
*UNIVERSIDAD COMPLUTENSE DE MADRID*

**Facultad de Informática**

**AUTORES:**
Esther Cocero Navarro
Jorge Díaz Bizarro
**PROFESOR DIRECTOR:**
Luis Javier García Villalba

Proyecto Sistemas Informáticos

# Estudio de aplicaciones de bluetooth para móviles de la serie S60 de Nokia

JUNIO 2007


*UNIVERSIDAD COMPLUTENSE DE MADRID*

**Facultad de Informática**

**AUTORES:**
Esther Cocero Navarro
Jorge Díaz Bizarro
**PROFESOR DIRECTOR:**
Luis Javier García Villalba

Proyecto Sistemas Informáticos

# Estudio de aplicaciones de bluetooth para móviles de la serie S60 de Nokia

JUNIO 2007

Los abajo firmantes, Esther Cocero Navarro y Jorge Díaz Bizarro, autores del proyecto *Estudio de aplicaciones Bluetooth para móviles de la serie S60 de Nokia* en la asignatura de Sistemas Informáticos, autorizan a la Universidad Complutense de Madrid a difundir y a utilizar con fines exclusivamente académicos, nunca comerciales, y mencionando expresamente a sus autores, los contenidos de este documento, así como el código, prototipos o documentación asociada a dicho proyecto.

Esther Cocero Navarro Jorge Díaz Bizarro

..... .....

Madrid, a 2 de julio de 2007

# ÍNDICE DE CONTENIDOS

# 1. RESUMEN

## 1.1. RESUMEN

El objetivo de este proyecto es el estudio y desarrollo de las aplicaciones bluetooth para móviles. Nos hemos centrado en móviles que utilizan la tecnología Symbian OS, en particular para la serie S60 de Nokia.

Este proyecto ha tenido una gran carga de investigación y formación debido al a que se trata de una tecnología relativamente nueva, muy cambiante y que actualmente no existe mucha información y gente experta en el tema, como puede ocurrir con otras tecnologías.

Las dos aplicaciones principales en las que nos hemos centrado son un chat bluetooth y un sniffer para bluetooth.

**PALABRAS CLAVE:** Symbian, Nokia, programación telefonía móvil, bluetooth, sniffer, chat

## 1.2. ABSTRACT

The aim of our proyect is research and development bluetooth mobile applications. We have concentrated in mobile phones with Symbian Operating System, specially Nokia Series 60 smartphones.

The main part of this proyect has been research and training because we have use a high technology which changes a lot and there are neither many information about it nor many experts.

We have worked in two software applications: a bluetooth Chat and a bluetooth sniffer.

**KEY WORDS:** Symbian, Nokia, mobile programming, bluetooth, sniffer, chat

# 2. INTRODUCCIÓN AL PROYECTO

## 2.1. INTRODUCCIÓN

El objetivo de este documento es el de mostrar el trabajo realizado en la elaboración de este proyecto, llevado a cabo dentro de la asignatura de *Sistemas Informáticos*, Estudio de aplicaciones de bluetooth para móviles de la serie 60 de Nokia.

De esta manera queremos mostrar cuales son los pasos a seguir a la hora de realizar un proyecto de características similares al nuestro: tecnologías necesarias, conocimientos previos, investigación, desarrollo de las aplicaciones… según lo hemos realizado nosotros.

Hoy en día el teléfono móvil se ha convertido en un instrumento casi indispensable para cualquier persona como herramienta de comunicación con su entorno: amigos, familia, negocios…

Al igual que ocurrió con los PCs en su momento, los teléfonos móviles han evolucionado a un ritmo acelerado, estando al alcance prácticamente de cualquiera, ofreciéndonos día tras día nuevos usos.

Atrás quedan aquellos tiempos en los que el teléfono móvil era un instrumento que servía simplemente para hablar. Después llegarían los sms, mms, correo electrónico, juegos, cámaras de fotos… convirtiendo el teléfono móvil prácticamente en un ordenador personal. Nuestro teléfono móvil dice mucho de nosotros; se podría decir que hay un móvil personalizado para cada uno: forma, color, tamaño, características, aplicaciones…

**Figura 1:diferentes modelos de teléfono**

Una de las últimas y principales aportaciones que nos ha ofrecido el mundo de la telefonía móvil es la incorporación a los mismos de la tecnología bluetooth. Esta tecnología nos permite comunicarnos con otros dispositivos (incluso con otros dispositivos que no son teléfonos), intercambiar archivos, ordenar tareas, y todo ello sin olvidarnos que esta comunicación carece de coste, no como ocurre, por ejemplo, con los sms. Esta última es una de las características que han hecho que el bluetooth en los teléfonos móviles haya tenido una gran acogida entre los usuarios.

Por esto creemos que es importante el estudio del desarrollo de aplicaciones bluetooth para móviles, cuáles son los pasos a seguir, que conocimientos se deben tener, como probar las aplicaciones diseñadas, como implantarlas en el teléfono… así como los posibles problemas que nos podemos encontrar según avance el desarrollo.

## 2.2. OBJETIVOS

Una vez que teníamos claro que es lo que queríamos desarrollar y la funcionalidad que debía tener, nos propusimos una serie de objetivos a cumplir con la realización de este proyecto, objetivos que creemos que son necesarios marcarse siempre que se quiera realizar un desarrollo de estas características:

- Estudio del sistema operativo Symbian. Las características que tiene y las posibilidades que nos ofrece. Estudiar cuáles son los fundamentos básicos para crear cualquier aplicación bluetooth.

- Presentación e introducción al funcionamiento del protocolo de comunicaciones Bluetooth, centrándonos especialmente la estructura y el modo de operar en los dispositivos móviles.

- Estudio del funcionamiento de las herramientas que están a nuestro alcance. Un buen uso de estas herramientas nos va a proporcionar una buena base para la creación de aplicaciones de cierta robustez, además de la comodidad que suponen pues nos permite probar y simular el funcionamiento de nuestros desarrollos sin necesidad de exportarlo a dispositivo físico.

- Otro de los objetivos que nos marcamos, y que consideramos de los más importantes, llevar a la práctica los conocimientos que hemos ido adquiriendo; para ello realizaremos una serie de aplicaciones bluetooth.

- Finalmente, una vez que hemos desarrollado una serie de aplicaciones, realizaremos las pruebas oportunas y depuraremos los posibles fallos que hayamos encontrado. Cuando estemos seguros de su robustez mediante simulación en las herramientas que hemos estudiado con anterioridad, trasladaremos la aplicación

a un dispositivo móvil y, de esta manera, probar de manera real si funciona la aplicación.

## 2.3. ESTRUCTURA DE LA MEMORIA

El trabajo que hemos ido realizando a lo largo de estos meses queda documentado en esta memoria. Dicha memoria se encuentra divida en los siguientes apartados:

- **APARTADO 1. Resumen:** En este apartado incluimos un pequeño resumen del proyecto que hemos llevado a cabo, presentado tanto en castellano como en inglés.

- **APARTADO 2. Introducción al proyecto:** Este es el apartado en el que nos encontramos. En este apartado exponemos una introducción del trabajo que hemos realizado a lo largo de estos meses.
  También presentamos cuáles son los objetivos que queremos alcanzar, la estructura de esta memoria (para una mejor comprensión de ésta), y las distintas fases en las que hemos divido nuestro proyecto.

- **APARTADO 3. Symbian:** En este apartado realizamos una introducción al sistema operativo de Symbian. En él, comentaremos son los pilares en los que se basa, cuáles son sus principales características, sus ventajas, las diferentes versiones existentes, fines para las que fueron creadas esta versiones, las mejoras que presentan unas y otras… de manera que podamos dar una visión de cuáles son las que más nos convienen utilizar dependiendo de los objetivos que queramos alcanzar.

- **APARTADO 4. Bluetooth:** Comentaremos una pequeña introducción a la tecnología bluetooth, cuáles son sus características principales.

- **APARTADO 5. Herramientas utilizadas y entorno de desarrollo:** Para poder realizar el desarrollo de cualquier aplicación Symbian que posteriormente funcione de manera correcta al exportarlo a un dispositivo móvil, es necesario el conocimiento de las distintas herramientas de las que contamos, su funcionamiento, requisitos, instalación…

- **APARTADO 6. Chat Bluetooth:** Para poner en práctica los conocimientos que hemos ido adquiriendo, hemos realizado la implementación de un bluechat para móviles de la serie S60 2ª edición de Nokia.
  Un bluechat es una charla o chat entre dos o más usuarios, donde cada uno utiliza un dispositivo bluetooth, en este caso un teléfono móvil, y lo nombra con lo que será su alias.

- **APARTADO 7. Funcionamiento de aplicaciones en dispositivos reales:** El objetivo de la realización de cualquier aplicación destinada a un teléfono móvil es que dicha aplicación pueda ejecutarse en un dispositivo real. Por ello, en este apartado explicamos como exportar a un dispositivo físico la palicación desarrollada.

- **APARTADO 8. Introducción a la tecnología Sniffer:** La idea inicial que teníamos cuando nos planteamos realizar este proyecto era, una vez terminado el estudio de cómo desarrollar aplicaciones Symbian con bluetooth, era construir un sniffer bluetooth para móvil. Sin embargo, debido a una serie de dificultades (que comentaremos en este apartado) que nos fuimos encontrando según íbamos avanzando el proyecto, tuvimos que apartar a un lado esta idea y orientarnos al desarrollo de otro tipo de aplicaciones.
  En este apartado, mostraremos cuáles fueron estas dificultades, y cuál era la idea que teníamos para desarrollar este sniffer.

- **APARTADO 9. Conclusiones:** En este apartado aportaremos una serie de conclusiones que extraído a nivel personal a lo largo de toda la elaboración del proyecto de Sistemas Informáticos. Además haremos una crítica acerca de los objetivos marcados para la realización del mismo, así como de los objetivos alcanzados y los diversos inconvenientes que nos han ido surgiendo a lo largo de estos meses.
  Así mismo, presentaremos una serie de posibles ampliaciones y líneas de trabajo futura, sobre esta base.

## 2.4. FASES DEL PROYECTO

Para realizar nuestro proyecto nos hemos organizado en una serie de fases, para seguir un desarrollo de una manera estructurada.

Estas fases se han ido organizando según los objetivos que nos íbamos marcando, de tal manera que eran los propios objetivos los que determinaban en que fase del proyecto nos encontrábamos:

- En la primera fase del proyecto nos dedicamos a recoger información sobre el sistema operativo Symbian, y a su estudio. Para nosotros OS Symbian era un sistema operativo desconocido. Sabíamos su utilidad, marcas comerciales que suelen utilizarlo… pero desconocíamos por completo cuáles los fundamentos de este sistema.
  Durante esta fase, estudiamos la arquitectura del sistema operativo, los pasos básicos a seguir en cualquier desarrollo,

lenguaje de programación… incluso asistimos a un curso ofertado por Nokia para el desarrollo de aplicaciones de la serie S60 Nokia.
También investigamos cuáles son las características que debe de tener un dispositivo móvil que quiera soportar aplicaciones Symbian, y el proceso de exportación de la aplicación desde el PC hasta el teléfono.

- El siguiente paso fue un estudio de la tecnología bluetooth, cuál es su estructura y funcionamiento, principalmente orientado a dispositivos móviles. En esta fase podemos decir que tuvimos dos subfases. Debíamos realizar un estudio de la tecnología bluetooth dependiendo del tipo de aplicaciones que fuésemos a desarrollar, esto es, no es el mismo estudio el que debemos realizar si queremos por ejemplo desarrollar un sniffer bluetooth, en el que tendremos que tener muy claro por ejemplo que tipo de trama existen (por lo que debemos bajar a un nivel de abstracción muy bajo) a por ejemplo una aplicación cuyo objetivo es comunicar a dos individuos mediante sus smartphones; en este último caso el nivel de abstracción es mucho mayor, no tenemos que bajar hasta un nivel de tramas como en el caso anterior, y nos podemos ayudar de las rutinas definidas por defecto por Symbian.

- Una vez que teníamos realizado un estudio profundo de las tecnologías que íbamos a usar en nuestro proyecto, pasamos a una fase en la que nos dedicamos a estudiar distintas herramientas para implementar y desarrollar aplicaciones.
  La mayoría de estas herramientas son herramientas desarrolladas por Nokia. La principal ventaja que tiene el uso de estas herramientas es que nos permite simular el comportamiento de nuestra aplicación en el pc sin necesidad de instalarlo en el teléfono, con lo que nos evitamos tener resultados inesperados en el dispositivo por haber instalado alguna aplicación con fallos.
  Entre estas herramientas se encuentra el *Carbide* (basado en la plataforma de desarrollo Eclipse) y el *Nokia Conectivity Framework* (que nos permite simular la comunicación entre varios dispositivos bluetooth que tuviesen instalada la aplicación que hayamos desarrollado).

- A continuación ya estábamos preparados para el desarrollo de aplicaciones. La primera idea que teníamos era la de desarrollar un sniffer bluetooth, pero tras realizar un estudio exhaustivo, tuvimos decantarnos por otro tipo de aplicación, debido a una serie de dificultades que comentaremos más adelante. Finalmente la aplicación que desarrollamos fue la de un bluechat, que utilizaba un nivel de abstracción distinto al del sniffer. Para este desarrollo nos servimos de las herramientas estudiadas en la fase anterior.

- Una vez realizada toda la implementación, pasamos a una fase de simulación en el pc, prueba y depuración de fallos que nos habíamos encontrado.

- Para terminar con la parte práctica, trasladamos la aplicación a un dispositivo físico, y las correspondientes pruebas sobre el mismo.

- Finalizamos este proyecto llevado a cabo a lo largo de unos meses con la elaboración de esta memoria, que muestra el trabajo realizado para ello.

# 3. SYMBIAN

## 3.1. INTRODUCCIÓN

Symbian es un sistema operativo que lleva cierto tiempo en el mercado y que ha sido diseñado específicamente para dispositivos móviles.

Según los últimos datos, la plataforma Symbian está presente en más de la mitad de los terminales inteligentes vendidos en el mundo durante el último año 2006. La cifra de terminales vendidos el pasado año es de unos 64 millones.

Symbian es producto de la alianza de varias empresas de telefonía móvil, entre las que se encuentran Nokia, Sony Ericsson, Samsung, Siemens, Panasonic… Fue creado con el objetivo de competir con el sistema operativo de Palm o Smartphone de Microsoft.

Inicialmente Psion desarrolló un sistema operativo de 32 bits denominado EPOC32, sin embargo, como resultado en la búsqueda de estándares de software para dispositivos móviles se funda Symbian en 1998, basándose en su antecesor EPOC32. Inicialmente fue desarrollado por Psion, Nokia y Ericsson. Posteriormente se añadirían otros miembros como Panasonic, Siemens y Sony Ericsson. Es en el año 2000 cuando se comercializa el primer teléfono basado en Symbian, el Ericsson R380, con la versión 6 del sistema operativo. A partir de este aparecerían nuevas versiones, la mayor parte de ellas creadas por Nokia.

Symbian es un software bajo licencia. Por tanto se debe distinguir entre las compañías que desarrollan Symbian y las compañías con licencia para su uso. Entre estas últimas podriamos encontrar Motorola, Sanyo, Benq, Lenovo, Sharp, Sendo, Fujitsu…

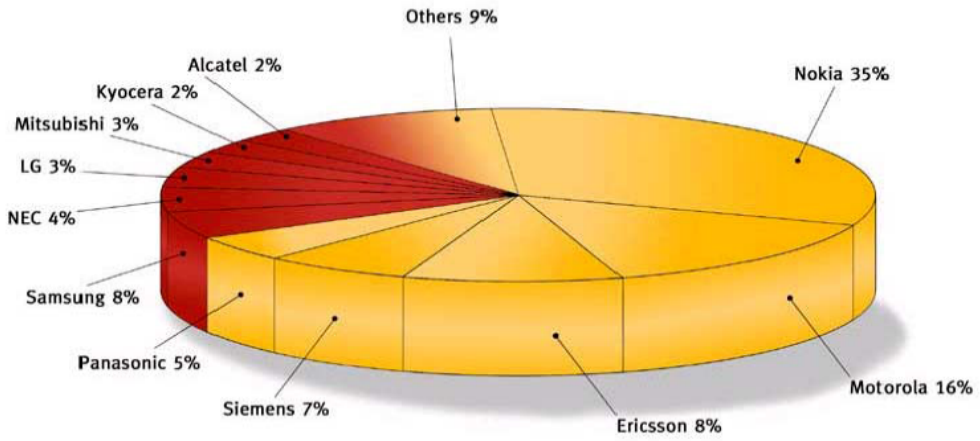


**Figura 2: Reparto licencias Symbian en el mercado móvil**

Actualmente se encuentra en su versión 9.1 3ª edición (ya se está preparando la 9.2) y existen dos grandes familias: por un lado las series S (40, 60, 80) de Nokia y por otro lado la plataforma UIQ utilizada por otras compañías como Sony Ericsson. Estas dos familias se diferencian sobre todo en la interfaz gráfica usada.

## 3.2. FUNDAMENTOS DE SYMBIAN

El SO de Symbian proporciona las rutinas y los servicios subyacentes para las aplicaciones. Por ejemplo, un software de email que interactúa recíprocamente con un usuario a través de la pantalla del teléfono móvil y descarga mensajes del email al inbox del teléfono a través de una red móvil o un acceso WiFi, está utilizando las rutinas de los protocolos de comunicación y control de archivos proporcionadas por el SO de Symbian.

La tecnología del SO de Symbian se ha diseñado teniendo en cuenta los siguientes puntos clave:

- Proporcionar la energía, memoria y gestión de entrada y salida de recursos requeridos específicamente en los dispositivos móviles.
- Entregar una plataforma abierta que se conforma con telecomunicaciones y estándares globales de Internet.
- Proporcionar herramientas para desarrollar software móvil para empresas, medios y otros usos.
- Asegurar una amplia disponibilidad de aplicaciones y accesorios para diversas exigencias del consumidor.
- Facilitar la conectividad inalámbrica para una variedad de redes.

Los principales beneficios que aporta el sistema OS Symbian son:

- Selección amplia de las aplicaciones disponibles para una gama de teléfonos móviles.
- Implementa los protocolos standard de la industria, los interfaces y la gestión de servicios para la integración de los sistemas de IT.
- Desarrollo del aplicaciones utilizando los lenguajes standard de la industria Java y C++.
- Extensas opciones de conectividad - incluyendo GSM, GPRS, CDMA, WCDMA, WiFi y Bluetooth.

