Tecnología WAP, WML y XHTML.

Autores

Matias Borgeaud

Pablo Paccapelo

Tabla de Contenido

Tabla de Contenido 2

Agradecimientos 6

Prefacio 10
- Presentación 11
- Objetivos de la tesis 11
- Introducción al WAP, WML y XHTML 12
- Estado del Arte 14
- Aportes 18
- Filosofía de la tesis 18
- Estructura 19

Capítulo 1. Las comunicaciones Celulares 21
- Principios de la comunicación móvil 22
  - La Arquitectura temprana 23
  - El concepto Celular 23
  - La Arquitectura del Sistema Celular 24
  - Celdas y Clusters 24
  - El reuso de frecuencias 25
  - La división de celdas 26
  - Handoff: Traspaso entre celdas 27
  - La Red AMPS 28
- Sistemas Digitales 28
  - TDMA y CDMA 28
- El protocolo WAP 28
  - Motivación 29
  - Usos 30
  - Dispositivos con soporte WAP 30
  - Sistemas Operativos que soportan WAP 30
  - Infraestructura de redes que dan soporte WAP 30
  - Redes fijas versus redes inalámbricas 30
  - Puntos de usabilidad 32
- La OMA 32
  - La especificación WAP de la OMA 33

Capítulo 2. La Arquitectura WAP 34
- Introducción 35
- El protocolo WAP 35
  - Los beneficios del WAP 36
  - Similaridad con Internet 37
  - Arquitectura 39
  - WAP Push Service 41
  - Seguridad en WAP 44

Capítulo 3. La Tecnología WAP 46
Introducción a la tecnología WAP 47
Componentes de la tecnología WAP 47
Proxies y Gateways WAP 48
Servidores de Soporte y WTA Servers 50
Web y WAP Servers 52
Configurando un servidor para brindar contenido WML: Tipos MIME 52

Capítulo 4. WML, El lenguaje de programación WAP 53
Introducción al WML 54
El lenguaje WML 54
Bloques de construcción del WML 54
Elementos del mazo 60
Elementos de evento 61
Elementos de tarea 62
Variables 64
Elementos de entrada de usuario 65
Imágenes y anchors 69
WBMP, Imágenes en WML 71
Formato de texto en WML 72
El DTD 76

Capítulo 5. WMLScript, el lenguaje de scripts de WML 81
Agregando lógica del lado del cliente 82
Sintaxis del lenguaje WMLScript 83
Las Librerías Estándar de WMLScript 84
Un ejemplo de WMLScript 85

Capítulo 6. XHTML, la evolución del WML 88
Introducción 89
Markup Languages 89
WAP y WML 90
Evolución del WAP 91
Evolución del WML 92
El Lenguaje XHTML 95
XHTML Básico 96
XHTML MP 97
Comparación entre el WML1.x y XHTML MP + WAP CSS ... 124
XHTML comparado con el HTML 125
CSS, Hojas de estilo en XHTML 128
Migrando desde el WML1.x a XHTML MP 136
El DTD 140
Protocolos de Transmisión 141
Cuestiones acerca de los navegadores 142
Configurando un servidor para brindar contenido XHTML/WML 145

Capítulo 7. Construyendo Aplicaciones WAP 146
Reglas de Usabilidad generales 147
Organizando el sitio WAP 148
Displays grandes y displays chicos 153
Principios de navegación 155
Dos ejemplos de aplicación WAP 156

Capítulo 8. UAProf, Adaptación de contenidos WAP 162
El desafío 163
Arquitectura 164
Adaptando contenidos 165
Atributos 166
Los Beneficios 167
Estandarización 168

Capítulo 9. El Futuro del WAP 170
La industria móvil 171
De dónde venimos 172
Los Escalones previos: de 1G a 3G 172
La primera generación (1G): Sistemas Celulares Analógicos 173
La segunda generación (2G): Sistemas Digitales Múltiples ... 173
La tercera generación (3G) 174
Hacia dónde vamos 174
WAP en el ambiente competitivo 175

Conclusión 177
Haciendo móviles a las aplicaciones Web 177
Los gateways WAP proveen conectividad 179
Un ojo crítico sobre las aplicaciones existentes 179
Mirando hacia adelante 180

Apéndice A. SMS 184
Definición 186
Introducción 186
Beneficios del SMS 187
Elementos de Red y Arquitectura 188
Funcionamiento 189
Aplicaciones SMS 192

Apéndice B. Glosario y Referencias 195
Términos y abreviaturas 197
Referencias 207

Apéndice C. Código y archivos 209

Apéndice D. Siu WAP 210

Apéndice E. El RUP aplicado 211

Modelo del Dominio....213
Requerimientos y Modelo de Casos de Uso 216
Modelo del Análisis....219
Modelo de Navegación....221
Modelo del diseño....222

Agradecimientos

Por Matias Borgeaud

En primer lugar, quisiera dedicar un par de párrafos para mí. Contar que siempre he soñado con ser alguien, y que he puesto mi mayor empeño en no ser sólo uno más.
No suelo conformarme con mis logros, y ésta es la prueba más grande de mi esfuerzo. Espero que los años que le sigan a estos veintiséis que ya he vivido, me llenen de pruebas cada vez mayores, para poder sortearlas con las mismas ganas que he expresado a lo largo de esta tesis, y poder mirar atrás y encontrar a quienes me aman alentándome a seguir por más.

Le agradezco en primer lugar a mi madre por ese espíritu inmenso y esa persona tan encantadora que es. Mami, sos mi guía y modelo de persona. Gracias por estar ahí siempre.

A mi familia, Susana, Jorge, Esteban y Celina, y a mis abuelos, Humberto y Beba, por apoyarme en las buenas y en las malas,
Los quiero con toda mi alma.

A Pablo, por ser un amigo incondicional y ayudarme a demostrar que si se quiere, se puede. Siempre. Gracias, Pacca.

Le agradezco también al Dr. Luis Olsina por las palabras que alguna vez me dedicó y que tanto han sido de inspiración para el que hoy soy.

A José Luis Filippi, que durante estos últimos tres años ha confiado tanto en mí y me ha dejado ser.

Y, por último, quiero dejarle a mi abuelo una parte de lo que él significa para mí, la persona más grande y llena de bien que he conocido. ¡Te quiero, Tata!

Te escapaste a devolverle a alguna nube,
el color que le quitaste para darle a tu cabello,
y así ganaste tiempo.

Te embarcaste por los mares y los cielos
buscando qué inspiraba el color de tu mirada,
y te olvidaste del regreso.

Te colaste entre un gentío que camina por tu Italia
y mezclaste tus sonidos con los pasos de esa gente,
y te volviste imperceptible.

Te mudaste a tus campos cabalgando por tu infancia,
acariciaste tus recuerdos y a tu madre que esperaba,
y te dormiste entre sus brazos.

Cambiaste tu nombre, tu apellido y dirección,
te mudaste a un sitio enorme y lleno de espacios,
y te volviste superior.

Le ganaste a todo, al invierno y al verano,
y naciste en primavera, un veintiuno de septiembre
y trazaste mi camino.

Hoy, que ya no estás más adelante,
me consuela saberte arriba, caminando como un ángel,
y cuidándome de a ratos.

Vos, que le ganaste al tiempo, y al invierno y al verano,
que te fuiste en busca de algo y te olvidaste del regreso,
que te hiciste imperceptible entre los brazos de tu madre,
que te hiciste superior y desde arriba, caminando como un ángel,
te dedico este camino, que trazaste en primavera,
y que hoy ando para vos.

A tu memoria, Humberto.

Agradecimientos

Por Pablo Paccapelo

En primer lugar, quiero agradecer a la gran familia que tengo, Mamá, Papá, Vale, Ren, a mis abuelos, tíos y primos, que nunca dejan de sorprenderme y que siempre me brindan todo su apoyo y cariño en todos mis emprendimientos. Estoy orgulloso de todos ustedes, gracias por caminar a mi lado, me hacen sentir muy seguro. Los quiero mucho.

A Marisol por ayudarme, brindarme afecto y compartirme con esta tesis por veces tan absorbente.

A mis compañeros, con los cuales compartí horas de estudio y recreación, por los buenos momentos que pasamos.

A Matías, por compartir horas de trabajo incesantes, por su voluntad y por sus ganas constantes de mejorar las cosas.

Quiero dar las gracias a todas las personas de buena voluntad, muchas de ellas desconocidas, que disfrutan la vida y que ayudan a construir un mundo mejor.

Para finalizar, a través de la letra de la canción One de U2, quiero significar mi deseo de disfrutar la vida al máximo y no decepcionar a nadie jamás.

Is it getting better?
Or do you feel the same?
Will it make it easier on you?
Now you got someone to blame?

You say
One love
One life
When it's one need
In the night
It's one love
We get to share it
It leaves you baby
If you don't care for it

Did I disappoint you?
Or leave bad taste in your mouth?
You act like you never had love
And you want me to go without

Well it's too late
Tonight
To drag the past out
Into the light
We're one
But we're not the same
We get to carry each other
Carry each other
One

Have you come here for forgiveness?
Have you come to raise the dead?
Have you come here to play Jesus?
To the lepers in your head

Did I ask too much?
More than a lot
You gave me nothing
Now it's all I got
We're one
But we're not the same
We hurt each other
Then we do it again

You say
Love is a temple
Love a higher law
Love is a temple
Love the higher law
You ask me to enter
But then you make me crawl
And I can't be holding on
To what you got
When all you got is hurt

One love
One blood
One life
You got to do what you should

One life
With each other
Sisters
Brothers

One life
But we're not the same
We get to carry each other
Carry each other
One.

Prefacio

Presentación
Objetivos de la tesis
Introducción al WAP, WML y XHTML
Estado del Arte
Aportes
Filosofía de la tesis
Estructura

Prefacio

Presentación

Este proyecto final de la carrera Analista Programador en Computación que hemos de desarrollar contempla un trabajo de investigación sobre la tecnología WAP y los lenguajes de programación WML (especificación WAP 1.x), y XHTML MP (especificación WAP 2.0).

El origen de este proyecto de investigación tiene lugar en el año 2003, mientras cursábamos la materia Laboratorio a cargo del Lic. Guillermo Lafuente, quien propusiera este tema para realizar una investigación para la materia. Particularmente, el tema despertó en nosotros interés y curiosidad, por lo que decidimos extender la investigación y presentarla como nuestro proyecto final.

Para cerrar el círculo de la investigación sobre la tecnología WAP, decidimos realizar una implementación de un sistema de información accesible mediante dispositivos móviles para la consulta de resultados y fechas de exámenes de las materias de la facultad de Ingeniería. El sistema involucra a los alumnos que estén inscriptos en materias y que deseen recibir notificaciones mediante este servicio a través de SMS o email WAP. Además el servicio cuenta con un portal de navegación que permite a los usuarios realizar las mismas consultas, tanto a través de dispositivos móviles como de navegadores Web estándar.

Para el desarrollo de este trabajo de tesis, utilizamos los conocimientos de las materias cursadas, bibliografía sobre el tema y material descargado de Internet.

Objetivos de la tesis

Los objetivos de este trabajo de investigación son nutrirnos intelectualmente y profundizar los conocimientos sobre el tema, además de establecer una base para que las personas interesadas en el tema comiencen su investigación y le den continuidad a través de los años.

Además de que este trabajo sirva para concluir nuestra carrera, es nuestra principal motivación, dejar un valioso

aporte a la comunidad de esta facultad que nos ha formado durante los años de cursada de la carrera Analista Programador en Computación y demostrar que con empeño y voluntad se puede lograr un desarrollo tan rico y concluyente como el que estamos introduciendo en esta investigación, siempre teniendo en mente que, aún con pocos recursos es posible.

Si bien el tema es muy amplio, quisimos en esta tesis, realizar un compendio de las porciones más importantes de la tecnología y los lenguajes de programación que le dan soporte.

Es nuestra intención que esta investigación sirva como una guía de soporte para el desarrollo de aplicaciones en el ambiente WAP, y gracias a ésta permitir a un principiante en el tema, poder atravesar las diferentes etapas en el desarrollo de aplicaciones.

Con este trabajo pretendemos mostrar el resultado de nuestras experiencias durante la carrera, aprovechando lo bueno y tratando de evitar caer en las falencias que tanto degradan una presentación y desvalorizan una investigación.

Introducción al WAP, WML y XHTML

Con la evolución de las comunicaciones, y las necesidades de estar en contacto con la información independientemente de dónde esté uno ubicado, surgen las comunicaciones celulares que echan por tierra la necesidad de estar conectados mediante los medios físicos.

El protocolo WAP (Wireless Application Protocol) es una especificación global y abierta que amplía las capacidades de los dispositivos móviles para acceder fácilmente e interactuar con información y servicios de forma instantánea. Este es uno de los servicios de valor agregado de la red celular digital que tiene la mayoría de los países medianamente desarrollados.

WAP es la convergencia de tres tecnologías de red de evolución rápida: data wireless, telefonía e Internet. Ambos mercados, el de wireless data y el de Internet, están creciendo muy rápidamente y continúan para llegar a más clientes. El crecimiento explosivo de la Internet ha potenciado la creación de nuevos y excitantes servicios de información.

La mayoría de las tecnologías originales desarrolladas para la Internet fueron diseñadas para computadoras de escritorio y computadoras conectadas a redes con anchos de banda de grandes a medianos, generalmente redes de datos confiables. En el mercado masivo, los dispositivos inalámbricos presentan un ambiente más restringido comparado con las computadoras de escritorio.

La OMA (Open Mobile Alliance) es una alianza, como su nombre lo indica, entre empresas proveedoras de equipos, especialmente de telefonía celular móvil, con el fin de crear un estándar y la especificación WAP, así como dar bases a la OMAI (Open Mobile Architecture Initiative), un conjunto de principios que aseguran la interoperabilidad y el beneficio de los usuarios finales en la experiencia de los servicios WAP y con el propósito de "extender las barreras del mercado"- según su lema.

WML (*Wireless Markup Language*) es un lenguaje de marcas basado en el XML (*Extensible Markup Language*) usado para crear contenido destinado a pantallas en dispositivos móviles. WML es una aplicación XML que permite formatear texto para un ambiente WAP. WML es el resultado de las primeras especificaciones WAP, conocidas como WAP 1.x.

La nueva generación de la especificación de servicios móviles es WAP 2.0, la cual está apoyada por las especificaciones de navegación de la Open Mobile Alliance (OMA). Hasta WAP 2.0, había dos especificaciones diferentes para los lenguajes usados en el mundo de las redes móviles y fijas: HTML para la World Wide Web (WWW) y WML para los servicios inalámbricos.

XHTML acerca la Internet móvil a la Internet fija; siendo la única especificación para la Web móvil y para la Web tradicional. XHTML permite mejor y más fácil control sobre la presentación que el WML, de ahí que mejora la usabilidad de los servicios móviles para los consumidores. La especificación WAP 2.0 tiene dos elementos clave en términos de navegación móvil: Juntamente con los cambios del lenguaje WML al XHTML MP, los cambios del protocolo de transmisión WAP al Wireless Profiled TCP/IP (wTCP/IP). Ambos XHTML y TCP/IP son estándares en la Internet fija y su migración soporta la convergencia entre los dominios fijos y móviles.

Argentina es uno de los países de Latinoamérica con más alto índice en el uso de telefonía celular: 5 millones de líneas celulares se encuentran activas actualmente. Toda la infraestructura tecnológica celular Argentina es reciente y existe un mercado constituido, lo que hará más fácil la incorporación de esta nueva tecnología entre los consumidores.

Para hacer una introducción más sólida de este desarrollo de investigación, podemos citar algunos de los servicios WAP que brindan empresas—tanto nacionales como extranjeras—como valor agregado a la prestación de sus servicios clásicos, y a los que estamos acostumbrados.

Páginas Doradas.

Páginas Doradas: Acceso a anunciantes con búsquedas por Nombre, Palabra Clave o Teléfono (permite acotar la búsqueda por provincia).

Fig.P-1 Sitio WAP de Páginas Doradas (http://www.paginasdoradas.com)

Lufthansa.

Salidas y llegadas de vuelos. Información sobre los vuelos de cabotaje. Check-In y reservas.

Fig.P-2 Sitio WAP de Lufthansa (http://wap.lufthansa.com)

Liverpool Football Club.

El portal WAP de LFC ofrece las últimas noticias sobre el club, información sobre partidos, resultados, artículos de prensa, etc.

Fig.P-3 Sitio WAP de LFC (http://wap.liverpoolfc.tv/)

WINWAP.

Servicios móviles generales como noticias, chat, búsqueda de personas, etc.

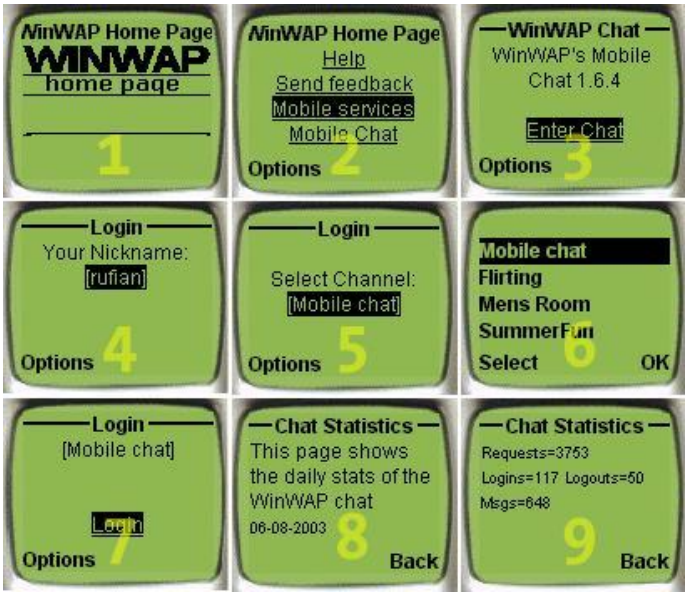


Fig.P-5 Sitio WAP de WINWAP (http://winwap.com)

Google.

El buscador clásico Google también tiene su portal WAP para realizar búsquedas, ya sea de información sobre la WWW o sobre WAP. Esta información incluye sitios, imágenes, y cualquier otra colección de páginas diseñadas específicamente para teléfonos móviles.

Fig.P-3 Sitio WAP de Google (http://www.google.com.ar/)

InterHabit.

Información sobre alojamiento en lugares turísticos de Argentina, Brasil, Chile y Uruguay.
La información incluye: hoteles, cabañas, alojamiento, hotel apart, posadas.

(El portal WAP y el sitio de Internet comparten una base de datos común para asegurar la calidad y precisión de toda la información disponible).

Fig.P-4 Sitio WAP de InterHabit (http://interhabit.com/wap/)

Además de aplicaciones WAP, en Internet se pueden encontrar sitios que proveen hosting WAP para páginas WML, aunque en su mayoría no son gratuitos y no son muy populares. Sin embargo, el sitio @WAPHOST (http://www.awaphost.com/), provee hosting gratuito con la limitación de espacio para un solo mazo WML.

