A mis Padres, Hermana y Amigos,
por su amor y apoyo.
Sebastián

A mi Padre, Madre, Hermana,
a mi pequeña sobrina
y a toda la gente que me ha apoyado.
Roberto

Índice

Índice de Ilustraciones

Índice de Tablas

Resumen

El “Sistema de acceso a servicios desde Cliente Móvil mediante Bluetooth” es un proyecto que busca desarrollar un sistema de software capaz de generar un canal bi-direccional que permita a los usuarios de clientes móviles acceder a servicios de distinta índole, como titulares de prensa, información deportiva, revisar su correo electrónico o el menú del café donde se encuentra, dependiendo de su contexto espacial. Utilizando para esto el Estándar de comunicaciones Bluetooth, lo que permitiría la iluminación de espacios públicos en que los usuarios podrían acceder a esta información de manera fácil y gratuita, directamente en su teléfono móvil.

Abstract

The “Service Access System from Mobile Client via Bluetooth” is a project that aims to develop a system capable to generate a bi-directional channel that allow mobile users to access different kind of services depending of it’s spatial context e.g. press news, sports info, the Coffee shop menu, and so on. Using Bluetooth to “illuminate” public spaces, allowing the users to access this info easy and freely directly to his/her mobile phone.

Capítulo 1 Introducción

El presente documento tiene como propósito referirse al trabajo realizado en el marco del proyecto llamado “Sistema de acceso a servicios desde cliente móvil mediante Bluetooth”, aplicado a una Red de Servicios de Cafetería.

La característica principal del proyecto es la construcción de un sistema de software capaz de generar una interacción entre un emisor Bluetooth[1] (Nodo Bluetooth) y un cliente móvil como un teléfono celular o una Palm, ofreciendo servicios de distinta índole dependiendo de la ubicación espacial de cada Nodo. Los servicios ofrecidos pueden ir desde la lectura de titulares de diarios on-line hasta la publicidad de alguna marca en específico.

Para analizar esta problemática es necesario mencionar sus causas. La más importante de ellas es el “Boom” de la tecnología móvil. Desde el momento en que surgió la telefonía móvil, que vino a solucionar el problema de las comunicaciones a distancia durante la segunda guerra mundial, la tecnología móvil ha evolucionado hasta formar parte casi imprescindible del quehacer diario del ser humano, las comunicaciones a través de medios inalámbricos ha crecido de manera explosiva, tanto que solo en Chile la estadística es impresionante, casi 8 de cada 10 chilenos posee teléfono celular, lo que, para un País del tercer mundo, es todo un record. Por lo anterior es que las empresas de tecnología móvil ofrecen cada vez nuevos y mejores servicios para este tipo de aparatos. Esto abre un gran y poco explotado nicho para el desarrollo de aplicaciones en este contexto, lo que motiva fuertemente a abordar este proyecto. Otra de las motivaciones pasa por profundizar el conocimiento en lo que a desarrollo de este tipo de proyectos respecta, queriendo agregar también, un aspecto innovador, en cuanto a la utilización del protocolo de comunicaciones Bluetooth y de herramientas Open Source, lo anterior con el afán de plantear una solución óptima para la problemática.

En el marco teórico metodológico se basó en los modelos definidos por la Ingeniería de Software para el desarrollo de proyectos de este tipo. Adecuándose a las características propias del proyecto y generando las instancias para el apropiado desarrollo de este.

Sin duda lo anteriormente mencionado motiva a caminar por el sendero de la investigación e innovación en tecnologías móviles, enfrentándonos a un gran desafío en el desarrollo de software, pero con mucha probabilidad de éxito gracias a la masificación que las tecnologías móviles ha demostrado, sobre todo en estos últimos años.

[1] Véase Anexo A

Capítulo 2 Objetivos

2.1 Objetivo General

El objetivo general del proyecto es desarrollar un sistema de software con la lógica necesaria para permitir acceso, desde un cliente móvil, a servicios ofrecidos dependiendo del contexto espacial donde este se encuentre.

2.2 Objetivos Específicos

- Estudiar las tecnologías inalámbricas existentes.
- La creación de mecanismos de seguridad que resguarden el flujo de información entre los Servidores y los Dispositivos Móviles.
- Facilitar el acceso de los usuarios a servicios que puedan ser de su interés, a través de medios electrónicos móviles.
- El diseño de una Arquitectura que permita llevar a cabo el proyecto.
- La implementación de un prototipo funcional del Sistema de software para la validación del proyecto.

Capítulo 3 Estado del Arte

3.1 Introducción

Este análisis del Estado del Arte se centra en la idea de poder ofrecer una gama de servicios que pudiesen diferir, dependiendo del contexto espacial (lugar físico) en que el usuario del cliente móvil se encuentre, considerando las características que los Teléfonos Móviles poseen en la actualidad.

3.2 Competencia

En la Tabla 3.1 se analizan las competencias a Bluetooth, tomando en cuenta tecnologías que actualmente pueden poseer los Teléfonos Móviles. Este análisis compara la situación actual que pudo ser observada en el mercado local y considerando factores como el costo, performance y disponibilidad.

Tabla 3.1 - Competencia

Bluetooth V/S	**Ventajas**	**Desventajas**	**Evolución**
WAP	Acceso a múltiples contenidos Cobertura bajo red celular	Caro - Lento No existe oferta de contenido contextual	Se espera que los costos disminuyan, lo que hará más frecuente su uso en el mercado.
GPRS	Cobertura bajo red celular Estándar ampliamente utilizado	Requiere de aplicación para generar acción en ambas direcciones. Requiere de georeferencia para brindar soporte contextual No existe oferta de contenido contextual.	Estándar como sistema de transmisión de datos en todo el mundo.
WIFI	Existen nodos de cobertura de acceso público Simple desarrollo de aplicaciones contextuales.	Son muy pocos los celulares que disponen de esta tecnología. Alto costo de los equipos.	No es prioridad para las compañías de telefonía móvil. Aumentará el uso de esta tecnología

3.3 Sustitutos

3.3.1 Marketing de Proximidad - Bluetooth

Es una nueva estrategia por la que se envían mensajes publicitarios a los móviles a través de Bluetooth. El sistema es sencillo, aunque son muchos los que empiezan a plantearse hasta qué punto se trata de una invasión de la intimidad.

El Marketing de Proximidad basado en Bluetooth hace uso fundamentalmente del Perfil de Carga de Objetos (OBEX Object Push) disponible en la mayoría de teléfonos móviles y smart phones. Por lo general, el acceso a este Perfil requiere autorización pero no autenticación. Esto significa que no es necesario que los dispositivos se encuentren emparejados, simplemente basta que el usuario del dispositivo destino autorice el envío de archivos del dispositivo origen.

El funcionamiento de un HotSpot Bluetooth capaz de enviar spam es bastante sencillo. Su tarea es descubrir otros dispositivos Bluetooth activos en su radio de cobertura y proceder al envío de objetos de información con conexiones a través del Perfil de Carga de Objetos (OBEX Object Push).

Los objetos de información enviados por el HotSpot Bluetooth pueden ser de diversa naturaleza y adaptables en función del modelo de teléfono móvil que vaya a recibir el archivo. Así, el objeto a enviar puede tratarse de un archivo de texto, una imagen, un archivo de audio, un video o incluso un paquete instalable con contenidos de publicidad, como un tema para la interfaz o un salva pantallas. El nivel de sofisticación del HotSpot para personalizar contenidos al teléfono móvil destino dependerá de su capacidad para identificar el modelo de dispositivo y conocer los tipos de archivos soportados por su sistema operativo. OBEX es soportado por casi todos los teléfonos de última generación, las PDA. Hay modelos de marketing de proximidad basados en GSM (con envío de SMS a los terminales que estén en una zona), pero requieren que lo haga la operadora.

3.4 Factores de Innovación

Los factores de innovación de mayor relevancia que se presentan en el proyecto son la incorporación de tecnologías Open Source, del estándar de Comunicaciones Bluetooth y de la posibilidad de Interacción entre el cliente móvil y el sistema. Esto seguiría la línea de las políticas de apoyo a los proyectos de innovación en el área de las TIC´s impulsadas por el Gobierno de Chile.

3.4.1 Open Source

El desarrollar el sistema utilizando solo tecnologías Open Source y/o GPL otorga al proyecto un gran matiz de innovación, pues agrega un factor de riesgo asociado, dado que podría acontecer alguna situación que retrase su ejecución u obligue a reformular alguno de los aspectos del proyecto.

3.4.2 Bluetooth

La incorporación del estándar de comunicaciones Bluetooth provee al proyecto de una importancia muy interesante, pues en Chile no ha sido muy investigado el tema, esto entonces, incentivaría y promovería el desarrollo de este tipo de software en el País, aumentando así el factor innovador de este.

3.4.3 Interacción

Al realizar este estudio del Estado del Arte solo se encontraron productos que desarrollan marketing mediante Bluetooth, pero ninguno que posea de una interacción entre el cliente móvil y el Nodo que le provee de la cobertura Bluetooth, en este sentido proyecto posee un factor innovador considerable, pues si se pudiese rescatar información útil, la gestión del marketing podría ser optimizada.

Capítulo 4 Proceso de Desarrollo

La ingeniería de software está compuesta por una serie de modelos que abarcan los métodos, las herramientas y los procedimientos. Estos modelos se denominan frecuentemente modelos de desarrollo de la ingeniería del software y la elección de una metodología se realiza básicamente de acuerdo al tipo del proyecto y de la aplicación, veremos a continuación que modelo de proceso se escogió, las ventajas y desventajas en comparación con las metodologías más usuales a fin de justificar la decisión tomada.

4.1 Proceso Unificado de Desarrollo de Software (UP).

Un proceso de desarrollo de software es un conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema informático. El proceso unificado es más que un simple proceso, es un marco de trabajo genérico que se puede especializar para una gran variedad de sistemas, para las distintas áreas de aplicación, diferentes tipos de organización, diferentes niveles de aptitud y diferentes tamaños de proyecto.

El Proceso Unificado está basado en componentes, lo cual quiere decir que el sistema en construcción está formado por componentes de software intercomunicados a través de interfaces bien definidas.

El Proceso Unificado utiliza el Lenguaje Unificado de Modelado (UML), para realizar todos los esquemas de un sistema, siendo éste una parte esencial del paradigma.

Los verdaderos aspectos definitorios del Proceso Unificado se resumen en tres frases claves: dirigido por casos de uso, centrado en la arquitectura, e iterativo e incremental.

- Dirigido por casos de uso: un caso de uso es un fragmento de funcionalidad del sistema que proporciona al usuario un resultado importante. Los casos de uso representan los requisitos funcionales, todos los casos de uso juntos constituyen el modelo de caso de uso el cual describe la funcionalidad total del sistema.
- Centrado en la arquitectura: la arquitectura de un sistema de software se describe mediante diferentes vistas del sistema en construcción que permite tener un detalle completo del sistema antes de la construcción.
- Iterativo e incremental: es práctico dividir el trabajo de un proyecto en partes pequeñas o mini proyectos, cada mini proyecto corresponde a una iteración que resulta en un incremento. Las iteraciones hacen referencia a pasos en el flujo de trabajo, y los incrementos al crecimiento del producto. Para una efectividad máxima se deben controlar las iteraciones es decir se deben seleccionar y ejecutar de manera controlada.

Debido a esto, el Proceso Unificado otorga ciertos beneficios al desarrollo de los sistemas:

- La iteración controlada reduce el coste de riesgo a los costes de un sólo incremento. Si los desarrolladores tienen que repetir la iteración solo se pierde el esfuerzo mal empleado de la iteración no el del desarrollo completo.
- Reduce el riesgo de no cumplir con los compromisos adquiridos dentro de un período determinado. Esto se logra gracias a la identificación de los riesgos en fases tempranas de desarrollo.
- Acelera el ritmo total del esfuerzo del desarrollo, ya que se trabaja para conseguir resultados claros a corto plazo, con lo cual existen altas posibilidades de cumplir con los calendarios de entrega previstos.
- La iteración controlada reconoce una realidad que a menudo se ignora y es que las necesidades de los usuarios y los requisitos no se pueden definir completamente al principio, estos se van refinando a medida que se avanza en las iteraciones, haciendo más fácil la adaptación a los requisitos cambiantes.

El Proceso unificado se repite a lo largo de una serie de ciclos que constituyen la vida de un sistema, cada ciclo de un sistema se compone de cuatro fases: inicio, elaboración, construcción y transición.

- Fase de inicio: El objetivo de la fase de inicio es desarrollar un análisis de negocio hasta el punto necesario para justificar la puesta en marcha del proyecto. Esta fase también debe de señalar las principales funciones del sistema para los usuarios, como debe de ser la arquitectura, y la planificación del proyecto y el costo de desarrollo del mismo.
- Fase de elaboración: Se especifican los casos de uso y se diseña la arquitectura del sistema a modo de vistas de todos los modelos del sistema.
- Fase de Construcción: En esta se crea el producto, en donde la descripción inicial evoluciona en un producto preparado para ser otorgado a los usuarios. Al final de esta fase, el producto contiene todo los casos de uso que se han acordado en el desarrollo.
- Fase de Transición: Se cubre el período durante el cual el producto se convierte en versión beta. Esta versión beta es la etapa en donde un grupo de usuarios experimentados prueba el producto e informa de las deficiencias y eventuales errores que tenga el sistema. De esta manera los desarrolladores pueden corregir los problemas y mejorar las versiones del producto. Esta fase conlleva actividades como la fabricación, formación del cliente, entrega de ayuda y asistencia, y la corrección de errores que se encuentren una tras la entrega.

Cada una de estas fases se subdivide a su vez en otras tareas, Requisitos, Análisis, Diseño, Implementación y Prueba. Donde cada uno de los ciclos genera una nueva versión del sistema, siendo estas un producto listo para la entrega. Constan de códigos fuentes, manuales y otros productos asociados. Sin embargo estos productos se deben ajustar no sólo a los requerimientos del cliente sino que también a la de la gente que va a trabajar con él.

4.2 Modelo de Proceso Escogido

Luego de analizar las ventajas y desventajas de las metodologías más comunes, se opta por usar el modelo de Proceso Unificado de Desarrollo de Software, el cual posee la ventaja de ser un desarrollo evolutivo, con el plus de soportar la construcción de prototipos.

Además el Proceso Unificado de Desarrollo de Software al ser evolutivo e incremental permite refinar los requerimientos hechos por el cliente, como así también generar "mini proyectos" los cuales serán refinados a lo largo del desarrollo del proyecto, dependiendo de los nuevos requisitos dados por el cliente, como los cambios generados por la propia evolución del sistema. Todo esto ayuda a tener un proceso de desarrollo más seguro y planificado. Con respecto a lo anteriormente mencionado, quizás el factor más preponderante a la hora de seleccionar el modelo de proceso fue el riesgo, un componente de gran impacto en este proyecto en particular, pues se aparta de lo que son los desarrollos tradicionales de sistemas de información en que el factor riesgo no va en directa relación con la viabilidad del proyecto, sino más bien con los tiempos en que este se puede completar.

4.3 Paradigma de Programación

Los principales paradigmas en el ámbito de la programación son las correspondientes a Estructurado Clásico, Estructurado Moderno y Orientación a Objetos. Sin embargo, en el punto anterior se decidió que el Modelo de Proceso escogido era el Proceso Unificado de Desarrollo, lo que conlleva por definición a la elección de la metodología Orientación a Objeto, la cual se revisa a continuación.

4.3.1 Orientación a Objetos

Una de las diferencias principales entre los paradigmas tradicionales y el paradigma Orientados a Objetos es que los procedimientos tradicionales estaban limitados al desarrollo de sistemas convencionales de procesamiento. Por otra parte, el paradigma Orientados a Objetos pueden utilizarse para desarrollar cualquier tipo de sistema.

El Análisis Orientado a Objetos maneja métodos que permiten al ingeniero de software modelar un problema a través de la representación de objetos, atributos y operaciones como las componentes primarias del modelado. Una amplia variedad de métodos de análisis orientado a objetos han sido propuestos, pero todos poseen un conjunto de características posibles:

- Representación de clases o jerarquías de clases.
- Creación de modelos objeto-relación.
- Derivación de modelos objeto-comportamiento.

El análisis de sistemas orientados a objetos se considera la primera fase de la mayor parte de los proyectos de sistemas. Se enfoca en la comprensión del negocio en función de sus actividades, reglas, localizaciones e información. El análisis va más allá del simple estudio de información de manera que pueda incluirse algo que resulte de interés para el negocio. El análisis identifica tipos de objetos, tipos de eventos y reglas de los negocios, y emplea técnicas de utilización de análisis. Estas resultan cruciales para el desarrollo

correcto de sistemas automatizados. El empleo de objetos y de eventos proporciona un modelo del negocio que es más cercano a la forma en que comprendemos el mundo.

El proceso de OO comienza con la definición de los casos de uso, escenarios que describen como se debe usar el sistema OO. Se aplica entonces la técnica de modelado clase-responsabilidad-colaborador (CRC) a clases documentos, a sus atributos y operaciones. También aporta una vista inicial de las colaboraciones que ocurren entre los objetos. La etapa siguiente en el OO es la clasificación de los objetos y la creación de una jerarquía de clases. Pueden usarse subsistemas (temas) para encapsular objetos relacionados. El modelo objeto-relación proporciona una indicación acerca de cómo están interconectadas unas clases con otras, y el modelo objeto-comportamiento indica el comportamiento de objetos individuales y el comportamiento global del sistema OO.

Para el desarrollo de ésta se consideran fundamentalmente los siguientes pasos:

- Identificar los objetos considerando que los depósitos de datos que aparecen en los Diagramas de Flujo de Datos.
- Identificar las operaciones asociadas a cada objeto, para lo cual se realizan los procesos y se determina a qué objetos potenciales la asociación resulta más natural. Para ello es necesario considerar que:
    - Algunos candidatos a objetos no tienen operaciones que los acrediten como susceptibles de convertirse en objetos.
    - Algunos procesos presentan correspondencia con más de un objeto, caso en el cual es necesario descomponer esos procesos.
    - Habrá procesos sin candidatos a objetos. En este caso, se crea un objeto para cada uno de estos procesos.
- Representar objetos identificados, operaciones asociadas a entidades externas, y ajustar los diagramas de flujo de datos a las modificaciones introducidas.
- Definir las interfaces de los objetos.
- Realizar una evaluación final.

La elección del paradigma debe ser acorde al proyecto a desarrollar, así como también a su naturaleza. El paradigma de orientación a objetos facilita la tarea de poder identificar los elementos que componen un sistema reduciendo las distancias entre las actividades de análisis, puesto que trata los atributos y métodos de los elementos del sistema como un todo (integración), esto reduce los riesgos relativos al desarrollo de sistemas complejos. Además, este paradigma, permite al analista y al cliente poder tener una mejor comunicación ya que esta tiene herramientas de representación muy comprensibles para ambas partes.

Cuando en un sistema que ocupa una metodología tradicional se produce un cambio, generalmente éste afecta a gran parte del sistema. El Análisis de Orientación a Objetos permite crear especificaciones que sean capaces de tolerar los cambios, ya que esta metodología hace que las estructuras de dominio del sistema sean dinámicas, otorgando la estabilidad suficiente en caso de cambios de requisitos de sistemas similares. Por lo tanto podemos decir que Análisis de Orientación a Objetos produce sistemas que son flexibles a

los cambios. La reutilización de los componentes, del diseño y de las aplicaciones creadas hacen que el trabajo de los desarrolladores se simplifique al momento de crear nuevos componentes y aplicaciones para un sistema diferente, esta es una característica esencial de esta metodología.

Capítulo 5 Desarrollo

5.1 Modelado del Negocio

5.1.1 Propuesta de Solución

Para proponer una Solución a los requerimientos del proyecto, se analizó la problemática planteada, crear un Sistema que tenga la capacidad de poder ofrecer servicios de distinta índole, para un cliente móvil, dependiendo del contexto espacial en que este se encuentre. Por ejemplo: noticias, información deportiva, el menú del café donde se ubica el sistema o contestar una encuesta. Los usuarios de teléfonos móviles deben poder acceder a una serie de servicios ofrecidos e interactuar con ellos en las distintas áreas de cobertura. Los servicios ofrecidos responderán a las necesidades propias del contexto y del entorno en que el cliente móvil se ubica.

Las áreas de cobertura en el sistema se denominan Nodos. Cada Nodo de la red dispondrá de un dispositivo que brinda cobertura de señal Bluetooth, para un número determinado de teléfonos móviles, y así los usuarios de cliente móvil que estén en dicho lugar podrán acceder a los servicios de información disponibles, pudiendo también interactuar con estos como se muestra en la Ilustración 5-1.

Un sistema de administración ofrecerá la funcionalidad gestionar lo relacionado a configuración del sistema y sus servicios.

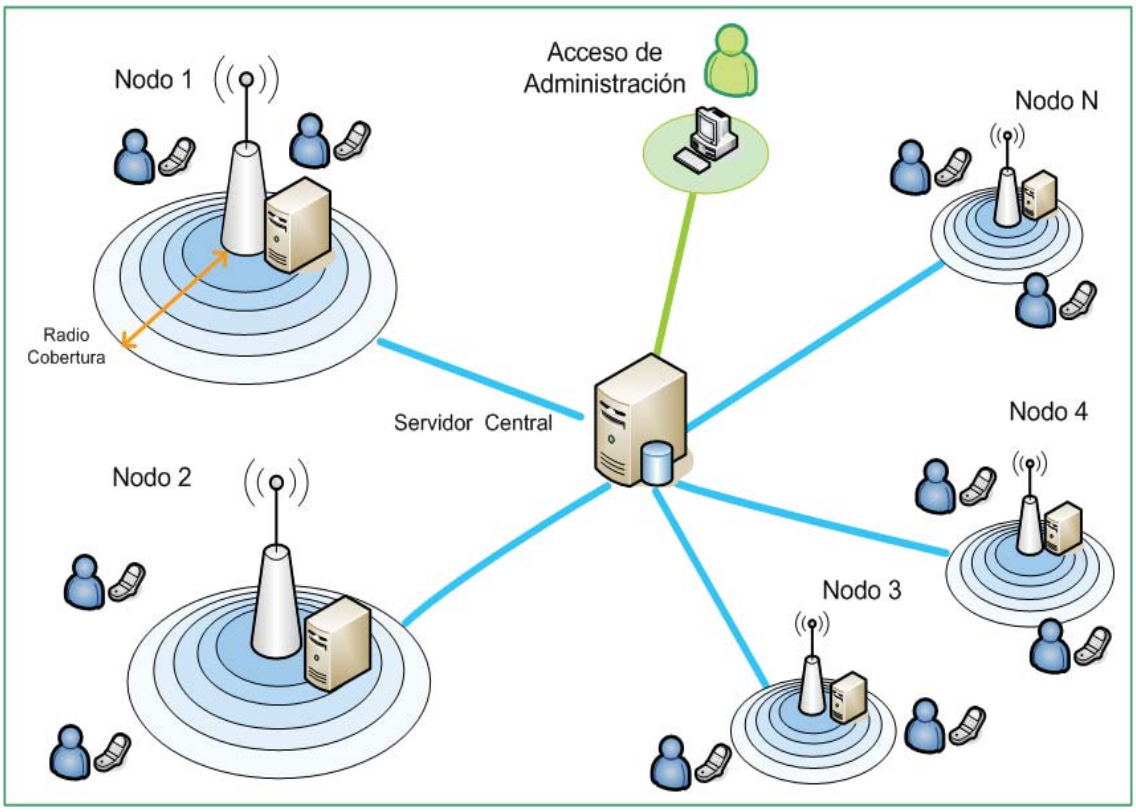

Ilustración 5-1 - Propuesta de Solución

Para cada Nodo en particular existe entonces la posibilidad de "customizar" el contenido de servicios ofrecidos, es decir, si por ejemplo un Nodo se encuentra en una ciudad determinada, podría ofrecer servicios que fueran de interés para esa ciudad en particular, si perjuicio de lo cual, otro Nodo ubicado en otro sector geográfico pudiera ofrecer servicios vinculados a su ambiente específico. Entre los dominios de aplicación donde este proyecto pudiese ser potencialmente implantado están, por ejemplo:

- Hoteles: Información de interés para los huéspedes directamente al móvil a través de Bluetooth, qué visitar, qué hacer, donde comer o donde comprar.
- Museos: Guiar mediante el móvil del visitante a través de Bluetooth con información sobre las áreas y obras expuestas en el museo.
- Centros Comerciales: Ofertas y promociones de las distintas tiendas de un recinto comercial al móvil a través de Bluetooth de forma dinámica según la planta del edificio.
- Aeropuertos: Información de interés turístico de la ciudad como hoteles, restaurantes, exposiciones o espectáculos directamente al móvil a través de Bluetooth.
- Restaurantes: Información sobre la carta así como sugerencias del chef y entretenimiento directamente al móvil de los clientes a través de Bluetooth.

El Dominio de aplicación escogido es el de una Red de Cafeterías, como Segafredo, pudiendo así, asociar cada Nodo del Sistema a un Local de Café en particular.

5.1.2 Estudio de Factibilidad

Un estudio de factibilidad trata de determinar si el proyecto es útil y viable para la empresa u organización. Factibilidad se refiere a la disponibilidad de los recursos necesarios para llevar acabo los objetivos o metas señaladas, todo esto se apoya en 4 aspectos básicos: técnico, operativo, legal, y económico.

5.1.2.1 Factibilidad Técnica

Un análisis de factibilidad técnica evalúa si el equipo o los equipos y software están disponibles, o si realmente el software se puede desarrollar, si tienen las capacidades técnicas requeridas por cada alternativa del diseño que se esté considerando, a la vez también considera las interfaces entre los sistemas actuales y nuevos. De acuerdo a las necesidades técnicas del proyecto y analizando cada uno de los dispositivos que intervienen en el sistema, se concluyó como requisitos mínimos para el correcto funcionamiento:

Cliente Móvil:

- Soporte para el Lenguaje de Programación Java (J2ME).
- Contar con el Perfil MIDP 2.0 o superior.
- Contar con la Configuración CLDC 1.1 o superior.
- Contar con el Paquete Opcional Bluetooth JSR-82.

Se consideró el contar con la API JSR-82 como atributo estrictamente necesario pues sin esta característica el software no puede ser instalado en el Cliente Móvil.

Nodo:

- Computador de escritorio Pentium III + 256 RAM o superior
- Sistema Operativo Debian GNU/Linux 4.0 (Etch) Kernel 2.6.18
- Soporte para el Lenguaje de Programación Java (J2SE)
- Dispositivo Bluetooth USB estándar 2.0 Clase 2
- Pila de Protocolo Bluetooth BlueZ
- API Java para manejo Bluetooth aventanaBT

Estas características fueron definidas tomando en cuenta que es necesario contar con soporte para Java, además contar con la API aventana Bluetooth y con la Pila de protocolo BlueZ ejecutándose al mismo tiempo, requiriéndose versiones específicas desarrolladas para esa versión del Kernel de Linux.

Servidor Central:

- Computador de escritorio Pentium IV + 512 RAM o superior
- Sistema Operativo Debian GNU/Linux 4.0 (Etch)
- Soporte para el Lenguaje de Programación Java (J2SE)
- Servidor Web Apache + Tomcat 5.5. + Axis
- Servidor de Base de Datos XML (eXist)

Las características del Servidor se definieron tomando en cuenta la conjunción de distintos software ejecutándose al mismo tiempo como Tomcat, Axis, Apache, eXist, etc. Por lo que se requiere de las cualidades antes mencionadas.

5.1.2.2 Factibilidad Operacional

El Análisis de la factibilidad operacional radica en determinar la capacidad potencial de la organización para llevar a cabo el proyecto en término de los planes, políticas y procedimientos vigentes, es decir, se trata de reconocer a qué se expone la organización al añadir un nuevo sistema, cual será la reacción que producirá en los demás procesos, en los recursos humanos y en general en toda la organización, considerando incluso los clientes.

Tomando en cuenta la naturaleza del proyecto se podría determinar que la empresa que hace el requerimiento, puede perfectamente llevar a cabo el proyecto en los términos y plazos planteados, pues la creación de este sistema tendría repercusiones favorables en los procesos llevados a cabo por la empresa.