# 3.3. PLATAFORMAS SYMBIAN

La plataforma Symbian ha ido evolucionando como se muestra en la tabla siguiente:

| | Series 60 | Series 80 | UIQ | Others |
|---|---|---|---|---|
| v6.0 | | Nokia 9210 | | |
| v6.1 | v0.9 / v1.2<br>Nokia 7650 | | | |
| v7.0 | | | Sony Ericsson P800/P900 | |
| v7.0s | 2nd edition<br>Nokia 6600 | Nokia 9500 | | Nokia7710 |
| v8.0 | 2nd edition v2.6<br>Nokia 6630 | | | |
| v9.1 | 3rd edition | | | |

**Figura 3**

- **Symbian OS v6.0**: también conocido como GT 6.0, fue la tecnología más usada en los primeros S80.
- **Symbian OS v6.1s**: también conocido como GT 6.1. actualmente es usada la versión v0.9 y v1.x en los dispositivos S60. Introdujo el soporte para GPRS, MMS y bluetooth sobre la versión anterior Symbian OS v6.0.
- **Symbian OS 7.0**: también conocido como GT 7.0. Es actualmente usada por los teléfonos Sony P800. Añadió la tecnología 3G, aunque este modelo de Sony no lo llevaba.
- **Symbian OS 7.0s**: Esta es usada por los móviles de la serie S60 2ª edición. Su principal aporte fue las ventanas multimedia.
- **Symbian OS 8.x**: La diferencia con respecto al anterior es que incorpora métodos de IPC, nuevas extensiones multimedia, y nuevas APIs. .
- **Symbian OS 9.x**: Actualmente son teléfonos que se encuentran en desarrollo. Mejora el soporte multimedia con respecto a sus antecesores, servicios network, mejores diseños gráficos y nuevas funciones para empresas, pues este tipo de teléfonos están más orientados al uso en empresas.

El despiece de cualquiera de los GT que hemos podría ser el siguiente:

- Un **20%** del total sería dedicado al diseño de **la interfaz gráfica**.
- **Kernel (5%):** el kernel del sistema operativo no sería accesible a los programadores
- **Symbian System Layer ( 55%)**: contiene el gran conjunto de APIs ofrecidas por Symbian, asi como la funcionalidad de cualquier evento que pueda ocurrir.
- **Symbian applications engines (20%):** permite crear funcionalidad como pudiera ser la lista de contactos o el calendario.

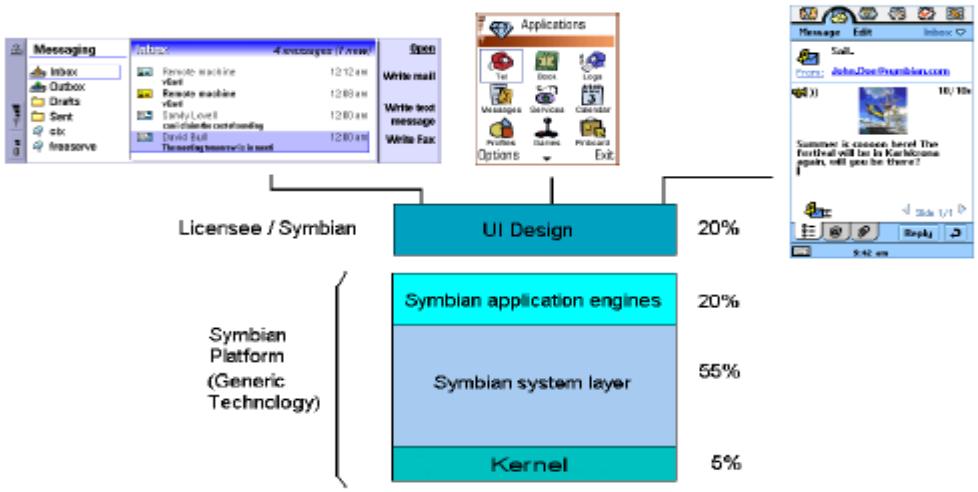


**Figura 4: despiece del GT**

Como hemos comentado anteriormente existen dos grandes familias en Symbian por un lado las series S (40, 60, 80) de Nokia y por otro lado la plataforma UIQ utilizada por otras compañías como Sony Ericsson. Estas dos familias se diferencian sobre todo en la interfaz gráfica usada. Nosotros nos vamos a centrar en las de la familia de Nokia.


**Figura 5: familias y móviles**

Los teléfonos móviles que usan la familia UIQ son teléfonos que suelen tener pantallas bastante grandes, con pantallas táctiles y teléfonos que muestran pantalla una gran cantidad de información. Está diseñado para que en centro de la pantalla se muestren todos los cambios, en la parte superior se va a mostrar una serie de opciones, y va a ser en centro de la pantalla donde se muestre toda la información. Una característica importante es que cualquier cambio se guarda de manera automática y no hay que esperar a una posible confirmación.

Una pantalla de UIQ, con una resolución de 208 x 320 presenta las siguientes zonas:

- **Application picker**: contiene los iconos que permite al usuario seleccionar las Aplicaciones. La aplicación elegida aparece en foreground.
- **Application space**: que corresponde con el área central de la pantalla.
- **Menu bar**: normalmente contiene dos menús a la derecha y a la izquierda.
- **Status bar**: se trata de un display de información acerca de la carga de batería, intensidad de señal…

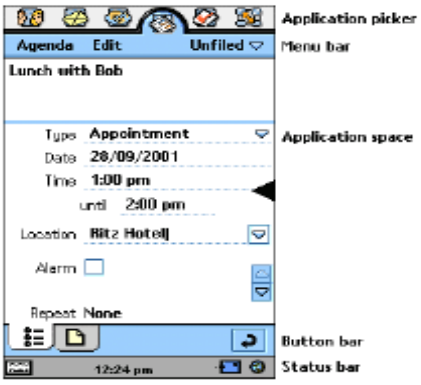


**Figura 6: pantalla UIQ**

En comparación con la S60 y S80 de Nokia (que estudiamos más adelante) se diferencia en que los móviles que tienen estas tecnologías estan diseñados para ser usados con una sola mano, como normalmente estamos habituados a hacerlo. Las pantallas presentan un menú bastante más sencillo, con una serie de iconos en el centro de la pantalla, por los cuales nos vamos a poder mover utilizando las flechas de la botonera del dispositivo físico.

Cualquier la ventana de cualquier aplicación que estén desarrolladas para estas plataformas va a estar divida en las siguientes partes:

- **Main pain**: es el área principal de la pantalla donde una aplicación puede mostrar sus datos.

- **Control pane**: muestra las dos etiquetas asociadas a las dos "softkeys".
- **Status pane**: muestra información acerca de la siguiente aplicación y estado, así como información general del estado del teléfono: batería, intensidad de señal...

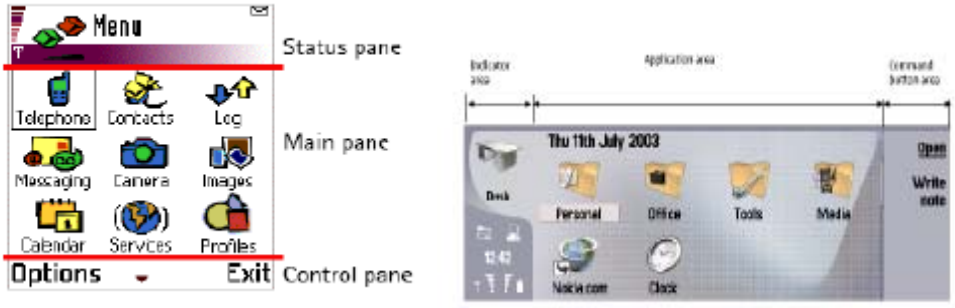


**Figura 7: pantalla S60 y S80**

En cuanto a lo que pantalla se refiere la principal diferencia es la orientación, pues para la serie S80 suele ser apaisada como mostraba la imagen anterior (son móviles más sofisticados, e intentan dar una imagen de pantalla de pc), y la resolución de la pantalla: en S60 es de 176 x 208 pixeles y en un S80 640 x 200.

Dentro de la plataforma Nokia encontramos tres grandes plataformas de desarrollo, cuya diferencia se basa en el uso para el que están diseñados:

- **Plataforma de desarrollo Serie 40**: La plataforma de desarrollo Serie 40 ofrece una oportunidad de mercado masivo a los desarrolladores de Java™.
- **Plataforma de desarrollo Serie 60**: La plataforma de smartphones compartida por marcas de teléfonos de todo el mundo.
- **Plataforma de desarrollo Serie 80**: Esta plataforma de comunicadores está orientada a uso profesional.

Nuestro proyecto va a estar dirigido hacia aplicaciones de la serie S60, por lo que estudiaremos con más calma esta plataforma.

### 3.3.1.NOKIA SERIE S40

Los teléfonos de la serie 60 implementan varias interfaces. La más común de ellas es la familia que tiene 12 bit (4096) de color con una pantalla de 128 x 128 pixels de resolución. También pueden incluir una resolución de 128 x 160 y 96 x 65.

Para interactuar con el teléfono se usa el teclado estándar, dos o cuatro botones de navegación.

Para los desarrolladores la serie S40 viene provista para trabajar con:

- Java2.
- XHTML Mobile Profile
- Integración MMS.
- DMR


**Figura 8: evolución serie 40**

## 3.3.2.NOKIA SERIE S80-S90

Están construidos sobre Symbian OS v7.0s.

Permite la instalación y la ejecución de de aplicaciones Java.

Soporta las tecnologías

- Java 2, J2ME
- XHTML
- HTML
- JavaScrip
- CSS, WAP CSS,
- SMS, MMS, e-mail, IMAP-4, conexión al PC por usb

- Aplicaciones que pueden soportar archivos de audio ( mp3,aac, amr, midi…), video (mpeg4…), tonos, politonos..

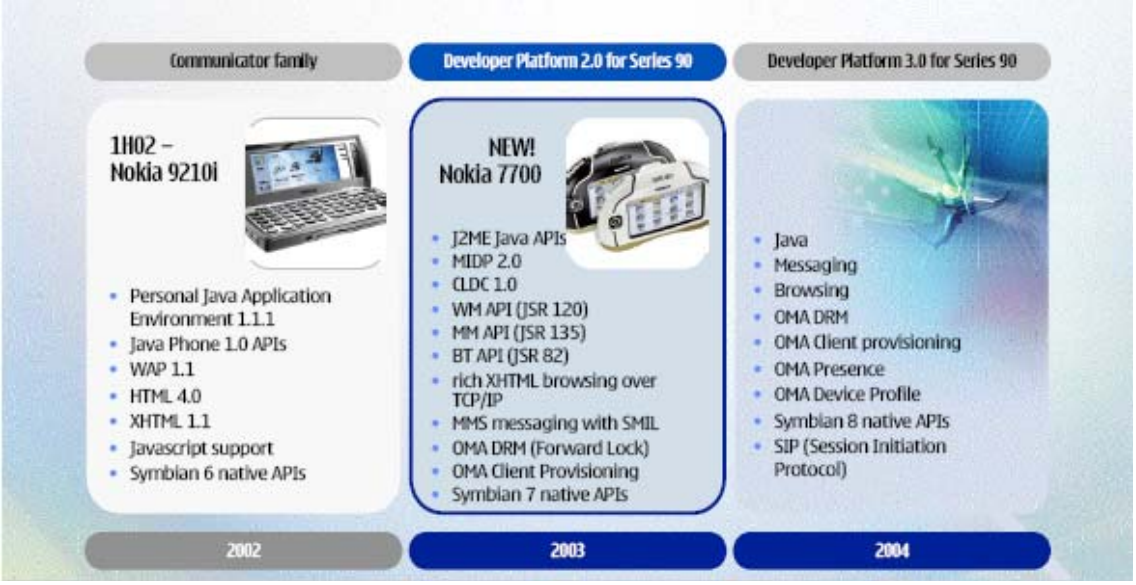


**Figura 9: evolución serie 80**

## 3.3.3.**NOKIA SERIE S60**

La plataforma Series S60 de Symbian ofrece un estándar abierto y soporte multi-fabricante para el mercado de teléfonos de última generación. La tercera edición de la plataforma S60 cuenta con soporte de las APIs del sistema operativo Symbian v9.1 y Java, lo que añade nuevos patrones en funcionalidad y seguridad.

S60 consiste en un conjunto de librerías o dll y de usos estándares, tales como telefonía, herramientas permitidas en equipos multimedia, teléfonos con sistema operativo mitad pda y mitad teléfonos, que se conocen comúnmente como smartphones.

El software S60 es un estándar multiplataforma para los smartphones que se desarrolló en base a Java MIDP, C++, y Pitón. Una característica importante de los teléfonos S60 es que permiten que los nuevos usos como programas o aplicaciones sean instalados después de compra. Éstas son algunas características comunes en S60:

- La resolución de la pantalla de los dispositivos es originalmente 176x208 pixeles. Puesto que el 2do paquete 3, S60 de la característica de la tercera edición apoya resoluciones múltiples, es decir básico (176x208), QVGA (240x320) y el doble (352x416). Nokia N90 es el primer dispositivo S60 que tiene una resolución más alta (352x416). Algunos dispositivos, sin embargo, tienen resoluciones no estándar, como el Siemens SX1, con 176x220. Nokia 5500 tiene una resolución de pantalla 208x208.

- soporta Java (J2ME/MIDP 1.0) y el lenguaje C++. Se emplea para ser fácil y rápida utilizar. Proporciona el marco versátil y de gran alcance del uso.
- Es significativo que software escrito para 1r la edición S60 (S60v1) o la 2da edición (S60v2) no es compatible binariamente hablando la edición S60 (S60v3), porque utiliza una versión mejorada del OS de Symbian (v9.1).

Con la tercera edición, los desarrolladores tienen acceso a un amplio conjunto de funciones, como: las extensiones, denominadas Symbian OS Extensions, que son un conjunto de capacidades que permiten a la plataforma S60 interactuar con las funciones de la circuitería del dispositivo, como los indicadores luminosos del terminal, alertas de vibración, y estado de carga de la batería.

Los servicios de la plataforma S60 son el núcleo fundamental para las aplicaciones y a su vez están compuestos por:

- Application Framework Services: suministra las habilidades básicas para lanzar aplicaciones y servidores, gestión del estado de persistencia y componentes de UI.
- UI Framework Services: proporciona el aspecto y comportamiento concreto para los componentes UI y maneja los eventos UI.
- Servicios gráficos: ofrece las capacidades de creación de gráficos y su dibujo sobre la pantalla del terminal.
- Servicios de localización: permite a la plataforma tomar en cuenta la localización del dispositivo.
- Servicios basados en Web: proporciona los servicios para establecer conexiones e interactuar con la funcionalidad basada en Web, incluyendo navegación, descarga de archivos y mensajería.
- Servicios multimedia: aporta las capacidades para reproducir audio y vídeo, así como soporte para streaming y reconocimiento de voz.
- Servicios de comunicación: lleva el soporte de las comunicaciones, tanto locales como de banda ancha, lo que incluye desde tecnologías Bluetooth hasta las llamadas de voz.
- Servicios de aplicación S60: un conjunto de habilidades que son empleadas por las aplicaciones de S60 y que pueden ser empleadas por los desarrolladores de terceras partes para suministrar funcionalidades básicas para las aplicaciones. Esto incluye Servicios de aplicaciones PIM, Servicios de aplicación de mensajería y Servicios de aplicación del navegador.
- Servicios de tecnología Java, que soporta la plataforma Java 2, Micro Edition (J2ME), así como la especificación JSR-185, Java Technology for the Wireless Industry (JTWI). La plataforma soporta la configuración JSR-139, Connected Limited Device Configuration (CLDC) 1.1 y la extensión Mobile Information Device Profile (MIDP)

2.0 (JSR-118). Además, también están soportadas otras APIs adicionales.

- Aplicaciones S60, todo un conjunto de aplicaciones disponibles para el usuario, que incluye gestor de información personal, PIM (personal information manager), mensajería, aplicaciones multimedia, perfiles, etc.
- La plataforma S60 define un estilo UI y sus APIs, pero no obliga a un tamaño de pantalla o a un determinado método de entrada. Los licenciados son totalmente libres de implementar sus propios UI personalizados. Los desarrolladores deben programar las aplicaciones UI con la escalabilidad en mente, ya que no hay que asumir unas dimensiones específicas.


**Figura 10: evolución S60**

# 3.4. PRIMEROS PASOS CON SYMBIAN

El primer paso a la hora de realizar cualquier desarrollo en Symbian es la elección del entorno de desarrollo. La elección de un entorno u otro es importante, pues no se puede editar directamente una aplicación en un entorno de desarrollo si ha sido realizada en otro entorno distinto, por ejemplo, la organización que hemos usado con el Carbide.c++ (que es la herramienta que hemos usado para desarrollar nuestro proyecto) no se puede editar directamente con otra herramienta como puede ser el Codewarrior.

Existe una serie de herramientas, que son proporcionadas con el SDK que estemos usando en ese momento, que nos van a facilitar la programación y el desarrollo de aplicaciones, que facilitan el proceso de compilación y enlazado. También podemos hacer uso de la línea de comandos.

### 3.4.1.EMULADOR EPOC

El emulador EPOC es una herramienta que nos permite simular el comportamiento de un teléfono móvil en un PC trabajando con Windows. Este emulador suele venir incorporado normalmente en el SDK de la plataforma de Symbian que estemos usando.

Existen varios modelos de emuladores, cada uno correspondiente a una versión de Symbian determinada (diseñado para un modelo de móvil determinado). Nosotros, como nuestras aplicaciones estaban dirigidas a teléfonos de la serie S60 de Nokia segunda edición, utilizamos el emulador correspondiente a esta versión.


**Figura 11: emuladores S60 (diferentes versiones)**

Como se puede observar en las figuras anteriores, los emuladores nos presentan una interfaz gráfica que simula el aspecto que tiene un teléfono móvil, dependiendo de la versión, aparecerá un mayor o menor número de botones en el teclado del teléfono.

Una vez instalado el emulador de Symbian que vamos a utilizar se configura con las correspondientes variables de entorno en Windows para ejecutarlo. En el caso que tengamos varias versiones de emulador distintas, estas

variables de entorno se harán referencia al último q se haya instalado. Si desde la línea de comandos tecleamos Consola > **epoc** se lanzará el emulador. Una vez que tenemos el emulador arrancado, podemos usarlo como si se tratase de un teléfono móvil (salvo la excepción que no se pueden realizar llamadas desde él), como son movernos por el menú de pantalla, ejecutar las aplicaciones, cambiar el aspecto del móvil, la resolución de la pantalla… como si se tratase de un dispositivo físico, con la ventaja que ello nos aporta: probar nuestra aplicación en el pc sin necesidad de migrarla al teléfono físico.