Puede ser muy útil en ciertas ocasiones (o necesario, tal vez), poder contar con un emulador WAP online, ya sea para mostrar una aplicación en tiempo de desarrollo, o para brindarle a los usuarios una forma similar de navegar contenidos en un dispositivo móvil, pero desde una PC de escritorio. Para tales casos, YOSPACE.COM (http://www.yospace.com) dispone de su SmartPhone Emulator.

Aparte de los clásicos, como Google o Yahoo!, existen buscadores especiales WAP; este es el caso de WAPALL Argentina (http://www.wapall.com/en/main.asp?Co=Argentina&u=107).

A través de los ejemplos anteriores se quiso mostrar una pequeña porción de la gran variedad de sitios WAP que existen para Internet móvil. El espectro que cubren las aplicaciones WAP existentes, es de lo más amplio, en su mayoría sitios de información, comerciales y de entretenimiento destinados a toda la gama de consumidores.

Otras referencias a sitios WAP en Argentina:

Compras: Cuánto y Dónde
http://wap.cuantoydonde.com

Entretenimiento: Planeta Global
http://www.planetaglobal.com/wap

Portales: Radar
http://wap.radar.com.ar

Horóscopo: Guía del Ocio
http://www.guiadelocio.com.ar/wap

Viajes: Despegar
http://www.despegar.com/wap/argentina.asp

Buscadores: Grippo
http://wap.grippo.com.ar

Deportes: Supergol
http://www.supergol.com/wap

Noticias: Empresas News
http://wap.empresasnews.com

Aportes

Como ya citamos entre los objetivos de este desarrollo de investigación, los aportes de esta tesis son de carácter educativo más que una mejora de algún servicio o alguna tecnología ya existente, de todas formas, a través del uso de estas tecnologías, se puede dar una forma alternativa de acceso a información a dispositivos con características móviles.

Si bien, este trabajo es meramente educativo, los contenidos presentados, junto con la aplicación propuesta sirven como pie para el desarrollo de herramientas que sean productivas y que añadan un valor agregado a servicios o aplicaciones ya existentes, como en este caso, el sistema de información de la Facultad de Ingeniería.

Filosofía de la tesis

Esta tesis está desarrollada bajo una filosofía de libre distribución, esto es, que todo el material, código y la misma aplicación final se distribuye libremente y, previo consentimiento expreso de los autores, con los permisos para extraer información total o parcial.

Por otro lado, nos interesaría que se nos comunicara cualquier uso de la misma y si es posible generar un flujo mutuo de aprendizaje e intercambio de conocimientos entre ambos grupos de desarrollo.

Además, el servicio que brinda la aplicación propuesta no requiere de grandes recursos de software y hardware, pudiendo ejecutarse sin problemas en cualquier plataforma de medianas prestaciones y sin afectar el rendimiento del hardware que soporte a la aplicación y a un posible sistema que colabore en conjunto con nuestro servicio, proveyendo una sinergia perfecta con otros sistemas de información destinados a otros dispositivos.

Estructura

Este documento está organizado en nueve capítulos, una conclusión, cinco apéndices y un CD-ROM con código fuente de aplicaciones, el código del sistema de información propuesto y un conjunto de DTDs y Perfiles de Usuario UAProf que pueden resultar de utilidad para el desarrollo de futuras aplicaciones.

• El primer capítulo tiene como objetivo introducir los principios básicos de las comunicaciones celulares, y servir como punto de partida para el resto de los capítulos, y a su vez ubicar la tecnología WAP dentro de las comunicaciones móviles.

• El segundo capítulo es una profundización más formal de los conceptos relacionados con la arquitectura WAP y sus principios básicos.

• El tercer capítulo aborda al WAP desde el punto de vista de la tecnología que involucra y el hardware que le da soporte.

• El cuarto capitulo trata sobre el lenguaje WML para la especificación WAP 1.x, como el lenguaje de programación destinado a producir contenido para los dispositivos móviles.

• El quinto capítulo reseña el lenguaje de script WMLScript y plantea las ventajas de su utilización como una forma de agregar lógica del lado del cliente.

• El sexto capítulo está destinado a mostrar la evolución del lenguaje WML en el lenguaje XHTML que estandariza la

estructura de presentación de los contenidos, y es la convergencia entre los lenguajes orientados a la Web fija y a la Web móvil.

• El séptimo capítulo es una guía para el desarrollo y construcción de aplicaciones WAP bien formadas. Provee una serie de reglas y principios de usabilidad y navegación así como ciertas cuestiones para la correcta organización de un sitio WAP.

• El octavo capítulo propone una solución estándar para la adaptación de contenido a los diferentes dispositivos que pueden tener acceso al mismo, mediante los perfiles de usuario UAProf.

• El noveno capítulo hace una recorrida a través de las generaciones de la tecnología WAP y hace una proyección sobre la dirección a la que apunta su evolución.

• La conclusión es una apreciación personal del resultado de esta investigación, incluyendo una crítica objetiva sobre el estado actual del arte, una breve reseña de la evolución en el tiempo de las tecnologías expuestas en esta investigación y una mirada hacia el futuro tan espectacular como promete la tecnología.

• El apéndice A hace un enfoque práctico sobre la funcionalidad Short Message Service (SMS) que presta el protocolo WAP.

• El apéndice B contiene toda la terminología usada en esta tesis, abreviaciones y acrónimos, a modo de glosario.

• El apéndice C reúne un conjunto de archivos, documentos y ejemplos que acompañan la etapa de aprendizaje y son de utilidad a la hora de una posible implementación.

• El apéndice D ilustra el proceso de desarrollo unificado (RUP), que se aplicó para la realización del sistema.

• El apéndice E contiene el código fuente de la aplicación propuesta.

Capítulo 1. Las comunicaciones Celulares

Principios de la comunicación móvil
La Arquitectura temprana
El concepto Celular
La Arquitectura del Sistema Celular
Celdas y Clusters
El reuso de frecuencias
La división de celdas
Handoff: Traspaso entre celdas
La Red AMPS

Sistemas Digitales
TDMA y CDMA

El protocolo WAP
Motivación
Usos
Dispositivos con soporte WAP
Sistemas Operativos que soportan WAP
Infraestructura de redes que dan soporte WAP
Redes fijas versus redes inalámbricas
Puntos de usabilidad

La OMA
La especificación WAP de la OMA

Capítulo 1. Las comunicaciones Celulares

Principios de la comunicación móvil

Los sistemas de comunicaciones móviles usan un gran número de transmisores inalámbricos de bajo alcance para crear celdas—el área de servicio básico de un sistema de comunicación inalámbrico. Los niveles de alcance variables permiten a las celdas ser ajustadas de acuerdo a la densidad de usuarios y la demanda dentro de una región particular. A medida que un usuario viaja de celda en celda, sus conversaciones son transferidas entre las celdas en orden de mantener el servicio. Los canales (frecuencias) usados en una celda pueden ser reusados en otras celdas alejadas. Se pueden agregar celdas para adaptarse al crecimiento, creando nuevas celdas en áreas sin servicio o superponiendo celdas en áreas ya existentes.

Cada móvil usa un canal de radio temporal e independiente, para comunicarse con la "base" dentro de una celda. Esta base se comunica con varios móviles a la vez, usando un canal por móvil. Los canales usan un par de frecuencias para la comunicación—una frecuencia para emitir y otra para recibir llamadas. Las ondas de radio se disipan con la distancia, por lo que los móviles deben permanecer cerca de la estación base para mantener las comunicaciones. La estructura básica de las redes móviles incluye sistemas telefónicos y servicios de radio. Esta estructura se muestra en la figura 1-1 a continuación.

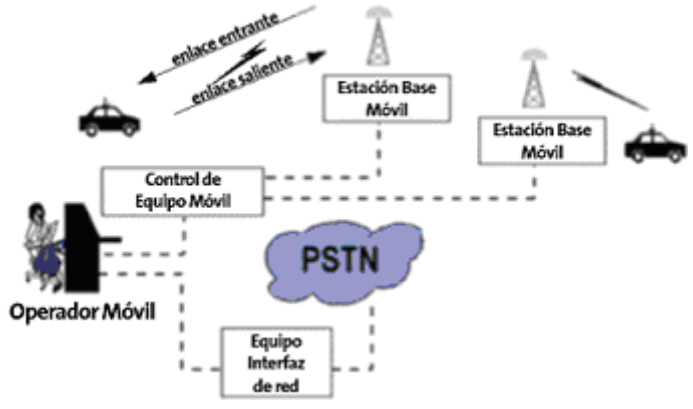


Fig.1-1 Estructura básica de las redes de servicio móvil

La Arquitectura temprana

El servicio móvil tradicional fue estructurado de forma similar al de las transmisiones de televisión: Un transmisor muy potente ubicado en el punto más alto del área transmitiría en un radio por encima de los 50 Km. El concepto celular estructuró a la red telefónica móvil de una forma diferente. En lugar de usar un transmisor muy potente, varios transmisores de menor poder son ubicados a lo largo del área a cubrir. Por ejemplo, dividiendo una región metropolitana en cientos de áreas diferentes (celdas) con transmisores de baja potencia usando varios canales de conversación cada una. La figura 1-2 muestra un área metropolitana configurada como una red tradicional telefónica móvil con un único transmisor de alta potencia.

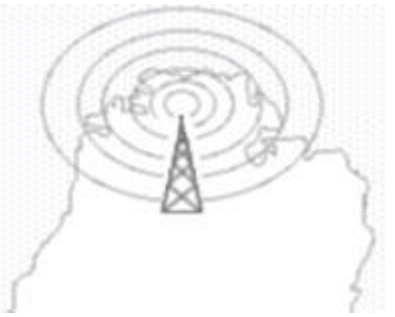


Fig.1-2 Arquitectura temprana de sistemas de telefonía móvil

El concepto Celular

Los problemas de interferencia causados por las unidades móviles usando el mismo canal en áreas adyacentes hace que no todos los canales puedan ser reusados. Se debía entonces, saltear algunas áreas antes de que el mismo canal pudiese ser reusado.

Los ingenieros descubrieron que los efectos de las interferencias no se debían a la distancia entre las áreas, sino al radio de las áreas en sí (distancia entre el transmisor y el límite del área). Reduciendo el radio al cincuenta por ciento, los proveedores de servicio podían incrementar el número de clientes potenciales en un área. Los sistemas basados en áreas con radio de 1 Km. podrían tener cien veces más canales que los sistemas basados en radios de 10 Km. La especulación llevó a la conclusión de que reduciendo el radio de las áreas a

unos cientos de metros, se podrían atender millones de llamadas.

El concepto celular emplea niveles variables de baja potencia, que permiten ajustar el tamaño de las celdas de acuerdo a la densidad de usuarios y la demanda del área dada. A medida que crece la población se deben agregar celdas para adecuarse a este crecimiento. Las frecuencias usadas en un grupo de celdas (cluster) podrían ser reusadas en otras celdas. Las conversaciones serían transferidas de celda a celda para mantener un servicio constante mientras el usuario se mueve entre ellas. La figura 1-3 muestra la arquitectura celular de los sistemas de telefonía móvil.

Fig.1-3 Arquitectura Celular de los sistemas de telefonía móvil

La Arquitectura del Sistema Celular

Los aumentos en la demanda y la pobre calidad de los servicios existentes llevaron a los proveedores de servicios móviles a mejorar la calidad de los servicios y aumentar la cantidad de usuarios soportados en sus sistemas. Ya que el espectro de frecuencias disponibles para el uso de móviles celulares era limitado, se necesitaba un uso eficiente de las frecuencias para la cobertura del servicio.

Celdas y Clusters

Una celda es la unidad geográfica básica de un sistema celular. El término *celular* viene de la forma hexagonal en la que se dividen las regiones de cobertura. Esta forma hexagonal es similar a la de un panal de abejas. Las celdas

son estaciones base que trasmiten sobre pequeñas áreas geográficas representadas como hexágonos. El tamaño de las celdas varía dependiendo de la topografía. Debido a las restricciones impuestas por el terreno natural y las estructuras hechas por el hombre, la verdadera forma de las celdas no es un perfecto hexágono.

Un cluster es un grupo de celdas. Los canales no son reusados dentro de un cluster. La figura 1-4 ilustra un cluster de siete celdas.

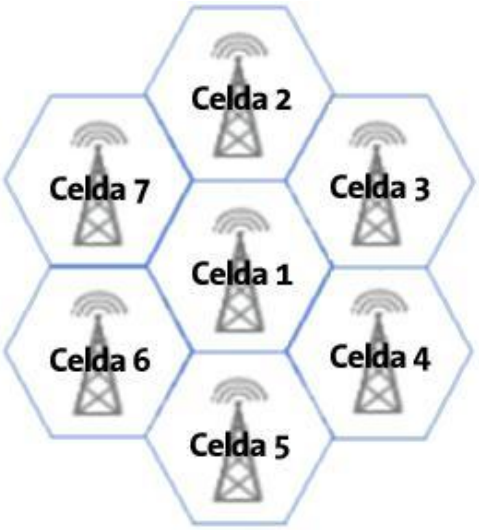


Fig.1-4 Un cluster de siete celdas

El reuso de frecuencias

El concepto de reuso de frecuencias está basado en la asignación de un grupo de canales de radio a cada celda dentro de una pequeña área geográfica. Este grupo de canales asignados a una celda son completamente diferentes de los asignados a celdas vecinas. El área de cobertura de las celdas se denomina *huella* (footprint). Esta huella está limitada por una frontera para que el mismo grupo de canales pueda ser usado en diferentes celdas que están alejadas suficientemente unas de otras, para que sus frecuencias no interfieran. La figura 1-5 muestra este reuso de frecuencias.

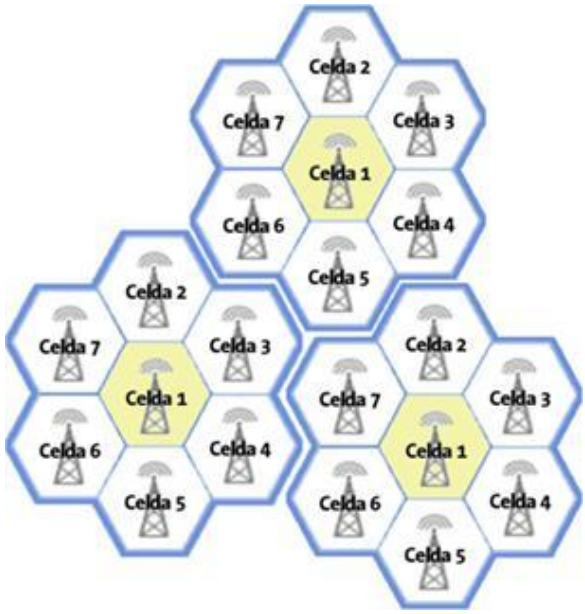


Fig.1-5 El reuso de frecuencias

Las celdas con el mismo número tienen el mismo conjunto de frecuencias. Aquí, ya que el número de frecuencias disponible es 7, el factor de reuso de frecuencias es 1/7. Esto significa que cada celda está usando 1/7 de los canales celulares disponibles.

La división de celdas

Desafortunadamente, las consideraciones económicas hicieron imposible de poner en práctica la construcción de sistemas completos con varias pequeñas áreas. Para sobrellevar esta dificultad, los operadores de sistemas desarrollaron la idea de separación de celdas (cell splitting). Cuando un área de servicio sufre sobrepoblación de usuarios, se usa este concepto de división de celdas para dividir un área en otras más pequeñas. De esta forma, los centros urbanos pueden ser divididos en varias áreas según fuese necesario para proveer niveles de servicio aceptables en regiones con tráfico pesado, mientras que celdas más grandes y menos costosas pueden ser usadas para cubrir regiones rurales remotas.

La figura 1-6 muestra este acercamiento.

Fig.1-6 División celular

Handoff: Traspaso entre celdas

El obstáculo final en el desarrollo de la red celular involucra el problema originado cuando un usuario móvil viaja de una celda a otra durante el curso de una llamada. Como las áreas adyacentes no usan los mismos canales de radio, la llamada debe ser descartada o transferida de un canal de radio a otro cuando el usuario atraviesa la línea entre celdas adyacentes. Como descartar la llamada es inaceptable, se creó el proceso de traspaso. El traspaso (handoff) ocurre cuando la red de telefonía móvil automáticamente transfiere la llamada de un canal de radio a otro, cuando un móvil cruza entre celdas adyacentes. La figura 1-7 muestra un ejemplo de traspaso de una comunicación de un usuario moviéndose entre celdas.

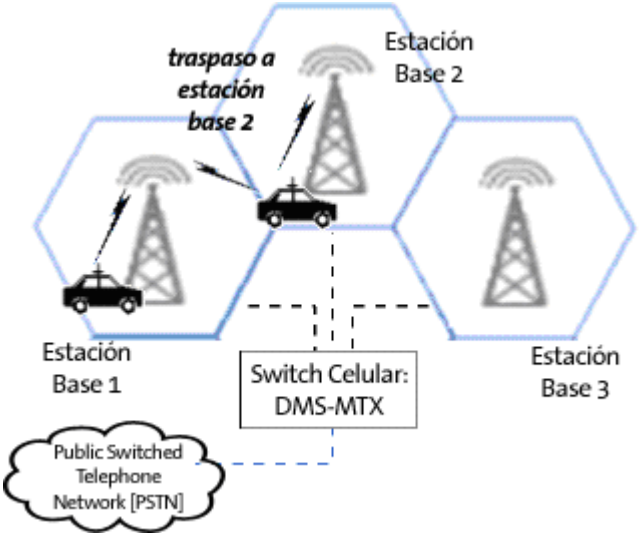


Fig.1-7 Traspaso entre celdas

La Red AMPS

Los diferentes dispositivos móviles operan en diferentes sistemas: analógicos, digitales o ambos. Cuando se trata de una simple llamada, no existen diferencias entre un sistema y otro—al menos superficialmente. Cuando se abandona un área de cobertura digital, el móvil escogerá un sistema analógico u otro sistema digital. Pero el sistema es importante en cuanto a la calidad de la voz y a los servicios de valor agregado extras.

AMPS (Advanced Mobile Phone Service) es un sistema estándar para el servicio celular de señal analógica que opera en frecuencias de entre 800 y 900 MHz.

Sistemas Digitales

La ventaja de la tecnología celular digital sobre la analógica es que las redes incrementan su capacidad y seguridad. Opciones como TDMA y CDMA ofrecen más canales en el mismo ancho de banda de un sistema analógico junto con voz y datos encriptados.

