UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

CARTELERA CULTURAL WEB Y BLUETOOTH

T E S I S

para obtener el título de:

INGENIERO EN COMPUTACIÓN

P R E S E N T A

ZAMIRA CRUZ MONTERROSAS

DIRECTOR DE TESIS:
M.C. ALEJANDRO VELÁZQUEZ MENA

México, D.F. Junio 2010

Agradecimientos

Como entrenamiento para mi futura entrega de Oscares, aprovecharé esta ocasión para darles las gracias a aquellas personas que, ya sea con su apoyo o indiferencia, han logrado que esta tesis por fin concluya. Dios sabe cómo mi equipo de trabajo y yo "worked our asses off" para llegar a este punto.

Madre, mi vida entera no bastaría para agradecerte… y mira que ya llevo unos cuantos años de trayectoria en este mundo. Esta tesis es más tuya que mía así que ¡Felicidades!

Padre, mi admiración por ti, así como mi necesidad de hacerte notar cómo a veces escucho tus palabras, me hicieron poseedora de la disciplina necesaria para terminar lo que inicio.

Dino, mi hermano, si de recobrar la tranquilidad se trataba, tus chistes, correos y recomendaciones de películas funcionaban mejor que "Dalai".

Centrina, si pudieras leer sabrías cuanto me han inspirado tus ojitos de capulín y el gracioso meneo de tu cola cuando me ves…o ves a cualquier otro ser humano.

Tanio, my soulmate, gracias por insultar la adversidad a mi lado y compartir la felicidad de este y todos mis demás proyectos, incluyendo el de korean madness.

Abril, hermana, nuestros momentos de frustración compartida me ayudaron a no darme por vencida y tu ejemplo a no tirarme al vicio antes de tiempo.

Jaque y Eli, gracias por creer en mí y en este trabajo. Cuando madure, intentaré ser tan positiva como ustedes.

Andrés, la pasión y entrega con la que haces las cosas son tan contagiosos que me inspiraron a poner más empeño en este y demás proyectos.

A mis compañeros del IINGEN (Noemí, Alejandro M., Alejandro G., Javier, Tomás, Yoshiro, Christian, Marco A., Tatiana, Julio, Carlos, Alberto, área de MAP), a mi grupo de Maya (Mons y Carlos) y a mi gente del CELEX (Adriana, Abraham, Alberto), gracias por su amistad y apoyo.

Al laboratorio de dispositivos móviles de la FI, a mis compañeros de la Facultad, a mi jurado, gracias por sus consejos.

Me temo que 45s no me serán suficientes y cuando eso pasa no queda más que decir: ¡GRACIAS A TODOS!

Índice

Índice de figuras

Índice de tablas

Introducción

La evolución de las telecomunicaciones desde las primeras investigaciones sobre ondas electromagnéticas hasta nuestra fecha, ha sido considerable. Pero no fue sino hasta 1997 que como consecuencia de la necesidad de conexión inalámbrica de bajo costo, bajo consumo y topología ad-hoc, nace el protocolo de comunicación bluetooth.

Sus características la hicieron idónea para ser instalada en dispositivos móviles como celulares o laptops, ampliando el espectro de posibilidades de uso que van desde la transmisión de datos hasta su explotación como medio de difusión e identificación.

En esta tesis se aplica el uso de bluetooth como medio de difusión, esto es, la distribución inalámbrica de contenidos informativos de forma no invasiva. A diferencia de la publicidad bluetooth manejada comúnmente, los que establecerán la conexión serán los usuarios quienes a través de una aplicación instalada en su dispositivo móvil, realizarán consultas personalizadas al servidor. De igual manera tendrán la posibilidad de consultar la cartelera cultural a través de internet.

Ya que la tecnología bluetooth no tiene como objetivo reemplazar la celular o la WiFi, es necesaria una comprensión global de los protocolos de comunicación inalámbrica existentes para poder elegir la más adecuada para nuestros objetivos. En el **Capítulo 1: Conceptos básicos** describo los varios tipos de redes inalámbricas según su alcance para posteriormente definir el protocolo Bluetooth como una red PAN.

Conocida las características históricas y técnicas de bluetooth, así como las ventajas y desventajas de su uso como protocolo de comunicación, continúo con el **Capítulo 2: JABWT**. Aquí detallo la forma de implementar la librería JSR-82 con los diferentes protocolos de transporte: L2CAP, RFCOMM, OBEX. Explico, de igual manera, el proceso de descubrimiento de servicios y dispositivos comunes a todos los protocolos.

Teniendo las bases del funcionamiento de la pila bluetooth y la librería JSR-82, en el **Capítulo 3: Programación** expongo la estructura de la aplicación cliente, también conocida como MIDlet. Asimismo defino los requerimientos básicos para la ejecución del mismo en el dispositivo móvil.

En el **Capítulo 4: Cartelera cultural web y bluetooth** retomo la información recabada en los capítulos anteriores al igual que la guía proporcionada por el marco de desarrollo del proceso unificado para desarrollar las aplicaciones web, cliente y servidor bluetooth.

Finalmente en el **Capítulo 5: Impacto ambiental y biológico** hablo d**e** las consecuencias que tiene el uso de ondas electromagnéticas según estudios de la INIRC.

Capítulo 1: Conceptos básicos

1.1 Introducción

Resolver la dificultad que presentaba la instalación de redes alámbricas en algunos sitios fue la tarea de los laboratorios IBM, quienes como solución debían proporcionar una alternativa inalámbrica que evitara interferencias con las señales electromagnéticas provenientes de maquinaria y cuyos procedimientos administrativos referentes al uso de frecuencias fueran mínimos o nulos. Después del fracaso de alcanzar 1Mbps en una cobertura razonable, el investigador principal abandonó el proyecto.

Paralelamente Ferrert de los laboratorios HP en Palo Alto, California, también realizaba investigaciones sobre redes inalámbricas. Su proyecto consistía en una red DSSS WLAN de 100kbps que operara en la banda de los 900MHz con CSMA como método de acceso. A causa de los impedimentos impuestos por la FCC para adquirir las frecuencias necesarias y debido a la complejidad administrativa, Ferrert abandonó el proyecto. Más tarde, Codex Motorola intentaría implementar una WLAN a 1.73GHz pero las negociaciones infructuosas con las FCC le obligaron a retirarse de la investigación. [1][2]

Aunque los proyectos pioneros en la creación de la WLAN fueron abandonados, la industria continuó llamando la atención y las negociaciones con la FCC para asegurar las frecuencias destinadas a estos propósitos, no pararon.

Las investigaciones permitieron definir retos que aún en nuestros días habrá que enfrentar, tales como:

- Complejidad y costo
- Ancho de banda
- Cobertura
- Interferencia
- Administración de la frecuencia

1.1.1 Clasificación de redes inalámbricas según su alcance

WWAN (Redes inalámbricas de área extensa) [23]

La tecnología se basa en el uso de redes celulares tales como WiMax, UMTS, GPRS, CDMA2000, GSM, CDPD, Mobitex, HSDPA o 3G para transmitir datos. También puede hacer uso de LMDS o WiFi para conectarse al Internet.

Es gracias a estas redes celulares que los servicios de las WWAN poseen un alcance tanto regional como nacional y global, con una cuota libre.

WMAN (Redes inalámbricas metropolitanas) [31]

Las redes WMAN también conocidas como Redes Inalámbricas de Bucle Local o WLL por sus siglas en inglés (Wireless Local Loop), permiten a los usuarios establecer conexiones inalámbricas entre varias ubicaciones dentro de un área metropolitana (radio de entre 4 a 10 km).

Su velocidad esta en el rango de 10Mbps. La mejor velocidad es alcanzada por las WiMax con 70Mbps en un radio considerablemente extenso.

WLAN (Redes inalámbricas locales) [32]

Su alcance es de aproximadamente 100 metros con velocidades máximas de hasta 54 Mbps. Su origen radica en la necesidad empresarial de redes inalámbrica; sin embargo, en la actualidad funcionan en una gran variedad de escenarios. Los estándares a los que obedecen estas redes son: 802.11 a, b y g.

WPAN (Redes inalámbricas personales)[11]

Caracterizadas por su corto alcance (hasta 10 m), topología ad-hoc, arquitectura plug and play, soporte de voz y datos, y bajo consumo de energía; se usa generalmente para conectar periféricos, asistentes personales a un ordenador e incluso hacer conexiones entre dos ordenadores próximos.

Comenzaron como BodyLANs que conectaban sensores y dispositivos de información fijados al cuerpo para aplicaciones militares y como redes personales para conectar equipos como laptops, notepads, y teléfonos celulares con aplicaciones comerciales.

La primera BodyLAN surge del proyecto DARPA a mediados de los 90's. Esta era una WPAN de bajo consumo, tamaño pequeño, barata, con un modesto ancho de banda que permitía la conexión de dispositivos personales en un rango de alrededor de cinco pies.

Entre las principales tecnologías utilizadas por las WPAN se encuentran: Zigbee (802.15.4), conexiones infrarrojas, Proyecto Oxygen (en desarrollo), HomeRF y Bluetooth (802.15.1), la tabla 1.1 muestra sus principales características[1][2][3][5].

Característica y Función	IrDA	LAN inalámbrica	Zigbee	HomeRF	Bluetooth
Tipo de Conexión	Infrarrojo, estrecha, ancha línea de vista	Espectro de difusión, esférico	Espectro de difusión, esférico	Espectro de difusión, esférico	Espectro de difusión, esférico
Espectro	Óptico 850-900nm	RF 2.4 GHz (5Ghz para 802.11 a)	RF 2.4 GHz	RF 2.4 GHz	RF 2.4 GHz
Poder de Transmisión	40-500mW/Sr	100mW	1mW	100mW	1-100mW
Máxima tasa de datos	9600bps-16 Mbps	11 Mbps (54 Mbps para 802.11 a, 802.11g)	250 Kbps	1-2 Mbps 10 Mbps en HomeRF2	1 Mbps
Rango	1m	100m	10-75m	50-100m	10-100m
Dispositivos soportados	2	Se conecta mediante un punto de acceso	65535 nodos distribuidos en subredes de 255 nodos	127	8 (activo), 200 (pasivo)
Canales de voz	No soporta	VoIP	No soporta	75 de 1MHz/15 de 5MHz para HomeRF2	3

Direccionamiento	ID físico de 32 bit	48 bit MAC	16-64 bit MAC	48 bit MAC	48 bit MAC

Tabla 1.1 Comparativa entre Tecnologías de la Comunicación Inalámbricas

1.1.2 Cronología[6][44]

1880 Hertz comienza investigaciones sobre comunicaciones móviles.

1897 Marconi lleva a cabo trabajos experimentales que dan como resultado la transmisión por radio hacia un barco.

1921 El Departamento de Policía de la ciudad de Detroit instala el primer sistema de radiotelefonía móvil (sistema de despacho), operando en la banda de los 2 MHz.

1930-1950 Varios canales se usan sobre una base experimental. A mediados de los años 40's nuevos sistemas comerciales simplex se instalan en las bandas de los 33 y 150 MHz.

1960 El desarrollo de módems y PBX´s (Private Branch Exchange lo que en español se conoce como Central Telefónica) marcan el inicio de la convergencia entre comunicación y computación. La comunicación evolucionó para permitir tanto el tráfico de voz como de datos en una misma red, lo que da paso a la aparición de las redes computacionales e Internet.

1978 En Chicago, EUA, comenzó a instalarse el sistema AMPS (Advance Mobile Service) en la banda de los 900 MHz con 666 canales disponibles. Este sistema tuvo una cobertura de 5400 km^2 con 10 células y 136 canales para 2 mil abonados.

1979 Ingenieros suizos utilizan un enlace infrarrojo para crear una red local en una fábrica. Esto se puede considerar como el punto de partida en la línea evolutiva de esta tecnología. En Japón, por su parte, se instala el primer sistema celular en la banda de los 900 MHz

1981 Entra en Europa el primer sistema celular en la banda de los 450 MHz, denominado NMT (Nordic Mobile Telephone System).

1982 Se forma un grupo de trabajo auspiciado por los gobiernos europeos (Conférence Européenne des Postes Télécommunications- CEPT) llamado GSM.

1983 El sistema AMP aumenta sus abonados a 30 mil.

1985 Se asignan las bandas IMS 902-928 MHz, 2,400-2,4835 GHz, 5,725-5,850 GHz a las redes inalámbricas con espectro disperso.

1990 Se conforma el comité IEEE 802.11 para tratar de generar una norma para las WLAN, no obstante, el primer borrador aparece hasta 1994.

1991 Se publican varios trabajos referentes a WLAN's operativas que superaban la velocidad de 1 Mbps, mínimo establecido por el IEEE 802.

1992 Se crea Winforum, consorcio liderado por Apple y formado por empresas del sector de las telecomunicaciones y de la informática para conseguir bandas de frecuencia para los sistemas PCS (Personal Communications Systems).

La ETSI (Instituto de Estándares de Telecomunicaciones Europeo), a través del comité ETSI-RES 10, inicia operaciones para crear una norma a la que denomina HiperLAN (High Performance LAN)

1993 La ETSI asigna las bandas de 5,2 y 17,1 GHz a la HiperLAN. En ese mismo año se constituye la IRDA (Infrared Data Association) para promover el desarrollo de las WLAN basadas en enlaces infrarrojos y comienza la instalación del sistema GSM en Europa.

1996 Un grupo de empresas del sector de la informática móvil y de servicios forman el Foro WLI (Interoperabilidad LAN inalámbrica) para potenciar este mercado mediante la creación de un amplio abanico de productos y servicios inter-operativos. Entre los miembros fundadores se encuentran empresas como ALPS Electronic, AMP, Data General, Contron, Seiko Epson y Zenith Data Systems.

1998 En Enero el grupo WPAN publica el requerimiento original de funcionalidad. Así mismo se conforma el grupo Home RF y anuncia el desarrollo de Bluetooth. Cinco empresas forman el grupo de interés especial (SIG) de Bluetooth.

1999 El estándar 802.15 fue aprobado como un grupo separado de la comunidad 802.11. Se publica la especificación 1.0 de la tecnología Bluetooth y el SIG celebra el primer UnPlugFest para los ingenieros miembros.

2000 Se crea el primer teléfono móvil, auricular y tarjeta PC Card. Sale al mercado el primer chip con funciones de frecuencia de radio, banda base, microprocesador y software inalámbrico bluetooth.

2001 El SIG se constituye como organización privada.

2002 El IEEE aprueba la especificación 802.15.1 en conformidad con la tecnología inalámbrica Bluetooth.

2003 El SIG adopta la versión 1.2 de la especificación principal de Bluetooth. La distribución de productos con tecnología Bluetooth alcanza 1 millón de unidades semanales.

2004 El SIG de Bluetooth adopta la versión 2.0 de la especificación principal Bluetooth y la especificación de transferencia de datos mejorada (EDR); el número de dispositivos equipados con tecnología bluetooth alcanza los 250 millones.

2005 El SIG de Bluetooth acoge a su miembro número 4.000, establece su sede en Bellevue (WA) y habilita oficinas en Malmo, Suecia, y Hong Kong.

2006 El SIG de Bluetooth anuncia que integrará la tecnología Bluetooth con la versión de UWB de WiMedia Alliance.

2007 Presentación del SIGnature, la publicación trimestral de Bluetooth, en el encuentro plenario del SIG celebrado en Viena, Austria. El foro Wibree se fusiona con el SIG de Bluetooth. La especificación principal, versión 2.1 + EDR es publicada.

2009 El SIG anuncia la liberación de la especificación bluetooth de alta velocidad, V3.0+HS. Esta especificación permitirá alcanzar velocidades mayores en las tasas de transferencia de dispositivos clientes haciendo uso de un radio secundario presente en el dispositivo.

La figura 1.1 muestra los avances alcanzados en la tecnología bluetooth de forma cronológica.

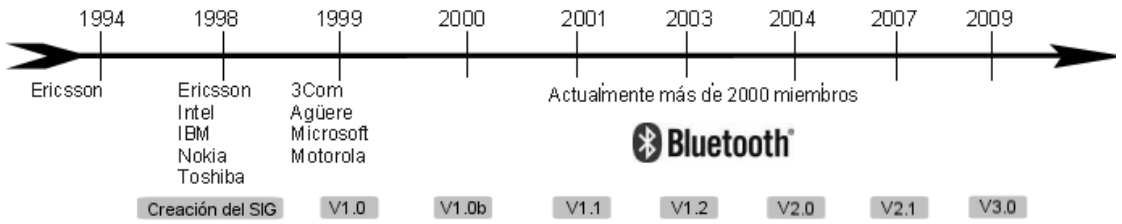


Fig. 1.1 Cronología de la tecnología Bluetooth

1.2 Bluetooth

1.2.1 Introducción

Originalmente esta tecnología fue desarrollada para funcionar como un reemplazo de las conexiones por cable, habilitando enlaces inalámbricos eficientes entre dispositivos a través de una red inalámbrica personal (PAN).

Fue en 1994 Ericsson comenzó investigaciones para encontrar alternativas en las conexiones entre teléfonos móviles y sus accesorios. Para 1998 Intel, IBM y Nokia se unieron a la investigación y nació el Grupo de Intereses Especiales sobre Bluetooth (SIG) quienes posteriormente anunciaron el nacimiento del SIG de Bluetooth Global y convocaron a más compañías a unirse. En Diciembre de 1999 3Com, Agüere, Microsoft y Motorola se unieron al Equipo.

1.2.2 Historia[47]

La historia detrás del nombre “Bluetooth” evoca al vikingo Harald Blätand (Figura 1.2), hijo del rey Grom “El Viejo” y Thyre Danebold. Blätand llevó a la cúspide los planes de unificación del país de su padre, convirtió a los daneses al cristianismo y amplió su territorio hasta tierras noruegas. La expansión de su imperio continuó bajo la dirección de su sucesor, Sweyn I, quien conquistó Inglaterra en 1013; pero no fue sino hasta el gobierno de Canute, nieto de Blätand e hijo de Sweyn I, que el imperio anglo-escandinavo alcanzó su mayor dimensión.

Una de las teorías sobre la etimología del nombre Blätand establece que el nombre traducido al inglés puede interpretarse como “diente azul” ya que se cree que Harald pudo haber padecido eritroblatosis fetal, la cual habría hecho que alguno de sus dientes tuviera un color azulado. Otra teoría señala que el nombre proviene de las dos antiguas palabras danesas “blä” y “tan” cuyo significado es piel oscura y hombre magnánimo respectivamente.

La analogía con la tecnología de comunicación bluetooth reside precisamente en la capacidad de unificación que el vikingo Harald Blätand mostró y el objetivo que se desea alcanzar con esta tecnología; esto es, la unificación del mundo computacional con el de las telecomunicaciones.

Fig. 1.2 Bautizo de Harald Blätand

1.2.3 Datos técnicos [6] [7] [44]

La tecnología inalámbrica bluetooth se caracterizada por un bajo costo, bajo consumo de energía, y corto alcance en comunicaciones inalámbricas de voz y datos ad hoc. Opera en la banda ISM1 a 2.4 GHz La máxima velocidad de transmisión de datos es de 1 Mbps y el rango de alcance depende de la potencia empleada en la transmisión (la mayor parte de los dispositivos que usan Bluetooth transmiten con una potencia nominal de salida de 0 dBm con un alcance de 10 metros en un ambiente libre de obstáculos).

Debido a la alta demanda de la banda ISM, Bluetooth adoptó un sistema de salto de frecuencia aplicado a una alta velocidad con una corta longitud de paquetes para evitar interferencias producidas por otros dispositivos. Estos paquetes de datos están protegidos por un esquema ARQ (recepción automática de consulta) en la que los paquetes son retransmitidos automáticamente.

El establecimiento automático de comunicación entre terminales bluetooth se conoce como piconet. En otras palabras, una piconet es la conexión de dos o más unidades bajo un mismo canal. Al conjunto de dos o más piconets se le conoce como scatternet. (Figura 1.3)

Las agrupaciones de dispositivos están limitadas a ocho unidades por canal dentro de una piconet y diez piconets en una misma área de cobertura.

La conexión de dispositivos bluetooth trabaja bajo el esquema maestro-esclavo(s); ésto es, una única unidad maestra controlará el tráfico de datos generado por una o más unidades esclavas. Con el fin de evitar congestionamiento en una scatternet es necesario que sólo aquellas unidades que requieran intercambiar información compartan el mismo canal.

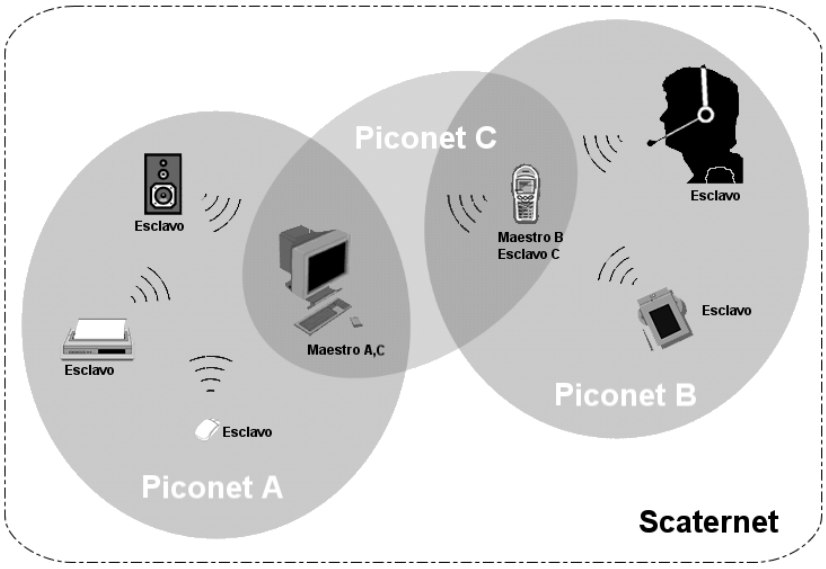


Fig. 1.3 Piconet y Scaternet

De forma resumida, las características técnicas que dictan el comportamiento de la tecnología bluetooth se exponen a continuación[50]:

- Banda de Frecuencia: 2.4 GHz (banda libre ISM1).
- Potencia del transmisor: 1 mW para 10 metros, y 100 mW para 100 metros.
- Modulación: Espectro expandido, secuencia directa híbrida y saltos en frecuencia.
- Canales máximos de voz: 3 por piconet (64 Kbps bidireccional).
- Canales máximos de datos: 7 por piconet.
- Velocidad de datos: hasta 721 kbit/s asimétrico (hasta 57.6 Kbps de retorno) o 433.9 Kbps simétrico. Se espera doblar en la siguiente generación.
- Rango esperado del sistema: 10 metros.
- Número de dispositivos: 8 por piconet y hasta 10 piconets.
- Seguridad: Sí, en la capa de enlace.
- Consumo de potencia: desde 30 μA en espera hasta 8-30 mA (a 2,7 v) transmitiendo
- Interferencia: Bluetooth minimiza la interferencia potencial al emplear saltos rápidos en frecuencia (1600 veces por segundo).

1.2.4 Especificación[44]

La especificación define el comportamiento en el aire de la señal que asegurará la compatibilidad de dispositivos Bluetooth de diferentes proveedores, así como la completa comunicación que va desde el radio hasta la aplicación.

En un nivel más alto, la especificación se divide en dos volúmenes. El Volumen 1, considerado el principal, describe el protocolo de pila y elementos relacionados con pruebas y calidad mientras que el volumen 2 define los perfiles Bluetooth disponibles para la utilización de aplicaciones en la pila Bluetooth.

Pila bluetooth

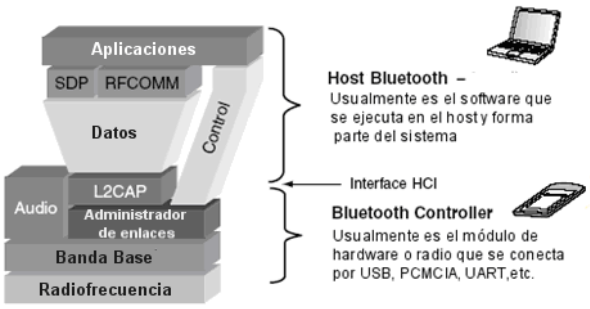


Fig. 1.4 Pila Bluetooth

Cada capa de la pila define un protocolo y es descrita separadamente de la especificación principal. Para cuestiones prácticas, la pila puede dividirse en dos componentes: el host y el controlador. (Figura 1.4)

La interfaz de control de host o HCI por sus siglas en inglés Host Control Interfaz, como su nombre lo dice, provee una interfaz estándar entre el host y el controlador. Ésta viene integrada al software del sistema o al sistema operativo del dispositivo; contrario al modulador de radio o controlador, generalmente un módulo de hardware que se conecta a un dispositivo.

Como se observa en la figura 1.4, las capas superiores se conectan con el módulo de radio a través del HCI; el módulo de radio se conecta con el sistema del host a través de algún mecanismo de entrada-salida estándar.

Perfiles[19]

El uso de la tecnología inalámbrica Bluetooth depende de la interpretación de los perfiles bluetooth. Su objetivo es proporcionar procedimientos de comunicación e interoperabilidad entre dispositivos a través de la descripción de aplicaciones. Por esta razón es considerable la cantidad de perfiles disponibles para el desarrollador.

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.

Los perfiles más utilizados por su utilidad en la intercomunicación entre dispositivos se muestran en la tabla 1.2:

Perfiles	Descripción
Perfil de distribución de audio avanzado (A2DP)	El perfil A2DP describe cómo transferir sonido estéreo de alta calidad de una fuente de sonido a un dispositivo receptor.
Perfil de control de audio y vídeo (AVRCP)	El perfil AVRCP proporciona una interfaz estándar para manejar televisiones, equipos de alta fidelidad o cualquier otro equipo electrónico, y permitir así que un único control remoto, o cualquier otro tipo de mando, controle todo el equipo de audio y vídeo al que el usuario tiene acceso.

Perfil básico de imagen (BIP)	El perfil BIP establece cómo puede controlarse remotamente un dispositivo de imagen, así como la forma de enviarle órdenes de impresión y de transferencia de imágenes a un dispositivo de almacenamiento.
Perfil básico de impresión (BPP)	El perfil BPP permite enviar mensajes de texto, de correo electrónico, tarjetas de visitas electrónicas e imágenes, entre otras cosas, a las impresoras disponibles dependiendo de las tareas de impresión.
Perfil de acceso RDSI común (CIP)	El perfil CIP establece cómo se deben transferir las señales RDSI a través de una conexión inalámbrica Bluetooth.
Perfil de telefonía inalámbrica (CTP)	El perfil CTP describe la implementación de un teléfono inalámbrico a través de un enlace inalámbrico Bluetooth.
Perfil de red de marcado (DUN)	El perfil DUN proporciona un acceso telefónico estándar a Internet y a otros servicios de marcado a través de una conexión Bluetooth.
Perfil de fax (FAX)	El perfil FAX describe cómo un dispositivo terminal puede utilizar a otro como puerta de enlace para la transmisión de faxes.
Perfil de transferencia de archivos (FTP)	El perfil FTP establece los procedimientos de exploración de carpetas y archivos de un servidor a través de un dispositivo cliente.
Perfil de distribución genérica de audio y vídeo (GAVDP)	El perfil GAVDP sienta las bases de los perfiles A2DP y VDP, pilar de los sistemas diseñados para la transmisión de sonido e imagen mediante la tecnología Bluetooth.
Perfil genérico de intercambio de objetos (GOEP)	El GOEP se utiliza para transferir objetos de un dispositivo a otro.
Perfil manos libres (HFP)	El perfil HFP describe cómo un dispositivo que actúa como puerta de enlace puede utilizarse para realizar y recibir llamadas a través de un dispositivo manos libres.
Perfil de sustitución de cable de copia impresa (HCRP)	El perfil HCRP describe cómo imprimir archivos mediante un enlace inalámbrico Bluetooth utilizando controladores en el proceso.
Perfil de auricular (HSP)	El HSP describe cómo un auricular con tecnología Bluetooth se comunica con otro dispositivo con tecnología Bluetooth.
Perfil de dispositivo de interfaz humana (HID)	El perfil HID recoge los protocolos, procedimientos y características empleados por las interfaces de usuario Bluetooth tales como teclados, dispositivos punteros, consolas o aparatos de control remoto.
Perfil de intercomunicador (ICP)	El perfil ICP establece cómo conectar dos teléfonos móviles con tecnología Bluetooth dentro la misma red sin utilizar la red telefónica pública.
Perfil de introducción de objetos (OPP)	Este perfil distingue entre servidor y cliente de introducción (push) de objetos.
Perfil de redes de área personal (PAN)	El perfil PAN describe cómo dos o más dispositivos con tecnología 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.
Perfil de aplicación de descubrimiento de servicio (SDAP)	El perfil SDAP detalla cómo una aplicación debe utilizar el perfil SDP para identificar los servicios de un dispositivo remoto.
Perfil de servicio de puerto (SPP)	El perfil SPP describe cómo configurar puertos de serie y conectar dos dispositivos con tecnología Bluetooth.
Perfil de sincronización (SYNC)	El perfil SYNC se utiliza junto al GOEP para sincronizar los elementos del administrador de información personal (PIM), como agendas y datos de contacto, entre dispositivos con tecnología Bluetooth.
Perfil de distribución de vídeo (VDP)	Este perfil dicta los pasos que deben seguir los dispositivos Bluetooth para la transferencia de flujos de datos de vídeo.

Tabla 1.2 Principales perfiles bluetooth

1.2.5 Protocolos

Los protocolos más comunes pueden agruparse en cuatro capas (Figura 1.5):

- Protocolos Bluetooth Centrales (Baseband, LMP, L2CAP, SDP)

- Protocolos de Reemplazo de Cable (RFCOMM)
- Protocolos de control de Telefonía (TCS Binary)
- Protocolos Adaptados (UDP/TCP/IP, OBEX, WAP, IrMC)

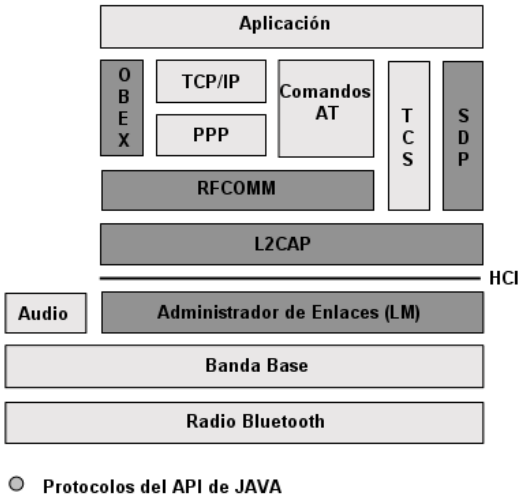


Fig. 1.5 Protocolos incluidos en el API de JAVA

Radio Bluetooth: define los requerimientos del transmisor y receptor operando en la banda ISM.

