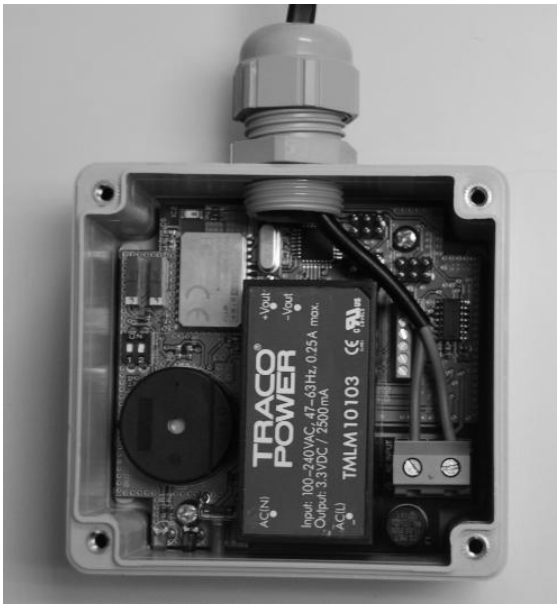

Figura 48: Imagen de la baliza de localización

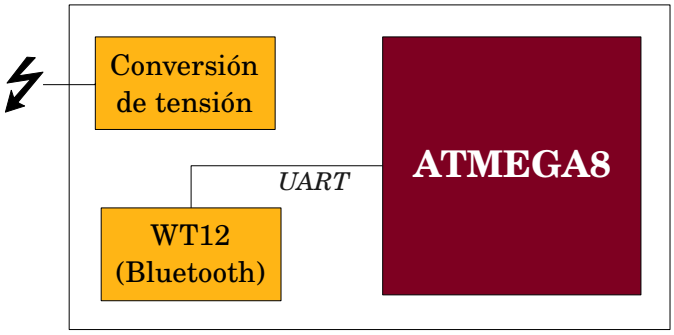

Figura 49: Diagrama de bloques de la baliza de localización

Terminal

Se ha realizado el diseño de un terminal de bolsillo específico para este tipo de aplicaciones. Se trata de un terminal autónomo alimentado por una batería recargable del tamaño aproximado de un teléfono móvil adaptado para invidentes, por lo que la información que emite es fundamentamente audiodescrita. Adicionalmente, el diseño incorpora algunos periféricos como una brújula electrónica, que aportan una información adicional que facilita particularmente las aplicaciones de orientación [89]. Las principales características técnicas del terminal se detallan a continuación:

- Procesador con núcleo ARM926EJ-S @ 266MHz
- Memoria SDRAM 64MB
- Memoria Flash de 32MB
- Salida de audio para auriculares
- Micrófono incorporado
- Ranura para tarjetas micro-SD
- Dispositivo Bluetooth intercambiable (dongle USB, preferentemente de clase 2)
- Puerto USB para la conexión a un PC
- Brújula electrónica de tres ejes
- Vibrador

En la figura 50 se muestra un diagrama de bloques del terminal diseñado.

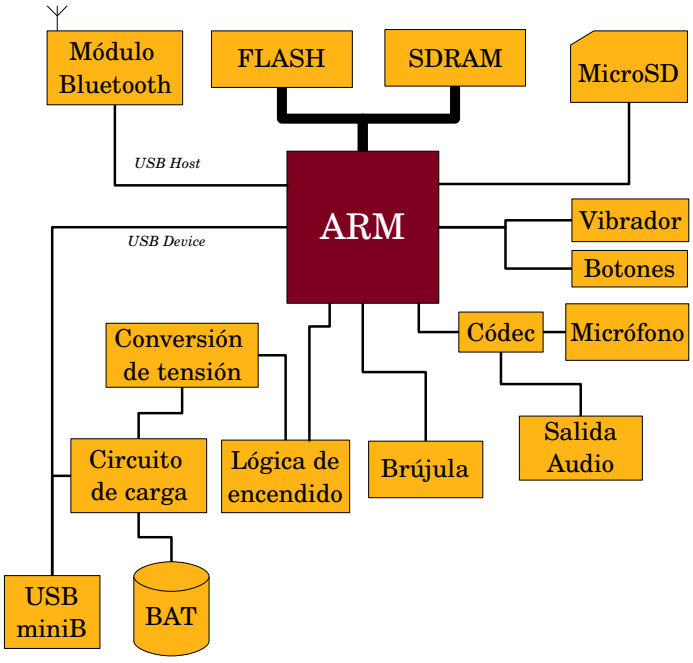

Figura 50: Diagrama de bloques del receptor

A fecha de escritura de esta Tesis, el terminal se encuentra en la fase previa a la fabricación de los prototipos. Como sistema operativo para el terminal se ha previsto adaptar e instalar una distribución específica de GNU/ Linux.

VisioWay OpenCounter

Los equipos VisioWay OpenCounter fundamentalmente extraen estadísticas de tráfico y, entre los parámetros que obtienen, está la ocupancia. La figura 51.a muestra una imagen del equipo y en la figura 51.b se muestra una imagen extraída del flujo de vídeo MPEG emitido por uno de estos equipos, donde se pueden apreciar las regiones de detección definidas.

(a) (b)

Figura 51: Equipo VisioWay OpenCounter e imagen capturada por el mismo

Las características más importantes del equipo se detallan a continuación:

- Microprocesador con núcleo ARM926EJ-S @ 266MHz
- Memoria SDRAM de 64MB
- Memoria Flash de 64MB
- Puerto Ethernet 10/100BaseT
- Ranura para tarjetas SD
- Entrada de vídeo
  - Analógica CVBS (vídeo compuesto) PAL/NTSC
  - Digital compatible con ITU-R BT.656
- Compresor MPEG-4 hardware.
  - Tamaño máximo: CIF (352x288)
  - Bitrate máximo: 384Kbps
  - Tasa de imágenes: hasta 30fps.
- Puerto USB OTG

Los equipos VisioWay OpenCounter utilizan una distribución específica de GNU/Linux como sistema operativo. Esta distribución está basada en las adaptaciones de GNU/Linux a procesadores ARM proporcionadas por una activa y exitosa comunidad de desarrolladores [90] . Esta comunidad desarrolla fundamentalmente drivers y mejoras sobre el kernel de Linux para dar soporte a los nuevos procesadores ARM y sus periféricos y es una referencia para muchas empresas que deciden apostar por el software libre en sus productos [91]. El hecho de utilizar GNU/Linux permite que el equipo

pueda soportar MICO y el desarrollo de las aplicaciones para estos equipos sea muy similar al de las balizas principales o los terminales.

A2. Método de medición de carga del servicio middleware

El método que se describe a continuación fue utilizaod para cuantificar el efecto de un servicio middleware bajo MICO sobre el sistema microprocesador de los equipos VisioWay OpenCounter. En estos equipos se ejecuta una aplicación de visión artifical que acapara la mayoría de los recursos del procesador. El método aprovecha la información disponible en los ficheros especiales ubicados en el directorio "/proc" para obtener datos sobre el tiempo de CPU asignado a los procesos por el planificador.