TDMA y CDMA

A mediados de los 90, los servicios digitales pasaron a estar disponibles. TDMA (Time Division Multiple Access) es una tecnología digital que permite dividir cada canal celular en tres ranuras de tiempo, para aumentar la cantidad de datos que pueden ser transportados. La señal digital es enviada en ráfagas de tiempo para que no interfieran con otras señales en la misma frecuencia.

El nuevo estándar CDMA (Code Division Multiple Access) usa un enfoque completamente diferente. Después de digitalizar los datos, CDMA los propaga a lo largo de todo el ancho de banda disponible. Las múltiples llamadas se superponen sobre un mismo canal, cada una con una secuencia de código única. CDMA opera en bandas entre 1850 y 1990 MHz.

El protocolo WAP

El protocolo WAP (Wireless Application Protocol) es una especificación global y abierta que amplía las capacidades

de los dispositivos móviles para acceder fácilmente e interactuar con información y servicios de forma instantánea. Este es uno de los servicios de valor agregado de la red digital.

Motivación

WAP es la convergencia de tres tecnologías de red de evolución rápida: data wireless, telefonía e Internet. En la figura 1-8 se muestra un gráfico de esta evolución.

Ambos mercados, el de wireless data y el de Internet, están creciendo muy rápidamente y continúan para llegar a más clientes. El crecimiento explosivo de la Internet ha potenciado la creación de nuevos y excitantes servicios de información.

La mayoría de las tecnologías originales desarrolladas para la Internet fueron diseñadas para computadoras de escritorio y computadoras conectadas a redes con ancho de banda grandes a medianos, generalmente redes de datos confiables. En el mercado masivo, los dispositivos inalámbricos presentan un ambiente más restringido comparado con las computadoras de escritorio.

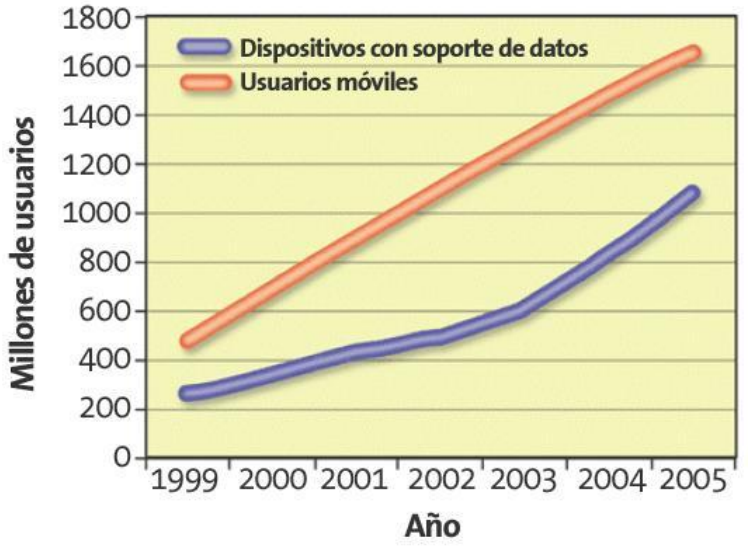


Fig.1-8 Crecimiento de usuarios de dispositivos móviles en contraste con la evolución de los dispositivos con soporte de datos

Usos

El propósito del protocolo WAP es permitir la entrega fácil y rápida de información a usuarios de dispositivos móviles.

Dispositivos con soporte WAP

Básicamente los dispositivos móviles más populares que tienen soporte WAP son los teléfonos celulares de última generación, PDAs (Personal Data Assistants), y HPCs (Handheld PCs).

Sistemas Operativos que soportan WAP

WAP es tanto un protocolo de comunicaciones como un ambiente de aplicación. Puede ser soportado en cualquier sistema operativo como cualquier otro protocolo. Actualmente la mayoría de los sistemas operativos de PC soportan el protocolo y también los dispositivos portátiles han incorporado, fundamentalmente en los últimos años, soporte WAP, incluyendo PalmOS, EPOC, Windows CE, FLEXOS, OS/9, JavaOS, etc. En los teléfonos celulares, por el contrario, son muy pocos los modelos que soportan el protocolo, pero es una tendencia a ser revertida.

Infraestructura de redes que dan soporte WAP

WAP fue diseñado para trabajar bajo la mayoría de las redes inalámbricas hasta hoy concebidas, tales como CDMA, TDMA, GSM y GRPS, entre otras. En Argentina, las operadoras de servicio celular solo operan bajo las normas CDMA (Movicom y CTI) y TDMA (Unifón y Personal).

Redes fijas versus redes inalámbricas

Las aplicaciones para dispositivos inalámbricos tienen premisas muy diferentes en varios aspectos.

Dispositivos. Las necesidades respecto de los dispositivos móviles en contraste con los dispositivos fijos recae en que los primeros tienden a tener menor capacidad de procesamiento (procesadores más pequeños), menor cantidad de memoria RAM/ROM y de almacenamiento secundario, energía de

operación limitada, pantallas pequeñas y de dimensiones no comunes y diferentes tipos de dispositivos de entrada (teclados, rueda de desplazamiento, pantallas sensibles al tacto, etc.).

Redes de soporte. Las redes para dispositivos inalámbricos presentan mayores restricciones que las redes fijas. Tienen menor ancho de banda, mayor tiempo muerto (latency) tanto en la conexión como en la transferencia, menor estabilidad en la conexión, y disponibilidad poco predecible, debido a las debilidades de las redes por aire (puntos ciegos y campos débiles)

Operadores de red. Debido a la gran cantidad de operadores de red, cada uno con su estándar y tecnología, se debe lograr que el protocolo móvil sea interoperable—permitiendo que dispositivos de diferentes proveedores se comuniquen mediante las redes de otros operadores (legacy networks), que sea escalable—las operadoras de servicio deben poder escalar sus servicios a las necesidades de los clientes, que sea eficiente—proveer un servicio de calidad de acuerdo con las posibilidades de las redes inalámbricas, que sea confiable—proveer una plataforma consistente para soportar sus servicios, y por último, que sea seguro—preservando tanto la privacidad como la integridad de los datos de los usuarios, protegiendo a los dispositivos y servicios de problemas de seguridad.

Protocolos actuales. La Internet actual usa una vasta selección de distintos estándares y protocolos para soportar diferentes servicios (HTTP, FTP, etc.) y capas de transporte y de red (TCP, UDP, IP, etc.). Estos protocolos fueron originariamente diseñados para computadoras con amplios recursos incluyendo redes fijas de gran ancho de banda. En ese entonces, la optimización para el ancho de banda no era una de las preocupaciones principales. La seguridad no era un tema importante ya que las redes se poblaban de un límite de hosts locales o remotos pero amigables por lo que la necesidad de seguridad en la información no era requerida en los protocolos.

El problema principal de los protocolos existentes, es que su implementación requiere más recursos de los que disponen los teléfonos celulares actualmente (excepto claro, los de última tecnología). Aunque en el futuro los problemas de energía se solucionen y la carga de las baterías se amplíe, los nuevos procesadores consumirán aún más en relación al avance, con lo que tampoco será suficiente.

Por lo tanto, el protocolo WAP está diseñado para ser consistente: solo las partes esenciales de los demás protocolos es implementada, todo está optimizado y minimizado, y con un grado de seguridad aceptable para el tipo de red inalámbrica.

Puntos de usabilidad

Debido al pequeño tamaño de los dispositivos móviles, la capacidad limitada de imágenes, y las diminutas pantallas, la usabilidad es más importante que nunca.

Un teléfono es un teléfono. Por sobre todo, la gente lo usa para comunicarse. Una PDA también es una PDA, principalmente usada con fines de agenda o cálculo. Aunque hoy en día los dispositivos móviles tienen capacidad de datos, los usuarios ven esta funcionalidad solo como un bonus.

El ancho de banda es mínimo e intermitente.

El ingreso de los datos es siempre difícil y consume tiempo.

Al igual que en un desarrollo Web, en el cual se pretende minimizar la cantidad de clicks para llegar a cierta información, en un desarrollo WAP se pretende minimizar la cantidad de teclas a presionar.

Las pantallas son pequeñas y de tamaño variado, por lo tanto es preferible utilizar la menor cantidad de líneas de texto posible.

La OMA

La OMA (Open Mobile Alliance) es una alianza, como su nombre lo indica, entre empresas proveedoras de equipos, especialmente de telefonía celular móvil, con el fin de crear un estándar y la especificación WAP, así como dar bases a la OMAI (Open Mobile Architecture Initiative), un conjunto de principios que aseguran la interoperabilidad y el beneficio de los usuarios finales en la experiencia de los servicios WAP y con el propósito de “extender las barreras del mercado”- según su lema.

En un principio fue formada por Nokia, Ericsson, y Motorola en conjunto con Unwired Planet, una empresa proveedora de dispositivos handheld.

La especificación de la OMA puede ser consultada a través de su sitio Web, **http://www.openmobilealliance.org/**.

La especificación WAP de la OMA

La especificación WAP propone un conjunto de normas para estandarizar y extender la tecnología y el conocimiento de las empresas, y es la base de conocimiento producto de la fusión de las tareas de investigación y desarrollo de las empresas que forman la OMA.

WAP fue desarrollado para ser el medio estándar para el transporte de información y la puerta al servicio de Internet para dispositivos de comunicación móvil. WAP es una plataforma hardware-software e independiente de la red, por ello se lo considera un protocolo que cumple con el estándar de la Tercera Generación de redes Inalámbricas.

Usa un navegador liviano, llamado micro navegador (o micro browser) como software cliente diseñado para tener requerimientos de hardware muy mínimos para abarcar el vasto espectro de dispositivos existentes.

WAP ha sido diseñado en base a varios estándares actualmente usados como el HTML, HTTP, XML, TCP, IP, UDP, y TLS. Las metas en el desarrollo de este nuevo protocolo fueron hacerlo optimizado para ancho de banda, para permitirle operar tan rápido como fuese posible en redes inalámbricas lentas (comparado con las redes fijas convencionales para las que se diseñaban protocolos) y lo suficientemente flexible como para trabajar en pequeñas pantallas de diferentes formas y tamaños y dispositivos de entrada del usuario poco o nada similares. Los requerimientos de ancho de banda son minimizados transfiriendo toda la información como datos binarios comprimidos y no como texto plano como otros protocolos similares como el HTTP. La información del header y el lenguaje de construcción de páginas WAP fueron optimizados para ser muy pequeños y desarrollar aplicaciones tanto pequeñas como flexibles.

WAP es un estándar en evolución. La versión actual de la especificación es 2.0. Pero la versión que más se utiliza, tal vez por ser la más testeada, es la 1.1.

La versión inicial de la especificación (1.0) fue lanzada en Abril de 1998, y la 1.1, en Mayo de 1999. A mediados del año 2000 aparece la versión 1.2 y a principios del año 2001, la 1.3. A fines de 2001 se lanza la versión 2.0.

Capítulo 2. La Arquitectura WAP

Introducción

El protocolo WAP
Los beneficios del WAP
Similaridad con Internet
Arquitectura
WAP Push Service
Seguridad en WAP

Capítulo 2. La Arquitectura WAP

Introducción

El protocolo WAP (Wireless Application Protocol) incrementa la funcionalidad de dispositivos móviles con servicios interactivos en tiempo real. El protocolo fue diseñado especialmente para las pantallas pequeñas y mínimo ancho de banda, y ofrece una amplia variedad de servicios inalámbricos—como por ejemplo acceso a Internet—independientes del proveedor, prestador de servicios y tecnología de estos dispositivos móviles.

WAP es un puente entre el mundo de los dispositivos móviles y la Internet o intranets corporativas, y ofrece la capacidad de entregar un rango ilimitado de servicios de valor agregado para usuarios de teléfonos celulares, PDAs, etc. independientemente de su red (analógica o digital), banda portadora (CDMA, TDMA, GSM, PCS, etc.), tipo y marca de dispositivo. De esta forma, los usuarios pueden acceder al mismo flujo de información desde un dispositivo de mano como lo hacen desde un dispositivo de escritorio.

El protocolo WAP

El protocolo WAP es un estándar global y no está controlado por una única compañía. Ericsson, Nokia, Motorota, y Unwired Planet fundaron el Foro WAP en 1997 con el propósito inicial de definir una especificación para el desarrollo de aplicaciones sobre las redes de comunicación inalámbricas. Esta especificación cubre un modelo de red por capas y un conjunto de protocolos incluidos en cada una de estas capas.

La figura 2-1 muestra el protocolo WAP y el modelo de capas.

WAE: Wireless Application Environment, ambiente de ejecución para aplicaciones WAP.

WSP: Wireless Session Protocol, protocolo de sesión WAP.

WTP: Wireless Transaction Protocol, protocolo de transacción WAP.

WTLS: Wireless Transport Layer Security, protocolo de seguridad WAP.

WDP: Wireless Datagram Protocol, protocolo de transporte WAP.

BEARER: Portadora de servicio. (CDMA, TDMA, GSM, etc.)

Fig.2-1 El protocolo WAP y el modelo de capas.

Los beneficios del WAP

Para los operadores de red celular, WAP promete disminuir costos e incrementar la cantidad de clientes mejorando los servicios existentes, por ejemplo dotándolo de soporte de mensajes de voz, sistemas prepagos, y facilitando una gama de nuevos servicios de valor agregado y aplicaciones, como manejo de cuentas y consultas sobre facturación. Todas estas mejoras pueden ser introducidas al sistema rápida y fácilmente, sin la necesidad de infraestructura adicional o modificaciones al dispositivo móvil. De esta forma, se permite a los operadores diferenciarse de sus competidores con servicios de información personalizados.

Para los proveedores de contenido, escribir sus aplicaciones es una tarea muy fácil. WAP especifica un formato simple, WML (Wireless Markup Language), que es no más que un subconjunto del XML (eXtensible Markup Language). Esto trae la ventaja de poder usar el mismo modelo que las aplicaciones tradicionales para Internet, e incluso adaptar las ya existentes para ser soportadas tanto en dispositivos de escritorio como portátiles.

Para el usuario final, los beneficios incluyen un acceso fácil y seguro a información y servicios, Internet, mensajería, banking y entretenimiento desde un dispositivo móvil. Además,

la información de intranets—como una base de datos corporativa—podría incluso ser accedida mediante la tecnología WAP.

Debido a que en la actualidad, la gran mayoría de los proveedores de dispositivos portátiles apoyan la iniciativa WAP, los usuarios tienen la gran libertad a la hora de elegir terminales móviles y las aplicaciones que éstos soportan. Además, los usuarios serán capaces de recibir y requerir información de una forma controlada, tanto rápidamente como a bajo costo, que es lo que hace más atractivo a los servicios WAP.

Similaridad con Internet

En orden de respetar el estándar existente para Internet tanto como sea posible, el Foro WAP siguió de manera cercana el modelo de la Web. Esto se muestra en la figura 2-2.

La arquitectura de capas que definió el Foro WAP permite a las aplicaciones usar las características de la pila WAP (conjunto de capas) a través de interfases bien definidas. Este vínculo cercano con la arquitectura de Internet permite además, utilizar el conocimiento preexistente y la experiencia cuando se desarrollan aplicaciones para dispositivos móviles.

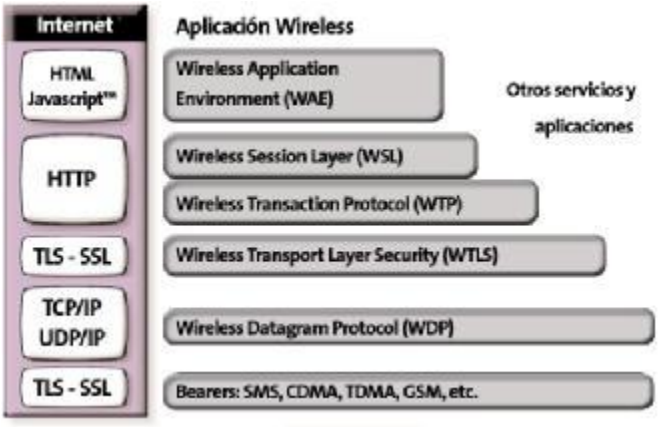


Fig.2-2 Similaridad entre los protocolos de Internet y WAP

Como el protocolo HTTP, el WAP usa el concepto de un browser (o minibrowser), incluido en el mismo dispositivo móvil, adaptado para funcionar en un ambiente limitado tan especial. Este microbrowser WAP coordina la interfaz del usuario y es análogo a los browsers Web estándar.

En ambos casos, se usa el mecanismo clásico de petición-respuesta (request-response), en los que un cliente inicia la

comunicación con una petición a un servidor que acto seguido responde a la petición con el contenido solicitado.

Además de este mecanismo clásico, y a diferencia del HTTP estándar, WAP ofrece otro mecanismo, denominado PUSH, que no necesita que el usuario haga un request al servidor para recibir algún contenido, sino que este contenido es entregado directamente al usuario, basándose en la ocurrencia de algún evento prefijado—a una determinada hora, o con una cierta regularidad, o cuando suceda algún evento disparador, etc. Más adelante, en este capítulo se discute en profundidad la característica Push.

Las figuras 2-3 y 2-4 a continuación, muestran el concepto cliente-servidor en ambos protocolos (HTTP y WAP): la figura 2-3 refleja la transacción request-response clásica del HTTP, mientras que la figura 2-4 muestra a su vez una transacción Push.

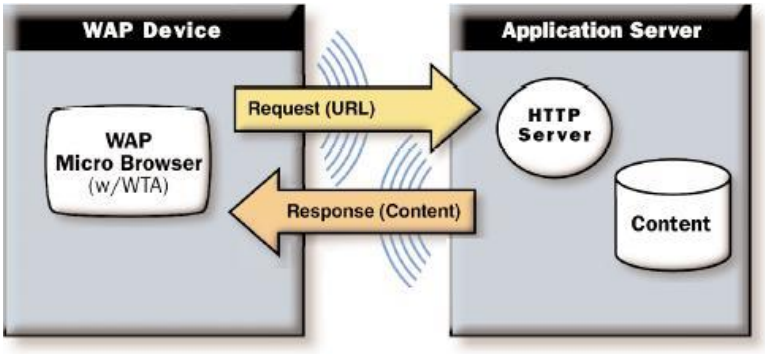


Fig.2-3 Modelo WWW y transacción clásica request-response

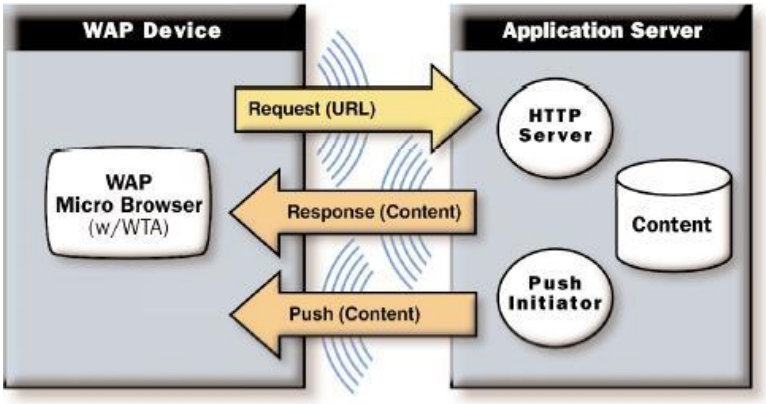


Fig.2-4 Modelo WAP y transacciones request-response y Push.