Banda base y control de enlaces: habilitan la conexión RF física entre unidades Bluetooth en red. La banda base especifica o introduce los procedimientos de acceso de medios y capa física entre dispositivos Bluetooth; el link de control gestiona el acceso al canal.

LMP (Protocolo de Gestión de Enlace): capa responsable del establecimiento y configuración de la conexión entre dispositivos, de la gestión y establecimiento del tamaño de los paquetes en la banda base y de la seguridad.

HCI (Interfaz del Controlador del Host): es un control intermedio entre las capas LMP, Banda base y Radio. Proporciona una interfaz para el acceso a los servicios de la banda base, el estatus del hardware y los registros de control.

L2CAP (Protocolo de adaptación y de control de enlace lógico) funciona como un multiplexor de conexiones lógicas hechas por capas superiores.

Las aplicaciones hacen uso de la capa SDP para la búsqueda de servicios y sus características.

El protocolo RFCOMM emula puertos seriales sobre L2CAP, permitiendo conexiones simultáneas de uno o varios dispositivos.

BPNEP (Protocolo de Encapsulación de Red Bluetooth) es un protocolo opcional que encapsula paquetes provenientes de diversos protocolos de red.

TCS (Especificación del Protocolo de Control Telefónico) define el señalamiento de control de llamadas de voz y datos entre dispositivos bluetooth.

1.2.6 Conexión[6]

Un dispositivo se conecta con otro mediante un mensaje o consulta. El mensaje es enviado en 32 frecuencias diferentes y es escuchado por otro dispositivo que salga de su estado de "espera" para procesar el mensaje.

Los dispositivos se encuentran por defecto en estado de espera y se activan o despiertan cada 1.28 segundos para escuchar posibles mensajes. La figura 1.6 ejemplifica el proceso de conexión bajo diferentes protocolos.

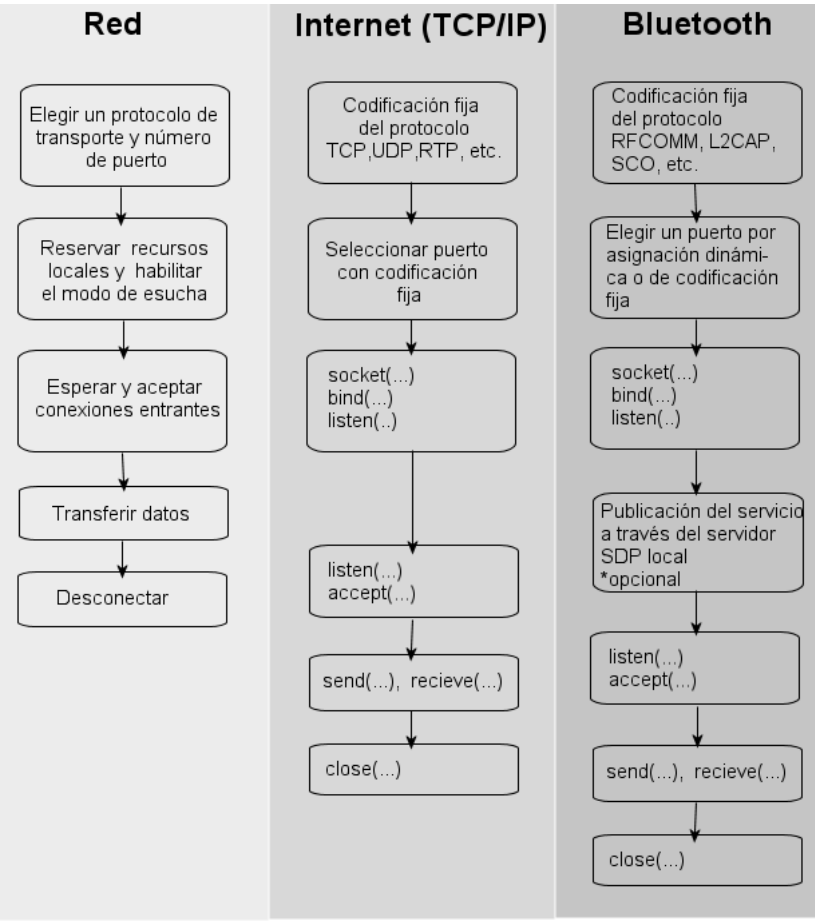


Fig. 1.6 Comunicación de dispositivos bajo diferentes protocolos

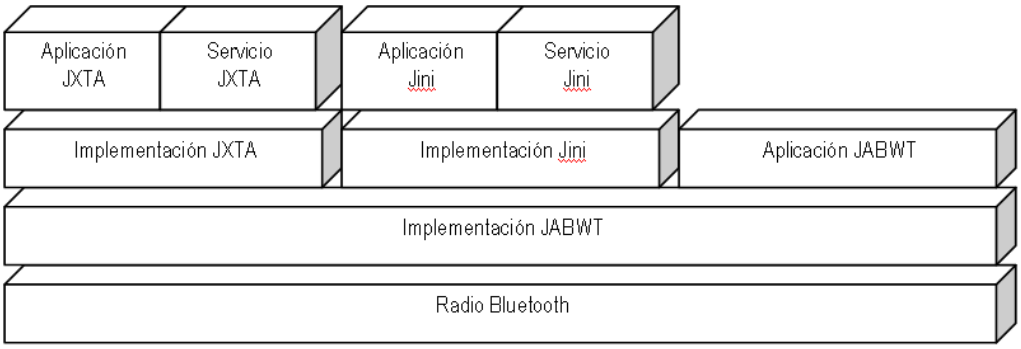
1.2.7 Otras tecnologías inalámbricas[7]

Otras especificaciones que proporcionan una forma dinámica de identificar dispositivos y servicios son JINI, JXTA. Lo que las hace diferente de Bluetooth es que mientras éste define un conjunto de protocolos de comunicación de alto y bajo nivel, JINI y JXTA operan sólo en alto nivel.

JINI es una arquitectura orientada a servicios cuyo modelo de programación permite explotar y ampliar las capacidades de construcción de sistemas seguros y distribuidos de la tecnología java.

Esta tecnología proporciona herramientas para la creación de sistemas escalables, evolutivos y flexibles basados en redes como lo requieren los entornos de computación dinámica.

JXTA (Yuxtaposición) es una plataforma punto a punto de código libre creada en el 2001 por Sun Microsystems. Su funcionamiento se basa en grupos de protocolos XML abiertos que permiten la comunicación descentralizada de dispositivos dentro de una red. Cada nodo es identificado por un ID único (URN SHA-1) de 160 bits en la implementación de Java, permitiendo que los dispositivos puedan modificar su dirección pero conservar su número de identificación. JXTA es la infraestructura p2p más robusta existente tanto por su amplia aceptación de dispositivos varios (computadoras, teléfonos móviles, PDAs) como por su compatibilidad con diversos lenguajes de programación.



Fig, 1.7 Jini y JXTA en un dispositivo bluetooth con JABWT

Capítulo 2: JABWT[6]

2.1 API's de Alto Nivel[11] [17]

La creación de interfaces gráficas para móviles a través de JAVA puede desarrollarse tanto con API's de alto nivel como de bajo nivel.

Los API's de alto nivel trabajan con los componentes nativos del sistema mientras que los de bajo nivel permiten, generalmente, la creación de componentes personalizados.
Dependiendo del fin de la aplicación será conveniente utilizar una u otra API. La tabla 2.1 muestra algunas diferencias entre ellas.

Ventajas	API Alto Nivel	API Bajo Nivel
Flexibilidad de diseño		*
Rapidez de respuesta		*
Compatibilidad y portabilidad	*	
Programación sencilla	*	
Control sobre eventos		*
Control sobre elementos		*
Útil para aplicaciones de alta demanda	*	
Útil en dispositivos limitados en hardware	*	

Tabla 2.1 Características principales del API de alto y bajo nivel

El API de alto nivel se encuentra disponible a través de las siguientes clases de pantalla (screen classes): listas, cajas de texto, alertas y formas.

La clase de forma depende a su vez de las subclases: elemento cadena, elemento imagen, campo de texto, campo de fecha, indicador, grupo de opciones múltiples, elemento personalizado y separador.

La funcionalidad JABWT se especifica en el API JSR-82, API de alto nivel empleado en la programación de dispositivos Bluetooth que permite el descubrimiento, comunicación y gestión de dispositivos.

JABWT depende de la configuración CLDC y usa el GCF, generalmente en conjunto con perfiles J2ME como el MIDP. La figura 2.1 muestra esta arquitectura.

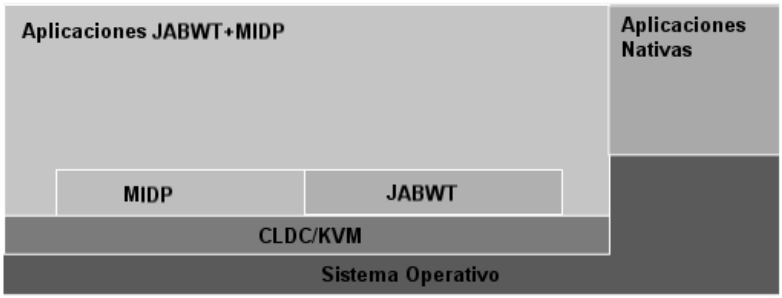


Fig. 2.1 Arquitectura CLDC+MIDP+Bluetooth

Las aplicaciones java nativas interactúan directamente con el sistema operativo; mientras que las implementaciones CLDC/KVM proveen una plataforma de ejecución y construcción de API's de alto nivel.

2.1.1 Conexión de dispositivos

Java define dos paquetes independientes para la comunicación de dispositivos: javax.bluetooth y javax.obex. Estos paquetes obedecen al modelo de comunicación cliente-servidor.

Un servicio bluetooth es una aplicación que actúa como servidor y proporciona asistencia (generalmente una función no disponible localmente) a dispositivos clientes. Éste se habilita a través de la definición de un registro de servicio que se anexa en la SDDB (Base de Datos para el Descubrimiento de Servicios). Una forma gráfica de visualizar la interacción de estos elementos se puede observar en la figura 2.2

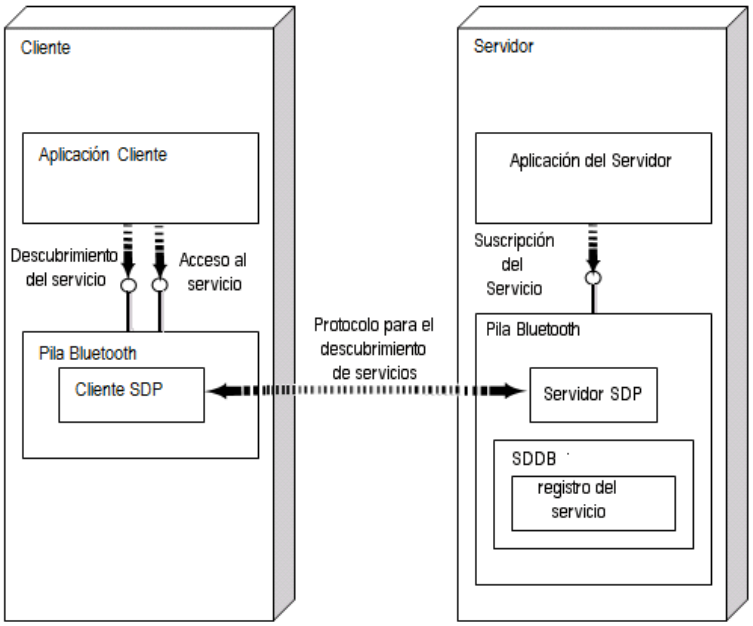


Fig 2.2 Componentes Bluetooth involucrados en el descubrimiento de servicios

Una vez registrado el servicio en el servidor, éste espera a que la aplicación cliente envíe una petición de acceso para establecer una conexión.

Aspectos como el formato interno o mecanismo de funcionamiento de la SDDB, interacción entre la pila bluetooth y las aplicaciones o tiempo y forma de conexiones con clientes remotos; varían entre pila y pila. No obstante, JABWT establece una estandarización para el registro del servidor facilitando así la comunicación. La tabla 2.2 muestra las actividades designadas a la pila, dispositivo servidor y el dispositivo cliente según JABWT:

	Servidor	Cliente
Dispositivo	Crear el registro de servicios. Añadir el registro de servicios a la SDDB. Registrar medidas de seguridad asociadas al servicio. Actualizar, eliminar o deshabilitar los registros de la SDDB.	Crear petición de servicios a SDDB's remotas a través del SDP Registrar las medidas de seguridad asignadas al servicio Iniciar la conexión con el servidor
Pila	Creación de repositorio para la actualización, anexión y eliminación de registros de servicios. Conexión con aplicaciones de clientes remotos	Búsqueda y extracción de registros en la SDDB del servidor. Conexión con aplicaciones del servidor.

Tabla 2.2 Principales actividades de la pila y el servidor.

Las aplicaciones bluetooth pueden actuar como servidor o cliente, por consiguiente son consideradas también como aplicaciones punto a punto (peer to peer)

2.1.2 Centro de Control Bluetooth (BCC)

BCC es un concepto definido en JABWT que funge como un comando central para la configuración local de dispositivos bluetooth.

El BCC puede presentarse como una aplicación nativa en un API externo o como parte de un grupo de configuraciones especificadas por el fabricante. Debido a que la pila no es directamente accedida por aplicaciones JABWT, se establecen peticiones por medio del BCC.

Fig. 2.3 El BCC

Las tareas principales del BCC son:

Resolver conflictos de peticiones entre aplicaciones. Cuando se realizan dos peticiones desde el mismo dispositivo se incurre en un conflicto entre aplicaciones, ya sea porque cada petición requiera una configuración de seguridad diferente o los dispositivos difieran en su modo de conexión.

Habilitar la modificación de las propiedades locales del dispositivo. Entiéndase como propiedades: nombre de reconocimiento, la clase para el registro del dispositivo, listado de dispositivos reconocidos, lista de dispositivos confiables, requisitos de seguridad mínima y soporte para los diferentes modos de descubrimiento.

Manejo de operaciones de seguridad. El BCC se encarga de la extracción de datos de seguridad del usuario para utilizarla en el proceso de seguridad de bluetooth.

2.2 RFCOMM[10]

La comunicación RFCOMM para bluetooth se realiza gracias al perfil SPP quien define una forma confiable de establecer una conexión bidireccional entre dos dispositivos bluetooth. La figura 2.4 muestra los perfiles bluetooth validados por el SIG para RFCOMM.

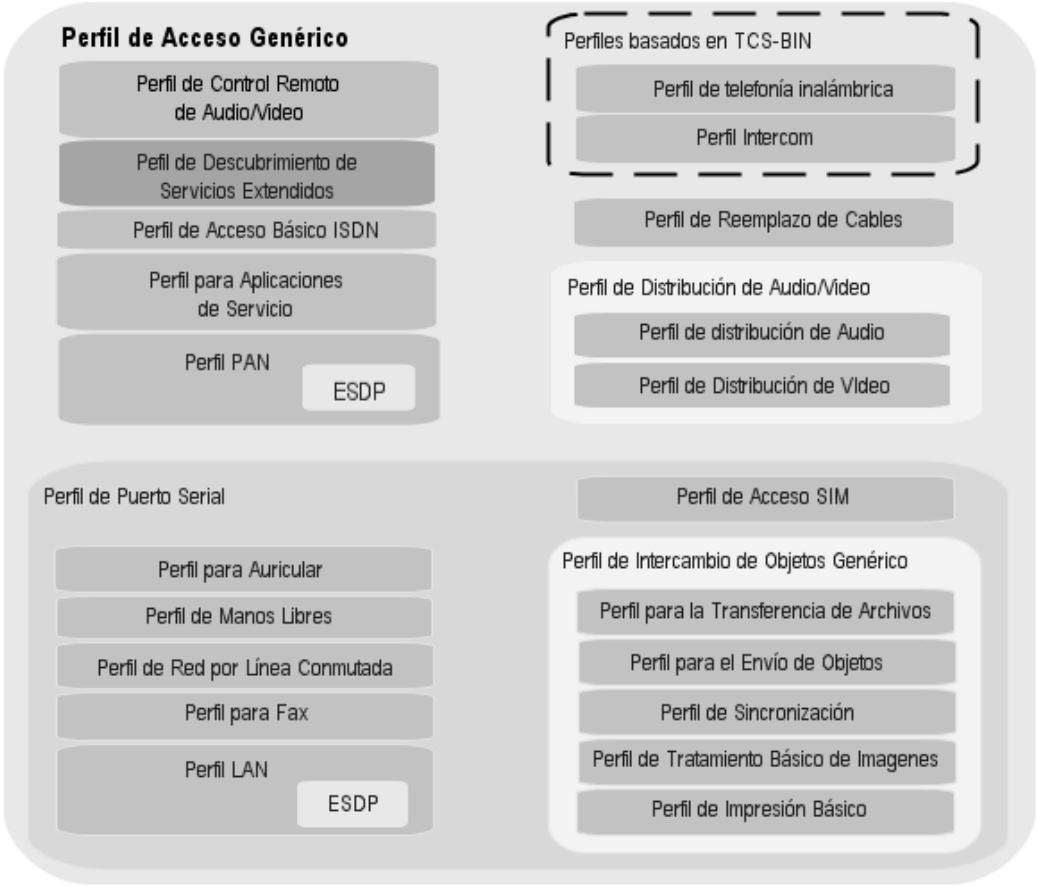


Fig. 2.4 Perfiles Bluetooth definidos por el SIG

El funcionamiento establecido por el protocolo RFCOMM emula la conexión de un puerto serial RS-232 en un sistema inalámbrico; esto es, los datos son enviados a los dispositivos por flujos (streams).

2.2.1 Seguridad

Los sistemas Bluetooth admiten la existencia de un único enlace físico entre dos dispositivos y una o más conexiones sobre el mismo (Figura 2.5). La seguridad de estos enlaces la proveen cuatro métodos complementarios: emparejamiento, autenticación, encriptación y autorización.

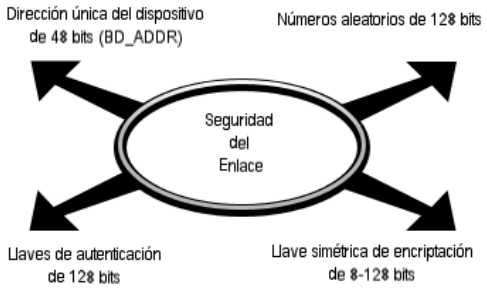


Fig. 2.5 Múltiples conexiones sobre un mismo enlace

El emparejamiento se establece la primera vez que se realiza una conexión entre dos dispositivos. Requiere que estos fijen un código de ingreso o PIN que será utilizado como identificador en el proceso de encriptación y autenticación. Una vez fijado el PIN, los dispositivos harán uso del mismo en posteriores enlaces.

Fig. 2.6 Emparejamiento de dispositivos

Durante la autenticación se verifica la identidad de los dispositivos acoplados a través de un esquema de desafío-respuesta. La autenticación bluetooth verifica las identidades de los dispositivos, no de los usuarios.

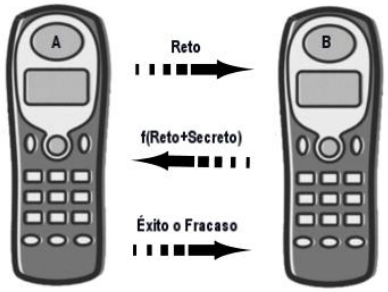


Fig. 2.7 Dispositivo A autenticando dispositivo B

En el proceso de encriptación se intenta evitar intrusiones durante la conexión. Para que este proceso funcione, ambos dispositivos deben aceptar la petición de encriptación por lo que su uso es opcional. (Ver fig. 2.7)

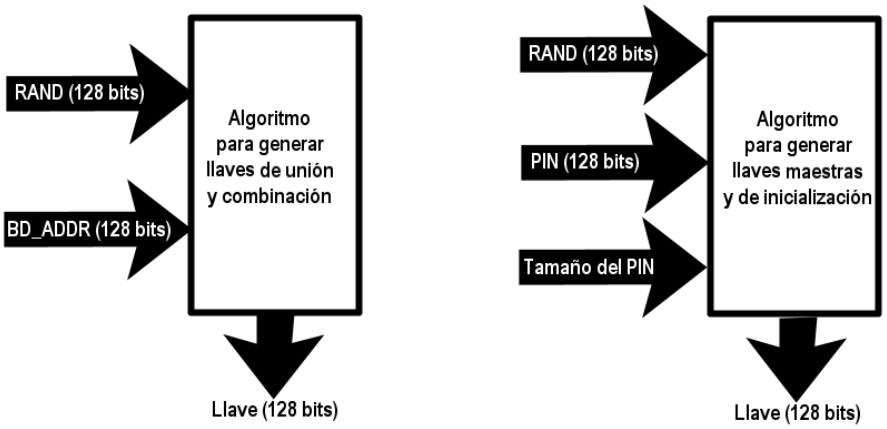


Fig. 2.8 Creación de llaves encriptadas

En la última etapa de protección se encuentra la autorización, proceso que determinará si una petición de conexión será o no concedida. Un dispositivo al que se le conceden autorizaciones automáticamente es denominado “dispositivo de confianza”. De ellos se encarga el BCC.

JABWT permite modificar los métodos de seguridad bajo dos circunstancias: cuando se establece por primera vez la conexión o después de haberla establecido.

2.2.2 Conexión con el servidor

La conexión comienza al invocar el método Connector.open() con una cadena válida de la forma: [esquema]:[objetivo][parámetros]

El esquema utilizado para las conexiones RFCOMM es btspp. El valor en el objetivo y parámetros dependerán del tipo de conexión (cliente o servidor). La tabla 2.3 muestra estos parámetros.

Nombre	Descripción	Valores válidos	Cliente o Servidor
Maestro	Especifica si el dispositivo se comportará como maestro	Verdadero, falso	Ambos
Autenticar	Especifica si el dispositivo remoto debe autenticarse antes de establecer la conexión.	Verdadero, falso	Ambos
Cifrar	Especifica si el enlace ha de ser cifrado	Verdadero, falso	Ambos
Autorizar	Especifica si las conexiones están autorizadas a utilizar servicios.	Verdadero, falso	Servidor
Nombrar	Especifica el nombre de servicio en el registro de servicios.	Cualquier cadena válida	Servidor

Tabla 2.3 Parámetros válidos para cadenas en conexiones RFCOMM

Valores ilegales causan excepciones por argumentos ilegales (IllegalArgumentException) mientras que combinaciones inválidas (p. ej. activar la encriptación y desactivar la autenticación) causan excepciones por conexión bluetooth (bluetoothConnectionException)

Ejemplo de una cadena de conexión válida es:
Btspp://localhost:10203040506070809OA1B1C1D1E100;namr=Print_Server;master=false

Una vez invocado el método, se espera recibir un objeto StreamConnectionNotifier para considerar una conexión como válida. Posteriormente acceptAndOpen() devuelve un objeto de conexión por flujo (StreamConnection). Este objeto permite la lectura y escritura en la aplicación cliente.

2.2.3 Conexión con el cliente

La cadena “btspp://” seguida de una dirección destino y del identificador del canal del servidor, conforma una cadena de conexión de cliente válida. Ésta reconoce como parámetros a “master”, “authenticate” y “encrypt”.

JABWT intentará establecer conexión con el dispositivo indicado y si tiene éxito devolverá un objeto StreamConnection. El objeto StreamConnection permitirá a la aplicación cliente leer y escribir en el servidor.

El identificador del canal de servicios asignado por la implementación JABWT para un servicio es un número entre 0 y 31. Este identificador se almacena en los servicios de registro de atributo ProtocolDescriptionList permitiendo al método getConnectionURL() del ServiceRecord, generar la cadena de conexión a usar.

Ejemplos de cadenas de conexión cliente son:

- “btspp://008003DD8901:1;authenticate=true”,
- “btspp://008012973FAE:5;master=true;encrypt=true”

Después de establecer la conexión y crear el objeto StreamConnection, se hace uso de flujos de entrada y salida (disponibles a través de openInputStream(), openDataInputStream(), openOutputStream() y openDataOutputStream()) para el envío y recepción de datos acompañados siempre del método close().

2.3 OBEX

El protocolo IrOBEX (Infrared Object Exchange Protocol) se definió por la IrDa como una alternativa al protocolo http para sistemas embebidos. IrOBEX permite el procesamiento de datos según se reciban y la eliminación de peticiones o respuestas. Su forma de transporte le hace posible trabajar casi sobre cualquier protocolo y su eficiencia sobre el ancho de banda se basa en su peculiar forma de segmentación de paquetes. La unión entre IrOBEX y Bluetooth se conoce simplemente como OBEX.

El perfil GOEP define la forma de trabajo de OBEX dentro de un ambiente Bluetooth. El API OBEX en JABWT es opcional y por lo tanto, independiente del API Bluetooth.

OBEX permite definiciones claras de una petición a otra y puede ser usado como un medio de intercambio electrónico de tarjetas de negocio o sincronización de sistemas embebidos en computadoras de escritorio entre otros procesos. Su API fue creado con la intención de ser usado en un amplio rango de aplicaciones.

El protocolo provee una plataforma de ejecución ideal para MIDlet's, ya que proporciona formas de recuperar conexiones perdidas, aumentando así la eficacia de la transmisión por aire

Seis de las operaciones básicas de OBEX son: Conectar (Connect), establecer ruta (setpath), obtener (get), insertar (put), descartar (abort), y desconectar (disconnect). El cliente inicia cada operación con una petición y espera la respuesta del servidor.

Dentro de cada petición y respuesta, se insertan cabeceras que proporcionan mayor información tales como:

- Nombre (Debe ser una cadena Unicode)
- Tamaño
- Tipo (MIME)
- Cuenta
- Descripción
- http

En el Cuerpo y Fin de cuerpo se envían o toman objetos del servidor a través de operaciones PUT y GET. END OF BODY le indica al receptor cuándo se ha terminado de transmitir la información.

La especificación OBEX define cómo deben ser codificadas las cabeceras, permitiendo agregar hasta 64 personalizadas, desglosadas en 4 grupos de 16 cabeceras cada uno: cadenas Unicode, enteros no signados de 4 bytes, bytes y secuencia de bytes. Cada grupo representa un tipo diferente de datos.

Dos operaciones adicionales disponibles son: PUT-DELETE y CREATE-EMPTY. La primera refiere a la operación PUT con las cabeceras NAME y no BODY; sirve para informar al servidor sobre el borrado de un objeto bajo un nombre específico. Similarmente, la segunda operación contiene una cabecera END OF BODY y NAME sobre una operación PUT; su función es señalar al servidor la creación de un objeto bajo un nombre específico sin contenido.

Cada sesión OBEX comienza con una petición de conexión. Cuando el servidor la recibe, procesa las cabeceras y decide si la acepta o no. En caso de aceptarla, responde con un código SUCCESS,OK; de lo contrario responde con algún código de respuesta http.

Si al establecer una conexión el cliente necesita re-direccionar la ruta hacia el servidor, la operación SETHPATH se lo permitirá pero dependerá del servidor aceptar o no la petición.

OBEX también permite manipular la forma de envío de los archivos; por ejemplo, que el cliente divida un archivo en segmentos para su envío al servidor si lo considera demasiado grande. Esto se logra a través de la operación PUT con las cabeceras NAME, para especificar el nombre del archivo, y BODY, conteniendo el primer segmento del archivo. Cada vez que el servidor reciba los segmentos, devolverá una respuesta CONTINUE hasta finalizar con éxito la operación.

Las sesiones finalizan en el momento en que se envíe una petición DISCONNECT al servidor. Una vez recibida, se liberarán los recursos asignados y enviará como respuesta a la petición el código OK, SUCCESS al cliente.

Cabe destacar que finalizar una sesión no involucra cerrar una conexión física, por lo que el protocolo de transporte también deberá finalizarse.

2.3.1 API OBEX

El API OBEX para JAVA provee una interfaz de bajo nivel, útil para el control sobre peticiones y respuestas; traduce las cabeceras a su correspondiente representación en bytes y protege las peticiones de conexión omitiendo detalles al usuario.

El API del cliente surge de la combinación entre las interfaces: javax.microedition.io.ContentConnection, javax.microedition.io.DatagramConnection; y javax.obex.Operation (Fig. 2.9)

El API también proporciona mecanismos para la autenticación a través de un esquema reto-respuesta. Cuando una aplicación en un dispositivo remoto intenta autenticarse, se hace uso de la cabecera AUTHENTICATION_CHALLENGE; la contraparte deberá entonces combinar MD5 el secreto compartido o contraseña con los bytes del reto recibido en la cabecera con un algoritmo. El valor resultante es enviado de regreso en una cabecera AUTHENTICATION_RESPONSE. Tanto la cabecera recibida como la creada son enviadas al dispositivo retador para su comparación.

Una diferencia visible en el modo de autenticar sobre bluetooth y el de OBEX radica en que el primero autentica dispositivos y el segundo usuarios o aplicaciones. Además, la autenticación en Bluetooth se lleva a cabo en la pila o en el radio, mientras que en OBEX se realiza en la capa de aplicación; por lo tanto, estos métodos no son excluyentes entre sí.

En cuanto a programación refiere, el API OBEX añade tres nuevas interfaces que enriquecen la ya conocida java.microedition.io.Connection, estas son:

- Java.obex.ClientSession. Interfaz enviado por el método Connector.open() cuando recibe una cadena de conexión del cliente.
- Javax.obex.SessionNotifier. Igualmente es enviada por el método Connector.open() para establecer conexiones con el servidor.
- Javax.obex.Operation. Esta interfaz se usa para procesar peticiones PUT y GET.

Adicionalmente se definen dos interfaces: javax.obex.Authenticator y javax.obex.HeaderSet; y tres nuevas clases: javax.obex.PasswordAuthentication, javax.obex.ResponseCodes y javax.obex.ServerRequestHandler.