A primera vista, una solución podría ser medir la carga de CPU usando el comando "top". En algunos casos sencillos esto podría ser suficiente, pero en muchos otros podría distorsionar las medidas como consecuencia de las características de funcionamiento de "top" y de la carga de CPU que el propio comando introduce. Algunos ejemplos de estas situaciones se describen a continuación:

- A veces se necesita sincronizarse con el principio y el final del period de medida. Esto es muy importante en pruebas donde las peticiones de los clientes se simulan mediante ráfagas. Ya que "top" no puede sincronizarse con estas ráfagas, éste puede considerar periodos de media mayores o menores y, por consiguiente, puede falsear los resultados.
- Bajo ciertas versiones de GNU/Linux, el lanzamiento de hilos por parte de los servidores pueden mostrarse como nuevos procesos con sus propios PIDs y sus ficheros asociados en el directorio "/proc". Si el número de estos procesos asociados es alto puede provocar una mayor carga en el proceso "top", ya que tendrá que leer más archivos.
- Si la tasa de refresco de los datos es elevada o si hay muchos procesos siendo monitorizados, "top" puede demandar una cantidad significativa de tiempo de CPU y esto puede resultar en medidas poco realistas.

Cuando el uso de "top" no es posible debido a sus limitaciones, hay que recurrir a un método más ligero y selectivo. En este sentido, se ha desarrollado un método capaz de medir el tiempo de CPU asignado

únicamente al servidor independientemente del número de procesos asociados que hayan surgido por el lanzamiento de nuevos hilos. Para ello se asume que se cumplen las siguientes condiciones:

- Se da la situación descrita anteriormente respecto a la aparición de procesos adicionales.
- Las peticiones de los clientes llegan en forma de ráfagas.
- El periodo de medida es configurable y debe estar sincronizado con las ráfagas.

Para medir los efectos del servicio middleware sobre la aplicación principal del equipo VisioWay se debe comparar la carga media de CPU de dicha aplicación con y sin el servicio middleware activo. Las medidas sin servicio middleware pueden realizarse directamente con el comando "top" consultando el porcentaje de CPU que el planificador asigna a la aplicación principal. Para minimizar los indeseables efectos de carga de "top" la salida se puede restringir al proceso de la aplicación principal. La figura 52 muestra el tiempo de CPU medio y real acumulado asignado a la aplicación principal cuando no está activo el servicio middleware.

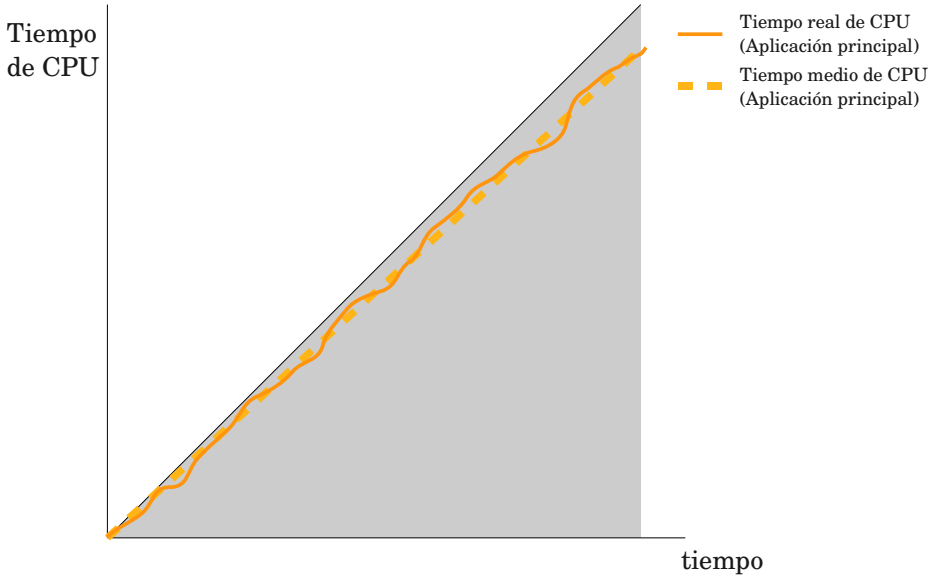

Figura 52: Tiempo medio y real acumulado de CPU sin servicio middleware

La aplicación principal normalmente acapara casi todos los recursos de

la CPU, por este motivo, la curva está muy cerca de la línea de tiempo total de CPU acumulado (línea en 45º). Cuanto mayor es la pendiente de la curva, mayor es el porcentaje de CPU usado.

En el caso de activar además el servicio middleware "top" no puede usarse debido a los problemas mencionados anteriormente. Ya que la información que muestra el proceso "top" puede obtenerse de los ficheros del directorio "/proc", se puede insertar un pequeño fragmento de código en el servidor middleware para leer dichos ficheros y extraer sólo la información relevante en determinados instantes. Cuando sólo hay unos pocos procesos a considerar, esta solución puede ser adecuada y sus efectos sobre la medida, en términos de carga de CPU extra, son insignificantes. Por ejemplo, si tanto la aplicación principal como el servidor middleware son ejecutados por dos procesos únicamente, le código de medición sólo abriría tres ficheros: el correspondiente a la aplicación principal, el correspondiente al servicio middleware y el que contiene el tiempo total de CPU. Desafortunadamente, esta situación no siempre se da. En el caso concreto del servicio analizado, se generan varios procesos asociados, lo que aumenta significativamente el tiempo para leer los datos de todos los PIDs asociados. Experimentos sobre los equipos VisioWay mostraron que esta operación puede llevar algunos segundos, algo inaceptable.

Para solucionar el problema se aprovechó una característica observada en las aplicaciones servidoras que hacen uso de MICO como ORB; cuando el servidor no está recibiendo peticiones de clientes prácticamente no consume tiempo de CPU, por eso se puede asumir que su curva de tiempo de CPU es plana en estos intervalos, tal y como se muestra en la figura 53 (línea roja clara). Ya que los valores contenidos en los ficheros asociados al servidor en el directorio "/proc" no cambian durante los periodos de inactividad (antes y después de la ráfaga), se pueden tomar medias dentro de estos periodos (t_{m1}, t_{m2}), a pesar de sus grandes retrasos, obteniéndose los mismos valores que los que se habrían conseguido si las mediciones se hubiesen efectuado justo al principio y al final de la ráfaga.

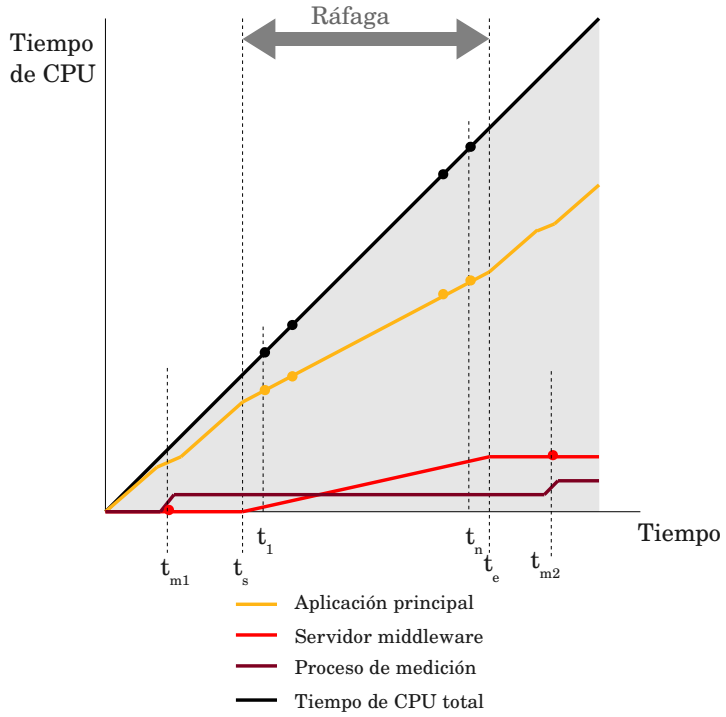

Figura 53: Tiempo medio y real acumulado de CPU con servicio middleware

Por tanto, estas medidas pueden tomarse en cualquier instante antes y después de la ráfaga por una aplicación sencilla. Esta aplicación tiene que obtener una lista de los PIDs relacionados con el servidor primero y después obtener el tiempo de CPU acumulado para cada fichero asociado a uno de los PIDs. El tiempo total de CPU acumulado es la suma de todos esos valores. Esta tarea puede llevarse a cabo con un script para "bash*".