Cuando un dispositivo móvil hace una petición de una página WML, el pedido es enviado a través de la red hacia un gateway. Este transforma la petición en una petición HTTP estándar, y se la pasa a un servidor de contenido (en la misma forma de una petición HTTP común). Éste entonces, será responsable de chequear la integridad del formato contenido y de comprimir los datos de la respuesta antes de ser devueltos al terminal móvil.

Así, un gateway WAP hace que un dispositivo WAP se parezca a cualquier dispositivo con acceso a Internet. Este mismo mecanismo se aplica al resto de los protocolos estándar.

La figura 2-5 muestra un acercamiento más completo del modelo cliente-servidor y los componentes físicos y lógicos del modelo WAP en la práctica. Este acercamiento, no obstante se trata en profundidad en el capítulo siguiente, en el que se hace mayor énfasis en estas cuestiones.

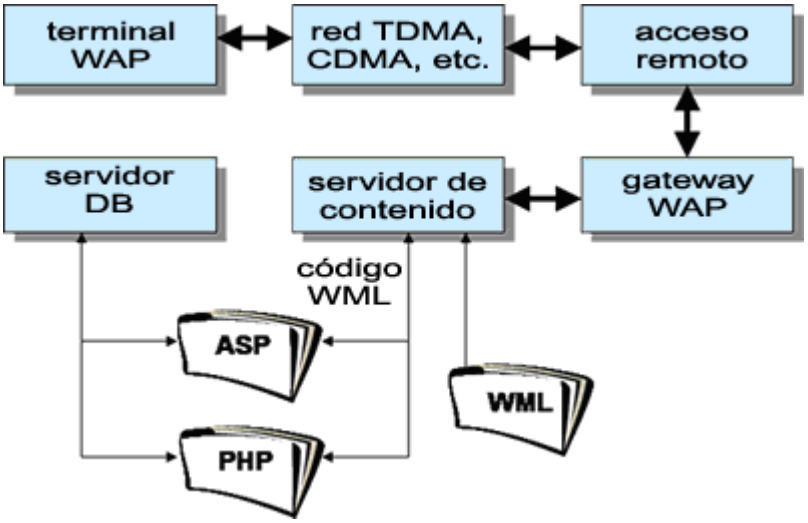


Fig.2-5 Modelo WAP cliente-servidor y componentes físicos y lógicos.

Arquitectura

Como se dijo anteriormente (ver figura 2-1), el protocolo WAP fue diseñado como un conjunto de capas. Ahora veamos en detalle estas capas y sus funciones principales.

WAE (Wireless Application Environment). La capa WAE brinda un ambiente de aplicación de propósito general que usa una combinación de las tecnologías de Internet y de dispositivos móviles. Provee un framework para el desarrollo de aplicaciones en estos dispositivos móviles y está embebido dentro del programa del microbrowser del terminal. Contiene el soporte para las siguientes funciones:

• WML, un lenguaje de presentación liviano, similar al HTML pero optimizado para el uso en los dispositivos móviles.

• WMLScript, un lenguaje de scripts similar a Java Script™.

• WTA (Wireless Telephony Application) y WTAI (Wireless Telephony Application Interface), servicios de telefonía.

• Formatos de Contenido, como vCard y vCalendar.

WSP (Wireless Session Protocol). La capa WSP provee a la capa de aplicación con una interfase para dos servicios de sesión. El primero, orientado a la conexión que opera sobre la capa de transacción. El segundo, sin conexión, que opera sobre un servicio de datagramas seguro o no seguro. Este protocolo está optimizado para bandas portadoras de ancho de banda cortos y latencias[1] grandes.

WTP (Wireless Transport Protocol). La capa WTP opera sobre el servicio de datagrama y provee un protocolo liviano orientado a transacciones, adecuado para el uso en dispositivos móviles. WTP también opera sobre redes seguras y no seguras. Sus funciones principales abarcan la segmentación de paquetes en datagramas y reensamble de datagramas en datos útiles. También se encarga de las retransmisiones y el envío de confirmaciones (ACKs).

WTLS (Wireless Transport Layer Security). La capa WTLS está basada en el estándar de la TLS (Transport Layer Security) de Internet, y está optimizada para el uso en canales de comunicación de bandas muy cercanas entre sí. Se podría usar la WTLS para una comunicación segura entre los terminales, y las aplicaciones pueden ser las encargadas de habilitar o no estas características de seguridad.

[1] Se denomina latencia (o *latency*) a los tiempos muertos del proceso de apertura o cierre de una conexión y al tiempo de propagación de los paquetes en el medio físico.

WDP (Wireless Datagram Protocol). Es responsable del proceso de transporte entre un terminal móvil y el gateway. Es independiente de la velocidad de transferencia de datos, y su función principal es proveer acceso a cualquier red (TDMA, CDMA, GSM, etc.) de forma transparente para a las capas superiores, y hacer que éstas no se preocupen por la capa física.

WAP Push Service

Como se dijo anteriormente en este capítulo, la principal diferencia entre el protocolo HTTP y WAP, o al menos la más sustancial, radica en lo que se conoce como servicio PUSH.

La figura 2-6 muestra las dos alternativas que presenta el protocolo WAP para la entrega de contenidos.

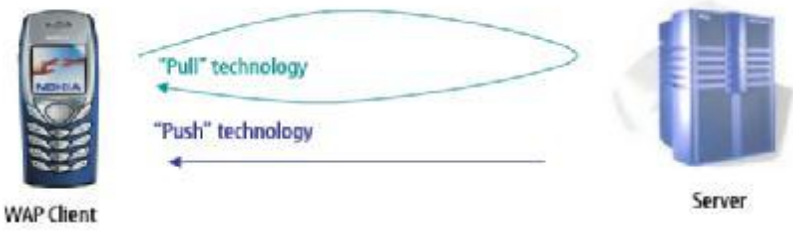


Fig.2-6 WAP PUSH Service

En los modelos de cliente-servidor clásicos, es siempre un cliente quien inicia la comunicación, mediante peticiones al servidor. El servidor, entonces, responde al cliente con el contenido que solicitó. En cambio, a partir del WAP PUSH Service, son los mismos servidores de contenido, quienes envían mensajes al cliente.

Estos mensajes pueden ser:

Indicación de servicio. Más comúnmente se refiere a la indicación de servicio como una "alerta". Presenta al usuario una etiqueta de texto con una URL asociada y la posibilidad de cargar la URL, o dependiendo de la configuración, puede almacenar esta etiqueta en la bandeja de entrada del usuario, para una futura presentación.

Carga de servicio. Es similar a una Indicación de Servicio salvo porque este automáticamente carga el contenido de la URL recibida, no permitiendo al usuario la elección de cargar o no este contenido.

La figura 2-7 muestra los tipos de mensajes Push y la figura 2-8 ilustra con un ejemplo el primer caso.

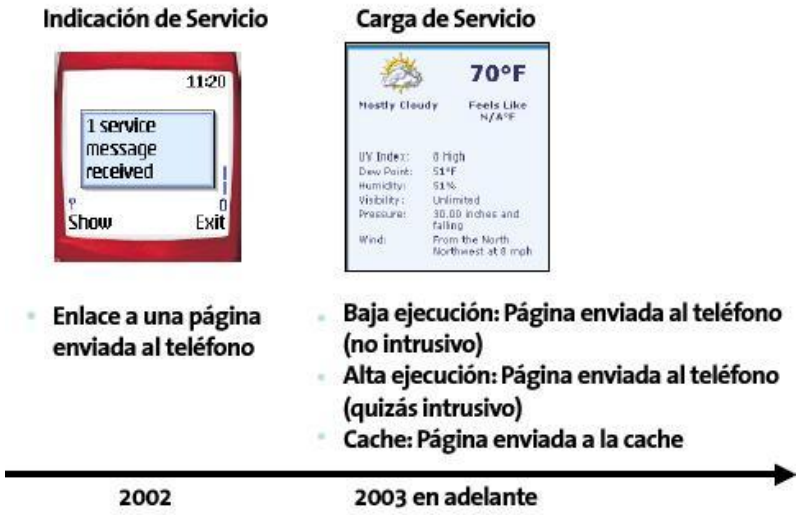


Fig.2-7 Tipos de Mensajes Push

A manera de corolario, podemos suponer un ejemplo como el siguiente, en el que un cliente recibe un mensaje de correo y el sistema se lo informa, mediante una Indicación de Servicio. En la figura 2-8 se muestra el proceso.

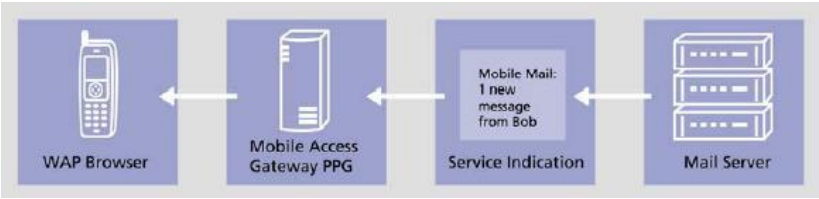


Fig.2-8 Ejemplo de un Servicio Push

Una aplicación sencilla para el ejemplo anterior, podría ser:

```
Content-Type:application/xml;charset=UTF-8
<?xml version="1.0"?>
<!DOCTYPEpapPUBLIC"-//WAPFORUM//DTDPAP1.0//EN"
"http://www.wapforum.org/DTD/pap_1.0.dtd">
<papproduct-
name="PIPushSubmissionServiceIndication)">
```

```
<push-messagepush-id="PushSI"source-
reference="PushTeam">
<addressaddress-
value="WAPPUSH=+18885551111/TYPE=PLMN@ppg.openwav
e.com"/>
</push-message>
</pap>

Content-type:text/vnd.wap.si;charset=UTF-8
<?xmlversion="1.0"?>
<!DOCTYPEsiPUBLIC"-//WAPFORUM//DTDSI1.0//EN"
"http://www.wapforum.org/DTD/si.dtd">
<si>
<indicationhref="http://www.mailservice.com/inbox
?user=jane.doe"si-id=1234>
MobileMail:1newmessagefromBob
</indication>
</si>
```

Cód.2-1 Segmento de código de un mensaje de Indicación de Servicio

Los beneficios de la incorporación del servicio PUSH radican en el valor agregado que representa para el usuario poder recibir el contenido en el que realmente está interesado, mensajes como promociones o descuentos, y hasta la idea de completar verdaderas transacciones.

La arquitectura del servicio PUSH. Una operación PUSH se logra permitiendo a un Iniciador Push (*PI, Push Initiator*) transmitir contenido Push y distribuir instrucciones al Push Proxy Gateway (PPG), el cual distribuye este contenido al cliente WAP. El PI es típicamente una aplicación que se ejecuta en un Servidor Web común. Se comunica con el PPG usando el protocolo PAP (*Push Access Protocol*). El PPG luego usa el protocolo OTA (*Over-The-Air*) para distribuir el contenido Push al cliente.

La figura 2-9 muestra la arquitectura del Servicio Push.

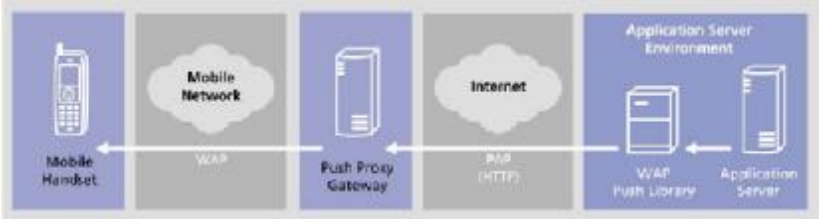


Fig.2-9 Arquitectura del Servicio Push

Seguridad en WAP

Como se mencionó anteriormente, la capa WTLS es utilizada para asegurar privacidad, integridad, y opcionalmente autenticación en los canales de comunicación entre el terminal y el gateway WAP. WTLS es rica en propiedades que hacen posible, por ejemplo:

• Encriptación y Desencriptación transparente de la información enviada entre el cliente WAP y el gateway WAP.

• Autenticación de las partes que intervienen en la comunicación para asegurarle a ambas partes que la otra es quien dice ser.

• Asegurar la integridad de la comunicación verificando que el contenido de los datos no ha sido cambiado desde que se envió. Depende del gateway indicar si la autentificación es necesaria o no.

La forma en la que se establecen las políticas de seguridad, se describe a continuación.

Primero, el terminal envía un mensaje al gateway WAP indicando que debe negociarse la seguridad. Este mensaje contiene una lista de los algoritmos que el terminal soporta. El gateway WAP elige los algoritmos que quiere usar para asegurar la conexión y podría opcionalmente retornar un certificado para que el terminal chequee la identidad de este gateway. Luego el usuario es notificado que la seguridad fue aceptada.

Nótese que no hay interacción entre el usuario y la selección de certificados; tampoco hay certificados en el terminal y la selección del Certificado de Autoridad (*CA, Certificate Authority*) correcto se hace automáticamente.

En el protocolo OTA (*over-the-air*) los desarrolladores pueden especificar si el terminal debería conectarse usando un puerto seguro o no. Sin embargo, el usuario puede editar esta selección. Esta configuración del navegador indica si la capa WTLS deberá ser usada o no. Si un proveedor de servicio requiere seguridad, el gateway WAP deberá denegar todas las conexiones en canales no seguros, o al menos denegar los pedidos de contenido sensible vía canales inseguros.

En WAP 1.x, los diferentes protocolos de seguridad usados entre el terminal y el gateway WAP (WTLS) y los usados entre el gateway WAP y la red (SSL) resultan en un agujero de seguridad en el gateway WAP ya que debido a la necesidad

de conversión entre uno y otro protocolo, los datos primero deben ser desencriptados y luego encriptados nuevamente. Aunque ambos, WTLS y SSL, son seguros, la seguridad punto a punto no puede ser asegurada debido a la situación antes mencionada.

En WAP 2.0, este agujero de seguridad no existe debido a la conformidad entre la especificación WAP y la especificación TCP/IP. Las cuestiones de seguridad están garantizadas por el mismo protocolo TLS (SSL).

En la figura 2-10 se muestran las diferencias en cuestiones de seguridad entre la especificación WAP 1.x y 2.0.

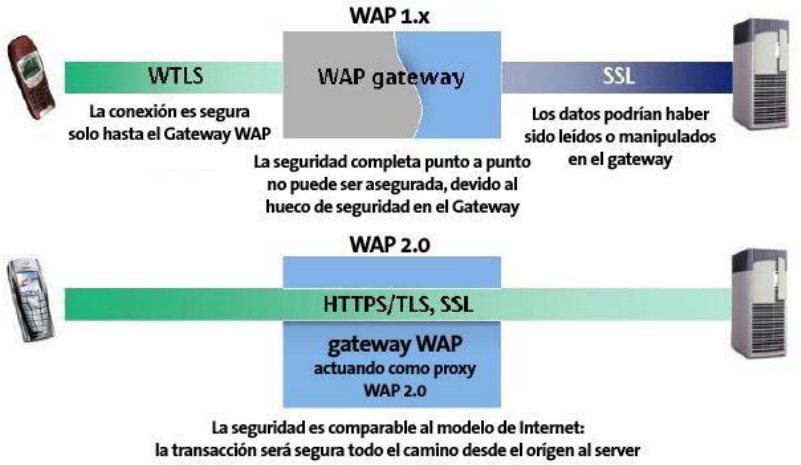


Fig.2-10 Diferencias en cuestiones de seguridad entre la especificación WAP 1.x y 2.0

Capítulo 3. La Tecnología WAP

Introducción a la tecnología WAP

Componentes de la tecnología WAP
Proxies y Gateways WAP
Servidores de Soporte y WTA Servers
Web y WAP Servers
Configurando un servidor para brindar contenido WML: Tipos MIME

Capítulo 3. La Tecnología WAP

Introducción a la tecnología WAP

La tecnología WAP no es muy diferente a la tecnología de la Web. Una aplicación Web se encuentra alojada físicamente en algún servidor Web y siguiendo el enfoque cliente-servidor, será transferida al cliente, según lo requiera, usando un protocolo de transferencia (HTTP) soportado por una arquitectura de red específica (TCP/IP).

De la misma forma, una aplicación WAP se encuentra alojada en algún servidor—con algunas características especiales—y será transferida al cliente usando algún protocolo de transferencia soportado por alguna arquitectura de red específica.

Componentes de la tecnología WAP

Servidor Web. Almacena aplicaciones escritas en WML (Wireless Markup Language[2]), junto con las demás aplicaciones HTML, PHP, JSP, etc.

Filtro HTML. Una aplicación HTML deberá ser convertida a WML antes de ser enviada al usuario. Este filtro podría ser parte del servidor Web o del Proxy WAP.

Proxy WAP. El Proxy WAP actúa como una pasarela (gateway) entre la red celular e Inter/Intranet. Se encarga de codificar de forma binaria la información que se envía al dispositivo móvil y decodificar la que recibe de él.

Servidor WTA. Se encarga de alojar características específicas de la red. Más adelante discutiremos en profundidad el funcionamiento de un servidor WTA.

La figura 3-1 muestra los elementos de una red WAP.

[2] Ver Capítulo 4 WML, El lenguaje de programación WAP.

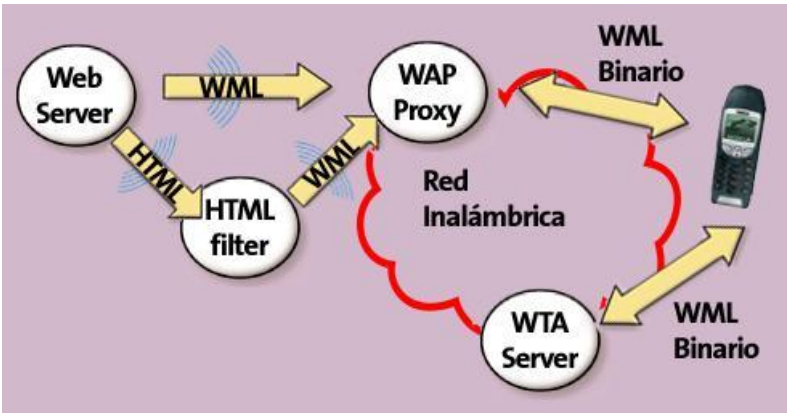


Fig. 3-1 Elementos de Red relacionados con las aplicaciones WAP

Proxies y Gateways WAP

Esencialmente, un gateway WAP sirve de traductor entre una red y otra. La entrega de contenidos WAP usa dos redes: una red inalámbrica de transporte e Internet. Si se dispone de un servidor Web capaz de alojar contenido WAP y éste es accesible en Internet vía TCP/IP, se debe usar un gateway WAP que actúe como agente que permita al WAP y al TCP/IP conversar, y de ese modo permitir a usuarios de dispositivos móviles acceder al contenido WAP.