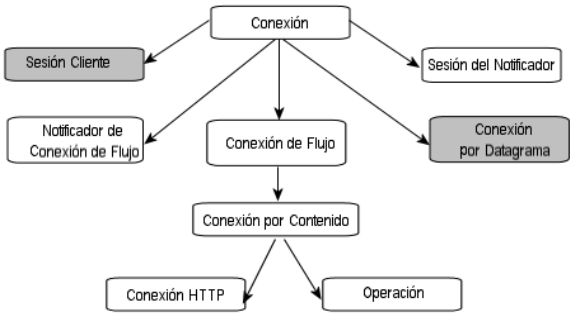


Fig. 2.9 GCF con el API OBEX

2.3.2 Conexión

Las cadenas de conexión aceptadas por el método Connector.open() del API OBEX presenta la forma general:{Esquema} :[{objetivo}][{parámetros}]

Es la compatibilidad con los diferentes protocolos de transporte la que exige especificar cuál de ellos se utilizará en la sección de {Esquema}.

Si se utilizara el protocolo TCP/IP, el esquema sería tcpobex {transporte}obex://{objetivo}{parámetros}. Aquí el objetivo lo especifica la dirección IP o número de puerto del servidor en las conexiones cliente, y el puerto en las de servidor. Este protocolo en el API OBEX carece de parámetros.

Ejemplo de cadenas de conexión cliente válidas son:

- btgoep://00802d5b12af:1;authenticate=yes
- tcpobex://163.10.70.75:1505
- irdaobex://discover;ias=MyAppOBEX,OBEX,OBEX:IrXfer;

Ejemplo de cadenas de conexión de servidor no válidas son:

- btgoep://localhost:1233212ADBAA9324BAFE23331231222C
- tcpobex://:1801
- irdaobex://localhost.0200

Para establecer una conexión cliente en la capa OBEX se debe invocar el método ClientSession.connect() y ClientSession.disconnect() al terminarla. Por el lado del servidor, el objeto SessionNotifier creado por Connector.open(), invocará el método acceptAndOpen(), quien utiliza como argumento ServerRequestHandler y, opcionalmente, Authenticator.

Toda la información proveniente de OBEX se envía a través de cabeceras, entre ellas: BODY, END OF BODY, AUTHENTICATION_CHALLENGE, AUTHENTICATION_RESPONSE y CONNECTION. Para acceder a ellas se hace uso de la interfaz *HeaderSet*.

El HeaderSet no permite implementaciones personalizadas, en su lugar ofrece el método createHeaderSet() con constantes definidas por la mayoría de las cabeceras más la opción de crear hasta 64 más.

Para incluir una cabecera en un objeto se hace uso del método setHeader() con su respectivo identificador y valor de cabecera (Ver Tabla 2.4). En caso de invocar el método de forma incorrecta, se generará una excepción IllegalArgumentException.

Valor de la cabecera	Significado	Tipo
COUNT	Utilizado por CONNECT para especificar el número de objetos que serán comunicados durante la sesión	Java.lang.long
NAME	Nombre del objeto	Java.lang.String
TYPE	Tipo MIME del objeto	Java.lang.String
LENGTH	Tamaño del objeto	Java.lang.Long
TIME_ISO_8601	Fecha del objeto	Java.util.Calendar
TIME_4_BYTE	Fecha del objeto	Java.lang.String
DECRIPTION	Breve descripción del objeto	Byte[]
TARGET	Servicio OBEX destino	Byte[]
HTTP	Especifica una cabecera http	Byte[]
WHO	Servicio OBEX que procesa las peticiones	Byte[]
OBJECT_CLASS	Clase del objeto OBEX	Byte[]
APPLICATION_PARAMETER	Parámetro específico de la aplicación	Byte[]
48 a 63	Cabeceras definidas por el usuario para enviar una cadena	Java.lang.String
112 a 127	Cabeceras definidas por el usuario para enviar un arreglo de bytes	Byte[]
176 a 191	Cabeceras definidas por el usuario para enviar un byte	Java.lang.Byte
240 a 255	Cabeceras definidas por el usuario para enviar un entero no signado dentro del rango de 0-232 a 1	Java.lang.Long

Tabla 2.4 Cabeceras OBEX en la interfaz HeaderSet, significado y tipo

El HeaderSet también facilita la identificación de cabeceras a través del método getHeaderList() quien proporciona el valor por medio de un arreglo de enteros.

La cabecera de identificación es única puesto que sirve para diferenciar los múltiples servicios de un mismo objeto notificador.

Una vez establecida la conexión con el servidor, se establece otra con la capa OBEX. Para ello, el cliente envía una petición de conexión con su respectiva cabecera, luego el objeto del HeaderSet le permite al cliente recibir las cabeceras y el código de respuesta provenientes del servidor. Para responder, el cliente invoca el método getResponseCode() esperando como respuesta del servidor la cadena OBEX HTTP OK.

2.4 Descubrimiento de dispositivos

El descubrimiento de dispositivos depende del descubrimiento de servicios (también denominado "consulta"), ya que la búsqueda se realiza con el objetivo de encontrar servicios activos en los dispositivos remotos.

Las consultas pueden presentarse bajo dos modalidades: general y limitada. La primera rastrea los dispositivos cuyo estado sea visible y la segunda los temporalmente visibles. A una consulta se responde según el modo de detección activado en el dispositivo bluetooth (general ó visible, limitado o temporalmente visible, y no disponible) con la dirección bluetooth y la clase del registro del dispositivo. Las direcciones bluetooth son identificadores asignados por el fabricante y constan de 6 bytes. La clase del registro del dispositivo describe el tipo de dispositivo bluetooth y provee información general sobre los servicios disponibles en el mismo.

El descubrimiento de servicios comienza cuando la aplicación JABWT especifica el tipo de consulta a realizar (general o limitada). Cada vez que se rastrea un nuevo dispositivo, el evento deviceDiscovered() envía información del dispositivo junto con la clase registro.

La clase registro del dispositivo se compone de la clase principal de servicio, la clase principal de dispositivo, y la clase menor de dispositivo. La clase principal de servicios define los servicios disponibles en el dispositivo. Pertenecen a esta clase: Posicionamiento, Red, Captura, Transferencia de Objetos, Audio, Información, Detección Limitada, Renderizado, Telefonía.

Un dispositivo pude servirse de múltiples clases principales de servicio pero sólo de una clase principal de dispositivo. Las clases principales de dispositivo definidas por el SIG son: Computadora, Teléfono, LAN/Network Access Point, Audio/Video, Periféricos, Imágenes, Misceláneas.

La clase menor de dispositivo únicamente proporciona una descripción más específica del dispositivo, permitiendo a los desarrolladores realizar búsquedas más selectivas.

JABWT agiliza la comunicación al proporcionar una lista de dispositivos probablemente disponibles en un área sin la necesidad de realizar una consulta. Esos dispositivos "predefinidos" se clasifican como: reconocidos y registrados.

Los dispositivos reconocidos son aquellas terminales con las que el dispositivo local interactúa frecuentemente; los dispositivos registrados son aquellos dispositivos descubiertos en alguna consulta previa.

2.4.1 Programación

La clase dispositivo local provee acceso al dispositivo, así como métodos para la extracción de información y la facultad de utilizar diferentes procesos de detección de dispositivos.

Para que una aplicación pueda extraer objetos LocalDevice de una implementación JABAWT debe invocar el método Local.Device.getLocalDevice(). En caso de que la pila o radio Bluetooth no funcionen correctamente, el método enviará una excepción BluetoothStateException.

La creación del objeto LocalDevice permite la recopilación de información adicional del dispositivo local tal como: dirección bluetooth -getBluetoothAddress()-, nombre –getFriendlyName()-, modo de descubrimiento –getDiscoverable()- y la clase registro del dispositivo –getDeviceClass()- Cualquier conflicto concerniente con peticiones de descubrimiento será resuelto por el BCC

La clase DiscoveryAgent proporciona las constantes para el descubrimiento general (GIAC, General Inquiry Access Code), descubrimiento limitado (LIAC, Limited Inquiry Access Code) y modo no descubrible (NOT_DISCOVERABLE), útiles para el método setDiscoverable quien tomará estos valores como argumentos o cualquier otro modo de descubrimiento que se encuentre dentro del rango 0x90x9E8B00-0x9E8B3F. Ejemplo de su uso es setDiscoverable(DiscoveryAgent.GIAC). Se espera un valor igual a true si el BCC acepta el cambio del modo de descubrimiento y false en caso contrario. Si el cambio no se puede realizar en el momento la aplicación responderá con una excepción BluetoothStateException.

getProperty() es otro método perteneciente a la clase LocalDevice. Su uso proporcionará información detallada del dispositivo local. La tabla 2.5 muestra los parámetros admitidos para este método así como el resultado esperado tras convocarlo.

Propiedad	Descripción	Valores válidos
Bluetooth.api.version	Versión JABWT soportada. Número de especificación bluetooth	1.0 para la versión actual de JABWT
Bluetooth.master.switch	¿Maestro/Esclavo permitido?	Verdadero (true) o falso (false)
Bluetooth.sd.attr.retrivable.max	Número máximo de atributos del servicio que pueden extraer por registro de servicio.	Entero de base 10
Bluetooth.connected.devices.max	Número máximo de dispositivos conectados soportados	Entero de base 10
Bluetooth.l2cap.receive.MTU.max	Tamaño máximo en bytes del MTU soportado en L2CAP	Entero de base 10
Bluetooth.sd.trans.max	Número máximo de transacciones simultáneas para el descubrimiento de servicios	Entero de base 10
Bluetooth.connected.inquiry.scan	¿El dispositivo puede responder a consultas cuando ya ha establecido conexión con otro dispositivo?	Verdadero (true) o falso (false)
Bluetooth.connected.page.scan	¿Puede el dispositivo local aceptar una conexión de un dispositivo remoto estando conectado con otro?	Verdadero (true) o falso (false)
Bluetooth.connected.inquiry	¿Puede el dispositivo local iniciar una consulta mientras sostiene una conexión con otro dispositivo?	Verdadero (true) o falso (false)
Bluetooth.connected.page	¿Puede el dispositivo local establecer una conexión de un dispositivo remoto estando conectado con otro?	Verdadero (true) o falso (false)

Tabla 2.5 Propiedades disponibles a través de localDevice.getProperty

2.4.2 Descubrimiento simple

JABWT permite descubrir servicios y dispositivos a través de la clase DiscoveryAgent y la interfaz DiscoveryListener.

DiscoveryAgent proporciona métodos para inicializar la búsqueda de servicios y dispositivos. La clase se sirve de la interfaz DiscoveryListener para informar a la aplicación sobre la aparición de dispositivos o servicios y notificar cuando la consulta o búsqueda de servicios haya sido completada. El método LocalDevice.getDiscoveryAgent() es el encargado de obtener los objetos DiscoveryAgent asociados al dispositivo local.

La clase DiscoveryAgent facilita el listado de dispositivos remotos a través de su método retrieveDevices(). Los parámetros aceptados por el método son DiscoveryAgent.CACHED para aquellos dispositivos con los que nos se interactúa comúnmente y DiscoveryAgent.PREKNOWN si los dispositivos ya listados anteriormente. La cantidad de dispositivos reconocidos y por cuánto tiempo depende de la implementación.

2.4.3 Descubrimiento de dispositivos por consulta

El proceso de consulta comienza al invocar el método startInquiry() más alguno de los parámetros permitidos para este método (DiscoveryAgent.GIAC y DiscoveryAgent.LIAC o algún valor definido de dentro del rango 0x9E8B00 a 0x9E8B3F). Como resultado se espera un valor verdadero si la consulta si inicializa con éxito o falso si el código de acceso de la consulta no se acepta. La excepción IllegalArgumentException es arrojada cuando se manejan códigos de acceso de consulta no válidos o una excepción BluetoothStateException cuando el estado del dispositivo impide completar la consulta.

Después de invocar startInquiry(), JABWT devuelve a la aplicación los dispositivos encontrados a través de los eventos deviceDiscovered(), Gracias a estos eventos se obtienen los objetos RemoteDevice y DeviceClass asociados.

El objeto DeviceClass proporciona la dirección bluetooth, así como los métodos de recuperación de nombre y controles de seguridad de los dispositivos remotos.

Para cancelar la consulta se invoca método cancelInquiry(). Si la cancelación se procesa con éxito el método devuelve el valor true, en caso contrario se espera un valor false.

Finalmente, el método inquiryCompleted() notificará a la aplicación el término de la consulta.

Para abortar una consulta antes de que se concluya es necesario añadir código a la aplicación donde el método commandAction() invoque al método cancelInquiry().

2.4.4 Obtención de información de un dispositivo remoto

Existen tres formas de obtener información de un dispositivo remoto a través de objetos:

- A través del proceso de descubrimiento. Los objetos se pasan a través de las aplicaciones como argumentos a través de los eventos deviceDiscovered().

- A través de una clase. Las extensiones del RemoteDevice pueden ser leídas e instanciadas por una aplicación.
- Usando el método estático RemoteDevice.getRemoteDevice(). El método devuelve un objeto representando el dispositivo con el que se realiza la conexión Bluetooth.

2.4.5 Clase DeviceClass

La clase DeviceClass es un objeto único que provee tres métodos de acceso a la clase del registro del dispositivo, el cual especifica tanto el tipo físico como los servicios que este ofrece:

- getServiceClasses() Obtiene un entero representando el número de las clases de servicio disponibles en el dispositivo.
- checkForRenderingService() Aísla el bit del servicio de renderizado haciendo uso de la operación lógica AND entre la clase de servicios del dispositivo y un número con el bit 18 encendido. La tabla 2.6 muestra las clases de servicio principales definidos por el SIG, el bit que debe ser encendido y su valor hexadecimal.

Servicio	Tipo de Servicio	Número de bit	Valor hexadecimal
Modo de Descubrimiento Limitado	El dispositivo está activado en el modo de descubrimiento limitado	13	0x2000
Posicionamiento	Identificación de la Localización	16	0x10000
Red	LAN, ad-Hoc, etc.	17	0x20000
Renderizado	Impresión, bocinas	18	0x40000
Captura	Scanner, micrófono, etc.	19	0x80000
Transferencia de Objeto	V-Inbox, V-Folder	20	0x100000
Audio	Bocinas, micrófono, auricular, etc.	21	0x200000
Telefonía	Telefonía inalámbrica, modem, auricular	22	0x400000
Información	Servidor web, servidor WAP, etc.	23	0x800000

Tabla 2.6 Clases principales de servicios definidas por el SIG

- getMajorDeviceClass()/getMinorDeviceClass().Devuelve el valor de la clase principal o menor del dispositivo. Las clases de dispositivo principales reportan el tipo físico del dispositivo al que se conecta la pila Bluetooth. La tabla 2.7 muestra las clases principales de dispositivo definidas por el SIG.

La clase menor de dispositivo aportada por el método getMinorDeviceClass() proporciona información más detallada del dispositivo remoto. Las clases menores con la base de las mayores.

Clase Principal de Dispositivo	Ejemplo	Valor hexadecimal
Computadora	PC, notebook, PDA, Organizador	0x100
Teléfono	Celular, modem, telefonía inalámbrica o de pago	0x200
Access Point		0x300

Audio/Video	Auricular, bocina, estéreo, despliegue de video, VCR	0x400
Periféricos	Mouse, joystick, teclado	0x500
Imagen	Impresora, escáner, cámara, pantalla	0x600
Miscelánea	Otros dispositivos	0x000

Tabla 2.7 Clase Principal de Dispositivo definida por el SIG

2.5 Descubrimiento de servicios

El descubrimiento de dispositivos se da durante la búsqueda de servicios. Si un dispositivo remoto posee algún servicio solicitado por otro dispositivo, éste devuelve el registro del servicio con su descripción a través de atributos. Los registros de servicio son almacenados en la SDDB (Service Discovery Database)

El descubrimiento de servicios sigue el modelo cliente-servidor; el cliente gestiona las peticiones de búsqueda en los servidores y el servidor determina si los servicios encontrados cumplen los criterios de búsqueda de las peticiones.

Si un servicio es definido por un perfil bluetooth, entonces la especificación del perfil describirá los requerimientos del registro de servicios, la seguridad, los modos de descubrimiento, etc. Las aplicaciones que no provean servicios descritos en algún perfil bluetooth, no necesitan pasar por el proceso de certificación.

La búsqueda de servicios va acompañada de un conjunto de UUID´s; esto es una secuencia de bits que identifican de forma única las características de un servicio. Una vez encontrado, se envía el registro del servicio a través del evento servicesDiscovered().

Por su parte, el método selectService() permite especificar un único UUID necesario para localizar el servicio demandado en el dispositivo remoto. Como respuesta, el método devuelve una cadena de conexión que en conjunto con el método Connector.open() , permite realizar la conexión. Es importante destacar que este método puede prolongarse un tiempo mayor a los 10 segundos en algunos casos.

2.5.1 Servicios " Ejecutar antes de conectar"

Comúnmente las aplicaciones de servidor se encuentran en ejecución y listas para una conexión antes de que el cliente realice una petición. Este tipo de requerimiento es conocido como servicios de ejecución antes de conexión. El diagrama de secuencia de la figura 2.10 ilustra los mensajes relacionados con estos tipos de servicios durante su registro.

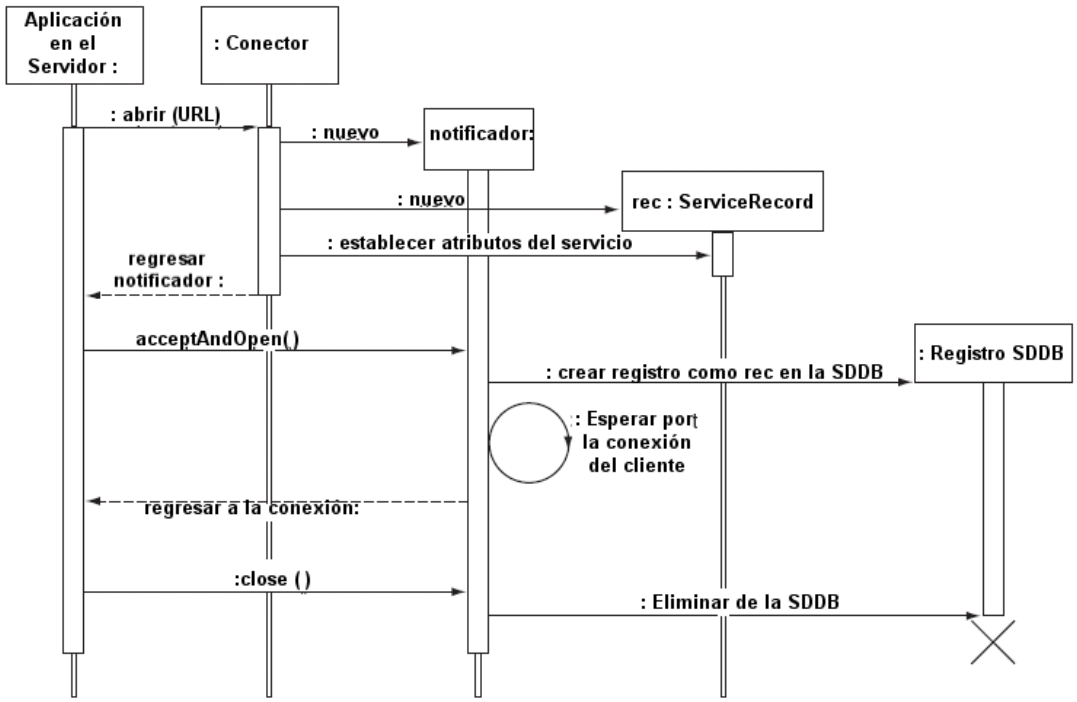


Fig. 2.10 Ciclo de vida de un registro de servicio "ejecutar antes de conectar"

En la mayoría de las aplicaciones se opta por el modo de descubrimiento general o limitado mas existen algunos escenarios en los que el modo no-descubrible se hace más factible; como cuando se conoce la existencia del dispositivo y su dirección. Cuando esto sucede, deben suponerse medidas alternas para las conexiones.

2.5.2 Registro de servicios

JABWT crea automáticamente registros de servicios para aplicaciones de servidor para agilizar la comunicación y facilitar a los desarrolladores decidir si un registro es suficiente para publicar un servicio y determinar la forma de modificarlo cuando sea necesario.

El SIG ha definido perfiles que proporcionan servicios estandarizados. Bajo esta modalidad, las especificaciones del perfil definirán los requerimientos para el registro de servicios, seguridad, modos de descubrimientos, etc.

Más allá de lo ya definido por el SIG, los desarrolladores pueden definir aplicaciones para el servidor y crear servicios para dispositivos remotos. En caso de desear integrar la aplicación a un perfil Bluetooth, la aplicación tendrá que pasar por un proceso de certificación.

Un registro de servicios se compone de un conjunto de atributos que especifican la forma de conexión al servicio, el nombre, su descripción e información importante. Para identificarlo, la aplicación deberá proveer un conjunto de UUID's.

Cuando el servicio con el UUID especificado se localiza, se devuelve el registro de servicio a través del evento serviceDiscovered().

Registros de servicio por puerto serial

Al ejecutar la sentencia:
Connector.open("btspp://localhost:68EE1441812D211D78EED00B0D03D76EC;name=SPPEx");

Se crean cuatro pares de valores, cada par describe un atributo del servicio. El SDP utiliza valores que van desde 0 hasta 2^{16}-1 para representar un identificador en el registro del servicio. La tabla 2.8 muestra una lista de los atributos más comunes para identificadores y su equivalencia en hexadecimal.

Cada valor del atributo es un DataElement; esto es, una estructura de datos autodescriptiva que contiene un valor y un tipo. Un DataElement puede darse como un valor: nulo, entero (1,2,4,8 y 16 bytes), entero no signado, URL, UUID, cadena, booleano, DATALT (Alternativa de Elemento de Datos), DATSEQ (Secuencia de Elementos de Datos).

Nombre	ID	Tipo	Descripción
ServiceRecordHandle	0x0000	Entero no signado de 32 bits	Identifica cada servicio en un dispositivo
ServiceClassIDList	0x0001	UUID's	Define las clases que describen servicios
ServiceID	0x0003	Secuencia de elementos de datos de UUID's	Identifica las instancias de servicios
ProtocolDescriptionList	0x0004	Secuencia de elementos de datos de una secuencia anterior de UUID's y parámetros opcionales	Describe el protocolo a usarse para conectarse a un servicio
ServiceInfoTimeToLive	0x0007	Entero no signado de 32 bits	Define el periodo de tiempo en que el servicio es válido y permanecerá inmutable
ServiceAvailability	0x0008	Entero no signado de 8 bits	Define la disponibilidad relativa de los servicios con el fin de aceptar conexiones adicionales
BluetoothProfileDescriptorList	0x0009	Secuencia de elementos de datos	Especifica los perfiles implementados por el servicio
DocumentationURL	0x000A	URL	URL que apunta a la documentación del servicio
IconURL	0x000C	URL	URL que apunta a un icono usado para representar un servicio

ServiceName	0x0100	Cadena	El nombre del servicio en el lenguaje del registro
ServiceDescription	0x0101	Cadena	Descripción de un servicio en el lenguaje del registro

Tabla 2.8 Atributos para identificadores del registro de servicios

Registro de servicios para L2CAP

Los protocolos L2CAP y OBEX se utilizan como alternativa a las comunicaciones seriales a través de btspp. La comunicación con el servidor bajo estos protocolos se realiza a través del método connector.open() con un argumento de la forma:

Bt12cap://localhost:BA661F1C148911D783C300B0D03D76EC; name=An L2CAP Server"

L2CAP es el último elemento de la ProtocolDescriptorList en describir la secuencia de capas involucradas en la obtención de una aplicación del servidor. Las aplicaciones que trabajan sobre esta capa necesitan del protocolo y el parámetro del multiplexor de servicios. Los servicios a su vez, son identificados con la ayuda del PSM.

Registro de servicios para OBEX sobre RFCOMM

Ejemplo de una cadena de conexión utilizada para este protocolo es:

Btgoep://localhost :BA661F1C148911D783C300B0D03D76EC; name=Am OBEX Server

El sufijo goep hace referencia al Perfil de Intercambio Genérico de Objetos, el cual funciona como base para los demás perfiles obex. Aquí no es L2CAP sino OBEX el último elemento del ProtocolDescriptorList, pues es éste quien tiene comunicación directa con la capa OBEX en la pila bluetooth. Cabe destacar que el ServiceClassIDList excluye el identificador de la clase del servicio por puerto serial incluido en btspp. La tabla 2.9 muestra las variaciones entre los diferentes protocolos.

Protocolo	Interfaz	Método	Especificación
btspp	Connector	Open(url) Open(url, modo) Open(url, modo, tiempo de espera)	CLDC
bt12cap	Connector	Open(url) Open(url	CLDC
btgoep	Connector	Open(url) Open(url	CLDC

Tabla 2.9 Métodos para la creación del registro del servicio

Los registros creados por el método connector.open() no son visibles para los dispositivos cliente, por ello las implementaciones JABWT deben añadir el registro a la SDDB la primera vez que lo invoquen.

Una vez descubiertos los servicios, se debe establecer conexión con los clientes a través del método acceptAndOpen(). Todos ellos accederán el mismo registro y se conectarán con el servidor a través del canal RFCOMM.

La tabla 2.10 muestra las variantes del método acceptAndOpen() bajo los diferentes protocolos de conexión.

Protocolo	Interfaz	Método	Especificación
btspp	StreamConnectionNotifier	acceptAndOpen()	CLDC
btl2cap	L2CAPCOnnectionNotifier	acceptAndOpen()	JABWT
btgoep	SessionNotifier	acceptAndOpen(controlador) acceptAndOpen(controlador, autenticador)	JABWT

Tabla 2.10 Métodos para añadir registros de servicio al SDDB

Después de cerrar el notificador asociado al servicio de ejecución antes de conexión, no es posible volver a invocar el método acceptAndConnect(); por esta razón las implementaciones JABWT deben eliminar o deshabilitar el registro de servicios de la SDDB. La tabla 2.11 muestra el método necesario para llevar a cabo esta tarea bajo los diferentes protocolos anteriormente mencionados.

Protocolo	Interfaz	Método	Especificación
btspp	StreamConnectionNotifier	Close()	CLDC
btl2cap	L2CAPConnectionNotifier	Close()	CLDC
btgoep	SessionNotifier	Close()	CLDC

Tabla 2.11 Métodos para eliminar registros de servicio al SDDB

2.5.3 Modificación del registro de servicios

En caso de necesitar modificar el registro de servicios, se cuenta con atributos opcionales descritos en la especificación SDP y se pueden definir otros más a través de aplicaciones.

Cualquier modificación hecha al ServiceRecord antes de invocar el método acceptAndOpen() por primera vez, se verá reflejada en el registro de servicios añadido al SDDB por acceptAndOpen().

En forma general, JABWT permite modificar los registros de servicios ya almacenados en la SDDB haciendo uso del método LocalDevice.updateRecord(serviceRecord).

2.5.4 Clases para los servicios del dispositivo

Existen dos diferentes formas en que una aplicación describe los servicios que ofrece:

- Añade el registro de servicios a la SDDB
- Activa los bits más representativos de la clase en la DeviceClass

La activación de bits se hace a través del método *setDeviceServiceClasses()* del *ServiceRecord*. Una aplicación de servidor no necesita invocar este método; sin embargo, es útil a la hora de establecer su servicio como el mejor.

Para desactivar los bits de la DeviceClass se utiliza el método close(). No obstante, en caso de que algún otro servicio haya intervenido en la activación de bits y continúe activo, el servicio no podrá ser desactivado.

2.5.5 Soporte para cadenas de atributos en diferentes idiomas

El SDP Bluetooth hace uso de un esquema "base más offset" para definir todos los atributos de un servicios del tipo cadena. La especificación SDP define un offset de 0x0001 para el ServiceDescription y un valor hexadecimal variable para cada lenguaje, p. ej. ID para el ServiceDescription en inglés= 0x0100 (valor base para la lengua inglesa)+0x0001(offset)=0x0101.

El LanguageBaseAttributeIDList contiene los valores base para cada idioma y es considerado como un atributo para servicios opcional. Los elementos en esta lista se presentan en triadas, el primer elemento representa el código del idioma; el segundo, su carácter cifrado (p. ej. UTF-8); el tercer el valor base del ID del atributo.

Los registros de servicios pueden variar los valores base para la misma lengua, las únicas reglas a seguir son las siguientes:

- La base para el ID del atributo del idioma principal usada en un registro de servicios particular debe ser 0x0100
- El primer idioma registrado en el LanguageBaseAttributeIDList debe ser el principal; es decir, el x0100.
- La base del ID del atributo usada para el segundo idioma no se encuentra estandarizada. Una recomendación es escoger una base para el ID del atributo de tal manera que al ser sumado con el offset se tenga como resultado un valor dentro del rango de 0x0100 a 0x01FF o del 0x0900 al 0xFFFF.

2.5.6 Descubrimiento de servicios

El descubrimiento de servicios se inicia cuando el dispositivo local envía la lista de UUID's al dispositivo remoto. Éste último responde con un objeto ServiceRecordHandle y los atributos del servicio demandados. Cada vez que se encuentre un servicio, se irá registrando en las aplicaciones como un evento.

El método utilizado en la búsqueda de servicios es DiscoveryAgent.searchServices() y los cuatro argumentos acepta son:

- Una lista de atributos extraída de cualquier registro de servicios que cumpla con los criterios de búsqueda establecidos, por defecto: ServiceRecordHandle, ServiceClassIDList, ServiceRecordState, ServiceID y ProtocolDescriptorList.
- Una lista de UUID's especificando el servicio que está buscando en el dispositivo remoto.

El dispositivo remoto a localizar, esto es, un objeto RemoteDevice resultado de la búsqueda o llamada del método retrieveDevices().

Los servicios descubiertos, el ID de la transacción y el registro de servicios serán enviados al DiscoveryListener a través del método servicesDiscovered().

El registro de servicios se presenta como un arreglo de objetos ServiceRecord. Cada uno de ellos contiene los atributos demandados en el método searchServices() y los que se asocian por defecto.

Cuando la búsqueda de servicios se completa, se invoca el método serviceSearchCompleted quien proveerá el identificador de la búsqueda y un código de estado.

La búsqueda de servicios puede ser abortada a través del método cancelServiceSearch(). Como argumento válido recibe el ID de la búsqueda.

A continuación se presenta la tabla con los códigos utilizados durante la búsqueda de servicios.