Para calcular la carga de la aplicación principal sólo se necesitan tomar medidas al principio y al final de la ráfaga. Esto puede hacerse insertando en la rutina del servicio un breve fragmento de código que recoja esa información de los ficheros del directorio "/proc" relacionados con la aplicación principal y el sistema cuando el servidor recibe la primera y la última petición. Pero esta solución plantea dos problemas:

1. Cómo determinar la primera y la última petición
2. Incluso si el código es capaz de determinar lo anterior, el proceso

* bash (Bourne-Again SHell) es uno de los intérpretes de órdenes disponibles en sistemas UNIX y derivados.

servidor puede estar funcionando antes y después de ejecutar el código de medición y, por tanto, consumiendo un poco más de tiempo del estimado.

Una manera directa de solucionar el primer problema consiste en marcar la primera y la última petición desde el lado del cliente, pero esto puede fallar en algunas circunstancias. Concretamente, cuando la tasa de peticiones (peticiones/minuto) es muy alta y el servidor lanza varios hilos. El planificador en el lado del servidor podría provocar un cambio de orden en la atención de las peticiones. Una solución sencilla pasaría por marcar las primeras y las últimas *n* peticiones, siendo *n* un número pequeño. Tras la finalización de la ráfaga se retendrían únicamente los valores más bajos y los más altos.

El segundo problema es difícil de evitar, pero puede ser mitigado si la ráfaga es lo suficientemente larga. Dado que lo que se pretende es obtener un comportamiento medio, se puede hacer la ráfaga tan larga como se necesite siempre que la tasa media de peticiones se mantenga. La expresión (16) muestra esto desde un punto de vista matemático. La carga media de CPU de la aplicación principal (en tanto por ciento) está definida como:

$$L_{MAIN} = 100 \cdot \left(\frac{TCPU_{MAIN}(t_e) - TCPU_{MAIN}(t_s)}{TCPU_{SYS}(t_e) - TCPU_{SYS}(t_s)} \right) = 100 \cdot \left(\frac{TCPU_{MAIN}(t_e) - TCPU_{MAIN}(t_s)}{t_e - t_s} \right) \quad (16)$$

Asumiendo t1 y tn como los tiempos de la primera y la última medición, respectivamente, la carga media de CPU de la aplicación principal según el método propuesto es:

$$L'_{MAIN} = 100 \cdot \left(\frac{TCPU_{MAIN}(t_n) - TCPU_{MAIN}(t_1)}{TCPU_{SYS}(t_n) - TCPU_{SYS}(t_1)} \right) = 100 \cdot \left(\frac{TCPU_{MAIN}(t_n) - TCPU_{MAIN}(t_1)}{t_n - t_1} \right) \quad (17)$$

Las diferencias entre los tiempos de medida esperados y los reales son:

$$\Delta t_s = t_1 - t_s ; \Delta t_e = t_e - t_n \quad (18)$$

Los errores en las mediciones debidos al problema explicado anteriormente pueden ser defindos como:

$$\begin{aligned} e_{Ms} &= TCPU_{MAIN}(t_1) - TCPU_{MAIN}(t_s) \\ e_{Me} &= TCPU_{MAIN}(t_n) - TCPU_{MAIN}(t_e) \end{aligned} \quad (19)$$

Sustituyendo (18) y (19) en (17) se obtiene:

$$L'_{MAIN} = 100 \cdot \left(\frac{TCPU_{MAIN}(t_e) + e_{Me} - TCPU_{MAIN}(t_s) - e_{Ms}}{t_e - t_s - (\Delta t_s + \Delta t_e)} \right) \quad (20)$$

Δt_e y Δt_s no dependen de la longitud de la ráfaga, por lo que si ésta es lo suficientemente larga se puede asumir que:

$$t_e - t_s \gg \Delta t_s + \Delta t_e \quad (21)$$

Y esto implica que:

$$L'_{MAIN} \approx 100 \cdot \left(\frac{TCPU_{MAIN}(t_e) + e_{Me} - TCPU_{MAIN}(t_s) - e_{Ms}}{t_e - t_s} \right) \quad (22)$$

Ya que la carga de CPU de cualquier proceso en ejecución está siempre por debajo del 100%, e_{Me} and e_{Ms} son menores o iguales que Δt_e y Δt_s, respectivamente. Estos errores son también independientes de la longitud de la ráfaga, por lo que se puede asumir que:

$$L'_{MAIN} = 100 \cdot \left(\frac{TCPU_{MAIN}(t_e) - TCPU_{MAIN}(t_s)}{t_e - t_s} + \frac{e_{Me} - e_{Ms}}{t_e - t_s} \right) \quad (23)$$
$$t_e - t_s \gg e_{Me} - e_{Ms}$$

Por lo tanto,

$$L'_{MAIN} \approx 100 \cdot \left(\frac{TCPU_{MAIN}(t_e) - TCPU_{MAIN}(t_s)}{t_e - t_s} \right) = L_{MAIN} \quad (24)$$

La carga de CPU del servidor middleware puede estimarse usando la siguiente expresión:

$$L_{SRV} = 100 \cdot \left(\frac{TCPU_{SRV}(t_e) - TCPU_{SRV}(t_s)}{TCPU_{SYS}(t_e) - TCPU_{SYS}(t_s)} \right) = 100 \cdot \left(\frac{TCPU_{SRV}(t_{m2}) - TCPU_{SRV}(t_{m1})}{t_e - t_s} \right) \quad (25)$$

Junto con las medidas de carga de la CPU se puede extraer también información útil, como el consumo de memoria, de los ficheros del directorio "/proc" sin que suponga una carga adicional significativa.

Para obtener una estimación más completa, el experimento debería repetirse varias veces cambiando la tasa de peticiones. Esto ayuda a entender los efectos de los servicios middleware sobre el proceso principal bajo diferentes condiciones. Dependiendo de la funcionalidad de la aplicación principal, la reducción de tiempo de CPU puede llegar a tener un impacto altamente negativo en su rendimiento.

A3. Funciones de densidad de probabilidad para el tiempo de detección

En las siguientes gráficas se muestra la función de densidad de probabilidad para el tiempo de detección en función de las diferentes combinaciones de reparto, desde 1:1 hasta 3:3. En el eje de ordenadas se representa la densidad de probabilidad normalizada y en el eje de abcisas se representa el tiempo en segundos. Todas las gráficas han sido obtenidas por simulación.