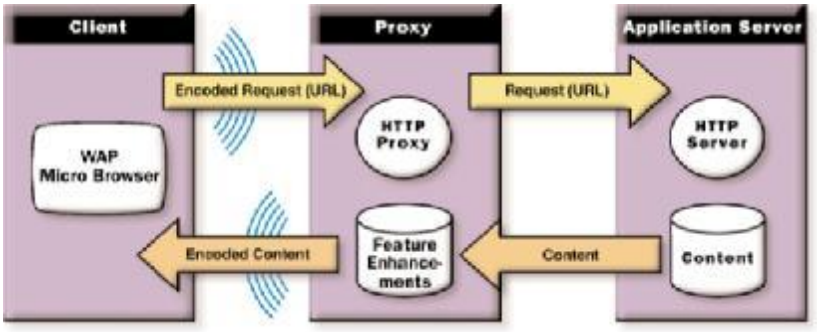


Fig.3-2 Un acercamiento a los Proxies WAP

Un número de gateways WAP están disponibles, incluyendo a los de Nokia y Ericsson, entre otros, además de la posibilidad de mantener un gateway propio vía open source.

Además de permitir a los dispositivos móviles comunicarse con Internet, los gateways WAP ofrecen un conjunto extra de

funciones, que varían dependiendo el tipo de gateway. Por ejemplo, algunos tienen la habilidad de traducir contenido HTML en WML en el acto.

En resumen, un gateway WAP es:

• Gateway de Protocolos. Un gateway de protocolos traduce pedidos de alguna capa de red WAP (por ejemplo para la especificación WAP 1.x –WSP, WTP, WTLS, y WDP) y hacia los protocolos WWW (HTTP y TCP/IP)[3]. El gateway también resuelve nombres (DNS lookups).

• Encoder y Decoder de contenidos. El codificador de contenido puede ser usado para traducir contenido HTML, PHP, etc. en contenido WAP y luego en un formato compacto que permita la mejor utilización del enlace reduciendo su tamaño (WAP binario).

• Agente gerenciador de Perfiles de Usuario (UAProf). Dota a las aplicaciones de perfiles de usuario que describen las capacidades y preferencias del usuario.

• Proxy de Cache. Un proxy cache puede mejorar la performance y la utilización de red manteniendo una cache de recursos frecuentemente accedidos.

Esta infraestructura asegura a los usuarios de terminales móviles el acceso a la gran variedad de contenidos y aplicaciones en Internet, y a los autores de aplicaciones la habilidad de construir servicios de contenido que puedan ejecutarse en la gran gama de dispositivos móviles.

Además, un proxy WAP permite alojar aplicaciones y contenido en servidores Web estándar y desarrollar estas aplicaciones mediante tecnologías muy probadas como CGI, PHP, ASP, JSP, etc.

La figura 3-3 muestra las capas de red que implementa el protocolo WAP, especificadas para todas las entidades participantes en el modelo: un dispositivo WAP, un Gateway Wap y un Servidor Web.

[3] Ver Figura 3-3 Arquitectura en capas de un gateway WAP

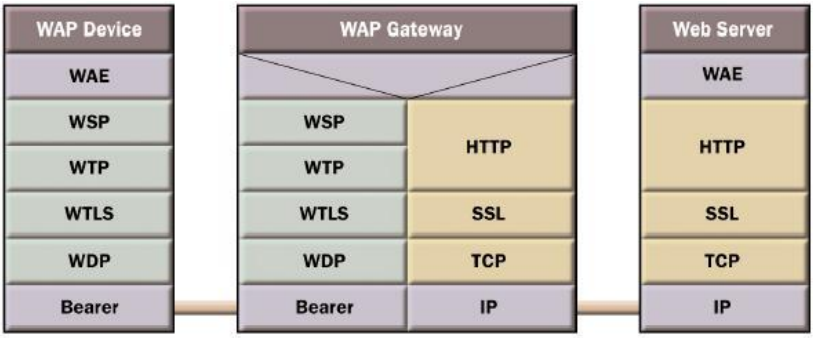


Fig.3-3 Arquitectura en capas de un gateway WAP

WAE: Wireless Application Environment, ambiente de ejecución para aplicaciones WAP.
WSP: Wireless Session Protocol, protocolo de sesión WAP.
WTP: Wireless Transaction Protocol, protocolo de transacción WAP.
WTLS: Wireless Transport Layer Security, protocolo de seguridad WAP.
WDP: Wireless Datagram Protocol, protocolo de transporte WAP.
BEARER: Portadora de servicio. (CDMA, TDMA, GSM, etc.)
HTTP: Hypertext Transfer Protocol, protocolo de transferencia TCP/IP.
SSL: Secure Sockets Layer, protocolo de seguridad conocido como TLS.
TCP: Transport Control Protocol, protocolo de transporte TCP/IP.
IP: Internet Protocol.

Servidores de Soporte y WTA Servers

La arquitectura WAP también incluye servidores de soporte, que proveen servicios a dispositivos, proxies y aplicaciones cuando es necesario. Estos servicios son a menudo específicos en su función, pero de uso general para una gran variedad de aplicaciones.

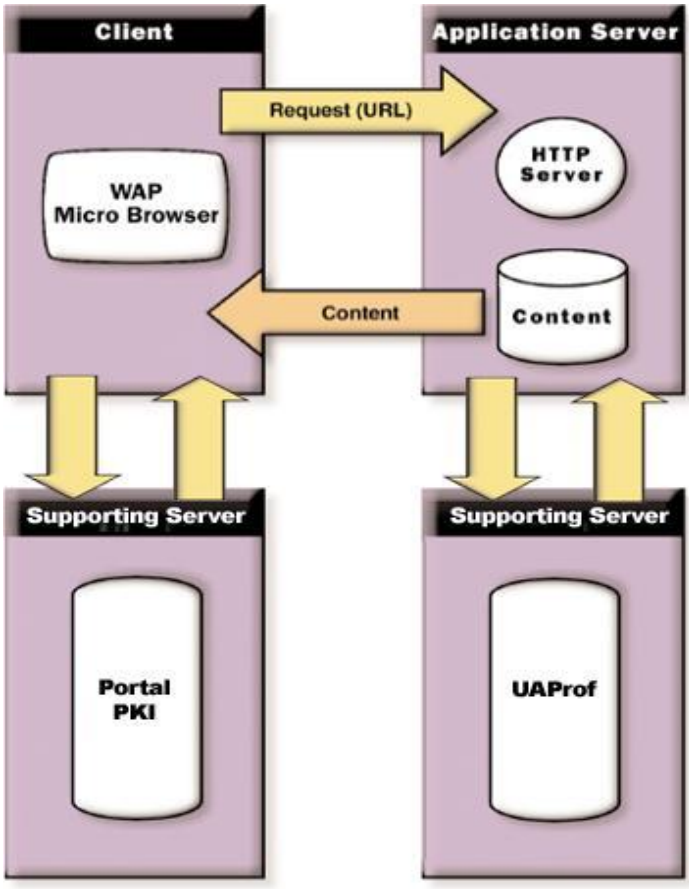


Fig. 3-4 Servers de Soporte

Los servidores de soporte definidos por la especificación WAP incluyen, pero no están limitados a:

• Portal PKI. Un Portal PKI [WPKI] permite a los dispositivos iniciar la creación de nuevos certificados de clave pública.

• Servidor UAProf. Un Servidor UAProf [UAProf] permite a las aplicaciones recuperar las capacidades y perfiles personales de cada usuario.

• Servidor de Aprovisionamiento. La provisión de información es confiada por el dispositivo WAP a un Servidor de Aprovisionamiento [ProvArch].

Web y WAP Servers

Es muy importante que quede claro que, a diferencia de lo que muchos creen, no es necesario disponer de un servidor WAP para ofrecer contenidos WAP: basta con un simple Web Server. Sin embargo, las características avanzadas WAP como autenticación, perfiles, etc. solo están disponibles a través de verdaderos servidores WAP.

Configurando un servidor para brindar contenido WML: Tipos MIME

Para que el servidor de HTTP sepa que debe servir las páginas WML como páginas WAP debemos indicárselo, de lo contrario el servidor responderá como que el tipo de fichero es de texto.

Para decirle al servidor que las páginas de extensión wml son páginas WAP existen los tipos MIME, con estos tipos indicamos al servidor como se debe comunicar con el cliente cuando le solicitan una página wml.

La configuración de tipos MIME, si el servidor ya no está configurado, es la siguiente:

Tipo MIME	Descripción	Extensión
text/vnd.wap.wml	Código WML	.wml
application/vnd.wap.wmlc	WML Compilado	.wmlc
text/vnd.wap.wmls	Código WMLScript	.wmls
application/vnd.wap.wmlsc	WMLScript Compilado	.wmlsc
image/vnd.wap.wbmp	Imagen WBMP	.wbmp

Capítulo 4. WML, El lenguaje de programación WAP

Introducción al WML

El lenguaje WML

Bloques de construcción del WML
Elementos del mazo
Elementos de evento
Elementos de tarea
Variables
Elementos de entrada de usuario
Imágenes y anchors
WBMP, Imágenes en WML
Formato de texto en WML

El DTD

Nota:
En este capítulo se discutirá una primer aproximación de la especificación WAP 1.x a fines de introducir al lector en el tema, y hacer una breve presentación del lenguaje WML.
En el capítulo 7 se avanza sobre la evolución de la especificación en lo que se conoce como WAP 2.0 y el lenguaje de soporte XHTML MP usados actualmente.

Capítulo 4. WML, El lenguaje de programación WAP

Introducción al WML

WML (*Wireless Markup Language*) es un lenguaje de marcas basado en el XML (*Extensible Markup Language*)— esto conlleva a que WML deba cumplir con la sintaxis de XML—, y es usado para crear contenido destinado a pantallas en dispositivos móviles.

El lenguaje WML

WML es una aplicación XML que permite formatear texto para un ambiente WAP. También provee un soporte limitado para imágenes definidas en una pequeña escala de grises.

Un código fuente WML (extensión wml), tiene un tipo MIME `text/vnd.wap.wml`. Un archivo WML compilado o codificado tiene un tipo MIME `application/vnd.wap.wmlc`. Un archivo WML, esté codificado o no, es llamado mazo. Un mazo es la unidad más pequeña del WML que un servidor Web puede enviar a un agente usuario para procesarlo en el dispositivo móvil.

Bloques de construcción del WML

El elemento básico que constituye a un mazo WML, es la carta, que especifica una interacción simple entre el dispositivo y el usuario. Cuando un browser recibe un mazo, típicamente activa la primera carta de ese mazo y a través de esta da la posibilidad de navegar hacia alguna carta particular en el mazo. La figura 4-1 muestra los mazos y las cartas del WML.

Fig.4-1 Mazos y Cartas de WML

WML es un lenguaje case-sensitive, por consiguiente <TABLE>, <Table>, <tABle> y <table> son diferentes unos de otros y con diferentes significados. WML además convierte marcas de nueva línea, retornos de carro, tabulaciones y espacios contiguos en espacios simples.

Conjunto de caracteres

El conjunto de caracteres definido por defecto es el ISO/IEC-10646 que es el mismo que el Unicode 2.0. WAP soporta los siguientes subconjuntos de Unicode:

UTF-8
ISO-8859-1 o ISO Latin-1
UCS-2

Este conjunto de caracteres se define en la etiqueta xml.

```
<?xml version="1.0" encoding="UTF-8"?>
```

Entidades

WML puede contener además, entidades nombradas, que especifican caracteres específicos. Por ejemplo, el ampersand (&) es representado por la entidad &. Todas las entidades comienzan con un ampersand y terminan con un ";".

Etiquetas (Tags)

Un tag es un descriptor de elementos del lenguaje. Describe a un elemento y contiene su nombre y un identificador único. Además, podría incluir atributos describiendo otras propiedades.
Todas las etiquetas en WML se escriben en minúsculas. Hay tres tipos de etiquetas.

<tag> Este tipo de etiqueta comienza un elemento y puede contener atributos.

</tag> Este tipo de etiqueta cierra un elemento.

<tag/> Este tipo de etiqueta indica un elemento vacío, por ejemplo <br/>, indicando un fin de línea.

Elementos

Los elementos especifican toda la información de estructura y de marca de un mazo WML. Los elementos pueden contener un tag de inicio, contenido, otros elementos, y un tag de fin. Los elementos tienen dos estructuras posibles: elementos que encierran contenido (definidos por un tag de inicio <tag> y un tag de fin </tag>) y elementos (<tag/>) que identifica un elemento sin contenido.

```
<tag> contenido </tag> ó <tag/>
```

La siguiente tabla resume los distintos elementos del WML, que se tratarán con mayor profundidad a lo largo de este capítulo.

Categoría	Elementos WML
Mazos y Cartas	wml
	card
	template
	head
	access
	meta
Eventos	do
	ontimer
	onenterforward
	onenterbackward
	onpick
	onevent
	postfield
Tareas	go
	prev
	refresh
	noop
Variables	setvar

Entradas de Usuario	input
	select
	option
	optgroup
	fieldset
Anchors, imágenes y temporizadores	a
	anchor
	img
	timer
Formato de texto	br
	p
	table
	tr
	td
	pre (WAP 1.2 y 1.3)

Los elementos WML, se usan para crear los diferentes componentes de una carta. Se podría tener componentes para crear una conexión entre un documento y otro, o entre una parte del documento y otra. Este tipo de componentes se denomina enlace (*link*). O, se podría tener componentes que simplemente hagan alguna actividad. Este tipo de componentes se denomina tarea (*task*).

Atributos

WML permite incluir atributos dentro de la mayoría de sus elementos. Los atributos especifican información adicional acerca de los elementos. Para definir un atributo a un elemento (y su valor), se debe incluir en el tag del elemento, la forma atributo = "valor".
Cada par atributo-valor se debe separar con espacios en blanco, y los valores deben estar encerrados por comillas dobles (“). Además, los nombres de atributo deben estar en minúscula. Algunos atributos son obligatorios—por ejemplo, el elemento *go* que requiere el atributo *href*.
Siempre se especifican los atributos en el tag inicial del elemento, usando la siguiente sintaxis:

```
<tag atributo1=”valor1” atributo2=”valor2”…>
```

Cometarios

Los comentarios WML siguen el estilo de los comentarios XML y tienen la siguiente sintaxis:

```
<!-- comentario -->
```

Variables

Se pueden setear parámetros para una carta o un mazo WML usando variables. Para sustituir una variable en una carta o un mazo se usa la siguiente sintaxis:

$identificador
$(identificador)
$(identificador:conversión)

Se requieren paréntesis si el final del nombre de la variable no puede ser interpretado naturalmente, por ejemplo con un espacio en blanco al final. Una variable “una variable” contiene un espacio en blanco en su nombre, y este espacio no indica el fin de la variable, por lo que en WML se debería declarar $(una variable).

Estructura del WML

Un mazo válido para WML es un documento XML, por lo que debe contener una declaración XML y un DTD (*Document Type Definition, o definición de tipo de documento*).

```
<?xml version=”1.0”?>
<!DOCTYPE wml PUBLIC “-//WAPFORUM//DTD WML
1.1//EN”
“http://www.wapforum.org/DTD/wml_1.1.xml”>
```

Un mazo WML comienza y termina con el elemento *wml*, mientras que todos los demás elementos se ubican dentro de estas etiquetas de inicio y fin.

```
<wml>
…todos los demás elementos…
</wml>
```

El siguiente es un ejemplo sencillo “Hola mundo” en WML, en el que se ha definido un mazo que contiene una única carta para ilustrar la estructura del lenguaje.

```
<?xml version=”1.0”?>                              (1)
<!DOCTYPE wml PUBLIC “-//DTD WML 1.2//EN”          (2)
“http://www.wapforum.org/DTD/wml12.dtd”>           (3)
<wml>                                              (4)
    <card id=”carta1” title=”Hola mundo!”>         (5)
        <p>                                        (6)
        Hola Mundo!                                (7)
        </p>                                       (8)
    </card>                                        (9)
</wml>                                            (10)
```

Cód.4-1 Hola Mundo!

A continuación se describe el ejemplo anterior línea por línea.

1. Número de versión de XML
2. Identificador del documento público SGML
3. Ubicación del DTD, local o remoto
4. Define el encabezado de un mazo WML
5. Define el inicio de una carta
6. El contenido de una carta se encierra en un bloque ...
7. Contenido de la carta: Hola Mundo!
9. Cierra una carta
10. Es el pié de un mazo. El browser interpreta un bloque contenido entre el encabezado y el pié de un mazo como un mismo mazo WML.

La figura 4-2 muestra el resultado del código anterior.

Fig.4-2 Hola Mundo!

Elementos del mazo

wml

Define un mazo y encierra la información y las cartas.

card

Una carta es la unidad básica de navegación dentro de un mazo. Es posible moverse de una carta a otra dentro del mismo mazo o en otro mazo diferente. A no ser que se especifique lo contrario, cuando se navega a un mazo, se muestra su primera carta. El siguiente es un ejemplo simple de una carta.

```
<wml>
<card id="una_carta" title="titulo">
    <p>
    Texto que contiene <i> información </i>
    </p>
</card>
</wml>
```

template

El elemento plantilla (template) permite especificar que ciertos elementos (indicados abajo) serán ejecutados siempre que se entre en una carta de un mazo. La tarea especificada por la plantilla será ejecutada al menos que sea sobrescrita por otra tarea de la carta.
Los siguientes elementos pueden incluirse en una plantilla: do, onevent, onenterforward, onenterbackward y ontimer.

access

Para poder tomar control de las variables que un mazo usa—asumiendo el potencial riesgo de seguridad que esto implica—se puede especificar en el encabezado del mazo, el dominio (y la ruta) de los otros mazos que tendrán acceso a él. Por ejemplo:

```
<wml>
<head>
  <access domain="midominio.com" path="/pub"/>
</head>
  <card id="carta1">
    <p>Contenido</p>
  </card>
</wml>
```

Este ejemplo indica que tanto midominio.com/pub/stock.cgi como www.midominio.com/pub/index.wml (por dar algunos ejemplos), podrán tener acceso al mazo. En cambio, otrodominio.com/pub/get.php o algunotrodominio.com/cgi-bin/call.cgi no podrán accederlo.
Por defecto, solo se puede acceder un mazo desde el mismo dominio en el que se encuentra alojado el documento WML que contiene al mazo.