Estado	Razón
SERVICE_SEARCH_COMPLETED	Al menos un registro fue encontrado y la búsqueda se completa normalmente
SERVICE_SEARCH_TERMINATED	La búsqueda es cancelada a través de cancelServiceSearch()
SERVICE_SEARCH_ERROR	Error durante la búsqueda de servicios
SERVICE_SEARCH_NO_RECORDS	Ningún registro se encontró durante la búsqueda
SERVICE_SEARCH_DEVICE_NOT_REACHABLE	El RemoteDevice especificado en searchServices() no está disponible

Tabla 2.12 Códigos de estado para la búsqueda de servicios

2.5.7 Procesamiento de registros de servicios

Después de obtener el ServiceRecord se debe determinar si el servicio descrito es el requerido. Luego, el método getConnectionURL() obtendrá la cadena de conexión que Connector.open() utilizará para establecer la conexión. El método getConnectionURL() también permite especificar los requerimientos de seguridad y la configuración del dispositivo local (maestro o esclavo).

El valor de los atributos que se encapsularán en el DataElement son visibles a través del método getAttributeValue(). Su llamado devolverá el ID del atributo especificado y en caso de no hallar semejante valor, uno nulo. Si lo que se desea no es acceder al valor sino al tipo de dato que representa el atributo, el método getDataType() debe ser invocado.

Una vez analizado el valor del servicio, los objetos ServiceRecords devueltos a la aplicación local se almacenarán en un Vector con el fin de reunir información concerniente al servicio en cualquier otro momento.

El ServiceRecord optimiza la cantidad de datos a enviar evitando almacenar todos los atributos del registro de servicios del dispositivo remoto. Si alguno hiciera falta después de haber completado la búsqueda, JABWT ofrece el método populateRecord(). El uso de este método puede afectar la experiencia del usuario ya que hasta no ser completado, bloqueará la aplicación en curso.

2.5.8 Descubrimiento simple de servicios y dispositivos

El descubrimiento de servicios y dispositivos se realiza a través del método selectService(). Su ejecución creará una cadena de conexión que el método connector.open() utilizará para establecer una conexión con el servicio.

El método selectService() acepta tres argumentos:

- UUID a identificar en la ServiceClassIDList (el SIG Bluetooth ha reservado un rango de valores que van de 00000000-0000-1000-8000-00805F9B34FB, incluyendo todos los valores 2^{12}, hasta FFFFFFFF-0000-1000-8000-00805F9B34FB).
- Los requerimientos mínimos de seguridad para la conexión;
- La configuración del dispositivo local (maestro o esclavo).

Para efectos de eficacia, selectService() debe insertarse en un hilo diferente al de la conexión.

2.5.9 Servicios de conexión instantáneos

Algunos dispositivos pueden iniciar aplicaciones no activas en el servidor al habilitar la opción "on demand" durante la conexión. Las aplicaciones así habilitadas reciben el nombre de "servicios de conexión instantáneos" en la especificación JABWT.

Estas aplicaciones aprovechan las capacidades del MIDP para iniciar automáticamente MIDlet's en el servidor durante las conexiones de los clientes.

2.6 L2CAP

L2CAP (Fig. 2.11) funciona como un control del enlace lógico y protocolo de adaptación. Es una capa multiplexora compatible con protocolos de alto nivel y aplicaciones destinadas a la comunicación bluetooth (útil sólo para el intercambio de datos); cuenta con el TCS para encargarse del señalamiento del control de llamadas de voz y datos, y de forma opcional con BNEP para la transmisión de paquetes IP compatibles con el perfil PAN.

Los perfiles de acceso LAN y Dial-Up se basan en el protocolo RFCOMM, en vez del BNEP, para ingresar a la pila del protocolo bluetooth.

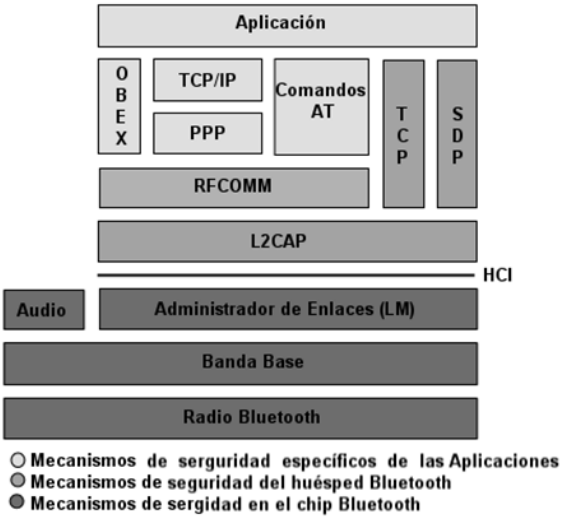


Fig. 2.11 Posición de la capa L2CAP en la pila Bluetooth

2.6.1 Canal y paquetes L2CAP

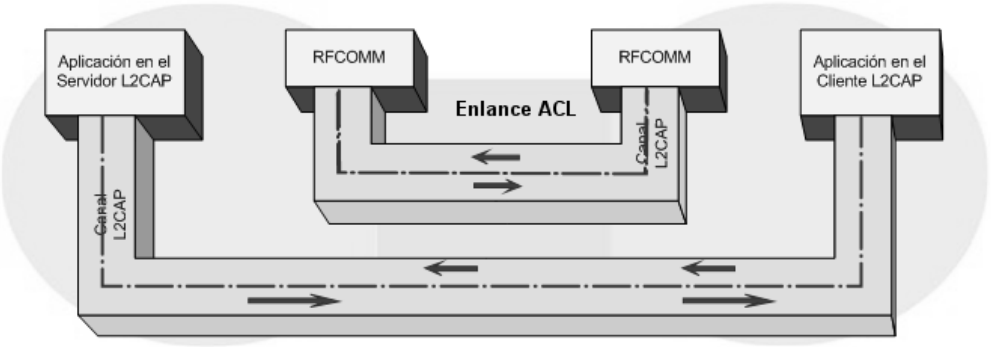


Fig. 2.12 Canales L2CAP durante la transmisión de paquetes

La figura 2.12 muestra la comunicación entre dos dispositivos. Del lado izquierdo se encuentra el cliente y del derecho el servidor, entre ellos se establece canales de conexión L2CAP para entidades de alto nivel. Las flechas representan los paquetes de datos durante la comunicación y su tamaño representa la cantidad de información en tránsito. La imagen muestra que bajo L2CAP es posible establecer una comunicación full-dúplex.

El tamaño de los paquetes está limitado por una *unidad máxima de transmisión* MTU. Cuando el canal se establece, los valores del MTU son renegociados por ambos componentes (la aplicación de servidor L2CAP y la de cliente).

La capa banda base de la pila de protocolos establece el enlace ACL, y sólo se crea por pareja de dispositivos. Este enlace y las funciones banda base proporcionan la infraestructura necesaria para soportar canales y paquetes L2CAP de alto nivel.

Los paquetes L2CAP deben ser transformados en paquetes banda base para su transmisión sobre el enlace ACL. El dispositivo receptor estará encargado de re-ensamblar estos paquetes y convertirlos en paquetes L2CAP.

Ya que los paquetes banda base son pequeños, sus detalles pasan inadvertidos a los altos niveles de la pila y a aplicaciones en el proceso de segmentación y re-ensamblado.
L2CAP también provee canales sin conexión además de los orientados a conexión.

Razones para utilizar L2CAP

Múltiples protocolos se definen sobre L2CAP. La tabla siguiente muestra algunos de estos protocolos.

Protocolos que usan L2CAP directamente	Perfiles dependientes del protocolo
Especificación del Protocolo para el Control de la Telefonía	Telefonía inalámbrica Enterco
Protocolo de Encapsulación de Redes Bluetooth	PAN Descubrimiento de Servicios Extendido
Canal de Control de Copias (Impresas)	Reemplazo de Cable para Copias Gráficas
Canal para la Notificación de Copias (Impresas)	
Protocolo de Transporte para el Control de Audio/Video	Control Remoto de Audio/Video
Protocolo de Transporte de Distribución de Audio/Video	Distribución de Audio/Video Genérico Distribución de Audio Avanzado
RFCOMM	Puerto Serial Red Dial-Up FAX Auriculares Manos libres
Protocolo para el Descubrimiento de Servicios	Aplicación para el Descubrimiento de Servicios

Tabla 2.13 Protocolos definidos sobre L2CAP

La primera columna lista los perfiles bluetooth actuales; la segunda los que dependen de la primera columna. La división separa los perfiles que pueden ser implementados en aplicaciones que dependan del API L2CAP de las que hacen uso del protocolo y API RFCOMM.

El API para el descubrimiento de servicios JABWT provee las capacidades clave del SDAP.

En la mayoría de las aplicaciones personalizadas se opta por el uso de RFCOMM y OBEX por las ventajas que estos pueden ofrecer; sin embargo, cuando se hace necesario el control del tráfico de datos, específicamente el tamaño de los paquetes, el uso del protocolo y API L2CAP será más conveniente.

2.6.2 Abrir conexiones L2CAP

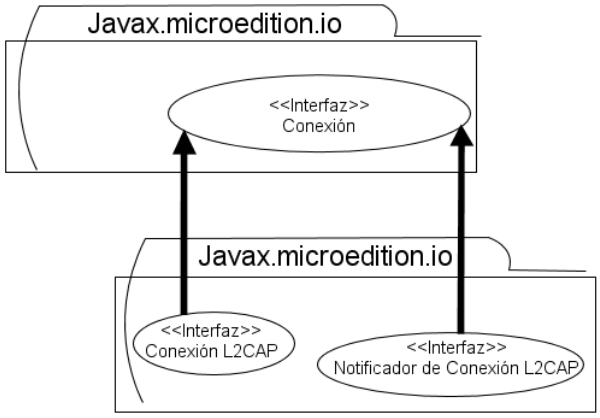


Fig. 2.13 Interfaces JABWT para L2CAP

El L2CAPConnectionNotifier le permite al servidor L2CAP mantenerse en espera de una petición de conexión. Si el cliente demanda una conexión, el notificador creará un objeto L2CAPConnection con el que se proveerá acceso al canal L2CAP.

Debido a que L2CAP utiliza una conexión orientada a paquetes en vez de a flujo, fue necesario re-definir los métodos send() y receive() de tal manera que acepten arreglos de bytes como argumentos en vez de datagrama. Cada conexión L2CAP o instancia L2CAPConnection definirá una aplicación de envío y recepción únicos.

El método receive() mantendrá el sistema bloqueado hasta haber finalizado la lectura de paquetes o cerrado el canal de comunicación. Para saber si este método se encuentra en ejecución, se hace uso del método read(). El valor que devuelva este último indicará la existencia (un valor verdadero) o inexistencia (un valor falso) de un paquete de lectura pendiente sin bloqueos.

Ejemplo de cadenas de conexión válidas son:

- Btl2cap://0050CD00321B:1001
- Btl2cap://0050CD00321B:1001; recieveMTU=512; transmitMTU=512
- Btl2cap://0050CD00321B:1001; authenticate=true; encrypt=true

“btl2cap” hace referencia al protocolo, “0050CD00321B” indica la dirección del dispositivo servidor; y “1001” el valor PSM de la aplicación (se obtiene del registro de servicios).

El PSM indica a la capa L2CAP del dispositivo remoto la aplicación de servidor que necesita el cliente como destino del nuevo canal L2CAP.

Los protocolos de alto nivel tienen valores PSM para L2CAP permanentemente asignados. Estos son generados dinámicamente por una aplicación.

Los PSM en cadenas de conexión btl2cap son interpretados como números hexadecimales mientras que los identificadores del canal de servicios en las cadenas de conexión btspp y btgoep son interpretados como decimales.

Tal como se había mencionado anteriormente, la capa L2CAP permite controlar el tamaño de los paquetes. Del lado del servidor, el parámetro receiveMTU controlará el tamaño en bytes de la

carga del paquete más grande que se aceptará y por parte del cliente es transmitMTU quien definirá el paquete más grande en bytes a enviar.

Las aplicaciones L2CAP generalmente obtienen sus cadenas de conexión al instanciar alguno de los siguientes métodos:

- Service.Record.getConnectionURL(int requieredSecurity, boolean master)
- DiscoveryAgent.selectService(UUID uuid, int security, Boolean master)

En caso de ser necesario se deben anexar los parámetros transmitMTU o receiveMTU a las cadenas resultantes de la ejecución de estos métodos.

Algunos ejemplos de cadenas de conexiones simples como con los parámetros transmit/receive MTU son:

- "btl2cap//lcoalhost:9C68A2AA1EC011D796E6C00B0D03D76EC; master=true"
- "btl2cap://localhost:9C68A2AA1EC011D79E6C00B0D03D76EC; receiveMTU=512"
- "btl2cap//lcoalhost:9C68A2AA1EC011D796E6C00B0D03D76EC; receiveMTU=512;"+ " transmitMTU=1024;"
- "btl2cap://localhost:9C68A2AA1EC011D79E6C00B0D03D76EC;authenticate=true;"

2.6.3 Configuración del canal L2CAP

LocalDevice.getProperty("bluetooth.l2cap.receiveMTU.max") define la carga más grande que puede recibir la pila bluetooth pero es la aplicación misma quien define si se ha de trabajar con el valor máximo o con uno de menor tamaño.

Es recomendable no modificar el valor de los parámetros receive/transmitMTU y trabajar con los predeterminados a menos que sea absolutamente necesario, ya que podrían suscitarse fallas de correspondencia como un valor $transmitMTU_A$ mayor que un valor $receiveMTU_B$ o un valor $transmitMTU_B$ mayor que $receiveMTU_A$ (donde A y B representan dispositivo A y B respectivamente)

Los valores MTU pueden establecerse antes y después de una conexión; no obstante, es preferible considerárseles como requerimientos no negociables. Éstos responden a 4 reglas para el desarrollo de código:

Regla 1

Los valores para los parámetros MTU en una cadena de conexión L2CAP no deben ser menores que L2CAPConnection.MINIMUM_MTU y no mayores que 65535.

Regla 2

El valor para receiveMTU debe ser menor o igual a LocalDevice.gerProperty("bluetooth.l2cap.receiveMTU.max"), la carga más grande que un paquete L2CAP puede recibirse en una pila bluetooth del dispositivo donde se esté ejecutando la aplicación.

Regla 3

Para la transmisión de paquetes de salida con send(byte[] outBuf), el arreglo de bytes outBuf no debe ser mayor que L2CAPConnection.getTransmitMTU().

Regla 4

Para la recepción de paquetes de entrada a través de send(byte[] inBuf), se debe reservar un arreglo de bytes del tamaño de L2CAPConnection.getReceiveMTU().

L2CAP no provee algún mecanismo de control de flujo para crear una transmisión confiable, en su lugar actuará la capa de enlace ACL, una capa inferior a L2CAP. Este procedimiento no es suficiente por lo que puede sucederse que los paquetes lleguen más rápido de lo que se procesan y los buffers se saturen. En estos casos las únicas opciones posibles son:

- Dejar que las capas inferiores resuelvan el problema activando y bloqueando paquetes sobre el enlace ACL
- Desechar paquetes.

Por estas insuficiencias es necesario que los protocolos y perfiles en las comunicaciones L2CAP posean sus propios mecanismos de control de flujo.

2.6.4 Tipos de aplicación

La aplicaciones para L2CAP pueden ser implementaciones de perfiles Bluetooth o personalizadas. Los pormenores dependerán del tipo de aplicación planeada.

Las aplicaciones basadas en perfiles bluetooth, por ejemplo, deben apegarse a las pruebas de certificación para asegurar la interoperabilidad con otros dispositivos. Algunas veces incluso es necesario establecer valores MTU para los canales L2CAP. Por su parte, aunque las aplicaciones personalizadas no requieren de un proceso de certificación precisan de un conocimiento previo del perfil; además, se deben establecer los esquemas de control de flujo que los perfiles y protocolos bluetooth adoptarán.

2.7 Implementación de JABWT

El uso de JABWT está destinado a:

- Fabricantes de dispositivos J2ME que planeen añadir tecnología inalámbrica a sus dispositivos.

- Fabricantes de dispositivos bluetooth que requieran extender las capacidades de programación de sus dispositivos a un vasto número de programadores java.

El API JABWT surge de la fusión entre la tecnología inalámbrica Bluetooth y el lenguaje de programación JAVA por lo que su implementación estará ligada tanto al hardware (pila y componentes físicos del dispositivo bluetooth), como al software (perfiles, sistema operativo, KVM, etc.).

2.7.1 Proceso de conversión (Porting Process)

Los requerimientos necesarios para ejecutar procesos de conversión son:

- Contar con una implementación CLDC/KVM, DC/JVM o J2SE+JSR-197
- Integrar pila y radio para protocolos bluetooth.
- Implementar JABWT y BCC
- Ejecutar el TCK para revisar que la implementación cumpla con las especificaciones de JABWT

Las características de cada dispositivo dictarán cuál de estos pasos puede omitirse. Así entonces, los dispositivos pueden clasificarse según las siguientes categorías (Figura 2.14):

Categoría 1: El dispositivo actual no tiene integrado J2ME ni algún soporte para bluetooth. (Los 4 pasos anteriores son necesarios)

Categoría 2: El dispositivo actual tiene J2ME pero ningún soporte bluetooth nativo (seguir los pasos 2-4)

Categoría 3: El dispositivo cuenta con soporte bluetooth pero no tiene habilitado el J2ME (se requiere de los pasos 1,3 y 4)

Categoría 4: El dispositivo cuenta con J2ME y soporte bluetooth pero no con JABWT (los pasos 3 y 4 son necesarios)

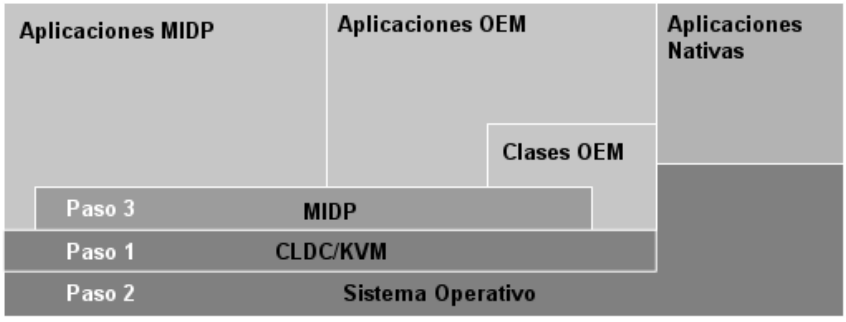


Fig. 2.14 Componentes del dispositivo JABWT

Los pasos y categorías anteriores asumen una máquina KVM con una interfaz para la pila bluetooth nativa no dependiente de java y la inexistencia de una interfaz nativa (JNI-Java Native Interface) para la implementación J2ME.

JABWT depende únicamente de las librerías CLDC pero complementa su funcionalidad con los perfiles J2ME p. ej. MIDP.

2.7.2 Añadir el soporte J2ME y soporte bluetooth

La posibilidad de implementar java en un dispositivo bluetooth requiere de la existencia de una máquina virtual apropiada, un set de librerías y una pila de protocolos bluetooth nativa.

La pila subyacente para el protocolo bluetooth debe estar certificada en al menos GAP, SDAP, y SPP. Si además cuenta con las herramientas a continuación listadas, la interfaz entre esta y la implementación JABWT se simplifica:

- Aplicaciones con acceso a la pila a través de API's
- Soporte para llamadas asíncronas
- Notificación de eventos asíncronos ya sea a través de funciones call-back o monitoreos a nivel de aplicación

Además de contar con estas herramientas, la pila debe asegurar acceso externo a la pila y ejecución directa de funciones L2CAP a través de:

- API´s para realizar conexiones RFCOMM, consultas y servicios para funciones de descubrimiento.
- API's para añadir y eliminar registros de servicios
- Acceso a las API's para realizar conexiones L2CAP

En cuanto a seguridad y soporte refiere, la pila debe:

- Determinar la confiabilidad de un dispositivo
- Autorizar una conexión dada y dispositivo remoto
- Establecer el NIP
- Habilitar/Deshabilitar la encriptación
- Autenticar el dispositivo remoto
- Identificar conexiones anteriormente autenticadas/autorizadas
- Determinar la encriptación de un enlace dado
- Obtener el nombre de usuario del dispositivo local y remoto
- Obtener información de la clase del dispositivo local
- Cambiar el modo de descubrimiento del dispositivo local
- Habilitar/deshabilitar el modo de conexión de los dispositivos locales.
- Obtener la dirección bluetooth del dispositivo local

2.7.3 Implementación de JABWT

Las librerías javax.bluetooth, opcionalmente la javax.obex y las extensiones KVM conforman la implementación JABWT (ver figura 2.15).

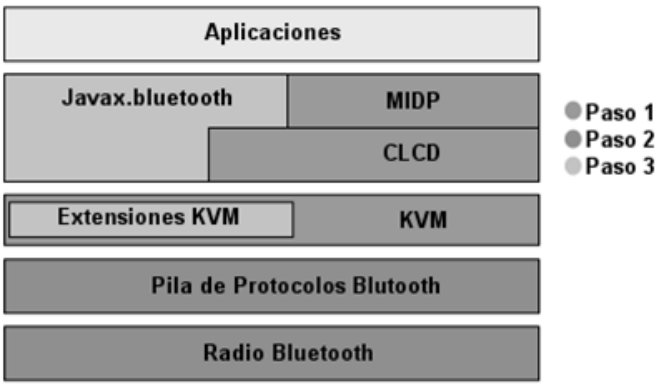


Fig. 2.15 Componentes para la implementación

La KVM y la interfaz del sistema operativo KVM (KOSI) integran la parte nativa del CLDC mientras que las extensiones KVM forman parte de la implementación JABWT (i.e. KVM del Bluetooth, KOSI del Bluetooth y BCC).

La importancia de la KVM radica en las herramientas que sus extensiones proporcionan:

- Funciones nativas que realizan llamados a la pila
- Código para el manejo de eventos durante call-back de la pila
- Resolución de conflictos entre aplicaciones y seguridad a través del BCC
- Código para interactuar con la pila y manipular estructuras de datos

KVM del bluetooth

El módulo trabaja con una KVM modificada para el manejo de eventos bluetooth y con funciones nativas adicionales; es independiente de la pila pero dependiente del KVM por lo que su arquitectura y diseño determinarán su desempeño.

Bluetooth-KOSI

Esta capa funciona como interfaz entre la KVM y la pila de protocolo Bluetooth, su implementación requiere una comprensión detallada del funcionamiento de la pila y su API, debido a su dependencia.

Interfaz KVM [20]

Algunos aspectos importantes de la implementación de los módulos KVM y KOSI del Bluetooth son:

- El manejo de conexiones. Los módulos permiten conservar la asociación entre los objetos Java de las aplicaciones J2ME y la conexión bluetooth actual en la pila de protocolos, aún si existen conexiones múltiples o simultáneas.
- Manejo de eventos. La interfaz nativa debe ser capaz de notificar los eventos de la pila a la KVM. Las notificaciones comúnmente se sirven de mecanismos de monitoreo realizados por la KVM o funciones call-back registradas en la pila.
- Bloqueo y desbloqueo de llamadas a la pila. La naturaleza de las comunicaciones bluetooth hace necesario limitar los bloqueos de llamadas a la pila cuando las implementaciones se llevan un tiempo considerable en completarse.

Algunos de los posibles escenarios donde el bloqueo y antibloqueo de llamadas cobra importancia son:

- La aplicación JABWT realiza una llamada de bloqueo al API y la KVM a la pila. En este caso tanto la aplicación como la KVM son bloqueadas hasta que la operación se complete. Este tipo de operaciones deben limitarse a llamadas rápidas de la pila.
- La aplicación bluetooth hace una llamada de bloqueo al API utilizando métodos controlados por eventos. Si la pila no provee mecanismos para el control de eventos, no debe considerársele para su implementación en dispositivos JABWT.
- La aplicación JABWT realiza llamadas de bloqueo en operaciones en las que no es recomendable bloquear la KVM (p. ej. métodos de acceso a dispositivos remotos). Bajo estas circunstancias la KVM debe implementar llamadas antibloqueo a la pila.

Implementación del BCC

El BCC es un componente más de las extensiones KVM, su implementación es completamente dependiente y requiere de un cierto nivel de portabilidad para cada dispositivo, incluso cuando la KVM y la pila son las mismas.

El BCC es el encargado de resolver problemas relacionados con peticiones conflictivas a la pila, así como establecer y reforzar las configuraciones de seguridad.

Implementación OBEX

El API OBEX en JABWT es un API independiente y opcional del API bluetooth. Su portabilidad facilita su implementación en dispositivos que no cuenten con protocolos de transporte bluetooth como IrDA o USB.

Las implementaciones OBEX pueden ser nativas o creadas con el lenguaje JAVA. Trabajar con las implementaciones nativas facilita la programación pero dificulta su portabilidad.

2.7.4 Normativa TCK

A través de un grupo de pruebas, herramientas y documentación, el TCK verifica que las implementaciones cumplan con las especificaciones del API de JAVA. El procedimiento de prueba es de caja negra con mínimo estrés y funcionalidad y sólo las implementaciones JABWT, no las programadas con JABWT, son analizadas. En caso de que una aplicación cumpla con las normativas, podrá ser ejecutada en cualquier dispositivo compatible con JABWT.

La especificación JSR-82 cuenta con dos TCK's para determinar el cumplimiento de las especificaciones JABWT en los dispositivos:

- TCK bluetooth. Examina el cumplimiento del API bluetooth
- TCK OBEX. Examina el cumplimiento del API OBEX

Estas pruebas pueden incluirse en el TCK del CLDC y la JavaTest™ que consisten en fuentes test-case, clases test-case y documentación. Las compañías que dirigen un JSR deben crear una implementación referencia (RI) y TCK para la especificación desarrolladas. En caso de que hagan uso de las API's JABWT deben utilizar el TCK del JSR-82 para demostrar que sus implementaciones cumplen con las especificaciones.

Instalación y configuración del TCK

El TCK de JABWT utiliza el mismo diseño cliente-servidor que el TCK CLDC, más el framework JavaTest; el servidor TCK ejecuta la aplicación JavaTest y un dispositivo cliente TCK ejecuta la aplicación CLDCAgent.

De forma más detallada, la implementación JABWT a probar en el dispositivo cliente TCK interactúa con un dispositivo bluetooth que ejecute una aplicación diseñada específicamente para la prueba. El examen recibe el nombre de agente TCK y es necesario para que el TCK del JSR-82 interactúe con el CLDCAgent.

El agente TCK en el dispositivo remoto puede presentarse como:

- Una aplicación CLDC (o MIDP)+JABWT ejecutándose en el dispositivo JABWT
- Una aplicación nativa (no JABWT) que use una implementación nativa bluetooth u obex

Capítulo 3: Programación[11][9][12]

3.1 Lenguajes de programación

Los Servicios de Telecomunicación IN son programados principalmente con lenguajes procedurales. Su modelo de servicio se basa en el uso de scripts, quienes especifican la lógica para formar un nuevo servicio desde una construcción elemental de bloques. Este modelo, sin embargo, presenta deficiencias en comparación con las técnicas orientadas a objetos, así como inconsistencias en la construcción de bloques.

Por otra parte, la modularidad que ofrecen las técnicas de programación OO facilita la creación y mantenimiento de las aplicaciones; ya que, por ejemplo, permiten el reciclado de código y/o datos a través de objetos; proporcionan funcionalidades específicas, interfaces de control, fácil acceso a servicios y la cómoda administración de las aplicaciones.

Entre los lenguajes de programación aceptados para el manejo de la tecnología Bluetooth se encuentran: Visual Basic, Python, Java, Open C, Flash Lite, Ruby, Lua, C++. La presente tesis ocupó como herramienta de desarrollo el lenguaje de programación Java.

3.1.1 J2ME

Java nació como un lenguaje destinado a la programación de dispositivos domésticos mas las características con las que fue implementado lo orientaron hacia un lenguaje de desarrollo de mayor escala. Fue así que Sun generalizó su aplicación diversificando su uso para dar lugar a tres diferentes aplicaciones (Fig. 3.1):

- Las de propósito general a través de J2SE
- Las orientadas a empresas por medio de J2EE
- Las destinadas a ejecutarse sobre sistemas con prestaciones limitadas gracias a J2ME

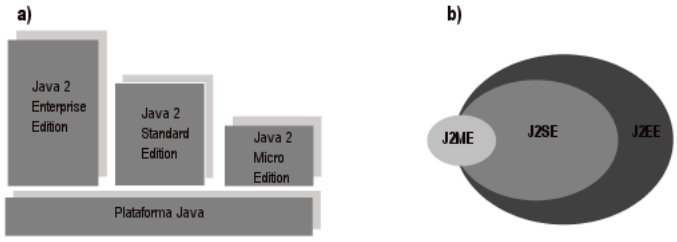


Fig 3.1 a) Arquitectura Sun para Java; b) Relación de API's

Estas derivaciones poseen un diseño jerarquizado permitiendo la existencia de semejanzas entre las API's de cada paquete (fig. 3.3 y 3.4).

Por su parte, la derivación J2ME se diferencia de las demás al hacer uso de la KVM (Máquina Virtual basada en Kernel), de la configuración CLDC y del perfil MIDP.

El diseño de la plataforma proporciona un soporte flexible y eficiente a las necesidades de los servicios en todos los canales móviles, también permite migrar fácilmente entre diferentes configuraciones y perfiles, siendo posible entregar un mismo servicio a través de diferentes canales (fig. 3.2).