5.1.2.3 Factibilidad Legal

De acuerdo con este punto del análisis la implementación del sistema en una entidad, no representaría problemas dado la naturaleza del sistema y al uso total de herramientas libres (Open Source) que cuentan con licencias como GPL, que permiten y protegen la libre modificación, distribución y uso de los software que cuenta con este tipo de licencias, por lo tanto el uso de las herramientas que se hace utilización para el desarrollo e implementación de este software no violaría ningún marco legal vigente en el ámbito nacional en lo que corresponde a leyes informáticas. También es importante considerar las políticas internas de la organización, tratando de afectar de la menor forma posible el accionar o el desarrollo interno de la empresa y que al utilizar el sistema incurra

en algún delito, en este caso en particular se deben verificar las licencias del software actuales. Por otra parte se puede considerar que el estándar de comunicaciones Bluetooth puede ser utilizado sin problemas legales pues trabaja en el rango de 2,4 a 2,48 GHz, rango reservado para móviles por la Subsecretaría de Telecomunicaciones del Gobierno de Chile, como se muestra en http://www.subtel.cl/prontus_subtel/site/artic/20070102/asocfile/20070102155828/diagrama_dec15_agosto_05.PDF

5.1.2.4 Factibilidad Económica

En lo que respecta a la factibilidad económica es importante resaltar lo comentado en el punto anterior, dado a que están directamente relacionados, pues la utilización de herramientas Open Source hace que los costos se disminuyan al mínimo en lo que respecta a la instalación e implementación del software y herramientas necesarias para el correcto funcionamiento de esta aplicación. Esto hace suponer que donde se implemente la solución conste con los requisitos mínimos para su puesta en marcha explicados en el punto anteriormente mencionado.

El costo incurrido en el desarrollo del sistema es casi nulo en lo que respecta al pago de licencias y herramientas. Por lo tanto se concluye que el sistema es económicamente factible de realizar.

5.1.3 Análisis de Riesgo

El análisis de riesgo (también conocido como evaluación de riesgo o PHA por sus siglas en inglés: Process Hazards Analysis) es el estudio de las causas de las posibles amenazas y, los daños y consecuencias que éstas puedan producir.

Este tipo de análisis es ampliamente utilizado como herramienta de gestión, en estudios financieros y de seguridad para identificar riesgos (métodos cualitativos) y otras para evaluar riesgos (generalmente de naturaleza cuantitativa).

A continuación se explican los riesgos identificados en el desarrollo del sistema, su calificación y método de mitigación. Lo anterior considerando el riesgo como un factor muy importante en el desarrollo del proyecto.

5.1.3.1 Identificación de riesgos

La Tabla 5.1 muestra los potenciales riesgos identificados dentro del desarrollo del sistema de software, estos riesgos se tipifican y explican.

Tabla 5.1 - Identificación de riesgos

Nombre	Tipo	Detalle
No disponibilidad de hardware	De hardware	Hardware, el cual es esencial para el proyecto, no será entregado dentro de lo planificado.
Heterogeneidad de los dispositivos móviles	De hardware	Diferencias significativas en relación a las distintas marcas de teléfonos móviles.
Desarrolladores con problemas de motivación.	Proyecto y de personas	Los desarrolladores presentan baja motivación, no participan activamente del proyecto.
Retrasos en la especificación	De estimación	Las especificaciones principales del sistema no estarán listas antes de la fecha de entrega.
Cambios en los Requerimientos	De requerimientos	Dependiendo de la evolución del proyecto irán apareciendo otros requerimientos en forma anticipada.
Subestimación de tamaño	Proyecto y de estimación	El tamaño del sistema ha sido subestimado.
Bajo desempeño de la herramienta IDE	De herramientas	La herramienta IDE que apoya el proyecto no tiene el desempeño anticipado.
Cambio de Tecnología	Negocio	La tecnología en la cual se construir el sistema ha sido sustituida por nueva tecnología.
Mala elección de Herramientas	De herramientas	Las herramientas seleccionadas para el desarrollo del proyecto presentan un desempeño deficiente.
Falta Capacitación	Proyecto y de personas	Falta de experiencia podría conducir a errores.

5.1.3.2 Clasificación de Riesgos

La Tabla 5.2 clasifica el riesgo, según su probabilidad de ocurrencia.

Tabla 5.2 - Probabilidad de riesgo

Probabilidad del riesgo	**Porcentajes**
Muy bajo	< 10%
Bajo	10% - 25%
Moderada	25% - 50%
Alta	50% - 75%
Muy alto	> 75%

La Tabla 5.3 clasifica las variables de riesgo identificadas en la Tabla 5.1 según su probabilidad de ocurrencia establecida en la Tabla 5.2, estableciendo cuales serían los efectos de la ocurrencia de alguna de estas.

Tabla 5.3 - Clasificación de riesgos

Riesgo	**Probabilidad**	**Efectos**
Cambios en los Requerimientos	Alta	Serio
No disponibilidad de hardware	Bajo	Serio
Heterogeneidad de los dispositivos móviles	Alta	Catastrófico
Cambio de Tecnología	Alto	Tolerable
Retrasos en la especificación	Moderada	Serio
Bajo desempeño de la herramienta IDE	Moderada	Insignificante
Subestimación de tamaño	Alta	Serio
Desarrolladores con problemas de motivación.	Bajo	Serio
Mala elección de Herramientas	Moderada	Catastrófico
Falta Capacitación	Bajo	Catastrófico

5.1.3.3 Planificación del riesgo

La Tabla 5.4 muestra para cada variable de riesgo identificada, la estrategia para mitigar las repercusiones de la ocurrencia de alguna de estas.

Tabla 5.4 - Planificación del riesgo

Riesgo	**Estrategia**
Cambios en los requerimientos	Rastrear la información para valorar el impacto de los requerimientos, maximizar la información oculta en ellos
Heterogeneidad de los dispositivos móviles	Se debe disponer de teléfonos móviles de distintas marcas para someterlos a pruebas con suficiente tiempo.
Retrasos en la especificación	Reorganizar las labores dentro del grupo de trabajo, y potenciar las virtudes del grupo laboral.
No disponibilidad de hardware	Reestructurar la planificación de manera de que el hardware que va estar no disponible, no sea esencial en algún periodo determinado.
Subestimación de tamaño	Aumentar la asignación de h/h a partes del proyecto que han sido subestimadas.
Desarrolladores con problemas de motivación	Incentivar trabajos en grupos y fijar objetivos a corto plazo.
Bajo desempeño de la herramienta Case	Desechar la herramienta y elegir una nueva, o no utilizar alguna.
Cambio de tecnología	Averiguar las posibilidades que ofrecen los distintos equipos que se encuentran en el mercado para la implementación del proyecto.
Mala elección de Herramientas	Buscar nuevas herramientas. Rehacer trabajo.
Falta Capacitación	Aumentar la capacitación de los desarrolladores.

5.1.3.4 Supervisión del riesgo

La Tabla 5.5 muestra cuales son los indicadores que podrían mostrar la presencia de una de las variables de riesgo identificadas anteriormente.

Tabla 5.5 - Supervisión del riesgo

Tipo de Riesgo	**Indicadores Potenciales**
Heterogeneidad de los dispositivos móviles	Problemas al implementar el sistema en distinto tipo de teléfonos móviles.
No Disponibilidad de Hardware	Entrega retrasada del hardware, problemas tecnológicos detectados.
Cambio en los Requerimientos	Demasiadas peticiones de cambios y surgimientos de nuevas necesidades por parte del cliente.
Retrasos en la especificación	Fracaso en el cumplimiento de alguno de los tiempos acordados.
Subestimación de tamaño	Aparecen nuevas funcionalidades que dan mayor complejidad al sistema
Desarrolladores con problemas de motivación	Desarrolladores no cumplen a cabalidad con las tareas designadas.
Desempeño de la herramienta IDE	Rechazo del equipo para utilizar la herramienta, quejas sobre la herramienta.
Cambio de Tecnología	Nueva tecnología sale al mercado reemplazando la ya existente.
Mala elección de Herramientas	Mal funcionamiento de la herramienta elegida.
Falta Capacitación	Atraso en el desarrollo de la aplicación.

5.2 Requerimientos: Ingeniería de requerimientos según el Modelo FURPS+

5.2.1 Funcionalidad del Sistema

5.2.1.1 Propiedades

- Permitir al Usuario del Cliente Móvil interactuar con el sistema a través del estándar Bluetooth, accediendo a contenidos como publicidad, un menú de postres o cafés, información publicada en diarios on-line, la información comercial de la Bolsa de Santiago o contestar alguna encuesta.
- Registrar la información recogida por el sistema, generada por la interacción de los usuarios del Cliente Móvil con este.

5.2.1.2 Capacidades

- Desplegar en el Cliente Móvil contenidos personalizados en cada nodo en particular, dependiendo de las especificaciones requeridas.
- Proporcionar una potente plataforma de marketing e información para los Clientes que posean un Nodo.
- Proporcionar al Usuario Móvil información de manera gratuita y actualizada.

5.2.2 Requerimientos No Funcionales

5.2.2.1 Facilidad de uso

- Interfaz amigable, simple e intuitiva que permite una rápida adaptación de los usuarios al sistema, no requiriendo así personal altamente capacitado en el uso de sistemas computacionales.

5.2.2.2 Confiabilidad

- Generar planes de anulación, disminución y/o contingencia para posibles fallos del sistema.

5.2.2.3 Desempeño

- El tiempo de respuesta para el procesamiento de transacciones y consultas debe ser rápido, pues los usuarios de clientes móviles por lo general, requieren tiempos de espera cortos.

5.2.2.4 Disponibilidad

- Una vez iniciado el sistema siempre estará en funcionamiento y en condiciones para su operación en cualquier momento.

5.2.3 Soporte

5.2.3.1 Adaptabilidad

- El sistema será implementado de forma de garantizar la adaptabilidad a los cambios de requerimientos o posibles mejoras solicitadas.

5.2.3.2 Mantención

- La implementación del sistema será en base a módulos para asegurar la facilidad de mantención si llegará a ser necesario.

5.2.3.3 Configuración

- El sistema deberá tener la capacidad de soportar cambios en la configuración de hardware y software que compone el sistema.

5.2.4 Implementación

5.2.4.1 Limitaciones de recursos, lenguajes y herramientas

- Las limitaciones de lenguajes y herramientas están autoimpuestas para poder realizar el proyecto de acuerdo con lo expresado en los objetivos específicos.

5.2.4.2 Limitaciones de hardware

  - El sistema deberá funcionar de manera eficiente según la especificación detallada en el punto 5.1.2.1 (Factibilidad Técnica).

5.3 Análisis del Sistema

5.3.1 Introducción

Mediante el Análisis del Sistema se busca tener una visión más funcional del proyecto, obteniendo con más claridad la labor del sistema. Además de la solución del dominio del problema que se ha identificado anteriormente. Para lograr este objetivo utilizaremos las herramientas de UML.

5.3.2 Casos de uso

El diagrama de casos de uso representa la forma en como un Cliente (Actor) opera con el sistema en desarrollo, además de la forma, tipo y orden en como los elementos interactúan, por lo demás que es una técnica para la captura de requisitos potenciales de un nuevo sistema o una actualización software. Cada "caso de uso" presta uno o más escenarios que indican cómo debería interactuar el sistema con el usuario o con otro sistema para conseguir un objetivo específico. A continuación se presentan los Diagramas de Casos Uso principales identificados en el sistema para cada Actor.

5.3.2.1 Diagrama de Casos de Uso - Usuario Móvil

Usuario Móvil: Este actor representa la persona que hace uso del sistema, a través de su teléfono móvil, de realizando las tareas de usar los distintos servicios ofrecidos en cada nodo del sistema.

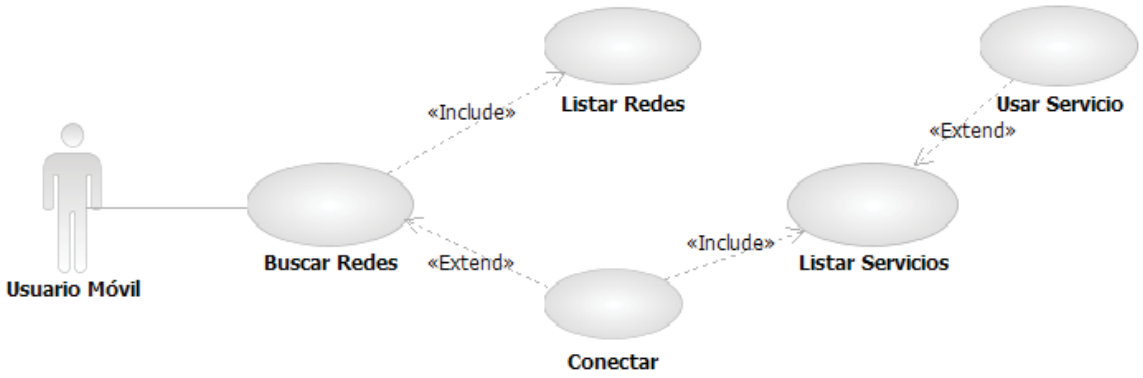

Ilustración 5-2 - Diagrama de Casos de Uso - Usuario Móvil

Del Diagrama anterior se pueden rescatar los siguientes Casos de Uso:

- Buscar Redes: El usuario del Teléfono Móvil (Usuario Móvil) desea ingresar a la red generada por el nodo para realizar alguna de las funcionalidades que el sistema ofrece, para esto, el dispositivo móvil debe conocer que redes están disponibles en ese espacio físico, pues puede haber más de una red disponible.
- Conectar: El actor elige conectarse a alguna de las redes disponibles en el área.
- Usar Servicio: Alguno de los servicios disponibles en el nodo es utilizado por el cliente móvil.

5.3.2.2 Diagrama de Casos de Uso - Administrador del Sistema

Administrador del Sistema: Este actor representa a la persona que gestiona y administra el sistema, a través de una interfaz que le permite manejar la creación de nuevos Clientes, Nodos y Servicios, la gestión de sus datos y su activación o desactivación.

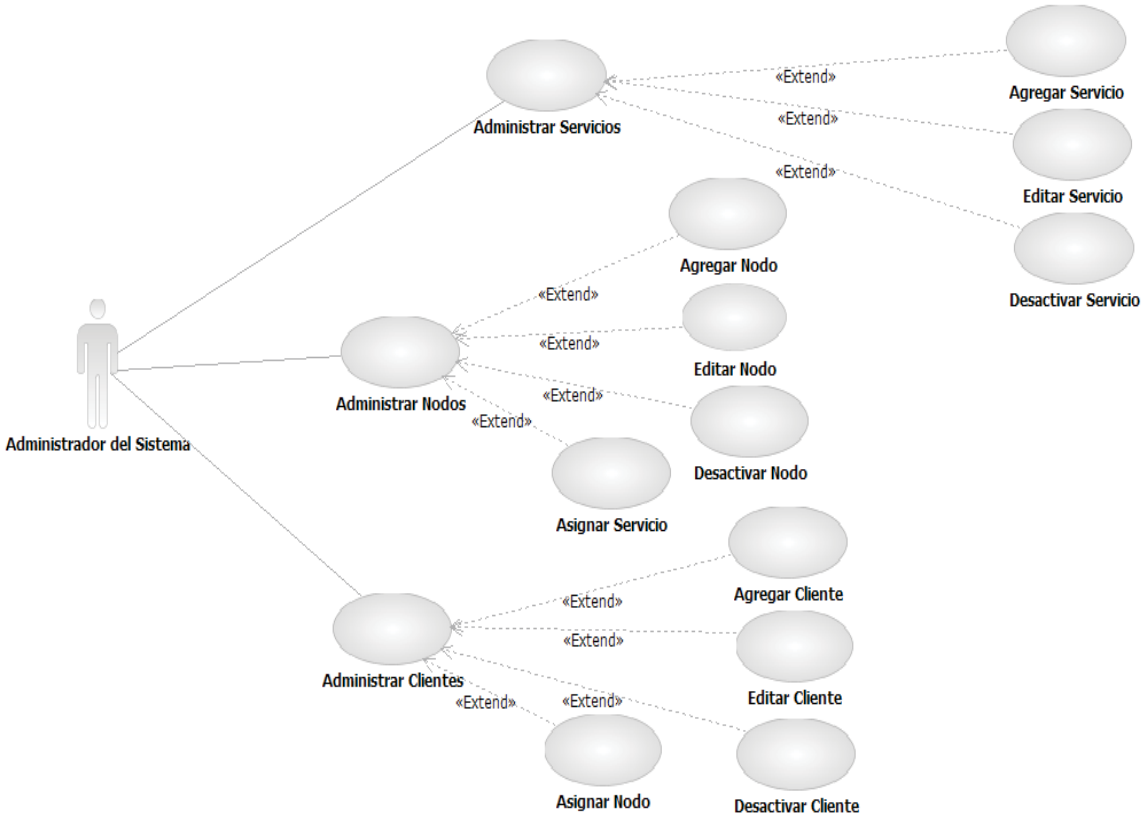

Ilustración 5-3 - Diagrama de Casos de Uso - Administrador del Sistema

Del Diagrama anterior se pueden rescatar los siguientes Casos de Uso:

- Administrar Servicios: El Actor desea realizar alguna operación sobre los servicios disponibles en el sistema, como Agregar, Editar o Eliminar alguno de los servicios existentes.
- Administrar Nodos: El Actor desea hacer realizar alguna operación sobre los Nodos de los que se compone el sistema, este caso de uso es muy importante pues representa la funcionalidad más significativa para este actor.
- Administrar Clientes: El Actor desea realizar alguna acción sobre los Clientes registrados en el sistema.
- Asignar Servicio: El Actor desea asociar un servicio determinado, por ejemplo, las noticias deportivas, a un determinado Nodo dentro de la red, para ello, ejecuta este caso de uso, asignando al Nodo este servicio para poder ser consumido desde el Nodo escogido.

5.3.3 Diagrama de Paquetes

La Ilustración 5-4 muestra la jerarquía de paquetes, esta jerarquía denota las Clases que participan en el funcionamiento del Cliente Móvil, el Servidor Central y el Nodo. Los paquetes se utilizan para la agrupación de clases con alta cohesión y bajo acoplamiento principio básico para un buen modelado Orientado a Objetos.

Los paquetes pertenecientes al contexto del Nodo, se dividen en el Módulo de comunicación, encargado de, utilizando la API Aventana, comunicar el Nodo con el Cliente Móvil, el Módulo XML, encargado de, utilizando la API dom4j, procesar la información proveniente de los documentos XML de la Base de Datos eXist y un Módulo de Sincronización entre la Base de Datos y el Nodo. El paquete Cliente Móvil contiene la GUI y la lógica del móvil dentro de él.

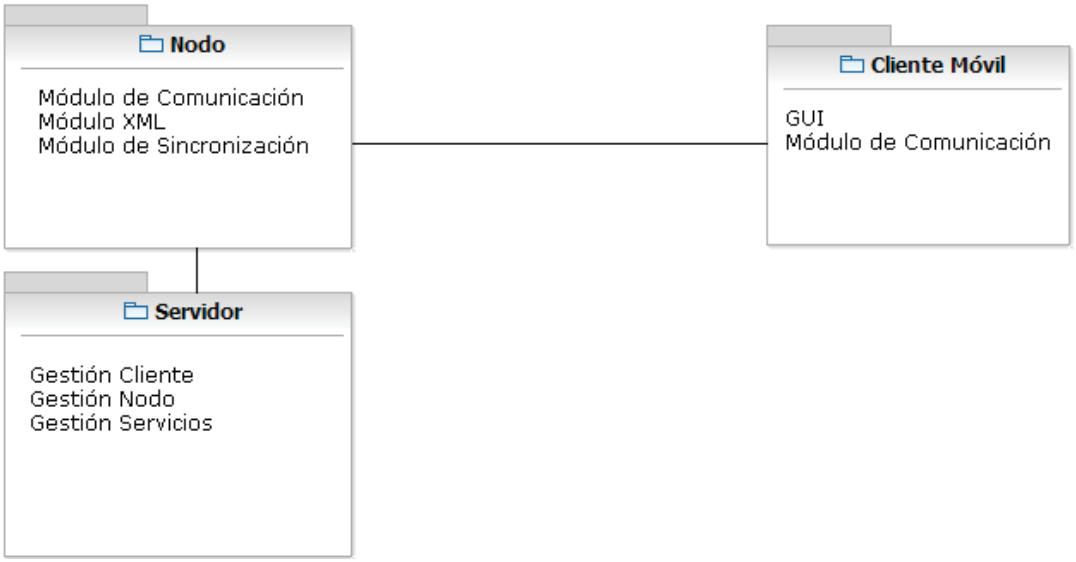

Ilustración 5-4 - Diagrama General de Paquetes

5.3.4 Diagramas de Clases

El Diagrama de Clases es el diagrama principal para el análisis y diseño. Un diagrama de clases presenta las clases del sistema con sus relaciones estructurales y de herencia. La definición de clase incluye definiciones para atributos y operaciones. El modelo de casos de uso aporta información para establecer las clases, objetos, atributos y operaciones. El mundo real puede ser visto desde abstracciones diferentes (subjetividad).

5.3.4.1 Diagrama de Clases - Módulo XML:

La Ilustración 5-5 muestra las relaciones entre las distintas clases del Módulo XML, una Clase de Control, maneja y comunica el flujo de datos desde y hacia el ParserXML, este por su parte reconoce esta información y va creando los Displayables que serán enviados al Cliente Móvil. Por otra parte el ParserXML puede reconocer la finalización (Comando de finalización de Displayables), al ocurrir esto, el Parser escribe la información recogida desde el Cliente Móvil para ser luego sincronizada con la Base de Datos XML.

Cada Displayable es creado a petición de la clase de Control, el ParserXML crea el Displayable y agrega a este cada ítem especificado en el Documento XML consultado, luego el resultado es retornado a la clase de Control, para su envío al Cliente Móvil.

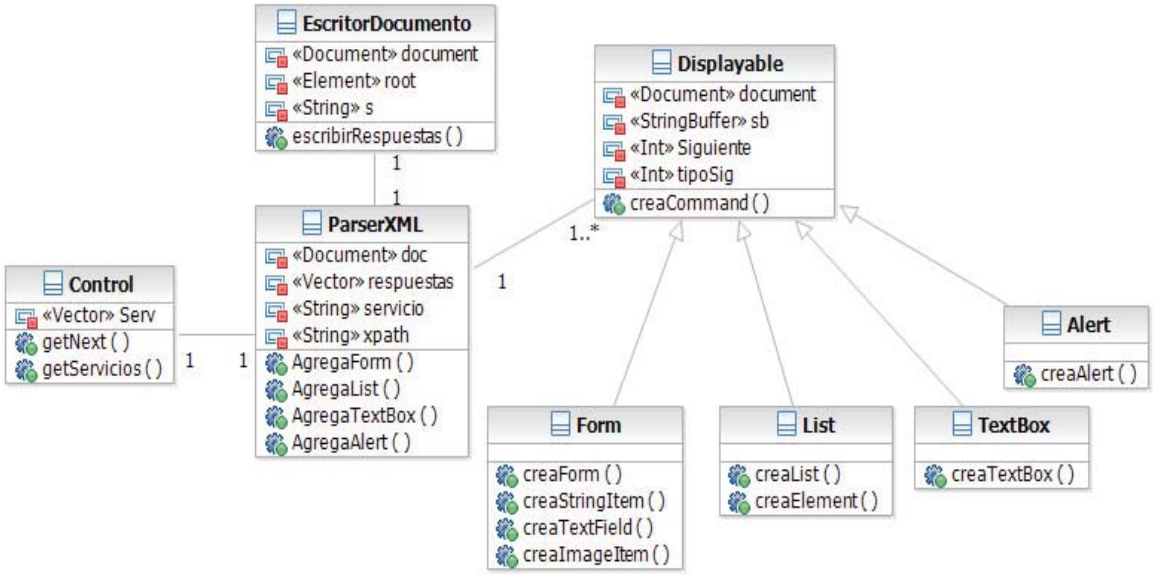

Ilustración 5-5 - Diagrama de Clases - Modulo XML

5.3.4.2 Diagrama de Clases - Cliente Móvil:

La Ilustración 5-6 muestra la relación entre las clases del Cliente Móvil, en ella clase MainBT controla la interacción con el usuario. Para la búsqueda del servicio dentro de los nodos se utiliza la clase BuscardorNodos y ListarDispositivos esta última encargada de mostrar al usuario los nodos con el servicio disponibles.

Para crear la conexión se utiliza la clase Cliente que se preocupa de gestionar el enlace con el Nodo. Al recibir un paquete de generación de Displayable la clase CrearVentana determina que tipo de Displayable se debe generar e invoca a la clase específica para su construcción y despliegue por pantalla.

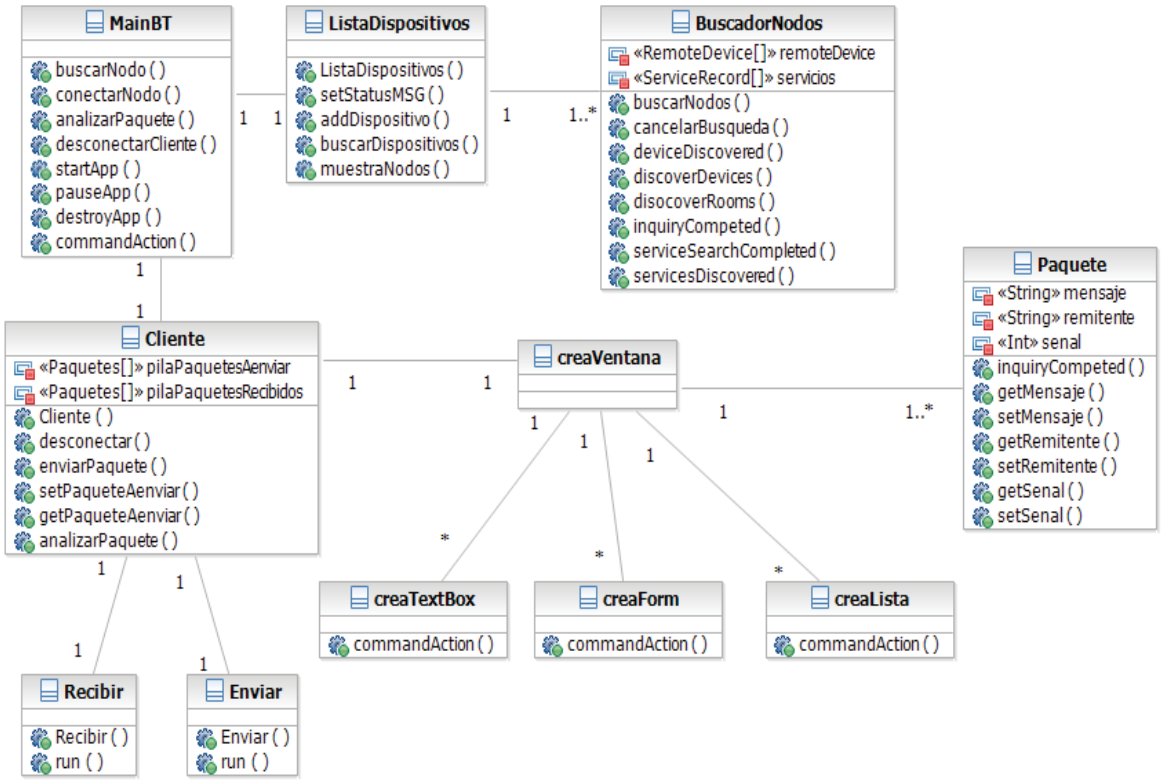

Ilustración 5-6 - Diagrama de Clases - Cliente Móvil

5.3.4.3 Diagrama de Clases - Módulo de Comunicaciones:

La Ilustración 5-7 muestra la relación entre las clases del Módulo de Comunicaciones alojado en el Nodo. La clase Main es la encargada de instanciar el Nodo, el cual esta diseñado para gestionar la conexión con cada uno de los clientes. Por otra parte la clase Cliente envía los paquetes recibidos desde un cliente hacia el Módulo XML a través de la clase Control.