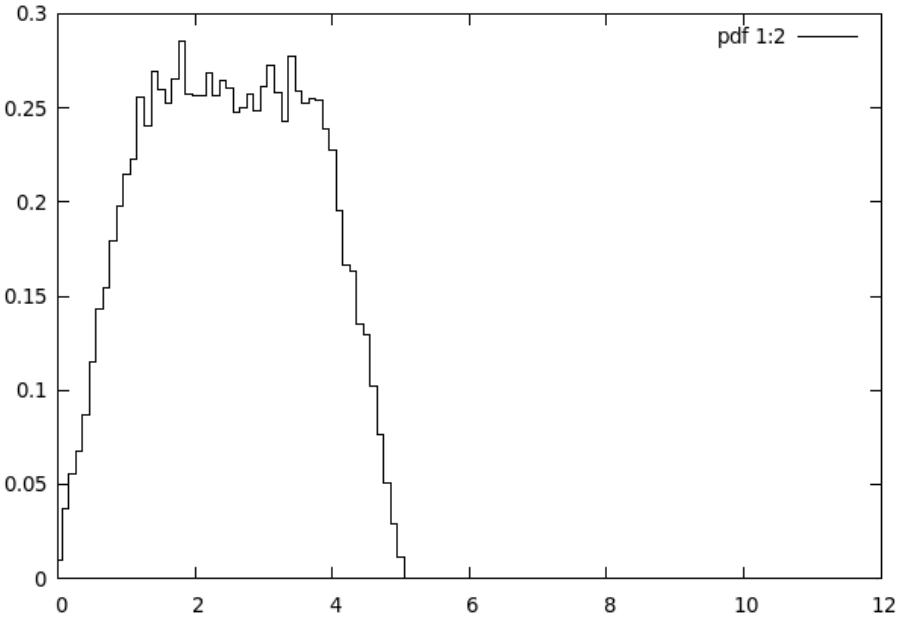

Figura 54: Función de densidad de probabilidad para un reparto 1:2

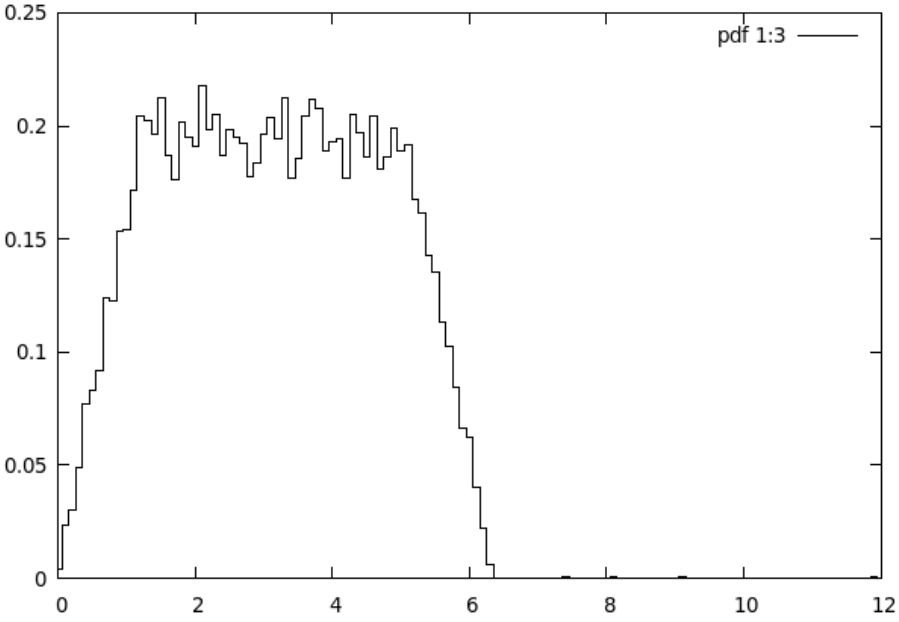

Figura 55: Función de densidad de probabilidad para un reparto 1:3

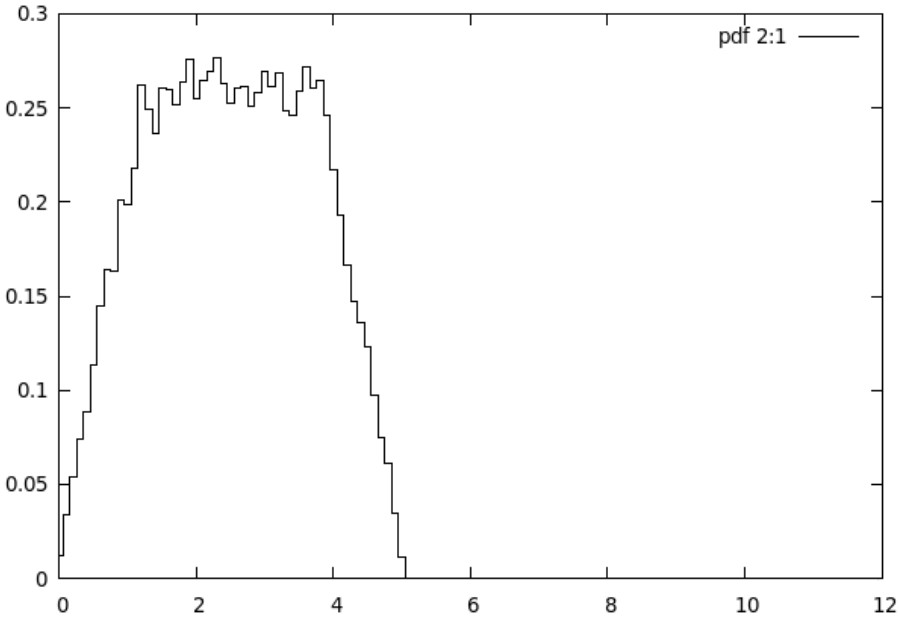

Figura 56: Función de densidad de probabilidad para un reparto 2:1

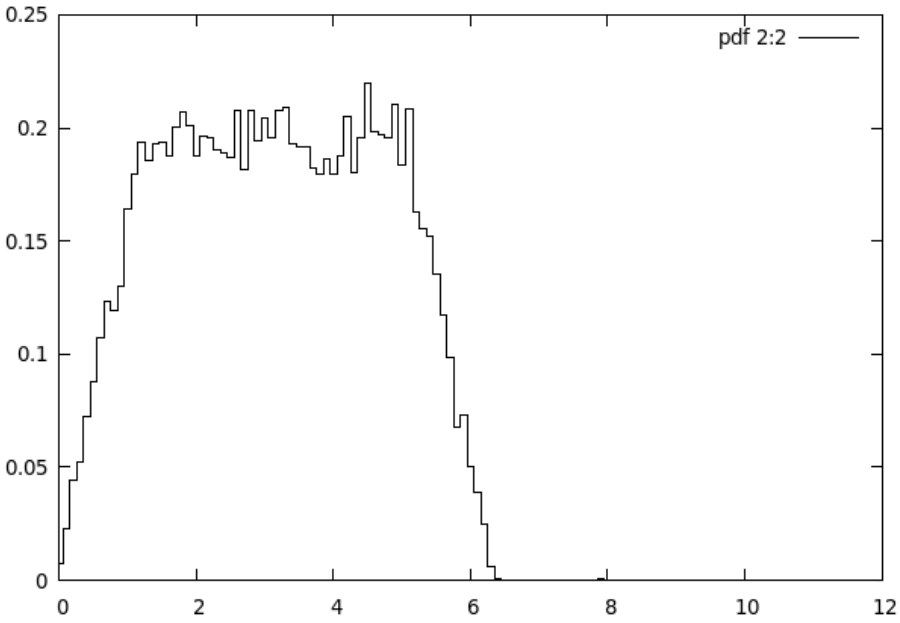

Figura 57: Función de densidad de probabilidad para un reparto 2:2

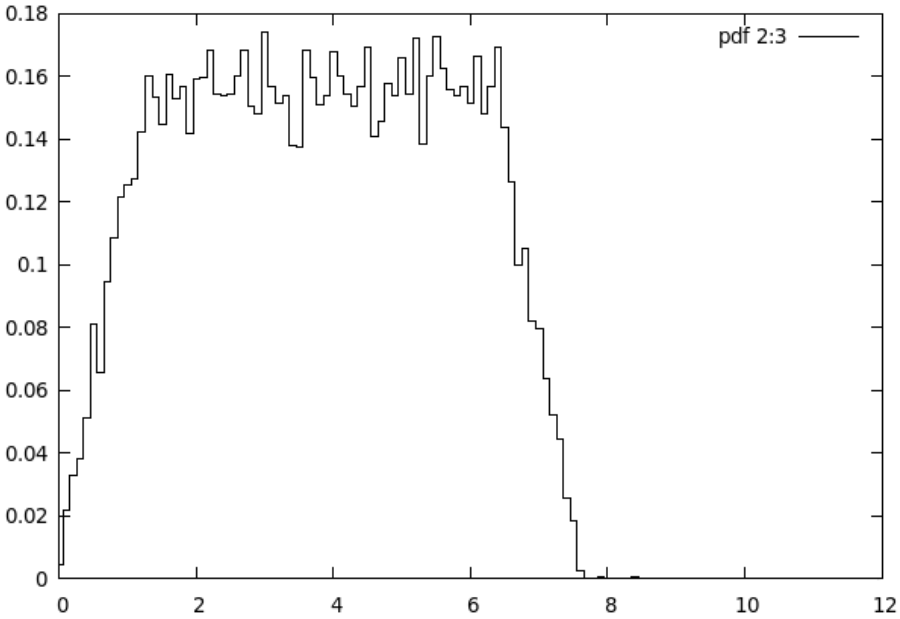

Figura 58: Función de densidad de probabilidad para un reparto 2:3

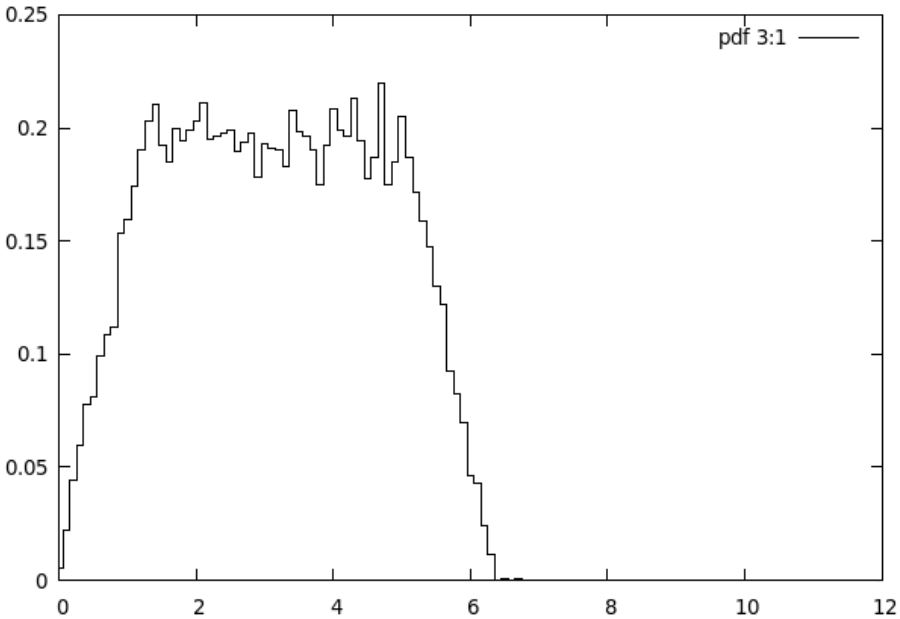

Figura 59: Función de densidad de probabilidad para un reparto 3:1