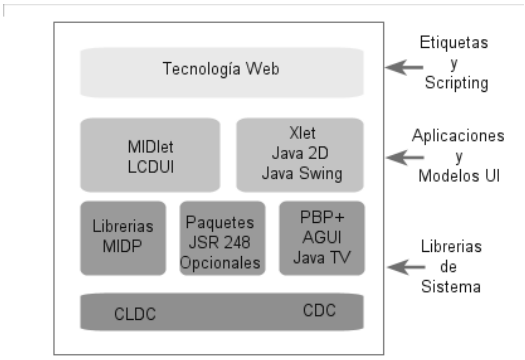


Fig. 3.2 Plataforma de Servicios Convergentes

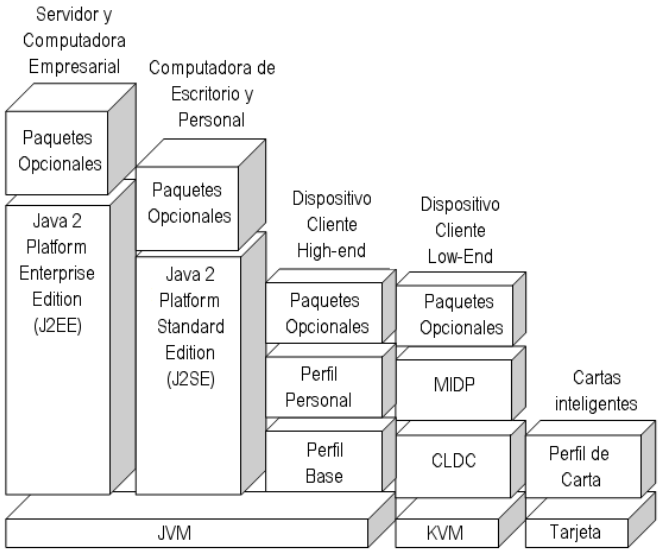


Fig. 3.3 Plataformas de Java 2

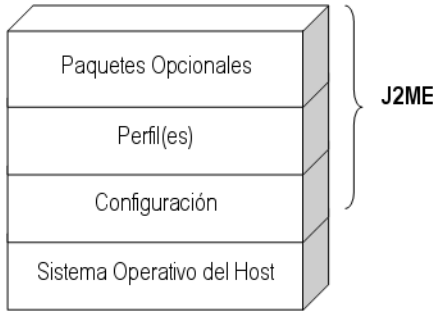


Fig. 3.4 Plataformas J2ME

3.1.2 Configuración CLDC

Se entiende como configuración el conjunto mínimo de aplicaciones cuyas API's describen características básicas comunes a todos los dispositivos dentro de un grupo. En J2ME la única configuración definida es CLDC del inglés Configuración de Dispositivo de Conexión Limitada.

Esta configuración consta básicamente de:

- Subconjunto de elementos básicos del lenguaje Java
- Una parte de la funcionalidad de la JVM
- Las API's básicas para el desarrollo de aplicaciones
- Requisitos hardware de los dispositivos englobados en el CLDC

La diferencias entre otras configuraciones y esta radica en la ausencia de soporte a operaciones con coma flotante, ausencia del método Object.finalize(), cambio en el manejo de excepciones y la seguridad. Esta última marca una serie de funcionalidades sensibles o críticas fuera del alcance de los programas, así como condiciones previas obligatorias para cualquier aplicación tales como:

- Los ficheros contenedores de las clases debieron haber sido verificados como una aplicación válida Java en un paso adicional dentro del desarrollo.
- Sólo se puede hacer uso del API CLDC predefinido.
- No se permite el uso de cargadores de clases definidas por el usuario.

En cuanto a los requisitos de hardware:

- 160 Kb de memoria disponible para Java (128 Kb de memoria no volátil y 32 Kb de memoria volátil.
- Un procesador de 16 bits
- Bajo consumo
- Conexión a red con un ancho de banda de 9600 bps

3.1.3 Perfiles MIDP

El perfil añade API's que definen las características de un dispositivo mientras que las configuraciones hacen lo propio con una familia de ellos. Por ello, los perfiles se apoyan en las configuraciones.

MIDP es el primer perfil hecho para describir dispositivos similares a teléfonos móviles o radio-localizadores cuyas características mínimas son:

- Al menos 128 Kbyte de memoria no volátil para almacenar el API MIDP
- Al menos 32 Kbyte de memoria volátil para el sistema de ejecución Java
- Mínimo 8 Kbyte de memoria persistente para el almacenamiento de información por parte de los programas.

En cuanto a hardware, un dispositivo MIDP debe contar con teclado, pantalla táctil o ratón como medio de entrada de datos; para visualización, una pantalla de mínimo 96 píxeles de ancho por 54 de alto con dos colores, una relación anchura: altura de 1:1, acceso a una red bidireccional con un ancho de banda de 9600 bps.

Los elementos en la gestión de plataforma requeridos son:

- La presencia de un Kernel
- Mecanismo para leer y escribir en memoria no volátil
- Gestión del tiempo para fijar temporizadores y añadir marcas temporales a la información persistente.
- Acceso de lectura escritura a la conexión inalámbrica del dispositivo.
- Medios para obtener la información introducida por el usuario a través de los dispositivos de entrada.
- Soporte a gráficos basados en mapa de bits.
- Medios para gestionar el ciclo de vida de una aplicación.

3.1.4 MIDLET

Es una aplicación J2ME desarrollada sobre el perfil MIDP que conserva la propiedad de portabilidad gracias a los siguientes requisitos definidos por el perfil:

- Ofrecer un gestor de aplicaciones encargado de la carga, ejecución y desinstalación de los MIDlet
- Proporcionar una misma interfaz a los gestores de aplicaciones para validar mecanismos de herencia

El ciclo de vida de un MIDlet se define básicamente a través de tres estados y cuatro métodos tal como lo muestra la figura 3.5.

Se considera a un MIDlet en ***Estado de Espera*** cuando es interrumpido por el gestor de aplicaciones, así como al iniciar la aplicación; hablamos de un ***Estado Activado*** cuando se está ejecutando la aplicación en sí y nos referimos a un ***Estado Destruido*** sólo si la aplicación termina su ejecución.

En cuanto a los métodos refiere, el método ***new*** invocará la aplicación y esta será cargada en el dispositivo gracias al método ***startApp. pauseApp*,** por su parte, entrará en acción cuando se ejecute una interrupción por parte del gestor de aplicaciones; su función es almacenar la información importante hasta ese momento generada. Al finalizar la ejecución del MIDlet se utilizará el método ***destroyApp*** para liberar los recursos reservados temporalmente y almacenar información para usos posteriores.

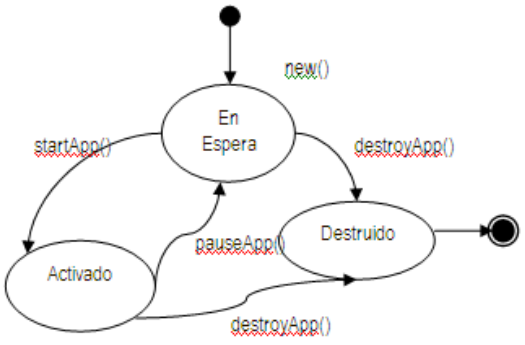


Fig. 3.5 Ciclo de Vida de un MIDlet

Aunque estos no son los únicos métodos disponibles para el manejo de un MIDlet, son los más importantes.

En la figura 3.6 se muestra el proceso de creación y depuración de un MIDlet. El primer paso consiste en la generación de código que al ser compilado generará las clases binarias. Para llevar a cabo la compilación se requiere del ambiente de desarrollo J2SDK y JWTK. El paso siguiente a la compilación es la verificación que tiene como objetivo realizar revisiones de seguridad, integridad y cumplimiento de estándares; terminada la revisión se realiza el empaquetado, donde las clases, interfaces y recursos obtenidos de las etapas anteriores son encapsulados dentro de un archivo distribuible con extensión .JAR. A la par se genera un archivo JAD cuya función es describir las propiedades del archivo JAR. Una vez comprimido el MIDlet podrá probarse en un emulador y en caso de prescindir de alguna modificación, podrá ser instalado en un dispositivo móvil.

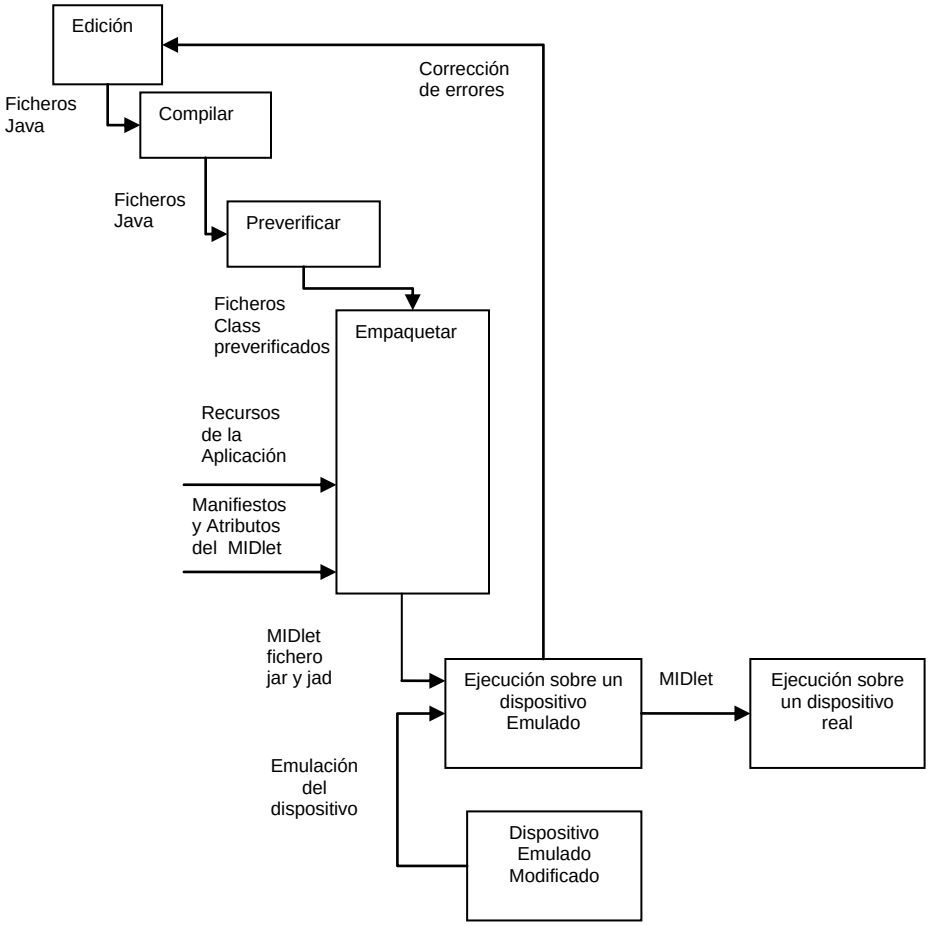


Fig. 3.6 Proceso de Generación y Depuración de aplicaciones J2ME

3.2 API de JAVA para Bluetooth

La necesidad de comunicación a bajo costo sobre dispositivos móviles, así como el avance en la tecnología Bluetooth hizo posible el desarrollo de librerías Java que estandarizaran la forma de desarrollar aplicaciones de este tipo.

Los API's estándar de Java son definidos por el JCP quien coordina la evolución del lenguaje. Cada nuevo API es desarrollado como un JSR bajo un proceso de votación en el que se analiza y define el estándar.

La estandarización de JABWT bajo JSR-82 quedó a cargo de un grupo de 18 compañías y 3 individuos. El API de alto nivel fue definido como un paquete opcional para dispositivos J2ME basados en CLDC.

Los JSR's que definen la configuración actual J2ME y perfiles se encuentran en la tabla 3.1:

Número	Alcance
JSR 30	Configuración J2ME del CLDC (Connected Limited Device Configuration)
JSR 37	MIDP (Mobile Information Device Profile for the J2ME Platform)
JSR 75	Perfil PDA para la plataforma J2ME
JSR 36	Configuración J2ME del CDC (Connected Device Configuration)
JSR 46	Perfil básico J2ME
JSR 129	Especificación PBP (Personal Basis Profile)
JSR 62	Personal PPS (¨Personal Profile Specification)
JSR 66	Perfil J2ME RMI
JSR 134	Perfil Java Game
JSR 209	Gráficos avanzados y paquete opcional de interfaz de usuario para la plataforma J2METM
JSR 218	Revisión de la especificación Java ME CDC

Tabla 3.1 JSR's del J2ME

Los JSR's no relacionados directamente con alguna configuración se muestran en la tabla 3.2:

Número	Alcance
JSR 82	API's de java para Bluetooth
JSR 120	API's WTCA (Wireless Telephony Communication)
JSR 135	API J2ME para Multimedia

Tabla 3.2 JSR's no relacionados

La tabla 3.3 muestra los JSR's en proceso:

Número	Alcance
JSR 68	Especificación de la plataforma J2ME
JSR 118	Siguiente generación del MIDP
JSR 139	Siguiente configuración del CLDC

Tabla 3.3 JSR's en proceso de desarrollo

El API del JSR 82 en específico, se divide en dos paquetes: javax.bluetooth y javax.obex. El primero permite detectar servicios, dispositivos y comunicación mediante flujos (streams) o arreglos de bytes; mientras que el segundo provee comunicación gracias al protocolo OBEX.

La anatomía de una aplicación Bluetooth está dividida en cuatro partes: inicialización de la pila; descubrimiento de servicios y dispositivos; manejo del dispositivo y comunicación.

En cuanto a hardware refiere, los APIS de java para bluetooth funcionan bajo dispositivos con las siguientes características:

- Mínimo 512 Kb de memoria libre
- Conexión Bluetooth
- Permitir la implementación de J2ME CLDC

Capítulo 4: Cartelera cultural web y bluetooth

4.1 Diseño[13][14][15]

El sistema tiene como finalidad la difusión de eventos culturales dentro de la facultad de ingeniería a través de la web y bluetooth.

Su diseño y estructura se basan en la metodología del proceso unificado; esto es, en casos de uso como elemento principal.

A continuación se presentan los diferentes diagramas que componen el sistema.

4.1.1 Casos de uso

Los casos de uso se definieron tanto para la parte web como para la móvil y sus principales actores son: organizador, administrador y usuario.

Se considera como organizador a toda asociación estudiantil dada de alta que, con el respectivo permiso de la secretaría académica, necesite publicar sus eventos culturales. La gestión de los organizadores (altas y bajas) estará a cargo de la secretaría académica a quien se le ha denominado como administrador. Por su parte, los usuarios son todas aquellas personas que tengan acceso a internet y, para el módulo bluetooth, que hayan descargado el MIDlet. La figura 4.1 resume los casos de uso posibles para los actores anteriormente establecidos.

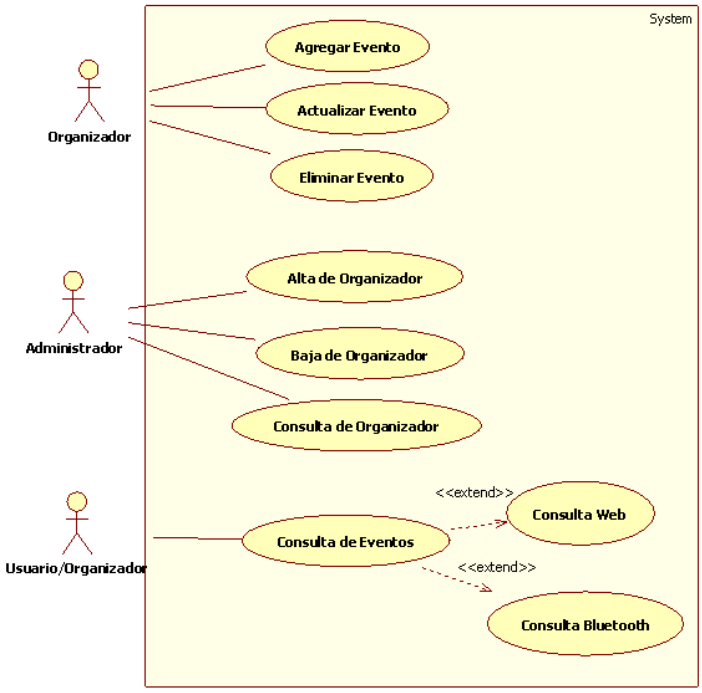
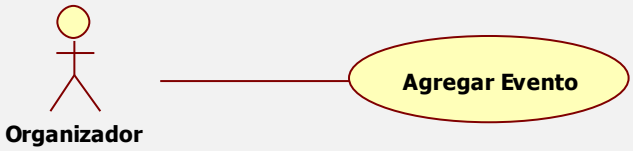


Fig. 4.1 Casos de Uso

Los dos casos de uso más importantes se detallan en las tabla 4.1 a) y b), la descripción de los casos restantes se encuentran en el anexo A.

Caso de uso:	Agregar evento
Actor:	Organizador



Descripción	El organizador entra al sistema para dar de alta un nuevo evento.
Precondiciones	1. El operador deberá haber accedido al sistema a través de un usuario y contraseña. 2. Los eventos que dé de alta deben estar previamente autorizados por la Secretaría Académica.

Flujo Normal del Evento

Actor		Sistema		Excepción
Paso	Acción	Paso	Acción	
		1	El sistema envía el formulario	
2	El organizador captura los datos del evento	3	El sistema valida la información proporcionada por el organizador	E1
		4	El sistema almacena la información en la BD y redirecciona a la página principal	E2

Flujo Excepcional de Eventos

Excepción	Descripción	Acción
E1	La información proporcionada no es correcta	El sistema notifica al usuario sobre el error
E2	La BD presenta algún error	El sistema notifica al usuario sobre el error

Tabla 4.1 a) Caso de uso: Agregar evento

Caso de uso:	6. Consulta de eventos
Actor:	Usuario /Organizador

Descripción	El organizador/usuario accede al sistema ya sea a través de la web o de la aplicación bluetooth cliente para realizar consultas de eventos.
Precondiciones	1. El organizador/usuario debe acceder al sistema por medio de un navegador de páginas Web o de la aplicación bluetooth cliente

Flujo Normal del Evento

Actor		Sistema		Excepción
Paso	Acción	Paso	Acción	
1	El usuario/organizador establece los parámetros de búsqueda	2	El sistema realiza la conexión con la base de datos y ejecuta la consulta	E1,E2
		3	El sistema devuelve el resultado de la consulta, ya sea a través de la página web o de la aplicación móvil	E3

Flujo Excepcional de Eventos

Excepción	Descripción	Acción
E1	La conexión con el sistema no puede ser procesada	El sistema notifica sobre la falla al usuario permitiéndole realizar otra operación
E2	La conexión con la base de datos no puede ser procesada	El sistema notifica sobre la falla al organizador/usuario permitiéndole realizar otra operación
E3	La consulta no puede ser enviada al móvil	El sistema notifica sobre la falla al usuario móvil permitiéndole realizar otra operación

Tabla 4.1 b) Caso de uso: Consulta de eventos

4.1.2 Diagrama de clases

A continuación se presenta el diagrama de clases (Figura 4.2). Su diseño considera la existencia previa de una base de datos de las organizaciones estudiantiles actualmente activas por lo que su registro sólo les concederá el acceso a la sección de publicaciones de eventos.

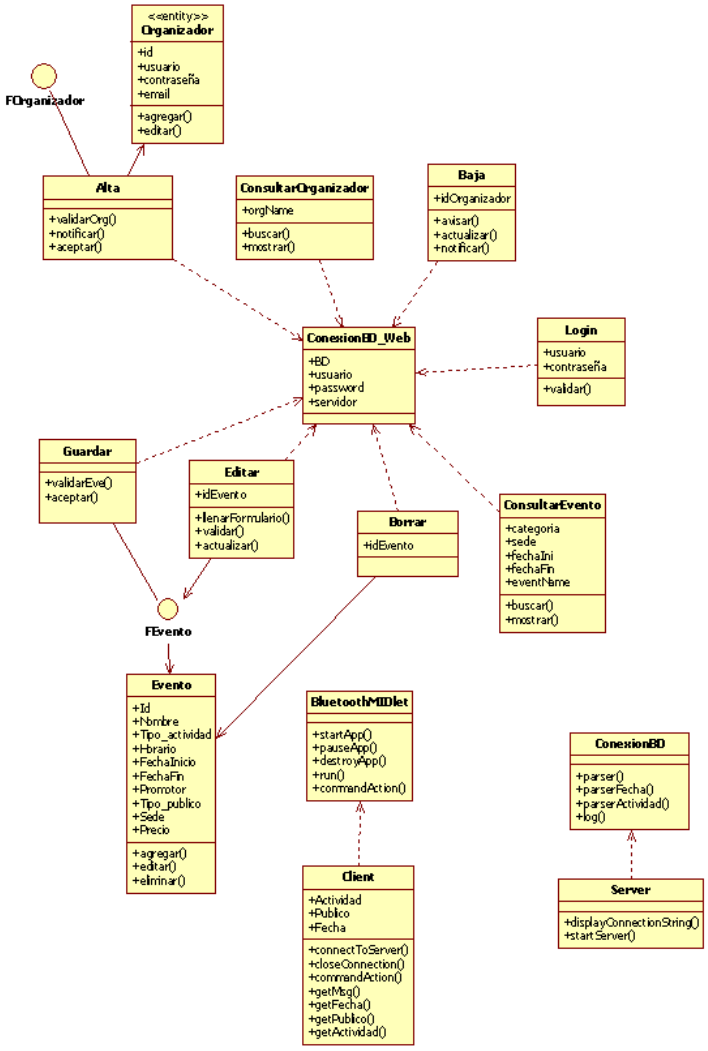


Fig. 4.2 Diagrama de Clases

4.1.3 Diagrama de implementación y ejecución

La distribución inalámbrica de contenidos publicitarios o informativos requiere de dispositivos receptores equipados con tecnología bluetooth; y si aparte de ello se desea personalizar una búsqueda se debe añadir a los requerimientos una aplicación que sirva como interfaz al usuario final. La configuración que cubre estos requerimientos de intercomunicación es J2ME-J2SE-BD-Servidor Web como se muestra en el diagrama de la figura 4.3.

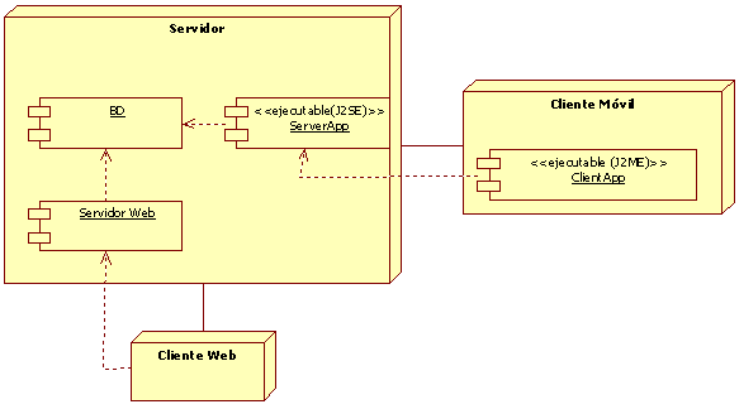


Fig. 4.3 Diagrama de implementación y ejecución

4.1.4 Diagrama de actividades y secuencia

Los diagramas de actividades y secuencia que a continuación se muestran representan los casos de uso más significativos, esto es: caso de uso agregar evento, consulta de evento web y consulta de evento bluetooth. Los demás diagramas pueden ser consultados en el anexo B.

Los diagramas mostrados en la figura 4.4 corresponden al caso de uso "Agregar Evento", mientras que las figuras 4.5 y 4.6 representan aquellos derivados del caso de uso "Consulta web" y "Consulta Bluetooth" respectivamente.

Caso de uso: Agregar Evento

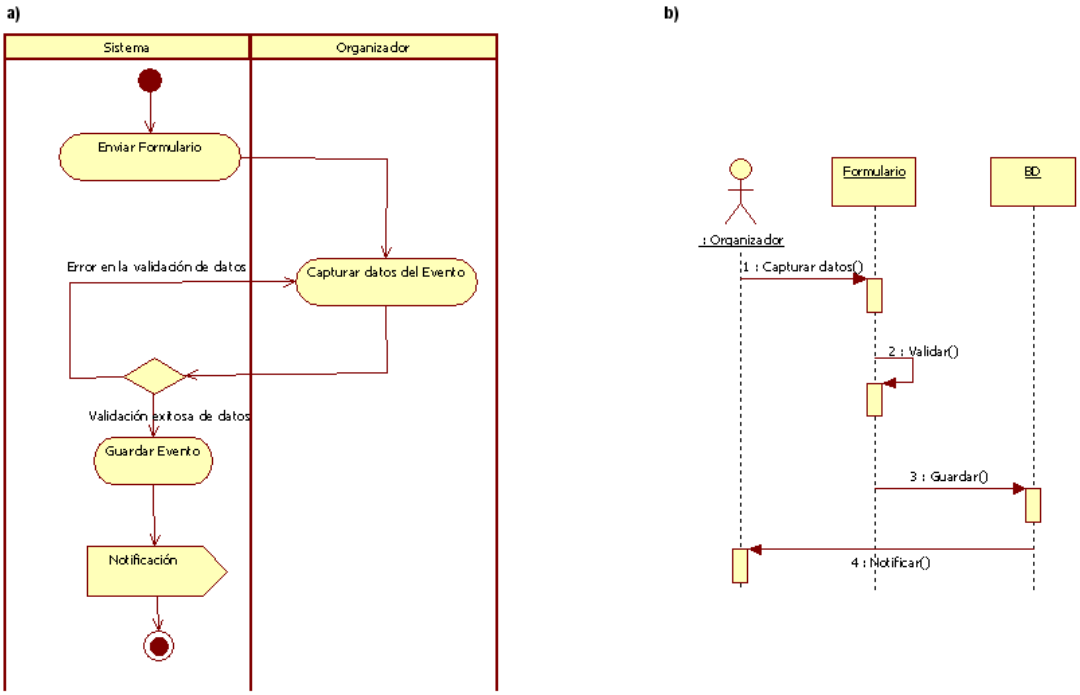


Fig. 4.4 Diagramas para el caso de uso "Agregar Evento": a) Diagrama de Actividades b) Diagrama de secuencia

Caso de uso: Consulta web

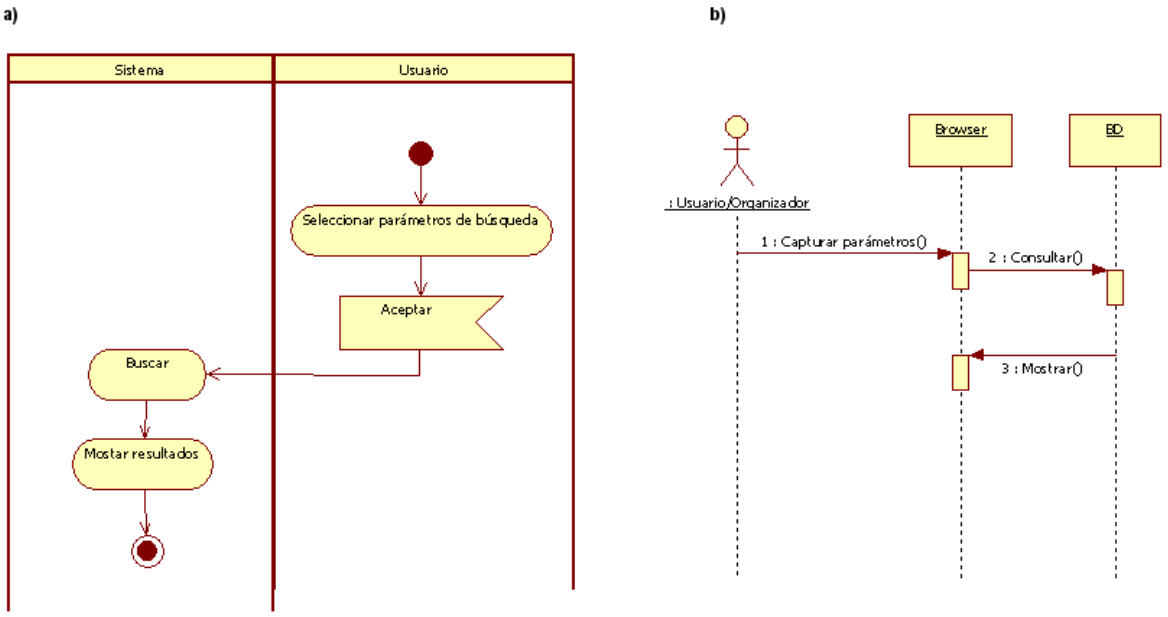


Fig. 4.5 Diagramas para el caso de uso "Consulta Web": a) Diagrama de Actividades b) Diagrama de secuencia

Caso de uso: Consulta Bluetooth

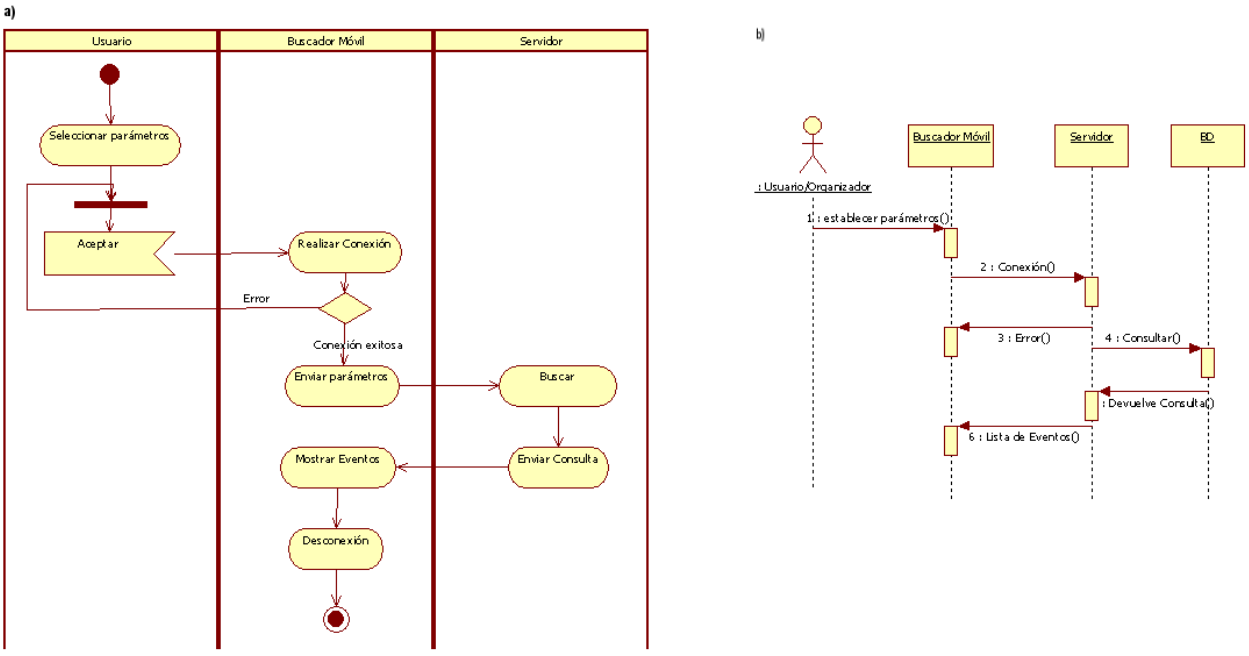


Fig. 4.6 Diagramas para el caso de uso "Consulta Bluetooth": a) Diagrama de actividad b) Diagrama de secuencia

4.2 Desarrollo

4.2.1 Requerimientos de ejecución

Para la correcta ejecución de la aplicación en el servidor se requiere de los siguientes elementos:

- Dispositivo bluetooth (tarjeta o dongle)
- Controlador para Pila bluetooth (Microsoft, Widcomm, Toshiba, Bluesoleil, BlueZ)
- Librería Bluecove
- JRE 6 o posterior

Para el funcionamiento del módulo web es necesario contar con:

- Servidor Apache
- Acceso a un manejador de base de datos MySQL 5.0 o posterior
- La plataforma de trabajo en la que se ejecute el MIDlet cliente precisa de la siguiente configuración:
- MIDP 2.0
- CLDC 1.1
- JSR-82 (Bluetooth)

4.2.2 Base de Datos

El presente proyecto contempla un modelo cliente-servidor en donde el cliente, a través del MIDlet instalado en el dispositivo móvil y de la aplicación en el servidor, realiza consultas a la base de datos.