Por defecto el emulador se instala en el siguiente directorio: C:\Symbian\7.0s\S60_2nd. Dentro de este directorio nos encontramos con los siguientes subdirectorios:

- \S60Doc: documentación del emulador.
- \Examples: ejemplos de programas en Symbian.
- \S60Ex: ejemplos de la plataforma S60.
- \S60Tools: herramientas para Symbian.
- \Epoc32: directorio del emulador propiamente dicho.

Dentro del subdirectorio Epoc nos podemos encontrar los siguientes directorios:

- \Epoc32\release\winscw\udeb: librerias symbian.
- \Epoc32\winscw\c
- \Epoc32\winscw\d
- \Epoc32\release\winscw\udeb\z
  Los tres últimos directorios corresponden con el sistema de ficheros del móvil: *c:, d:* y *z:*. El sistema de ficheros es análogo al presentado por los teléfonos móviles y consta de dos discos en RAM (c y d) y un disco en ROM (z), de sólo lectura, en el que se almacena el kernel del sistema operativo de forma segura.

## 3.4.2. **LÍNEA DE COMANDOS**

Para compilar los proyectos que estemos realizando, hay dos maneras de hacerlo. Una es a través del la herramienta que utilicemos para la programación (en este caso el Carbide) o bien a través de la línea de comandos, con las herramientas que nos ofrece Symbian:

- **bldmake** realiza una compilación de los ficheros fuente y los prepara para su construcción. Genera un ejecutable abld.bat que se utilizará en el siguiente paso. Se ejecuta de la siguiente forma: Consola > **bldmake bldfiles**

- **abld** que realiza la construcción de la aplicación para una determinada plataforma; las más comunes son winscw, que es el emulador de Symbian, y armv5, que es el dispositivo móvil en sí.

No es lo mismo construir la aplicación para el emulador que para el móvil y se pueden encontrar problemas al migrar una aplicación de emulador a móvil y viceversa.
Consola > **abld build [winscw | armv5] [udeb | urel]**

**udeb** crea los binarios con información simbólica para realizar depuración mientras que **urel** los crea sin esta información de depuración para usarlos como versión de salida (binarios más pequeños).

Para usar correctamente estas herramientas, el proyecto debe ser organizado correctamente y deben existir unos archivos que le indiquen a las herramientas lo que deben hacer:

- Por un lado tenemos los ficheros **fuente**: ficheros de cabecera **.h** con sus correspondientes ficheros de implementación **.cpp**.

- Debe existir un fichero **.mmp** que incluya información tal que el nombre final del ejecutable, ficheros fuente a compilar, librerías incluidas y permisos de la aplicación

# 4. BLUETOOTH

## 4.1. INTRODUCCIÓN

Bluetooth es el nombre común de la especificación industrial IEEE 802.15, que define un estándar global de comunicación inalámbrica, que posibilita la transmisión de voz y de datos entre diferentes equipos mediante un enlace por radiofrecuencia segura, de corto rango. Los principales objetivos que se pretende conseguir con esta norma son:

- Facilitar las comunicaciones entre equipos móviles y fijos.
- Eliminar cables y conectores entre éstos.
- Ofrecer la posibilidad de crear pequeñas redes inalámbricas y facilitar la sincronización de datos entre equipos personales.

La tecnología Bluetooth está disponible en todo un abanico de dispositivos, desde teléfonos móviles hasta instrumental médico, pasando por automóviles, y abarca una gran variedad de usuarios, desde consumidores particulares hasta mercados industriales. Su bajo consumo de energía, reducido tamaño y el escaso coste de los chips permite emplear la tecnología Bluetooth hasta en los dispositivos más pequeños. Consulte el índice de productos Bluetooth y el listado de componentes para ver la amplia gama de opciones que le ofrecen nuestros miembros.

La tecnología Bluetooth comprende hardware, software y requerimientos de interoperabilidad, por lo que para su desarrollo ha sido necesaria la participación de los principales fabricantes de los sectores de las telecomunicaciones y la informática, tales como: Ericsson, Nokia, Toshiba, IBM, Intel y otros. Posteriormente se han ido incorporando muchas más compañías, y últimamente también se han incorporado empresas de sectores tan variados como: automatización industrial, maquinaria, ocio y entretenimiento, fabricantes de juguetes, electrodomésticos, etc., con lo que en poco tiempo se nos presentará un panorama de total conectividad de nuestros aparatos tanto en casa como en el trabajo.

El estándar Bluetooth es una tecnología ad hoc, lo que significa que no se necesita una infraestructura fija y es sencilla de instalar y configurar. La conexión se realiza sin cables. El proceso resulta muy sencillo para los nuevos usuarios: una vez adquirido el producto Bluetooth, basta con comprobar los perfiles disponibles y conectarlo a otro dispositivo Bluetooth con los mismos perfiles. A continuación, se debe introducir un código PIN, similar al que se emplea al sacar dinero en un cajero. El usuario lleva consigo en todo momento su red de área personal (PAN) e incluso puede conectarse a otras.

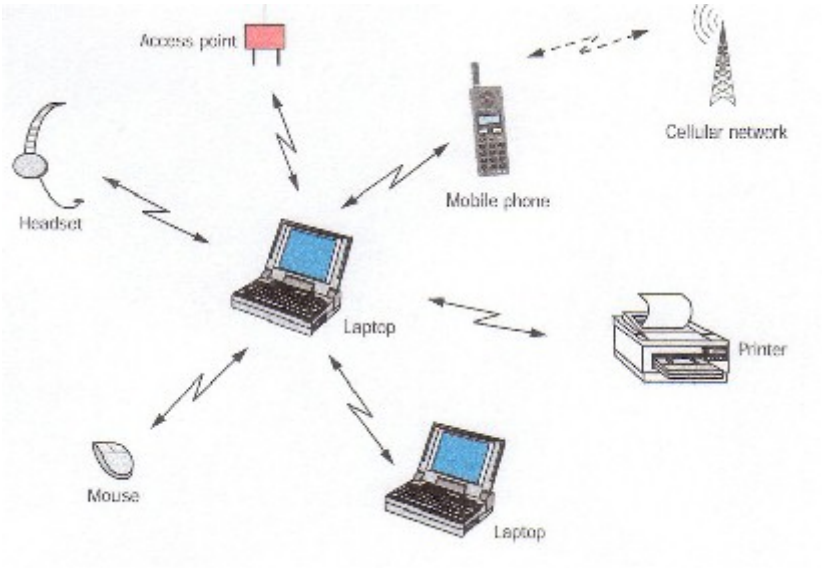


**Figura 12: usos bluetooth**

El nombre procede del rey danés y noruego Harald Blåtand cuya traducción al inglés sería Harold Bluetooth (Diente Azul, aunque en lengua danesa significa 'de tez oscura') conocido por buen comunicador y por unificar las tribus noruegas, suecas y danesas.

De la misma manera, Bluetooth intenta unir diferentes tecnologías como las de los ordenadores, los teléfonos móviles y el resto de periféricos.

En 1994 Ericsson inició un estudio para investigar la viabilidad de una interface vía radio, de bajo coste y bajo consumo, para la interconexión entre teléfonos móviles y otros accesorios con la intención de eliminar cables entre aparatos. El estudio partía de un largo proyecto que investigaba sobre unos multi-comunicadores conectados a una red celular. Se llegó a un enlace de radio de corto alcance llamado MC link. Conforme este proyecto avanzaba se fue viendo claro que este tipo de enlace podía ser utilizado ampliamente en un gran número de aplicaciones, ya que tenía como principal virtud el que se basaba en un chip de radio relativamente económico.

A comienzos de 1997, según avanzaba el proyecto MC link, Ericsson fue despertando el interés de otros fabricantes de equipos portátiles. Rápidamente se vio claramente que para que el sistema tuviera éxito, un gran número de equipos deberían estar equipados con esta tecnología. Esto fue lo que originó, a principios de 1998, la creación de un grupo de interés especial (SIG), formado por 5 promotores: Ericsson, Nokia, IBM, Toshiba e Intel. La idea era lograr un conjunto adecuado de áreas de negocio: dos líderes del mercado de las telecomunicaciones, dos líderes del mercado de los PCS portátiles y un líder de la fabricación de chips. El propósito principal del consorcio fue y es el establecer un estándar para la interface aérea junto con su software de control, con el fin de asegurar la interoperabilidad de los equipos entre los diversos fabricantes.

Existen varias versiones:

- Bluetooth v.1.1
- Bluetooth v.1.2
- Bluetooth v.2.0
- Bluetooth v.2.1

La versión 1.2, a diferencia de la 1.1, provee una solución inalámbrica complementaria para coexistir bluetooth y Wi-Fi en el espectro de los 2.4 GHz, sin interferencia entre ellos.

La versión 1.2 usa la técnica "Adaptive Frequency Hopping (AFH)", que ejecuta una transmisión más eficiente y un cifrado más seguro. Para mejorar las experiencias de los usuarios, la V1.2 ofrece una calidad de voz (Voice Quality - Enhanced Voice Processing) con menor ruido ambiental, y provee una más rápida configuración de la comunicación con los otros dispositivos bluetooth dentro del rango del alcance, como pueden ser PDAs, HIDs (Human Interface Devices), ordenadores portátiles, ordenadores de sobremesa, Headsets, impresoras y celulares.

La versión 2.0, creada para ser una especificación separada, principalmente incorpora la técnica "Enhanced Data Rate" (EDR) que le permite mejorar las velocidades de transmisión en hasta 3Mbps a la vez que intenta solucionar algunos errores de la especificación 1.2.

La version 2.1, simplifica los pasos para crear la conexión entre dispositivos, además el consumo de potencia es 5 veces menor.

## 4.2. FUNCIONAMIENTO BLUETOOTH

Los dispositivos Bluetooth operan a 2,4 Ghz (Concretamente de 2400 Mhz a 2483,5 Mhz) en la banda ISM de libre licencia y globalmente disponible. Esta banda está reservada para uso general de aplicaciones industriales, científicas y médicas (ISM-Industrial, cientific and Medical), y obedece un conjunto básico de especificaciones en cuanto a potencia, emisión de espectro e interferencias. Esto significa que el protocolo Bluetooth debe ser muy robusto, al trabajar en una frecuencia de gran uso.

La banda operativa está dividida en canales espaciados de 1MHz, cada uno enviado datos a 1 Megasímbolo por segundo con lo que se obtiene el máximo ancho de banda disponible. Con el esquema de modulación GFSK (Guassian Frequency Shift Keying) esto equivale a 1Mb/s. Usando GFSK, un “1” binario se traduce en una desviación positiva de la frecuencia respecto a

la frecuencia simbólica, mientras que un “0” binario se refleja como una desviación negativa.

La información que se intercambia entre dos unidades Bluetooth se realiza mediante un conjunto de slots que forman un paquete de datos. Cada paquete comienza con un código de acceso de 72 bits, que se deriva de la identidad maestra, seguido de un paquete de datos de cabecera de 54 bits. Éste contiene importante información de control, como tres bits de acceso de dirección, tipo de paquete, bits de control de flujo, bits para la retransmisión automática de la pregunta y chequeo de errores de campos de cabecera. Finalmente, el paquete que contiene la información, que puede seguir al de cabecera, tiene una longitud de 0 a 2745 bits. En cualquier caso, cada paquete que se intercambia en el canal está precedido por el código de acceso.

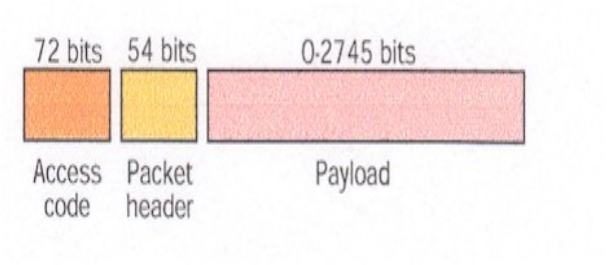


**Figura 13: trama bluetooth**

Los receptores del piconet comparan las señales que reciben con el código de acceso. Si éstas no coinciden, el paquete recibido no es considerado como válido en el canal y el resto de su contenido es ignorado.

Tras cada paquete, emisor y receptor vuelven a afinar la frecuencia a un canal diferente, mediante un salto de canal de radio en canal de radio determinado por la técnica de FHSS (Frecuency Hopping Spread Spectrum o Espectro Ensanchado por Salto de Frecuencia).

El espectro ensanchado por salto de frecuencia es una técnica de modulación en espectro ensanchado en el que la señal se emite sobre una serie de radiofrecuencias aparentemente aleatorias, saltando de frecuencia en frecuencia síncronamente con el transmisor. Los receptores no autorizados escucharán una señal ininteligible. Si se intentara interceptar la señal, sólo se conseguiría para unos pocos bits. Es un método para transmitir señales cambiando rápidamente la portadora entre muchas frecuencias, utilizando una secuencia pseudo aleatoria conocida solamente por el transmisor y el receptor. Una transmisión en espectro ensanchado ofrece tres ventajas principales:

- Las señales en espectro ensanchado son altamente resistentes al ruido y a la interferencia.
- Las señales en espectro ensanchado son difíciles de interceptar. Una transmisión de este tipo suena como un ruido temporal, o

como un incremento en el ruido en cualquier receptor, excepto para el que esté usando la secuencia que fue usada por el transmisor.

- Transmisiones en espectro ensanchado pueden compartir una banda de frecuencia con muchos tipos de transmisiones convencionales con mínima interferencia.

De esta forma, los dispositivos Bluetooth usan la totalidad de la banda ISM disponible y si una transmisión es comprometida por interferencias en un canal, la retransmisión será siempre a través de un canal distinto. La duración de cada slot de transmisión Bluetooth es de 625 microsegundos, y generalmente los dispositivos realizan un salto tras cada paquete, que puede cubrir uno, tres o cinco slots.

Si los dispositivos deben saltar a una nueva frecuencia tras cada paquete, han de ponerse de acuerdo previamente en la secuencia de frecuencias a seguir. Los dispositivos Bluetooth pueden actuar en dos modos: maestro o esclavo, siendo el maestro el encargado de fijar la secuencia de saltos, mientras que los esclavos se sincronizan con el reloj y la frecuencia del maestro para seguir la misma secuencia.

Todo dispositivo Bluetooth tiene una dirección única y un reloj interno. En el nivel de banda base se describe un algoritmo para el cálculo de la secuencia de salto de frecuencias a partir de la dirección y el reloj de un dispositivo. Cuando los esclavos se conectan a un maestro, les es comunicado tanto la dirección como el reloj de dicho maestro, información que usarán para calcular la secuencia de saltos en la que trabajarán sincronizados con el dispositivo Bluetooth que actúa como maestro de la comunicación. Como todos los esclavos usan el reloj y la dirección del mismo maestro, todos ellos estarán sincronizados con su secuencia de saltos.

Además de controlar la secuencia de saltos de frecuencia, el maestro controla cuando debe permitir que transmita cada dispositivo, para lo cual hace una distinción entre tráfico de voz o de datos. En caso de que los slots sean de tráfico de datos, solo se permite transmitir a los esclavos cuando se trate de una respuesta a una transmisión que les halla hecho el maestro. Mientras que en tráfico de voz, los esclavos disponen de unos slots reservados regularmente para transmitir sea o no en respuesta al maestro.

El maestro controla cómo dividir el total de ancho de banda disponible entre los esclavos decidiendo cómo y cuándo comunicarse con cada uno de ellos. El número de slots de tiempo que se dedica a cada dispositivo depende de sus necesidades de transmisión de datos. El sistema de división de los slots de tiempo entre los múltiples dispositivos se denomina Multiplexación por División de tiempo o TDM (Time Division Multiplexing).

Un conjunto de dispositivos esclavos trabajando juntos con un maestro común es conocido con el término de piconet o picored. Todos los

dispositivos en una piconet siguen la secuencia de saltos de frecuencia y el cronometraje del maestro. En la Figura la piconet de la izquierda con un solo esclavo representa una conexión punto a punto. La piconet de la derecha con tres esclavos hablando con el maestro representa una conexión de punto a multipunto. Los esclavos en una piconet solo tienen enlace con el maestro, no existiendo enlace directo entre los distintos esclavos

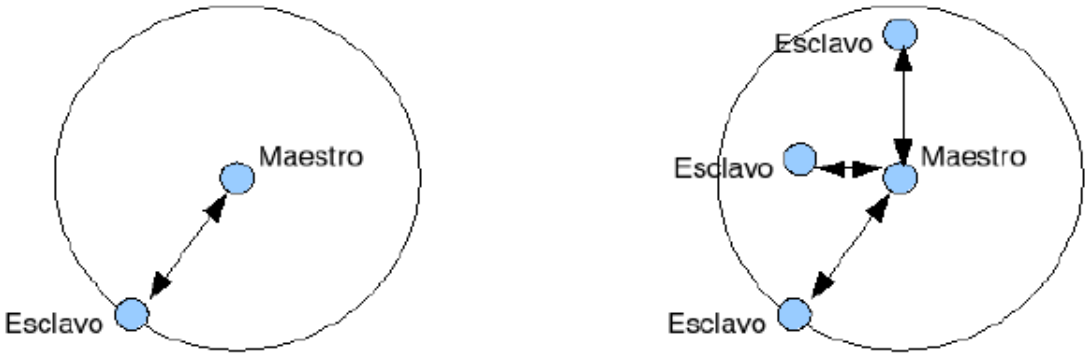


**Figura 14: Esquema del piconet**

La especificación limita el número de esclavos en una piconet a siete, donde cada esclavo solo se comunica con el maestro compartido. Sin embargo, es posible crear áreas de mayor cobertura o con un mayor número de miembros enlazando distintas piconets en una scatternet o red dispersa, donde algunos dispositivos son miembros de más de una piconet.

Cuando un dispositivo está presente en más de una piconet, debe dividir su tiempo, empleando ciertos slots en una piconet y otros slots en la otra piconet. La ilustración de la izquierda en la figura 15 muestra una scatternet donde un dispositivo en esclavo en una piconet y maestro en la otra. La ilustración de la derecha es una scatternet donde un dispositivo es esclavo en dos piconets. No es posible que un mismo dispositivo sea maestro de dos piconets distintas, pues todos los esclavos de una piconet están sincronizados con la secuencia de saltos del maestro.

Por definición, todos los dispositivos con el mismo maestro deben pertenecer a la misma piconet.