No es necesario que se coloquen todas las cartas que colaboran para brindar un servicio, juntas en un mismo mazo. De hecho, es recomendado poner un límite al tamaño del mazo, ya que los diferentes dispositivos tienen diferentes restricciones de memoria. Además, cuando una carta tiene demasiado contenido—y no cabe en pantalla—el usuario debe desplazarla, en el mejor de los casos usando un scroller si el dispositivo posee uno, para poder ver todas las líneas. Por ello es recomendable, dividir la información en varias cartas para que el usuario pueda navegar rápidamente a través de la información.

Elementos de evento

do

Una carta puede contener un elemento do para asignar una tarea a una cierta tecla. El siguiente es un ejemplo de esto.

```
<do type="tipo" label="etiqueta" name="nombre"
optional="false">
```

Donde tipo podría ser *accept*, para navegar a la siguiente carta, y *prev* para volver a la carta anterior. Además se podría dar un nombre a la tecla, usando el atributo etiqueta. Depende de lo que se indique en optional, el navegador podría ignorar el elemento (optional="true") o no.

ontimer

Un elemento *timer* declara la existencia de un temporizador de carta, que se inicia cuando se entra en la carta y se detiene cuando se la abandona. Este timer puede ser usado para procesar inactividad o tiempo

ocioso. El valor del timer se decrementará desde un valor inicial, disparando un evento ontimer cuando el timer llegue a cero. Por ejemplo:

```
<wml>
<card id=”carta1” ontimer=”/next”>
    <timer name=”reloj” value=”50”/>
    <p>Hola mundo!</p>
</card>
</wml>
```

Cada vez que se entre en la carta, el timer se reinicia al valor indicado. Este valor está representado en una décima de segundo por lo que el timer del ejemplo se activa a los 5 segundos.

onpick

Un atributo del elemento <options> es el evento onpick. En una lista de selección múltiple, cuando un usuario selecciona o deselecciona la opción que tiene un atributo onpick, el navegador se redirige a la URL especificada por el atributo onpick, de acuerdo al formato onpick=”/url”.

onevent

onevent liga una tarea a uno de los tres eventos anteriores—ontimer, onenterforward y onenterbackward. Por ejemplo:

```
<card>
 <onevent type=”onenterforward”> <!--1 evento-->
    <go href=”/url”/>            <!--redirige-->
  </onevent>                     <!--a un URL-->
  <p>Hola mundo!</p>
</card>
```

Elementos de tarea

Un elemento *do*, como se vio en la sección anterior, puede especificar tareas que el navegador realizará cuando el usuario presione una tecla o navegue a una carta o mazo. Estas tareas se describen a continuación.

go

El elemento *go* redirige al browser a una ubicación especificada por un *href*. El href puede apuntar a un mazo o una carta WML. El href es el único atributo requerido de este elemento.

```
<go href="URI" sendreferer="true|false"
method="post|get" enctype="tipo_de_contenido"
accept-charset="tipo" id="id1" class="clase">
```

Donde *sendreferer* indica si el mazo que contiene esta tarea debe enviar su URL al href destino o no.

method, especifica el método de submit, equivalentes a los del HTTP. Cuando se indica un método de submit, enctype establece el tipo de codificación del contenido: application/x-www-form-urlencoded (por defecto, o si se usa get) o multipart/form-data (solo para WAP 1.2).

accept-charset especifica la lista de caracteres de codificación de contenido que el webserver debe aceptar cuando procesa una entrada.

id provee un elemento de un nombre único dentro de un mismo mazo.

Por último, *class* afilia un elemento a una o más clases.

prev

Un elemento *prev* le indica al browser navegar hacia el URl previo en el historial. Por ejemplo:

```
<anchor>
  Atrás
  <prev/>
<anchor>
```

Al igual que *go*, pueden especificarse los atributos *id* y *class*.

refresh

El elemento *refresh* indica una tarea de refresco y actualización del contexto del navegador, incluida la interfaz de usuario.

noop

El elemento *noop (no operation)* especifica que no se debe realizar ninguna tarea. Este elemento es de especial utilidad para anular la acción de un template.

Variables

Es posible establecer parámetros para un contenido WML. Esto brinda flexibilidad en la creación de cartas dinámicamente, cambiando el contenido mostrado y navegando en base a entradas del usuario. Las variables WML pueden ser usadas y sustituidas en tiempo de ejecución, a tal fin.

Un nombre de variable consiste de un signo pesos "$" o un guión bajo "_", seguido de cero o más letras, dígitos o guiones bajos. Cualquier otro caracter es ilegal. Además, se debe tener en cuenta que WML es case-sensitive, por lo que las variables también lo serán; de este modo, $variable, $VARIABLE y $varIABLE son diferentes variables.

Siempre que una variable cuyo fin de nombre no pueda ser inferido desde su contexto circundante, por ejemplo si la variable termina con el carácter ilegal de espacio, se debe encerrar entre paréntesis, $(variable).

setvar

Cuando un usuario ingresa un dato en un elemento de entrada o selecciona una opción de una lista, se setea el valor de una variable. En el primer caso, el valor de la variable que representa el campo de entrada (especificada por el atributo *name*) se llenará con lo que sea que el usuario haya ingresado; en el segundo caso, el valor especificado en el atributo *value* de la lista, tomará el valor de la opción elegida.

Pero a veces, es necesario establecer el valor de una variable manualmente, y bajo alguna condición, evento o acción del usuario.

El elemento *setvar* especifica una variable y el valor a asignarle, por ejemplo:

```
<setvar name="nombre_variable" value="valor"/>
```

Donde *nombre_variable* es el nombre de la variable sin el signo "$".

valor podría ser directamente el valor a asignarle a la variable, o el valor de otra variable, en tal caso:

```
<setvar name="var1" value="$(var2)"/>
```

El elemento *setvar* puede ser especificado dentro de los elementos de tarea *go*, *prev* y *refresh*, para producir la asignación de una variable al mismo tiempo en que se inicia la tarea.

Elementos de entrada de usuario

Los elementos de entrada de usuario proveen de un mecanismo para capturar las entradas del usuario. A continuación se detallan los elementos de entrada del WML.

input

El elemento *input* especifica un objeto para la entrada de texto. Se puede determinar el formato de la cadena ingresada por el usuario e incluso escribir el valor por defecto del campo de entrada.

```
<card>
  Fecha: <input name="dia" default="dd-mm-aa"
format="máscara" emptyok="true|false"
size="tamaño" maxlength="largo_maximo">
</card>
```

El atributo *name* establece la variable que contendrá el resultado de la entrada del usuario.
El valor de *default* indica el valor por defecto que mostrará el campo apenas se muestre al usuario.
format especifica una máscara de formato para las entradas del usuario. Por defecto, se permite al usuario ingresar cualquier caracter. (Más adelante en este capítulo, se aborda este tema con mayor detalle.)
Si *emptyok* se declara true, la entrada del usuario no es requerida. Si es falso (valor por defecto), el usuario debe ingresar texto.
size especifica el ancho en caracteres del área del campo de texto.
maxlength especifica el máximo número de caracteres que el usuario puede ingresar en el campo de texto. Por defecto es ilimitado.

Otros atributos del elemento input son:

tabindex="num"	Especifica el orden relativo en la que los elementos de una misma carta son recorridos cuando se pulsa la tecla de tabulado.
value="valor"	Especifica el valor de la variable especificada por el atributo name.
type="tipo"	Se permite dos tipos para el campo de entrada: *text* y *password*. En el último caso, los caracteres ingresados son mostrados ocultos tras algún carácter "oscuro".
title="titulo"	Especifica el título para el campo de entrada. Este título podría ser usado en cuestiones de presentación del objeto.
accesskey="tecla"	Asigna una tecla de acceso rápido al campo de entrada, para permitir al usuario activar un elemento particular usando una simple tecla.

Al igual que los demás elementos, pueden especificarse los atributos *id* y *class*.

El siguiente ejemplo muestra el uso de los campos de entrada de texto para capturar la entrada del usuario. La figura 4-3 ilustra el ejemplo.

```
<card id="carta1" title="Carta 1">
 <do type="accept">
    <go href="#carta2"/>
 </do>
 <p>Escriba su nombre: <input name="nombre"/></p>
</card>
```

Cód.4-2 Campos de entrada de texto

Fig.4-3 Uso de los campos de entrada de texto

Formato en las entradas del usuario

Carácter	Explicación
A	Permite el ingreso de cualquier caracter alfabético en mayúscula o un signo de puntuación.
a	Permite el ingreso de cualquier caracter alfabético en minúscula o un signo de puntuación.
N	Permite el ingreso de cualquier caracter numérico.
X	Permite el ingreso de cualquier caracter en mayúscula.
x	Permite el ingreso de cualquier caracter en minúscula.
M	Permite el ingreso de cualquier carácter (valor por defecto). El navegador asume que el carácter está en mayúscula.
m	Permite el ingreso de cualquier caracter. El navegador asume que el carácter está en mayúscula.
*f	Permite el ingreso de cualquier cantidad de caracteres de formato f, donde f es uno de los formatos anteriores.
nf	Permite el ingreso de n caracteres de formato f, donde n es un número del 1 a 9 y f es uno de los formatos anteriores.
\c	Muestra un caracter c en el campo de texto, donde c puede ser cualquiera, incluso un código de escape o un valor entre comillas.

select

Una lista de selección especifica una lista de opciones de entre las que el usuario puede elegir. WML soporta listas de selecciones simples y múltiples.

```
<select name="opc" default="valor_por_defecto"
iname="índice" ivalue="indice1;indice2;…"
multiple="true|false">
```

El atributo *name* especifica el nombre de la variable que recibe el valor de el o los ítems elegidos.

El atributo *default* indica un valor por defecto a la variable definida por el atributo name.

El atributo *title* le da un título a la lista de selección y puede ser usado para cuestiones de presentación del objeto.

El atributo *multiple* indica, si está en true, que el objeto es una lista de selección múltiple o de selección simple si está en false.

Además de estos atributos, se pueden especificar los siguientes:

iname="variable" El índice del resultado de una selección de usuario se almacena en la variable especificada por *iname*. Un índice 1 indica que se ha seleccionado el primer ítem, un índice 2, que se ha seleccionado el segundo y así sucesivamente. Un índice 0 indica que no se ha seleccionado ninguna opción.

ivalue="índices" Se puede presentar al usuario un conjunto de ítems preseleccionados. Los ítems en este conjunto deben ir separados por un punto y coma ";".

tabindex="num" Especifica el orden relativo en la que los elementos de una misma carta son recorridos cuando se pulsa la tecla de tabulado.

Al igual que los demás elementos, pueden especificarse los atributos *id* y *class*.

El siguiente ejemplo muestra una aplicación WML usando un formulario con dos campos de entrada, nombre y apellido, y que resuelve el inconveniente de la falta de un botón *Submit* mediante el uso de *anchors* que usan elementos *postfield*.

```
<?xml version="1.0"?>
<!DOCTYPE wml SYSTEM "wml_1.3.xml">
<wml>
<card id="1" title="Bienvenido">
      <onevent type="onenterforward">
     <refresh>
        <setvar name="apellido" value=""/>
        <setvar name="nombre" value=""/>
     </refresh>
```

```
 </onevent>
 <p align="center">
    <img src="images/mundo.wbmp"/>
        <a href="#2">Consultas</a>
     </p>
</card>
<card id="2" title="Buscar">
 <p>
     <do type="prev" label="Atras"><prev/></do>
     Apellido: <input type="text"
name="apellido" value="" size="20"/>
     Nombre: <input type="text" name="nombre"
value="" size="20"/>
     <anchor title="registrarse">Buscar
   <go href="abonados.php">
        <postfield name="apeAbonado"
value="$(apellido)"/>
        <postfield name="nomAbonado"
value="$(nombre)"/>
       </go>
     </anchor>
 </p>
</card>
</wml>
```

Cód.4-3 Formulario WML con campos postfield

Imágenes y anchors

anchor

Los elementos *anchor* y *a* especifican el encabezado de un enlace. El pie de un enlace, se especifica como parte de otros elementos, como por ejemplo, el atributo *name* de una carta.

Un enlace anchor debe tener asociado una tarea que especifique el comportamiento que debe exhibir al ser activado. Se podría asociar una tarea *go*, *prev* o *refresh* a un enlace anchor.

No está permitido en WML anidar anchors o especificar más de una tarea dentro de un enlace anchor.

```
<anchor>haga clic aquí
  <go href=”mipagina.com”/>
</anchor>
```

El elemento *a* es una forma corta del elemento *anchor*. El siguiente ejemplo es semánticamente idéntico al anterior:

```
<a href="mipagina.com">haga clic aquí</a>
```

Los atributos que admite anchor son:

title="título"	Especifica el título del anchor.
accesskey="tecla"	Asigna una tecla de acceso rápido al enlace para activarlo usando una simple tecla.

El siguiente es un ejemplo del uso de un enlace anchor. La figura 4-4 ilustra al ejemplo.

```
<wml><card id="carta1" title="enlaces">
  <p>Este es un texto normal, pero este es un
    <anchor title="ENLACE">enlace!
      <go href="mipagina.com"/>
    </anchor>
  </p>
</card></wml>
```

Cód.4-4 Enlaces

Fig.4-4 Uso de anchor

img

El elemento *img* indica que se incluirá una imagen dentro del flujo de texto.

El elemento *img* puede tener los siguientes atributos:

alt	Especifica el texto alternativo a mostrar, en caso de que la imagen no pueda ser mostrada por el navegador
src	Especifica el URI de la imagen a mostrar

localsrc="filen"	Especifica una representación interna alternativa de la imagen. Si esta existe, es usada en lugar de la especificada por src
hspace="length" vspace="length"	Especifica la cantidad de espacio en blanco que se insertará a los lados de la imagen.
align="..."	Especifica la alineación de la imagen con respecto a la línea de texto que la precede. (top | middle | bottom)
height="length" width="length"	Especifica el tamaño de la imagen.

Al igual que los demás elementos, pueden especificarse los atributos id y class.

WBMP, Imágenes en WML

WBMP representa el formato Wireless Bitmap. Este es un formato de imágenes por defecto para WAP. La versión actual de WBMP es llamada *type 0*. Un WBMP es un bitmap monocromático (blanco y negro) no comprimido, pensado para el uso en dispositivos con pantallas pequeñas y conexión de ancho de banda pequeño.

Cuando consideramos el uso de un WBMP en un mazo de WML, se debe pensar en el tamaño de la pantalla y capacidad de memoria de los dispositivos. Como regla, un WBMP no debería ser más ancho que 96 píxeles y más alto que 48 píxeles (a 72 dpi).

Un ejemplo del uso de imágenes WBMP es el siguiente:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">
<wml>
   <card id="carta1" title="Carta1">
     <p>
     <img src="snail3.wbmp" alt="caracol"
vspace="2"
          hspace="3" height="70" width="78"/>
     </p>
   </card>
```

```
</wml>
```

Cód.4-5 Imágenes WBMP

Fig.4-5 Imágenes WBMP

Formato de texto en WML

Las siguientes secciones describen el uso de elementos WML para darle formato al texto que es mostrado en un navegador WAP.

Énfasis

Las etiquetas de énfasis especifican información de marcas de énfasis de texto. Estas etiquetas pueden ser:

	Texto enfatizado
	Texto fuerte
	Texto en cursiva
	Texto en negrita
<u></u>	Texto subrayado
<big></big>	Texto grande
	Texto pequeño

También se admite la anidación entre estos elementos para producir efectos combinados en el formato del texto.

El siguiente es un ejemplo (ver figura 4-6) de cómo se usan y cuál es el efecto que causan, las etiquetas de énfasis.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//DTD WML 1.2//EN"
    "http://www.wapforum.org/DTD/wml12.dtd">
```

```
<wml>
<card id="carta1" title="Enfasis">
  <p align="left">
    <em>Enfatizado</em><br/>
    <strong>Fuerte</strong><br/>
    <i>Cursiva</i><br/>
    <b>Negrita</b><br/>
    <u>Subrayado</u><br/>
    <big>Grande</big><br/>
    <small>Pequeño</small>
    <br/>
  </p>
</card>
</wml>
```

Cód.4-6 Formato de texto: énfasis

Fig.4-6 Énfasis

Alineación y ajuste de texto

Para iniciar una nueva línea en la pantalla del dispositivo, simplemente basta con insertar una etiqueta <br/> donde se quiera iniciar la nueva línea.
En cambio, agregar saltos de línea o retornos de carro no producen ningún efecto: el compilador de WML los trata como un simple caracter de espacio.
El elemento <p> establece tanto los parámetros de alineamiento como los de ajuste de texto en un párrafo. WML tiene dos modos de ajuste de línea: *breaking* (o *wrapping*) y *non-breaking* (o *non-wrapping*). Este modo de ajuste controla como son tratadas las líneas muy largas para encajar en la pantalla.

• Si se especifica el modo “nowrap”, el dispositivo deberá proveer de algún mecanismo para mostrar líneas largas, por ejemplo, un scroller horizontal.

• Si se especifica el modo “wrap", la línea es separada por palabras en múltiples líneas. En este caso, cualquier

espacio entre palabras puede comportarse como un corte de línea.
Para evitar este comportamiento, se pueden reemplazar los espacios por el caracter de escape   (*non-breaking-space*) para indicar al navegador que ese espacio no debe ser tratado como tal.
Otra alternativa, es usar el caracter de escape (*soft-hypen*, o guión suave) para indicar que ese espacio podría ser usado como un corte de línea. En tal caso, el navegador reemplazará el espacio con un guión real.

Para el resto de los casos, el será ignorado.

Tablas

En WML también se pueden hacer tablas, aunque estas están mucho más limitadas que en HTML. En WML no se pueden anidar tablas y se debe siempre especificar el número de columnas en la etiqueta <table>.

Los atributos más comunes de una tabla son:

columns="n"	Número de columnas que tendrá la tabla. Es obligatorio especificar este valor.
title="titulo"	Titulo de la tabla.
align="L|C|R"	Especifica la alineación de las columnas. C indica centrado, L alineado a la izquierda y R alineado a la derecha. Se escribe una letra por cada columna, de manera que align="CCR", indica que la primera y segunda columnas van centradas y la tercera alineada a la derecha.

Por cada columna de una tabla se incluye una par de etiquetas <tr></tr> y por cada celda un par de etiquetas <td></td> tal y como se hace en HTML, salvo que en WML las etiquetas <tr> y <td> solo pueden llevar atributos genéricos.