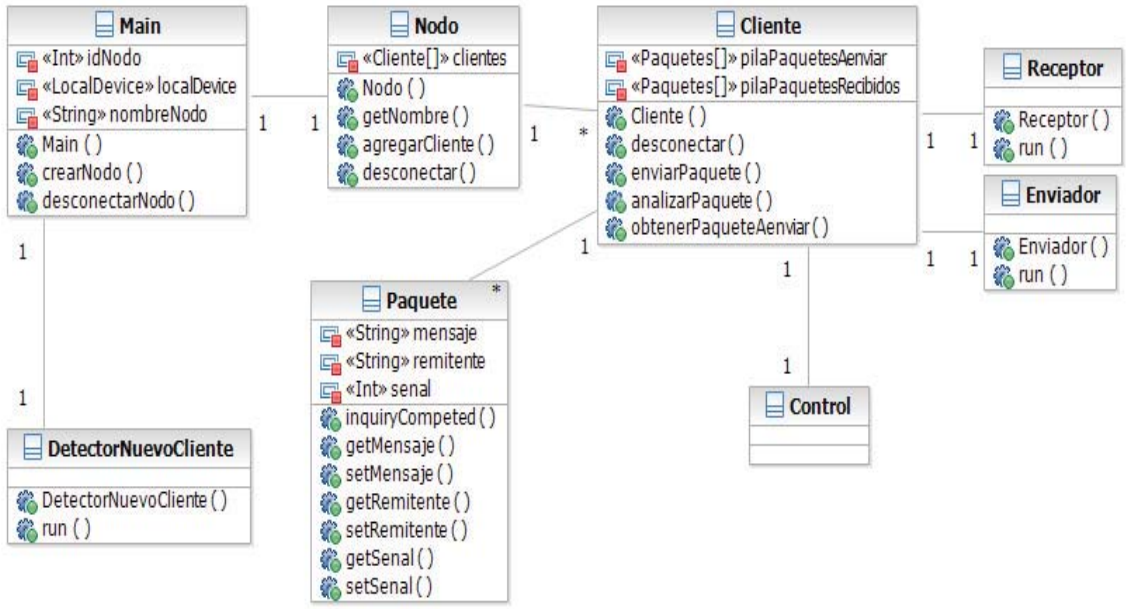

Ilustración 5-7 - Diagrama de Clases - Módulo de Comunicaciones

5.3.5 Diagramas de Secuencia

Los diagramas de secuencia exponen gráficamente los eventos que fluyen de los actores del sistema, además de ser una representación que muestra un determinado escenario de un caso de uso, de manera que se pueda conocer mejor el sistema en desarrollo.

Los siguientes diagramas de secuencia muestran la interacción entre los actores identificados anteriormente (Usuario Móvil y Administrador del Sistema) y los componentes funcionales del sistema, haciendo más claro y sencillo el entendimiento del sistema en su conjunto, logrando una integración de los diferentes componentes que pertenecen al proyecto. A continuación se presentan algunos de los diagramas más relevantes rescatados desde los Casos de Uso.

5.3.5.1 Diagrama de Secuencia Conectar

La Ilustración 5-8 describe la interacción del Usuario del Cliente Móvil al conectarse a alguna red generada por los nodos del sistema. Para que esto suceda el actor indica su deseo de detectar las redes disponibles, para luego escoger una de acuerdo a la respuesta obtenida desde del MIDLet, luego de eso, el MIDLet se conecta a la red escogida. Este diagrama describe el escenario base del actor Usuario Móvil.

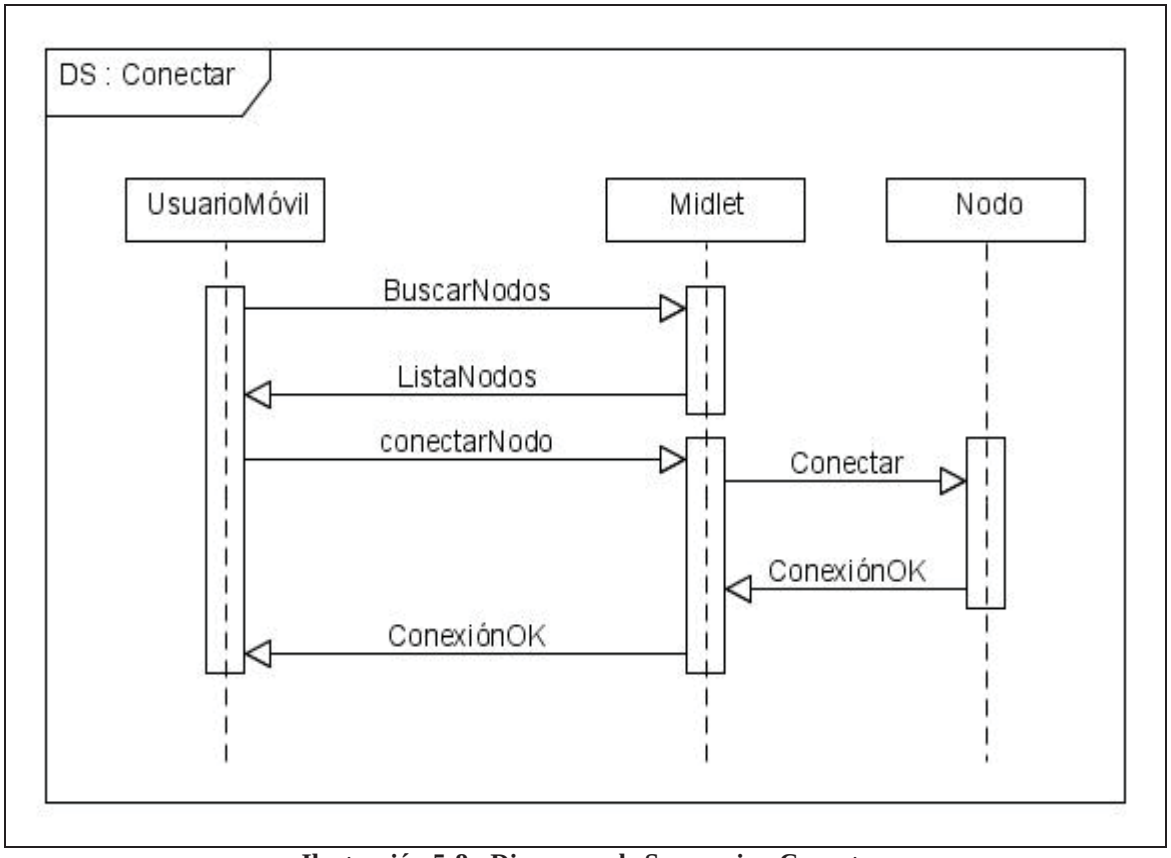

Ilustración 5-8 - Diagrama de Secuencia - Conectar

5.3.5.2 Diagrama de Secuencia Usar Servicio

Una vez conectado a una red (piconet) de un Nodo en particular el Usuario Móvil obtiene la lista de servicios disponibles y elige usar uno de estos, como se muestra en la Ilustración 5-9, luego de esto el primer Displayable es enviado automáticamente al Cliente Móvil y mostrado por pantalla por el MIDLet, los siguientes Displayables son generados a partir de las información recogida desde el MIDLet, hasta reconocer una señal de salida, lo que finaliza exitosamente esta este escenario.

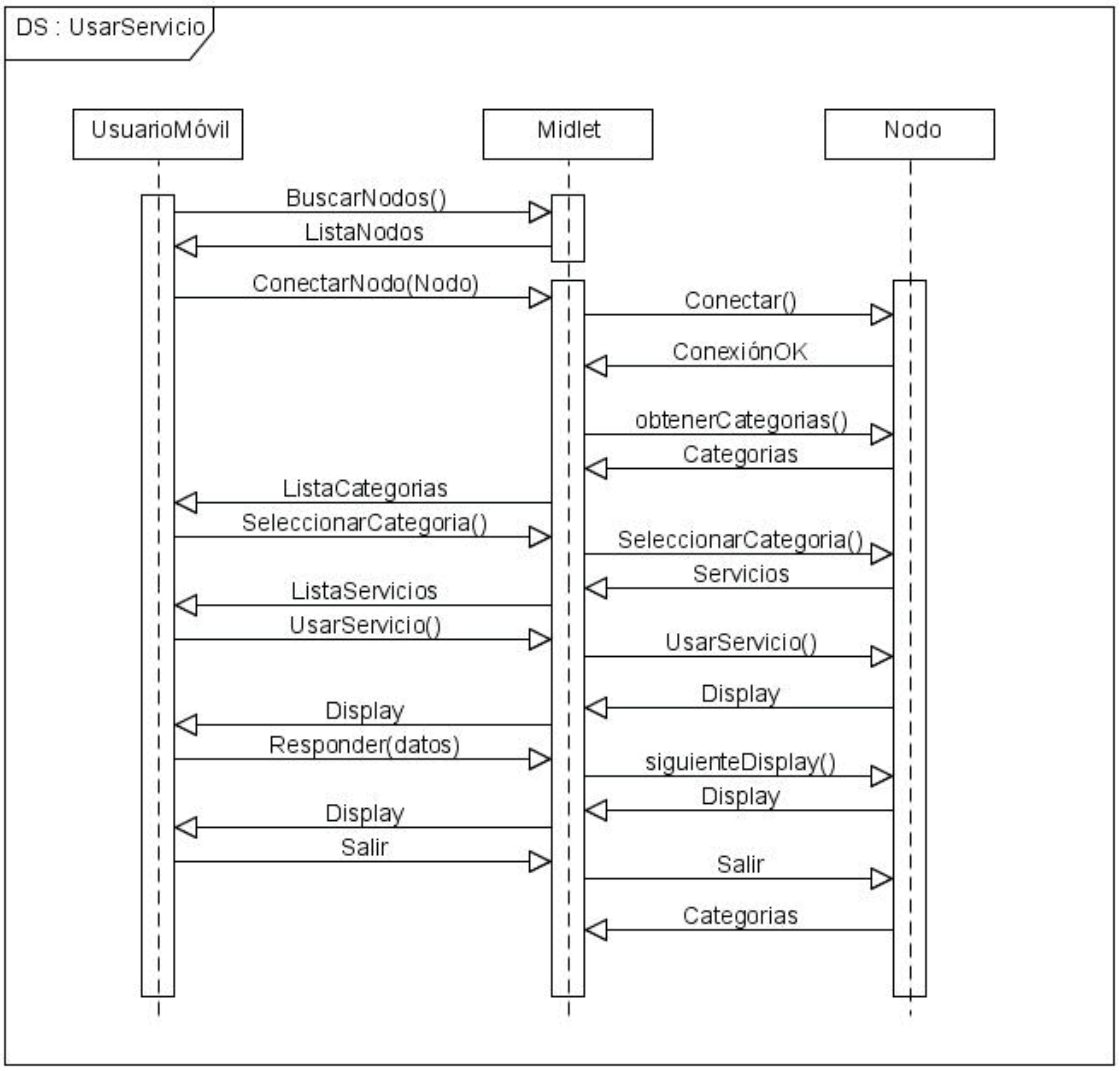

Ilustración 5-9 - Diagrama de Secuencia - Usar Servicio

5.4 Diseño

5.4.1 Introducción

El diseño es un refinamiento que toma en cuenta los requerimientos no funcionales, por lo cual se centra en como el sistema cumple sus objetivos.

Los objetivos específicos del análisis y diseño son:

- Transformar los requerimientos al diseño del futuro sistema.
- Desarrollar una arquitectura para el sistema.
- Adaptar el diseño para que sea consistente con el entorno de implementación.

5.4.2 Elementos físicos y lógicos de la arquitectura

Como se puede apreciar en la Ilustración 5-10 el Cliente Móvil se conecta a la piconet creada por el nodo, haciendo uso del MIDLet, para poder utilizar los servicios disponibles en ese contexto espacial. La información enviada desde el Cliente Móvil es recogida y almacenada en un repositorio XML y luego sincronizada con la Base de datos. La conexión entre cada Nodo y el servidor de BD XML (eXist) se realiza a mediante XMLRPC y a través de la VPN.

El Servidor Central aloja el Servidor de Base de Datos XML (eXist), el contenedor Tomcat con los Servicios Web desarrollados y el Servidor de VPN. Cada Nodo por su parte posee uno o varios dispositivos Bluetooth, eso permite a los Usuarios del Cliente Móvil, conectarse y utilizar los servicios disponibles para su Nodo en particular, además este nodo debe ser capaz de consumir el o los Servicios Web alojados en el Servidor Central.

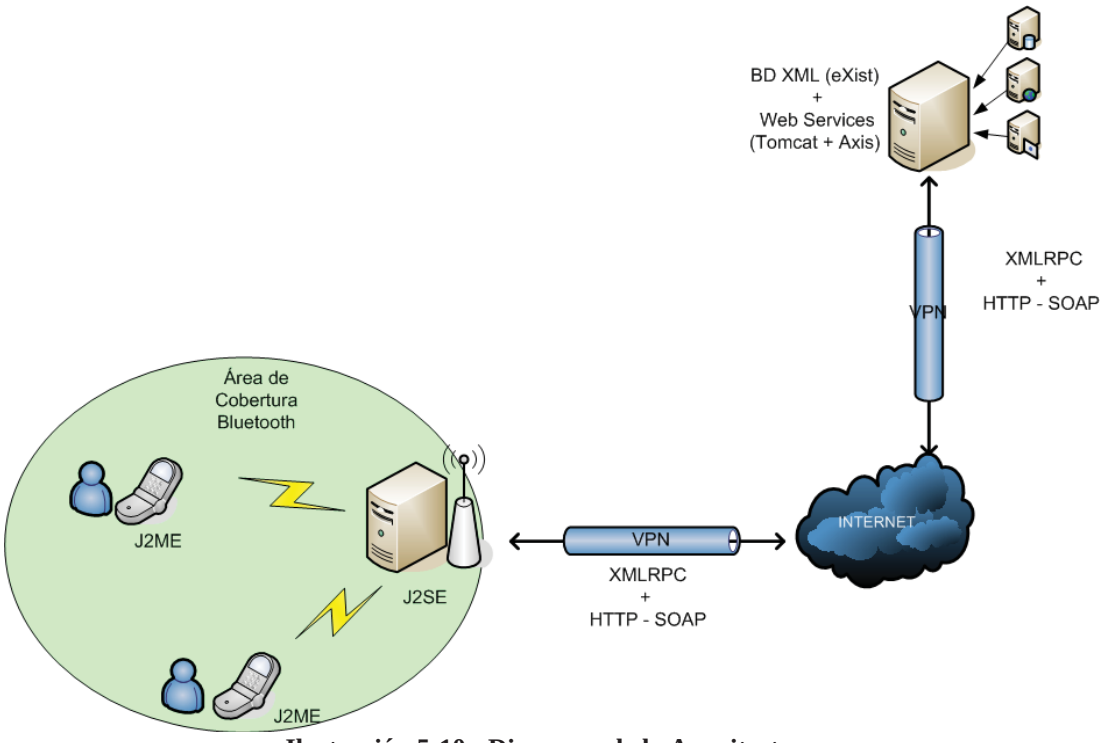

Ilustración 5-10 - Diagrama de la Arquitectura

5.4.3 Componentes de la arquitectura de desarrollo

- **J2ME:** La plataforma J2ME[1] es una familia de especificaciones que definen varias versiones minimizadas de la plataforma Java 2; estas versiones minimizadas pueden ser usadas para programar en dispositivos electrónicos; desde teléfonos celulares, en PDAs, hasta en tarjetas inteligentes. Estos dispositivos presentan en común que no disponen de abundante memoria ni mucha potencia en el procesamiento, ni tampoco necesitan de todo el soporte que brinda el J2SE, (la plataforma estándar de Java usada en sistemas de escritorio y servidor).

- **Tomcat:** Tomcat (también llamado Jakarta Tomcat o Apache Tomcat) funciona como un contenedor de Servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation. Tomcat es un servidor Web con soporte de Servlets y JSPs. Incluye el compilador Jasper, que compila JSPs convirtiéndolas en Servlets. El motor de Servlets de Tomcat a menudo se presenta en combinación con el servidor Web Apache. Dado que Tomcat fue escrito en Java, funciona en cualquier sistema operativo que disponga de la máquina virtual Java.

- **J2SE:** Java Platform, Standard Edition o Java SE (conocido anteriormente hasta la versión 5.0 como Plataforma Java 2, Standard Edition o J2SE), es una colección de APIs del lenguaje de programación Java útiles para muchos programas de la Plataforma Java.

- **XML:** Sigla en inglés de Extensible Markup Language[2] (lenguaje de marcas extensible), es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML). Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG, MathML. XML no ha nacido sólo para su aplicación en Internet, sino que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo y casi cualquier cosa imaginable. XML es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

- **XML Schema:** XML Schema es un lenguaje de esquema utilizado para describir la estructura y las restricciones de los contenidos de los documentos XML de una forma muy precisa, más allá de las normas sintácticas impuestas por el propio lenguaje XML. Se consigue así, una percepción del tipo de documento con un nivel alto de abstracción. Fue desarrollado por el World Wide Web Consortium (W3C) y alcanzó el nivel de recomendación en mayo de 2001.

[1] Para mayor información véase Anexo B J2ME
[2] Para mayor información véase Anexo D XML

- **Axis:** Es una implementación Open-Source de un "SOAP Engine"; a través de este componente es posible llevar acabo una comunicación mediante "Web-Services" una de las variaciones más prometedoras del lenguaje XML. En el caso de "Axis", éste se encuentra diseñado para ser ejecutado dentro de un "Java Application Server" o bien un "Servlet Engine" como Tomcat, aunque hoy en día ya existen otros "SOAP Engines" que pueden ser ejecutados de manera independiente de un "Java Application Server" o "Servlet Engine", al residir un "SOAP Engine" dentro estos, se permite que métodos residentes puedan ser publicados como "Web-Services" y de esta manera ser accesibles de otras plataformas/lenguajes que también soporten "SOAP"
- **XMLDB:** Conjunto de APIs que proveen una interfaz de comunicaciones entre la Base de Datos XML y Java.
- **VPN:** La Red Privada Virtual (RPV), en inglés Virtual Private Network (VPN)[1], es una tecnología de red que permite una extensión de la red local sobre una red pública o no controlada, como por ejemplo Internet. Ejemplos comunes son, la posibilidad de conectar dos o más sucursales de una empresa utilizando como vínculo Internet, permitir a los miembros del equipo de soporte técnico la conexión desde su casa al centro de cómputo, o que un usuario pueda acceder a su equipo doméstico desde un sitio remoto, como por ejemplo un hotel. Todo ello utilizando la infraestructura de Internet.
- **Bluetooth:** Es el nombre común de la especificación industrial IEEE 802.15.1, que define un estándar global de comunicación inalámbrica que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia segura, globalmente y sin licencia de corto rango.
- **Servicios Web:** Un Servicio Web[2] (en inglés Web Service) es una colección de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios Web para intercambiar datos en redes de ordenadores como Internet. La interoperabilidad se consigue mediante la adopción de estándares abiertos.
- **XML-RPC:** Es un protocolo de llamada a procedimiento remoto que usa XML para codificar los datos y HTTP como protocolo de transmisión de mensajes. Es un protocolo muy simple ya que sólo define unos cuantos tipos de datos y comandos útiles, además de una descripción completa de corta extensión. La simplicidad del XML-RPC está en contraste con la mayoría de protocolos RPC que tiene una documentación extensa y requiere considerable soporte de software para su uso.

[1] Para mayor información véase Anexo C VPN
[2] Para mayor información véase Anexo E Web Services

5.4.4 Diagrama de despliegue

El Diagrama de Despliegue es un tipo de diagrama que se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes.

Los Diagramas de Despliegue muestran la disposición física de los distintos componentes del sistema y el reparto de los componentes. La vista de despliegue representa la disposición de las instancias de componentes de ejecución conectados por enlaces de comunicación. Un recurso de ejecución tal como un computador, un dispositivo o memoria. A continuación, en la Ilustración 5-11, se presenta el diagrama de despliegue del sistema:

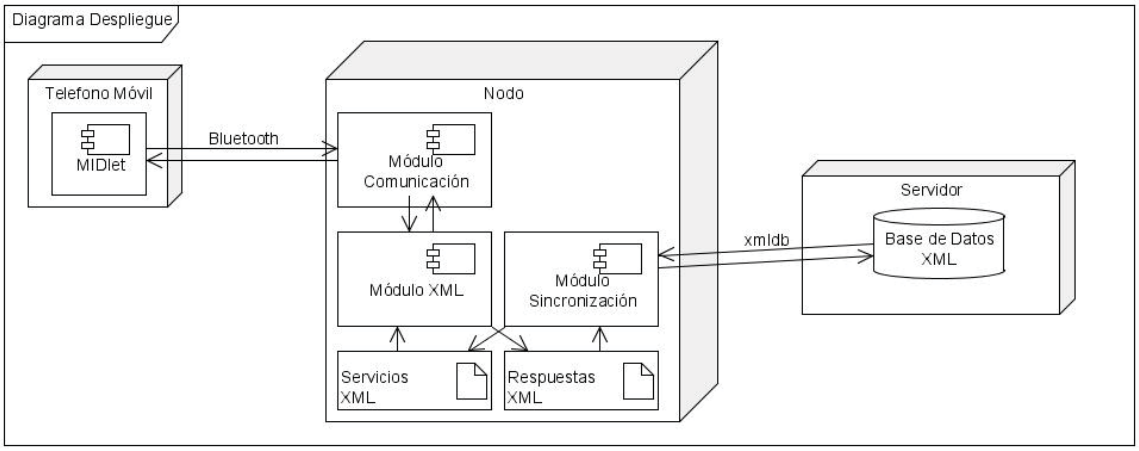

Ilustración 5-11 - Diagrama de Despliegue

El diagrama de despliegue de la Ilustración 5-11, muestra la disposición física de los componentes y sus enlaces de comunicaciones, el Teléfono Móvil se comunica mediante Bluetooth con el Dispositivo Bluetooth, este es controlado mediante un Módulo de Comunicaciones que se encarga de enviar información al Módulo XML que procesa la información y genera respuestas hacia el Cliente Móvil, estas respuestas son generadas a partir de los Servicios XML (Documentos XML). Por otra parte el Servidor contiene una Interfaz desarrollada en SWING para la gestión por parte del Administrador del Sistema, además de esto el Servidor contiene la Base de Datos XML que se comunica con los Nodos a través de un Módulo de Sincronización.

5.4.5 Diseño de la Base de Datos XML (eXist)

Las bases de datos XML nativas se diferencian a sí mismas y ganan el término nativas porque almacenan los datos estructurados como XML sin la necesidad de traducir los datos a una estructura relacional o de objeto.

Las bases de datos XML nativas son diseñadas para trabajar con XQL (eXtensible Query Language), el cuál sirve un propósito similar a SQL en una base de datos relacional. XQL está diseñado para trabajar con documentos XML jerárquicamente estructurados y puede proveer características de consulta como filtros y joins. Los esquemas XML son implementados en bases de datos XML nativas para registrar reglas de almacenamiento e indexación de datos y para proveer y obtener información de almacenamiento a los mecanismos de bases de datos XML nativas.

La jerarquía de la Base de datos muestra en el primer nivel compuesto de los directorios de Usuarios del sistema. Dentro del directorio usuario en el segundo nivel de la estructura, podemos encontrar el documento Usuario.xml que almacena los datos referentes al usuario de la cuenta, el documento Estrucutura.xml encargado de llevar el control de los Nodos pertenecientes al usuario y de los servicios instalados en cada Nodo. Dentro del directorio de cada Nodo se encuentran dos documentos, Sincronizacion.xml, el cual almacena el método particular de sincronización del Nodo, y el documento, Descripcion.xml, el cual registra información relativa al Nodo. A su vez en este mismo directorio se encuentran los subdirectorios respuestas y servicios, donde se almacenan las copias de los archivos de servicios, respuesta y configuraciones, como se muestra en la Ilustración 5-12.

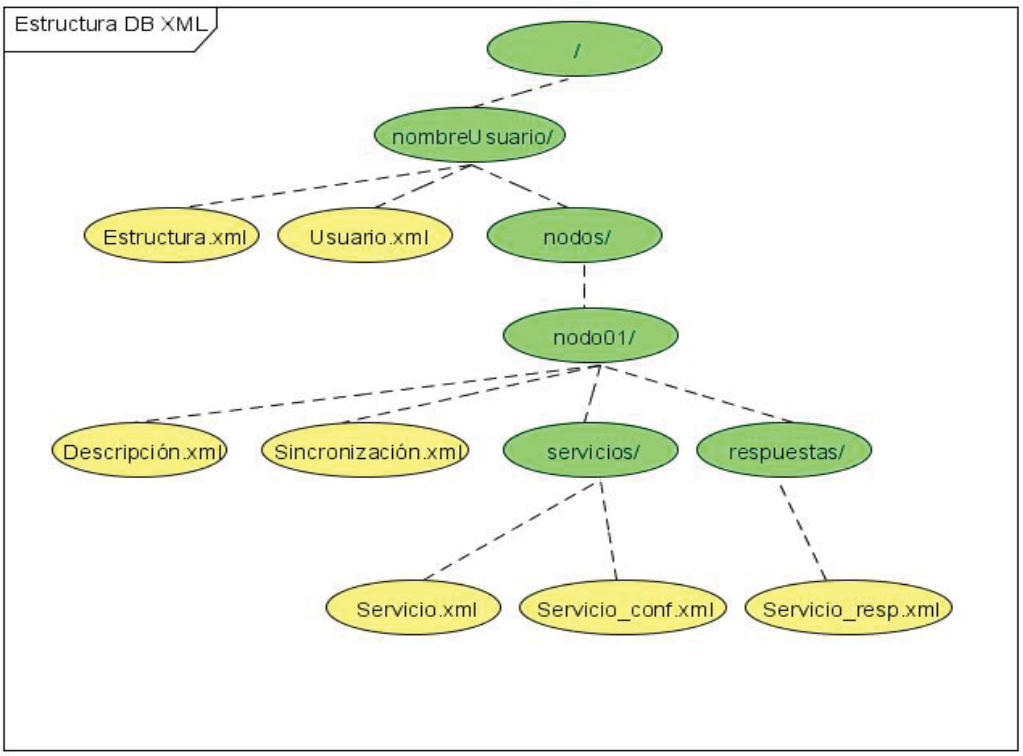

Ilustración 5-12 - Estructura de la Base de Datos XML - eXist

5.5 Implementación

El objetivo principal de la implementación del sistema es convertir los elementos del diseño en elementos de implementación, dichos elementos son códigos fuentes, ejecutables, binarios, entre otros. Otra parte de esta disciplina son las pruebas de unidad, las cuales se limitan a los componentes de software implementados. De esta disciplina se obtiene un sistema ejecutable estable, constituido de los resultados producidos por los programadores individuales.

Los objetivos específicos de la implementación fueron:

- Planificar qué subsistemas deben ser implementados y en que orden deben ser integrados.
- Cada desarrollador decide en que orden implementa los elementos del subsistema.
- Notificar los errores de diseño, si se encuentran.
- Probar los subsistemas individualmente.
- Integrar el sistema siguiendo el plan.

Para la implementación del sistema se tuvieron en cuenta aspectos como las fuentes de datos (Documentos XML), la configuración asociada a este, y como transmitir estos datos al MIDLet que finalmente interpreta los datos creando Displayables que se despliegan por la Interfaz del Cliente Móvil, para esto, se desarrolló un protocolo de comunicaciones de la aplicación, que contiene los datos transmitidos desde el Nodo al Cliente Móvil mediante Bluetooth.

5.5.1 Diseño de los Documentos XML

Como se puede apreciar en la Ilustración 5-13, los contenidos que se despliegan en la interfaz del Cliente Móvil (Ventanas o Displayables) están representados en Documentos XML que describen la forma y contenido de cada Ventana y la secuencia en que estas se muestran.

Además de esto, cada Servicio tiene asociado un Archivo de configuración que contiene información relevante para el uso o no de algún servicio en específico, eso quiere decir que, por ejemplo, existe la posibilidad de mostrar un servicio como disponible solo en una cierta “banda horaria” como contenidos de publicidad en específico o la posibilidad de asignar cuotas de descargas o uso a ese mismo u otro Servicio XML.

Para poder representar una Ventana o Displayable, fue necesario desarrollar una estructura de datos XML (XML Schema) que permitiera representar las Ventanas de una forma fácil e intuitiva y luego, a partir de eso, generar los Documentos XML que contienen la información de cada Servicio en particular, para definir la estructura tanto de los Documentos XML que describen cada unos de los Servicios, como para describir el Archivo de Configuración asociado.

5.5.1.1 Schema Servicios XML

A continuación en la Ilustración 5-13, se muestra una vista general del XML Schema que describe los Servicios. Para el Modelado se ocupo la IDE Netbeans 6.0.