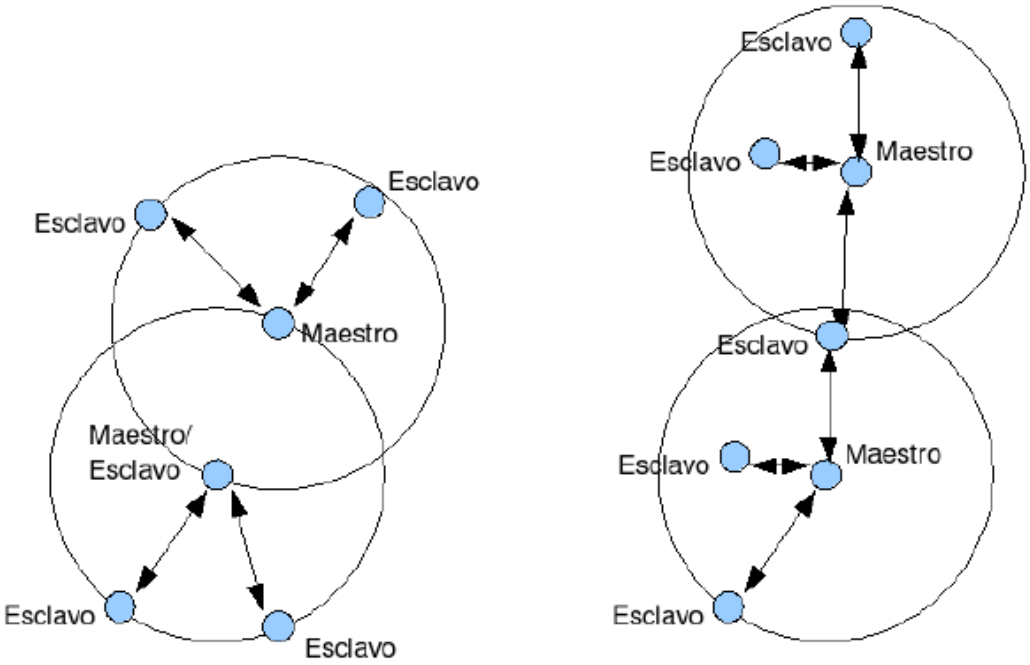


**Figura 15: piconets en una scatternet**

Además de otras fuentes de interferencia (por ejemplo, redes WiFi), la mayor fuente de interferencia para un dispositivo Bluetooth son otros dispositivos Bluetooth. Aunque los dispositivos que compartan una piconet estarán sincronizados para evitarse entre ellos, tras piconets en la misma área pueden colisionar aleatoriamente en las mismas frecuencias. Si existe una colisión en un canal particular, los paquetes implicados serán perdidos y consecuentemente retransmitidos, o en el caso de que sean paquetes de voz, serán ignorados. Por lo tanto, cuantas más piconets haya en un área, más retransmisiones serán necesarias, provocando una caída en las tasas de transmisión. Este efecto que obtenemos con varias piconets independientes en la misma área, pasará también en las scatternets, pues cuando varias piconets se juntan en una scatternet, como ya hemos visto, no coordinan sus saltos de frecuencias.

La especificación Bluetooth permite tres tipos distintos de potencias de radio:

- Clase 1 = 100mW (20 dBm)
- Clase 2 = 2,5mW (4 dBm)
- Clase 3 = 1mW (0 dBm)

| Clase | Potencia máxima permitida (mW) | Potencia máxima permitida (dBm) | Rango (aproximado) |
|---|---|---|---|
| **Clase 1** | 100 mW | 20 dBm | ~100 metros |
| **Clase 2** | 2.5 mW | 4 dBm | ~20 metros |
| **Clase 3** | 1 mW | 0 dBm | ~1 metro |

**Figura 16: potencias de radio**

Estas clases de potencia permiten a los dispositivos Bluetooth conectarse a diferentes rangos. Los dispositivos de Clase 3 pueden comunicarse con un radio de acción de unos 10 metros, sin embargo, como los obstáculos que puedan existir en el camino tienen un efecto de absorción de las microondas, no existe una comunicación fiable en su mayor separación entre dispositivos.

Obviamente, a mayor potencia de emisión se consiguen mayores rangos de alcance. El mayor rango se alcanza con los dispositivos de Clase 1, con los que se podría llegar a los 100 metros. Existe también un rango mínimo de separación entre las conexiones Bluetooth, pues si las emisiones de radio se aproximan mucho el receptor se satura, con lo que se establece un mínimo en torno a los 10 cm.

Así mismo, en función de la revisión del núcleo, los dispositivos Bluetooth tienen distintas capacidades de transmisión:

- Versión 1.1: Hasta 723,1 Kbps
- Versión 1.2: Hasta 1 Mbps
- Versión 2.0 + EDR: Entre 2.1 y 3 Mbps

Bluetooth permite tanto comunicación de datos en los que el tiempo es crítico, como la requerida para comunicación de voz y audio, así como comunicación de paquetes de datos en los que el momento de recepción no es determinante. Para cubrir ambas posibilidades, pueden definirse dos tipos distintos de enlaces entre dos dispositivos cualesquiera. Estos son los enlaces síncronos orientados a conexión o SCO (Synchronous Connection Oriented) para comunicaciones de voz y enlaces asíncronos sin conexión o ACL (Asynchronous Connectionless) para comunicación de datos.

Los paquetes de datos ACL están construidos a partir de un código de acceso de 72 bits, una cabecera de 54 bits y un código CRC o Código de Redundancia Cíclica de 16 bits, junto al campo de carga de datos. Existe una variedad de tipos de paquetes que permiten que sean enviadas distintas cantidades de información. El tipo de paquete que ofrece un mayor campo de carga de datos es el DH5. Un paquete DH5 puede transportar 339 bytes, o 2712 bits de datos. Con lo que son enviados 2858 bits para 2712 bits de información.

Un paquete DH5 usa hasta cinco slots de transmisión, y la longitud mínima de respuesta de un slot. Así, el ratio máximo de datos en la banda base en una dirección es de 723,2 kb/s. En este caso, con paquetes de cinco slots enviados en un sentido, los paquetes de un slot enviados en el otro sentido transportarán tan solo 57,6 kb/s, con lo que esto sería un enlace asimétrico con más datos viajando en el sentido en el que se utilizan paquetes de cinco slots. Si se empleasen paquetes de cinco slots en ambos sentidos, el ratio de datos obtenido sería de 433,9 kb/s, que supone una reducción respecto al ratio de 1 Mb/s que viaja por el aire. Esta sobrecarga en codificación de datos y saltos de frecuencia es necesaria principalmente para proveer de un enlace robusto a través de la banda ISM compartida por tantos dispositivos.

Las capas superiores de la pila del protocolo también usan parte del ancho de banda, por lo que a nivel de aplicación, el máximo ratio de datos puede estar en torno a los 650 kb/s. Los enlaces SCO trabajan a 64 kb/s, y es posible tener hasta tres enlaces de voz fullduplex simultáneamente, o combinar voz y datos. Estos canales de voz ofrecen comunicaciones de audio de la calidad que se esperaría de un sistema de telefonía móvil GSM, sin embargo, no es la suficiente para otras finalidades, como escuchar música, para las cuales no están realmente preparados. Por ejemplo, una alternativa para soportar la transmisión de música es usar un canal ACL para transportar audio.

## 4.3. PILA BLUETOOTH

Como se acaba de ver, Bluetooth es una tecnología de radio de bajo coste y consumo, desarrollada en sus orígenes para sustituir a las conexiones por cable en los dispositivos de telefonía móvil, auriculares y ordenadores portátiles. Esto nos permitiría empezar a hablar, en caso de estandarizarse las comunicaciones inalámbricas entre este tipo de dispositivos, del concepto de Red de Área Personal (PAN – Personal Area Network) , como un tipo de red inalámbrica de corto alcance que revolucionaría la forma en la que las personas interactúan con la información que les rodea.

La especificación Bluetooth es abierta y global, definiendo el sistema completo desde las capas más inferiores de radio hasta las capas superiores

de aplicación. La pila del protocolo suele ser implementada parte en hardware y parte en el software ejecutado en un microprocesador, con diferentes implementaciones que dividen la funcionalidad entre el hardware y el software de distintos modos.

Una parte clave en la especificación Bluetooth es que pretende que un amplio espectro de dispositivos de diferentes fabricantes se comuniquen entre si sin problemas. Con este objetivo, Bluetooth no solo define un sistema de radio, sino que también define la pila software necesaria para que cada dispositivo pueda buscar otros dispositivos próximos a él, descubrir los servicios que estos ofrecen y usar estos servicios. Por lo tanto podemos deducir que dicha especificación no cubre solo la parte más interna o núcleo del protocolo, sino que también describe las partes más altas de la pila y la forma en la que las aplicaciones interactúan con el protocolo.

La Figura 17 muestra a un alto nivel los componentes de la pila del protocolo Bluetooth. Los elementos de dicha pila (protocolos, capas, aplicaciones, etc.) están lógicamente particionados en tres grupos:

- Grupo de protocolo de transporte
- Grupo de protocolo middleware
- Grupo de aplicación

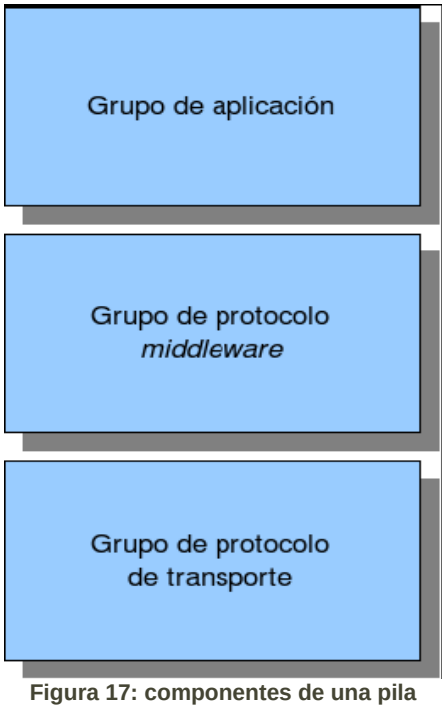


**Figura 17: componentes de una pila**

El grupo de protocolo de transporte está compuesto por los protocolos diseñados para permitir a los dispositivos Bluetooth encontrarse entre ellos así como crear, configurar y gestionar los enlaces físicos y lógicos que permiten a los protocolos de capas superiores y aplicaciones pasar datos a través de estos protocolos de transporte. Los protocolos de este grupo son el de radio, banda base, gestor de enlace, enlace lógico y adaptación y el Interfaz de Controlador de Host (HCI – Host Controller Interface).

El grupo de protocolo middleware consta de protocolos adicionales de transporte necesarios, así como aplicaciones para operar sobre los enlaces Bluetooth. El grupo de protocolo middleware incluye protocolos de terceros y estándares en la industria, así como protocolos desarrollados por el SIG específicamente para comunicaciones inalámbricas Bluetooth. Entre los protocolos que conforman este grupo se incluyen protocolos relacionados con Internet (PPP, IP, TCP, etc.), protocolos de aplicaciones inalámbricas, protocolos de intercambio de objetos adoptados de IrDA, etc. El grupo de protocolos middleware incluye tres protocolos con un diseño independiente de las comunicaciones Bluetooth que facilita un elevado número de otras aplicaciones sobre enlaces Bluetooth:

- Un emulador de puerto serie llamado RFCOMM permite la existencia de aplicaciones heredadas que normalmente interactuarían con un puerto serie.

- Un control de telefonía basada en paquetes nos aporta la posibilidad de operar con sistemas telefónicos, como gestión de grupo y soporte de movilidad para teléfonos inalámbricos y estaciones base.

- Finalmente, un protocolo de descubrimiento de servicios (SDP – Service Discovery Protocol) permite que los dispositivos descubran servicios ofrecidos por otros dispositivos, obteniendo información de cómo hacer uso de los mismos.

El grupo de aplicación consiste en las aplicaciones reales que hacen uso de los enlaces Bluetooth. Estas aplicaciones podrían ser aplicaciones heredadas que desconocen el transporte Bluetooth, como aplicaciones de marcación de modem o navegadores Web.

En conjunto, la pila que representa el protocolo Bluetooth queda representada como se muestra en la Figura 18. En la Figura 19 se muestra una comparativa con la pila OSI (Open System Interconnect), modelo de referencia estándar para pilas de protocolos de comunicaciones.

Aunque Bluetooth no coincide exactamente con el modelo, nos puede resultar útil el comparar las partes de las que constan ambas pilas, pues así

tendremos una aproximación de las labores que cubren cada nivel en la implementación Bluetooth.

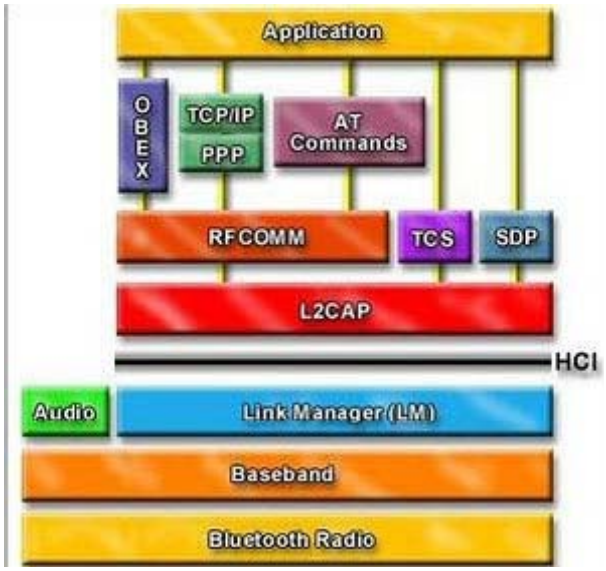


**Figura 18: pila que representa el protocolo bluetooth**

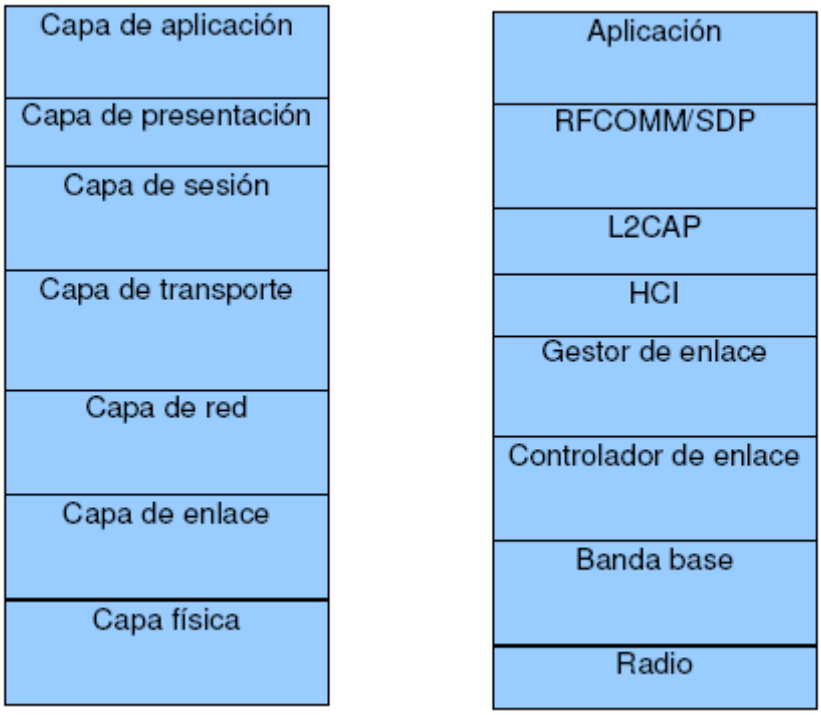


**Figura 19: comparativa con la pila OSI**

Las funciones realizadas por la capa física, como interfaz hacia el medio de comunicación, incluyendo modulación y codificación de canal, serían cubiertas por la capa de radio y parte de la banda base.

La capa de enlace de datos, encargada de la transmisión y control de errores sobre un enlace concreto, se solaparía con el controlador de enlace y las tareas más altas de la banda base, entre las que se incluyen la comprobación y corrección de errores.

La capa de red en el modelo OSI, responsable de la transferencia de datos sobre la red, independientemente del medio o topología específica que siga dicha red, vendría a coincidir con la parte alta del controlador de enlace, configurando y manteniendo múltiples enlaces, así como con la mayoría de las tareas del gestor de enlace.

La fiabilidad y la multiplexación en la transmisión de datos sobre la red hacia el nivel provisto por la aplicación, que en OSI sería cubierto por el nivel de transporte, coincidiría con la parte alta del gestor de enlace y cubriría el interfaz de controlador de host.

La capa de sesión realiza los servicios de gestión y control de flujo de datos, mientras que en Bluetooth serán la capa L2CAP y el límite inferior de RFCOMM/SDP las encargadas de estos fines.

La capa de presentación aporta una representación común para los datos de la capa deaplicación, que es la tarea principal de RFCOMM/SDP. Finalmente, la capa de aplicación en ambas pilas se encarga de la gestión de la comunicación entre las aplicaciones de host.

### 4.3.1.**EL GRUPO PROTOCOLO DE TRANSPORTE**

La Figura siguiente muestra la organización de los protocolos en el grupo de transporte. Estos son los protocolos desarrollados por el SIG para llevar el tráfico de sonido y de datos entre los dispositivos.