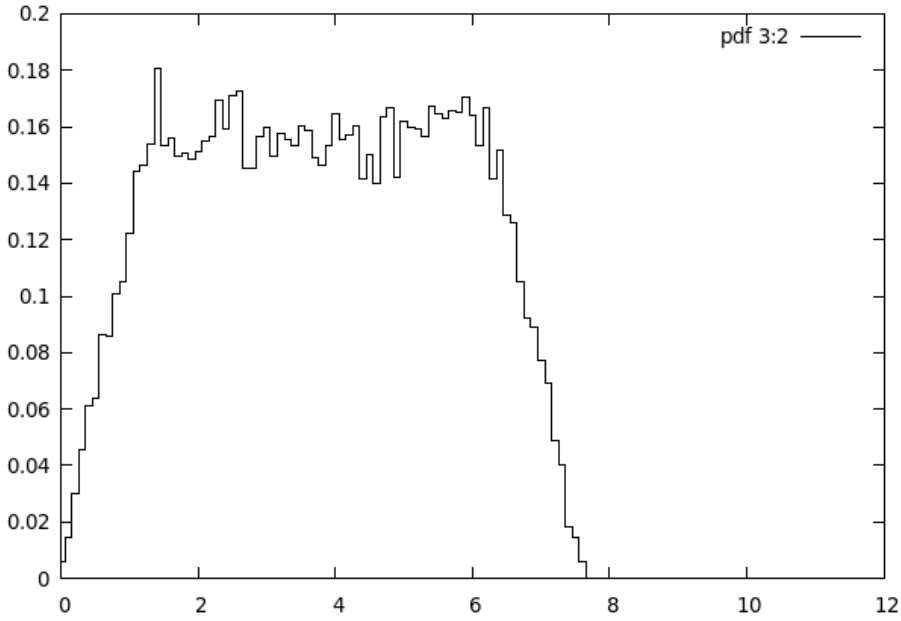

Figura 60: Función de densidad de probabilidad para un reparto 3:2

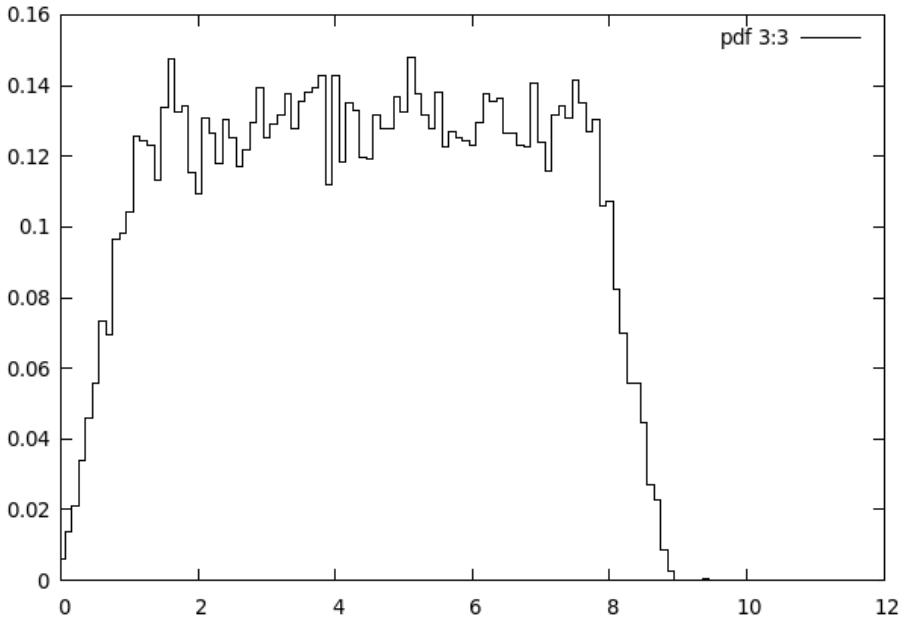

Figura 61: Función de densidad de probabilidad para un reparto 3:3

A4. Opciones de uso de los gestores de conexión

Uso de los gestores de conexión

nccserver

Descripción:

nccserver es el módulo servidor que permite a un terminal ser detectado a través de la interfaz Bluetooth. El servidor permanece a la espera de conexiones entrantes de las balizas que detectan el terminal y emite notificaciones a TBCM. En muchos sistemas este proceso debe ejecutarse con privilegios de superusuario.