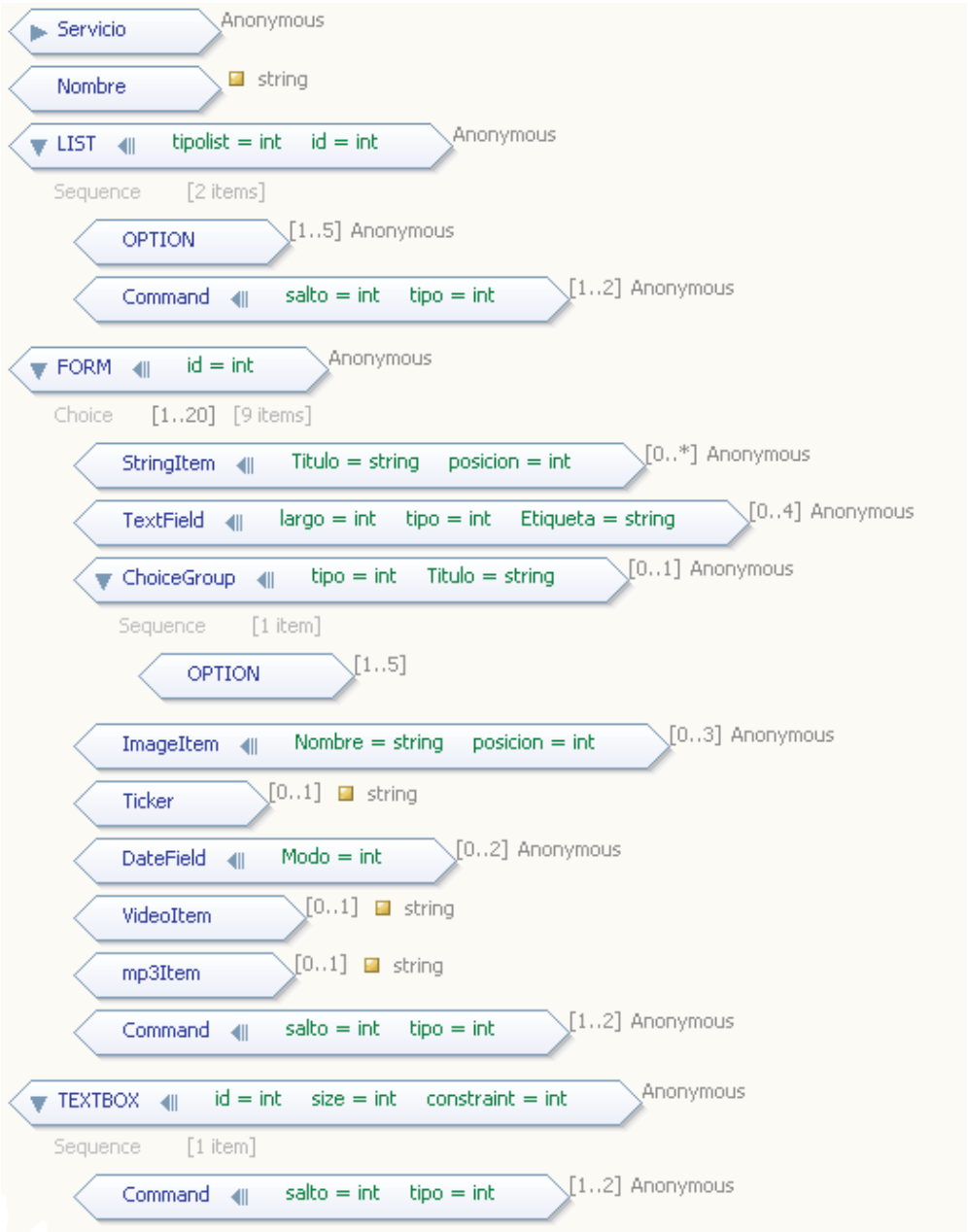

Ilustración 5-13 - Schema de los Servicios XML

Como se puede apreciar, un Servicio XML, puede estar compuesto por múltiples ventanas o displayables, ventanas de tipo Form (Formularios), del tipo List (Listas de Selección) o TextBox (Cajas de texto). Esto permite la combinación de este tipo de ventanas en un mismo Servicio XML. El Schema provee de reglas de sintaxis para la estructuración de los Servicios XML, así se puede escribir un Servicio basándose en estas reglas, es decir, un documento bien formado y validado contra este XML Schema.

5.5.1.2 Schema Documentos de Configuración

Los archivos o documentos de configuración asociados a cada Servicio XML, definen un conjunto de variables que representan el estado en que se encuentra ese Servicio en particular, por ejemplo, se define la banda horaria en que ese Servicio está habilitado o no, la cuota de descarga o uso del Servicio, etc.

Así, estos documentos especifican el comportamiento de los servicios, es decir, si un Servicio no debe ser mostrado en cierto horario, este se no se mostrará, o si la cuota del uso ha llegado a 0, sucederá lo mismo. Además, se especifica que cada servicio pertenece a una Categoría en especial. Existen otros campos que aún no tienen una implementación y están en fase de estudio para su habilitación.

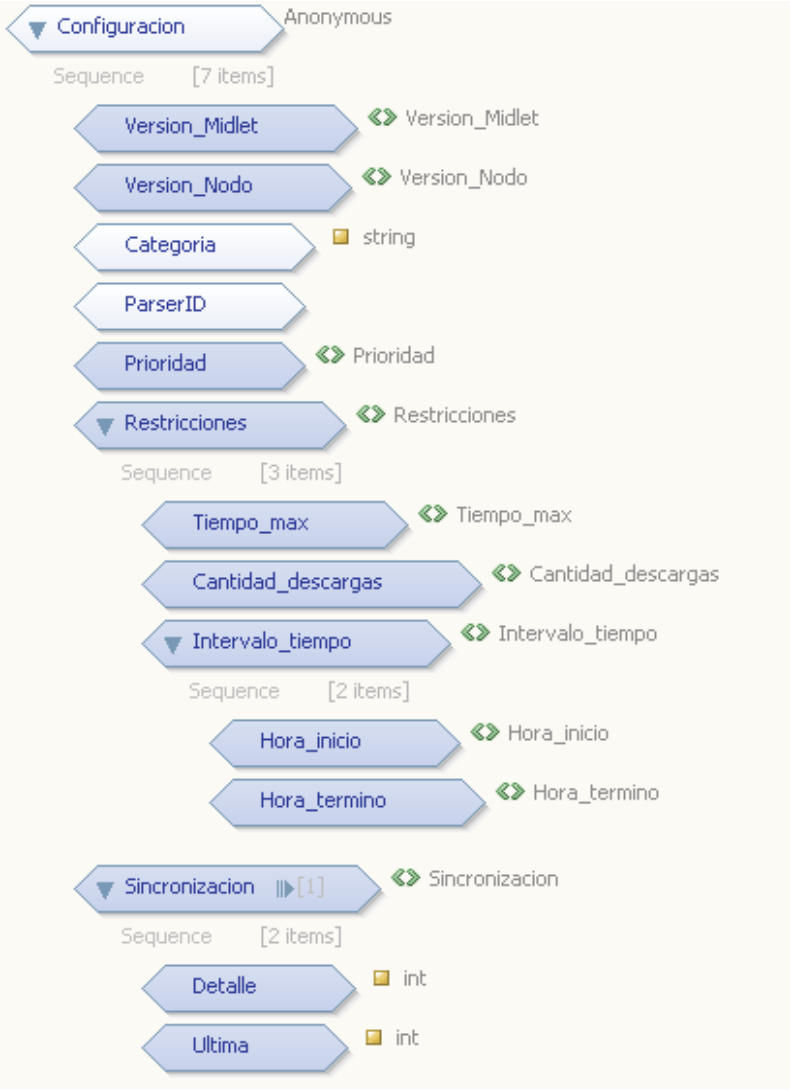

Ilustración 5-14 - Schema de los Documentos de Configuración

5.5.2 Protocolo de Aplicación

Para comunicar el dispositivo Móvil con lo que llamamos Nodo mediante Bluetooth se envían 2 clases de Strings, un String generador y un String respuesta, obteniendo con estos una comunicación síncrona entre el dispositivo móvil y el Nodo. El String generador, como su nombre indica, es el contenedor del displayable que el MIDLet del dispositivo móvil debe desplegar por pantalla, el String respuesta en cambio, contiene la información recogida desde el dispositivo móvil.

5.5.2.1 String Generador

El String generador es una concatenación de strings unidas por un caracter separador, que le permite al módulo generador de ventanas del móvil, parsear los diferentes campos obtenidos dentro del String generador. El primer paso es definir el tipo de Displayable. Los tipos de Displayables se muestran en la Tabla 5.6.

Tabla 5.6 - Tipos de Displayable

Displayable	Código	Parámetro 1	Parámetro 2	Parámetro 3
Form	1	S título		
List	2	S título	I tipo	
TextBox	4	S título	I maxsize	I constraints

Luego se debe definir cuántos y qué comandos serán desplegados por pantalla, existen 2 tipos básicos de comandos, FORWARD (Hacia adelante) y EXIT (Salir) como se detalla en la Tabla 5.7.

Tabla 5.7 - Tipos de Comando

Tipo de Comando	Código	Acción
FORWARD	1	Se comunica al nodo que se requiere del próximo String Generador
EXIT	2	Se comunica al nodo que se desea volver al Menú Principal

En orden debe ir el código del Displayable en primer lugar, seguido por sus opciones (Parámetros) y finalmente los comandos del displayable. Es preciso concatenar un comando como mínimo y dos como máximo al Displayable. Como se puede observar los comandos controlan la interacción del usuario móvil con el sistema, pues al desplegar por pantalla el comando elegido, este se convierte en una opción para escoger.

A continuación se detallan por separado las características propias de cada Displayable, sus parámetros y opciones particulares que posibilitan su correcto despliegue por pantalla.

- **Displayable TextBox**

Un Textbox posee 3 parámetros como se muestra en la Tabla 5.6, el parámetro Título que representa, como su nombre lo dice, el nombre o título del TextBox, el parámetro maxsize, que representa la cantidad máxima de caracteres que se pueden ingresar en la Caja de Texto, y el parámetro constraints que son restricciones o filtros para el ingreso de datos en el Cliente Móvil.

Este puede tomar varios valores:

Tabla 5.8 - Restricciones de TextBox (Constraints)

Constraints	Código	Descripción
ANY	0	Sin Restricción (puede integrar el diccionario T9)
EMAILADDR	1	Valida direcciones de correo
NUMERIC	2	Solo se permite de dígitos del 1 al 9
PHONENUMBER	3	Busca números de teléfonos en la agenda del móvil
URL	4	Ofrece listo el http://
DECIMAL	5	Para números decimales

Al String Generador de un TextBox no se debe concatenar nada, su definición es suficiente.

- **Displayable List**

Como se muestra en la Tabla 5.9, un Displayable List posee 2 parámetros, el primero al igual que en un Displayable TextBox, su título, el segundo (Parámetro 2) representa el Tipo de lista a desplegar por pantalla, esta puede tomar los siguientes valores:

Tabla 5.9 - Tipos de Listas (List)

Tipo	Valor	Descripción
MULTIPLE	1	Se pueden seleccionar más de una opción
EXCLUSIVA	2	Se marca una sola opción.
IMPLICITA	3	Al seleccionar una opción termina automáticamente la lista.

- **Displayable Form**

Un Form puede estar compuesto de varios ítems, de distinto tipo y con diferentes parámetros a diferencia de la lista (List) que tiene sólo tipo de elemento, estos ítems se muestran en la Tabla 5.10.

Tabla 5.10 - Tipos de Ítems para un Form

Tipo de Ítem	Código	Descripción
StringItem	1	Un área de Texto Fijo
ImageItem	2	Poner una imagen en el formulario
TextField	3	Un área de Input por el usuario
Ticket	4	Marquesina móvil
DateField	5	Ingresos para fecha y hora
ChoiceGroup	6	Botones de selección
Gauge	7	Muestra un grafico de movimiento

Parámetros de los Ítems del Diplayable Form:

- **Gauge (Barra de Progreso):** Sin parámetros. (Sólo se soporta una instancia Gauge por Formulario)
- **StringItem:**

Tabla 5.11 - Parámetros de StringItem

Parámetro	**Tipo de Dato**
Título	String
Texto	String
Posición	Integer

En la Tabla 5.11 se muestran los parámetros de un StringItem. El valor del título o el valor del texto no son obligatorios. Si falta el Texto, el StringItem se despliega en pantalla en negrita. Si falta el Título, el StringItem se muestra no ennegrecido. Cuando el valor de ambos es no vacío, el título se muestra en negrita y texto en normal, como se ejemplifica en la Tabla 5.12.

Tabla 5.12 - Ejemplo de StringItem

Valor del Título	**Valor del Texto**	**Resultado (Salida por Pantalla)**
Hola	Hola	**Hola** Hola
Hola		**Hola**
	Hola	Hola

Un StringItem puede ser desplegado en distintas posiciones en pantalla (layout). La posición en que se despliega cada StringItem[1] debe ser explicitada. La Tabla 5.13 muestra las posiciones posibles y el código asociado a cada una.

Tabla 5.13 - Posiciones de StringItem

Posición	**Valor (Integer)**	
Layout default	0	No aplica layout
Layout left	1	
Layout right	2	
Layout center	3	

[1] No se establece un número máximo de StringItem para agregar en el Displayable Form, ni tampoco el largo del texto en el.

- **TextField:**

El ítem TextField[1] consta de 4 parámetros, estos parámetros representan la configuración que tendrá este TextField al ser desplegado por pantalla. La Tabla 5.14 detalla los parámetros de este Displayable.

Tabla 5.14 - Parámetros de TextField

Parámetro	Tipo de Dato	Descripción
Etiqueta	String	Nombre del TextField a su izquierda
Contenido	String	Generalmente vacío
MaxSize	Integer	Cantidad de caracteres máxima
Constraints[2]	Integer	Los mismos que para el Displayable TextBox

- **ImageItem:**

Un ImageItem[3] se compone de 3 parámetros, el título de la imagen, su codificación en Base64[4] y su posición (similar a la de un StringItem). Estos parámetros se detallan en la Tabla 5.15.

Tabla 5.15 - Parámetros de ImageItem

Parámetro	Tipo de Dato	Descripción
Label	String	Título de la imagen
Base64	String	Codificación en Base64 de la imagen
Posición	Integer	Layout

Como se menciona anteriormente un ImageItem también posee un atributo de posición (layout), este define la ubicación en pantalla en que esta imagen será desplegada por pantalla, lo que se muestra en la Tabla 5.16 - Posiciones de ImageItemTabla 5.16.

Tabla 5.16 - Posiciones de ImageItem

Posición	Integer
Layout default	0
Layout left	1
Layout right	2
Layout center	3

[1] El número máximo soportado de ítems TextField que un Form puede contener es 4 (cuatro).
[2] Las restricciones o constraints de un ítem TextField son idénticos a los de un Displayable TextBox, véase Tabla 5.8.
[3] No hay un número máximo de ImageItem que pueden estar contenidos en un Form, sin embargo, se recomienda no agregar más allá de 2 (dos) imágenes por Displayable, ya que una imagen pequeña puede incrementar el tamaño del String Generador unas 15 veces.
[4] Base 64 es un sistema de numeración posicional que usa 64 como base.

- **Ticker (Marquesina Móvil):**

La marquesina móvil o Ticker1 se muestra siempre en la parte superior del Displayable, y muestra un texto que avanza por la pantalla. La Tabla 5.17 explica la composición del ítem.

Tabla 5.17 - Parámetros de Ticker

Parámetro	Tipo de Dato	Descripción
Contenido	String	El texto contenido en el ticker en movimiento

- **DateField:**

El ítem DateField es similar al ítem TextField, siendo un DateField un ítem especial para la captura de fechas, está formado por parámetros que indican la Etiqueta del DateField[2] y por el modo de este, explicados en la Tabla 5.18 y en la Tabla 5.19.

Tabla 5.18 - Parámetros de DateField

Parámetro	Tipo de Dato	Descripción
Label	String	Etiqueta superior del DateField
Mode	Integer	Tipo de DateField (siguiente tabla)

Tabla 5.19 - Modos de DateField

Mode	Valor (Integer)
DATE	1
TIME	2
DATETIME	3

- **ChoiceGroup:**

Un ítem ChoiceGroup[3] representa una lista de selección, esta lista está contenida en un Displayable Form y consta de 3 parámetros, el tipo, la etiqueta y la cantidad de elementos u opciones dentro de la lista. Como se detalla en la Tabla 5.20.

Tabla 5.20 - Parámetros de ChoiceGroup

Parámetro	Tipo de Dato	Descripción
Tipo	Integer	Tipo del Choice
Label	String	Etiqueta del ChoiceGroup
Cantidad[4]	Integer	Cantidad de elementos

[1] El número máximo de ítems Ticker soportado por Displayable es 1.
[2] El número máximo de ítems DateField soportado por Displayable es 2.
[3] El número máximo de ítems ChoiceGroup soportado por Displayable es 1.
[4] No se establece un número máximo de opciones dentro de un ChoiceGroup, siendo la cantidad mínima 1.

5.5.2.2 String Respuesta

El String respuesta, es la concatenación de los datos capturados desde la interfaz del Cliente Móvil. Estos datos son recogidos y ordenados para su envío al Nodo, el cual procesa y registra esta información.

En el caso un formulario (Displayable Form), es la concatenación de cada uno de los datos ingresados en el TextField, en el orden de definición separados por un caracter especial y el nombre del comando que se apretó en el Displayable. Se capturan respuestas solo de 3 tipos de Ítems. [TextField, DateField, ChoiceGroup]. Siempre en primer lugar, se concatenan las respuestas de todos los TextField, no importando su orden en el Form, pero sí su orden en el String Generador. Luego, se concatenan todos los DateField que existan en el Form. Por último, se concatenan las respuestas de los ChoiceGroup. Independientemente de si existen TextField, DateField o ChoiceGroup. Siempre al final se debe concatenar el comando apretado. Si no existen TextField ni DateField ni ChoiceGroup el valor de retorno será solo el comando escogido.

En el caso del TextBox, el String Respuesta corresponde a la concatenación del texto ingresado concatenado con un caracter especial más el comando seleccionado en el Displayable.

En el caso de las listas (List), es la concatenación del nombre de la opción seleccionada más un caracter especial y el comando seleccionado.

5.5.3 Interfaz de administración del sistema

La interfaz de Administración del Sistema es la encargada de ejecutar las tareas de Ingreso, Eliminación (desactivación) y Edición de Clientes y Nodos, también es la encargada de asignar, actualizar y/o deprecar Servicios a cada Nodo, esto lo hace comunicándose con la BD XML, creando y/o eliminando colecciones. La tarea de esta interfaz es, en esencia, dejar a disposición del Módulo de Sincronización los archivos en específico, asignados a cada Nodo en particular, convirtiéndose en una herramienta que permite "automatizar" de cierta medida la administración de la Red de Nodos. La Interfaz es desarrollada con Swing/AWT, para así cumplir con el objetivo autoimpuesto de desarrollar el proyecto utilizando únicamente herramientas OpenSource. A continuación se muestra una captura de pantalla de la Interfaz de Administración,

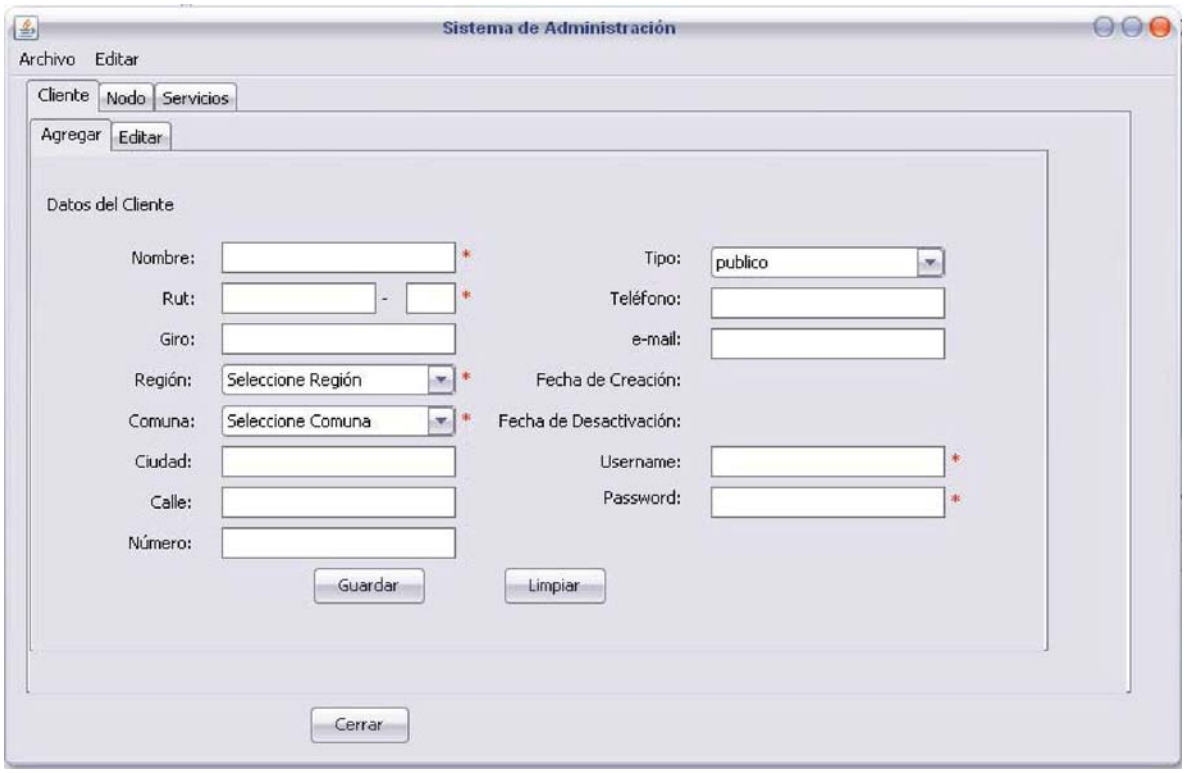

Ilustración 5-15 - Snapshot del Sistema de Administración

5.6 Pruebas

El principal objetivo de las pruebas es evaluar la calidad del producto que se está desarrollando a través de las diferentes fases por las cuales este pasa, mediante la aplicación de pruebas concretas para validar que las suposiciones hechas en el diseño y los requerimientos se estén cumpliendo satisfactoriamente, esto quiere decir que se verifica que el producto funcione como se diseñó y que los requerimientos son satisfechos cabalmente. Esta disciplina brinda soporte para encontrar y documentar (y solucionar) defectos en la calidad del sistema a las otras disciplinas. Las pruebas deben estar presentes en todo el ciclo de vida del desarrollo del sistema para ir refinándolo y no al final del mismo.

Sus objetivos específicos son:

- Encontrar y documentar defectos en la calidad del software.
- Notificar la calidad percibida del software.
- Proveer un medio de validación para las suposiciones hechas en el diseño y especificaciones de requerimientos por medio de demostraciones concretas.
- Validar las funciones del producto de software según lo diseñado.
- Validar que los requerimientos fueron implementados apropiadamente.

El papel de las pruebas no es asegurar la calidad, pero sí evaluarla, y proporcionar una realimentación a tiempo, de forma que los aspectos de calidad puedan resolverse de manera efectiva en tiempo y costo.

Los principales aspectos a ser evaluados en un producto software son la Funcionalidad (hace lo que debe), la Fiabilidad (resistente a fallos), y el Rendimiento (lleva a cabo su trabajo de manera efectiva).

5.6.1 Plan de pruebas

5.6.1.1 Descripción de aspectos generales

El plan de pruebas, fue desarrollado aprovechando al máximo los recursos propios. Dada las limitantes económicas, se desarrollo una herramienta capaz de emular teléfonos móviles con la aplicación cliente, pudiendo programar mediante scripts[1] distintos recorridos que seguiría el usuario ficticio por el sistema. Con dicha herramienta se pudo llegar a simular tantos clientes como puertos USB disponía la maquina además de poder registrar los tiempos de respuesta de cada petición. El ideal es realizar las pruebas con teléfonos móviles reales, y computadores actuales.

Objetivo: El objetivo del plan de pruebas es testear el correcto funcionamiento del sistema y sus capacidades.

Arquitectura técnica:

Se dispone de 3 máquinas de prueba:

- Máquina Nodo Test: en su interior esta el nodo en ejecución, esperando clientes.

[1] Un script es un guión o conjunto de instrucciones. Permiten la automatización de tareas creando pequeñas utilidades.

- Máquina Simulación 1: Máquina con el cliente modificado capaz, de crear 4 conexiones.
- Máquina Simulación 2: Máquina con el cliente modificado capaz, de crear 4 conexiones.

Especificaciones de Hardware:

- Máquina Nodo Test: Pentium IV 1.8 Gz, 512 RAM. Debian GNU/Linux 4.0
- Máquina Simulación 1: Pentium III 800 mhz, 128 RAM. Debian GNU/Linux
- Máquina Simulación 2: Pentium III 800 mhz, 128 RAM. Debian GNU/Linux

5.6.1.2 Definición estrategia de pruebas

Objetivo: Demostrar que el Nodo funciona de acuerdo a las especificaciones.

Técnica: Para llevar acabo esta prueba se utilizara teléfono móvil, se comprobara, que el usuario pueda:

- Detectar el nodo.
- Conectarse al nodo.
- Interactuar con los servicios locales
- Interactuar con los servicios remotos (utilizan Web Services)
- Registrar los datos en los documentos de respuesta.
- Sincronizar adecuadamente con la base de datos.

Casos de prueba:

- El teléfono móvil, ejecuta la aplicación, y comienza a detectar dispositivos.
- Se conecta al nodo, comienza a interactuar por cada uno de los servicios.
- Rellena datos en los servicios creados para testear el registro de información.

Criterios de término: Para que la prueba sea exitosa, en la base de datos, deberán estar todos y cada uno de los datos, registrados por el usuario en cada uno de los documentos de respuesta relativos a los servicios que este uso. De lo contrario, la prueba habrá fracasado.

Recursos requeridos:

- Maquina Nodo Test con 1 dispositivo Bluetooth.
- Un teléfono con Bluetooth (API JSR-82 incluida)

Resultados:

- Se demostró que el sistema funciona correctamente, además muestra mucha estabilidad.

5.6.2 Estrés

Objetivo: Demostrar que el nodo del sistema es capaz de soportar y manejar 7 conexiones simultáneas.

Técnica: Para llevar a cabo la prueba, se hará uso de una herramienta especial una versión del MIDLet modificada para funcionar sobre J2SE, capaz de solicitar al nodo distintos servicios, de acuerdo a un script creado previamente, entregar y registrar los tiempos de respuesta de cada petición.

Casos de prueba: Utilizando la herramienta desarrollada para emular clientes, en la maquina de simulación 1, se lanzaron 4 clientes simultáneos.

- En la máquina de simulación 2, se lanzaron 3 clientes.
- Al conectar el dispositivo numero siete, el nodo entra en modo oculto.

Criterios de término: La prueba será exitosa si, al conectar el séptimo dispositivo, el dispositivo se oculta. Al comprobar vía línea de comandos la cantidad de enlaces Bluetooth creados y si las 7 conexiones se mantienen estables durante la ejecución del script de simulación.

Recursos requeridos:

- Máquina Nodo Test
- Máquina Simulación 1
- Máquina Simulación 2

Resultado: Efectivamente, se comprobó que el sistema es capaz de manejar 7 conexiones. Los tiempos de respuestas observados están por el orden de los 500 ms. en promedio.

5.7 Desarrollo de las Fases de UP

El ciclo de vida del proyecto de software se inspira en UP, motivo por el cual se descompone en el tiempo en cuatro fases secuenciales como se muestra abajo en la Ilustración 5-16 - Fases del Proceso Unificado, que son:

- Inicio
- Elaboración
- Construcción
- Transición

Al final de cada fase el equipo gestor del proyecto realiza una evaluación para determinar si los objetivos se cumplieron y así pasar a la fase siguiente.

Ilustración 5-16 - Fases del Proceso Unificado

5.7.1 Fase de Inicio

Su propósito general es establecer los objetivos para el ciclo de vida del sistema de software. Durante esta fase se definió el Modelo del Negocio[1], el Objetivo General y los Objetivos Específicos[2] del proyecto. Se identificaron los actores principales y casos de uso.

Los objetivos específicos de esta fase fueron:

- Establecer el ámbito del proyecto y sus límites.
- Encontrar los casos de uso críticos del sistema, los escenarios básicos que definen la funcionalidad.
- Definir al menos una arquitectura candidata para los escenarios principales.
- Estimar el costo en recursos y tiempo de todo el proyecto.
- Estimar los riesgos y las fuentes de incertidumbre.