El diagrama E-R de la base de datos se presenta en la figura 4.7

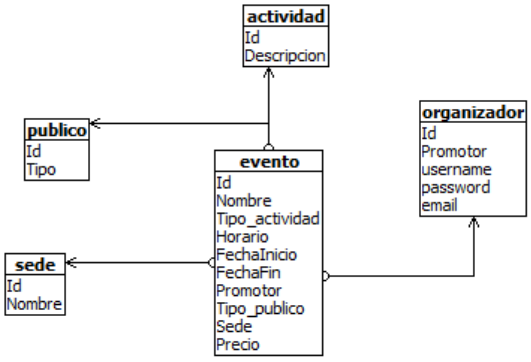


Fig. 4.7 Diagrama E-R de la base de datos "Cartelera C"

4.2.3 Web

Para ingresar registros a la base de datos se creó una interfaz web que además de permitir la inserción de datos también proporciona información a los usuarios no registrados sobre las actividades culturales en curso. En la figura 4.8 se observa la página de inicio accesible al público en general.

Fig. 4.8 Página de Inicio

El acceso a la sección de publicación de eventos requiere de un registro previo de las organizaciones a través de la entidad correspondiente, quien les proporcionará un nombre de usuario y contraseña necesarios para firmarse en el sistema. La figura 4.9 muestra la página en donde se realizará la validación de usuarios.

Fig. 4.9 Validación de usuarios

Tras la validación el organizador se redireccionará a la página de inicio del usuario. Esta difiere de la página de inicio general en cuanto a las herramientas ofrecidas y la información de eventos desplegada. Esto es, el despliegue de información toma como parámetro inicial de filtro el valor del nombre del organizador, eliminando la primera opción del menú de búsqueda y enlistando sólo aquellos registros de los que el usuario firmado sea dueño. Cada registro podrá ser eliminado o editado independientemente. La figura 4.10 muestra el ejemplo de una página de inicio con una sesión activa.

Fig. 4.10 Página de Inicio para los usuarios registrados

El usuario también podrá dar de alta eventos. El formulario para el alta de eventos se muestra en la figura 4.11. Cabe destacar que en los campos de categoría y sede se tiene la opción de agregar un nuevo valor en la base de datos que identifique de manera más apropiada el evento que se llevará a cabo en caso de no encontrarse en ese momento.

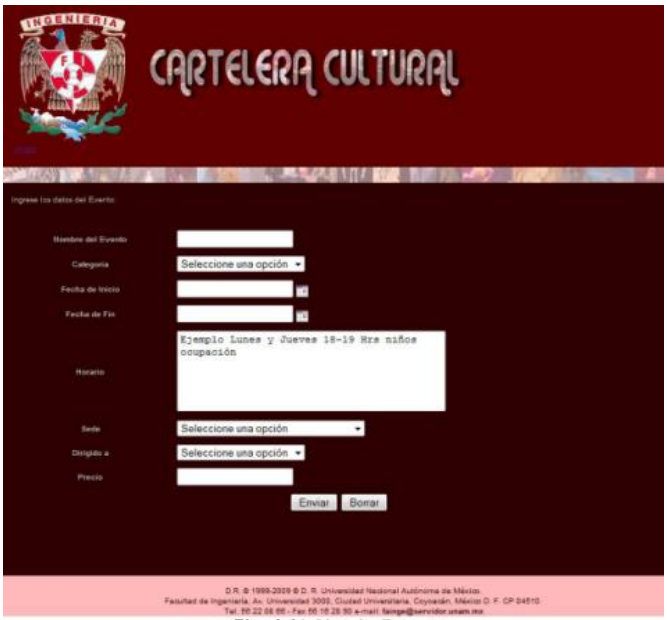


Fig. 4.11 Alta de Eventos

La edición de eventos retoma el formulario de alta y, tras consultar la base de datos, carga los valores del registro en los correspondientes campos.

4.2.4 Aplicación en el servidor

Debido a las limitantes que presenta la programación en móviles y J2ME, la conexión a la base de datos no puede hacerse directamente por lo que se desarrolló una aplicación J2SE que funge como enlace entre el dispositivo bluetooth y la base de datos.

El programa en el servidor, una vez inicializada la pila bluetooth, se mantendrá a la escucha de solicitudes de conexión por parte del cliente bluetooth. Cada vez que se acepte una conexión, se desplegará la MAC Bluetooth y el nombre del dispositivo cliente (Ver tabla 4.2).

```
public BTServer1(boolean continua, StreamConnectionNotifier notifier) {
public void begin(){
      try {
         ..,
         notificador = (StreamConnectionNotifier) Connector.open("btspp://localhost:1101;name=Servidor Bluetooth");
         inicia = new BTServer1(notificador);
         iniServidor = new Thread(inicia);
         iniServidor.start();
      } catch (IOException ex) {
         textArea.append("\nError al iniciar el servidor");
      }
   }
   …
   public void run() {
      ...
      try{
         cadenaDeConexion(streamConnNotifier);
         while(continua){

            textArea.append("\nEl servidor ha iniciado. Esperando solicitud de conexión del cliente...");
            connection = streamConnNotifier.acceptAndOpen();
            textArea.append("\nAbriendo conexión");
            RemoteDevice dev = RemoteDevice.getRemoteDevice(connection);
            textArea.append("\nDirección del cliente: "+dev.getBluetoothAddress());
            textArea.append("\nNombre del cliente: "+dev.getFriendlyName(true));

            …
         }
        }       …
      }
   }
}
```

Tabla 4.2 *Código de conexión del servidor*

Establecida la conexión en RFCOMM entre ambos dispositivos, el MIDlet en el móvil enviará los parámetros de búsqueda al servidor. La aplicación en el servidor recibirá la secuencia de bits que, a través de un parser, transformará en una instrucción de búsqueda válida para la consulta en la base de datos (Tabla 4.3).

```
public BTServer1(boolean continua, StreamConnectionNotifier notifier) {  …
   public void run() {  ...
      try{ …
         while(continua){
           //Recepción y lectura de la cadena enviada por el cliente
            inStream = connection.openInputStream();
            bReader = new BufferedReader(new InputStreamReader(inStream));
            String lineRead = bReader.readLine();
            textArea.append("\n Cadena leída");
            ConexionBD bd = new ConexionBD();
            answer = bd.consulta(lineRead);
            if(answer.isEmpty()){
               textArea.append("\nNingún evento con esas características");
               answer = "\nNingún evento con esas características";}
            else
               textArea.append("\n"+answer+"\n");
           … }   … }
   }
}
```

Tabla 4.3 *Consulta de la base de datos y envío de información*

Una vez recibido el set de resultados generado por la base de datos, la aplicación en el servidor enviará la cadena al móvil y cerrará parcialmente la conexión. En otras palabras, el buffer de entrada y salida de datos se cerrará pero se mantendrá abierto el canal de conexión a la espera de una nueva petición de conexión (Tabla 4.4).

```
public BTServer1(boolean continua, StreamConnectionNotifier notifier) {   …
   public void run() {  ...
      try{
         cadenaDeConexion(streamConnNotifier);
         while(continua){
            textArea.append("\nEl servidor ha iniciado. Esperando solicitud de conexión del cliente...");
            connection = streamConnNotifier.acceptAndOpen();
            textArea.append("\nAbriendo conexión");
            …
            //Envío de respuesta al cliente
            outStream = connection.openOutputStream();
            pWriter = new PrintWriter(new OutputStreamWriter(outStream));
            pWriter.write(answer);
            textArea.append("\nEnvío Exitoso");
         //Cierre de buffer de entrada/salida
```

```
pWriter.close();
bReader.close();
outStream.close();
inStream.close();
connection.close();
...
```

Tabla 4.4 *Envío de información y cierre de la conexión*

La figura 4.12 muestra la aplicación del servidor en funcionamiento. Como se observa, se tiene la opción de guardar el log en caso de requerirlo, al igual que de cerrar el socket de conexión o reiniciar la aplicación.

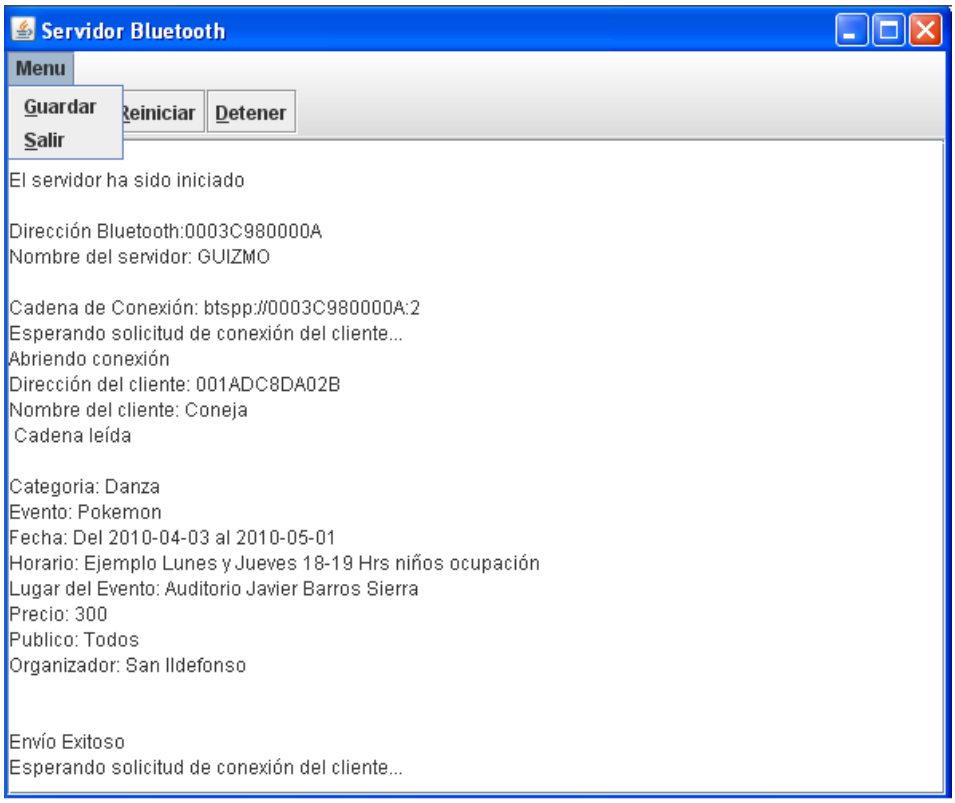


Fig. 4.12 Aplicación en el servidor

4.2.5 Aplicación en el móvil

El MIDlet en el móvil permite al usuario personalizar la consulta de eventos a través de un mini buscador bluetooth que se conecta con el servidor bluetooth.

En la figura 4.13 se ilustran las opciones de búsqueda presentes en el MIDlet. Una vez seleccionados los parámetros, se establecerá la conexión con el servidor y enviarán los valores establecidos en el formulario del MIDlet a través de la opción conectar del menú. El código que genera esta interfaz se muestra en la tabla 4.5.

Fig. 4.13 Aplicación en el cliente

```
public class Client extends BluetoothMIDlet {
   …
   public void startApp() throws MIDletStateChangeException {

      connForm = new Form("Cartelera Cultural");
      connectCommand = new Command("Connect", Command.OK, 1);
      connForm.addCommand(connectCommand);
      connForm.addCommand(new Command("Exit", Command.EXIT, 1));
      connForm.setCommandListener(this);
      connForm.append(getActividad());
      connForm.append(getPublico());
      connForm.append(getFecha());
      Display.getDisplay(this).setCurrent(connForm);
   }
…
```

Tabla 4.5 *Código que genera la interfaz en el móvil.*

De la clase BluetoothMIDlet se heredan los métodos que definen los estados de una aplicación móvil (activo, pausa, destruir). Su contenido puede verse en la tabla 4.6.

```
public class BluetoothMIDlet extends MIDlet implements Runnable, CommandListener {
   public BluetoothMIDlet() {}

   public void startApp()throws MIDletStateChangeException {
      new Thread(this).start();
   }
   public void pauseApp() { … }
```

```
    public void destroyApp(boolean unconditional) {…}

    public void run() {…}

    public void commandAction(Command c, Displayable d) {
        notifyDestroyed();
    }
}
```

Tabla 4.6 *Métodos que definen el comportamiento de cualquier MIDlet*

Finalmente, en la figura 4.14 se puede ver el resultado generado por la base de datos y enviado a la aplicación cliente a través de la aplicación en el servidor. Los resultados se despliegan en una nueva pantalla con la opción de regresar a la anterior y realizar una nueva búsqueda con parámetros diferentes. Las instrucciones utilizadas para el envío y recepción de datos se encuentran en la tabla 4.7.

Fig. 4.14 Recepción de datos en la aplicación

```
public class Client extends BluetoothMIDlet {
    private Form connForm;
    private Command connectCommand;

    class Message implements Runnable{
        …
        }
        public void run(){
            try{
                byte[] data = theMessage.getBytes();
                output.write(data);
                output.flush();
                cadena = "Cadena enviada" ;
                connForm.append(cadena);
```

```
        int length = input.read(mess);
        mensaje = mensaje.concat(new String(mess, 0, length));
        mensaje = mensaje.trim();
        connForm.append("\n");
        connForm.append(mensaje);
        cadena = "\n Cadena leída\n";
        connForm.append(cadena);
}...
```

Tabla 4.7 *Envío y recepción del flujo de datos*

La interfaz de la aplicación en el móvil varía dependiendo de la marca y modelo del dispositivo; no obstante, su funcionamiento se mantiene constante.

Para agilizar el tráfico de datos, la conexión con el servidor no se cierra sino hasta salir de la aplicación, en su lugar lo que se cierra son los buffer de entrada y salida como lo muestra la tabla 4.8.

```
public class Client extends BluetoothMIDlet {
   private Form connForm;
   private Command connectCommand;

    public void commandAction(Command c, Displayable d){
      switch (c.getCommandType()){
         case Command.OK:
            …
            connected = true;
            connForm.deleteAll();
            Display.getDisplay(this).setCurrent(connForm);
            new Thread(this).start();
            connectToServer(address);
           …
            String msg = getMsg();
            new Thread(new Message(msg + "\n", input, output)).start();
            break;
         case Command.BACK:
           ..
            connForm.deleteAll();
            Display.getDisplay(this).setCurrent(queryForm);
            break;

         case Command.EXIT:
            if(connected){
               closeConnection();
```

```
            }
            destroyApp(false);
            notifyDestroyed();
            break; }
    }
    …
    class Message implements Runnable{
        …
        public void run(){
            try{
                …
                output.write(data);
                ..
                int length = input.read(mess);
                ..
                connForm.append(mensaje);
                cadena = "\n Cadena leída\n";
                connForm.append(cadena);
input.close();
                output.close();
 }…
```

Tabla 4.8 *Gestión de la conexión*

Capítulo 5: Impacto ambiental y biológico

5.1 Introducción

Son muchas las ventajas que ofrece la comunicación inalámbrica pero también debe considerarse el impacto ecológico y biológico que representan.

En el área ecológica, se considera que la multiplicada densidad electromagnética del ambiente ha generando un nuevo tipo de polución, intangible e inmaterial, denominada "contaminación electromagnética". La acumulación de estas emisiones genera un fenómeno que se ha llamado "electrosmog". [49]

En 1974, la Asociación Internacional para la Protección contra la Radiación (IRPA) formó un grupo de trabajo para Radiaciones No- Ionizantes, el cual examinó los problemas suscitados en el campo de la protección contra los varios tipos de Radiaciones No- Ionizantes (NIR por sus siglas en inglés Non-ionizing Radiation). En el Congreso de la IRPA en París en 1977, este grupo de trabajo se convirtió en Comité Internacional para las Radiaciones No- Ionizantes (INIRC)

En cooperación con la División de Salud Ambiental de la Organización Mundial de la Salud (OMS), la IRPA/ INIRC desarrolló un número de documentos sobre criterios de salud en relación a las RNI, como parte del Programa de Criterios de Salud Ambiental de la OMS, Cada documento incluye una visión panorámica de las características físicas, mediciones e instrumentación, fuentes, y aplicaciones de las RNI, una revisión total de la literatura sobre los efectos biológicos y una evaluación de los riesgos a la salud provenientes de la exposición a las RNI. [48]

5.2 Propósito y campo de acción

El término de radiación no ionizante se aplica a todas las formas de radiación electromagnética cuya interacción con la materia no produce ionización (Ver figura 5.1).

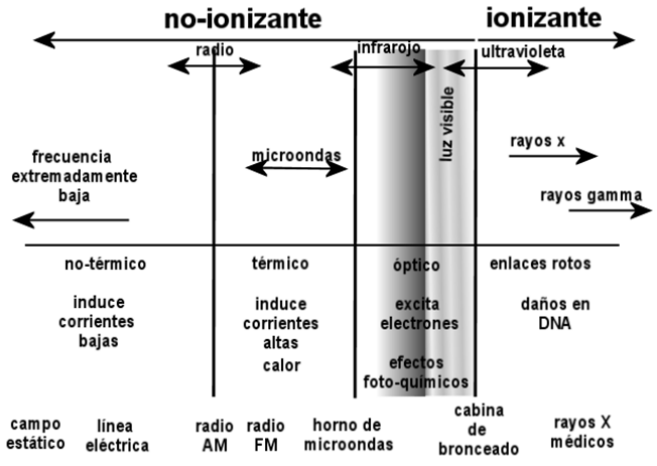


Fig. 5.1 Radiación ionizante y no-ionizante

Se considera como NIR a la radiación electromagnética con longitudes de ondas mayores a los 100nm, equivalente a energías cuánticas de 12 eV.

Para cuestiones de salud la NIR electromagnética puede subdividirse en rangos que tengan como parámetros de referencia la longitud de onda (λ) y la frecuencia (v):

- Radiación ultravioleta (UV), 100 nm ≤ λ ≤ 400 nm (radiación óptica)
- Radiación visible 400 nm ≤ λ ≤ 760 nm (radiación óptica)
- Radiación Infrarrojo (IR), 760 nm ≤ λ ≤ 1 mm(radiación óptica)
- Radiofrecuencia (RF) radiación que incluye microondas (MW), 300 Hz ≤ v ≤ 300 GHz correspondientes a 1000 km ≥ λ ≥ 1 mm
- Frecuencia extremadamente baja (ELF por sus siglas en Inglés Extremely Low Frequency) campos (v ≤ 300 Hz), en la práctica principalmente frecuencias de 50-60 Hz

La ICNIRP establece recomendaciones para limitar la exposición (ocupacional y poblacional) a los CEM (Campos Electromagnéticos) con el objetivo de proveer protección contra efectos adversos a la salud conocidos.

Se considera como un efecto adverso a la salud aquel que cause un deterioro detectable de la salud de los individuos expuestos o su descendencia. Los efectos de los CEM se consideran como directos cuando resultan de la interacción directa de los campos con el cuerpo y como indirectos si involucran la interacción con un objeto a un potencial eléctrico diferente del cuerpo.

Son dos las clases de recomendaciones que presenta la ICNIRP en sus estudios sobre “Bases para limitar la exposición a campos eléctrico, magnético y electromagnético”: restricciones básicas y niveles de referencia.

Restricciones básicas

Son restricciones a la exposición a campos eléctricos, magnéticos y electromagnéticos variables en el tiempo basadas directamente en los efectos en la salud establecidos. Dependiendo de la frecuencia del campo, las cantidades físicas usadas para especificar estas restricciones son la densidad de corriente (J), la tasa de absorción específica de energía (SAR), y la densidad de potencia (S).

Niveles de referencia

Algunos niveles de referencia son derivados de restricciones básicas (algunos están basados en percepciones y efectos indirectos adversos por la exposición a los CEM) mediante el uso de modelos matemáticos y por extrapolación de los resultados de las investigaciones de laboratorio en frecuencias específicas. Las cantidades derivadas son la intensidad de campo eléctrico (E), la intensidad de campo magnético (H), la densidad de flujo magnético (B), la densidad de potencia (S) y las corrientes que fluyen a través de las extremidades (IL). Las cantidades que están dirigidas a la percepción y otros efectos indirectos son las corrientes de contacto (IC) y, para campos pulsantes, la absorción de energía específica (SA). Si los valores medidos o calculados exceden los niveles de referencia, no necesariamente son excedidas las restricciones básicas. Sin embargo, siempre que un nivel de referencia sea excedido, es necesario evaluar el cumplimiento de la restricción básica relevante y determinar si son necesarias medidas de protección adicionales.

5.3 Bases para limitar la exposición

Las recomendaciones dadas por el ICNIRP están basadas en efectos inmediatos a la salud proveniente de exposiciones de corto plazo, tales como la estimulación en los nervios periféricos y músculos, choques eléctricos y quemaduras causadas por tocar objetos conductores, y la generación de temperaturas elevadas en los tejidos resultante de la absorción de energía

durante la exposición a CEM. Los estudios in vitro no fueron considerados para proporcionar información útil como base primaria para evaluar los posibles efectos a la salud provenientes de los CEM.

Se consideran tres tipos de mecanismos de acoplamiento básicos a través de los cuales interactúan los campos eléctricos y magnéticos variables en el tiempo con la materia viva:

- Acoplamiento a campos eléctricos de baja frecuencia
- Acoplamiento a campos magnéticos de baja frecuencia
- Absorción de energía de los campos electromagnéticos

La exposición de materia a los campos eléctricos y magnéticos produce una absorción insignificante de energía; sin embargo, la exposición a los campos electromagnéticos a frecuencias por encima de los 100 KHz puede producir una absorción de energía y un incremento de temperatura significativos.

Los campos electromagnéticos pueden ser divididos en cuatro rangos de frecuencias (Durney et al. 1985) según la capacidad de absorción de energía del cuerpo humano:

- 100 kHz < f < 20 MHz, en las cuales la absorción en el tórax decrece rápidamente con la disminución de la frecuencia, y absorción significante puede ocurrir en el cuello y las piernas.
- 20 MHz < f < 300 MHz, en las cuales una absorción relativamente alta puede ocurrir en todo el cuerpo, y aún valores más altos si se consideran las resonancias parciales del cuerpo (ej. cabeza).
- 300 MHz< f < a varios GHz, en las cuales ocurre una absorción no- uniforme significativamente local.
- 10 GHz<<f en las cuales la absorción de energía ocurre principalmente en la superficie del cuerpo.

En los tejidos, el SAR es proporcional al cuadrado de campo eléctrico interno, sus valores dependen de los siguientes factores:

- Los parámetros de campos incidentes, p. ej., la frecuencia, la intensidad, polarización, y la configuración fuente-objeto (campo cercano o lejano)
- Las características del cuerpo expuesto, es decir su tamaño, su geometría interna y externa y las propiedades dieléctricas de sus tejidos varios.
- Los efectos de la tierra eléctrica y los efectos de reflexión de otros objetos en el campo cercano del cuerpo expuesto.

La utilidad de cálculos matemáticos de modelos numéricos, así como las mediciones de corrientes inducidas e intensidad de campo en los tejidos, para evaluar la exposición de campo cercano ha sido demostrada para teléfonos móviles, walkie-talkies, torres de radiodifusión, fuentes de comunicación entre barcos y calentadores dieléctricos. La importancia de estos estudios se basa en su demostración de que la exposición a los campos cercanos puede producir un SAR alto localizado (ej. en la cabeza, muñecas y tobillos) y que el SAR de cuerpo entero y el SAR localizado son fuertemente dependientes de la distancia de separación entre la fuente de alta frecuencia y el cuerpo. Finalmente, los datos de SAR obtenidos mediante mediciones son consistentes con la información obtenida de los cálculos matemáticos con modelación numérica. El SAR promedio de cuerpo entero y el SAR localizado son cantidades convenientes para comparar los efectos observados bajo condiciones variadas de exposición. A frecuencias mayores de 10 GHz, la profundidad de penetración del campo en los tejidos es pequeña, y el SAR no es una buena medida para evaluar la energía absorbida.

Los estudios por Chatterjee et al. demostraron que conforme la frecuencia aumenta de aproximadamente 100 kHz hasta 10 MHz, el efecto dominante de la exposición a un campo

electromagnético de alta intensidad cambia del estímulo del nervio y del músculo a la calefacción. En 100 kHz la sensación primaria era la de un nervio que zumbaba, mientras que en 10 MHz era uno de calor en la piel. En este rango de frecuencia, por lo tanto, los criterios básicos de protección sanitaria deben ser por ejemplo evitar el estímulo de tejidos excitables y los efectos térmicos. En las frecuencias a partir de 10 MHz a 300 GHz, la calefacción es el efecto principal de la absorción de la energía electromagnética, y las subidas de temperatura de más de 1-2 °C pueden tener efectos de salud adversos tales como agotamiento por calor y ataque de calor). Los estudios de trabajadores en ambientes térmicos agotadores han mostrado el empeoramiento del funcionamiento de tareas simples conforme la temperatura del cuerpo se eleva a un nivel que se acerca al estrés de calor fisiológico. Una sensación de calor ha sido señalada por voluntarios que experimentaban una corriente de alta frecuencia de cerca de 100-200 mA a través de una extremidad. El valor del SAR resultante es poco probable que produzca un incremento localizado de la temperatura de más de 1 °C en las extremidades, que se ha sugerido como el límite superior de incremento de temperatura sin efectos perjudiciales a la salud. Datos sobre los voluntarios reportados para las frecuencias hasta 50 MHz y para las frecuencias de hasta 110 MHz (el límite superior a la banda de la radiodifusión en FM) reportan un nivel de referencia para corrientes en las extremidades de 100 mA para evitar efectos térmicos excesivos. En general, éstos han demostrado que la exposición por hasta 30 minutos, bajo condiciones en las cuales el SAR de todo el cuerpo era menos de 4 W kg^{-1}, causó un aumento en la temperatura del cuerpo de menos de 1 °C.

5.4 Estudios celulares y animales

La exposición de animales de laboratorio a CEM que producían una absorción en exceso de aproximadamente 4 W kg^{-1} ha revelado un modelo característico de la respuesta termorreguladora, en el cual la temperatura del cuerpo inicialmente sube y luego se estabiliza siguiendo la activación de los mecanismos termorreguladores. La fase temprana de esta respuesta es acompañada por un aumento en el volumen de la sangre debido al movimiento del líquido del espacio extracelular hacia la circulación y por aumentos en el ritmo cardíaco y la presión interventricular de la sangre. Estos cambios cardiodinámicos reflejan las respuestas termorreguladoras que facilitan la conducción del calor a la superficie del cuerpo. La exposición prolongada de animales a niveles de radiación de microondas que elevan la temperatura del cuerpo conduce en última instancia al colapso de estos mecanismos termorreguladores.
Varios estudios en roedores y monos demuestran una disminución del rendimiento para la realización de tareas en valores del SAR en el rango 1-3 W kg^{-1}. En los monos, la alteración del comportamiento causada por el sistema termorregulador se inicia cuando la temperatura de la región hipotalámica se incrementa en valores tan pequeños como 0.2-0.3°C. El hipotálamo es considerado ser el centro control del proceso termorregulador normal, y su actividad puede ser modificada por un pequeño aumento de temperatura local bajo condiciones en que la melanoma en ratones o del glioma cerebral en ratas.

En el rango de frecuencia de cerca de 100 kHz.- 110 MHz, shocks eléctricos y quemaduras pueden darse en un individuo que toca un objeto de metal que no está conectado a tierra, y ha adquirido una carga en un campo, o del contacto de un individuo cargado y un objeto de metal puesto a tierra. Las corrientes de umbral que dan lugar a efectos biológicos, que se extienden en severidad desde la percepción hasta el dolor se han medido en experimentos controlados en voluntarios; éstos se resumen en la Tabla 5.1. En general, se ha mostrado que las corrientes de umbral que producen la percepción y el dolor varían poco en el rango de frecuencia 100 kHz- 1 MHz y es poco probable que varíen significativamente en el rango de frecuencia hasta cerca de 110 MHz Según lo observado anteriormente para frecuencias más bajas, las variaciones significativas entre las sensibilidades de hombres, mujeres, y niños también existen para los campos de frecuencia más altas.

Efecto Indirecto	Umbral de corriente (mA) a una frecuencia dada	
	100 KHz	1 MHz
Percepción al tocar	25- 40	25-40
Dolor en el dedo que hace contacto	33- 55	28- 50
Descarga dolorosa/ umbral let-go	112- 224	No determinado
Descarga severa/ dificultad para respirar	160- 320	No determinado

Tabla 5.1 Rangos de corrientes umbral para efectos indirectos, incluyendo niños, mujeres y hombres

5.5 Restricciones básicas

Diferentes bases científicas fueron usadas en el desarrollo de las restricciones básicas para varios rangos de frecuencia:

- Entre 1 Hz y 10 MHz, las restricciones básicas están dadas en términos de la densidad de corriente, para prevenir daños funcionales en el sistema nervioso.
- Entre 100 kHz y 10 GHz, las restricciones básicas son provistas en términos del SAR para prevenir el estrés térmico de todo el cuerpo y un calentamiento localizado excesivo en los tejidos. En el rango de 100 kHz – 100 MHz, las restricciones son provistas en términos de la densidad de corriente y del SAR.
- Entre 10 y 300 GHz, son provistas en términos de la densidad de potencia para prevenir el calentamiento excesivo en los tejidos o cerca de la superficie del cuerpo.