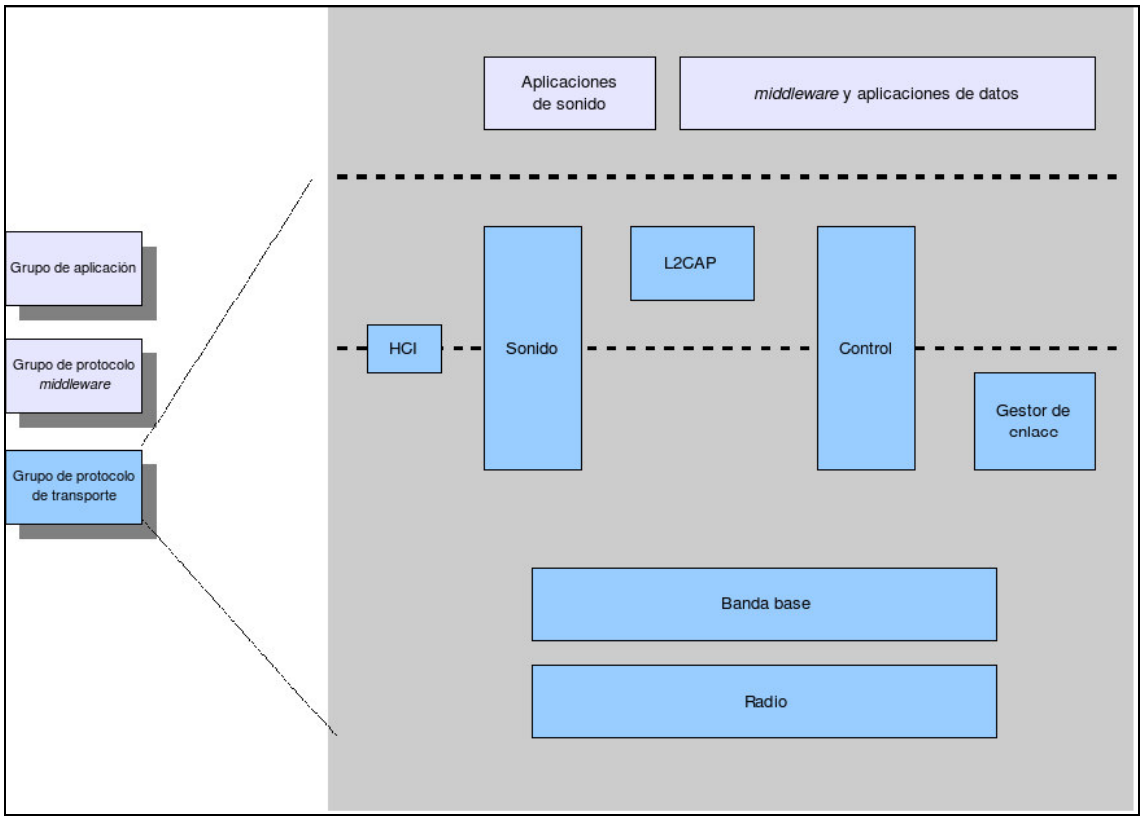


**Figura 20: protocolos en el grupo de transporte**

Los protocolos de transporte soportan tanto transmisión asíncrona para comunicaciones de datos, como síncronas para comunicaciones de voz de una calidad telefónica (64 Kbps). Para mantener la alta calidad de servicio esperada en aplicaciones de sonido, el tráfico es tratado con una alta prioridad. El tráfico de sonido sobrepasa todas las capas de protocolos intermedios y en conducido directamente desde la aplicación de sonido hasta la banda base.

En conjunto, los protocolos del grupo de transporte forman una "tubería virtual" que es usada para transportar los datos de un dispositivo a otro a través del medio aéreo. Es decir, estos protocolos definen el mecanismo de transporte de datos entre los dispositivos implicados en la comunicación, de ahí el nombre elegido para este grupo.

## CAPA L2CAP

El tráfico de las aplicaciones de datos es en un primer momento conducido a través de la capa del Protocolo de Adaptación y Control del Enlace Lógico (L2CAP – Logical Link Control and Adaptation Protocol). La capa L2CAP abstrae a los protocolos de capas superiores y aplicaciones de los detalles de los protocolos de capas más bajas. Esta capa soporta multiplexación de protocolos, permitiendo que múltiples protocolos y aplicaciones compartan el mismo medio. También aporta la segmentación de los grandes paquetes usados en capas superiores en paquetes más pequeños para las transmisiones de banda base, y el correspondiente reensamblado en el dispositivo receptor. Además, las respectivas capas L2CAP de dos puntos de

comunicación, facilitan el mantenimiento de la calidad de servicio deseada mediante negociación, basándose en el nivel de servicio solicitado.

## CAPA DE GESTOR DE ENLACE

Los gestores de enlace de cada dispositivo negocian las propiedades del interfaz aéreo entre los respectivos dispositivos Bluetooth usando el Protocolo Gestor de Enlace (LMP – Link Manager Protocol). Estas propiedades incluyen el fijar el ancho de banda para soportar la calidad de servicio deseada para tráfico de datos (L2CAP) y la reserva periódica de ancho de banda para transmitir el tráfico de sonido. El gestor de enlace Bluetooth usa un esquema de desafío-respuesta para autenticar los dispositivos implicados en la comunicación. Este también se encarga de supervisar el emparejamiento de dispositivos y el cifrado, cuando sea necesario, del flujo de datos transmitido. Si la autenticación falla, el gestor de enlace puede cortar en enlace entre los dispositivos, prohibiendo cualquier comunicación entre los mismos. El control de energía es gestionado en este nivel negociando los modos de operación de baja actividad de la banda base.

## CAPAS DE BANDA BASE Y RADIO

La capa de banda base determina e instancia el interfaz aéreo Bluetooth. Define el proceso por el cual los dispositivos buscan a otros dispositivos y averiguan como conectarse a ellos, además del esquema maestro/esclavo en el que se estructura toda comunicación Bluetooth entre varios dispositivos. Detallar que este concepto de maestro/esclavo tan solo es distinguido en las capas más bajas, y a partir del gestor de enlace (capa L2CAP), hacia capas superiores, tan solo se habla de un modelo punto a punto, no haciéndose distinción entre papel de maestro y de esclavo. En la capa de banda base se definen distintos parámetros de bajo nivel de la comunicación, como:

- Secuencia de salto de frecuencia
- Reglas para compartir el medio aéreo; estas reglas están basadas en un esquema de Duplicación por División de Tiempo (TDD – Time Division Duplex).
- Soporte de tráfico síncrono y asíncrono
- Procedimientos de procesamiento de paquetes, entre los que se incluye la detección y corrección de errores.

## CAPA HCI

Radio, banda base y gestor de enlace pueden ser agrupados en el módulo Bluetooth. El módulo unido al dispositivo host, permite que dicho dispositivo haga uso de las comunicaciones inalámbricas Bluetooth. En esta configuración, el host contiene la capa L2CAP y las capas superiores de la pila.

Para permitir a los desarrolladores el operar con módulos Bluetooth de distintos fabricantes, la especificación define un interfaz común para acceder a las capas bajas de a pila que reside en el módulo, con independencia del interfaz físico particular que conecta el host al módulo. El Interfaz de Controlador de Host (HCI – Host Controller Interface) permite a las capas altas de la pila, incluidas las aplicaciones, el acceso a la banda base, gestor de enlace y otros registros hardware a través de un interfaz simple estándar.

Aunque la capa HCI típicamente reside bajo la capa L2CAP, no es una parte requerida de la especificación. Ha sido desarrollada tan solo con objetivo de permitir la interoperabilidad entre dispositivos host y módulos Bluetooth, cada uno de los cuales puede venir de distintos fabricantes o desarrolladores.

El camino de control mostrado en la Figura 3.6 es usado para comunicar información de control entre capas. Típicamente, aunque no exclusivamente, los controles que están expuestos a capas superiores (incluyendo al usuario final) son para fijar un modo de operación para el dispositivo que persiste hasta que ese modo es modificado explícitamente de nuevo a través de una acción originada en una capa superior. Un cambio de este tipo es, por ejemplo, habilitar o deshabilitar manualmente la autenticación o el cifrado para un dispositivo dado. El camino de control no está definido explícitamente en la especificación, pero se aprecia entrelazando varios protocolos de la pila. Sin embargo, la especificación HCI incluye el conjunto de información que transporta el camino de control.

## 4.3.2.EL GRUPO PROTOCOLO MIDDLEWARE

La Figura siguiente esquematiza este grupo. Los protocolos middleware hacen uso de los ya vistos protocolos de transporte y presentan interfaces estándar a las capas superiores que pueden ser usados para la comunicación a través de los transportes. Cada una de las capas middleware define un protocolo estándar que permite a las aplicaciones usar un nivel superior de abstracción que dirigiría las comunicaciones con los protocolos de transporte de capas inferiores. Los protocolos middleware consisten en:

- RFCOMM, una abstracción del puerto serie
- Protocolo de Descubrimiento de Servicios (SDP – Service Discovery Protocol), usado para describir los servicios disponibles y localizar los necesitados
- Un conjunto de protocolos de interoperabilidad IrDA adoptados del estándar IrDA que permiten el uso de aplicaciones disponibles para dicho estándar

- Especificación de Control de Telefonía (TCS – Telephony Control Specification), usado

- para controlar las llamadas telefónicas que deben ser usadas para voz o para datos

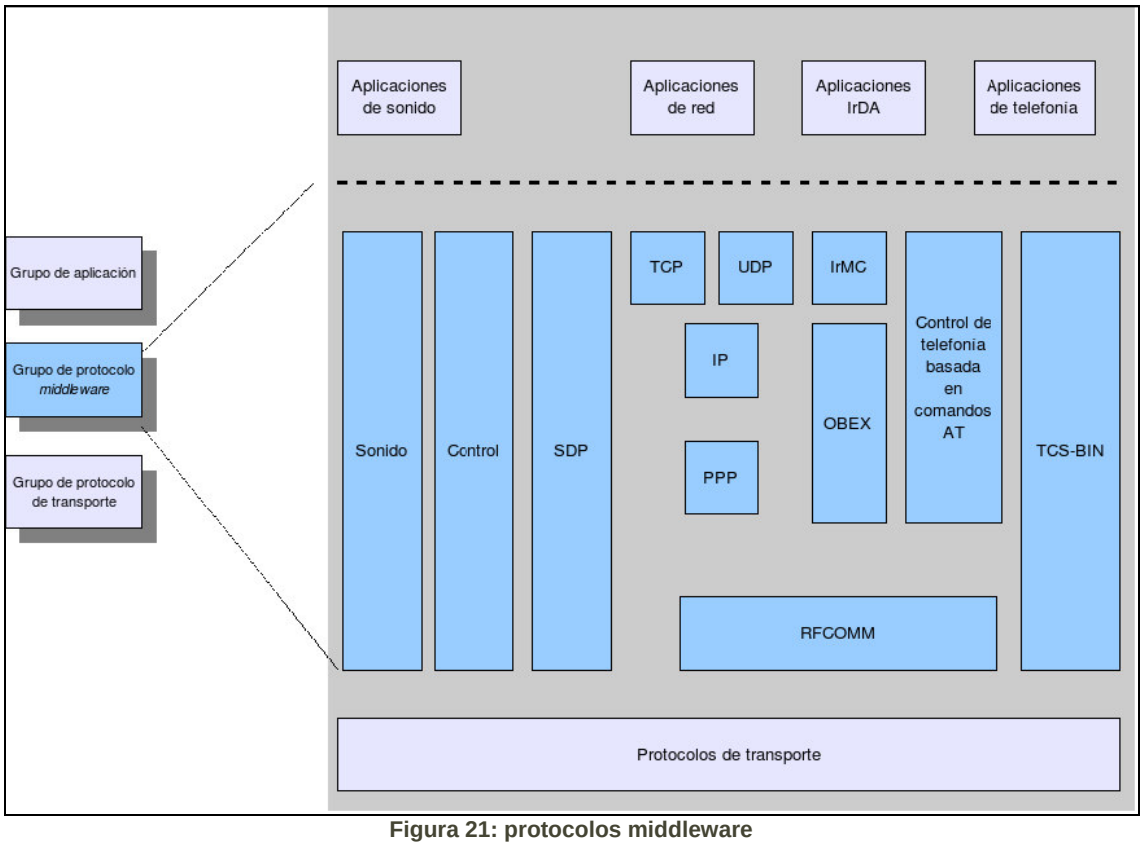


**Figura 21: protocolos middleware**

## CAPA RFCOMM

Los puertos serie son uno de los interfaces de comunicación más comunes hoy en día usados en informática y comunicaciones. La mayoría de las comunicaciones serie implican el uso de un cable para transferir datos a través de los puertos serie. En su intento de reemplazar los cables, para las comunicaciones inalámbricas Bluetooth, es una característica importante el tener soporte para comunicaciones serie y aplicaciones relacionadas.

Para facilitar el uso de comunicaciones serie sobre enlaces inalámbricos Bluetooth, la pila del protocolo define una abstracción del puerto serie llamada RFCOMM (Radio Frecuency Communication). RFCOMM presenta un puerto serie virtual que permite la herencia de aplicaciones que hicieran uso de dicho puerto.

RFCOMM está modelado según el estándar TS 07.10 del European Telecommunications Standards Institute (ETSI), el cual define las comunicaciones multiplexadas en serie sobre un enlace serie simple.

## CAPA SDP

En algunos aspectos, la instanciación de cualquiera de los modelos de uso Bluetooth podría ser visto como hacer uso de un conjunto de servicios. Una motivación principal para la creación de redes de dispositivos es la de comunicar los mismos y hacer uso de los servicios implementados por otros. Por tanto, una vez que se ha establecido un canal de comunicación, el siguiente paso podría ser la búsqueda de servicios ofrecidos por otros dispositivos, y esto es lo que aborda el Protocolo de Descubrimiento de Servicios (SDP – Service Discovery Protocol).

SDP define un método estándar a los dispositivos Bluetooth para descubrir y saber acerca de los servicios ofrecidos por otros dispositivos; igualmente, define una forma para que los dispositivos describan los servicios de los que disponen para otros dispositivos.

## PROTOCOLOS DE INTEROPERABILIDAD IRDA

El estándar IrDA (Infrared Data Association) define protocolos para intercambio y sincronización de datos en entornos inalámbricos. El SIG ha adoptado varios de los protocolos y modelos de datos de IrDA porque éste y Bluetooth comparten varios atributos importantes, escenarios de uso y aplicaciones.

Un requisito fundamental para el intercambio de datos entre dispositivos es definir el formato, sintáctica y semánticamente, de los propios datos. El protocolo de intercambio de objetos por infrarrojos (IrOBEX – Infrared OBject Exchange, más conocido simplemente como OBEX) desarrollado por el IrDA es un protocolo de sesión para comunicaciones punto a punto.

Entre las aplicaciones que hacen uso de OBEX está el intercambio de objetos bien definidos. Objetos de datos como tarjetas de presentación electrónicas (formato vCard) y otros pueden ser intercambiados usando el protocolo OBEX, pues este es una parte fundamental en los modelos de uso de la transferencia de ficheros (intercambio de objetos) y de la carga de objetos.

## CAPAS DE RED

Las comunicaciones inalámbricas Bluetooth usan una topología de red punto a punto. Sin embargo, se tiene en cuenta el trabajo en red, y la viabilidad de conexión a redes mayores a través de conexiones por marcado o a través de puntos de acceso. Bluetooth también trata la interoperabilidad con el Protocolo de Aplicaciones Inalámbricas (WAP – Wireless Aplication Protocol), una especificación para transmisión inalámbrica usada por dispositivos como teléfonos móviles.

La conexión por marcado usa la capa de comandos AT del grupo de protocolos middleware de la pila. En muchos casos, la red a la que se accede está basada en el Protocolo de Internet, a la que nos referiremos como red IP. Una vez que se establece una conexión por marcado a una red IP, los protocolos estándar de Internet (como TCP, UDP, HTTP, etc.) pueden ser usados por el dispositivo que inició la conexión de red para interactuar con la misma.

Un dispositivo puede conectarse también a una red IP a través de un punto de acceso. En este caso, un enlace Bluetooth conecta el dispositivo al punto de acceso a la red, conectándose entonces el punto de acceso a la propia red. El Protocolo Punto a Punto (PPP – Point-to-Point Protocol) es usado sobre el enlace Bluetooth para realizar la conexión al punto de acceso. Al igual que el acceso por marcado, una vez que la conexión PPP está establecida, los protocolos estándar de Internet pueden ser usados para interactuar con la red.

## CAPA TCS Y SONIDO

Como ya se ha dicho, una ventaja clave de las comunicaciones inalámbricas Bluetooth es su capacidad de transportar tanto tráfico de voz como de datos. La capa de Especificación de Control de Telefonía (TCS – Telephony Control Specification) Bluetooth está diseñada para soportar funciones de telefonía, incluyendo control de llamadas y gestión de grupos. Estas operaciones están frecuentemente asociadas con llamadas de voz en las cuales se usa TCS para configurar los parámetros de la llamada; una vez que se establece la llamada, un canal de sonido Bluetooth puede transportar el contenido de voz de la propia llamada. TCS puede ser usado también para configurar las llamadas de datos, como son las utilizadas por el perfil de marcado para acceso a la red; en este caso, el contenido de la llamada es transportado como paquetes de datos sobre L2CAP.

Como estos protocolos TCS están codificados en binario, están definidos en la especificación como TCS-BIN. Durante el desarrollo de la especificación Bluetooth, el SIG también consideró un segundo protocolo TCS denominado TCS-AT. Este definía un protocolo de control de modem (frecuentemente llamado comandos AT) que eran inducido a través de la capa RFCOMM. Aunque los comandos AT sobre RFCOMM son ampliamente usados por algunas aplicaciones, la especificación finalmente no define un protocolo separado para TCSAT.

El protocolo TCS-BIN incluye funciones de control de llamada, funciones de gestión de grupo y un método para que los dispositivos intercambien información de señalización de llamada sin llevar a cabo realmente la llamada o tener establecida una conexión de lamada. El sonido, especialmente la voz, es tratado de forma privilegiada en una comunicación Bluetooth. Como el tráfico de sonido tiene asociado un factor tiempo, típicamente es encaminado directamente a y desde la capa de banda base, sin tener que pasar por ciertas capas como L2CAP. Son definidas

estructuras especiales de paquetes de la banda base, llamados paquetes síncronos orientados a conexión (SCO – Synchronous Connection-Oriented) para este tipo de tráfico. La comunicación Bluetooth permite hasta tres canales de sonido al mismo tiempo, dejando cierto ancho de banda para tráfico de datos.

### 4.3.3.EL GRUPO APLICACIÓN

Con este grupo nos referimos al software que reside sobre la pila del protocolo tal y como es definida por el SIG. Es decir, el software que es suministrado por los fabricantes de dispositivos, desarrolladores de software independientes o cualquier otro que haga uso de la pila del protocolo para llevar a cabo ciertas funciones que resulten de utilidad para el usuario del dispositivo Bluetooth. En la Figura siguiente se muestran las posibilidades de organización del software de aplicación Bluetooth.