Esta fase se realizó en 2 iteraciones y finalizó con el establecimiento del ámbito del proyecto, e identificación de los principales riesgos.

[1] Véase Capítulo 5 Sección 5.1
[2] Véase Capítulo 2

5.7.2 Fase de Elaboración

Su objetivo general es plantear la arquitectura para el ciclo de vida del software. Se construyó un modelo de la arquitectura, que se desarrolló en iteraciones sucesivas hasta obtener un prototipo funcional, este prototipo contiene los casos de uso críticos que fueron identificados en la fase de inicio. En esta fase se realizó la captura de la mayor parte de los requerimientos funcionales, manejando los riesgos que interfieran con los objetivos del sistema, acumulando la información necesaria para el plan de construcción y obteniendo suficiente información para hacer realizable el modelo del negocio.

Los objetivos específicos de esta fase fueron:

- Definir, validar y establecer la arquitectura.
- Crear un plan fiable para la fase de construcción.
- Demostrar que la arquitectura propuesta soportará los objetivos del proyecto con un costo razonable y en un tiempo razonable.

La fase de elaboración se realizó en 3 iteraciones y finalizó con la obtención de una base de la arquitectura del sistema, la captura de la mayoría de los requerimientos y la reducción de los riesgos importantes.

5.7.3 Fase de Construcción

El objetivo general de esta fase es alcanzar la capacidad operacional del software de forma incremental a través de las sucesivas iteraciones. En esta fase todas las características, componentes, y requerimientos fueron integrados, implementados, y probados, obteniendo una versión aceptable del software o versión Beta.

Los objetivos específicos de esta fase fueron:

- Minimizar los costos de desarrollo evitando el tener que rehacer un trabajo o desecharlo.
- Conseguir versiones funcionales (alfa, beta, y otras versiones de prueba) lo más rápido posible.

Esta fase se desarrolló en 3 iteraciones y culminó con el desarrollo del sistema en una versión Beta.

5.7.4 Fase de Transición

Tuvo como objetivo general entregar el producto funcional en manos de los usuarios finales una vez realizadas las pruebas, y en general tareas relacionadas con la configuración, instalación y usabilidad del software.

Los objetivos específicos de esta fase fueron:

- Garantizar que el usuario aprenda a operar el sistema.
- Conseguir un producto final que cumpla los requerimientos esperados.

La fase de transición se completó en 2 iteraciones y corresponden a haber decidido que los objetivos se cumplieron.

Capítulo 6 Conclusiones

En la sociedad actual los sistemas de computación sólo se pueden entender como un conjunto de ordenadores conectados de alguna manera entre sí. Esto se refleja claramente en el desarrollo de este proyecto, que, en palabras simples es un conjunto de dispositivos (computadores, servidores y teléfonos móviles) conectados entre sí para cumplir la tarea encomendada. La tarea de entregar a los usuarios de los teléfonos móviles información fácilmente y adecuada al contexto espacial de estos .

El desarrollo de este proyecto ha pasado por las distintas etapas que la ingeniería de software define para llegar a un software de calidad, el resultado del proceso fue sin duda exitoso. El objetivo general del proyecto de generar un sistema de software capaz de proporcionar al cliente móvil acceso a contenidos de su interés fue logrado cabalmente. Añadiendo a lo anterior, el cumplimiento íntegro de los objetivos específicos planteados, como la generación de mecanismos de resguardo del flujo de información, la facilitación de acceso a información relevante para los usuarios de teléfonos móviles, el diseñó de una arquitectura que permitiera el desarrollo del proyecto, etc. desembocando en una implementación de un prototipo funcional del sistema.

Este desarrollo sin duda provee de una potente y versátil herramienta de marketing que puede transformarse en una nueva forma de llegar al público de manera directa y personalizada, facilitando y acercando información a los usuarios, sin necesidad para ellos de un equipo y conexión a Internet. El “Boom” de las tecnologías móviles potencia e impulsa las proyecciones para el sistema pues la tasa de recambio en telefonía celular es muy elevada, lo que ayudaría a la masificación y expansión de su uso.

Todo lo anterior, considerando la utilización de herramientas Open Source y/o GPL, lo que demuestra que es posible desarrollar proyectos innovadores bajo la premisa del software libre.

Los desafíos que el proyecto propuso y la forma en que se resolvieron, muestran claramente que es posible desarrollar proyectos de investigación, proyectos que impliquen desafíos mayores que desarrollar un Sistema de Información tradicional y pongan a prueba las habilidades del grupo de trabajo. En este sentido el desarrollo de este sistema ejemplifica que con esfuerzo e imaginación, el mundo de las TI's abre un gran espectro de posibilidades de nuevos desarrollos como este, que lleven por el camino de la investigación e integración de nuevas tecnologías.

Las proyecciones futuras apuntan al desarrollo de nuevas funcionalidades, la maduración de una plataforma que pueda brindar soporte al sistema, la integración con más sistemas externos que abran el abanico de posibilidades para el crecimiento de este y la implementación de desarrollos a la medida.

Referencias

[Apache]	http://www.apache.org
[Bluetooth Dev]	http://developers.sun.com/techtopics/mobility/midp/articles/Bluetooth1/
	http://developers.sun.com/techtopics/mobility/midp/articles/Bluetooth2/
[Bluetooth]	http://www.bluetooth.org
[Debian Doc]	http://www.debian.org/doc/
[Debian]	http://www.debian.org
[Dom4J]	http://www.dom4j.org
[J2ME]	http://www.java.com/es/download/faq/whatis_j2me.xml
	http://java.sun.com/javame/index.jsp
[James Martin, James J.Odell.]	Object-Oriented Analysis and Design. Prentice Hall, Englewood Cliffs, NJ, 1992.
[JSR-82]	http://jcp.org/en/jsr/detail?id=82
[Kenneth P. Birman]	Reliable Distributed Systems: Technologies, Web Services, and Applications. Springer-Verlag, 2005
[NetBeans]	http://www.netbeans.org
	http://www.netbeans.org/kb/articles/mobility.html
[Rumbaugh J., Jacobson I., Booch G., 2000]	Proceso Unificado de Desarrollo del Software, Addison Wesley, Madrid, 2000.
[Sourceforge]	http://sourceforge.net/
[Tomcat]	http://tomcat.apache.org/tomcat-5.5-doc/index.html
[UML]	http://www.uml.org/
[W3C]	http://www.w3.org
[Wikipedia]	http://es.wikipedia.org
[XML]	http://www.w3.org/XML/
[xml-db]	http://xmldb-org.sourceforge.net/
[XMLRPC]	http://www.xmlrpc.com

Glosario de Términos

- Alcance: Área que cubre la señal de radio Bluetooth. Puede variar dependiendo de distintos factores.
- Anfitrión Bluetooth: Un anfitrión Bluetooth es un dispositivo electrónico, como un periférico informático, un teléfono móvil o un punto de acceso a una red pública de telefonía conmutada (PSTN) o de área local (LAN). El anfitrión Bluetooth está conectado al controlador Bluetooth y puede comunicarse con otros dispositivos con controladores Bluetooth.
- Área de cobertura: Radio en el que dos dispositivos con tecnología Bluetooth pueden intercambiar mensajes con calidad y ofrecer un buen rendimiento.
- Baliza: Patrón de ranuras reservadas en un canal básico o adaptado de una piconet. Las transmisiones que se inician en estas ranuras se utilizan para volver a sincronizar dispositivos en espera (modo park).
- Banda base Bluetooth: Parte del sistema Bluetooth que especifica e implementa los procedimientos seguidos en las capas físicas y en los canales de acceso para permitir el intercambio de flujos de datos y voz en tiempo real, así como la creación de redes ad-hoc entre dispositivos con tecnología Bluetooth.
- BD_ADDR: Denominación técnica o abreviatura utilizada para referirse a la dirección de los dispositivos Bluetooth. La dirección BD_ADDR permite identificar los dispositivos Bluetooth.
- Búsqueda: También llamado detección, es un proceso por el que un dispositivo Bluetooth transmite mensajes de búsqueda y espera la respuesta para encontrar otros dispositivos compatibles en su área de cobertura.
- Canal físico de la piconet: Canal dividido en ranuras de tiempo donde cada ranura se asocia a una frecuencia de salto RF. Los saltos consecutivos suelen corresponder a diferentes saltos de frecuencia RF. Siguen un patrón de salto pseudo aleatorio entre los 79 canales de radiofrecuencia y normalmente presentan una velocidad nominal de 1.600 saltos/segundo.
- Canal físico: Canal caracterizado por estar ocupando, de forma sincronizada, una secuencia de portadoras RF en uno o más dispositivos. Existen diferentes tipos de canales físicos y, dependiendo de su propósito, tienen unas características u otras.
- Canal L2CAP: Conexión lógica en la capa L2CAP entre dos dispositivos que comparten una aplicación o un protocolo de capa superior.
- Canal lógico: Idéntico al canal L2CAP pero relegado debido a que tenía un significado distinto en la versión 1.1 del estándar Bluetooth.
- Canal: Los canales pueden ser físicos o L2CAP dependiendo del contexto.

- Cifrado: Método de codificación de los datos para evitar que otros puedan interpretarlos.
- Clave de enlace: Clave secreta conocida únicamente por dos dispositivos y que utilizan para autenticarse entre sí.
- Comunicación lógica: En la tecnología inalámbrica Bluetooth se utiliza para representar a los distintos enlaces lógicos que comparten protocolos de confirmación e identificadores de enlace.
- Conectar (a un servicio): Proceso de establecimiento de una conexión para utilizar un servicio. Al entablarse, se crea un enlace físico, una comunicación y enlace lógicos, así como un canal L2CAP, si éstos no están presentes.
- Conexión: Unión o enlace entre dos aplicaciones iguales o protocolos de capas superiores idénticos asignada a un canal L2CAP.
- Contraseña: Al emparejar dispositivos, se recomienda utilizar contraseñas para autentificar las conexiones entrantes. Además, en determinados casos, deberá asegurarse de que se conecta al dispositivo o persona deseados. La contraseña consiste en una combinación de códigos alfanuméricos. Utilícelas con precaución, ya que algunos dispositivos no transmiten los caracteres del mismo modo. Las contraseñas sólo son válidas para una conexión concreta y pueden ser diferentes para otros dispositivos o usuarios.
- Controlador Bluetooth: Subsistema que engloba la capa RF, la banda base, el controlador de recursos, el gestor de enlaces, el gestor del dispositivo y la interfaz HCI.
- Datos isócronos: Flujo de información donde cada entidad tiene una relación temporal con las entidades anteriores y posteriores que conforman el flujo.
- Descubrimiento de dispositivos: Procedimiento para acceder a los campos de dirección del dispositivo Bluetooth, reloj y tipo de dispositivo, así como el procedimiento de paginación de los dispositivos detectados.
- Descubrimiento de servicios: Conjunto de procedimientos para buscar y explorar los servicios ofrecidos por un dispositivo Bluetooth o por otro aparato al que se accede mediante un dispositivo Bluetooth.
- Descubrimiento del nombre: Procedimiento para obtener un nombre más familiar del dispositivo Bluetooth susceptible de ser conectado.
- Detección de búsqueda: También llamado exploración. Proceso por el que un dispositivo Bluetooth comprueba si le han llegado mensajes de búsqueda al canal físico correspondiente.
- Detección de paginación: También llamado exploración. Proceso por el que un dispositivo Bluetooth comprueba si le han llegado mensajes de paginación al canal físico correspondiente.

- Difusión del dispositivo esclavo activo (ASB): La comunicación lógica ASB se utiliza para transferir datos L2CAP del usuario a todos los dispositivos activos de la piconet.
- Dirección del dispositivo Bluetooth: Dirección de 48 bits exclusiva y con la que se identifica cada dispositivo Bluetooth. A menudo, suele aparecer en las fichas o datos técnicos de la siguiente forma: BD_ADDR.
- Display: gestiona qué objeto Displayable se mostrará al usuario.
- Displayable: Clase nativa de la Plataforma J2ME, que representa la interfaz gráfica capas de desplegarse por pantalla.
- Dispositivo con tecnología Bluetooth: Un dispositivo Bluetooth es aquél que puede mantener una comunicación inalámbrica de corto alcance empleando para ello la especificación Bluetooth.
- Dispositivo conocido: Dispositivo Bluetooth del que se conoce al menos la dirección BD_ADDR.
- Dispositivo de detección: Dispositivo Bluetooth que está llevando a cabo un proceso de búsqueda o detección.
- Dispositivo de paginación: Dispositivo Bluetooth que está llevando a cabo un proceso de paginación o conexión.
- Dispositivo desconocido: Se trata de un dispositivo equipado con tecnología Bluetooth del que no se tiene información guardada (dirección del dispositivo Bluetooth, clave de enlace, etc.).
- Dispositivo emparejado: Dispositivo Bluetooth con el que se ha intercambiado una clave de enlace, bien antes de solicitar el establecimiento de la conexión o durante ésta.
- Dispositivo en espera: Dispositivo con funcionamiento básico dentro de una piconet, sincronizado con el maestro pero sin comunicación lógica ACL predeterminada. También llamado dispositivo en modo park.
- Dispositivo silenciado: Dispositivo Bluetooth que no responde a la búsqueda realizada por un dispositivo remoto.
- Dispositivo susceptible de ser detectado: Se denomina así a los dispositivos Bluetooth que comprueban periódicamente su canal físico de búsqueda y que responderán a las solicitudes que reciban por dicho canal. Normalmente, los dispositivos susceptibles de ser detectados también son susceptibles de ser conectados.
- Emparejamiento: Proceso de establecimiento de un nuevo vínculo entre dos dispositivos equipados con tecnología Bluetooth durante el que se intercambia una clave de enlace, bien antes de solicitar el establecimiento de la conexión o durante ésta.

- Enlace físico: Conexión a nivel de banda base entre dos dispositivos establecida por un proceso de paginación.
- Enlace lógico: Nivel más bajo de la arquitectura. Ofrece servicios de transmisión de datos de forma independiente a los clientes dentro del sistema Bluetooth.
- Enlace: Forma abreviada de enlace lógico.
- Esclavo de la piconet: Cualquier dispositivo de una piconet que no actúa de maestro, sino que está conectado a éste.
- Establecimiento de enlace: Procedimiento para establecer un enlace ACL predeterminado y la jerarquía de enlaces y canales entre los dispositivos.
- Establecimiento de la conexión: Procedimiento para crear una conexión asignada a un canal.
- Establecimiento de una conexión de confianza: Procedimiento por el que un dispositivo remoto se marca o identifica como dispositivo de confianza. Durante el proceso se guarda una clave de enlace común para su autenticación y emparejamiento posterior (si no se utiliza ninguna clave de enlace concreta).
- Establecimiento de una conexión segura: Procedimiento por el que se crea una conexión con fases de autenticación y cifrado.
- Establecimiento del canal L2CAP: Procedimiento para establecer una conexión lógica en la capa L2CAP.
- Fase de conexión: Fase de comunicación entre dos dispositivos en la que se está estableciendo una conexión entre ambos. Se inicia una vez finalizada la creación del enlace.
- HCI Bluetooth: Interfaz de comandos entre el controlador de la banda base y el gestor de enlaces. Permite acceder al estado y los registros de control del aparato, además de proporcionar un método uniforme de acceder a todas las funciones de la banda base Bluetooth.
- Maestro de la piconet: Dispositivo de una piconet cuyo reloj y dirección de dispositivo Bluetooth se utilizan para definir las características del canal físico de la piconet.
- Paginación: Fase inicial del procedimiento de conexión donde un dispositivo envía varios mensajes hasta que recibe una respuesta del dispositivo receptor o expira el tiempo de espera establecido.
- Paquete: Formato de datos en forma de bits que se transmite a través de un canal físico.
- Participante en varias piconets (PMP): Dispositivo que forma parte de varias piconet a la vez, lo que se consigue mediante la multiplexación de división de tiempo (TDM). Así, puede compaginar su actividad en el canal físico de cada piconet.

- Perfil básico de imagen (BIP): El perfil BIP establece cómo puede controlarse remotamente un dispositivo de imagen, así como la forma de enviarle órdenes de impresión y de transferencia de imágenes a un dispositivo de almacenamiento. Un ejemplo de este perfil es un teléfono móvil que se utiliza para controlar el obturador de una cámara digital.
- Perfil básico de impresión (BPP): El perfil BPP permite enviar mensajes de texto, de correo electrónico, tarjetas de visita electrónicas e imágenes, entre otras cosas, a las impresoras disponibles dependiendo de las tareas de impresión. A diferencia del perfil HCRP, no requiere controladores de impresión específicos. Por ello, es más indicado para dispositivos que no pueden actualizarse fácilmente con controladores específicos de cada fabricante, como móviles o cámaras digitales.
- Perfil de acceso SIM (SAP): El perfil SAP permite a dispositivos con transmisores GSM integrados, como los sistemas de teléfono de los automóviles, conectarse a la tarjeta SIM de un móvil equipado con tecnología Bluetooth. De esta forma, el teléfono del automóvil no requiere ninguna otra tarjeta SIM.
- Perfil de aplicación de descubrimiento de servicio (SDAP): El perfil SDAP detalla cómo una aplicación debe utilizar el perfil SDP para identificar los servicios de un dispositivo remoto. El perfil SDAP se ocupa de que cualquier aplicación pueda localizar los servicios disponibles en el dispositivo conectado.
- Perfil de dispositivo de interfaz humana (HID): El perfil HID recoge los protocolos, procedimientos y características empleados por las interfaces de usuario Bluetooth compatibles tales como teclados, dispositivos punteros, consolas o aparatos de control remoto.
- Perfil de distribución de audio avanzado (A2DP): El perfil A2DP describe cómo transferir sonido estéreo de alta calidad de una fuente de sonido a un dispositivo receptor. El perfil hace la distinción, pues, entre fuente de sonido y receptor. Un caso típico sería un reproductor de sonido tipo “discman”. La fuente de sonido sería el reproductor y el receptor del audio sería un auricular inalámbrico. El perfil A2DP establece los protocolos y procedimientos que realizarán la distribución del contenido de audio de alta calidad en modo mono o estéreo a través de los canales ACL.
- Perfil de distribución de vídeo (VDP): Este perfil dicta los pasos que deben seguir los dispositivos con tecnología Bluetooth para la transferencia continua de vídeos. Esto permite, por ejemplo, visualizar en reproductores portátiles los vídeos almacenados en aplicaciones multimedia de cualquier ordenador o transferir las imágenes desde una videocámara digital a la televisión.
- Perfil de intercomunicador (ICP): Al igual que los ruidos pueden dificultar a veces la comunicación humana, las radiofrecuencias con tecnología Bluetooth pueden resultar inaudibles debido a las interferencias con otras radios. Esta cuestión constituye un motivo de preocupación especial, ya que la tecnología inalámbrica Bluetooth utiliza una banda que no requiere licencia. Afortunadamente, fue diseñada para no producir demasiado ruido y para evitar cualquier tipo de interferencia. Entre los dispositivos que pueden afectar a los productos con

tecnología Bluetooth se encuentran los hornos microondas y algunos modelos de teléfonos inalámbricos.

- Perfil de sincronización (SYNC): El perfil SYNC se utiliza junto al GOEP para sincronizar los elementos del administrador de información personal (PIM), como agendas y datos de contacto, entre dispositivos con tecnología Bluetooth. Se utiliza, por ejemplo, al intercambiar datos entre una PDA y un ordenador.
- Perfil de transferencia de archivos (FTP): El perfil FTP establece los procedimientos de exploración de carpetas y archivos de un servidor a través de un dispositivo cliente. Una vez que el cliente localiza el archivo en el servidor, puede copiarlo. El cliente también puede enviar archivos al servidor mediante el perfil GOEP.
- Perfil WAP sobre Bluetooth (WAP): El perfil WAP describe cómo aplicar el conjunto de protocolos de la aplicación en la tecnología inalámbrica Bluetooth. Se utiliza, por ejemplo, al conectar un teléfono móvil a un punto de acceso público para explorar contenidos a través de portales WAP. Es compatible con distintas tecnologías WAN y proporciona acceso a Internet desde dispositivos móviles.
- Perfiles Bluetooth: Estos perfiles son guías que indican los procedimientos por los que los dispositivos equipados con tecnología Bluetooth se comunican entre sí. Existe un amplio abanico de perfiles que detallan los diferentes tipos de uso y aplicaciones de la tecnología inalámbrica Bluetooth. Para que un dispositivo pueda utilizar la tecnología inalámbrica Bluetooth, debe saber interpretar los perfiles Bluetooth que describen las distintas aplicaciones posibles.
- Piconet: Grupo de dispositivos que comparten un mismo canal físico donde uno de ellos actúa de maestro y el resto se conecta a él.
- PIN: Número fácil de recordar que puede utilizarse para autenticar la conexión con un dispositivo antes de que tenga lugar el emparejamiento.
- Protocolo de capas de servicios: Protocolo que utiliza el canal L2CAP para transportar unidades PDU.
- Protocolo de intercambio de Objetos (OBEX): OBEX es un protocolo de transferencia que define los objetos de datos y el protocolo de comunicaciones que deben utilizar dos dispositivos para intercambiar objetos. También permite a las aplicaciones operar en la pila de protocolos Bluetooth y en la pila de protocolos de infrarrojos (IrDA). En los dispositivos Bluetooth, sólo se admiten conexiones OBEX para el intercambio de objetos. Se han desarrollado tres perfiles de aplicaciones basadas en el protocolo OBEX: SYNC, FTP y OPP.
- Red ad-hoc: Red que se crea espontáneamente. No requiere una infraestructura fija y está limitada en el espacio y en el tiempo.
- Reloj Bluetooth: Reloj interno de 28 bits del subsitema controlador Bluetooth, que se mueve a una velocidad de 312,5 ms. Se utiliza como referencia para numerar y programar las ranuras de varios canales físicos.
- Scatternet: Dos o más piconets con uno o varios dispositivos PMP.

- Tecnología inalámbrica Bluetooth: La tecnología inalámbrica Bluetooth es un enlace de comunicación inalámbrica que funciona en la banda de 2.4 GHz, una de las bandas de radio industrial, científica y médica (ISM) que no requiere licencia. Permite la transmisión de datos, vídeo y audio en tiempo real entre dispositivos con tecnología Bluetooth. El protocolo de enlace utiliza ranuras de tiempo para las comunicaciones.

Lista de Abreviaturas

3DES: Triple DES

ACL: Asynchronous Connection-Less

ADSL: Asymmetric Digital Subscriber Line

AES: Advanced Encryption Standard

API: Application Programming Interface

ASA: American Standard Asociation

BLOB: Binary Large Object

CA: Certificate Authority ó Certification Authority

CDATA: Character Data

CDC: Connected Limited Configuration

CLDC: Connected Limited Device Configuration

CORBA: Common Object Requesting Broker Architecture

CSS: Cascading Style Sheets

CVM: Compact Virtual Machine

DBMS: Database Management System

DCOM: Distributed Component Object Model

DES: Data Encryption Standard

DOM: Document Object Model

DSRC: Dedicated short-range communications

DTD: Document Type Definition

EDI: Electronic Data Interchange

EDR: Enhanced Data Rate

EJB: Enterprise JavaBeans

ETSI: European Telecommunications Standards Institute

FHSS: Frequency-hopping Spread Spectrum

FTP: File Transfer Protocol

FURPS: Functionality, Usability, Reliability, Performance, Supportability

GNU: Acrónimo recursivo que significa GNU No es Unix (GNU is Not Unix)

GPL: General Public Licence

GPRS: General Packet Radio Service

GSM: Groupe Spécial Mobile
GUI: Graphical User Interface
HCI: Host Controller Interface
HTTP: HiperText Transport Protocol
IDE: Integrated Development Environment
IEEE: Institute of Electrical and Electronics Engineers
IP: Internet Protocol
IPSEC: Internet Protocol Security
IrDA: Infrared Data Association
ISM: Industrial, Scientific and Medical
ISO: International Organization for Standardization
ISP: Internet Service Provider
J2EE: Java 2 Enterprise Edition
J2ME: Java 2 Micro Edition
J2SE: Java 2 Standard Edition
JDBC: Java Database Connectivity
JSR: Java Community Process
JVM: Java Virtual Machine
KVM: Kilo Virtual Machine
L2CAP: Logical Link Control and Adaptation Protocol
L2F: Layer 2 Forwarding
L2TP: Layer 2 Tunneling Protocol
LAN: Local Area Network
LMP: Link Manager Protocol
MAC: Media Access Control
MAN: Metropolitan Area Network
MathML: Mathematical Markup Language
MD2: Message-Digest Algorithm 2
MD5: Message-Digest Algorithm 5
MIDP: Mobile Information Device Profile
NFC: Near Field Communication
OASIS: Organization for the Advancement of Structured Information Standards
OBEX: OBject EXchange

OFDM: Orthogonal frequency-division multiplexing

OO: Orientación a Objetos

PDA: Personal Digital Assistant

PHY: Physical Layer

POP3: Post Office Protocol 3

PPTP: Point to Point Tunneling Protocol

RAM: Random Access Memory

REST: Representational State Transfer

RF: Radio Frecuency

RMI: Java Remote Method Invocation

ROM: Read-Only Memory

RPC: Remote Procedure Call

SAX: Simple API for XML

SGML: Standard Generalized Markup Language

SHA: Secure Hash Algorithm

SMS: Short Message Service

SMTP: Simple Mail Tranfer Protocol

SOAP: Simple Object Access Protocol

SSH: Secure Shell

SVG: Scalable Vector Graphics

TCP: Transmisión Control Protocol

TDD: Time-Division Duplex

TIC: Tecnologías de la Información y la Comunicación

TLS: Transport Layer Security

TPC: Transmit Power Control

UDDI: Universal Description, Discovery and Integration

UDP: User Datagram Protocol

UI: User Interface

UML: Unified Modeling Language

UMTS: Universal Mobile Telecommunications System

UP: Unified Process

USB: Universal Serial Bus

UTF: Unicode Transformation Format

UWB: Ultra Wide Band

VPN: Virtual Private Network

W3C: World Wide Web Consortium

WAP: Wireless Application Protocol

WIFI: marca de la Wi-Fi Alliance (anteriormente la WECA: Wireless Ethernet Compatibility Alliance), la organización comercial que adopta, prueba y certifica que los equipos cumplen los estándares 802.11.

WLAN: Wireless LAN

WSDL: Web Service Description Language

XACML: eXtensible Access Control Markup Language

XHTML: Extensible Hypertext Markup Language

XML: Extensible Markup Language

XPath: XML Path Language

XQJ: XQuery API for Java

XQuery: XML Query Language

XSLT: Extensible Stylesheet Language Transformations

Anexo A. Bluetooth

A.1 ¿Qué es Bluetooth?

La tecnología inalámbrica Bluetooth es un sistema de comunicaciones de corto alcance, cuyo objetivo es eliminar los cables en las conexiones entre dispositivos electrónicos, tanto portátiles como fijos. Las características principales de esta tecnología son su fiabilidad, bajo consumo y mínimo coste. Varias de las funciones de la especificación principal son opcionales, lo que permite la diferenciación de los productos.

El núcleo del sistema Bluetooth consiste en un transmisor de radio, una banda base y una pila de protocolos. El sistema permite la conexión entre dispositivos y el intercambio de distintos tipos de datos entre ellos.

A.2 Descripción general del funcionamiento

La capa física de radio (RF) opera en la banda de 2.4 GHz libre para ISM (banda de frecuencia industrial, científica y médica). El sistema emplea un transmisor de salto de frecuencia para restar las interferencias y la pérdida de intensidad, y cuenta con gran número de portadoras de espectro ensanchado por salto de frecuencia (FHSS). Para minimizar la complejidad del transmisor, se utiliza una modulación de frecuencia binaria. La velocidad de símbolo es de 1 MS/s (mega símbolo por segundo), que admite una velocidad de transmisión de 1 Mbps en el modo de velocidad básica y una velocidad de transmisión aérea total de 2 a 3 Mbps en el modo de transferencia de datos mejorada (EDR).

Normalmente, varios dispositivos sincronizados por un reloj y una secuencia de salto de frecuencia comparten el mismo canal físico de radio. Uno de ellos proporciona los valores de referencia, el denominado dispositivo maestro. Los demás reciben el nombre de esclavos. Este tipo de conexión entre dispositivos es lo que se conoce como una piconet, la forma de comunicación básica en la tecnología inalámbrica Bluetooth, ejemplificada en la Ilustración A-1.