Sintaxis:

```
nccserver [-a ADDR#PSM] [ORB_opts]
```

-a ADDR#PSM

Indica la dirección Bluetooth del dispositivo a utilizar y el puerto L2CAP sobre el que escuchar. La dirección debe expresarse en parejas de dígitos hexadecimales separadas por dos puntos ':'. El caracter '#' se utiliza para separar la dirección del puerto L2CAP. Si no se especifica esta opción, se utiliza el disposivo Bluetooth por defecto (hci0).

ORB_opts

Opciones particulares del ORB. Para más información consultar las opciones permitidas por MICO.

abcmd

Descripción:

abcmd es el gestor de conexiones para AccessBridge (ABCM) y se ejecuta en el lado de la baliza. Entre sus principales funciones está el control de las conexiones de AccessBridge y la detección de terminales.

Sintaxis:

```
abcmd [-dev hci] -srvf sfile -ctxf cfile -brname abname -inqt i -scc sc [ORB_opts]
```

-dev hci

Indica el dispositivo Bluetooth a utilizar, identificado por su descriptor en el sistema operativo (hci0, hci1, etc.). El listado de dichos dispositivos puede obtenerse con el comando "hciconfig". Si no se especifica esta opción, se utiliza el disposivo Bluetooth por defecto (hci0).

-srvf sfile

Especifica la ruta al fichero de referencias a servicios. El formato del fichero se describe en el apartado "Fichero de referencias a servicios".

-ctxf cfile

Especifica la ruta al fichero de contexto. El formato del fichero se describe en el apartado "Fichero de descripción de contexto".

-brname abname

Indica el nombre asociado a AccessBridge tal y como aparece publicado en el servicio de nombres.

-inqt i

Indica el número de bloques de 1,28 segundos dedicados al proceso de inquiry en cada ciclo.

-scc sc

Indica el número de bloques de 1,28 segundos que dura el ciclo de inquiry completo (inquiry + datos).

ORB_opts

Opciones particulares del ORB. Para más información consultar las opciones permitidas por MICO.

tbcmd

Descripción:

tbcmd es el gestor de conexiones para TerminalBridge (TBCM) y se ejecuta en el lado del terminal. Entre sus principales funciones está el control del establecimiento y liberación de conexiones de TerminalBridge y la notificación de eventos relacionados con las redes a las aplicaciones clientes del terminal.

Sintaxis:

```
tbcmd [ORB_opts]
```

ORB_opts

Opciones particulares del ORB. Para más información consultar las opciones permitidas por MICO.

Orden de lanzamiento de los gestores

Para una correcta operación de los gestores de conexión, los siguientes elementos deben lanzarse exactamente en el orden especificado a continuación.

En el lado de la baliza:

1) Servicio de nombres de CORBA
2) Servicio de eventos de CORBA
3) AccessBridge (ab)
4) Servidores particulares
5) Gestor de conexiones abcmd

En el lado del terminal:

1) Servicio de nombres de CORBA
2) Servicio de eventos de CORBA
3) TerminalBridge (tb)
4) Gestor de conexiones tbcmd
5) Servidor NCC (nccserver)
6) Clientes particulares

Formato de los ficheros adicionales

Fichero de referencias a servicios

Los ficheros de referencias a servicios son archivos de texto plano que definen los servicios cuyas referencias serán enviadas a través del protocolo NCC cuando la baliza detecte a un terminal. Cada línea del archivo define un servicio, que puede ser identificado de dos formas:

a) Únicamente por su nombre. Si el servicio ha publicado previamente su referencia en el servicio de nombres, se puede indicar únicamente el nombre de éste. ABCM buscará automáticamente la referencia IOR correspondiente al servicio en el servidor de nombres.
b) Por su nombre y referencia IOR. En este caso se debe especificar el nombre del servicio y su referencia IOR en texto plano, separada del nombre por un tabulador.

A continuación se muestra un ejemplo de archivo de referencias a servicios:

```
Servicio1
Servicio2	IOR:010000001400000049444c3a73796e...
Servicio3	IOR:010000001400000049882daf2289e3...
Servicio4
```

Fichero de descripción de contexto

Los ficheros de descripción de contexto son archivos de texto plano que describen el contexto jerarquizado en el que se encuentra la baliza. Cada línea del archivo representa un nivel de jerarquía inferior al anterior y contiene el nombre del contexto a ese nivel. Por ejemplo: una baliza instalada en el departamento de Ingeniería Electrónica de la Universidad de Sevilla podría presentar el siguiente fichero describiendo un contexto de tres niveles:

```
Universidad de Sevilla
Escuela Superior de Ingenieros
Departamento de Ingeniería Electrónica
```

Bibliografía

[1] Diane Cook, Sajal Das. *Smart environments: Technology, protocols and applications*. Wiley , 2004

[2] Wei-Hsun Lee, Shian-Shyong Tseng, Wern-Yarng Shieh. Collaborative real-time traffic information generation and sharing framework for the intelligent transportation system. *Information sciences*, Vol. 180, num. 1, pp. 62 - 70, 2010

[3] Centro Nacional de Referencia de Aplicación de las TIC basadas en Fuentes Abeirtas. *Informe sobre la situación del software de fuentes abiertas en las empresas españolas proveedoras e integradoras de sistemas empotrados*. 2010

[4] Plataforma tecnológica española de sistemas con inteligencia integrada. *PROMETEO website*. 2009 (http://www.prometeo-office.org)

[5] ARTEMIS strategic research agenda working group. *Strategic research agenda*. 2006

[6] Derek Chapman. Government, railway regulation and train control. *Course on railway signalling and control systems (RSCS 2010)*. RSCS 2010, pp. 21 - 29, 2010

[7] D. Jiang, L. Delgrossi. IEEE 802.11p: Towards an international standard for wireless access in vehicular environments. *Proceedings of the vehicular technology conference 2008*. VTC 2008, pp. 2036 - 2040, 2008

[8] The CALM Forum. *The CALM Handbook*. 2006

[9] SAFESPOT. *SAFESPOT*. 2010 (http://www.safespot-eu.org/)

[10] CVIS. *CVIS*. 2010 (http://www.cvisproject.org/)

[11] COOPERS. *COOPERS*. 2010 (http://www.coopers-ip.eu/)

[12] Jaime García Reinoso. *Contribución al desarrollo de aplicaciones alternativas de Bluetooth: localización de usarios y telemando*. 2003

[13] Jason Stewart, Jakob Hilden, Michelle Escobar, Kumud Bihani, Sara Baumann, Mark Newman. Contextual information system for urban orientation of sighted and non-sighted users. *Proceedings of the Accesible design in the digital world conference*. ADDW08, pp. , 2008

[14] Talking points. *Talking-Points project webpage*. 2009 (http://talking-points.org)

[15] NEC, KDDI. *Pedestrian navigation with InfoSign*. 2009 (http://www.nec.co.jp/techrep/en/journal/g08/n01/080110.html)

[16] S. Bohonos, A. Lee, A. Malik, C. Thai, R. Manduchi. Universal real-time navigational assistance (URNA): an urban bluetooth beacon for the blind. *Proceedings of the 1st ACM SIGMOBILE International workshop on systems and networking support for healthcare and assisted living environments*.