En el rango de frecuencia de unos pocos Hz a 1 kHz, para niveles de densidad de corriente inducida por encima de 100 mA m^{-2}, los umbrales para cambios agudos en la excitabilidad del sistema nerviosos central y otros efectos agudos como la reversión del potencial evocado visualmente, son excedidos. En el rango de las frecuencias de 4Hz a 1 kHz, la exposición ocupacional debería estar limitada a campos que induzcan densidades de corriente menores a 10 mA m^{-2}. Para el público en general un factor adicional de 5 es aplicado, dando una restricción básica de la exposición de 2 mA m^{-2}. Por debajo de 4 Hz y por encima de 1 kHz, la restricción básica basada en la densidad de corriente inducida se incrementa progresivamente, correspondiendo al incremento del umbral para la estimulación de los nervios para estos rangos de frecuencia.

Los efectos biológicos y a la salud establecidos en el rango de frecuencias de 10 MHz a unos pocos GHz son consistentes con las respuestas al incremento de temperatura del cuerpo en más de 1 ºC. Este nivel de incremento de temperatura resulta de la exposición de individuos bajo condiciones ambientales moderadas a un SAR de cuerpo entero de 4 Wkg^{-1} por cerca de 30 minutos. Por lo tanto se ha escogido un SAR de cuerpo entero promedio de 0,4 Wkg^{-1} como la restricción que provee protección adecuada para exposición ocupacional. Un factor de protección adicional de 5 es introducido para exposición al público dando un límite de SAR de cuerpo entero promedio de 0,08 Wkg^{-1}. Las restricciones básicas más bajas para exposición al público en general toman en cuenta el factor que su edad o estado de salud puede diferir del de los trabajadores.

Las restricciones básicas para densidades de corriente, SAR de cuerpo entero promedio y SAR localizado para frecuencias entre 1Hz y 10 GHz son presentadas en la Tabla 5.1 y las Restricciones Básicas para densidad de potencia para frecuencias entre 10 y 300 GHz se encuentran en la tabla 5.2.

Características de la exposición	**Rango de frecuencias**	**Densidad de corriente para cabeza y tronco** $\left(\frac{mA}{m^2}\right)(rms)$	**SAR promedio en todo el cuerpo** $\frac{W}{kg}$	**SAR localizado cabeza y tronco** $\frac{W}{kg}$	**SAR localizado (extremidades)** $\frac{W}{kg}$
Exposición ocupacional	Hasta 1Hz	40	--	--	--
	1-4 Hz	40/f	--	--	--
	4Hz-1kHz	10	--	--	--
	1-100 KHz	f/100	--	--	--
	100 kHz-10MHz	f/100	0.4	10	20
	10MHz-10GHz	--	0.4	10	20
Exposición del público en general	Hasta 1Hz	8	--	--	--
	1-4 Hz	8/f	--	--	--
	4Hz-1kHz	2	--	--	--
	1-100 KHz	f/500	--	--	--
	100 kHz-10MHz	f/500	0.8	2	4
	10MHz-10GHz	--	0.8	2	4

Tabla 5.2 Restricciones básicas para exposiciones a campos eléctrico y magnéticos para frecuencias de hasta 10GHz

Tipo de Exposición	**Densidad de Potencia (Wm^{-2})**
Exposición Ocupacional	50
Exposición del Público en General	10

Tabla 5.3 Restricciones básicas para densidad de potencia para frecuencias entre 10 y 300 GHz

Aunque no se tiene la certeza de los efectos sanitarios de las ondas electromagnéticas, se sugiere el uso de tecnologías alternativas como la fibra óptica y disminuir la utilización de tecnologías inalámbricas hasta que no se compruebe que estas radiaciones no son peligrosas para la salud.

Conclusiones

La razón por la que seleccioné la tecnología bluetooth por sobre: WiFi o IrDA se debe a la difusión, accesibilidad y alcance que tiene esta tecnología. Prueba de ello son los más de 4 mil miembros inscritos en el SIG Bluetooth y los más de 250 millones de dispositivos equipados con bluetooth. Aunado a ello encontré muy conveniente el hecho de que su uso, tanto para el usuario objetivo como para mí, no involucrara algún costo más allá de la adquisición del hardware (tarjeta, dongle o chip bluetooth). Estas características, considero, son indispensables para una efectiva propaganda de la cartelera cultural ya que permitirá su difusión a un mayor número de personas.

Una vez seleccionada la tecnología a emplear fue preciso comprender la estructura de la pila bluetooth ya que esto me proporcionaría las bases para establecer el protocolo de transporte y controlador necesarios para la aplicación. Consideré entonces una visión generalizada en la que ésta se divide en dos componentes: el host bluetooth y el controlador bluetooth.

El controlador lo conforma el hardware: USB, PCMCIA, UART, etc., y con él las capas de radiofrecuencia, banda base, y el administrador de enlace. La parte lógica se encuentra en el software del host y forma parte del sistema. Las capas que la componen son: Aplicación, SDP, RFCOMM, L2CAP, Control y Datos. La conexión entre la capa lógica y la física se realiza a través del HCI. El módulo de radio se conecta con el host a través de un mecanismo de entrada/salida.

Aunque la mayoría de los Sistemas Operativos cuentan con soporte básico para bluetooth (controlador bluetooth); algunos no son compatibles con ciertos protocolos de transporte, generalmente L2CAP y RFCOMM. Aunado a ello, la elección del controlador también se supeditaba a la compatibilidad con las librerías de bluecove (API bluetooth para java). Son 3 los controladores compatibles con bluecove y con el SO Windows: Broadcom WIDCOMM, Microsoft Winsock, IVT Bluesoleil.

La primera prueba la realicé con el controlador de Bluesoleil. El SDK de Bluesoleil es muy limitado y presentó problemas a la hora de abrir el socket de conexión. El siguiente controlador fue el de WIDCOMM, seguido del de Microsoft. Ambos controladores funcionaron correctamente con la aplicación pero debe destacarse que el controlador de Microsoft sólo soporta conexiones por RFCOMM mientras que el de WIDCOMM requiere ciertos requisitos para que funcione correctamente como lo son algunas dll's y la pila BTW 5.5 /6.1.

Saber esto es importante para una futura actualización y modificación del funcionamiento de la aplicación. Si se requiriera del envío de archivos de audio y video, sería preciso contemplar un controlador con soporte para L2CAP, en caso de sólo manejar objetos binarios entonces OBEX bastaría.

En cuanto a las características de la señal electromagnética, bluetooth opera sobre la banda ISM1 a 2.4GHz, transmite a una velocidad promedio de 1Mbps y su alcance varía según la antena que se ocupe. La antena bluetooth más potente alcanza una distancia de poco más de cien metros en condiciones óptimas. En esta tesis trabajé con una antena de clase 2 (alcance de 10m y potencia de salida de 2.5mW) y clase 3 (alcance de aproximadamente 1m y potencia de salida de 1mW), obteniendo una comunicación exitosa de 7 metros con obstáculos entre los equipos con la primera y de 2 metros de distancia sin obstáculos entre el servidor y el dispositivo móvil con la segunda. Esto resalta la importancia de contar con una antena bluetooth lo suficientemente potente para lograr un mayor alcance y, por ende, un mayor número de beneficiados.

Respecto a la configuración de red, 8 dispositivos bluetooth conforman una piconet, de los cuales uno fungirá como maestro y los demás como esclavos. La dualidad de comportamiento de los dispositivos para que asuman el papel de maestro o esclavo permite a su vez crear scatternet's; esto es, una unión entre dos o más piconet's. Por otra parte, RFCOMM soporta hasta 60 conexiones simultáneas (canales), lo que significa una ventaja para los dispositivos bluetooth en cuestión porque podrán utilizar los demás canales para cualquier otra aplicación.

Considerando lo anteriormente dicho resalta el hecho de que el limitado número de dispositivos que pueden conformar una piconet plantea un problema; esto es, el número de consultas simultáneas se reduce a siete dispositivos clientes y dado que el tiempo de envío-recepción de datos de la aplicación de esta tesis es de aproximadamente 10 segundos con un canal en estado de escucha, los demás dispositivos que requieran realizar una consulta deberán mantenerse a la espera de la liberación de los canales por ese mismo lapso de tiempo.

Más allá de conocer la configuración de la pila bluetooth y sus formas de conexión, también fue necesario establecer el tipo de API (alto o bajo nivel) con el que se deseaba trabajar ya que de ello depende la interfaz gráfica y el comportamiento de las aplicaciones. La cadena de conexión y apertura de canales que se generan para el envío y recepción de datos dependen directamente de la capa de transporte, fuera de eso, el ciclo de vida del MIDlet siguió el de cualquier aplicación que no requiera conexión bluetooth (pausado, activado, destruido).

La elección del API de alto nivel y del lenguaje de programación java obedece a la compatibilidad con los diferentes dispositivos y al soporte que existe para las aplicaciones desarrolladas con este lenguaje.

Seleccionados el protocolo de comunicación, protocolo de transporte, controlador y lenguaje de programación, el paso siguiente era verificar las cuestiones de seguridad.

Bluetooth es conocido por ser un protocolo de comunicación inseguro aún a pesar de la implementación de los cuatro métodos de seguridad establecidos en la cadena de conexión: emparejamiento, autenticación, encriptación y autorización; mas esto se puede corregir añadiendo un cifrado propio en la aplicación del host. Para los fines que a esta tesis conviene el emparejamiento basta como técnica de seguridad ya que considero que la información manejada es de dominio público y el enlace no requiere de algún tipo de cifrado. Después de realizar el emparejamiento por primera vez, se creará un registro dentro el dispositivo local haciendo innecesario repetir el procedimiento, agilizando así la comunicación.

Otra de las características útiles por agilizar el proceso de conexión que implementé, fueron los servicios denominados "ejecutar antes de conectar". Esto es: el servidor ejecuta la aplicación y se mantiene a la espera de solicitudes de conexión de dispositivos remotos mientras que de forma remota los dispositivos cliente intentarán establecer una conexión con él a través de una dirección bluetooth y canal especificados en la cadena de conexión. Esto disminuía el tiempo de conexión en aproximadamente 2 segundos.

Implementar una aplicación JABWT requirió de ciertas exigencias en software (la versión correcta del CLDC y MIDP, contar con la KVM, compatibilidad con el JSR-82) y hardware (pila y radio bluetooth, 8kb de memoria persistente, 160kb en memoria volátil y procesador de 16 bits). Propiamente la pila bluetooth debe facilitar el soporte para métodos de seguridad e identificación.

En cuanto al MIDlet se refiere, los dispositivos cliente contaron con un gestor de aplicaciones encargado de la carga, ejecución y desinstalación de éste. Algunos dispositivos requirieron de los archivos jad y jar para la instalación de la aplicación, otros, la mayoría, sólo requirió del jar.

Tomando en cuenta el funcionamiento y alcance de las comunicaciones bluetooth, así como de los usuarios que habrían de tener acceso al sistema, generé un diseño del que se desprendieron

siete casos de uso: consulta de cartelera, con sus respectivas modalidades web y bluetooth; agregar, editar y eliminar evento; agregar, editar y eliminar usuario.

El diseño de la aplicación contempló la parte web principalmente para agregar registros a la base de datos de forma remota y restringir el acceso a la misma. La aplicación en el servidor sirve como enlace entre la base de datos y el MIDlet instalado en el móvil.

Consideré la existencia de una base de datos donde parte de los usuarios a los que va dirigida la aplicación ya se encuentran registrados. Esta base de datos es gestionada por la secretaría de servicios académicos.

El alcance de este modelo para la divulgación de contenidos es tal que su implementación está siendo ampliamente difundida en diferentes sectores bajo el nombre de “publicidad bluetooth”. Esto se explica dado que su uso e implementación representan un bajo costo y sus beneficios son cuantiosos. Empero, debe estarse consciente de las limitaciones del uso del protocolo.

Glosario

Abstracción de grupo: Función realizada en la capa L2CAP. Permite el mapeo eficiente de grupos de protocolos en piconets.

ACL: Asynchronous Connectionless (Sin conexión asíncrona) Conexión de bajo nivel que permite controlar el flujo de tráfico.

Ad hoc: Locución latina que significa literalmente "para esto". En redes de comunicación, hace referencia a una red en la que no hay un nodo central, sino que todos los ordenadores están en igualdad de condiciones.

API: Application Programming Interfaz (Interfaz de Programación de Aplicaciones) Es un conjunto de funciones y procedimientos que permiten a las aplicaciones ejecutarse en diferentes tipos de hardware, sistemas operativos y dispositivos.

Aplicaciones heredadas: Aquellas que son heredadas de lenguajes, plataformas y técnicas anteriores a las actuales tecnologías. En escenarios de seguridad en la capa de enlace, se refieren a las aplicaciones que no pueden hacer llamadas al administrador de seguridad por sí mismas sino que hacen uso de un adaptador para realizar esta tarea.

Asimétrico: Un tipo de enlace sin conexión asíncrona que opera bajo dos velocidades, una para el upstream y otra para el downstream.

Asíncrono: Forma de comunicación que asigna a cada byte con un bit de inicio y uno de término como la forma de sincronizar la transmisión entre los dispositivos de envío y recepción.

Autenticación: Verificación de la identidad del dispositivo remoto en el local.

Autorización: Concesión de acceso a servicios. Se puede dar a través de la confirmación del usuario o a través de una relación de confianza.

Bandabase: Especifica la forma de procesamiento de la señal digital en el hardware, particularmente en el control del enlace. La bandabase es la encargada de la gestión de los enlaces y canales físicos. La especificación define dos tipos de enlaces: SCO (Orientado a Conexiones Síncronas) y ACL (Enlaces Sin Conexión Asíncronos).

BCC: Bluetooth Control Center (Centro de Control Bluetooth) set de capacidades que permiten al usuario, fabricante o proveedor de servicios bluetooth, definir valores específicos para la configuración de parámetros en la pila bluetooth.

Bit: Unidad binaria (1 ó 0). En grupos de ocho forman un Byte.

BQB: Bluetooth Qualification Bodies (Cuerpo de Certificación Bluetooth) Entidad responsable de la revisión de declaraciones y documentos que vayan en contra de los requerimientos bluetooth establecidos; revisión de reportes de pruebas a productos y listado de aquellos que cumplan con las normas en la base oficial de productos certificados.

BQRB: Qualification Review Board (Mesa de Revisión de Certificación Bluetooth) Entidad responsable de la gestión, análisis y mejoras de los programas de certificación.
Byte Grupo de ocho bits.

Canal de Usuario Isócrono: Canal utilizado para información relacionada con el tiempo.

Canal Físico: La frecuencia de salto de una radio frecuencia sincronizada. Esta asociación en la capa bandabase se establece a través del paginado.

Canal Lógico: Canal que se conduce sobre un enlace físico.

Capa de Aplicación: Grupo de protocolos concernientes a las interfaces de usuario.

Carga: Aplicado a campos de datos, la carga es un paquete discreto de información. También se le considera como la medida de la cantidad de trabajo que un sistema computacional puede procesar.

Carga útil: Porción de una trama de datos que contiene la información enviada útil al protocolo de la capa superior o al usuario.

CEM: Campo Electromagnético

Clase de Dispositivo: Parámetro utilizado durante el descubrimiento de dispositivos. El parámetro es establecido desde un dispositivo remoto indicando el tipo de dispositivo y servicios que soporta.

Clase Dispositivo Bluetooth: Parámetro que indica el tipo de dispositivo y de servicios que soporta. Su uso se da durante el proceso de descubrimiento en el nivel de interfaces de usuario.

CLDC: Connected Limited Device Configuration (Configuración de Dispositivo Conectado Limitado) Configuración con las clases e interfaces Java necesarias para el desarrollo de aplicaciones bajo la arquitectura J2ME.

Cliente: Dispositivo con la capacidad de guardar o extraer objetos desde y hacia el servidor.

Comunicación síncrona: Transmisión de datos que depende de un reloj común para la sincronización de flujo de datos entre dispositivos. El tiempo define el límite de datos.

Conexión: Fase subsecuente al enlace entre dispositivos donde todos los parámetros de conexión han sido establecidos.

Configuración: Concreción del API generalmente restrictiva que redefine parte de las clases.

Consciente: Se refiere al proceso en el que la intervención del usuario físico es necesaria para su implementación.

Consulta: Procedimiento en el que una unidad transmite mensajes con el fin de descubrir otras unidades activas dentro del área de cobertura.

Control de flujo: Control de la transferencia de datos entre dispositivos. En RFCOMM se utilizan los comandos se activa o desactiva el flujo de control a través de los parámetros FCON/FCOFF (flow control on/flow control off).

CRC: Cyclic Redundancy Check (Código de Redundancia Cíclica) Proceso que asegura la integridad de los paquetes de datos.

CSMA/CA: Carrier Sense Multiple Access with Collision Detection (Acceso múltiple por detección de portadora con evasión de colisiones) Método de acceso a redes en el que cada dispositivo anuncia su intención de transmitir datos para que los demás dispositivos eviten enviar información y de esta manera eludir colisiones.

Descubrimiento: Termino usado para describir los protocolos y mecanismos por los que un dispositivo en red o servicio de software identifica la red a la que está conectado e identifica los servicios en red disponibles.

Descubrimiento de dispositivos: Mecanismo de consulta y recepción de parámetros: dirección bluetooth, reloj, clase de dispositivo, modo de exploración de página utilizado y nombre de dispositivo.

Dispositivo confiable: Dispositivo que usa la tecnología bluetooth previamente autenticado y con acceso a otros dispositivos bluetooth haciendo uso de una llave en el nivel de enlace.

Dispositivo desconocido: Dispositivo bluetooth no conectado a un equipo local por lo que el proceso de emparejamiento no se ha llevado a cabo.

Dispositivo Descubrible: Dispositivo dentro del rango de alcance que puede responder a una consulta.

Dispositivo Local: Dispositivo que inicia el descubrimiento de servicios en el SDP. Para ello el dispositivo debe contar mínimamente con la parte cliente de la arquitectura SDP; específicamente la aplicación de descubrimiento de servicios.

Dispositivo no confiable: Dispositivo bluetooth desconocido para otro dispositivo. Puede requerir que el usuario realice algún tipo de operación antes de que el acceso le sea concedido.

Dispositivo Remoto: Cualquier dispositivo que participe en el descubrimiento de servicios respondiendo a consultas generadas por un dispositivo local bajo el perfil SDP.

Dispositivo silencioso: Dispositivo que no responde a las consultas de otro dispositivo ya sea porque se encuentra configurado como no-descubrible o por una congestión en la banda base.

DSSS WLAN: Direct Sequence Spread Spectrum WLAN (WLAN de Espectro Ensanchado de Secuencia Directa) Este tipo de enlace utiliza radio- transmisores que operan en un amplio rango de frecuencias bajo una frecuencia central fija.

Emparejamiento: Procedimiento de inicialización de la comunicación en el que dos dispositivos se comunican por primera vez y crean una llave de enlace para su futura autenticación.

Enlace Físico: Secuencia de intervalos de transmisión en un canal físico alternándose entre maestro y esclavo.

Escaneo/exploración de página: Proceso por el que un dispositivo escucha page-messages que contienen su propio DAC.
Esclavo: Unidad dentro de una piconet sincronizada con el maestro a través de su reloj y secuencias de salto.

Espectro Disperso: Técnica de decodificación digital en la que para transmitir la señal se "ensancha" a lo largo de una banda muy ancha de frecuencias.

Establecimiento del canal: Procedimiento en el que se establece un canal de comunicación a nivel lógico entre dos dispositivos que hagan uso del FTP.

Evento: Mensajes de entrada en la capa L2CAP junto con cualquier desconexión.

FCC: Federal Communications Commission (Comisión de Comunicaciones Federales) Agencia independiente de los Estados Unidos. Se encarga de la regulación de la comunicación interestatal e internacional por radio, televisión, cable, satélite y alámbrica.

FHSS: Frequency Hopping Spread Spectrum (Espectro Ensanchado por Salto de Frecuencia) Técnica de modulación donde la señal de emite sobre una serie de radiofrecuencias aleatorias saltando entre una y otra sincrónicamente con el transmisor. Normalmente se divide en 79 canales de longitud 1 MHz y realiza 1600 saltos por segundo.

GAP: Perfil de Acceso Genérico (GAP) Uno de cuatro perfiles generales manejados por el SIG. Define el uso de las capas inferiores de la pila bluetooth: el protocolo del gestor de enlace (LMP) el control de enlace (LC).

GCF: Generic Connection Framework (Esquema de Conexión Genérico) Set de interfaces definido en javax.microedition.io.package. El GCF proporciona siete interfaces y métodos de conexión compatibles con diferentes protocolos.

Gestor de Enlaces: Software encargado de la instalación y configuración del enlace, autenticación y otras funciones administrativas.

GSM: Global System for Mobile (Sistema Global para Móviles) Servicio digital celular estándar desarrollado en 1982 con soporte para voz y datos.

Home RF: Alternativa europea a los estándares IEEE 802.11b y Bluetooth en la banda de 2.4 GHz La velocidad máxima de HomeRF es 10 Mbps. Entre las ventajas anunciadas para esta tecnología se encuentra la seguridad y el menor consumo de potencia además de soporte para aplicaciones de telefonía y video.

ICNIRP: Comité Internacional para las Radiaciones No-ionizantes

IDE: Integrated Development Environment (Entorno de Desarrollo Integrado) Software que provee herramientas gráficas de desarrollo a los programadores como: editor de texto, compilador, intérprete, herramientas de construcción automática y depurador.

Inconsciente: Se refiere al proceso que no requiere de intervención explícita del usuario del dispositivo para su implementación.

Iniciador: Aquel dispositivo bluetooth que inicie una acción en un dispositivo remoto. El dispositivo que acepte la acción recibe en nombre de "acceptor".

Interfaz de Terminal del Host: Interfaz entre el host y la unidad bluetooth.

IRPA: Asociación Internacional para la Protección contra la Radiación

ISM: Internacional Scientific Medical Band (Banda Internacional Médico Científica) Banda de uso no comercial. El uso público de estas frecuencias hace necesario el uso de mecanismos contra interferencias como el ensanchado del espectro. Las redes que funcionan en esta banda se denominan de “espectro ensanchado”.

J2ME: Derivación del estándar Java creada para el desarrollo de programas aptos para ser descargados y ejecutados en dispositivos con recursos limitados.

Java: Lenguaje de programación en red desarrollado originalmente por Sun Microsystems. Fue creado para redes de plataformas cruzadas.

JCP: Java Community Process (Proceso de la Comunidad Java) El JCP coordina la evolución del lenguaje de programación Java.

Jini: Tecnología desarrollada por Sun Microsystems que provee mecanismos simples de conexión para formar una comunidad improptu -comunidad establecida sin planeación, instalación o intervención humana. Jini trabaja en capas altas mientras que bluetooth lo hace en más bajas.

JSR: Java Specification Request (Solicitud para Especificaciones Java) Documentos formales que describen las especificaciones y tecnologías propuestas para que sean añadidas a la plataforma Java.

Kit de Desarrollo Bluetooth: Set de herramientas de desarrollo que facilitan la creación de aplicaciones gracias al diseño de su ambiente de desarrollo.

KVM: Kernel-based Virtual Machine (Máquina Virtual basada en el Kernel) Es una versión de la Java Virtual Machine hecha en C para pequeños dispositivos con memoria limitada.

L2CAP: Logical Link Control and Adaptation Protocol (Protocolo de Control y Adaptación del Enlace Lógico) En esta capa se utiliza para pasar paquetes con y sin orientación a conexión a capas superiores. Sus funciones son: segmentación y re-ensamblado, multiplexación, gestionar conexiones unidireccionales y gestionar la calidad de servicio (QoS).

LAN inalámbrica: Usualmente se refiere a la red inalámbrica que sigue el estándar IEEE 802.11

Latencia: Retardo máximo aceptable entre la trasmisión de un bit y su transmisión inicial en el aire. Intervalo de tiempo que ocurre entre la ejecución de la operación de envío y el instante en que los datos comienzan a estar disponibles en el destino.

Lenguaje Procedural: Estos lenguajes basan su programación en el diseño ordenado de instrucciones que se ejecutarán una por una, de principio a fin. Cada instrucción es un comando para que el sistema realice una tarea específica. Ejemplos de lenguajes procedurales o imperativos son: Fortran, COBOL, Pascal, C y Ada

Llamada de Registro: Habilidad de la terminal para solicitar una "llamada al registro" y de la puerta de enlace para transmitir peticiones en la red local. Hacer una llamada al registro significa capturar el registro para permitir la entrada de futuros dígitos o realizar otras acciones. Esto también se conoce como gancho flash.

MAC Address: Media Access Control Address (Dirección de Control de Acceso Medio) Es un identificador de 48 bits (6 bloques hexadecimales) que corresponde de forma única a una Ethernet de red
Maestro: Dispositivo cuyo reloj y salto de frecuencia son utilizados para la sincronización con otros dispositivos dentro de una piconet.

MIDP: Mobile Information Device Profile (Perfil para Dispositivos de Información Móvil) Especificación publicada para el uso de Java en sistemas embebidos como teléfonos móviles o PDA’s.

Modo: Grupo de directivas que definen como debe responder un dispositivo bluetooth a ciertos eventos.

Modo Espera: Modo de operación en el que el dispositivo no recibe ni envía datos en un lapso de tiempo relativamente largo.

Modo Estático/Estacionado: Cuando un dispositivo esclavo no necesita participar en algún canal piconet pero permanece sincronizado al servidor.

Modo Inactivo: Un dispositivo entra en el modo inactivo cuando no posee enlaces establecidos con otros dispositivos.

Modo Sniff: Modo de operación en el que el ciclo de trabajo de una estación esclava se reduce, ya que sólo escucha cada M intervalos. Donde M es negociado con el gestor de enlace. El dispositivo maestro sólo puede comenzar la transmisión en intervalos específicos.

Modos de conexión: Estos hacen alusión al modo de conexión "conectable" y "no conectable". El primero se coloca así mismo en la página de exploración; el segundo no tiene la capacidad de cambiar este estado en la página de exploración.

Modulación por Pulsos Codificados: Procedimiento de modulación utilizado para transformar una señal analógica en una secuencia de bits.

Modo Público: Modo de operación en el que el dispositivo es visible a las consultas de la banda base.

Multiplexación del Protocolo: Función realizada en la capa L2CAP. L2CAP es capaz de distinguir entre los protocolos de las capas superiores como SDP, RFCOMM y TCS a través del campo "tipo."

NIR: Radiación No-Ionizante

Nivel de Seguridad de Dispositivo: Esta capa permite o deniega el acceso al dispositivo local. Existen dos niveles de seguridad: dispositivos confiables y dispositivos no confiables.

Nivel de Seguridad del Servicio: El nivel de seguridad de servicios puede conceder o denegar acceso a estos. Existen 3 niveles de seguridad: autorización y autenticación, autenticación, y sin seguridad.

OBEX: Object Exchange (Intercambio de Datos) Protocolo de comunicaciones para el intercambio de objetos binarios entre dispositivos. Se implementa sobre una pila en Bandabase/Gestor de Enlaces/L2CAP/RFCOMM, trabaja con transmisiones binarias y soporta sesiones.

OTA: Over the Air (En el Aire) Métodos para la distribución de actualizaciones de un software en teléfonos celulares o para proveer el entorno necesario para acceder servicios WAP o MMS.

OTAP: Over the Air Provisioning. Ver OTA

OTAPA: Over the Air Parameter Administration. Ver OTA

OTASP: Over the Air Service Provisioning. Ver OTA

Paginado: Procedimiento que involucra la transmisión de una serie de mensajes con el objetivo de establecer un enlace de comunicación hacia una unidad activa dentro del área de cobertura.

PAN: Personal Area Network (Red de Área Personal). Red formada por dispositivos cercanos a un punto de acceso debido a su corto alcance.

Paquete: En la especificación bluetooth, el formato de los bits agregados que comprenden una unidad discreta de información que puede ser transmitida en 1,3, o 5 intervalos de tiempo. Cada uno de los bloques en los que se divide la información a enviar a nivel de red.

PBX: Private Brand Exchange (Central Secundaria Privada Automática) Central Telefónica conectada directamente a la red pública de teléfono por medio de líneas troncales para la gestión de llamadas.

Perfil: Conjunto de características de un dispositivo o familia de dispositivos, define los protocolos y herramientas compatibles con un modelo de uso particular.

Perfil de Auricular: Define protocolos y procedimientos usados por dispositivos que implementen el modelo "Ultimate Headset".

Perfil de Fax: Define protocolos y procedimientos usados por dispositivos que implementen el modelo "Puntos de Acceso, WAN".

Perfil de Intercambio de acceso genérico: Define los protocolos y procedimientos usados en aplicaciones que provean modelos que requieran el intercambio de objetos.

Piconet: Red entre dos a ocho unidades bluetooth que comparten un mismo canal. La unidad que establece la conexión tomará el papel de maestro y las demás serán consideradas esclavas. Hasta diez piconets pueden coexistir en una misma área de cobertura.

PIN: Personal Identification Number (Número de Identificación Personal). Se usa para autenticar dos dispositivos que no hayan intercambiado previamente una llave de enlace.

Primitivos: Descripciones abstractas de funciones usadas por un servicio.

Protocolo: Conjunto de normas y/o procedimientos para la transmisión de datos que ha de ser observado por los dos extremos de un proceso comunicacional (emisor y receptor). Estos protocolos «gobiernan» formatos, modos de acceso, secuencias temporales, etc.

Protocolo de Control de Transmisión: El protocolo que garantiza la entrega de datos sin errores y conservando el orden en que se transmitieron. También proporciona un mecanismo para distinguir distintas aplicaciones dentro de una misma máquina, a través de puertos.

Protocolo de Aplicación Inalámbrica: Especificación para el envío y lectura de contenido en Internet y mensajes en dispositivos inalámbricos de capacidad limitada.

Protocolo de Datagramas de Usuario: Estándar TCP/IP. UDP proporciona un servicio de datagramas sin conexión que ofrece entrega de mejor esfuerzo, lo que significa que UDP no garantiza la entrega ni comprueba la secuencia de los datagramas. Un host de origen que necesita comunicación confiable debe utilizar TCP o un programa que proporcione sus propios servicios de secuencia y confirmación.

Protocolo de sesión inalámbrica: Establece la relación entre la aplicación cliente y el servidor WAP.

Protocolo de transferencia inalámbrica: Provee servicios que cumplen con muchos de los requerimientos TCP. WTP provee transporte bidireccional y unidireccional seguro y no seguro.

QoS: Quality of Service (Calidad del servicio). Las peticiones de configuración QoS contienen parámetros como latencia, variación del retardo y pico de la banda base.