Un ejemplo del uso de tablas se muestra a continuación:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd">
```

```
<wml>
<card id="t1" title="Tabla">
  <table columns="3">
  <tr>
    <td>D1</td>
    <td>D2</td>
    <td>Otro</td>
  </tr>
  <tr>
    <td>4</td>
    <td>5</td>
    <td>6</td>
  </tr>
  </table>
</card>
</wml>
```

Cód.4-7 Tablas

Fig.4-7 Tablas

Caracteres especiales

WML soporta entidades tanto nombradas como numeradas. Estas últimas, son referencias Unicode, y no al *charset* (página de códigos) actual del documento.

Esto significa que " siempre refiere al mismo caracter lógico, en este caso, el "?".

WML soporta entidades:

- nombradas. Por ejemplo, & o <
- numeradas (decimal). Por ejemplo, +
- numeradas (hexadecimal). Por ejemplo, &#x12;

Entidad	Notación	Explicación
"	"	signo de pregunta

		(?)
&	&	ampersand (&)
<	<	menor que (<)
>	>	mayor que (>)
		non-breaking space
		soft-hypen

El DTD

El *document type definition*, o DTD, establece las reglas que definen qué elementos pueden ser anidados dentro de otros. Un DTD, además define:

• Los nombres y contenidos de todos los elementos permitidos en un cierto documento.

• Cuántas veces puede aparecer un elemento.

• Si el comienzo o el final de una etiqueta puede ser omitido.

• El contenido de todos los elementos, esto es, los nombres de identificadores genéricos que se permiten aparecer dentro de ellos.

• Los atributos y sus valores por defecto.

• Los nombres de las referencias simbólicas que podrían ser usadas.

De acuerdo con la W3C, el DTD identifica qué significa cada etiqueta y especifica como debe ser tratada. En resumen, provee reglas para validar documentos WML, lo que asegura que un documento podrá ser comprendido por cualquier dispositivo que sea compatible con la especificación estándar del DTD.

Para cada diferente versión del lenguaje WML, existe un DTD. Como se dijo anteriormente, se debe declarar la ruta y la versión del DTD a usar en un archivo fuente WML.

La declaración DOCTYPE.

En el comienzo de cada documento, se debe especificar el nombre del DTD con el cual el documento está en conformidad. Esto es porque los intérpretes (*parsers*) SGML pueden determinar el DTD y asegurar que el documento está conforme a éste.

Esta información es generalmente expresada en una línea, en la declaración DOCTYPE.

Una declaración típica para un documento escrito conforme a la versión 1.2 del DTD WML es como sigue:

```
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML
1.2//EN" "http://www.wapforum.org/DTD/wml12.dtd">
```

Esta línea contiene un número de diferentes componentes.

<! Indica que ésta es una declaración SGML. Esta línea está declarando el tipo de documento.

DOCTYPE Muestra una declaración SGML para el tipo de documento.
wml Es el nombre del primer elemento que aparecerá en el documento.

PUBLIC "-//WAPFORUM//DTD WML 1.2//EN" Lista el Formal Public Identifier (*FPI*) para el DTD con el cual el documento está conforme. El intérprete SGML usará esto para encontrar el DTD correcto cuando procese el documento.
PUBLIC no es una parte del FPI, pero indica al procesador SGML como encontrar el FPI referenciado. Otras maneras de decirle al interprete SGML como encontrar el DTD son mostradas debajo.

> Retorna al documento.

Formal Public Identifiers (FPIs).

Establecer un FPI es útil y podría ayudar al interprete SGML cuando este no pueda ubicar el DTD específico que se está usando.
El FPI debe seguir una sintaxis específica. Esta sintaxis es como sigue:

```
"Propietario//Identificador
Descripción//Lenguaje"
```

Propietario Indica el propietario del FPI.

Si la cadena comienza con "ISO" entonces ISO es el propietario del FPI. Por ejemplo, el FPI "ISO 8879:1986//ENTITIES Greek Symbols//EN" indica que ISO 8879:1986 es el propietario de entidades para los símbolos griegos. ISO 8879:1986 es el número ISO para el estándar SGML.

De todas formas, esta cadena se asemejará a esto:

```
-//Propietario/… ó +//Propietario/…
```

(Note que la diferencia se remite al + o - que antecede al FPI).

Si la cadena empieza con -, el propietario no está registrado, con un + se identifica un propietario registrado.

Identificador Existen una serie de identificadores que indican el tipo de información en el archivo. Algunos de los más comunes son DTD, ELEMENT, ENTITIES, y TEXT. DTD es usado solo para archivos DTD, ELEMENT es usualmente usado para fragmentos de DTD que contienen solo declaraciones de entidades o elementos. TEXT es usado para contenido SGML (texto y etiquetas).

Descripción Cualquier descripción que se quiera proporcionar para el contenido del archivo. Esto podría incluir el número de la versión o cualquier texto corto que sea significativo.

Lenguaje Es un código ISO de dos caracteres que identifica el lenguaje nativo del archivo. EN es usado para Inglés.

Archivos de catálogo.

Si se usa la sintaxis anterior, y se procesa el documento usando un procesador SGML, el procesador necesitará tener alguna forma de transformar el FPI en el nombre del archivo que contiene el DTD. Para lograr esto se puede usar un archivo de catálogo. Un archivo de catálogo (típicamente llamado *catálogo*) contiene líneas que

mapean FPIs en nombres de archivos. Por ejemplo, si el catálogo contiene la línea:

```
PUBLIC "-//WAPFORUM//DTD WML 1.2//EN"
"http://www.wapforum.org/DTD/wml12.dtd"
```

el procesador SGML sabrá que tendrá que buscar el wml12.dtd en el subdirectorio DTD del sitio www.wapforum.org en Internet (lo mismo hubiese sido si el DTD estuviera en un directorio local).

Alternativas al FPI.

En lugar de usar FPIs para indicar el DTD con el que el documento está en conformidad (y de ahí, qué archivo contiene el DTD), se puede especificar explícitamente el nombre del archivo. La sintaxis para esto es ligeramente diferente:

```
<!DOCTYPE wml SYSTEM "ruta/al/archivo.dtd">
```

La palabra clave SYSTEM indica que el procesador SGML debería localizar el DTD en una forma específica. Esto típicamente (pero no siempre) significa que el DTD será provisto como un nombre de archivo.
Es preferido usar FPIs por cuestiones de portabilidad: no es deseable adjuntar una copia del DTD con los documentos, y si se usó el identificador SYSTEM entonces se deberían mantener estos DTDs en el mismo lugar.

En el Apéndice C, se incluyen la mayoría de los DTDs usados actualmente y algunas versiones previas.

Capítulo 5. WMLScript, el lenguaje de scripts de WML

Agregando lógica del lado del cliente

Sintaxis del lenguaje WMLScript

Las Librerías Estándar de WMLScript

Un ejemplo de WMLScript

Capítulo 5. WMLScript, el lenguaje de scripts de WML

Agregando lógica del lado del cliente

Así como el lenguaje HTML utiliza el soporte de scripts que brinda JavaScript™, WML también lo hace con el WMLScript (*Wireless Markup Language Script*).

WMLScript es similar a JavaScript. Ambos son lenguajes de scripting que permiten incorporar lógica del lado del cliente.

Mientras que JavaScript coopera con los navegadores Web y es embebido en documentos HTML, WMLScript lo hace con los mini browsers y es embebido en los documentos WML. WMLScript es la solución WAP para proveer lógica del lado del cliente combinado con WML. Ambos, WML y WMLScript, son compilados por un mismo gateway WAP.

WMLScript es parte de la capa de aplicación WAP, y se usa, como se dijo anteriormente, para agregar lógica procedural del lado del cliente a los mazos y cartas del WML.

Este lenguaje está basado en ECMAScript—al igual que JavaScript—pero fue modificado para un mejor soporte de los dispositivos de menor ancho de banda y memoria, como los teléfonos celulares.

Además, WMLScript puede ser usado como una herramienta *standalone*.

Con WMLScript se pueden suplementar las funciones de WML en cuanto a:

• Validar las entradas del usuario.

• Acceder a las prestaciones del terminal. Por ejemplo, en un teléfono celular, permitiendo al programador la incorporación de realización de llamadas, envío de mensajes, agregar números de teléfono a la agenda, etc.

• Generar mensajes y diálogos localmente, y así permitir alertas, mensajes de error, confirmaciones, etc. y mostrarlas al usuario.

• Construir extensiones al software del dispositivo y configurarlo una vez que éstas se instalen.

Sintaxis del lenguaje WMLScript

La sintaxis WMLScript está basada en el estándar ECMAScript. Sin embargo, a diferencia de ECMAScript, la especificación de WMLScript define las características del bytecode y las referencias a la arquitectura del intérprete para la utilización óptima de las comunicaciones de menor ancho de banda y en dispositivos portátiles de poca memoria.

Como sucede con el WML y el HTML, el WMLScript y el JavaScript son muy similares, pero, sin embargo, presentan ciertas características de importancia que hacen a la necesidad de algunas consideraciones:

• En WMLScript toda sentencia debe terminar en un punto y coma (;).

• WMLScript es case-sensitive.

• Los comentarios pueden incluir una sola línea (comenzando con //) o múltiples líneas (encerrado entre /* y */).

• Una cadena de caracteres literales es definida como una secuencia de cero o más caracteres encerrados entre comillas dobles (“”) o simples (‘ ‘).

• Los valores booleanos corresponden a true y false.

• Las nuevas variables son declaradas usando la palabra clave var (var x;)

Tipos de dato

WMLScript es un lenguaje débilmente tipado. Esto significa que no se hace chequeo de tipo en tiempo de compilación o ejecución y los tipos de variable no son especificados explícitamente. Internamente, soporta los siguientes tipos:

• Boolean
• Integer
• Floating-point
• String
• Invalid

El programador no necesita especificar el tipo de las variables; WMLScript automáticamente tratará de convertir entre los diferentes tipos cuando sea necesario. En este aspecto es más similar al Basic.

Otro punto a tener en cuenta, es que WMLScript no es orientado a objetos. Por lo tanto es imposible crear tipos de dato propios del programador.

Operadores

WMLScript soporta una variedad de operadores: desde operaciones de asignación de valores, operaciones aritméticas, lógicas y de comparación, de cadenas y de arrays.

Sentencias de control de flujo

WMLScript es virtualmente idéntico a JavaScript en este aspecto. Esto incluye sentencias if-else, bucles for, bucles while, break, y continue.

Funciones

Las funciones se declaran siguiendo la sintaxis:

```
extern function identifier(FormatParameterList)
Block;
```

El parámetro *extern* es opcional y es usado para especificar una función que puede ser llamada desde afuera de la unidad de compilación actual en la que la función está declarada.

```
function t(d, v) { var t = d / v; return t; };
```

El ejemplo de arriba simplemente usa dos variables, *d* (por distancia) y *v* (por velocidad), y las usa para calcular una variable *t* (por tiempo). El parámetro *return* es usado para devolver un valor, en este caso, *t*.

Las Librerías Estándar de WMLScript

Las Librerías Estándar de WMLScript

Mientras que WMLScript no soporta la creación de nuevos objetos mediante la programación OO, provee seis librerías que ayudan en el manejo de tareas comunes. Estas librerías incluyen:

• Lang. Esta librería contiene un conjunto de funciones cercanamente relacionadas al núcleo del lenguaje

WMLScript. Incluye funciones para manipulación de datos, cálculo de valores absolutos y generación de números aleatorios.

• Float. La librería float es opcional y es soportada solo en los clientes con capacidades de punto flotante. Típicamente las funciones provistas por esta librería son sqrt(), round(), y pow().

• String. La librería String contiene un conjunto de funciones para las operaciones de cadena. Algunas de las funciones incluidas son length(), charAt(), find(), replace(), y trim().

• URL. Esta librería contiene un conjunto de funciones para el manejo de URLs relativas y absolutas. Algunas de las funciones incluidas son getPath(), getReferer(), y getHost().

• WMLBrowser. Esta librería contiene funciones para que WMLScript pueda acceder al contexto WML asociado. Estas funciones no deben tener ningún efecto colateral y deben retornar inválido en casos en el que el sistema no soporte el WMLBrowser y donde el intérprete no es invocado por el WMLBrowser. Las funciones usadas más comúnmente de ésta librería son include(), go(), prev(), next(), getCurrentCard(), y refresh().

• Dialogs. Esta librería contiene un conjunto de funciones varias de interface de usuario, incluyendo prompt(), confirm(), y alert().

Cuando se llama a una función incluida en una de las librerías estándar de WMLScript, se debe incluir el nombre de la librería junto con la llamada a la función. Por ejemplo, para llamar a la función length() de la librería String:

```
var a = String.length("1234567890");
```

Un ejemplo de WMLScript

A modo de ejemplo, se presenta a continuación un segmento de código de una pequeña aplicación que elige un número al azar y lo presenta al usuario. Esta aplicación se compone de dos archivos: el primero, *random.wml*, contiene dos cartas para la presentación de la aplicación y para el

resultado de la elección por azar; el segundo, *random.wmls*, contiene el script que genera un número aleatorio.

```
<!-- random.wmls -->
extern function getRandom() {
  var r = Lang.random(100);
  WMLBrowser.setVar("RESULT", r);
  WMLBrowser.go("random.wml#carta2");
}
```