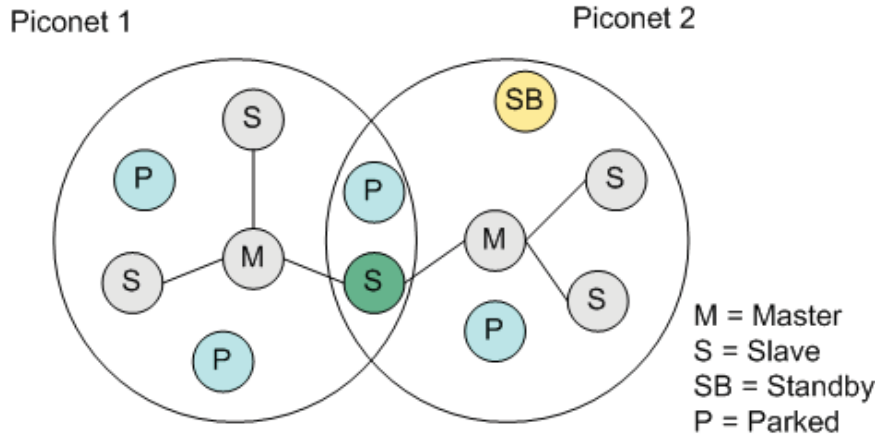

Ilustración A-1 - Piconets Bluetooth

Los dispositivos de una piconet utilizan una secuencia de salto de frecuencia determinada por algoritmos en ciertos campos del reloj y de la dirección de especificación Bluetooth del maestro. La secuencia básica de salto consiste en un ordenamiento pseudo aleatorio de las 79 frecuencias de la banda ISM. Esta secuencia puede adaptarse para excluir la sección de frecuencias utilizadas por los dispositivos que están causando interferencias. La técnica de salto adaptable mejora la coexistencia de la tecnología Bluetooth con los sistemas ISM estáticos (es decir, sin salto) cuando éstos se encuentran localizados.

El canal físico se subdivide en unidades de tiempo denominadas ranuras. Los datos intercambiados entre los dispositivos Bluetooth se transmiten en forma de paquetes que se emplazan en estas ranuras. Cuando la situación lo permite, se pueden asignar varias ranuras consecutivas a un único paquete. El salto de frecuencia se produce durante la transmisión o recepción de los paquetes. La tecnología Bluetooth consigue la transmisión bidireccional mediante la técnica de acceso múltiple o dúplex por división de tiempo (TDD).

Sobre el canal físico hay una capa de enlaces y canales con sus respectivos protocolos de control. La jerarquía de abajo a arriba de los canales y enlaces es la siguiente: canal físico, enlace físico, comunicación lógica, enlace lógico y canal L2CAP.

En el canal físico, se forma un enlace físico entre dos dispositivos que se intercambian paquetes, sea cual sea la dirección. Pero no todos los dispositivos pueden conectarse mediante un enlace físico dentro de la piconet. Se crea un enlace de este tipo entre un dispositivo esclavo y el maestro, pero dos esclavos no pueden conectarse directamente de esta forma.

El enlace físico se utiliza como medio de comunicación entre uno o dos enlaces lógicos que admiten tráfico síncrono, asíncrono e isócrono de unidifusión, y tráfico de difusión. El tráfico de los enlaces lógicos se multiplexa en el enlace físico ocupando las ranuras asignadas por el programador del gestor de recursos.

A través de los enlaces lógicos, además de los datos del usuario se transporta el protocolo de control de la banda base y las capas físicas: protocolo de gestión de enlace (LMP). Los dispositivos que están activos dentro de la piconet tienen una comunicación lógica asíncrona predeterminada para el transporte de la señalización del protocolo LMP. Es lo que se conoce como comunicación lógica ACL. Esta comunicación es la que se establece cuando un dispositivo se une a una piconet. Se pueden crear comunicaciones lógicas adicionales si resulta necesario para transportar el flujo de datos síncronos.

El gestor de enlaces se sirve del protocolo LMP para controlar el funcionamiento de los dispositivos en la piconet y proporcionar servicios de gestión en las capas inferiores de la arquitectura: capa de radio y de banda base. El protocolo LMP sólo se transporta a través de la comunicación lógica ACL y la comunicación lógica de difusión predeterminadas.

La capa L2CAP está por encima de la de banda base y se encarga de ofrecer una abstracción de canales de comunicación a las aplicaciones y los servicios. Realiza la segmentación y la unificación de los datos de las aplicaciones y la multiplexación y demultiplexación de varios canales a través de un enlace lógico compartido. La capa L2CAP dispone de un canal de control de protocolos a través de la comunicación lógica ACL predeterminada. Los datos de la aplicación enviados al protocolo L2CAP pueden transferirse a través de un enlace lógico compatible con el protocolo L2CAP.

A.3 Comparación con Otras Tecnologías

El mundo de las conexiones inalámbricas sigue creciendo a un ritmo frenético con el desarrollo de tecnologías más seguras y fiables por parte de los ingenieros. Estas tecnologías nos permiten soltar amarras y olvidarnos de los cables creando entornos más sencillos, eficaces y prácticos; un escenario que ha pasado a formar parte de nuestra vida cotidiana. La tecnología Bluetooth es sólo una de las opciones inalámbricas disponibles, aunque tiene una gran variedad de aplicaciones. Resulta muy útil compararla con otras tecnologías para decidir cuál es la más apropiada para una aplicación concreta o a la hora de adquirir un nuevo dispositivo.

A.3.1 Tecnología inalámbrica Bluetooth

La tecnología inalámbrica Bluetooth está orientada a aplicaciones de voz y datos. Funciona en la banda de frecuencia de 2.4 GHz, que no precisa de ninguna licencia. Tiene un radio de acción de 10 o 100 metros dependiendo de la clase del dispositivo Bluetooth. La máxima velocidad de transmisión es de 3 Mbps. Los objetos sólidos no suponen ningún obstáculo para la tecnología inalámbrica Bluetooth. Tampoco es necesario que los dispositivos estén situados en la misma línea de visión, es decir, orientados uno frente a otro, ya que se transmite en todas direcciones. La seguridad siempre ha sido una de las prioridades en el desarrollo de la tecnología Bluetooth y continúa siéndolo. La especificación Bluetooth ofrece tres modos de seguridad. El coste de los chips Bluetooth es inferior a tres dólares estadounidenses.

A.3.2 Banda ultra ancha (UWB)

La revolucionaria tecnología inalámbrica UWB permite transmitir datos digitales a través de un gran espectro de bandas de frecuencia y con un consumo mínimo. Alcanza velocidades de transmisión realmente altas en aplicaciones de redes de área local inalámbricas. Hasta la fecha, el uso de esta tecnología sólo está permitido y regulado en Estados Unidos. Debido a los desacuerdos sobre su estándar y a la falta de aprobación global, hay muy pocos productos con tecnología UWB en el mercado. Con esta tecnología se persigue un consumo de energía mínimo, gran velocidad de transmisión, bajo coste, uso de un amplio espectro de radiofrecuencias, transmisión de señales a través de obstáculos (como puertas, por ejemplo) y un gran abanico de aplicaciones: defensa, industria, hogar, etc. Actualmente, compiten dos estándares UWB en el mercado. El foro UWB respalda un estándar basado en una secuencia directa (DS-UWB). La alianza WiMedia defiende otro estándar basado en la modulación por división de frecuencia ortogonal (OFDM). Ambos estándares ofrecen una velocidad de transmisión de hasta 500 Mbps en un radio de dos metros y de hasta 110 Mbps en un radio de 10 metros. El SIG de Bluetooth anunció en mayo de 2005 su intención de colaborar con ambos grupos para desarrollar una velocidad de transmisión mayor en la especificación Bluetooth a través de la radiofrecuencia UWB.

A.3.3 Tecnología USB inalámbrica certificada

Caracterizada por su velocidad, la tecnología USB inalámbrica se ha diseñado para alcanzar hasta 480 Mbps en un radio de dos metros y hasta 110 Mbps a menos de diez metros de distancia. Un concentrador USB inalámbrico puede admitir hasta 127 dispositivos USB inalámbricos. Esta tecnología se servirá de la banda de frecuencia UWB patrocinada por la alianza WiMedia. Permite las conexiones punto a punto entre los

dispositivos y el concentrador USB inalámbrico. En febrero de 2004 Intel creó el grupo promotor del USB inalámbrico. El foro de desarrolladores USB (USB-IF) se encarga de realizar las pruebas de calidad y certificar los equipos con tecnología USB inalámbrica.

A.3.4 Wi-Fi (IEEE 802.11)

La implementación de la tecnología Bluetooth supone tan sólo un tercio del coste de la tecnología Wi-Fi. El consumo de la tecnología Bluetooth es cinco veces menor al de la tecnología Wi-Fi. La alianza Wi-Fi se encarga de realizar las pruebas de calidad y certificar los equipos inalámbricos basados en el estándar 802.11.

- 802.11a: Utiliza OFDM, funciona en una banda de frecuencia de 5 GHz y proporciona una velocidad máxima de transmisión de 54 Mbps.
- 802.11b: Funciona en la banda de frecuencia de 2.4 GHz, proporciona una velocidad máxima de transmisión de 11 Mbps y utiliza el espectro ensanchado de secuencia directa (DSSS). 802.11b es el estándar original de la tecnología Wi-Fi.
- 802.11g: Funciona en una banda de frecuencia de 2.4 GHz, utiliza OFDM y proporciona una velocidad máxima de transmisión de 54 Mbps. Dispone de compatibilidad inversa con el estándar 802.11b.
- 802.11e: Esta especificación mejorará la calidad de servicio.
- 802.11h: Este estándar complementará al 802.11a en Europa y ofrecerá una mayor gestión del espectro y del consumo de energía, ya que añade características como la selección de frecuencia dinámica (DFS) y el control de potencia de transmisión (TPC) a la especificación 802.11a.
- 802.11i: Esta norma mejorará la seguridad, ya que incluye un avanzado estándar de cifrado (AES). No ofrece compatibilidad inversa, por lo que algunos usuarios tendrán que actualizar su hardware. Este estándar también se conoce con el nombre de WPA2.
- 802.11k: Este estándar, que se encuentra actualmente en desarrollo, permitirá una mayor gestión de los recursos de radiofrecuencia en las redes 802.11.
- 802.11n: Se espera que este estándar funcione en la banda de frecuencia de 5 GHz y ofrezca una velocidad máxima de transmisión de 100 Mbps (aunque algunas propuestas barajan velocidades de hasta 500 Mbps). Se espera que el estándar 802.11n gestione las aplicaciones multimedia inalámbricas mejor que otras especificaciones 802.11.
- 802.11p: Este estándar opera en el espectro de frecuencias de 5.9 GHz, especialmente indicado para automóviles. Será la base de las comunicaciones dedicadas de corto alcance (DSRC) en Norteamérica. La tecnología DSRC permitirá el intercambio de datos entre vehículos y entre automóviles e infraestructuras en carretera.
- 802.11r: Esta modificación del estándar facilitará a los usuarios el desplazamiento entre los distintos puntos de acceso y las estaciones base. El grupo encargado del desarrollo de este estándar se formó en torno a la primavera-verano de 2004.

- 802.11s: Las modificaciones realizadas en el estándar permitirán un entramado de redes interconectadas en las redes 802.11. El grupo encargado del desarrollo de este estándar se formó en torno a la primavera-verano de 2004.

A.3.5 WiMAX (interoperabilidad mundial para acceso por microondas) y la norma IEEE 802.16

WiMax es una tecnología inalámbrica para redes de área metropolitana (MAN). Su alcance es de 50 Km. y ofrece una velocidad de transmisión de 70 Mbps. Las células habituales tienen un alcance menor. El primer estándar 802.16 operaba en bandas de 10 a 66 GHz y requería que los dispositivos estuvieran en la misma línea de visión. La nueva versión, el estándar 802.16a, funciona en frecuencias que oscilan entre 2 y 11 GHz y, ahora, también permite la conexión entre dispositivos que no se encuentren en la misma línea de visión. La aprobación de esta norma en Europa está sufriendo retrasos debido a cuestiones relacionadas con el uso de los espectros en las bandas de frecuencia de 2.8 y 3.4 GHz. Esta tecnología es compatible con usuarios móviles que viajen a velocidades de entre 20 y 100 Km/h (e incluso a velocidades superiores). El estándar 802.16e ofrecerá a los usuarios movilidad y portabilidad. La norma IEEE 802.16a y el estándar ETSI HiperMAN (Red de área metropolitana de alto rendimiento) comparten las mismas capas físicas (PHY) y MAC. La norma 802.16 se ha diseñado para que sea compatible con el estándar europeo. Esta tecnología, creada como alternativa a las líneas ADSL y la conexión por cable, es ideal para zonas rurales o áreas en las que el cableado resulta poco viable.

A.3.6 Infrarrojos (IrDA)

La tecnología de infrarrojos permite conectar de forma inalámbrica dispositivos que normalmente requieren una conexión por cable. Se trata de un estándar de transmisión de datos punto a punto en modo ad-a.C. con un ángulo de recepción estrecho (30°), diseñado para distancias de menos de un metro y que alcanza velocidades de entre 9.600 bps y 16 Mbps.

Los infrarrojos no pueden atravesar objetos sólidos y sus aplicaciones de intercambio de datos son limitadas en comparación con otras tecnologías.

Se utiliza principalmente en procedimientos de pago, sistemas de control remoto o para la sincronización de dos PDA.

A.3.7 Tecnología de comunicación a corta distancia (NFC)

El foro NFC se centra en el desarrollo y en la promoción de la tecnología NFC. Las doce empresas promotoras de este foro son las siguientes: MasterCard International, Microsoft, Motorola, NEC, Nokia, Panasonic, Philips, Renesas, Samsung Electronics, Sony, Texas Instruments y Visa. Ofrece una velocidad de transmisión de 212 Kbps a una distancia de hasta 20 cm. y opera en una banda de frecuencia de 13.56 MHz. El estándar NFC se basa en la tecnología RFID. Entre otras aplicaciones propuestas hallamos: compra y emisión de entradas, formas de pago y juegos. Al utilizar un modelo de comunicación pasiva permite un gran ahorro de energía.

A.3.8 Red de área local de radio de altas prestaciones (HiperLAN).

Las características de la tecnología HiperLAN de tipo 2 (HiperLAN 2) son las siguientes: velocidad de transmisión de 54 Mbps y alcance de 50 a 100 m. Actualmente no tiene ninguna aplicación destacable.

A.3.9 Red de área metropolitana de alto rendimiento (HiperMAN)

Se trata de un estándar de acceso inalámbrico fijo desarrollado por el Instituto Europeo de Normalización de las Telecomunicaciones (ETSI). Opera en el espectro de frecuencia de 2 a 11 GHz y es compatible con el estándar IEEE 802.16a de 2003.

A.3.10 802.20

Se trata de una tecnología de acceso inalámbrico de banda ancha móvil. Se espera una velocidad de transmisión máxima de 1 Mbps y opera en bandas inferiores a 3.5 GHz y que precisan de licencia. Es compatible con usuarios móviles que vayan a velocidades de hasta 250 km/h.

Anexo B. J2ME

B.1 Introducción

En la década de los 90, Sun Microsystems lanza el lenguaje de programación JAVA, lo que en un principio fue diseñado para generar aplicaciones que controlaran electrodomésticos extendió su funcionamiento para la creación de componentes interactivos integrados, como páginas Web y programación de aplicaciones independientes, esto, debido a la gran robustez e independencia de la plataforma donde se ejecuta el código.

Con los años, Java ha evolucionado enormemente en varios ámbitos como servicios HTTP, servidores de aplicaciones, acceso a bases de datos (JDBC). Debido a la explosión tecnológica de estos últimos años Java ha desarrollado soluciones personalizadas para cada ámbito tecnológico. Sun ha agrupado cada uno de esos ámbitos en una edición distinta de su lenguaje Java. Estas ediciones son: Java 2 Stantard Edition, orientada al desarrollo de aplicaciones independientes y de applets, Java 2 Enterprise Edition, enfocada al entorno empresarial y Java 2 Micro Edition, orientada a la programación de aplicaciones para pequeños dispositivos. Esta última edición de JAVA será la utilizada para el desarrollo del Software para el Cliente Móvil.

B.2 Análisis Comparativo

Sun, dispuesto a proporcionar las herramientas necesarias para cubrir las necesidades de todos los usuarios, creó distintas versiones de Java de acuerdo a las necesidades de cada uno. Según esto nos encontramos con que el paquete Java 2 lo podemos dividir en 3 ediciones distintas.

B.2.1 Java 2 Platform, Standard Edition (J2SE)

Esta edición de Java es la que en cierta forma recoge la iniciativa original del lenguaje Java. Tiene las siguientes características:

- Inspirado inicialmente en C++, pero con componentes de alto nivel, como soporte nativo de strings y recolector de basura.
- Código independiente de la plataforma, precompilado a bytecodes intermedio y ejecutado en el cliente por una JVM (Java Virtual Machine).
- Modelo de seguridad tipo sandbox proporcionado por la JVM.
- Abstracción del sistema operativo subyacente mediante un juego completo de APIs de programación.

Esta versión de Java contiene el conjunto básico de herramientas usadas para desarrollar Java Applets, así como las APIs orientadas a la programación de aplicaciones de usuario final: Interfaz gráfica de usuario, multimedia, redes de comunicación, etc.

B.2.2 Java 2 Platform, Enterprise Edition (J2EE)

Esta versión está orientada al entorno empresarial. El software empresarial tiene unas características propias marcadas: está pensado no para ser ejecutado en un equipo, sino para ejecutarse sobre una red de ordenadores de manera distribuida y remota mediante EJBs (Enterprise Java Beans). De hecho, el sistema se monta sobre varias unidades o aplicaciones.

En muchos casos, además, el software empresarial requiere que se sea capaz de integrar datos provenientes de entornos heterogéneos. Esta edición está orientada especialmente al desarrollo de servicios Web, servicios de nombres, persistencia de objetos, XML, autenticación, APIs para la gestión de transacciones, etc. El cometido de esta especificación es ampliar la J2SE para dar soporte a los requisitos de las aplicaciones de empresa.

B.3 Java 2 Platform, Micro Edition (J2ME)

Esta versión de Java está enfocada a la aplicación de la tecnología Java en dispositivos electrónicos con capacidades computacionales y gráficas muy reducidas, tales como teléfonos móviles, PDAs o electrodomésticos inteligentes. Esta edición tiene unos componentes básicos que la diferencian de las otras versiones, como el uso de una máquina virtual denominada KVM (Kilo Virtual Machine, debido a que requiere sólo unos pocos Kilobytes de memoria para funcionar) en vez del uso de la JVM clásica, inclusión de un pequeño y rápido recolector de basura. La Ilustración B-1 muestra la ubicación de J2ME dentro de la Plataforma Java 2.

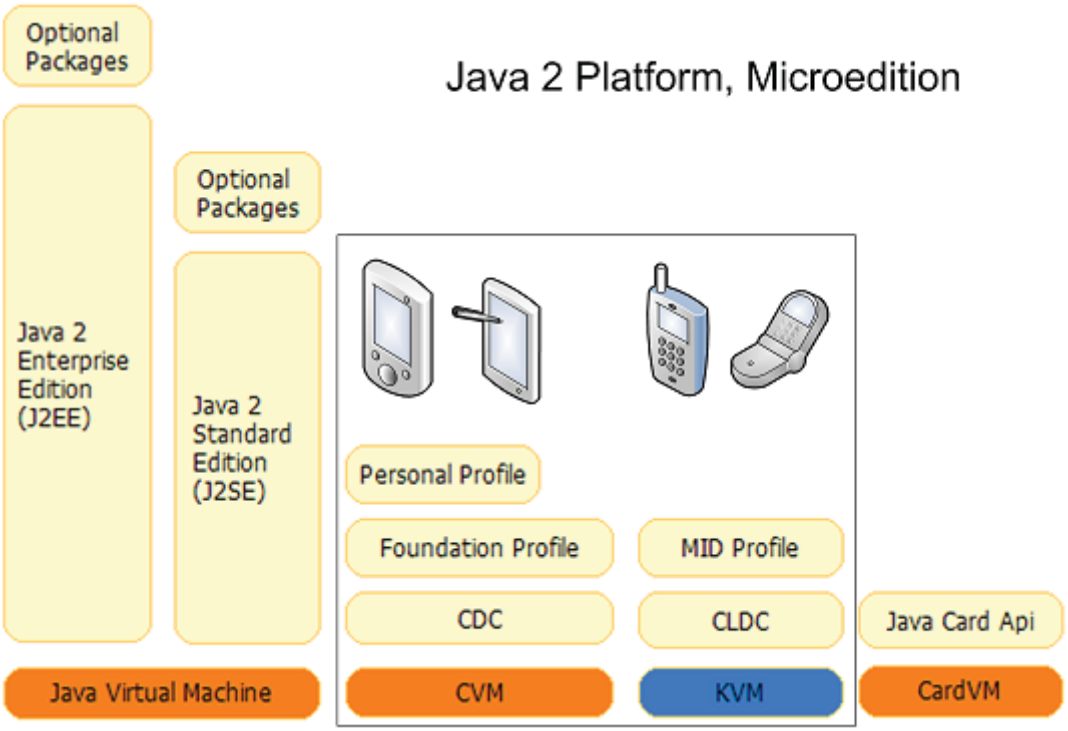

Ilustración B-1 - La Plataforma Java 2

B.3.1 Introducción

La edición Java 2 Micro Edición es relativamente nueva, fue presentada en 1999 con el propósito de desarrollar aplicaciones Java para pequeños dispositivos, ya para ese año existía un auge en el desarrollo de dispositivos móviles. En esta presentación, lo que realmente se diseñó fue una versión de Java Virtual Machine (JVM) que podía ejecutarse en dispositivos Palm.

Java Micro Edición es la versión del lenguaje Java que está orientada al desarrollo de aplicaciones para dispositivos pequeños con capacidades restringidas tanto en pantalla gráfica, como de procesamiento y memoria bajo esta categoría podemos mencionar; teléfonos móviles, PDA`s, Handhelds, Pagers, etc.

Actualmente las compañías telefónicas y los fabricantes de móviles están implantando los protocolos y dispositivos necesarios para soportar la tecnología que envuelve a Java 2 Micro Edición, es así como en este mercado la mayoría de los nuevos teléfonos móviles ya les viene integrada la maquina virtual necesaria para implementar aplicaciones que se desarrollen bajo esta edición abriendo un mundo de posibilidades para los desarrolladores independientes.

J2ME contiene una mínima parte de las APIs de Java. Esto es debido a que la edición estándar de APIs de Java ocupa 20 Mb, y los dispositivos pequeños disponen de una cantidad de memoria mucho más reducida. En concreto, J2ME usa 37 clases de la plataforma J2SE provenientes de los paquetes java.lang, java.io, java.util.

Otras diferencias con la plataforma J2SE vienen dadas por el uso de una máquina virtual distinta de la clásica JVM denominada KVM. Esta KVM tiene unas restricciones que hacen que no posea todas las capacidades incluidas en la JVM.

B.3.2 KVM

Es la Máquina Virtual más pequeña entre todas las ediciones. Su nombre KVM proviene de Kilobyte (haciendo referencia a la baja ocupación de memoria, entre 40Kb y 80Kb). Se trata de una implementación de Máquina Virtual reducida y especialmente orientada a dispositivos con bajas capacidades de procesamiento y de memoria. La KVM está escrita en lenguaje C, aproximadamente unas 24000 líneas de código, y fue diseñada para ser:

- Pequeña, con una carga de memoria entre los 40Kb y los 80 Kb, dependiendo de la plataforma y las opciones de compilación.
- Alta portabilidad.
- Modulable.
- Lo más completa y rápida posible y sin sacrificar características para las que fue diseñada.

B.3.3 Configuraciones

Una configuración es el conjunto mínimo de APIs Java que permiten desarrollar aplicaciones para un grupo de dispositivos. Éstas APIs describen las características básicas, comunes a todos los dispositivos:

- Características soportadas del lenguaje de programación Java.
- Características soportadas por la Máquina Virtual Java.
- Bibliotecas básicas de Java y APIs soportadas.

  Existen dos configuraciones en J2ME:

  - CLDC: Orientada a dispositivos con limitaciones computacionales y de memoria.
  - CDC: Orientada a dispositivos con no tantas limitaciones.

B.3.3.1 Configuración de dispositivos con conexión, CDC (Connected Limited Configuration)

La Configuración CDC está orientada a dispositivos con cierta capacidad computacional y de memoria. Por ejemplo, decodificadores de televisión digital, televisores con Internet, algunos electrodomésticos y sistemas de navegación en automóviles. CDC usa una Máquina Virtual Java similar en sus características a una de J2SE, pero con limitaciones en el apartado gráfico y de memoria del dispositivo. Ésta Máquina Virtual es conocida como CVM (Compact Virtual Machine). La Configuración CDC está enfocada a dispositivos con las siguientes capacidades:

- Procesador de 32 bits.
- Disponer de 2 Mb o más de memoria total, incluyendo memoria RAM y ROM.
- Poseer la funcionalidad completa de la Máquina Virtual Java 2.
- Conectividad a algún tipo de red.

Las peculiaridades de CDC están contenidas principalmente en el paquete javax.microedition.io, que incluye soporte para comunicaciones http y basadas en datagramas.

B.3.3.2 Configuración de dispositivos limitados con conexión, CLDC (Connected Limited Device Configuration)

La Configuración CLDC está orientada a dispositivos dotados de conexión y con limitaciones en cuanto a capacidad gráfica, cómputo y memoria. Un ejemplo de estos dispositivos son: teléfonos móviles, buscapersonas (pagers), PDAs, organizadores personales, etc.

Algunas de estas restricciones vienen dadas por el uso de la KVM, necesaria al trabajar con la CLDC debido a su pequeño tamaño. Los dispositivos que usan CLDC deben cumplir los siguientes requisitos:

- Disponer entre 160 Kb y 512 Kb de memoria total disponible. Como mínimo se debe disponer de 128 Kb de memoria no volátil para la Máquina Virtual Java y las bibliotecas CLDC, y 32 Kb de memoria volátil para la Máquina Virtual en tiempo de ejecución.
- Procesador de 16 o 32 bits con al menos 25 Mhz de velocidad.

- Ofrecer bajo consumo, debido a que estos dispositivos trabajan con suministro de energía limitado, normalmente baterías.
- Tener conexión a algún tipo de red, normalmente sin cable, con conexión intermitente y ancho de banda limitado (unos 9600 bps). La CLDC aporta las siguientes funcionalidades a los dispositivos:
  - Un subconjunto del lenguaje Java y todas las restricciones de su Máquina Virtual (KVM).
  - Un subconjunto de las bibliotecas Java del núcleo.
  - Soporte para E/S básica.
  - Soporte para acceso a redes.

En cualquier caso, una determinada Configuración no se encarga del mantenimiento del ciclo de vida de la aplicación, interfaces de usuario o manejo de eventos, sino que estas responsabilidades caen en manos de los perfiles.

B.3.4 Perfiles

Los perfiles identifican un grupo de dispositivos por la funcionalidad que proporcionan y el tipo de aplicaciones que se ejecutarán en ellos. Las librerías de la interfaz gráfica son un componente muy importante en la definición de un perfil. Aquí podemos encontrar grandes diferencias entre interfaces, desde el menú textual de los teléfonos móviles hasta los táctiles de los PDAs.

El perfil establece unas APIs que definen las características de un dispositivo, mientras que la configuración hace lo propio con una familia de ellos. Esto hace que a la hora de construir una aplicación se cuente tanto con las APIs del perfil como de la configuración. Tenemos que tener en cuenta que un perfil siempre se construye sobre una configuración determinada. De este modo, podemos pensar en un perfil como un conjunto de APIs que dotan a una configuración de funcionalidad específica.

Anteriormente vimos que para una configuración determinada se usaba una Máquina Virtual Java específica. Teníamos que con la configuración CDC usábamos la CVM y que con la configuración CLDC usábamos la KVM. Con los perfiles ocurre lo mismo. Existen unos perfiles que construiremos sobre la configuración CDC y otros que Construiremos sobre la CLDC.

Para la configuración CDC tenemos los siguientes perfiles:

- Foundation Profile.
- Personal Profile.
- RMI Profile.

Y para la configuración CLDC tenemos los siguientes:

- PDA Profile.
- Mobile Information Device Profile (MIDP).

La Ilustración B-2 muestra la estructura de las Configuraciones y Perfiles de J2ME en relación con los demás componentes de Hardware y Software.