RAD: Rapid Application Development (proceso de desarrollo de software. Es un método que comprende el desarrollo interactivo, la construcción de prototipos y el uso de utilidades CASE.

Radio Bluetooth: Transmisor-receptor que envía y recibe señales eléctricas moduladas entre dispositivos bluetooth.

RFCOMM: Radio Frequency Communication (Comunicación por radio frecuencia). Set de comandos de control que emulan una o más interfaces seriales.
Scatternet: Unión de dos o más piconets dentro de una misma área con o sin comunicación inter-piconet.

SAR: Absorción específica de energía

SCO: Synchronous Connection Oriented (Orientado a Conexiones Síncronas). Este tipo de conexión soporta la transferencia de audio en tiempo real.

SDAP: Perfil de Aplicación del Descubrimiento de Servicios (SDAP-Service Discovery Application Profile). Define los procedimientos por los que un dispositivo puede descubrir servicios, así como la información con ellos relacionada.

SDP: Services Discovery Protocol (Protocolo de Descubrimiento de Servicios). Protocolo encargado de la búsqueda de otros dispositivos Bluetooth y servicios Bluetooth dentro del rango de alcance.

Segmentación y Re-ensamble: Función realizada en la capa L2CAP que les proporciona a los dispositivos soporte para protocolos que usen paquetes más grandes de los soportados por la banda base. Los paquetes se segmentan antes de ser transmitidos. En la parte receptora, los paquetes se re-ensamblan y se efectúa un análisis simple de integridad.

Servidor: Dispositivo de almacenamiento de datos desde y hacia el que los objetos pueden guardar y extraer información.

Sesión Bluetooth: Sesión marcada por la actividad y participación de un dispositivo en una piconet.

SIG: Special Interest Group (Grupo de Interés Especial). Asociación privada sin ánimo de lucro con sede en Bellevue, Washington. A fecha de septiembre de 2007, el SIG está formado por más de 9000 compañías de telecomunicaciones, informática, automovilismo, música, textil, automatización industrial y tecnologías de red. Tiene pequeños grupos de personal dedicado al grupo en Hong Kong, Suecia y Estados Unidos. Los miembros del SIG dirigen el desarrollo de la tecnología inalámbrica Bluetooth, además de implementar y comercializar la tecnología en sus productos. El Bluetooth SIG por sí mismo no fabrica ni vende dispositivos Bluetooth.

Simétrico: Un tipo de ACL que ofrece la misma tasa de datos en el envío y recepción.

SPP: Serial Port Profile (Perfil de Puerto Serial). Define RFCOMM, L2CAP, SDP y los requerimientos y capacidades para la interoperabilidad con las capas más bajas del protocolo para realizar la emulación de cable serial.

SrvDscApp: Service Discovery Application (Aplicación para el Descubrimiento de Servicios). Aplicación en un dispositivo local que realiza descubrimiento de servicios en dispositivos remotos con los que tenga conexión.

Supervisión de enlace: Cada enlace bluetooth tiene un temporizador usado para la supervisión del enlace, Este temporizador se usa para detectar enlaces perdidos.
Tasa de Transferencia La tasa a la que se maneja para el tráfico de datos son permitidos, en bytes por segundo.

TCP: Transmission Control Protocol (Protocolo de Control de Transmisión), es un protocolo de comunicación de la capa 4 del modelo OSI, orientado a la conexión.

TCS: Telephony Control Protocol (Protocolo de control telefónico). Envía señales de control a dispositivos que requieren utilizar señales de audio en bluetooth.

TCK: Technology Compatibility Kit (Kit de Compatibilidad de Tecnología). Es una suite de pruebas encargadas de verificar el cumplimiento de supuestas aplicaciones JSR.

TDMA: Time Division Multiple Access (Acceso múltiple por división de tiempo) Técnica que permite la transmisión de señales digitales ocupando sólo un canal de trasmisión (frecuentemente de gran capacidad) a partir de distintas fuentes.

Tipo de Servicio Bluetooth: Uno o más servicios que el dispositivo local puede proveer a los remotos.

Unidad Bluetooth: Equipo con capacidad de transmisión inalámbrica de voz/datos en un rango de corto alcance.

Unidades de Datos de Protocolo: Se utiliza para el intercambio entre unidades parejas, dentro de una capa del modelo OSI. Los PDU's puede ser de datos o de control.

vCalendar: Formato para calendarizar y agendar información de forma virtual.

vCard: Tarjeta de presentación virtual.

WEP: Wired Equivalent Privacy (Privacidad Equivalente a Cableado). Sistema de cifrado incluido en el estándar IEEE 802.11 como protocolo para redes inalámbricas.

WLAN: Wireless Local Area Network (Red Inalámbrica de Área Local). Es un sistema de comunicación de datos inalámbrico flexible, muy utilizado como alternativa a las redes LAN cableadas o como extensión de éstas. Utiliza tecnología de radiofrecuencia que permite mayor movilidad a los usuarios al minimizar las conexiones cableadas. Las WLAN van adquiriendo importancia en muchos campos, como almacenes o para manufactura, en los que se transmite la información en tiempo real a una terminal central. También son muy populares en los hogares para compartir el acceso a Internet entre varias computadoras.

WMAN: Wireless Metropolitan Area Networks (Redes Inalámbricas de Área Metropolitana). Red de alta velocidad (banda ancha) que dando cobertura en un área geográfica extensa, proporciona capacidad de integración de múltiples servicios mediante la transmisión de datos, voz y vídeo.

WPAN: Wireless Personal Area Network (Red Inalámbrica de Área Personal). Red de dispositivos para la comunicación entre distintos equipos cercanos al punto de acceso.

WWAN: Wireless Wide Area Network (Red Inalámbricas de Área Extensa). Tienen el alcance más amplio de todas las redes inalámbricas. Por esta razón, todos los teléfonos móviles están conectados a una red inalámbrica de área extensa. Las tecnologías principales son: GSM (Global System for Mobile Communication), GPRS (General Packet Radio Service), UMTS (Universal Mobile Telecommunication System)

Referencias

[1] Muñoz Rodríguez David. *Sistemas Inalámbricos de Comunicación Personal*. Alfaomega 2002. México. Primera Edición. Pág. 1-44

[2] Kaveh Pahlavan Prashant Krishnamurthy. *Principles of Wireless Networks.* Prentice Hall 2002. EUA. Segunda Edición. Pág. 1-60

[3] Minoru Etoh. *Next Generation Mobile Systems 3G and Beyond.* John Wiley & Sons Ltd 2005. Inglaterra. Primera Edición. Pág. 257-282

[4] Álvarez García Alfonso, Morales Grela José Ángel. *Guía Práctica para Usuarios J2ME*. Anaya 2002. España. Primera Edición. Pág. 1-120

[5] Bala Kumar C., Kline Paul J., Thompson Timothy J. *Bluetooth application programming with the Java API's*. Morgan Kaufmann Publishers 2004. Estados Unidos de América. Primera Edición. Pág. 1-318

[6] Kammer David, McNutt Gordon, Senese Brian, Bray Jennifer. *Bluetooth application developer's guide: The short ranges interconnect solution*. Syngress 2002. Estados Unidos de América. Primera Edición. Pág. 1-101, 483

[7] Muller Nathan J. *Bluetooth Demystified*. Mc Graw Hill 2001. Estados Unidos de América. Primera Edición. Pág. 4-46,107-120,347-381

[8] Huang Albert S., Rudolph Larry. *Bluetooth Essentials for Programmers*. Cambridge Press 2007. Estados Unidos de América. Primera Edición. Pág. 137-156

[9] Kim Topley. *J2ME in a Nutshell, a desktop quick reference*. O'Reilly 2002. Estados Unidos de América. Primera Edición. Pág. 8-176, 254, 267-296

[10] Hopkins Bruce, Ranjith Antony. *Bluetooth for Java.* Apress 2003. Estados Unidos de América. Primera Edición. Pág. 1-83, 99-146, 194

[11] Froufe Quintas. *J2ME Java 2 Micro Edition, Manual de Usuario y Tutorial*. Agustín Jorge. Grupo Editorial Alfaomega 2004. España. Primera Edición. Pág. 1-44,47-90

[12] Álvarez García Alonso, Morales Grela José Angel.*J2ME.* Anaya Multimedia 2002. España. Primera Edición. Pág. 1-183

[13] Hunt John. *Guide to the Unified Process featuring UML, Java and design Patterns.* Springer 2003. Reino Unido. Primera Edición. Pág. 3-7, 21-37, 1-102, 131-139, 150

[14] Kruchten Philippe. *The Rational Unified Process, an Introduction*. Addison Wesley 2000. Estados Unidos. Primera Edición. Pág. 17-33

[15] Fowler Martin, Scott Kensall.*UML Distilled: A brief guide to the standard Object Modeling Language.* Addison Wesley 1999. Estados Unidos. Primera Edición. Pág. 1-16, 25, 35-37,53-61, 97

[16] Schach Stephen R. *Análisis y diseño orientado a objetos con UML y el proceso unificado*. Mc Graw Hill 2005. México. Primera edición. Pág. 1-240

[17] Roger Riggs. *Programming Wireless Devices with the Java 2 Platform*. Addison Wesley 2003. Estados Unidos de América. Primera Edición. Pág. 1-464

[18] Dreamtech Software Team. *Cracking the Code. WAP, Bluetooth and 3G Programming*. Hungry Minds 2002. Estados Unidos de América. Primera Edición. Pág. 127-147

[19] Klingsheim André N. *J2ME Bluetooth Programming (Master Thesis)*. Universitas Bergensis. Noruega 2004. Pág. 5-20

[20] http://www.netrino.com/Embedded-Systems/How-To/KVM-J2ME-Java-Virtual-Machine Blog dedicado a los sistemas embebidos (funcionamiento, lenguajes de programación, algoritmos, etc.). 27-Abril-2010

[21] http://java.sun.com/products/cldc/wp/ Informe cuyo propósito es describir el funcionamiento de la KVM para los dispositivos embebidos. 27-Abril-2010

[22] http://open.movilforum.com/taxonomy/term/71 Tutoriales para el manejo de J2ME. 27-Abril-2010

[23] http://es.kioskea.net/wireless/ Artículo sobre la clasificación y características de las redes inalámbricas según su alcance. 27-Abril-2010

[24] http://forums.sun.com/thread.jspa?forumID=82&threadID=772953 Foro Sun con tema de discusión concerniente a la simulación de la comunicación de aplicaciones desarrolladas en J2SE y J2ME. 27-Abril-2010

[25] http://www.todosymbian.com/postt11618.html Foro cuyo tema de discusión se centra en el soporte bluetooth j2me. 27-Abril-2010

[26] http://www.it.uc3m.es/celeste/docencia/j2me/tutoriales/bluetooth/EstudioTecnologico1_0.pdf Documento sobre el soporte bluetooth para j2me hecho por la Universidad de Madrid. 27-Abril-2010

[27] http://www.mygnet.net/codigos/j2me/apis/ejemplos_basicos_de_bluetooth.1409 Ejemplos prácticos de j2me y bluetooth. 27-Abril-2010

[28] http://www.todosymbian.com/secart43.html Artículo sobre la interacción entre Bluetooth y J2ME. 27-Abril-2010

[29] http://www.todosymbian.com/secart43-page7.html#nota1 Descripción del manejo del IDE Sun ONE Studio 5 ME para la creación de MIDlet's. 27-Abril-2010

[30] http://www.chinaitpower.com/A200508/2005-08-10/190999.html Aplicaciones con J2ME para su uso con Bluetooth. 27-Abril-2010

[31] http://es.kioskea.net/wireless/wman.php3 Definición de red MAN. 27-Abril-2010

[32] http://technet2.microsoft.com/WindowsServer/es/Library/f2552467-f693-4c14-b421-49cb2491bb363082.mspx?mfr=true Artículo sobre las redes inalámbricas. 27-Abril-2010

[33] http://forums.sun.com/thread.jspa?forumID=76&threadID=5400999 Foro Sun cuyo tema de estudio son las tecnologías Java ME, el CLDC y el MIDP. 27-Abril-2010

[34] http://en.wikipedia.org/wiki/IEEE_802.16 Definición del estándar IEEE 802.16. 27-Abril-2010

[35] http://paulmcpd.blogspot.com/2009/02/bluetooth-j2me-server-j2se-server-and.html Blog cuyo tema se enfoca en la interacción entre aplicaciones desarrolladas en J2ME y J2SE. 27-Abril-2010

[36] http://www.jsr82.com/simulating-jabwt-applications-using-jsr-82-simulators/ Artículo sobre la simulación de aplicaciones JABWT en ambientes JSR-82. 27-Abril-2010

[37] http://docs.google.com/viewer?a=v&q=cache%3A4vfETEUnfRkJ%3Aanteproyecto-protesis-mano-robotica.googlecode.com%2Ffiles%2FTEORIA_11_UML_componentes%2520e%2520interfaces%2520%28buen%25C3%25ADsimo%29.pdf+diagrama+de+componentes&hl=es&gl=mx&sig=AHIEtbRieaooEdgj7o6bQo95BdVmA_hMtQ&pli=1 Documento google sobre UML (Lenguaje de Modelado Unificado). 27-Abril-2010

[38] http://blog.undermedia.com.ec/index.php/relaciones-entre-clases-con-uml/ Artículo sobre el tipo de relaciones existentes entre entidades UML. 27-Abril-2010

[39] http://www.dcc.uchile.cl/~psalinas/uml/modelo.html Publicación sobre UML y el modelado de clases. 27-Abril-2010

[40] http://edutechwiki.unige.ch/en/UML_activity_diagram Definición de UML. 27-Abril-2010

[41] http://www.gentleware.com/fileadmin/media/archives/userguides/poseidon_users_guide/activitydiagram.html Artículo sobre la creación de Diagramas de Actividad en UML. 27-Abril-2010

[42] http://pensaenbinario.blogspot.com/2009/06/diagramas-de-secuencias.html Artículo sobre la creación de Diagramas de Secuencia en UML. 27-Abril-2010

[43] http://www.scribd.com/doc/2568098/UML-Diagramas-de-actividad Documento on-line sobre los diagramas de Actividad en UML. 27-Abril-2010

[44] http://www.bluetooth.com/Spanish/Technology/Pages/default.aspx Sitio oficial de Bluetooth. 27-Abril-2010

[45] http://es.wikipedia.org/wiki/Bluetooth Artículo sobre la tecnología Bluetooth: historia, usos y aplicaciones, versiones, información técnica. 27-Abril-2010

[46] http://profesores.elo.utfsm.cl/~agv/elo326/1s07/projects/DiegoGonzalez/archivos/MemoriaBluetooth.pdf Documento sobre la programación de dispositivos bluetooth con java bajo una configuración cliente-servidor (JSR82). 27-Abril-2010

[47] http://es.wikipedia.org/wiki/Harald_Bl%C3%A5tand Biografía de Harald Blätand. 27-Abril-2010

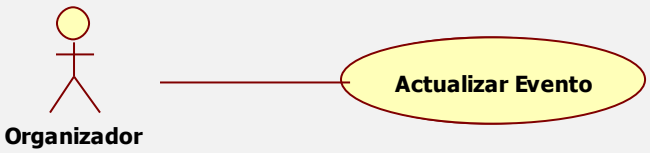
[48] http://www.icnirp.de/ Comisión internacional encargada divulgar información y consejos sobre los daños a la salud que puede causar la exposición a radiación no-ionizada. 27-Abril-2010

[49] http://contaminacion.ecoportal.net/content/view/full/87901 . Portal dedicado al medio ambiente, la naturaleza, los derechos humanos y la calidad de vida. 27-Abril-2010

[50] http://www.casadomo.com/noticiasDetalle.aspx?c=46&idm=56&m=164&n2=148&pat=148 Portal de domótica y empresas que ofrecen servicios relacionados con tecnologías para el hogar. 27-Abril-2010

Anexo A: Casos de uso

Caso de uso:	Actualizar evento
Actor:	Organizador



Descripción	El organizador edita los datos de un evento a la vez.
Precondiciones	1. El organizador debe contar con un usuario y contraseña que deberá proporcionar para su ingreso al sistema. 2. El organizador debe acceder al sistema por medio de un navegador de páginas Web 3. El organizador debe ser dueño del evento en cuestión.

Flujo Normal del Evento				
Actor		Sistema		Excepción
Paso	Acción	Paso	Acción	
1	El organizador selecciona el evento	2	El sistema busca el evento en la base de datos y despliega la información en el formulario “nuevo evento”	
3	El organizador edita los campos necesarios	4	El sistema valida la información, elimina el antiguo registro y crea el nuevo	E1
		5	El sistema redirecciona a la página principal	

Flujo Excepcional de Eventos		
Excepción	Descripción	Acción
E1	La base de datos no responde	Notificación al organizador

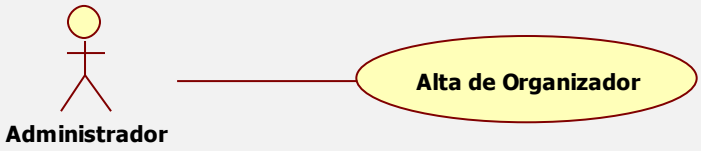
Caso de uso:	Eliminar evento
Actor:	Organizador

Descripción	El organizador elimina el evento seleccionado.
Precondiciones	1. El organizador debe acceder al sistema por medio de un navegador de páginas Web 2. El organizador debe contar con un usuario y contraseña que deberá proporcionar para su ingreso al sistema. 3. El organizador debe ser dueño del evento en cuestión.

Flujo Normal del Evento				
Actor		Sistema		Excepción
Paso	Acción	Paso	Acción	
1	El organizador selecciona el evento	2	El sistema elimina el evento y redirecciona	E1

Flujo Excepcional de Eventos		
Excepción	Descripción	Acción
E1	La base de datos no responde	Notificación al organizador

Caso de uso:	Alta de organizador
Actor:	Administrador



Descripción	El administrador ingresa un nuevo organizador a la BD cuyo estado será activo, por defecto.
Precondiciones	1. El administrador debe acceder al sistema por medio de un navegador de páginas Web 2. El administrador debe contar con un usuario y contraseña que deberá proporcionar para su ingreso al sistema. 3. El administrador debe considerar la existencia previa del organizador en la base de datos de las sociedades estudiantiles.

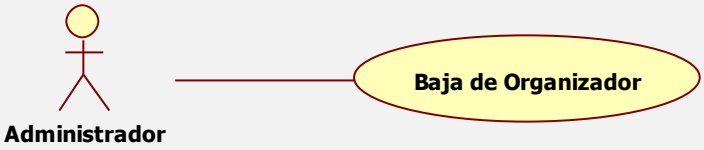
Flujo Normal del Evento

Actor		Sistema		Excepción
Paso	Acción	Paso	Acción	
1	El administrador proporciona los datos del Organizador	2	El sistema valida los datos y genera un registro en la BD	E1
		3	El sistema redirecciona a la página principal	

Flujo Excepcional de eventos

Excepción	Descripción	Acción
E1	Error al escribir en la BD	Notificación al administrador

Caso de uso:	Baja de organizador
Actor:	Administrador



Descripción	El administrador cambia el estado del organizador de activo a baja
Precondiciones	1. El administrador debe acceder al sistema por medio de un navegador de páginas Web 2. El administrador debe contar con un usuario y contraseña que deberá proporcionar para su ingreso al sistema.

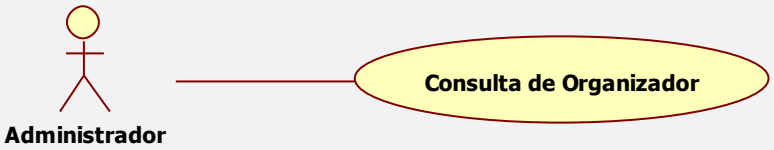
Flujo Normal del Evento

Actor		Sistema		Excepción
Paso	Acción	Paso	Acción	
1	El administrador selecciona el organizador a dar de baja	2	El sistema actualiza el estado del organizador en la base de datos	E1
		3	El sistema redirecciona a la página principal	

Flujo Excepcional de Eventos

Excepción	Descripción	Acción
E1	Error al escribir en la BD	Notificación al administrador

Caso de uso:	Consulta de organizador
Actor:	Administrador



Descripción	El administrador consulta la lista de administradores registrados en el sistema.
Precondiciones	1. El administrador debe acceder al sistema por medio de un navegador de páginas Web 2. El administrador debe contar con un usuario y contraseña que deberá proporcionar para su ingreso al sistema.

Flujo Normal del Evento

Actor		Sistema		Excepción
Paso	Acción	Paso	Acción	
1	El administrador elige los parámetros de búsqueda	2	El sistema realiza la consulta en la BD	E1
		3	El sistema despliega la información en pantalla	

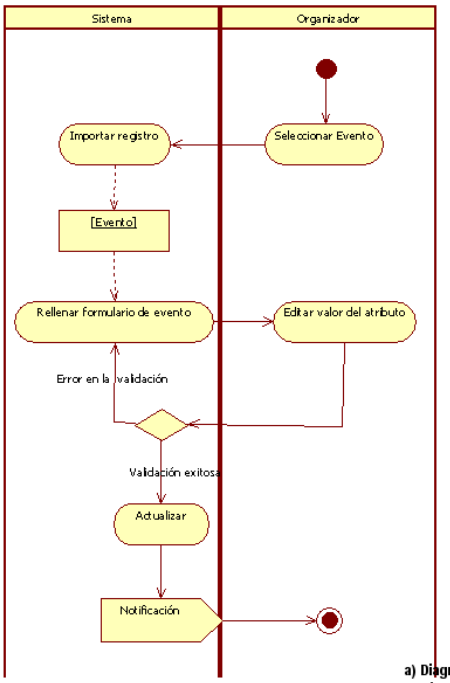
Flujo Excepcional de Eventos

Excepción	Descripción	Acción
E1	Error al escribir en la BD	Notificación al administrador

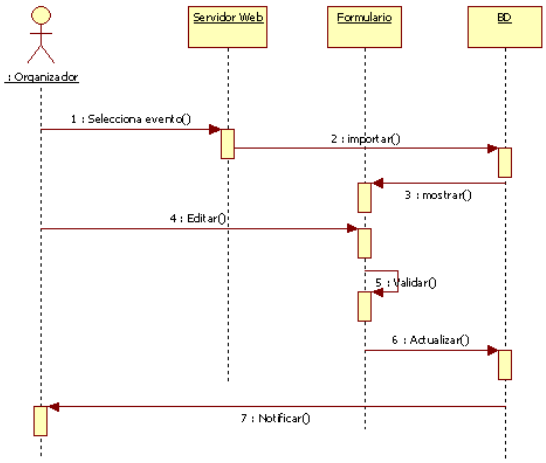
Anexo B: Diagramas de actividades y secuencia

Caso de uso: Actualizar Evento

a)



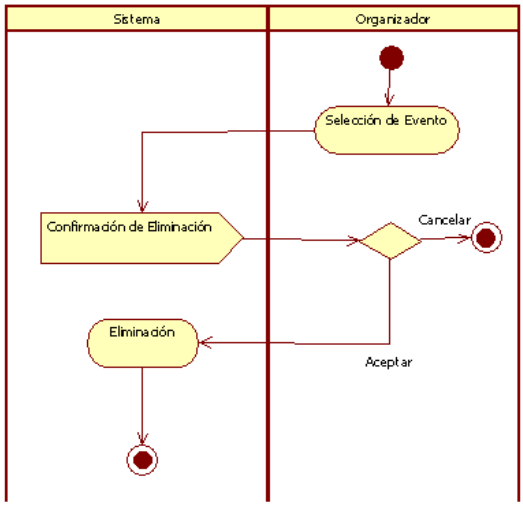
b)



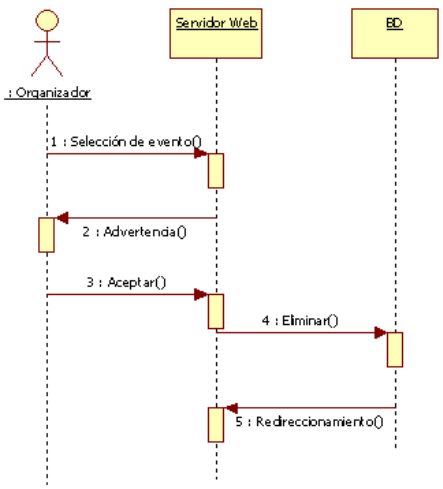
a) Diagrama de Actividades
b) Diagrama de Secuencia

Caso de uso: Eliminar Evento

a)



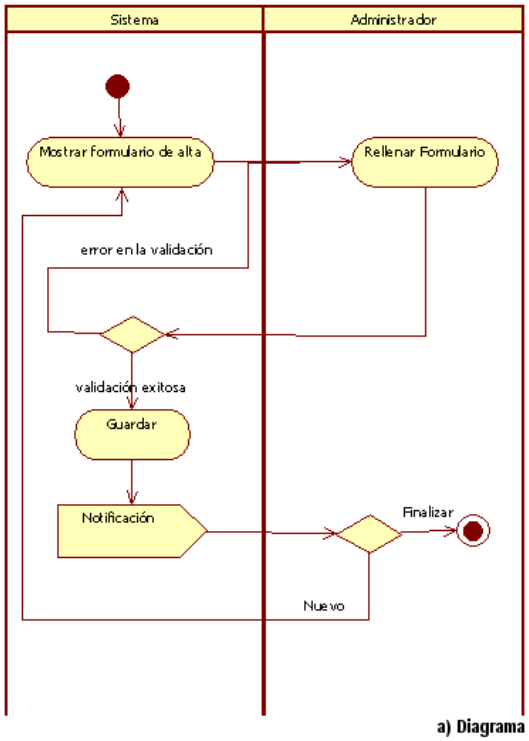
b)



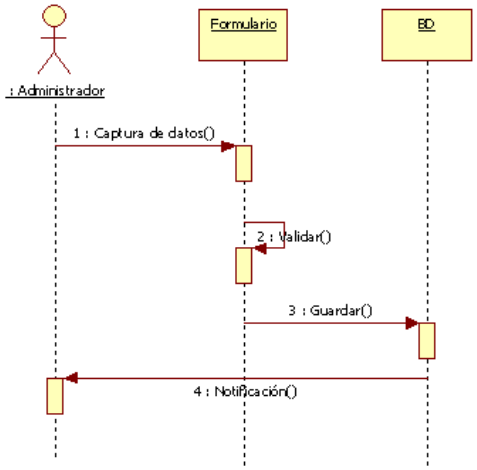
a) Diagrama de Actividades
b) Diagrama de Secuencia

Caso de uso: Alta de Organizado

a)



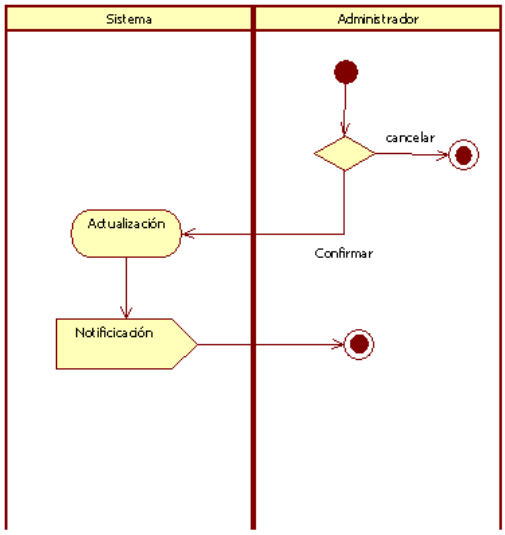
b)



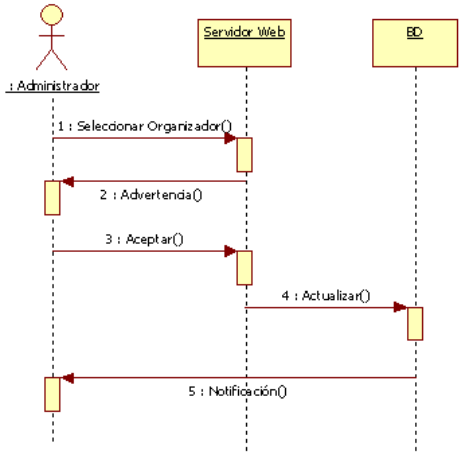
a) Diagrama de Actividades
b) Diagrama de Secuencia

Caso de uso: Baja de Organizador

a)



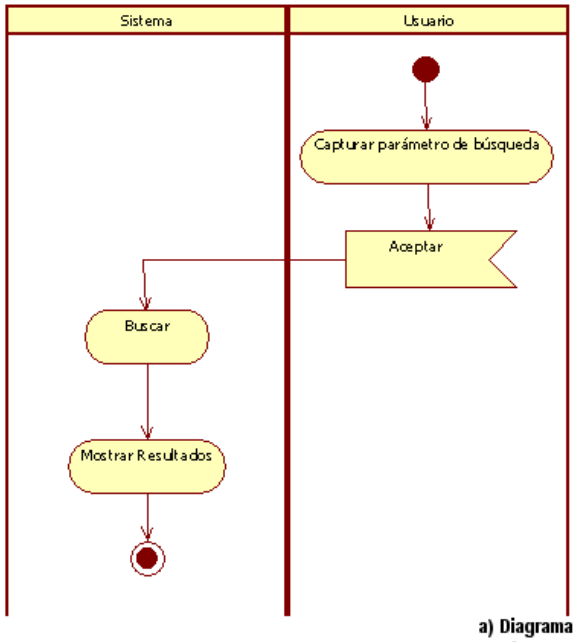
b)



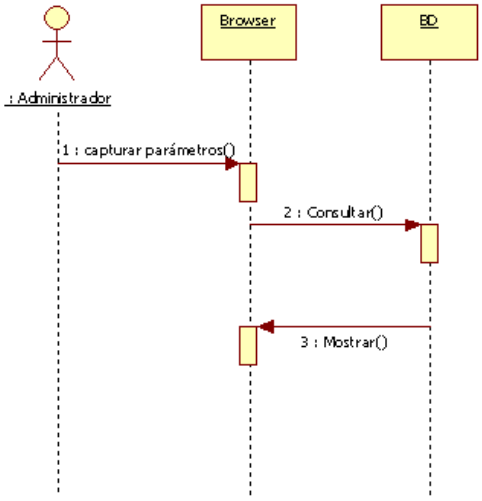
a) Diagrama de Actividades
b) Diagrama de Secuencia

Caso de uso: Consulta de Organizador

a)



b)



a) Diagrama de Actividades
b) Diagrama de Secuencia