HealthNet 2007, pp. 83 - 88, 2007
[17] URNA project. *URNA - Universal Real-time Navigational Assistance*. 2009 ()
[18] B. Tjan, P. Beckmann, R. Roy, N. Giudice, G. Legge. Digital Sign System for indoor wayfinding for the visually impaired. *Proceedings of the conference on computer vision and pattern recognition 2005 (CVPR'05)*. IEEE computer society, pp. 30, 2005
19: Andy Harter, Andy Hopper, Pete Steggles, Andy Ward, Paul Webster, The anatomy of a context-aware application, 2002
[20] Adam Smith, Hari Balakrishnan, Michel Goraczko, Nissanka Priyantha. Tracking moving devices with the CRICKET location system. *Proceedings of the 2nd international conference on mobile systems, applications and services (MobiSys2004)*. MobiSys2004, pp. , 2004
[21] Y. Fukuju, M. Minami, H. Morikawa, T. Aoyama. DOLPHIN: an autonomous indoor positioning system in ubiquitous computing environment. *Workshop on Software technologies for future embedded systems (WSTFES2003)*. IEEE, pp. 53, 2003
[22] J. Hallberg, M. Nilsson, K. Synnes. Positioning with Bluetooth. *Proceedings of the 10th international conference on telecommunications*. ICT 2003, pp. 954 - 958, 2003
[23] Raffaele Bruno, Franca Delmastro. Design and analysis of a bluetooth-based indoor localization system. *Proceedings of the international conference on personal wireless communications*. PWC 2003, pp. 711 - 725, 2003
[24] J. Hallberg, M. Nilsson, K. Synnes. Bluetooth positioning. *Proceedings of the 3rd symposium on computer science and electrical engineering*. CSEE 2002, pp. 1 - 5, 2002
[25] Tiago Fernández, Javier Rodas, Carlos Escudero, Daniel Iglesia. Bluetooth sensor network positioning system with dynamic calibration. *Proceedings of the 4th international symposium on wireless communication systems*. ISWCS 2007, pp. 45 - 49, 2007
[26] S. Zhou, J. Pollard. Position measurement using Bluetooth. *IEEE Transactions on consumer electronics*, Vol. 52, num. 2, pp. 555 - 558, 2006
[27] Albert S. Huang. *The use of Bluetooth in Linux and location aware computing*. 2005
[28] Antti Kotanen, Marko Hannikainen, Helena Leppakoski, Timo Hamalainen. Experiments on local positioning with Bluetooth. *Proceedings of the international conference on information technology: coding and computing*. ITCC2003, pp. 297 - 303, 2003
[29] Alessandro Genco. Three step Bluetooth positioning. *Location- and context-awareness*. LoCA 2005, pp. 52 - 62, 2005
[30] A. Hossain, W. Soh. A comprehensive study of Bluetooth signal

parameters for localization. *Proceedings of the 18th international symposium on personal, indoor and mobile radio communications*. PIMRC 2007, pp. , 2007

[31] Gunter Fischer, Burkhart Dietrich, Frank Winkler. Bluetooth indoor localization system. *Proceedings of the 1st workshop on positioning, navigation and communication*. WPNC'04, pp. 147 - 156, 2004

[32] Sami Saalasti, Juha Jääskeläinen, Ari Valtaoja. Terminal bridge extension over distributed architecture. . Workshop on Applications of Wireless Communications (WAWC'04), pp. , 2004

[33] Ural Mutlu, Reuben Edwards, Paul Colton. QoS aware CORBA middleware for Bluetooth. *Proceedings of the 10th international symposium on consumer electronics*. ISCE'06, pp. 1 - 7, 2006

[34] Xinyu Chen, Michael Lyu. Message logging and recovery in wireless CORBA using access bridge. *Proceedings of the 6th international symposium on autonomous decentralized systems*. ISADS 2003, pp. 107 - 114, 2003

[35] Luiz Lima, Alcides Calsavara. A framework for CORBA interoperability in ad hoc networks. *Proceedings of the 2007 ACM symposium on applied computing*. SAC'07, pp. 930 - 934, 2007

[36] Mayank Mishra. Architecture for locating mobile CORBA objects. *Advanced distributed systems*. ISSADS 2004, pp. 207 - 218, 2004

[37] Domenico Cotroneo, Armando Migliaccio, Stefano Russo. The Esperanto broker: a communication platform for nomadic computing systems. *Software - Practice and experience*, Vol. 37, num. 10, pp. 1017 - 1046, 2007

[38] Amy Murphy, Gian Pietro Picco, Gruia-Catalin Roman . LIME: a middleware for physical and logical mobility. *Proceedings of the 21st international conference on distributed computing systems*. ICDSC 2001, pp. 524 - 533, 2001

[39] M. Cinque, D. Cotroneo, C. di Flora, A. Migliaccio, S. Russo. ESPERANTO: a middleware platform to achieve interoperability in nomadic computing domains. *Proceedings of the 3rd international conference on computer systems and applications*. AICCSA 2005, pp. 106 - 114, 2005

[40] Domenico Cotroneo, Armando Migliaccio, Stefano Russo. A communication broker for nomadic computing systems. *High performance computing and communications*. HPCC 2005, pp. 1011 - 1020, 2005

[41] José Cano, Juan Cano, Pietro Manzoni, David Ferrández. On the design of spontaneous networks using P2P approach and Bluetooth. *Proceedings of the 10th symposium on computers and communications*. ISCC 2005, pp. 125 - 130, 2005

[42] Sewook Jung, Uichin Lee, Alexander Chang, Dae-Ki Cho, Mario Gerla. BlueTorrent: Cooperative content sharing for Bluetooth users. *Proceedings of the 5th international conference on pervasive computing and communications*.

PERCOM2007, pp. 47 - 56, 2007
[43] C. Boldrini, M. Conti, F. Delmastro, A. Passarella. Context- and social-aware middleware for opportunistic networks. *Network & computer applications*, Vol. 33, num. 5, pp. 525 - 541, 2010
[44] Olof Rensfelt. *LUNAR over Bluetooth*. 2004
[45] M. Carmen Domingo Aladrén. *Diferenciación de servicios y mejora de la supervivencia en redes ad hoc conectadas a redes fijas*. 2005
[46] Miguel A. Ortuño Pérez. *Protocolo de encaminamiento en origen con identificadores no únicos para redes ad-hoc de dispositivos con recursos limitados*. 2006
[47] Thor Egil Skaug. *A middleware framework providing adaptive quality of service for Bluetooth*. 2004
[48] Jelena Misic, Vojislav B. Misic. *Performance modeling and analysis of Bluetooth networks: Polling, scheduling and traffic control* . Taylor & Francis Group, 2006
[49] Veselin Rakocevic, Muttukrishnan Rajarajan, Kerry-Ann McCalla, Charbel Boumitri. QoS constraints in Bluetooth-based wireless sensor networks. *Proceedings of the 5th international workshop on quality of future Internet services*. QofIS 2004, pp. 214 - 223, 2004
[50] Alvin Chan, Siu-Nam Chuang. MobiPADS: A reflective middleware for context-aware mobile computing. *IEEE Transactions on software engineering*, Vol. 29, num. 12, pp. , 2003
[51] Licia Capra, Wolfgang Emmerich, Cecilia Mascolo. CARISMA: Context-aware reflective middleware system for mobile applications. *IEEE Transactions on software engineering*, Vol. 29, num. 10, pp. 929 - 944, 2003
[52] A. Mody, M. Akram, K. Rony, M. Sajjad, R. Kamoua. Enhancing user experience at museums using smart phones with RFID. . Systems, applications and technology conference (LISAT'09), pp. 1 - 5, 2009
[53] Sebastian Grafling, Petri Mahonen, Janne Riihijarvi. Performance evaluation of IEEE 1609 WAVE and IEEE 802.11p for vehicular communications. *Proceedings of the 2nd international conference on ubiquitous and future networks*. ICUFN 2010, pp. 344 -348, 2010
[54] Thomas King, Thomas Haenselmann, Stephan Kopf, Wolfgang Effelsberg. A measurement study on 802.11 concurrently used for positioning and communications. . International conference on wireless pervasive computing 2008, pp. 610 - 615, 2008
[55] Hyun-Sung Park, Seung-Chur Yang, Doo-Hwan Oh, Jong-Deok Kim. Design and implementation of WLAN-based automatic vehicle identification. . International Conference on Computational Science and Engineering, pp. 310 - 317, 2009
[56] Todd Arnold, Wyatt Lloyd, Jing Zhao, Guohong Cao. IP address passing