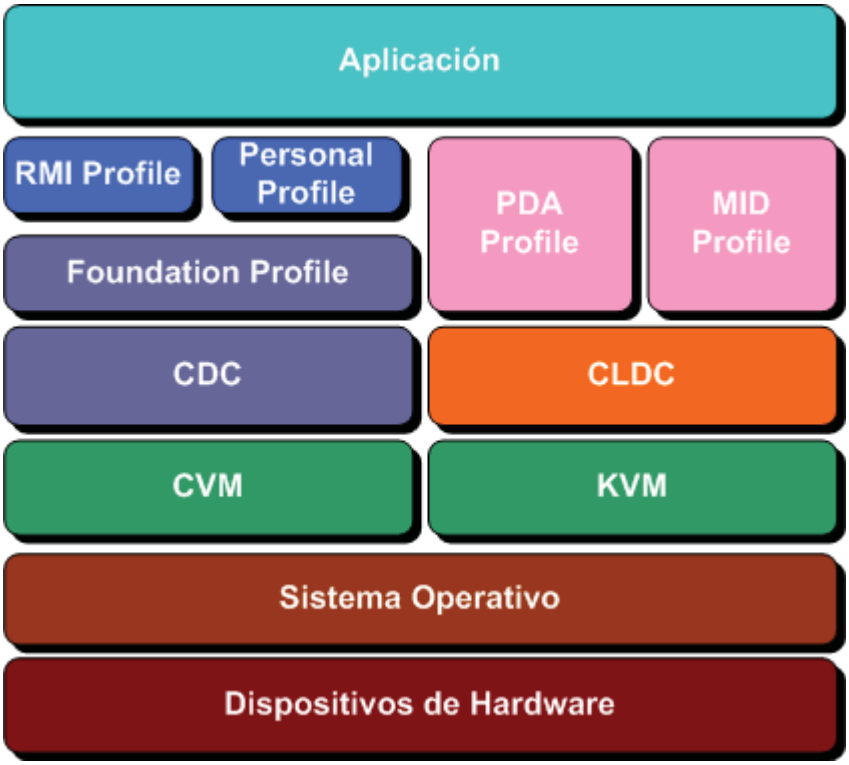

Ilustración B-2 - Configuraciones y Perfiles de J2ME

B.3.4.1 Mobile Information Device Profile (MIDP)

Este perfil está construido sobre la configuración CLDC. Al igual que CLDC fue la primera configuración definida para J2ME, MIDP fue el primer perfil definido para esta plataforma.

Este perfil está orientado para dispositivos con las siguientes características:

- Reducida capacidad computacional y de memoria.
- Conectividad limitada (en torno a 9600 bps).
- Capacidad gráfica muy reducida (mínimo un display de 96x54 pixels monocromo).
- Entrada de datos alfanumérica reducida.
- 128 Kb de memoria no volátil para componentes MIDP. 8 Kb de memoria no volátil para datos persistentes de aplicaciones. 32 Kb de memoria volátil en tiempo de ejecución para la pila Java.

Los tipos de dispositivos que se adaptan a estas características son: teléfonos, móviles, PDAs de gama baja con conectividad. El perfil MIDP establece las capacidades del dispositivo, por lo tanto, especifica las APIs relacionadas con:

- La aplicación (semántica y control de la aplicación MIDP).
- Interfaz de usuario. Almacenamiento persistente.

- Trabajo en red. Temporizadores.

Las aplicaciones realizadas utilizando MIDP reciben el nombre de MIDlets. Por lo tanto un MIDlet es una aplicación Java realizada con el perfil MIDP sobre la configuración CLDC.

B.4 J2ME y comunicación

El soporte que da MIDP2.0 a la programación en red se basa, como en MIDP1.0, en el Generic Connection Framework de CLDC. CLDC Generic Connection Framework define una serie de interfaces para dar soporte a la variedad de tipos de conexiones que nos podemos encontrar en dispositivos móviles, pero no implementa ninguna de ellas. Es en los perfiles donde se debe realizar esta implementación. En MIDP1.0, se daba soporte únicamente a conexiones HTTP, a través de la implementación dLa interfaz HttpConnection.Sin embargo, MIDP2.0 ofrece nuevas interfaces de conexión de red, lo que supone un conjunto de abstracciones más completo que es de gran utilidad para el programador.

Este cambio sustancial de MIDP2.0 respecto a MIDP1.0 surge como respuesta a la constante evolución que experimentan tanto los terminales móviles (tecnologías GPRS y UMTS) como las aplicaciones que se diseñan para dichos terminales. Se imponen nuevos requisitos a los mecanismos de conexión que se traducen en la necesidad de un nuevo conjunto de abstracciones que pueda ser utilizado a nivel de programador.

Los dispositivos que trabajan en conmutación de circuitos necesitan conexiones basadas en flujo de bits, como por ejemplo el protocolo TCP.

Los dispositivos que trabajan en conmutación de paquetes necesitan conexiones basadas en data gramas así que requerirán conexiones como las que ofrece el protocolo UDP.

Habrá dispositivos con mecanismos específicos de conexión.

Como consecuencia MIDP2.0 responde al reto planteado ofreciendo multitud de formas de conexión manejables a nivel del programador y que no introducen un mayor grado de complejidad ya que todas ellas están englobadas dentro del mismo conjunto de abstracciones mencionado anteriormente.

B.4.1 CLDC Generic Connection Framework

En la configuración CLDC Generic Connection Framework, todas las conexiones se crean utilizando el método estático open() de la clase Connector. Si no se produce ningún error, este método devuelve un objeto que implementa una de las interfaces definidas en el CLDC Generic Connection Framework e implementadas por MIDP2.0, la Ilustración B-3 muestra la jerarquía y los distintos tipos de conexión de MIDP2.0:

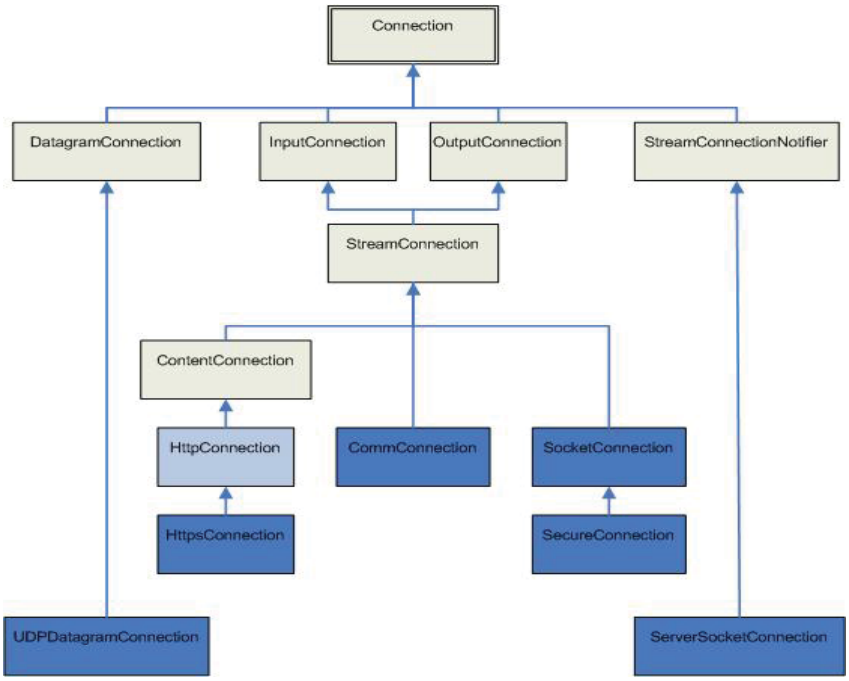

Ilustración B-3 - Conexiones MIDP

La interfaz Connection es el nodo raíz en este árbol de jerarquía por lo que el resto de interfaces serán subinterfaces de Connection. En la Ilustración B-3, los interfaces en un recuadro verde son parte de CLDC 1.0. La interfaz HttpConnection fue añadido por MIDP 1.0. Los interfaces dentro de los recuadros azules son interfaces añadidos por MIDP 2.0.

Todas estas interfaces están contenidas en el paquete javax.microedition.io:

- La interfaz Connection es el tipo básico de conexión. Esta conexión sólo puede abrirse y cerrarse.
- La interfaz InputConnection representa un dispositivo desde el que se pueden leer datos. Proporciona el método openInputStream que devuelve un stream de entrada para la conexión.
- La interfaz OuputConnection representa un dispositivo en el que se pueden escribir datos. Proporciona el método openOutputStream que devuelve un stream de salida para la conexión.
- La interfaz StreamConnection que combina las conexiones de entrada y de salida anteriores.

- La interfaz ContentConnection que es una subinterfaz de StreamConnection. Proporciona acceso a la información de algunos metadatos proporcionados en una conexión HTTP.
- La interfaz StreamConnectionNotified que espera a que se establezca una conexión. Devuelve un StreamConnection a través del cual puede establecerse un enlace de comunicación bidireccional.
- La interfaz DatagramConnection que representa el destino de un data grama.
- La interfaz HttpConnection.
- La interfaz HttpsConnection (http seguro).
- La interfaz CommConnection, conexiones por puerto serie.
- La interfaz SocketConnection.
- La interfaz SecureConnection (sockets seguros).
- La interfaz ServerSocketConnection.
- La interfaz UDPDatagramConnection.

Como vemos hay un número considerable de interfaces y cada una de ellas nos permite establecer un tipo de conexión específico y con un protocolo determinado. Sin embargo J2ME nos permite trabajar con estas interfaces de la manera más sencilla posible.

B.4.2 Tipos de pantalla (Interfaces de Usuario en MIDP)

Para finalizar la clase de hoy veremos, en el apartado siguiente, algunas clases básicas de interfaz de usuario (UI) de MIDP, incluidas todas ellas en el paquete javax.microedition.lcdui.

B.4.2.1 Displayable y Display

En J2ME, un objeto Displayable contiene la información que va a ser visualizada, y un objeto Display gestiona qué objeto Displayable se mostrará al usuario. En MIDP existen tres categorías de Displayable:

Screen con estructura predefinida: Alert, List y TextBox, que encapsulan componentes de interfaz complejos y que las aplicaciones no pueden enriquecer con nuevos componentes.

Screen genérica: Form, las aplicaciones pueden llenar este tipo de pantalla con texto, imágenes u otros componentes de interfaz de usuario.

Canvas, las aplicaciones tienen control total sobre la aparición de componentes en el display y puede acceder directamente a eventos de bajo nivel.

La clase Display proporciona los métodos que nos permiten controlar la visualización de los objetos Displayable y obtener propiedades del display (si soporta color o no, número de colores. Sólo existe una instancia del objeto Display por MIDlet, y la aplicación obtiene una referencia a ese objeto realizando una llamada al método getDisplay() que suele invocarse en el constructor del MIDlet.

B.4.2.2 Eventos y su gestión

En MIDP la gestión de eventos sigue el mismo modelo que se usa en AWT en J2SE, este modelo comprende dos componentes: las fuentes de eventos, que los generan y los "listeners" de eventos, que son los que los procesan. En MIDP los "listeners" de eventos van a estar asociadas a objetos Displayables, que son las fuentes de eventos. El UI de MIDP a alto nivel utiliza dos tipos de eventos: Command y ItemStateChanged, cada uno de estos eventos tiene el correspondiente listener asociado: CommandListener y ItemStateListener, respectivamente. Cualquier objeto Displayable puede ser fuente de eventos Command mientras que sólo Form puede ser fuente de ItemStateListener.

A bajo nivel el UI de MIDP permite gestionar un mayor número de eventos, incluyendo pulsación de cualquier tecla, pantalla táctil,...Para poder capturar eventos a bajo nivel se deben de utilizar Displayables de bajo nivel, es decir la clase Canvas o subclases de la misma. Tanto los eventos a alto nivel, como a bajo nivel tienen una cosa en común: la gestión de eventos debe de realizarse siempre en el mismo thread en el que se produce el evento.Un objeto Command tiene tres valores importantes:

- Label: un String que representa el significado del Command, y es el que la aplicación muestra a los usuarios
- Type: un entero que especifica lo que va a realizar el Command. Los tipos definidos son BACK, CANCEL, HELP, EXIT, ITEM, OK, SCREEN, y STOP
- Priority: un entero que indica la importancia del comando. El que tenga un número menos será el más importante

Las implementaciones de MIDP mapean estos comandos con lo que se denominan "soft-button", típicamente los botones que no son numéricos en el teléfono y que sirven para navegar por los menús. CommandListener es una interfaz que proporciona el procesamiento de un evento Command, a través del método commandAction(Command c, Displayable d) que deberemos implementar con las acciones oportunas. La Ilustración B-4 muestra las relaciones y jerarquías de los distintos tipos de Displayables existentes en MIDP.

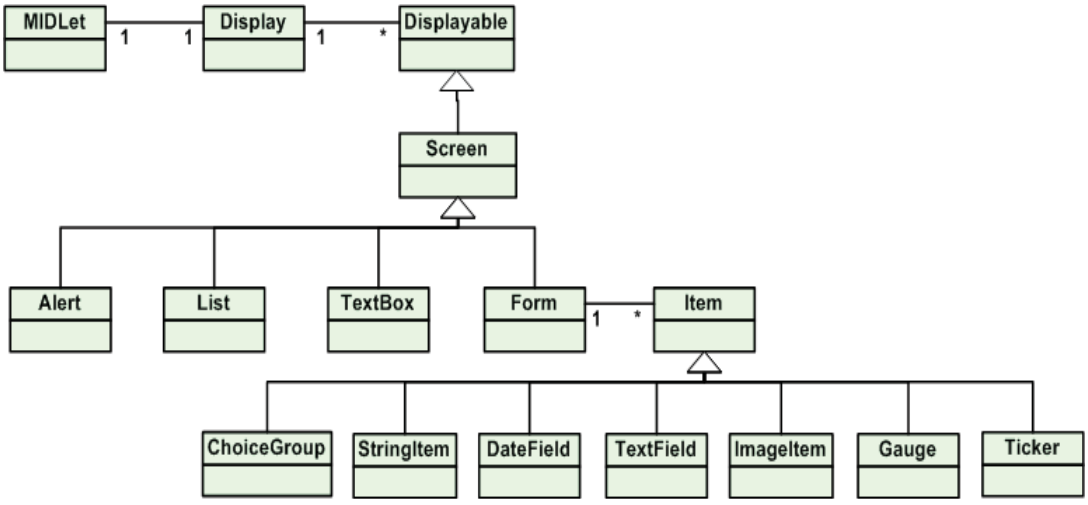

Ilustración B-4 - Tipos de Displayables en MIDP

Anexo C. VPN

C.1 Introducción

La Red Privada Virtual (RPV), en inglés Virtual Private Network (VPN), es una tecnología de red que permite una extensión de la red local sobre una red pública o no controlada, como por ejemplo Internet.

Ejemplos comunes son, la posibilidad de conectar dos o más sucursales de una empresa utilizando como vínculo Internet, permitir a los miembros del equipo de soporte técnico la conexión desde su casa al centro de cómputo, o que un usuario pueda acceder a su equipo doméstico desde un sitio remoto, como por ejemplo un hotel. Todo ello utilizando la infraestructura de Internet.

Para hacerlo posible de manera segura es necesario proporcionar los medios para garantizar la autenticación, integridad y confidencialidad de toda la comunicación:

- Autenticación y autorización: ¿Quién está del otro lado? Usuario/equipo y qué nivel de acceso debe tener.
- Integridad: La garantía de que los datos enviados no han sido alterados. Para ello se utiliza funciones de Hash. Los algoritmos de hash más comunes son los Message Digest (MD2 y MD5) y el Secure Hash Algorithm (SHA).
- Confidencialidad: Dado que los datos viajan a través de un medio potencialmente hostil como Internet, los mismos son susceptibles de intercepción, por lo que es fundamental el cifrado de los mismos. De este modo, la información no debe poder ser interpretada por nadie más que los destinatarios de la misma. Se hace uso de algoritmos de cifrado como Data Encryption Standard (DES), Triple DES (3DES) y Advanced Encryption Standard (AES).
- No repudio: es decir, un mensaje tiene que ir firmado, y el que lo firma no puede negar que el mensaje lo envió él.

C.2 Requerimientos básicos

- Identificación de usuario: Las VPN deben verificar la identidad de los usuarios y restringir su acceso a aquellos que no se encuentren autorizados.
- Codificación de datos: Los datos que se van a transmitir a través de la red pública (Internet), antes deben ser cifrados, para que así no puedan ser leídos. Esta tarea se realiza con algoritmos de cifrado como DES o 3DES que solo pueden ser leídos por el emisor y receptor.
- Administración de claves: Las VPN deben actualizar las claves de cifrado para los usuarios.

C.3 Tipos de VPN

Básicamente existen tres arquitecturas de conexión VPN:

C.3.1 VPN de acceso remoto

Es quizás el modelo más usado actualmente y consiste en usuarios o proveedores que se conectan con la empresa desde sitios remotos (oficinas comerciales, domicilios, hotel, aviones preparadas), etcétera) utilizando Internet como vínculo de acceso. Una vez autenticados tienen un nivel de acceso muy similar al que tienen en la red local de la empresa. Muchas empresas han reemplazado con esta tecnología su infraestructura «dial-up» (módems y líneas telefónicas), aunque por razones de contingencia todavía conservan sus viejos módems.

C.3.2 VPN punto a punto

Este esquema se utiliza para conectar oficinas remotas con la sede central de la organización. El servidor VPN, que posee un vínculo permanente a Internet, acepta las conexiones vía Internet provenientes de los sitios y establece el túnel VPN. Los servidores de las sucursales se conectan a Internet utilizando los servicios de su proveedor local de Internet, típicamente mediante conexiones de banda ancha. Esto permite eliminar los costosos vínculos punto a punto tradicionales, sobre todo en las comunicaciones internacionales. Es más común el punto anterior, también llamada tecnología de túnel o tunneling:

C.3.3 Tunneling

Internet se construyó desde un principio como un medio inseguro. Muchos de los protocolos utilizados hoy en día para transferir datos de una máquina a otra a través de la red carecen de algún tipo de cifrado o medio de seguridad que evite que nuestras comunicaciones puedan ser interceptadas y espiadas. HTTP, FTP, POP3 y otros muchos protocolos ampliamente usados, utilizan comunicaciones que viajan en claro a través de la red. Esto supone un grave problema, en todas aquellas situaciones en las que queremos transferir entre máquinas información sensible, como pueda ser una cuenta de usuario (nombre de usuario y contraseña), y no tengamos un control absoluto sobre la red, a fin de evitar que alguien pueda interceptar nuestra comunicación por medio de la técnica del hombre en el medio (man in the middle), como es el caso de la Red de redes.

El problema de los protocolos que envían sus datos en claro, es decir, sin cifrarlos, es que cualquier persona que tenga acceso físico a la red en la que se sitúan las máquinas puede ver dichos datos. De este modo, alguien que conecte su máquina a una red y utilice un sniffer recibirá y podrá analizar por tanto todos los paquetes que circulen por dicha red. Si alguno de esos paquetes pertenece a un protocolo que envía sus comunicaciones en claro, y contiene información sensible, dicha información se verá comprometida. Si por el contrario, se cifran las comunicaciones con un sistema que permita entenderse sólo a las dos máquinas que son partícipes de la comunicación, cualquiera que intercepte desde una tercera máquina los paquetes, no podrá hacer nada con ellos, al no poder descifrar los datos.

Una forma de evitar este problema, sin dejar por ello de utilizar todos aquellos protocolos que carezcan de medios de cifrado, es usar una técnica llamada tunneling. Básicamente, esta técnica consiste en abrir conexiones entre dos máquinas por medio de un

protocolo seguro, como puede ser SSH (Secure SHell), a través de las cuales realizaremos las transferencias inseguras, que pasarán de este modo a ser seguras. De esta analogía viene el nombre de la técnica, siendo la conexión segura (en este caso de ssh) el túnel por el cual se envían los datos para que nadie más aparte de los interlocutores que se sitúan a cada extremo del túnel, pueda ver dichos datos. Este tipo de técnica requiere de forma imprescindible tener una cuenta de acceso seguro en la máquina con la que se quiere comunicar.

C.3.4 VPN interna WLAN

Este esquema es el menos difundido pero uno de los más poderosos para utilizar dentro de la empresa. Es una variante del tipo "acceso remoto" pero, en vez de utilizar Internet como medio de conexión, emplea la misma red de área local (LAN) de la empresa. Sirve para aislar zonas y servicios de la red interna. Esta capacidad lo hace muy conveniente para mejorar las prestaciones de seguridad de las redes inalámbricas (WiFi).

Un ejemplo clásico es un servidor con información sensible, como las nóminas de sueldos, ubicado detrás de un equipo VPN, el cual provee autenticación adicional más el agregado del cifrado, haciendo posible que sólo el personal de recursos humanos habilitado pueda acceder a la información.

C.4 ¿Por qué VPN?

Las VPN son una salida al costo que puede significar el pagar una conexión de alto coste, para usar líneas alquiladas que estén conectadas a otros puntos que puedan hacer uso de la conexión a Internet o para hacer negocios con clientes frecuentes a través de la red.

Los datos son codificados o cifrados y recién enviados a través de la conexión, para de esa manera asegurar la información y el password que se esté enviando.

Esta tecnología proporciona un medio para aprovechar un canal público de Internet como un canal privado o propio para comunicar datos que son privados. Más aún, con un método de codificación y encapsulamiento, una VPN básica, crea un camino privado a través de Internet. Esto reduce el trabajo y riesgo en una gestión de red

La tecnología de túneles esta basado en estándares. Esta tecnología permite transmitir datos entre dos redes similares. A esto también se llama "encapsulamiento", es decir, a la tecnología que coloca algún tipo de paquetes dentro de otro protocolo (TCP). Aparte de todo esto, también se añade otra información necesaria para poder descifrar la información que se encuentra codificada. Estos paquetes llegan a su destino después de haber atravesado Internet, pero para verificar que ha llegado al destino correcto se realiza un proceso de autentificación.

Las VPNs son una gran solución a distintos problemas, pero solo en el campo de la economía de los usuarios porque por ejemplo en el caso de que se realice una conexión entre dos sedes de empresas, una en Japón y la otra en Chile, sería muy costoso el realizar un cableado entre estos dos países, y un enlace inalámbrico satelital sería muy costoso. Es por ello que una red privada virtual es más económica porque solo se hace uso de Internet que es un conjunto de redes conectadas entre si.

Pero observándolo desde el punto de vista de los usuarios particulares de Internet, como son los estudiantes o investigadores que utilizan el Internet como un medio de

comunicación, de compartimiento de información, este tipo de redes perjudican este desarrollo, debido a varios factores como son, el consumo de ancho de banda y el consumo de direcciones IP.

C.4.1.1 Coste

La principal motivación del uso y difusión de esta tecnología es la reducción de los costos de comunicaciones directos, tanto en líneas dial-up como en vínculos WAN dedicados. Los costos se reducen drásticamente en estos casos:

En el caso de accesos remotos, llamadas locales a los ISP (Internet Service Provider) en vez de llamadas de larga distancia a los servidores de acceso remoto de la organización. O también mediante servicios de banda ancha.

En el caso de conexiones punto a punto, utilizando servicios de banda ancha para acceder a Internet, y desde Internet llegar al servidor VPN de la organización. Todo esto a un costo sensiblemente inferior al de los vínculos WAN dedicados.

C.4.1.2 Ancho de banda

Podemos encontrar otra motivación en el deseo de mejorar el ancho de banda utilizado en conexiones dial-up. Las conexiones VPN de banda ancha mejoran notablemente la capacidad del vínculo, pero los costos son más altos.

C.4.2 Implementaciones

El protocolo estándar de hecho es el IPSEC, pero también tenemos PPTP, L2F, L2TP, SSL/TLS, SSH, etc. Cada uno con sus ventajas y desventajas en cuanto a seguridad, facilidad, mantenimiento y tipos de clientes soportados.

Actualmente hay una línea de productos en crecimiento relacionada con el protocolo SSL/TLS, que intenta hacer más amigable la configuración y operación de estas soluciones.

Las soluciones de hardware casi siempre ofrecen mayor rendimiento y facilidad de configuración, aunque no tienen la flexibilidad de las versiones por software. Dentro de esta familia tenemos a los productos de WatchGuard, Nortel, Cisco, Linksys, Netscreen, Symantec, Nokia, US Robotics, D-Link, etc.

Las aplicaciones VPN por software son las más configurables y son ideales cuando surgen problemas de interoperatividad en los modelos anteriores. Obviamente el rendimiento es menor y la configuración más delicada, porque se suma el sistema operativo y la seguridad del equipo en general. Aquí tenemos por ejemplo a las soluciones nativas de Windows, Linux y los Unix en general. Por ejemplo productos de código abierto como OpenSSH, OpenVPN y FreeS/Wan.

En ambos casos se pueden utilizar soluciones de Firewall o Cortafuegos, obteniendo un nivel de seguridad alto por la protección que brinda el Cortafuegos, pero se pierde en rendimiento.

Ventajas:

- Integridad, confidencialidad y seguridad de datos.
- Las VPN reducen costos y son sencillas de usar.

C.4.3 Tipos de Conexión

C.4.3.1 Conexión de acceso remoto

Una conexión de acceso remoto es realizada por un cliente o un usuario de un computador que se conecta a una red privada, los paquetes enviados a través de la conexión VPN son originados al cliente de acceso remoto, y este se autentica al servidor de acceso remoto, y el servidor se autentica ante el cliente.

C.4.3.2 Conexión VPN router a router

Una conexión VPN router a router es realizada por un router, y este a su vez se conecta a una red privada. En este tipo de conexión, los paquetes enviados desde cualquier router no se originan en los routers. El router que realiza la llamada se autentica ante el router que responde y este a su vez se autentica ante el router que realiza la llamada y también sirve para la intranet

C.4.3.3 Conexión VPN Firewall ASA a Firewall ASA

Una conexión VPN Firewall ASA a Firewall ASA es realizada por un Firewall ASA, y este a su vez se conecta a una red privada. En este tipo de conexión, los paquetes enviados desde cualquier Usuario en Internet. El Firewall ASA que realiza la llamada se autentica ante el Firewall ASA que responde y este a su vez se autentica ante el Firewall ASA que realiza la llamada.

C.4.4 Certificados

Un certificado SSL es una credencial digital que permite a los visitantes de un sitio Web verificar la autenticidad del sitio y comunicarse con él de manera segura mediante el protocolo SSL. Este lo concede una entidad certificadora: la Certification Authority (CA). Esta entidad concede dicho certificado después de haber controlado la correcta configuración del proceso de encriptación (SSL) y haber comprobado los datos de la empresa solicitante. El certificado de servidor seguro se concede a una entidad cuyas referencias han sido comprobadas, para asegurar que efectivamente quien recibe los datos encriptados es quien debe de recibirlos.

Cuando nos conectamos a un servidor seguro, los navegadores avisan de esta circunstancia mediante un candado de color amarillo en alguna parte de la ventana, en el caso de FireFox la barra de direcciones aparecerá también en color amarillo. Para más seguridad podemos comprobar que la dirección empieza por https://www..., con una ‘s’ después de la ‘p’. Si pinchamos sobre el candado aparecerá la información contenida en el certificado digital que lo autoriza como servidor seguro.

Anexo D. XML

D.1 Introducción

XML, sigla en inglés de Extensible Markup Language, es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML). Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG, MathML.

XML no ha nacido sólo para su aplicación en Internet, sino que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo y casi cualquier cosa imaginable.

XML es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

D.2 Historia

XML proviene de un lenguaje inventado por IBM en los años setenta, llamado GML (Generalized Markup Language), que surgió por la necesidad que tenía la empresa de almacenar grandes cantidades de información. Este lenguaje gustó a la ISO, por lo que en 1986 trabajaron para normalizarlo, creando SGML (Standard Generalized Markup Language), capaz de adaptarse a un gran abanico de problemas. A partir de él se han creado otros sistemas para almacenar información.

En el año 1989 Tim Berners Lee creó la Web, y junto con ella el lenguaje HTML. Este lenguaje se definió en el marco de SGML y fue de lejos la aplicación más conocida de este estándar. Los navegadores Web sin embargo siempre han puesto pocas exigencias al código HTML que interpretan y así las páginas Web son caóticas y no cumplen con la sintaxis. Estas páginas Web dependen fuertemente de una forma específica de lidiar con los errores y las ambigüedades, lo que hace a las páginas más frágiles y a los navegadores más complejos.