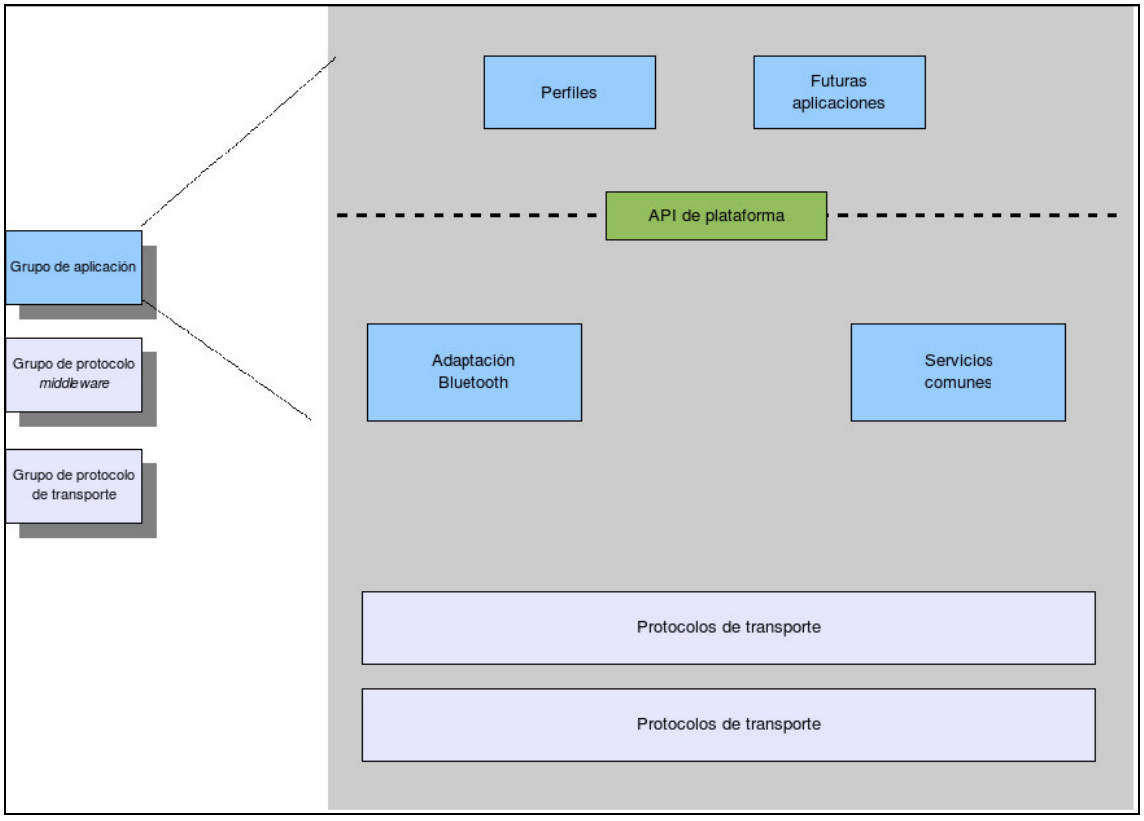


**Figura 22: protocolo de aplicación**

Son de gran interés en el grupo de aplicación aquellas aplicaciones que instancian los perfiles Bluetooth. Esto es, dada una pila de protocolo Bluetooth en un dispositivo, es necesario aun software de aplicación que conduzca esa pila a realizar funciones como acceso en red por marcado, transferencia de ficheros, comunicaciones por auriculares o muchas otras. El SIG define tan solo los protocolos middleware y de transporte para la pila;

pero no define protocolos de aplicación en sí mismos, ni define interfaces de programación de aplicaciones (API – Application Programming Interfaces). Con lo que se necesitan aplicaciones para realizar los escenarios de uso previstos para comunicaciones inalámbricas Bluetooth.

Un perfil Bluetooth es el código de aplicación añadido que usa la pila de protocolo subyacente para aportar un valor práctico al usuario final.

## 4.4. SEGURIDAD EN BLUETOOTH

Los datos que enviamos a través del Bluetooth son datos que se mandan de un dispositivo a otro sin necesidad de cables. Esta información privada debe ser enviada a su destinatario de forma segura, sin ser interceptada. Los estándares que rigen las comunicaciones inalámbricas están evolucionando y presentan varios formatos para garantizar la seguridad de sus usuarios, como en el caso de la tecnología inalámbrica Bluetooth.

Desde su creación, la tecnología inalámbrica Bluetooth ha hecho hincapié en la protección de las comunicaciones inalámbricas. El grupo de interés especial (SIG) de Bluetooth cuenta con un grupo de expertos en seguridad formado por ingenieros de las empresas afiliadas. Este grupo suministra información decisiva sobre cuestiones de seguridad y sus consejos se tienen en cuenta en el proceso de desarrollo de las especificaciones inalámbricas de la tecnología Bluetooth.

Existen tres modos de seguridad para las conexiones entre dos dispositivos con tecnología Bluetooth:

- Modo de seguridad 1: no seguro
- Modo de seguridad 2: seguridad impuesta a nivel del servicio
- Modo de seguridad 3: seguridad impuesta a nivel del enlace

El fabricante de cada producto determina el modo de seguridad del mismo. Los dispositivos y los servicios también cuentan con distintos niveles de seguridad. Los niveles para los dispositivos son dos: "***dispositivo de confianza***" y "***dispositivo poco fiable***". Un dispositivo de confianza, una vez emparejado con otro dispositivo, tiene acceso sin restricciones a todos los servicios. Los servicios cuentan con tres niveles de seguridad:

- servicios que precisan autorización y autenticación.
- servicios que sólo necesitan autenticación.

- servicios abiertos a todos los dispositivos.

El algoritmo de cifrado de las especificaciones Bluetooth es seguro. Entre otros dispositivos que utilizan este algoritmo encontramos ratones o teclados conectados a PC, teléfonos móviles sincronizados con ordenadores o PDA que utilizan teléfonos móviles como módem.

Los casos en los que ha peligrado la seguridad de los datos de teléfonos móviles han sido el resultado de problemas de implementación en esa plataforma en particular. Los problemas registrados últimamente en relación con el acceso de "piratas informáticos" expertos a la información almacenada en teléfonos móviles, mediante la tecnología Bluetooth, se deben a una implementación incorrecta. Estos métodos de acceso ilegal a la información reciben los nombres de "***bluesnarfing***" y ***"bluebugging***".

El ***bluejacking*** se refiere al envío anónimo de tarjetas de visita usando tecnología inalámbrica Bluetooth. No modifica ni suprime ningún dato del dispositivo receptor. Por lo general, las tarjetas de visita enviadas contienen un mensaje de tono ligero, en lugar del nombre y el número de teléfono del remitente. Los bluejackers suelen estar atentos a la señal sonora de recepción del mensaje o a la reacción del usuario. Una vez identificado el dispositivo receptor, envían otro mensaje, más personalizado. Para ello, es necesario que los dispositivos emisor y receptor se encuentren situados a menos de diez metros de distancia entre ellos. Los receptores de este tipo de mensaje no deben añadir el contacto a la agenda telefónica de su dispositivo. Los dispositivos en modo invisible no son susceptibles a este tipo de ataques.

El ***bluebugging*** consiste en que personas con los conocimientos necesarios obtienen acceso a las funciones del teléfono móvil a través de la tecnología inalámbrica Bluetooth, sin que el usuario del teléfono sea notificado o alertado de ello. Esta vulnerabilidad permite al pirata informático iniciar llamadas telefónicas, enviar y recibir mensajes de texto, leer y escribir datos en la agenda telefónica, escuchar conversaciones y conectarse a Internet. Como en los demás casos, el pirata debe hallarse en un radio de diez metros del teléfono en cuestión, a menos que cuente con equipo especializado. Se trata de una vulnerabilidad distinta a la del bluesnarfing, y no afecta necesariamente a los mismos teléfonos.

El ***bluesnarfing*** es el procedimiento por el cual los piratas informáticos acceden a los datos almacenados en un teléfono con tecnología Bluetooth usando esa misma tecnología, sin alertar al usuario del teléfono de que se ha establecido una conexión con su dispositivo. Este tipo de ataque permite acceder a la agenda telefónica y a sus imágenes asociadas, a la agenda y al código de identificación internacional del dispositivo móvil (IMEI). Si se configura el dispositivo en modo invisible, es mucho más difícil de localizar y atacar. Sin el equipo apropiado, el pirata informático debe encontrarse a menos de diez metros de distancia del dispositivo y contar con software específico. Sólo algunos teléfonos antiguos con tecnología Bluetooth son vulnerables al bluesnarfing.

El ***car whisperer*** es un sistema desarrollado por investigadores en seguridad para conectarse y permitir la recepción y envío de audio desde dispositivos manos libres con tecnología Bluetooth instalados en vehículos. Con este sistema, una persona podría conectarse remotamente y comunicarse con un automóvil desde un dispositivo no autorizado, enviar contenido sonoro al auricular y recibirlo a través del micrófono del dispositivo. A menos que se cuente con un equipo sofisticado, hay que estar situado a diez metros del coche como máximo y transmitir desde un ordenador portátil equipado con el sistema car whisperer. El objetivo de estos investigadores era poner de manifiesto la vulnerabilidad de algunos modelos de manos libres para el coche con tecnología Bluetooth e instar a los fabricantes a que mejoraran la seguridad de su tecnología Bluetooth.

Para que el car whisperer pueda acceder al dispositivo manos libres, éste ha de estar en modo de emparejamiento constantemente, tener un código PIN predeterminado de cuatro dígitos y no estar conectado a ningún teléfono. Si el usuario tiene un teléfono emparejado constantemente con el dispositivo manos libres, ningún equipo no autorizado podrá conectarse.

En estos momentos, si se toman las medidas de protección apropiadas, como activar la seguridad del dispositivo, usar números PIN razonablemente largos y emparejar los dispositivos en privado, las únicas posibilidades de ataque son las mencionadas con anterioridad y sólo afectan a un número limitado de productos del mercado.

***Cabir*** es un programa de software maligno, también conocido como "***malware***". Cuando se instala en un teléfono, utiliza la tecnología Bluetooth para propagarse, enviándose a sí mismo a otros dispositivos vulnerables. Este comportamiento de replicación automática le otorga la clasificación de "***gusano***". En la actualidad, el gusano cabir afecta únicamente a teléfonos móviles que utilizan la interfaz de usuario Symbian serie 60 y la tecnología inalámbrica Bluetooth. Para que el teléfono resulte infectado, el usuario debe aceptar el gusano e instalar el malware de forma manual. Puede obtenerse más información sobre este gusano en Symbian.

El número de identificación personal (PIN) es un código alfanumérico compuesto de cuatro o más caracteres que se asocia temporalmente a un producto para realizar un emparejamiento de forma segura. Se recomienda el uso de un PIN alfanumérico con ocho o más caracteres siempre que sea posible. Los propietarios de los dispositivos deben compartir su número PIN únicamente con individuos y dispositivos de confianza. No es posible emparejar dispositivos sin utilizar este número PIN.

En teoría, un pirata podría supervisar y registrar la actividad del espectro de frecuencias y usar un ordenador para regenerar los códigos PIN intercambiados. Para ello, es preciso disponer de dispositivos específicos y un conocimiento exhaustivo de los sistemas Bluetooth. Si se usan códigos PIN con ocho o más caracteres alfanuméricos, el pirata podría tardar años en descubrirlos. Si se opta por códigos PIN de cuatro caracteres numéricos,

en cambio, el pirata podría descubrirlo en cuestión de horas. En cualquier caso, el pirata debe disponer de programas de software muy avanzados.

Los productos con tecnología Bluetooth garantizan una conexión segura por medio del proceso de emparejamiento inicial. En dicho proceso, se solicita que uno o ambos dispositivos introduzcan un código PIN, a partir del cual los algoritmos internos generan la clave de seguridad que se utilizará para la autenticación de dichos dispositivos cada vez que se conecten.

Un nuevo estudio ha planteado un procedimiento mediante el cual, en teoría, se podría deducir la configuración de seguridad de dos dispositivos Bluetooth emparejados. Para poder hacerlo, el dispositivo que pretende infiltrarse tendría que registrar el sonido del proceso inicial de emparejamiento, que se realiza en una única ocasión. A continuación, podría servirse de un algoritmo para adivinar la clave de seguridad y hacerse pasar por el otro dispositivo Bluetooth. La novedad de dicho estudio es que expone un procedimiento que obliga a la introducción de una nueva secuencia de emparejamiento entre los dos dispositivos y presenta un método mejorado en el proceso de deducción, lo que reduce considerablemente el tiempo respecto a otros ataques anteriores.

Sin embargo, es imprescindible que el pirata oiga el proceso de emparejamiento inicial, que normalmente sólo se produce una vez, en un entorno privado y que no dura más de una fracción de segundo. Los autores del estudio han propuesto algunos métodos con los que se podría forzar la eliminación de la clave de seguridad en uno de los dos dispositivos Bluetooth, de forma que se iniciara un nuevo proceso de emparejamiento que el pirata podría oír. Para ello, el pirata debería hacerse pasar por el segundo dispositivo durante la conexión. El equipo necesario para realizar este proceso es muy costoso y solamente lo suelen usar los desarrolladores. De tener éxito dicho proceso, en el dispositivo del usuario aparecería un mensaje solicitándole que introdujera de nuevo su código PIN. Si el usuario lo introdujera en presencia del atacante, y el nuevo código PIN fuera lo suficientemente corto, en teoría, el pirata podría hacerse con él.

Es decir, en el caso de que el código PIN empleado constara sólo de cuatro caracteres numéricos, un ordenador con gran capacidad de procesamiento podría calcular la clave de seguridad en menos de una décima de segundo. Cuanto más largo sea el código PIN, mayor será el tiempo que se tardará en descifrarlo. En el caso de que el código constara de ocho caracteres alfanuméricos, se tardaría más de cien años en calcularlo, lo cual haría prácticamente imposible un ataque de este tipo.

Se trata, no obstante, de un estudio teórico sobre la seguridad de los productos que utilizan la tecnología Bluetooth. Los métodos que propone son técnicamente posibles, pero es muy poco probable que un usuario normal llegue a ser víctima de uno de estos ataques. No hay que olvidar que estos ataques se aprovechan de la ingenuidad de los usuarios, de modo

que tratar de conocer cómo funciona el proceso de emparejamiento es una forma de defenderse.

Los ***ataques de denegación de servicio (DoS)*** son muy populares y suelen dirigirse a sitios Web y redes. Ahora, también pueden utilizarse contra dispositivos compatibles con la tecnología inalámbrica Bluetooth. Se trata más que nada de una molestia y el procedimiento ni resulta ingenioso ni tiene nada de especial. Consiste, simplemente, en una solicitud constante de respuesta que se envía a un dispositivo Bluetooth desde un sistema pirata igualmente compatible con esta tecnología y equipado con software específico. Esta solicitud constante produce una degradación temporal de la batería del dispositivo receptor. Mientras ocupa el enlace Bluetooth con solicitudes de comunicación no válidas, el pirata informático puede desactivar temporalmente los servicios Bluetooth del producto.

Los ataques DoS solamente ofrecen al pirata informático la satisfacción de causar una molestia temporal; no tienen la capacidad de permitir acceso a los datos o servicios del dispositivo. El atacante no podrá utilizar ni robar la información almacenada en el dispositivo receptor.

Los ataques DoS pueden dirigirse contra cualquier dispositivo Bluetooth visible. En algunos casos, piratas informáticos expertos pueden descubrir, también, la dirección de dispositivos Bluetooth invisibles.

Hasta la fecha, los ataques DoS en dispositivos Bluetooth solamente se han llevado a cabo en pruebas de laboratorio. Si tenemos en cuenta los requisitos y el corto alcance de la tecnología inalámbrica Bluetooth, el riesgo de un ataque DoS intencionado debe considerarse mínimo.

Para más información acerca de seguridad bluetooth, consultar el sitio web bluetooth.com de donde hemos extraído esta información.

# 5. ENTORNO DE DESARROLLO Y HERRAMIENTAS UTILIZADAS

## 5.1. CARBIDE

Tanto Carbide.c++ Developer Edition como Carbide.c++ Professional Edition son dos suites para la programación de software para smartphones Symbian desarrollados por Nokia.

Carbide.c++ está basado en el framework open source Eclipse, los desarrollos con esta herramienta serán compatibles con todos los terminales que funcionen bajo Symbian, incluyendo los basados en UIQ, MOAP y por supuesto, S60 3rd Edition.

El Carbide.c++ ayuda a incrementar la productividad y reduce los tiempos de desarrollo de aplicaciones Symbian. Esta basado en una intuitiva interfaz. En la versión Professional incluye una paleta para arrastrar y soltar los elementos usados en el diseño de la aplicación a la interfaz del programa. Ambos También constan con herramientas para depurar el código y perfeccionar las aplicaciones.

Para empezar a trabajar con el Carbide.c++, una vez ejecutado, tenemos dos opciones: importar un proyecto existente a través de su archivo .mmp o crear un proyecto nuevo eligiendo una de las plantillas disponibles. En ambos casos, la herramienta creará un directorio de proyecto con, a su vez, varios directorios que contendrán los diferentes archivos utilizados. De estos directorios los más importantes, del punto de vista del programador, son:

- NombreProyecto
- NombreProyecto\.generated
- NombreProyecto\.settings
- NombreProyecto\data
- NombreProyecto\gfx
- NombreProyecto\inc
- NombreProyecto\S60 2.0 Emulator Debug

- NombreProyecto\sis
- NombreProyecto\src

Contienen, respectivamente, los archivos de recurso (data), los archivos de cabecera (inc) y los archivos fuente (src).

Los archivos de recurso contienen código que representan elementos de la interfaz gráfica.

En el caso de crear un proyecto nuevo, deberemos elegir qué tipo de aplicación nos interesa. Fundamentalmente son los siguientes tipos:

- **Librería dinámica (DLL):** si queremos crear un proyecto que simplemente ofrezca una funcionalidad en forma de librería dinámica.
- **Librería estática (LIB):** si queremos crear un proyecto que simplemente ofrezca una funcionalidad en forma de librería estática.
- **Aplicación básica de consola:** útil si sólo queremos hacer un programa que funcione sin necesidad de interfaz gráfica. No funciona en todos los emuladores.
- **Aplicación de GUI:** aplicación con interfaz gráfica.

Habrá que seleccionar también para qué objetivo queremos construir la aplicación. En el caso de que tengamos varios emuladores instalados (por ejemplo uno de Series 60 y uno de UIQ), Carbide.c++ nos dejará elegir entre ellos.

Una vez hecho esto, la herramienta nos generará los archivos necesarios, estructurados según los directorios citados anteriormente. Para construir la aplicación bastará con hacer clic en Project / **Build All**; para lanzar el emulador hay que hacer clic en **Run**. Una vez dentro del emulador hay que encontrar el ejecutable del nuevo programa y lanzarlo; en el caso del Series 60 2nd edition hay que acceder al directorio Installat, en el cual se encontrará la aplicación recientemente creada.