for VANETs. . International conference on pervasive computing, pp. 70 - 79, 2008
[57] Stephan Eichler. Performance evaluation of the IEEE 802.11p WAVE communication standard. *Proceedings of the 66th Vehicular technology conference*. VTC 2007, pp. 2199 - 2203, 2007
[58] IETF. *Mobility Support in IPv6*. 2004 ()
[59] IETF. *IP Mobility Support for IPv4*. 2002 ()
60: S. Ding, Mobile IP handoffs among multiple internet gateways in mobile ad hoc networks, 2009
[61] ZigBee Alliance. *ZigBee Alliance website*. 2010 ()
[62] Drew Gislason. *ZigBee Wireless Networking*. Newnes, 2008
[63] Bluetooth SIG. *Bluetooth specification Version 4.0*. 2009
[64] Jin-Shyan Lee, Yu-Wei Su, Chung-Chou Shen. A comparative study of wireless protocols: Bluetooth, UWB, ZigBee, and Wi-Fi. . 33rd Annual conference of the IEEE industrial electronics society (IECON) , pp. 46 -51, 2007
[65] Bluetooth SIG. *Bluetooth Technical Comparison*. 2010 ()
[66] Jaako Kangasharju. *Implementing the Wireless CORBA Specification*. 2002
[67] Nokia. *The VIVIAN project*. 2010 ()
[68] OMG. *Wireless acess and terminal mobility in CORBA specification*. 2001
[69] OMG. *CORBA wireless access & terminal mobility: Bluetooth tunneling*. 2005
[70] Xinyu Chen, Michael Lyu. Reliability analysis for various communication schemes in Wireless CORBA. *IEEE Transactions on reliability*, Vol. 54, num. 2, pp. 232 - 242, 2005
[71] Brian S. Peterson, Rusty O. Baldwin, Jeffrey P. Kharoufeh. Bluetooth inquiry time characterization and selection. *IEEE Transactions on mobile computing*, Vol. , num. 9, pp. 1173 - 1187, 2006
[72] Gergely V. Záruba, Imrich Chlamtac. Accelerating Bluetooth inquiry for personal area networks. . IEEE Globecom, pp. 702 - 706, 2003
[73] Oliver Kasten, Marc Langheinrich. First experiences with Bluetooth in the smart-Its distributed sensor network. *n Workshop on Ubiquitous Computing and Communications*. PACT, pp. , 2001
[74] Charles D. Knutson, Derek Joos, Ryan Woodings. Infrared data communications in wireless personal area networks. *Proc. of the 5th World Multiconference onSystemics, Cybernetics and Informatics*. , pp. 427-431, 2001
[75] Jean Tourrilhes, Venky Kirshnan. Co-link configuration: using wireless diversity for more than just connectivity. *Wireless Communications and Networking (WCNC 2003)*. WCNC 2003, pp. 1090 - 1095, 2003
[76] Fintan Bolton. *Pure CORBA*. Sams Publishing, 2002

[77] Sergio Toral, Federico Barrero, Francisco J. Cortés, Daniel Gutiérrez, Esteban Marsal, José M. Hinojo, Mateo Soto. A wireless indoor system for assisting victims and rescue equipments in a disaster management. *Proceedings of the II International conference on intelligent networking and collaborative systems*. INCOS 2010, pp. 502 - 506, 2010
[78] Sergio Toral, Francisco J. Cortés, Federico Barrero, Derlis Gregor. Development of a Bluetooth CORBA based intelligent infrastructure for multi-user urban environments. *Proceedings of the II International conference on intelligent networking and collaborative systems*. INCOS 2010, pp. 132 - 136, 2010
[79] VisioWay. *Página web de los productos VisioWay*. 2010 (http://www.visioway.com)
[80] OpenMoko. *Página del proyecto OpenMoko*. 2010 (http://www.openmoko.org)
[81] José M. Hinojo Montero, Daniel Gutiérrez Reina, Esteban Marsal, Mateo Soto Ramos, Francisco J. Cortés Martínez, Federico Barrero García, Sergio Toral Marín. Plataforma hardware para el aprendizaje de tecnologías inalámbricas y redes de sensores basada en el sistema open hardware denominado OpenMoko. *Actas del IX congreso de tecnologías aplicadas a la enseñanza de la electrónica (TAEE 2010)*. TAEE 2010, pp. , 2010
[82] Francisco J. Cortés ....(rellenar). Plataforma para el aprendizaje de tecnologías inalámbricas y redes de sensores basada en el sistema open hardware denominado Openmoko. *RITA*, Vol. , num. , pp. , 2010
[83] Rocío Martínez Torres, Carmen Díaz Fernández, Sergio Toral Marín, Federico Barrero García. Identification of added value ITS services using concept mapping techniques. *Proceedings of the international IEEE conference on intelligent transportation systems (ITSC-2009)*. IEEE Intelligent Transportation Systems Society, pp. 54 - 59, 2009
[84] Federico Barrero García, Sergio Toral Marín, Manuel Vargas Villanueva, Francisco J. Cortés Martínez, José M. Milla Carrillo. Internet in the development of future road-traffic control systems. *Internet Research*, Vol. 20, num. 2, pp. 154-168, 2010
[85] Sergio Toral Marín, Derlis Gregor Recalde, José M. Milla Carrillo, Manuel Vargas Villanueva, Federico Barrero García, Francisco J. Cortés Martínez. Smart cameras for cooperative urban applications. *Proceedings of IADIS international conference on collaborative technologies (CT-2010)*. IADIS, pp. 93 - 100, 2010
[86] Manuel Vargas Villanueva, Sergio Toral Marín, Federico Barrero García, Francisco J. Cortés Martínez. A license plate extraction algorithm based on edge statistics and region growing. *Proceedings of the international conference on image analysis and processing (ICIAP-09)*. ICIAP-09, pp. 317 - 326, 2009

[87] S. Toral, F. Barrero, M. Vargas. Development of an embedded vision based vehicle detection system using an ARM video processor. *Proceedings of the 11th International IEEE Conference on intelligent transportation systems*. ITSC 2008, pp. 292 - 297, 2008

[88] Sergio Toral, Derlis Gregor, Manuel Vargas, Federico Barrero, Francisco J. Cortés. Distributed urban traffic applications based on CORBA event services. *International journal of space-based and situated computing*, Vol. , num. , pp. , 2010

[89] José M. Hinojo Montero, Daniel Gutiérrez Reina, Esteban Marsal, Mateo Soto Ramos, Francisco J. Cortés Martínez, Federico Barrero García, Sergio Toral Marín. Sistema de orientación basado en brújula electrónica para la realización de prácticas. *Actas del IX congreso de tecnologías aplicadas a la enseñanza de la electrónica (TAEE 2010)*. TAEE 2010, pp. , 2010

[90] Sergio Toral Marín, Rocío Martínez Torres, Federico Barrero García, Francisco J. Cortés Martínez. An empirical study of the driving forces behind online communities. *Internet Research*, Vol. 19, num. 4, pp. , 2009

[91] Rocío Martínez Torres, Sergio Toral Marín, Federico Barrero García, Francisco J. Cortés Martínez. The role of Internet in the development of future software projects. *Internet Research*, Vol. 20, num. 1, pp. 72 - 86, 2010