Otra limitación de SGML es que cada documento pertenece a un vocabulario fijo, establecido por el DTD. No se pueden combinar elementos de diferentes vocabularios. Asimismo es imposible para un intérprete (por ejemplo un navegador) analizar el documento sin tener conocimiento de su gramática (del DTD). Por ejemplo, el navegador sabe que antes de una etiqueta debe haberse cerrado cualquier <p> previamente abierto. Los navegadores resolvieron esto incluyendo lógica ad hoc para el HTML, en vez de incluir un analizador genérico. Ambas opciones, de todos modos, son muy complejas para los navegadores. Se buscó entonces definir un subconjunto del SGML que permita:

- Mezclar elementos de diferentes lenguajes. Es decir que los lenguajes sean extensibles.
- La creación de analizadores simples, sin ninguna lógica especial para cada lenguaje.
- Empezar de cero y hacer hincapié en que no se acepte nunca un documento con errores de sintaxis.

Para hacer esto XML deja de lado muchas características de SGML que estaban pensadas para facilitar la escritura manual de documentos. XML en cambio está orientado a hacer las cosas más sencillas para los programas automáticos que necesiten interpretar el documento.

D.3 Ventajas de XML

Es extensible, lo que quiere decir que una vez diseñado un lenguaje y puesto en fabricación, igual es posible extenderlo con la adición de nuevas etiquetas de manera que los antiguos consumidores de la vieja versión todavía puedan entender el nuevo formato.

El analizador es un componente estándar, no es necesario crear un analizador específico para cada lenguaje. Esto posibilita el empleo de uno de los tantos disponibles. De esta manera se evitan bugs y se acelera el desarrollo de la aplicación.

Si un tercero decide usar un documento creado en XML, es sencillo entender su estructura y procesarlo. Mejora la compatibilidad entre aplicaciones.

D.4 Estructura de un documento XML

La tecnología XML busca dar solución al problema de expresar información estructurada de la manera más abstracta y reutilizable posible. Que la información sea estructurada quiere decir que se compone de partes bien definidas, y que esas partes se componen a su vez de otras partes. Entonces se tiene un árbol de pedazos de información. Ejemplos son un tema musical, que se compone de compases, que están formados a su vez por notas. Estas partes se llaman elementos, y se las señala mediante etiquetas. Una etiqueta consiste en una marca hecha en el documento, que señala una porción de éste como un elemento. Un pedazo de información con un sentido claro y definido. Las etiquetas tienen la forma <nombre>, donde nombre es el nombre del elemento que se está señalando. A continuación se muestra un ejemplo para entender la estructura de un documento XML:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE Edit_Mensaje SYSTEM "Lista.dtd" [<!ELEMENT Edit_Mensaje (Mensaje)*>]>
<Edit_Mensaje>
   <Mensaje>
      <Remitente>
         <Nombre>Nombre del remitente</Nombre>
         <Mail> Correo del remitente </Mail>
      </Remitente>
      <Texto>
         <Asunto>  Este es mi documento con una estructura muy sencilla          </Asunto>
         <Parrafo>  Este es mi documento con una estructura muy          </Parrafo>
      </Texto>
   </Mensaje>
</Edit_Mensaje>
```

D.5 Documentos XML bien formados

Los documentos denominados como "bien formados" (del inglés well formed) son aquellos que cumplen con todas las definiciones básicas de formato y pueden, por lo tanto, analizarse correctamente por cualquier analizador sintáctico (parser) que cumpla con la norma. Se separa esto del concepto de validez que se explica más adelante.

Los documentos han de seguir una estructura estrictamente jerárquica con lo que respecta a las etiquetas que delimitan sus elementos. Una etiqueta debe estar correctamente incluida en otra, es decir, las etiquetas deben estar correctamente anidadas. Los elementos con contenido deben estar correctamente cerrados.

Los documentos XML sólo permiten un elemento raíz del que todos los demás sean parte, es decir, solo pueden tener un elemento inicial. Los valores atributos en XML siempre deben estar encerrados entre comillas simples o dobles.

El XML es sensible a mayúsculas y minúsculas. Existe un conjunto de caracteres llamados espacios en blanco (espacios, tabuladores, retornos de carro, saltos de línea) que los procesadores XML tratan de forma diferente en el marcado XML. Es necesario asignar nombres a las estructuras, tipos de elementos, entidades, elementos particulares, etc. En XML los nombres tienen alguna característica en común.

Las construcciones como etiquetas, referencias de entidad y declaraciones se denominan marcas; son partes del documento que el procesador XML espera entender. El resto del documento entre marcas son los datos "entendibles" por las personas.

D.6 Partes de un documento XML

Un documento XML está formado por el prólogo y por el cuerpo del documento.

D.6.1 Prólogo

Aunque no es obligatorio, los documentos XML pueden empezar con unas líneas que describen la versión XML, el tipo de documento y otras cosas.

El prólogo contiene:

- Una declaración XML. Es la sentencia que declara al documento como un documento XML.
- Una declaración de tipo de documento. Enlaza el documento con su DTD, o el DTD puede estar incluido en la propia declaración o ambas cosas al mismo tiempo.
- Uno o más comentarios e instrucciones de procesamiento.

D.6.2 Cuerpo

A diferencia del prólogo, el cuerpo no es opcional en un documento XML, el cuerpo debe contener un y solo un elemento raíz, característica indispensable también para que el documento esté bien formado.

D.6.2.1 Elementos

Los elementos XML pueden tener contenido (más elementos, caracteres o ambos), o bien ser elementos vacíos.

D.6.2.2 Atributos

Los elementos pueden tener atributos, que son una manera de incorporar características o propiedades a los elementos de un documento. Deben de ir entre comillas.

D.6.2.3 Entidades predefinidas

Entidades para representar caracteres especiales para que, de esta forma, no sean interpretados como marcado en el procesador XML.

D.6.2.4 Secciones CDATA

Es una construcción en XML para especificar datos utilizando cualquier carácter sin que se interprete como marcado XML. Solo se utiliza en los atributos. No confundir con (#PCDATA) que es para los elementos.

D.6.2.5 Comentarios

Comentarios a modo informativo para el programador que han de ser ignorados por el procesador.

Los comentarios en XML tienen el siguiente formato:

```
<!--- Esto es un comentario --->
  <!-- Otro comentario -->
```

D.7 Validez

Que un documento esté "bien formado" solamente habla de su estructura sintáctica básica, es decir que se componga de elementos, atributos y comentarios como XML manda que se escriban. Ahora bien, cada aplicación de XML, es decir cada lenguaje definido con esta tecnología, necesitará especificar cuál es exactamente la relación que debe verificarse entre los distintos elementos presentes en el documento.

Esta relación entre elementos se especifica en un documento externo o definición (expresada como DTD (Document Type Definition = Definición de Tipo de Documento) o como XSchema). Crear una definición equivale a crear un nuevo lenguaje de marcado, para una aplicación específica.

D.7.1 Document type definition (DTD)

La DTD define los tipos de elementos, atributos y entidades permitidas, y puede expresar algunas limitaciones para combinarlos. Los documentos XML que se ajustan a su DTD se denominan válidos.

D.7.1.1 Declaraciones tipo elemento

Los elementos deben ajustarse a un tipo de documento declarado en una DTD para que el documento sea considerado como válido.

D.7.1.2 Modelos de contenido

Un modelo de contenido es un patrón que establece los subelementos aceptados, y el orden en que se aceptan.

D.7.1.3 Declaraciones de lista de atributos

Los atributos se usan para añadir información adicional a los elementos de un documento.

Tipos de atributos:

- Atributos CDATA y NMTOKEN
- Atributos enumerados y notaciones
- Atributos ID e IDREF
- Declaración de entidades

XML hace referencia a objetos que no deben ser analizados sintácticamente según las reglas XML, mediante el uso de entidades. Las entidades pueden ser:

- Internas o externas
- Analizadas o no analizadas
- Generales o parametrizadas
- Espacios de nombre

Los espacios de nombres XML permiten separar semánticamente los elementos que forman un documento XML.

D.7.2 XML Schemas (XSD)

Un Schema es algo similar a un DTD, define qué elementos puede contener un documento XML, cómo están organizados y qué atributos y de qué tipo pueden tener sus elementos.

Ventajas de los Schemas frente a los DTDs:

- Usan sintaxis de XML, al contrario que los DTDs.
- Permiten especificar los tipos de datos.
- Son extensibles.

Este elemento del Lenguaje XML se abordará en un apartado especial más adelante en este documento, pues forma una parte muy importante dentro del funcionamiento del Sistema.

D.8 XML Schema

La programación en Schema XML se basa en Namespaces. Podemos encontrar una analogía entre éstos y los llamados packages en Java. Cada Namespace contiene elementos y atributos que están estrechamente relacionados con el Namespace. Así, a la hora de definir un elemento o un atributo de un Namespace, siempre se creará una conexión entre los diferentes campos de éste. Además, esta forma de trabajar nos permite relacionar elementos que no están en el mismo Namespace.

Después de escribir un Schema XML se puede confirmar la correcta realización mediante la validación de esquemas XML: Validación XML.

Un ejemplo de la estructura de un documento esquema vacío sería el siguiente:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" version="0.1" xml:lang="es">
</xsd:schema>
```

Un ejemplo de definición con XML Schema sería el siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:element name="Libro">
       <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="Título" type="xsd:string"/>
            <xsd:element name="Autores" type="xsd:string" maxOccurs="10"/>
            <xsd:element name="Editorial" type="xsd:string"/>
         </xsd:sequence>
         <xsd:attribute name="precio" type="xsd:double"/>
      </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Podemos ver como en ambos casos se inician las declaraciones indicando la versión de XML que se va a utilizar y la codificación que se usa. Estos dos campos son necesarios para poder interpretar el esquema. Además, en la siguiente línea de código podemos ver como se redirecciona al usuario a la página que ofrece las pautas de creación de XML Schema en las que se basará la descripción del esquema.

El elemento raíz se llama “Libro” y tiene tres hijos (elementos anidados) y un atributo. Los hijos son “Titulo”,”Editorial” que deben aparecer una vez y “Autores” que puede aparecer de una a diez veces. El hecho de que estén agrupados en una secuencia indica que los elementos deben aparecer en orden, es decir, primero el “Titulo”, luego los “Autores” y por último la “Editorial”. Los tres elementos son de tipo string. El atributo de libro se llama “precio” y es de tipo double.

D.9 Bases de Datos XML

Es conocido por todos, que las bases de datos son una parte fundamental de todas las organizaciones, pues en ellas se almacenan información crucial para el buen desempeño de las mismas. Además de ello sabemos que XML es el presente y futuro de la administración de datos, pues este lenguaje ha permitido romper barreras y crear una manera estándar de procesar la información.

Pues bien, XML está provocando la aparición de nuevas tecnologías, entre ellas, la aparición de una nueva generación de bases de datos, que si bien se encuentran en una fase de investigación y desarrollo, en un futuro pueden ser una buena alternativa a las ya conocidas bases de datos relacionales. Estos son aquellos basados netamente en XML o "native XML database".

Este tipo de bases de datos son completamente distintas a las relacionales, las cuales en la actualidad tienen soporte para XML, pero aún siguen almacenando toda la información de manera relacional, es decir en forma tabular (tablas, registros y columnas) o caso contrario almacenan todo el documento en formato Binary Large Object (BLOB), pero la principal característica que brindan estas bases de datos es la capacidad de obtener los resultados de las consultas en formato XML; es por ello que dichas bases de datos pertenecen a la categoría de "XML-enabled database"

D.9.1 ¿En que consisten las Bases de Datos nativas en XML?

Las bases de datos nativas definen un modelo lógico para el documento XML (a diferencia de dicho modelo), además, almacena y recupera documentos de la misma manera que los XML. Por lo menos este modelo debe incluir atributos como PCDATA, y documentos en orden. Ejemplos de estos modelos son XPath, XML Infoset y modelos que implican DOM y SAX 1.0.

D.9.1.1 Base de datos centrada en documentos

Todas las bases de datos relacionales son centradas en los Datos, pues lo que ellos almacenan en sus campos son datos simples más conocidos como datos atómicos. Una base de datos nativa en XML, no posee campos, ni almacena datos atómicos, lo que ella almacena son documentos XML, por lo tanto a este tipo de bases de datos se les denomina bases de datos centradas en documentos.

D.9.1.2 Características

Diversos productos brindan diferentes características para las bases de datos nativas en XML, pero generalmente tienen las siguientes características:

- **Procesamiento de datos:** El procesamiento de datos en este tipo de bases de datos parecería ser algo muy beneficioso, pero realmente no es así, debido al formato jerárquico en el que está almacenada la información. Muchas bases de datos necesitan que recuperar todo el documento XML, actualizarlo con el XML API escogido y posteriormente volverlo a almacenar el documento en el repositorio. Esto se debe a que aún no existe un lenguaje estándar que permita la actualización, inserción o eliminación de elementos de un documento XML. Existe un lenguaje que permite realizar actualizaciones en un documento XML pero aún no es un

estándar y muchos gestores de este tipo de bases de datos no lo soportan, este lenguaje es Xupdate.

- **Almacenamiento:** Por deducción lógica, una base de datos nativa en XML almacena la información en formato XML, pero esto es solamente una deducción lógica, pues este tipo de bases de datos tienen repositorios con un formato "tipo XML", como puede ser DOM o Infoset. En este mismo "repositorio" (paquete de archivos) se almacenan los índices que se generan por cada documento XML almacenado.
- **Búsquedas:** Este tipo de bases de datos no utiliza SQL como lenguaje de consulta. En lugar de ello utilizan Xpath. Algunas bases de datos permiten seleccionar los elementos que deberán tener índices mientras que otras bases de datos indexan todo el contenido del documento. El problema que tienen las búsquedas en este tipo de bases de datos es que no permiten realizar búsquedas muy complicadas, como por ejemplo ordenamiento y cross join, debido a que Xpath no fue creada realmente para búsquedas en bases de datos, sino simplemente para búsquedas en un solo documento.

Muchas bases de datos permiten realizar búsquedas utilizando la tecnología Full-Text Search, de esta manera se pude agilizar la búsqueda de datos en los documentos XML.

D.9.1.3 Áreas de aplicación

Algunas áreas potenciales de aplicación podrían ser:

- Portales de información corporativa
- Información de catálogos
- BD en partes de manufactura
- Información medica
- BD personalizadas

Una gran parte de usos necesita encontrar documentos enteros. Por ejemplo, un portal Web podría permitir a usuarios buscar todos los documentos sobre una empresa particular y un sistema de dirección podría permitir a usuarios encontrar todos los documentos que se relacionen con una cierta parte. El modo menos complejo de buscar documentos es con búsquedas texto completas. En bases de datos natales XML, estos son XML-aware. Es decir esto distingue entre el contenido (que es buscado) y el margen (que no es). Búsquedas más complejas estructuralmente, que pueden preguntar el margen, el texto, o ambos. (XPath y XQUERY son los ejemplos en lenguajes de búsquedas estructuradas; bases de datos natales XML apoyan un número de lenguajes propietarios también.). Por ejemplo, considere las preguntas siguientes:

- Encuentre todos los articulos escritos despúes del 1 de Junio de 2004, con las palabras “eleccion presidencial” en el título.

```
for $a in collection("articulos")
where $a//Date > 2004-06-01 and
fn:contains($a//Title, "elección presidencial")
return $a
```

- Encuentes todos los procedimientos con más de 7 pasos.

```
for $p in collection("procedimientos")
let $s := $p//Pasos
where fn:count($s) > 7
return $p
```

Sin embargo estas búsquedas son relativamente simples, ninguno puede estar satisfecho por una búsqueda de texto completa: las dos primeras preguntas restringen la búsqueda a las ciertas secciones del documento y la tercera no pregunta el texto. Es también interesante notar que estas búsquedas no requieren que todos los documentos usen el mismo esquema; ellos sólo requieren que los documentos contengan ciertos elementos comunes que tienen aproximadamente el mismo significado.

D.9.2 eXist

eXist-db es un DBMS de código abierto construido totalmente con la tecnología XML. Almacena los datos XML de una manera eficiente, basado en los índices de procesamiento de XQuery. eXist-db soporta muchos estándares de la tecnología Web creando una excelente plataforma de aplicaciones:

- XQuery 1.0 / XPath 2.0
- XSLT 1.0 o XSLT 2.0
- Interfaces HTTP : REST, WebDAV, SOAP, XMLRPC, Atom Publishing Protocol
- XML database specific: XMLDB, XQJ/JSR-225 (en desarrollo), XUpdate, XQuery update extensions.

eXist-db es altamente compatible con el estándar XQuery. El motor de búsqueda es altamente extensible y cuenta con una gran colección de módulos de funciones XQuery. eXist-db provee un potente entorno para el desarrollo de aplicaciones Web basadas en Xquery. Aplicaciones Web pueden ser desarrolladas completamente utilizando XSLT, XHTML, CSS y JavaScript.

El sistema de gestión de la base de datos esta construido totalmente con la tecnología XML, por eso es llamada una base de datos XML nativa. A diferencia de la mayoría de los DBMS relacionales, Exist utiliza XQuery, el cual es un Estándar de la W3C, para manipular sus datos.

eXist-db provee soporte para los siguientes estándares y tecnologías:

- XPath - XML Path language
- XQuery - XML Query langage
- WebDAV - Web distributed authoring and versioning
- REST - Representational state transfer (URL encoding)
- SOAP - Simple Object Access Protocol
- XACML - XML Access Control Language
- XInclude - server-side include file processing (limited support)

Anexo E. Web Services

E.1 Introducción

Un servicio Web (en inglés Web Service) es una colección de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios Web para intercambiar datos en redes de ordenadores como Internet. La interoperabilidad se consigue mediante la adopción de estándares abiertos. Las organizaciones OASIS y W3C son los comités responsables de la arquitectura y reglamentación de los servicios Web. Para mejorar la interoperabilidad entre distintas implementaciones de servicios Web se ha creado el organismo WS-I, encargado de desarrollar diversos perfiles para definir de manera más exhaustiva estos estándares.

E.2 Estándares empleados

- Web Services Protocol Stack: Así se denomina al conjunto de servicios y protocolos de los servicios Web.
- XML (Extensible Markup Language): Es el formato estándar para los datos que se vayan a intercambiar.
- SOAP (Simple Object Access Protocol) o XML-RPC (XML Remote Producer Call): Protocolos sobre los que se establece el intercambio.
- Otros protocolos: los datos en XML también pueden enviarse de una aplicación a otra mediante protocolos normales como HTTP (Hypertext Transfer Protocol), FTP (File Transfer Protocol), o SMTP (Simple Mail Transfer Protocol).
- WSDL (Web Services Description Languages): Es el lenguaje de la interfaz pública para los servicios Web. Es una descripción basada en XML de los requisitos funcionales necesarios para establecer una comunicación con los servicios Web.
- UDDI (Universal Description, Discovery and Integration): Protocolo para publicar la información de los servicios Web. Permite comprobar qué servicios Web están disponibles.
- WS-Security (Web Service Security): Protocolo de seguridad aceptado como estándar por OASIS (Organization for the Advancement of Structured Information Standards). Garantiza la autenticación de los actores y la confidencialidad de los mensajes enviados.

E.3 Ventajas de los Servicios Web

- Aportan interoperabilidad entre aplicaciones de software independientemente de sus propiedades o de las plataformas sobre las que se instalen.
- Los servicios Web fomentan los estándares y protocolos basados en texto, que hacen más fácil acceder a su contenido y entender su funcionamiento.

- Al apoyarse en HTTP, los servicios Web pueden aprovecharse de los sistemas de seguridad firewall sin necesidad de cambiar las reglas de filtrado.
- Permiten que servicios y software de diferentes compañías ubicadas en diferentes lugares geográficos puedan ser combinados fácilmente para proveer servicios integrados.
- Permiten la interoperabilidad entre plataformas de distintos fabricantes por medio de protocolos estándar y abiertos. Las especificaciones son gestionadas por una organización abierta, la W3C, por tanto no hay secretismos por intereses particulares de fabricantes concretos y se garantiza la plena interoperabilidad entre aplicaciones. Es el futuro de las aplicaciones.

E.4 Inconvenientes de los Servicios Web

- Para realizar transacciones no pueden compararse en su grado de desarrollo con los estándares abiertos de computación distribuida como CORBA (Common Object Request Broker Architecture).
- Su rendimiento es bajo si se compara con otros modelos de computación distribuida, tales como RMI (Remote Method Invocation), CORBA, o DCOM (Distributed Component Object Model). Es uno de los inconvenientes derivados de adoptar un formato basado en texto. Y es que entre los objetivos de XML no se encuentra la concisión ni la eficacia de procesamiento.
- Al apoyarse en HTTP, pueden esquivar medidas de seguridad basadas en firewall cuyas reglas tratan de bloquear o auditar la comunicación entre programas a ambos lados de la barrera.

E.5 Razones para crear Servicios Web

La principal razón para usar servicios Web es que se basan en HTTP sobre TCP (Transmission Control Protocol) en el puerto 80. Dado que las organizaciones protegen sus redes mediante firewalls -que filtran y bloquean gran parte del tráfico de Internet-, cierran casi todos los puertos TCP salvo el 80, que es, precisamente, el que usan los navegadores. Los servicios Web se enrutan por este puerto, por la simple razón de que no resultan bloqueados.

Otra razón es que, antes de que existiera SOAP, no había buenas interfaces para acceder a las funcionalidades de otros ordenadores en red. Las que había eran ad hoc y poco conocidas, tales como EDI (Electronic Data Interchange), RPC, u otras Application Programming Interface APIs.

Una tercera razón por la que los servicios Web son muy prácticos es que pueden aportar gran independencia entre la aplicación que usa el servicio Web y el propio servicio. De esta forma, los cambios a lo largo del tiempo en uno no deben afectar al otro. Esta flexibilidad será cada vez más importante, dado que la tendencia a construir grandes aplicaciones a partir de componentes distribuidos más pequeños es cada día más asada. Se espera que para los próximos años mejoren la calidad y cantidad de servicios ofrecidos basados en los nuevos estándares.

Anexo F. Usabilidad Móvil

F.1 El contexto de uso del teléfono móvil

El entorno en el que un usuario usa un teléfono móvil y las características del propio dispositivo son dos elementos clave a tener en cuenta en el diseño de interfaces de aplicaciones. En el caso del móvil, el entorno es cambiante, dinámico. El usuario puede estar distraído o tener prisa, por lo que la estructura de navegación tiene que ser muy simple y hay que evitar los pasos innecesarios. Algunas personas, por ejemplo, compran las entradas de cine apresuradamente desde su móvil mientras hacen cola en el propio cine si sospechan que la sala se llenará antes de ser atendidos en la taquilla. También son frecuentes las consultas de saldo de una cuenta bancaria antes de realizar una compra (sobre todo si va a emplearse una tarjeta de débito). En ambos casos el usuario tiene mucha prisa y se encuentra en un lugar público. Por otro lado, la tarea que está realizando el usuario puede interrumpirse por pérdida de cobertura, por una llamada entrante o por una simple distracción. Por lo tanto, el diseño debería permitir recuperar el proceso en curso tras la interrupción (adquiere especial importancia el principio de no hacer que el usuario tenga que recordar las cosas). Las distracciones, concretamente, son mucho más habituales de lo que uno podría pensar. Es habitual usar el móvil (para otros objetivos que no sean el de llamar) mientras se está esperando en un restaurante, en un andén o incluso en un semáforo. En estas situaciones se dan interrupciones por pura lógica:

- El camarero trae la comida o la bebida, llega el tren, el semáforo se pone verde.
- Otros aspectos que actualmente hay que tener en cuenta son:
- Los usuarios saben que pagan por tiempo o tráfico, esto se da en conexiones por medio de GPRS, lo que no es aplicable al dominio del sistema en desarrollo.
- Percepción de los usuarios respecto del medio: ¿es seguro?
- Los teléfonos y el medio están optimizados para voz, no para datos.

Se conectan para realizar tareas concretas, no para navegar. De hecho, un estudio de julio de 2003 realizado en Japón revela que el uso de Internet en los móviles (MobileNet) se concentra en el servicio de e-mail y chat (75,7%), seguido por la descarga de imágenes y tonos (5,2%). Ningún otro servicio superó el 4% (Noticias, información sobre el tráfico y el tiempo, entretenimiento, banca, compras, etc.).

F.2 Heterogeneidad de dispositivos

Los dispositivos en sí tienen unas características físicas que afectan negativamente a la usabilidad de una aplicación, como por ejemplo:

- Tamaño de la pantalla
- Tamaño de las teclas
- Dificultad para escribir texto

- Ancho de banda e inestabilidad de la conexión (esto, en realidad, atañe más al servicio que al dispositivo), sobre todo en conexiones GPRS.

De todas formas, el problema más grave es la heterogeneidad de los dispositivos. A diferencia de un ordenador, que prácticamente si sabes usar uno enseguida eres productivo usando cualquier otro, aquí las limitaciones inherentes al dispositivo, junto con la heterogeneidad de los mismos son una fuerte barrera a la usabilidad:

- Algunos tienen más líneas y caracteres por línea (los hay que sólo tienen 3 líneas de 14 caracteres de longitud).
- Cada uno tiene puestas de una manera las hardkeys (controles hardware) y softkeys (controles programables que aparecen típicamente en la parte inferior de la pantalla). Esto hace que las teclas de borrar e ir atrás, por ejemplo, puedan estar mapeadas en lugares distintos en cada móvil.
- Algunos usan fuentes proporcionales, otros no; algunos tienen negritas y otros estilos, otros no.
- Algunos pueden mostrar imágenes, otros no.
- Algunos soportan escritura predictiva, otros no.
- Los caracteres especiales (puntos, comas, paréntesis, acentos, etc.) cada modelo los tiene en teclas diferentes.
- Las paletas de colores pueden ser diferentes (si es que el móvil tiene colores).
- Algunos tienen acceso a menús vía teclado numérico, otros no.
- El formato de los enlaces y de las barras de scroll puede diferir en función del móvil.
- Muchos fabricantes implementan extensiones propias del lenguaje estándar.

En general se sigue un estándar para el mapeo de los números y de las letras, y también es habitual encontrar 2 softkeys. Otros controles no están tan estandarizados, como por ejemplo el número, posición y forma de las hardkeys, el mapeo de caracteres especiales (blanco, signos de puntuación, etc.) y las teclas de navegación (botón para ir atrás, menús con opciones, control de paginación).

F.3 Criterios fundamentales de diseño

F.3.1 Escribir es difícil

Tanto si es en modo triple tap como con escritura predictiva, esta tarea resulta difícil. Siempre es preferible la selección de opciones a la escritura, aunque lleve más pasos. Si no hay más remedio, es importante tener controlado el formato de entrada (evitar la escritura predictiva a menos que sea muy clara la ventaja de hacerlo).

Se recomienda preinformar campos siempre que sea posible predecir el valor más probable de entrada. En este sentido, hay que garantizar que si el usuario va hacia atrás no se “limpien” los campos ya rellenados.

F.3.2 Menos es más

No hay que dar datos simplemente porque se tienen; sólo hay que dar información relevante. Por ejemplo, en la pantalla de lista de cuentas bancarias de un usuario no haría falta indicar los céntimos del saldo de cada cuenta (aunque sí habría que hacerlo en la consulta del detalle de la cuenta). Igual pasa en fechas: no dar minutos y segundos de una operación si no son relevantes. En este sentido también es recomendable usar abreviaturas y un estilo de escritura conciso.

La información más importante debe aparecer en la parte de arriba de la pantalla, y las acciones por defecto (mapeadas en las softkeys) deben de ser las más importantes. Hay que vigilar con el uso del espacio en blanco, porque las líneas vacías pueden despistar e inducir al usuario a pensar que no hay nada más bajo ellas. Es mejor usar como separador una serie de guiones: "-------".

Hay que evitar contenidos multimedia, gráficos y efectos visualmente vistosos. Es preferible diferenciarse por la calidad de los contenidos y los servicios que ofrezca la aplicación, no por la interfaz de usuario.

F.3.3 El usuario es móvil

Partiendo de la idea de que gran parte de los usuarios se conectan con objetivos concretos y que pagan por tiempo o Bytes, hay que tener cuidado con alterar su camino principal de acción con publicidad, contenidos poco relevantes o pasos innecesarios.

F.3.4 Cada pulsación de tecla empeora la usabilidad

Es aconsejable reducir el número de pulsaciones de tecla entre el principio y el final de cada tarea. Esto no sólo afecta a la introducción de texto, también requieren pulsación el scroll vertical, el despliegue de menús y las softkeys.