Un aspecto importante a la hora de la compilación es la inclusión de librerías utilizadas para el proceso de enlazado. En Carbide.c++ hay que realizar dos pasos muy sencillos: primero se debe incluir el archivo de cabecera de la librería utilizada en nuestro código; acto seguido hay que acceder a las propiedades del proyecto y en el apartado C/C++ Build, en WINSCW C/C++ Linker y por último Libraries se debe hacer clic en Add y recorrer los directorios hasta encontrar la librería deseada (archivo .lib).

# 5.2. NOKIA CONECTIVITY FRAMEWORK

Nokia Connectivity Framework (NCF) es una herramienta proporcionada por Nokia que nos permite visualizar, testear y simular entornos que utilicen los emuladores Nokia SDK. Sirve para simular el envío de SMS, MMS o mensajes y datos mediante el bluetooth o los infrarrojos entre varios terminales móviles.


**Figura 23: NCF 1.2.**

Esta herramienta nos da la posibilidad de probar aplicaciones que hayamos realizado sin necesidad de exportar dicha aplicación a un terminal físico, con los problemas que ello puede conllevar, como puede ser la elaboración de un código defectuoso, con resultados no esperados, perjudiciales para el dispositivo físico.

NCF puede descargarse del forum Nokia. Para mas información sobre su funcionamiento se puede consultar su UserGuide.

## 5.3. FORUM NOKIA

Se trata de una comunidad de desarrolladores que impulsa la creación de programas de software para móviles.

Dentro de este forum encontramos diversos foros, enlaces, material, software… que ayudan al desarrollo de aplicaciones para móviles Symbian:

- Informaciones sobre nuevos dispositivos y tecnologías disponibles en cada región.
- Herramientas, SDK y emuladores.
- Instrucciones, preguntas frecuentes y soporte técnico.
- Manuales diversos.

Este foro nos ha ayudado en diferentes dudas que nos iban surgiendo. Para poder hacer uso de él es necesario abrirse una cuenta en el mismo.

Como variante tenemos Forum Nokia PRO es un programa creado para ayudar a empresas de desarrollo. Ofrece un acceso exclusivo a la información sobre nuevos lanzamientos de aplicaciones, herramientas de desarrollo de software, documentación técnica, atención preferente para resolver cualquier duda o problema técnico, apoyo al desarrollo del negocio, préstamo de terminales... Se trata de un servicio de pago dirigido exclusivamente a empresas de desarrollo.

# 6. CHAT BLUETOOTH

## 6.1. INTRODUCCIÓN

Una de las partes de nuestro proyecto ha sido la implementación de un Chat bluetooth para móviles. La intención de este desarrollo ha sido principalmente didáctica, ya que debido al corto alcance del bluetooth solo permite chatear con gente que este próxima, en la misma habitación o en la de al lado, lo que hace que nuestro Chat no sea muy útil, aunque también se le podría dar otros usos, como para copiar en los exámenes (aunque por supuesto nuestra intención) o evitar que una tercera persona se entere de la conversación.

## 6.2. PASOS PARA EL DESARROLLO

Nuestro Chat sigue una funciona con estructura cliente-servidor. Cada Terminal realiza los dos roles, el servidor se encarga de publicar que el móvil esta conectado al Chat y de escuchar a la espera de recibir mensajes. El servidor se encarga de conectar con el móvil deseado y enviar mensajes. También hay otro tercer bloque, la interfaz de usuario. A continuación vamos a describir cada parte mas detalladamente.

### 6.2.1.SERVIDOR

Las tareas del servidor son principalmente tres:

- **Publicar que el móvil esta conectado**:
  Una aplicación Bluetooth necesita publicar que esta conectado para que los demás terminales pueden encontrarlo y conectar con el. El sistema Operativo Symbian nos proporciona la SDP (Service Discovery Protocol) database para este fin. Los terminales conectados y los servicios que estos ofrecen aparecen publicados como registros en estas bases de datos. Por eso lo primero que tiene que hacer nuestro servidor es registrarse en la SDP.

- **Fijar el tipo de seguridad que vamos a usar:**
  Fijar el modo en que nuestro Terminal se muestra al resto (oculto o publico), si nuestros mensajes , si es necesaria autorización o no para comunicar con nosotros, si nuestros mensajes con cifrados o no, los puertos que vamos a usar etc.

- **Estar a la escucha de conexiones:**
  Estar esperando la conexión y envío de otros móviles conectados al Chat.

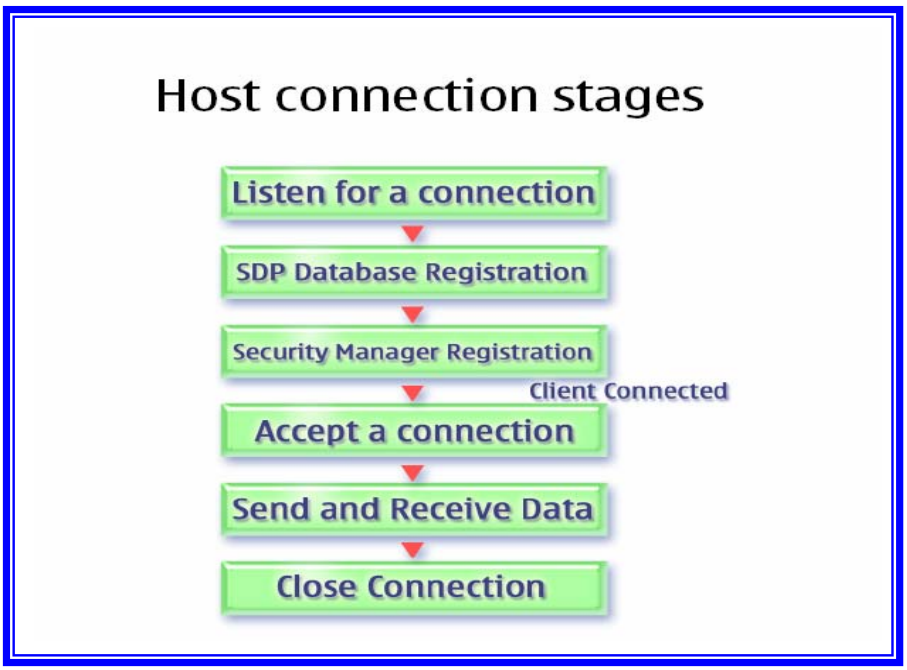


**Figura 24: servidor**

## 6.2.2. **CLIENTE**

El cliente es el encargado de la conexión con otros móviles y el envío de mensajes. Debido a las características de la tecnología bluetooth en los móviles no se puede establecer una conexión permanente Terminal a Terminal, cada vez que queramos enviar un mensaje hay que conecta con el móvil elegido, enviar el mensaje y desconectar.

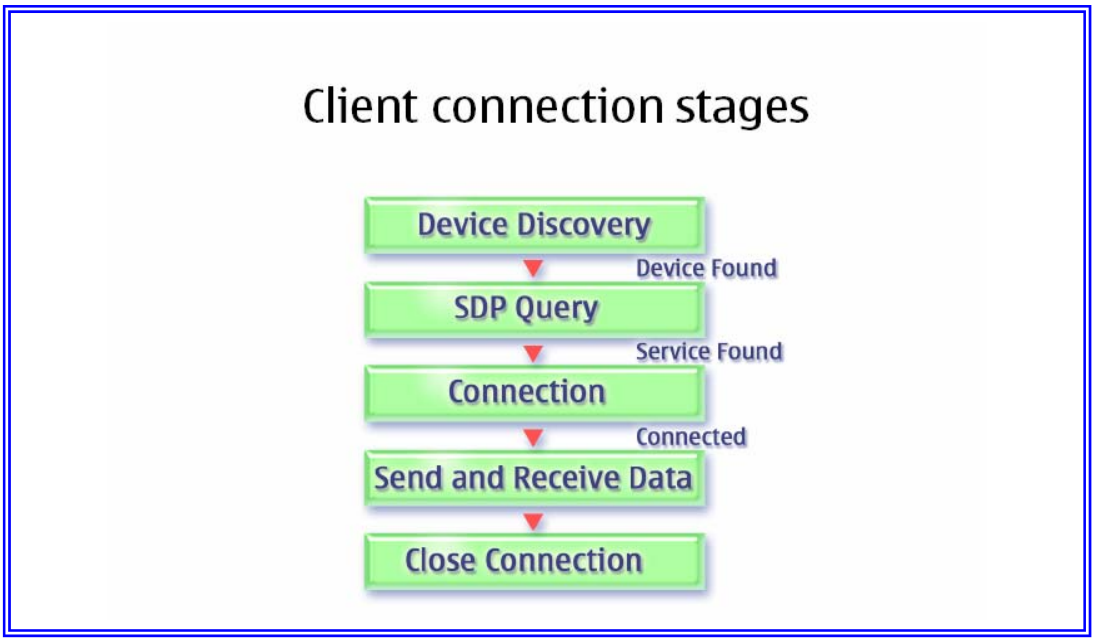


**Figura 25: cliente**

## 6.2.3.INTERFAZ DE USUARIO

La UI presenta al usuario las distintas opciones del Chat, muestra los móviles conectados, los mensajes recibidos y el estado de las distintas operaciones (conectando, buscando terminales, enviando mensaje…).

Las clases encargadas de interactuar con el usuario son:

- **AppUI:** Es la encargada de crear el menú y las barras de estado (baterías, cobertura, fecha y hora…).

- **AppView:** Es la encargada de la parte central de la pantalla. Consta de dos vistas, una muestra los mensajes recibidos y otra nos permite escribir mensajes. Esta clase es la que realmente pone en comunicación al usuario y la AppUI con la estructura interna del cliente servidor.

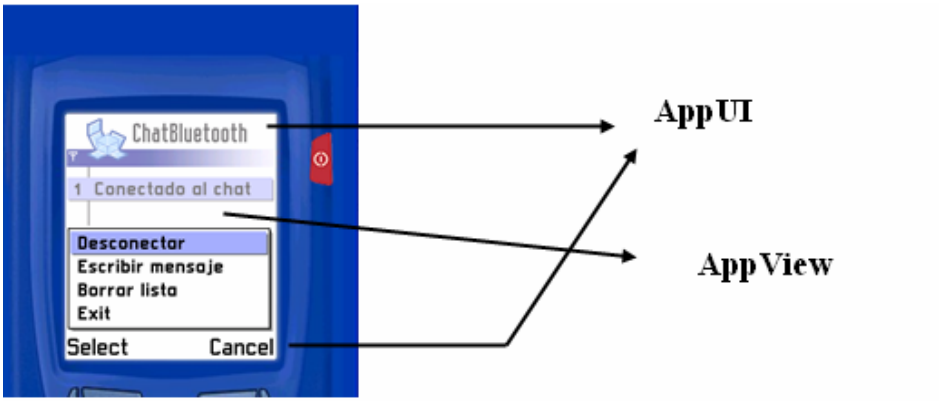


**Figura 26: gráfica de clases**

## 6.3. DIAGRAMA DE ESTADOS Y FUNCIONAMIENTO

Como hemos explicado en el apartado anterior, cada Terminal funciona a la vez como cliente y servidor. Para que un móvil pueda enviar un mensaje a otro tiene que conectar la parte cliente (del móvil que quiere enviar) con la parte servidor (del móvil receptor del mensaje). Lo dos móviles tiene que estar conectador al Chat. La siguiente imagen nos sirve para hacernos una idea del funcionamiento:

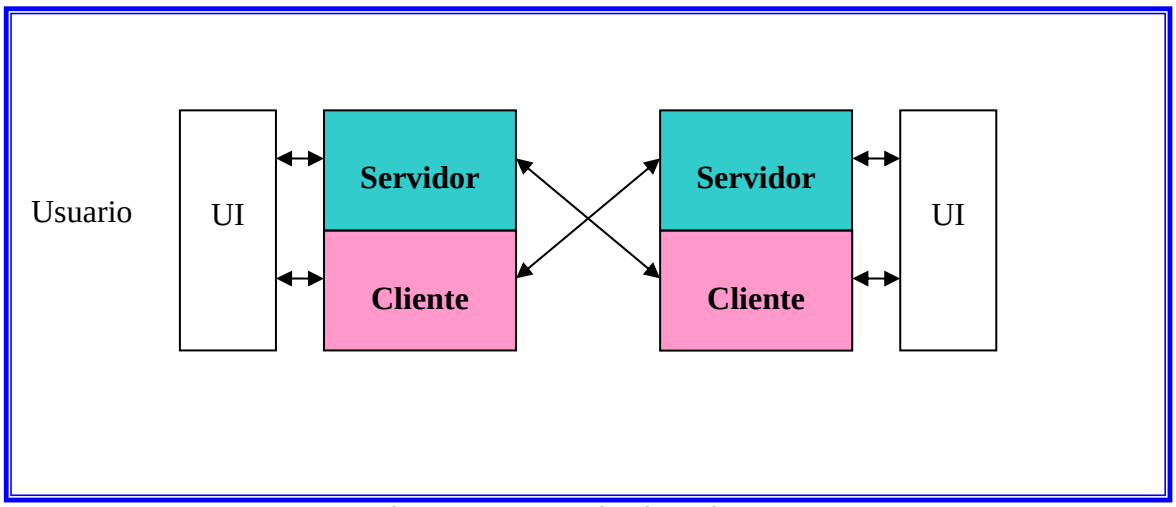


**Figura 27: esquema funcionamiento**

Un Terminal inicialmente esta desconectado. Lo primero que tiene que hacer es abrir nuestra aplicación (inicialmente tiene que haber conectado el bluetooth, esta parte la explicaremos en la configuración en el apartado 5). Elegimos la opción del menú “Conectar” y pasamos a estar conectados y visibles para los demás terminales conectados. Ahora tenemos dos opciones “Desconectar”, volviendo al estado anterior o “Enviar un mensaje”.

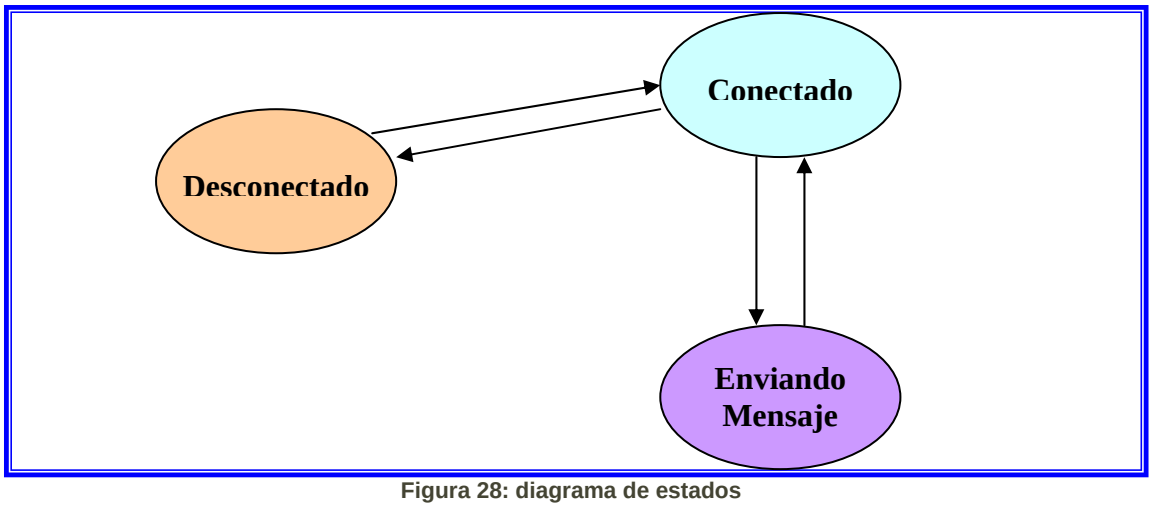


**Figura 28: diagrama de estados**

# 7. FUNCIONAMIENTO DE APLICACIONES EN DISPOSITIVOS REALES

## 7.1. INSTALACIÓN EN DISPOSITIVOS

Para poder instalar aplicaciones desde el ordenador al teléfono móvil previamente debe tener instalado el Nokia Pc Suite, que le habrán dado al adquirir el teléfono o que puede descargar desde el forum Nokia.

Una vez instalado el Pc Suite los pasos a seguir son:

- **1:** Guarde el archivo de instalación en móviles .SIS en una carpeta de su ordenador.
- **2:** Conecte el teléfono al PC (puede usar cable, bluetooth o infrarrojos, siga las instrucciones de su teléfono).
- **3:** Busque el archivo .SIS con el Explorador de Windows y haga doble clic sobre el. Nokia Application Installer instala la aplicación en el teléfono.
- **4:** Ahora debe finalizar la instalación desde el teléfono, que le preguntara si desea instalar la aplicación.

(Los tipos de archivo que puede instalar con Nokia Application Installer son .SIS, .SISX, o .JAD y .JAR.)

## 7.2. CONFIGURACIÓN

Para agilizar el funcionamiento del Chat Bluetooth en su teléfono es recomendable haber vinculado los dispositivos con los que quiere conectar previamente.

- **1:** Entre en Menú->Conectividad->Bluetooth.
- **2:** Asegúrense que el bluetooth este activado y la visibilidad sea “Mostrar a todos”

- **3:** Pulse el botón derecho del menú para acceder a la lista de dispositivos vinculados.

- **4:** Pulse Opciones-> Nuevo Dispositivo Vinculado.

- **5:** El teléfono rastreara todos los móviles bluetooth dentro de su alcance. Selecciones el deseado.

- **6:** Dar al nuevo teléfono vinculado permisos de conexión sin tener que ser aceptado. (Esto no es necesario, pero hace mas ágil el funcionamiento del Chat, ya que si no cada vez que le manden un mensaje le aparece un cuadro de dialogo preguntándole si acepta una conexión de este teléfono.

# 8. INTRODUCCIÓN AL SNIFFER

## 8.1. INTRODUCCIÓN

Un sniffer es un programa informático que registra la información que envían los periféricos, así como la actividad realizada en un determinado equipo, normalmente ordenadores, aunque en nuestro caso queremos desarrollar un sniffer bluetooth para móvil, de manera que registre la información sobre el tráfico de transmisión en bluetooth en una determinada área de una extensión no muy grande.

Para comprender cuáles son los fundamentos de un sniffer y cómo es su funcionamiento, vamos a estudiar en primer lugar un sniffer de ordenador

El sniffing es el "arte" de capturar todos los paquetes que pasan por una red...por ejemplo: en una red que consta de tres ordenadores y ponemos ahí en ejecucion un sniffer, el programa se dedica a captar todo el trafico que pasa por nuestra red, tanto sean peticiones.

Supone una amenaza grave para la seguridad no sólo de una máquina sino también de toda una red. Lamentablemente, una gran cantidad de tráfico confidencial viaja en claro, sin ningún tipo de cifrado, por las redes de la mayoría de las empresas. Ese es el entorno ideal para un sniffer, que puede acceder de forma transparente a esa información, y permitir que alguien abuse de su conocimiento. Por eso es muy importante realizar búsquedas periódicas de sniffers dentro de las redes de cualquier empresa, no sólo por el daño que puedan causar, sino también porque encontrarlos es señal de que se ha producido y explotado una grave brecha y hay que tomar medidas inmediatas.

Existen casos en los que un sniffer no es peligroso. A veces, explorando una red en busca de sniffers se detectará que hay algunos, por ejemplo, en máquinas que dependen del departamento de administración de redes. Esto puede ocurrir porque, en realidad, un sniffer no se diferencia demasiado de una herramienta de monitorización y diagnosis del tráfico de red que puede estar siendo legítimamente utilizada por personal encargado de la administración de la red. Otros dispositivos, especialmente routers y hub, suelen producir falsos positivos que hay que tener en cuenta.

Como acabamos de comentar, el sniffer es una herramienta peligrosa si son utilizados por usuarios no autorizados. Sin embargo, hay pocas herramientas tan poderosas como estas para detectar problemas en nuestra red. Entre los más útiles encontramos a:

- **Tcpdump**: Uno de los sniffers más comunes. Forma parte del sistema base de OpenBSD, está empaquetado para prácticamente todas las distribuciones de Linux y los otros BSDs, y está disponible para cualquier otro sistema Unix. Nos permite trabajar rápidamente desde línea de comando especificando los patrones que nos interesan, puede examinar una gran cantidad de protocolos, puede guardar el flujo capturado en un archivo o tomar un archivo como fuente para el flujo a analizar.

- **darkstat y traffic-vis**: Diseñados para funcionar como proceso demonio, recolectando estadísticas de uso de la red. Ambos reportan sus resultados a través de una interfaz Web, un reporte Postscript u otros formatos.

- **Ngrep:** Tiene una filosofía de uso muy similar a la del comando 'grep' de Unix, tomando como entrada el flujo de la red en vez de archivos locales.

- **Snort:** Muy completa herramienta de detección de intrusos en red, toma como entrada el tráfico capturado en una red y lo va comparando cun una serie de reglas, registrando cualquier tráfico sospechoso de llevar un ataque. Snort únicamente lo registra, pero puede trabajar en conjunto con otras herramientas (hogwash, ACID, etc.) para sanear el tráfico, bloquear a la máquina atacante, generar reportes, etc.

- **Nwatch:** Formalmente es un sniffer, pero es más bien una herramienta para realizar lo que sus autores definen como barridos de puertos pasivos: Para detectar puertos que están abiertos por muy cortos periodos de tiempo y para no mostrar actividad sospechosa de barrido, nwatch se queda escuchando la actividad de la red, y manteniendo una lista de qué hosts proveen qué servicios.

- **Ethereal:** Un magnífico sniffer con interfaz gráfica de usuario, nos brinda un análisis completo y detallado de cada paquete a varios niveles, desde nivel Ethernet hasta detalles de diversos protocolos. Es capaz de convertir en adicción el comprender cómo funcionan determinados protocolos.

- **Ettercap:** Pocas de estas herramientas funcionan adecuadamente en redes switcheadas. Ettercap utiliza técnicas más de sombrero negro, como el ARP spoofing/poisoning, para permitir sniffear redes switheadas. Además de sniffer es interceptor (permite inyectar datos en conexiones existentes o "secuestrar" conexiones).

- **Kismet:** Sniffer específico a Linux para redes inalámbricas. Funciona correctamente con los dos principales tipos de tarjetas inalámbricas.

# 8.2. FUNCIONAMIENTO SNIFFER

Nos centramos en las redes Ethernet para estudiar el funcionamiento de un sniffer por ser la tecnología más importante en materia de conectividad. Las redes tipo Ethernet fueron creadas en los 70s, y su tecnología ha demostrado ser eficiente y extensible, al grado de que, partiendo de un estándar a 1Mbps sobre cable coaxial, hoy tenemos estándares de hasta 1Gbps, y las topologías derivadas de Ethernet ahora engloban desde el medio coaxial hasta topologías de estrella e inclusive inalámbricas.

Las redes Ethernet funcionan basadas en el método CSMA-CD (Carrier Sense Multiple Access - Collision Detection). Esto significa que cada nodo en una red Ethernet tiene la capacidad de detectar si está conectado a una red o no hay un enlace válido (Carrier Sense), y que el mismo medio físico es compartido entre varias computadoras (Multiple Access). Al tener un mismo medio compartido, dos computadoras podrían intentar transmitir datos a la vez, lo que llevaría a que ambos flujos de datos se corrompieran, por lo que se hace necesario que haya una detección de colisiones (Collision Detection) y un mecanismo de respuesta a las colisiones . En caso de haber una colisión, ambas computadoras esperarán un tiempo aleatorio e intentarán re-enviar sus paquetes.

Las redes Ethernet originalmente estaban conformadas por un sólo cable que conectaba, una a una, a todas las computadoras. Aún hoy, con los cambios topológicos que han sufrido, toda red Ethernet emula este comportamiento: cualquier paquete que es enviado a la red llega a todos los nodos de la misma (excepto en las redes switcheadas, de las que hablaremos más tarde). Esto significa que cada computadora de la red tiene la capacidad de escuchar el tráfico dirigido a cualquier otra computadora de la red.

Procesar un paquete que llega por la red siempre supone trabajo para el sistema operativo. Es por ello que las tarjetas Ethernet por defecto no reportan al sistema operativo de paquetes que no estén destinados a esa computadora (dando explícitamente su dirección física o MAC) o a todas las computadoras de la red (enviadas a la dirección física de broadcast, 00:00:00:00:00:00). Para que el sistema operativo reciba todos los paquetes es necesario desactivar este filtro, lo que es conocido como colocar la interfaz en modo promiscuo.

Una vez que la tarjeta está en modo promiscuo, ésta entregará al sistema operativo todos los paquetes que pasen por su cable. Utilizando bibliotecas como libpcap, programas en espacio de usuario pueden solicitar al kernel que les entregue todos estos paquetes para procesarlos y reportar al usuario los datos obtenidos de ellos. Esto es lo que hemos mencionado anteriormente en la introducción como sniffing (olfateo en inglés).

Un segmento de red Ethernet que va creciendo en actividad presenta cada vez más colisiones, y su rendimiento cae de manera abrupta. Como las redes medianas y grandes son cada vez más comunes, a fines de los 90s comenzaron a popularizarse los switches. Equipos de conectividad Ethernet similares a los concentradores que, en vez de enviar cada paquete a todas las computadoras del segmento, los envía únicamente al puerto donde está conectada la computadora destinatario.

Al aparecer los switches, todo parecía indicar que sniffear las redes sería ya imposible, a menos que fuera hecho desde el segmento donde estuvieran las computadoras en cuestión. Tristemente, esta ilusión no duró mucho tiempo, gracias al advenimiento del ARP spoofing/poisoning. Para entrar en detalles, veamos rápidamente cómo funciona el pegamento entre Ethernet y TCP/IP: El protocolo ARP.

Cada tarjeta de red Ethernet tiene un identificador de 48 bits único en el mundo, llamado dirección MAC (Media Access Control). Las direcciones IP son direcciones de 32 bits, y no guardan relación alguna con las direcciones MAC.

Cuando una computadora intenta comunicarse con otra que debe estar (según su dirección/máscara IP) en la misma red que ésta, lanza un paquete ARP (Address Resolution Protocol) de tipo 'who-has', dirigido a todas las computadoras del segmento físico (con la dirección broadcast de Ethernet), con la IP de la máquina destino. A esta solicitud, la computadora dueña de la IP solicitada responde con un nuevo paquete ARP (ya en unicast) a la computadora que originó la solicitud, indicándole su dirección física. Después de esto, ambas conocen ya la relación entre MAC e IP necesaria, y pueden comenzar a enviarse paquetes IP.

Parte del diseño del protocolo ARP estipula que, si una computadora tiene registrada la relación IP <-> MAC de otra en su tabla de ARP y escucha un nuevo paquete ARP anunciando que la IP en cuestión está relacionada con otra ARP, debe olvidar la relación que tenía declarada y registrar la nueva. Por tanto, una computadora cualquiera en la red puede envenenar fácilmente las tablas ARP de las demás, recibiendo los paquetes destinadas a una computadora aún en otro segmento de una red switcheada, e inclusive actuar como proxy, logrando escuchar (e incluso intervenir) de manera completamente transparente la comunicación. Claro está, quien lo esté haciendo tendrá que cuidar el volver a envenenar las tablas ARP cada que haya una solicitud para mantenerse como escucha.

### 8.2.1. SNIFFER MODO PROMISCUO

Como acabamos de estudiar, para un sniffer es fundamental tener la tarjeta de red del equipo en el que está instalado el sniffer en modo promiscuo.

En informática, el modo promiscuo o también conocido como modo monitor, es aquel en el que una computadora conectada a una red compartida, tanto

la basada en cable de cobre como la basada en tecnología inalámbrica, captura todo el tráfico que circula por ella.

En las redes de ordenadores, la información se transmite en una serie de paquetes con la dirección (o dirección MAC) del que lo envía y el que lo tiene que recibir, de manera que cuando transmitimos un fichero, este se divide en varios paquetes con un tamaño predeterminado y el receptor es el único que captura los paquetes leyendo si llevan su dirección.

En el modo promiscuo una maquina intermedia captura todos los paquetes, que normalmente desecharía, incluyendo los paquetes destinados a el mismo y al resto de las maquinas. Dependiendo de las topologías y el hardware que se use para comunicar las redes, influye en su funcionamiento, ya que las redes en bus, redes en anillo, así como todas las redes que obliguen a que un paquete circule por un medio compartido, al cual todos tienen acceso, los modos promiscuos capturaran muchos mas paquetes que si están en una red con topología en árbol. Para completar el modo, las maquinas en modo promiscuo suelen simplemente copiar el paquete y luego volverlo a poner en la red para que llegue a su destinatario real (en el caso de topologías que requieran de retransmisión).

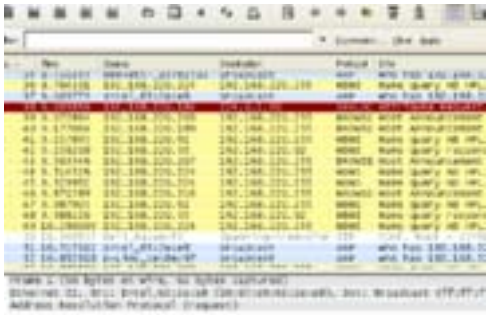

**Figura 29: Ethereal**

El modo promiscuo resulta muy útil para ver que paquetes atraviesan una red. Su utilidad se basa en que todos los paquetes que pasan por una red tiene la información de a que protocolo pertenece y las opciones de reensamblado. Incluso si no están encriptados, tienen la información en claro, es decir, se puede saber que contiene el paquete, todo esto usando un sniffer como hemos visto antes.

Es especialmente útil en los routers que unen varias redes, ya que con herramientas que analizan los paquetes podemos detectar errores, ataques, pérdida de paquetes, sobrecargas, etc... Al capturar todo el tráfico que atraviesa un router, se pueden determinar también, usos, servicios que tienen que recibir ancho de banda prioritario, accesos no permitidos a equipos o protocolos, etc...

También es usado en el lado contrario: para realizar ataques contra redes. Últimamente, este término es muy usado para tratar de atacar redes WIFI encriptadas así como el Wardriving que es la detección de redes WIFI.

Existen herramientas para la detección de interfaces de red que se encuentren en modo promiscuo. Se basan en el envío de paquetes que nadie responderá, salvo por equipos en modo promiscuo.

- Detección de Latencia en paquetes ICMP: Este método lanza muchas peticiones, TCP erróneas para que ningún equipo las tenga en consideración. Tras esto, se manda ping a todas las maquinas. La maquina en modo promiscuo tardará en responder ya que esta ocupando procesando los paquetes.

- Detección mediante paquetes ping ICMP: Se lanza un ping a una maquina sospechosa, con la MAC del paquete errónea. Si la maquina está en modo promiscuo, responderá sin comprobar que la MAC es errónea.

## 8.3. EL SNIFFER Y NUESTRO PROYECTO

Como ya comentamos anteriormente, la idea inicial que teníamos era, una vez realizado un estudio teórico sobre las tecnologías que íbamos a usar, era desarrollar como aplicación práctica un sniffer bluetooth para móviles Nokia de la serie S60.

Una vez realizo la investigación y un estudio sobre la viabilidad de la aplicación nos encontrábamos con que no podíamos realizar una parte esencial en un sniffer, “poner la tarjeta en modo promiscuo”.

Como hemos visto, el modo promiscuo es el fundamento del que parte un sniffer.

Al tratarse Symbian de una tecnología relativamente reciente, es poca la información que se pueda encontrar de este sistema operativo en comparación con otros.

El primer problema que nos encontramos es que la mayor parte de código, subrutinas, información… que ofrece Symbian es para desarrollo de aplicaciones a un “alto nivel”, esto es, aplicaciones que no bajan hasta un nivel de trasporte… como puede ser el de esta aplicación, que necesitamos analizar los paquetes.

Además, como Symbian se trata de un software bajo licencia, todo este código lo tiene oculto al programador ajeno.

Consultando a expertos, nos comentaron que para poder ponerla tarjeta en modo promiscuo, seguramente deberíamos manipular los drivers, cosa que probablemente Nokia no nos daría permiso, por no ser un proyecto interno directo de Nokia.

Por mas que hemos investigado, no hemos encontrado ningún libro, web, memoria, lugar… que tratase de este tema, la creación de un sniffer para bluetooth, ni tampoco ningún proyecto que éste o haya llevado a cabo un desarrollo sobre este tema.

Lo más parecido que hemos visto es el desarrollo de un sniffer bluetooth como aplicación de pc, pero no como aplicación para exportar a un teléfono móvil.

A continuación, si lo hubiésemos conseguido, tendríamos que habernos planteado cuestiones como que tipo de paquetes bluetooth captaría nuestro sniffer, como introducirnos en una conexión establecida entre dos dispositivos que estaban comunicándose entre ellos, cómo mostrar tal cantidad de información un una pantalla de móvil de dimensiones reducidas,… pero al no conseguirlo, fueron cuestiones a las que no buscamos respuesta.

Este es el motivo por el que tuvimos que cambiar de tipo de aplicación, un chat para bluetooth, orientado a otros fines

# 9. CONCLUSIONES

Creemos que la realización de este proyecto es de gran aporte didáctico por la importancia que tiene el bluetooth en dispositivos móviles.

Se trata de un tema de gran interés puesto que los móviles que utilizan Symbian están en pleno auge, y puesto que actualmente no es un campo muy explorado, un buen conocimiento del tema puede abrir bastantes puertas en cuanto al mundo laboral se refiere.

Al tratarse de una tecnología que desconocíamos por completo, nos llevo un gran trabajo en su parte inicial, en la que tuvimos que tuvimos que realizar una ardua tarea de investigación tanto de Symbian como de bluetooth, así como de las herramientas de desarrollo con las que íbamos a trabajar y llevar a cabo nuestras aplicaciones.

También nos hemos encontrado con una serie de dificultades a lo largo de estos meses. Entre ellos el más importante, como hemos comentado ya, en el proceso de creación del sniffer, poner la tarjeta de bluetooth en modo promiscuo. Buscamos bastante información sobre el tema, incluso llegó a pararnos momentáneamente el desarrollo en busca de una solución. Después de consultar a especialistas y comentarnos las dificultades que esto conllevaba y el tiempo que nos llevaría seguir buscando nuevas soluciones, decidimos orientar nuestro trabajo práctico hacia otro tipo de aplicaciones, el chat para bluetooth.

Otra de las dificultades que nos encontramos fue cómo simular las comunicaciones bluetooth entre dos dispositivos. Los emuladores que nos proporcionan los SDK de Symbian con los que trabajamos permitían emular el comportamiento de un dispositivo, pero era el comportamiento de una aplicación que corría en este dispositivo sin comunicarse con otro, y mucho menos a través de bluetooth. Esto quedó resuelto cuando investigamos y encontramos el NCF, que permite simular la comunicación entren estos dispositivos.

Creemos que hemos cumplido satisfactoriamente con los objetivos que nos marcamos al inicio del proyecto. Aún no habiendo podido desarrollar por completo el sniffer, pensamos que hemos cubierto el estudio tanto de manera teórica como de manera práctica (con la realización del bluechat) el estudio de la creación de aplicaciones bluetooth para móviles.

Estamos contentos con el trabajo realizado, con lo que hemos aprendido y con la utilidad que éste tiene. Sigue abierta la línea de investigación del sniffer, para una futura ampliación de este proyecto, que sería más fácil de enfrentar, pues poseemos un conocimiento básico suficiente sobre las tecnologías Symbian y bluetooth, de las que empezamos el estudio partiendo de cero, algo que ya no tendríamos que hacer en un futuro

Sin más, esperamos que este documento sea de utilidad a todo aquel que lo consulte para su propio trabajo, del mismo modo que deseamos que este tema les produzca el mismo interés que nos ha producido a nosotros.

# 10. **BIBLIOGRAFÍA**

Para el desarrollo de este proyecto nos ha sido de mucha utilidad el siguiente material, del cual hemos extraído bastante información, y queremos recomendar y felicitar los siguientes documentos:


- http://www.bluetooth.com/bluetooth/ para el estudio de la arquitectura y seguridad de bluetooth.


También mencionar las siguientes memorias de proyectos de fin de carrera de la universidad Carlos III de Madrid, de las que hemos obtenido gran información acerca de Bluetooth, de Symbian y de Carbide, de lectura recomendada para profundizar en estos temas:


- "Análisis de seguridad en dispositivos móviles con sistema Symbian", Ángel García Moreno, 6 de Octubre de 2006

- "Seguridad en Symbian: Entorno de Determinación de la Integridad Adaptable(EDIA)", Jaime Tejeda de Hoyos, Junio de 2006


Por último, el material proporcionado por Nokia, de gran interés, entre los que destacamos:


- Course Number 04300, "Symbian OS Basics", developer training, Nokia 2004

- Course Number 05300, "S60 C++ Introduction", developer training, Nokia 2006

- Course Number 06300, "S60 3rd Edition Update", developer training, Nokia 2006