```
<!-- random.wml -->
<wml>
  <card id="carta1" title="Ejemplo Random">
    <p align="center">
      Presione Tirar
    </p>
    <do type="accept" label="Tirar">
      <go href="random.wmls#getRandom()"/>
    </do>
  </card>
  <card id="carta2" title="Resultado Random">
    <p align="center">
      Ud. sacó $(RESULT)
    </p>
  </card>
</wml>
```

Cód.5-1 Un ejemplo de WMLScript: random

Capítulo 6. XHTML, la evolución del WML

Introducción
Markup Languages
WAP y WML
Evolución del WAP
Evolución del WML

El Lenguaje XHTML
XHTML Básico
XHTML MP
Comparación entre el WML1.x y XHTML MP + WAP CSS
XHTML comparado con HTML
CSS, Hojas de estilo en XHTML
Migrando desde el WML1.x a XHTML MP
El DTD
Protocolos de Transmisión
Cuestiones acerca de los navegadores
Configurando un servidor para brindar contenido XHTML/WML

Capítulo 6. XHTML, La evolución del WML

Introducción

En este capítulo se describe la evolución de los lenguajes de marcas orientados al WAP, desde el Wireless Markup Language (WML) 1.x hasta el Extensible Markup Language Mobile Profile (XHTML MP), destinado a los desarrolladores de aplicaciones (WAP) quienes estén creando servicios móviles de Internet usando WML. La meta es dar a los desarrolladores una guía para migrar desarrollos WML a XHTML.

La nueva generación de la especificación de servicios móviles es WAP 2.0, la cual está apoyada por las especificaciones de navegación de la Open Mobile Alliance (OMA). Hasta WAP 2.0, había dos especificaciones diferentes para los lenguajes usados en el mundo de las redes móviles y fijas: HTML para la World Wide Web (WWW) y WML para los servicios inalámbricos.

XHTML acerca la Internet móvil a la Internet fija; siendo la única especificación para la Web móvil y para la Web tradicional. XHTML permite mejor y más fácil control sobre la presentación que el WML, de ahí que mejora la usabilidad de los servicios móviles para los consumidores. La especificación WAP 2.0 tiene dos elementos clave en términos de navegación móvil: Juntamente con los cambios del lenguaje WML al XHTML MP, los cambios del protocolo de transmisión WAP al Wireless Profiled TCP/IP (wTCP/IP). Ambos XHTML y TCP/IP son estándares en la Internet fija y su migración soporta la convergencia entre los dominios fijos y móviles.

Markup Languages

El SGML (Standard Generalized Markup Language) fue el primer lenguaje de marcas de propósito general para documentos destinados a ser procesados por computadoras. Consistía de un conjunto de etiquetas o elementos, los que identificaban o etiquetaban la naturaleza o el propósito de su contenido. Los tags estaban delimitados por los caracteres “<” y “>” para distinguirse del contenido del documento.

Usando SGML, se hace posible diseñar y reforzar la estructura de un documento estableciendo reglas usando los elementos dentro del documento. Un Document Type Definition (DTD) es un tipo de documento que, usando una sintaxis

especial, define qué elementos pueden estar presentes dentro de un documento y en qué orden deberían o no aparecer. Esto dio a luz a un conjunto de reglas de anidamiento de elementos, y demás.

Ya que SGML es un lenguaje de marcas de propósito general, es posible crear diferentes tipos de documentos usando diferentes tipos de DTDs. Entonces, por ejemplo, se puede separar un DTD usado para escribir manuales de software, o para publicaciones matemáticas, o para proposiciones, etc.

HTML es un lenguaje de marcas, basado en SGML, usado para escribir contenido destinado a mostrarse en los navegadores Web. Este ampliamente usado y exitoso lenguaje, aunque está modelado sobre SGML, no es estrictamente una instancia conforme al lenguaje SGML porque no requiere un DTD y no requiere adhesión a las convenciones del SGML para el uso de elementos: por ejemplo, HTML permite el uso de elementos de apertura (como) sin una instancia siguiente de su par de cierre ( ).

Debido a que por ese entonces la gente deseaba usar un contenido con marcas con propósitos más allá de simples pantallas (para lo que HTML es perfectamente adecuado), el lenguaje XML fue inventado. Con XML, debido a que es estricto y flexible a la vez, es posible usar contenido con marcas para otros propósitos como comercio electrónico, procesamiento del lado del servidor con base de datos, etc.

XHTML representa la evolución del HTML, acercándolo a una mayor conformidad con reglas más estrictas comunes a todos los lenguajes XML. XHTML es usado para mostrar contenido en los navegadores Web así como el HTML ha siempre sido usado.

WML (*Wireless Markup Language*) es un lenguaje XML, con su propio DTD, usado para escribir contenido destinado a pantallas de terminales móviles.

WAP y WML

El WAP (*Wireless Application Protocol*) es un conjunto de estándares globales que especifican el comportamiento de todos los protocolos, lenguajes, y componentes requeridos para la implementación de aplicaciones y navegación de la Internet móvil. Está diseñado para acomodar las limitaciones de dispositivos pequeños con una memoria limitada y pantallas

pequeñas con conexiones de bajo ancho de banda a la Internet, la que puede ser poco confiable y tener alta latencia.

En 1998 el Foro WAP creó un nuevo lenguaje de marcas llamado WML 1.1 y un protocolo de red llamado Wireless Protocol Stack (WPS). WML 1.1 fue actualizado a la versión 1.3; esta generación de WAP sería referida como “WAP 1.x” y el lenguaje como “WML 1.x”

Mientras que WAP 1.x funciona razonablemente bien, el éxito comercial de WAP ha sido limitado, en parte por la dificultad y costo del desarrollo de aplicaciones móviles usando formatos completamente nuevos. Los desarrolladores tenían que aprender el nuevo lenguaje, a menudo usando nuevas herramientas y los servidores Web tenían que ser reconfigurados para soportar los nuevos tipos de contenido. El lenguaje de marcas WML está también limitado en su habilidad para controlar formatos, resultando en páginas que son similares a la “primera generación” de páginas Web—sin color, con la mayoría de los ítems alineados a la derecha y desde arriba hacia abajo a lo largo de la página. No solo estas páginas son menos visualmente atractivas, que otras páginas Web más familiares, sino que también son a veces, más difíciles de usar.

Pero el problema más serio con WML es que la especificación no define reglas para la visualización de elementos en la página. De ahí que cada fabricante de navegadores para dispositivos móviles muestra el mismo contenido en formas drásticamente diferentes, aunque todos son conforme a la especificación WML. Esto ha llevado a problemas de interoperabilidad que han afectado a los desarrolladores de contenido móvil, quienes deben detectar qué dispositivo está buscando contenido y conducir los resultados para que funcionen sorteando las diferencias entre las implementaciones del WML.

XHTML MP y WCSS son una enorme mejora para el control de la distribución y consistencia, aliviando la mayoría de los problemas de interoperabilidad.

Evolución del WAP

La primera generación de WAP especifica un ambiente de aplicación que incluye al WML 1.x y una pila de protocolos de comunicación optimizados. Para los dispositivos móviles, el futuro del WAP reside en su alineación cercana con los

vastamente aceptados estándares de Internet. El Foro WAP y la *World Wide Web Consortium* (W3C) han exitosamente definido los estándares de Internet móvil durante varios años. Para permitir la convergencia entre el contexto de desarrollo para la Web fija y móvil, el Foro WAP adoptó el estándar XHTML junto con *Cascading Style Sheets* (CSS) de la W3C como una base para WAP 2.0. Desde mediados de 2002, las especificaciones del Foro WAP han sido continuadas por la *Open Mobile Alliance* (OMA). La transición XHTML MP y WAP CSS refuerzan la posición de los navegadores móviles dentro del flujo principal de Internet y permite un mayor rango de diseños de presentación y formatos que antes eran imposibles.

Evolución del WML

XHTML es el cimiento de la navegación móvil de la nueva generación. El lenguaje de marcas XHTML Mobile Profile junto con WAP CSS son lo que mayormente forman el WAP 2.0. Este ofrece algunas ventajas sobre el WML original, particularmente en el área de la presentación de contenidos, pero toma prestados una gran cantidad de cosas que funcionan bien en WAP. Por ejemplo, scripting, push, e integración con funcionalidad de telefonía (WTAI), adoptados de las más tempranas especificaciones WAP. WML es un lenguaje de marcas basado en el *Extensible Markup Language* (XML). La especificación oficial de WML fue inicialmente desarrollada y mantenida por el Foro WAP, la industria-consorcio conformada por Nokia, Openwave, Motorola, y Ericsson. El *Hypertext Markup Language* (HTML) es un lenguaje de marcas usado para crear documentos hipertextuales que son portables de una plataforma a otra. Los documentos HTML son similares a los documentos SGML, con una semántica genérica, apropiada para representar información para un amplio rango de aplicaciones. HTML ha sido la lingua franca de las publicaciones WWW. XHTML MP, de acuerdo a la W3C, representa los cambios más importantes al HTML desde que el HTML 4.0 fue lanzado en 1997. De hecho, la última versión de HTML (4.1) forma la base del XHTML: toda la definición de etiquetas y la sintaxis es igual.

En la figura siguiente se muestra la evolución del HTML en el XHTML.

Fig.6-1 La evolución del HTML en XHTML

XHTML simplemente agrega modularidad y vigoriza una adherencia estricta a las reglas del lenguaje. Como resultado, XHTML incorpora una estructura limpia a las páginas Web, lo que es especialmente importante cuando se trata con las pequeñas pantallas y la energía limitada de los dispositivos móviles.

La W3C recomienda XHTML para todos los futuros desarrollos Web para dispositivos de escritorio, así como cualquier otro dispositivo, incluido el conjunto de dispositivos móviles. XHTML Básico es la versión de XHTML 1.0 para los dispositivos móviles. XHTML Básico está diseñado para los clientes Web que no soportan el conjunto completo de las características del XHTML, por ejemplo, los clientes Web de teléfonos celulares, PDAs, busca personas, etc. XHTML MP es un conjunto estricto del XHTML, comenzando con el XHTML Básico y agregando algunos elementos del XHTML 1.0 completo, que son útiles en navegadores móviles que incluyen soporte para elementos de presentación y hojas de estilo internas.

Las hojas de estilo, o Cascading Style Sheets (CSS) describen la forma en que los documentos son presentados en la pantalla del navegador. CSS separa la presentación del contenido. Los cambios a la presentación pueden hacerse en una hoja de estilo, permitiendo que estos cambios se reflejen automáticamente a lo largo del documento entero. WAP CSS es la versión móvil de CSS definido por la OMA. Es un subconjunto de CSS, que omite características que no son apropiadas para dispositivos pequeños y que agrega un pequeño conjunto de extensiones específicas de WAP al CSS.

La especificación del WML 2.0 define extensiones de compatibilidad con el WML 1.x, que son usadas para lograr la compatibilidad hacia atrás en vez de implementar la funcionalidad completa del WML 1.x. Estas extensiones hacen posible usar características específicas del WML 1.x en clientes

compatibles con WAP 2.0 que no tienen navegadores duales[4] que soporten ambos XHTML MP y WML 1.x.

La figura 6-2 muestra la relación entre los lenguajes de marcas.

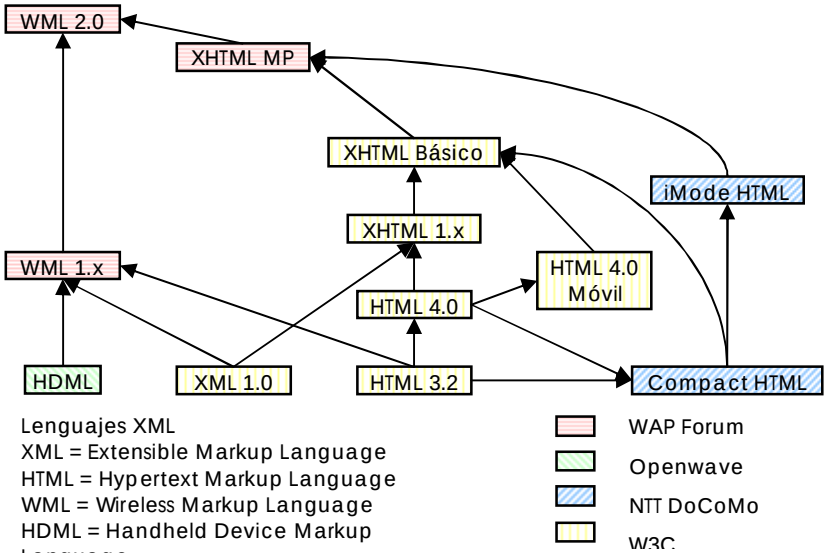


Fig.6-2 Relación entre los lenguajes de marcas

[4] Más adelante en este capítulo (en la sección Protocolos de Transmisión) se trata con mayor detenimiento el concepto de navegador dual.

La evolución de la navegación móvil se muestra en la figura 6-3 que ilustra las características más significativas que trae a la Internet móvil cada versión WAP.

Fig.6-3 Evolución del protocolo WAP

El Lenguaje XHTML

El Foro WAP ha tomado el DTD del XHTML Básico de la W3C y lo ha especializado para el uso en navegadores móviles agregándole los siguientes elementos, así como permitiendo el uso de atributos de estilo.

<hr> <b> <i> <big> <small> <style>

Con XHTML MP, un documento puede ser presentado a un número máximo de clientes, incluyendo una variedad de dispositivos móviles caracterizados por pantallas de diferentes tamaños, formatos y capacidades de presentación. Ya que XHTML MP está aprobado por la W3C y el Foro WAP, es una parte integral del conjunto de los estándares de Internet, el

cual aseguraría su distribución y su aceptación desde ahora, y asegura la viabilidad de su continuidad a largo plazo.

XHTML Básico

De acuerdo a la especificación de la W3C, el DTD del XHTML Básico define un documento lo suficientemente rico para ser usado en la creación de contenido y distribución precisa de los elementos en el documento, aún para ser usado a lo largo de diferentes tipos de dispositivos: de escritorio, PDAs, TV, y teléfonos celulares. El XHTML Básico es la versión móvil del XHTML 1.0, e incluye en su mayoría el contenido del XHTML 1.0 excepto aquellas capacidades que no son apropiadas para los dispositivos con pequeñas pantallas. El Foro WAP definió el XHTML MP para mejorar el XHTML Básico y agregar algunos elementos útiles y atributos del XHTML[5]. El XHTML es la piedra fundamental en el esfuerzo de la W3C en crear un estándar que provea contenido Web más rico en un siempre creciente rango de plataformas. Usando XHTML, los proveedores de contenido encontrarán más fácil producir contenido para un conjunto de plataformas y con mucha más seguridad sobre la forma en la que el contenido será mostrado. En contraste, WML 1.x no asegura una distribución de elementos consistente a lo largo de los diferentes dispositivos. Esto es de particular importancia para los desarrolladores y prestadores de servicios que tratan con varios proveedores y clientes con diferentes interfaces del usuario en diferentes dispositivos.

Algunas de estas ventajas del XHTML sobre el HTML incluyen:

- Los documentos pueden ser validados más fácilmente.
- Los documentos pueden ser transformados en otros formatos internamente en los dispositivos móviles, por ejemplo usando XSLT.
- Los fragmentos de documentos pueden ser descargados más rápido.
- Los textos pueden ser almacenados más eficientemente en bases de datos Orientadas a Objetos.

[5] Vea “El Lenguaje XHTML” más adelante en este capítulo para ampliar este concepto.

En la figura 6-4 se muestra una aplicación WAP tanto en un teléfono celular como en un navegador clásico de PC.

Fig.6-4 La misma página XHTML en un navegador de un teléfono celular Nokia y en Microsoft Internet Explorer

XHTML MP

La siguiente es una guía para usar el XHTML MP al momento de diseñar servicios para los navegadores de los dispositivos móviles. Provee un resumen de los elementos del lenguaje que son críticos desde el punto de vista de las capacidades soportadas por los navegadores.

En esta sección no se incluyen todos los elementos posibles del XHTML MP. Para más información sobre el lenguaje completo, se recomienda visitar el sitio del Foro WAP en Internet, en http://www.wapforum.org

Formato del texto

El contenido de una página XHTML se muestra en el área de display principal de la aplicación. El orden de los elementos es significativo, ya que aparecerán en pantalla en este mismo orden. Los espacios en blanco son

ignorados a no ser que se usen dentro de un elemento <pre>...</pre>[6]

La siguiente tabla contiene todos los elementos de formato de texto soportados.

Elemento	Formato
em	Fuente Itálica
strong	Fuente Negrita
b	Fuente Negrita
i	Fuente Itálica
u	Subrayado
big	Fuente tamaño grande
small	Fuente tamaño chico
p	Se inserta un párrafo
br	Se inserta un salto de línea

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Formato XHTML MP</title>
</head>
<body>
<h1>Elementos de Format</h1>
<p>
<big>Big</big>
<small>small</small><br/>
<b>Bold</b> <i>Italic</i>
</p>
<br/>
</body>
</html>
```

Cód.6-1 Formato de texto

[6] Para más información sobre el elemento <pre> de XHTML, vea más adelante en este capítulo el apartado pre.

Fig.6-5 Formato de texto

Además de los anteriores se pueden utilizar los siguientes elementos:

<pre>, , <q>, <code>, <cite>, <dfn>, <abbr title="">, <acronyn="">, <blockquote>, <var>, <kbd>, <samp>.
En el siguiente ejemplo se ilustra el uso de estas etiquetas.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Texto XHTML</title>
</head>
<body>
<h1>Elementos de Format</h1>
<p>
<em>Emphasized.</em>
<strong>Strong emphasis.</strong>
<var>Una variable</var>
<kbd>Texto de Teclado</kbd>
<samp>Ejemplo</samp>
<code>Código</code>
<dfn>Definición</dfn>
Como <cite>Shakespeare</cite> dijo:</p>
<blockquote>
  <p>Una rosa por cada nombre.</p>
</blockquote>
<p><cite>Cicero</cite> termina su discurso con
<q>ceterum censeo</q>.
<acronym title="Hypertext Markup
Language">HTML</acronym> es un acrónimo.
<abbr title="Abreviación">abr.</abbr> en una
abreviación.</p>
<pre>
```

```
    Y  este  es  un
texto  preformateado
</pre>
</body>
</html>
```

Cód.6-2 Formato avanzado de texto

Fig.6-6 Formato avanzado de texto

Zooming

La propiedad zooming afecta el tamaño del texto mostrado en el área principal de la aplicación. Esta función solo afecta a los tamaños de las fuentes, no a las imágenes, tablas o al espacio alrededor del texto u otros elementos que no son proporcionalmente específicos al tamaño del texto.

Se puede aplicar zooming incluso a un texto cuyo tamaño está especificado en WCSS. Si el tamaño de la fuente se define usando CSS, el tamaño disponible más cercano soportado por el dispositivo es usado. En resumen, el zooming no se aplica al tamaño de una fuente especificado en una CSS, sino al tamaño de la fuente mostrada en la pantalla de cada dispositivo, para asegurar, de este modo, que siempre se muestre el texto con el mayor tamaño posible. La mayoría de los

dispositivos móviles tienen tamaños seleccionables de fuente para restringir la cantidad de zooming disponible en el navegador. La tabla siguiente muestra los tamaños de texto con distintos niveles de zoom aplicados.

Nivel de Zoom	Tamaños de texto definidos en píxeles		
	12	13	17
All large	17	17	17
Larger text	13	17	17
Normal text	12	13	17
Smaller text	12	12	13
All small	12	12	12

Párrafos y alineación de contenidos

El elemento párrafo, <p>, habilita el word wrapping y la alineación de contenido. Un párrafo siempre comienza con una nueva línea. El contenido dentro de un párrafo puede ser alineado a la izquierda, centro o derecha; por defecto la alineación es hacia la izquierda. La alineación se especifica usando atributos de párrafo (ver el siguiente ejemplo). El método recomendado para especificar alineación en XHTML es usando hojas de estilo. El texto es cortado automáticamente a no ser que se especifique lo contrario. El *word wrapping* puede ser establecido usando el atributo *mode* de un párrafo. Sin embargo, es importante notar que el usuario puede especificar el estado de wrapping en la configuración del dispositivo.

```
<p align="center">Align &amp;quot;center&amp;quot;</p>
<p align="right">Align &amp;quot;right&amp;quot;</p>
<p align="right">
   <img alt="Ave" src="ave.gif" align="top"
hspace="0" vspace="0"/>
</p>
```

Salto de Línea

El elemento <br> fuerza los saltos (cambios) de la línea de texto actual. El atributo *clear* especifica como es cortado el texto, luego de un salto de línea forzado, a los lados de un objeto flotante, por ejemplo, una imagen. Los valores posibles son *left*, *right*, *all*, y *none*. El valor por defecto es

left. *left* previene que los elementos flotantes se muestren a la izquierda del elemento. *right* previene que los elementos flotantes se muestren a la derecha del elemento. *all* previene que los elementos flotantes se muestren a ambos lados del elemento. En todos estos casos, el margen superior del elemento es incrementado hasta que el borde superior del elemento esté justo debajo del margen inferior del elemento flotante. *none* permite a los elementos flotantes aparecer a ambos lados del elemento.

Fieldset

El elemento fieldset, <fieldset>, permite agrupar textos y campos relacionados. La idea detrás del agrupamiento es tratar y asegurarse de que los ítems agrupados se muestren en una misma vista. Sin embargo, si los ítems no pueden ser mostrados en la misma pantalla, el usuario no recibe ninguna notificación. <fieldset> siempre comienza una nueva línea y termina con un salto de línea. El elemento <fieldset> no es parte del XHTML MP pero es soportado en la mayoría de los navegadores de los dispositivos móviles, de ahí que puede ser usado en los documentos XHTML.

Pre

El elemento pre, <pre>, causa que el texto sea mostrado lo más "preformateado" posible. Preformateado significa que los espacios en blanco se dejen intactos, la fuente sea mostrada como texto plano, y el word wrapping esté habilitado o deshabilitado de acuerdo a como esté configurado el navegador.

Links

Un link se especifica con el elemento <a>. Por defecto, los links se muestran subrayados, y un enlace que apunta a una página que no existe en la cache es de color azul, un link que apunta a una página que ya existe en la cache, de color púrpura.
Si hay una imagen y texto declarados dentro de un link, la imagen se muestra con un borde y el texto subrayado. Los desarrolladores pueden especificar diferentes colores de links usando WCSS. Cuando un usuario selecciona un link, se realiza una tarea asociada. Para ciertos teléfonos celulares, cuando un link es seleccionado, la lista de

Opciones del navegador se extiende incorporando la opción Abrir. El usuario puede también activar un link seleccionado usando la tecla de acción por defecto del dispositivo (a menudo conocida como Navi Key™).
Si un link no tiene titulo o etiqueta, la dirección del URL se muestra completa como un link común.

```
<p>
<a href="ave.xhtml"><img src="ave.gif"
alt="ave"/></a>
</p>
```

Teclas de acceso rápido (Access Keys)

Algunos navegadores poseen el atributo access key para páginas tanto XHTML como WML (aunque tienen diferente efecto dependiendo del lenguaje usado).
Pueden ser usadas con los elementos <a>, <input>, y <textarea>.

Este atributo puede ser usado para asignarle teclas de hardware a los elementos de la página, así como links y áreas de texto, facilitando de este modo la navegación. Las teclas de hardware del 0-9 pueden ser asignadas a ciertos elementos de la página.

Si se asigna el # o * como una tecla de acceso, es ignorado por el navegador.

Para indicarle al usuario que tecla presionar para activar un elemento, se puede agregar la etiqueta de la tecla al elemento. Por ejemplo, si se asocia la tecla 1 con un link a la página de inicio, la etiqueta del link podría ser “1. Inicio”.

```
<a href="index.html" accesskey="1">1.Inicio</a>
```

Una tecla de acceso rápido tiene efecto sobre un elemento aún si éste no está resaltado o visible. Después de que una tecla de acceso rápido ha sido presionada, el elemento es seleccionado si no lo estaba.

Fig.6-7 Teclas de acceso rápido para los links

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD HTML
Mobile 1.0//EN" "http://www.wapforum.org/1999/
xhtml-mobile10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>XHTML 1</title></head>
<body>
<a name="top"/>
<h2>Anchors</h2>
<p>
<a href="#bottom">Abajo</a><br/>
1 <a href="file2.xhtml"
accesskey="1">Archivos</a><br/>
2 <a href="wtai://wp/mc;8005551212"
accesskey="2">Directorio</a><br/>
3 <a
href="mailto:joe@company.com?subject=Asunto&amp;
body=Mensaje" accesskey="3">Email Joe</a><br/>
<a href="#top">Arriba</a><br/>
<a name="bottom"/>
</p>
</body>
</html>
```

Cód.6-3 Anchors

Procesamiento de las entradas en XHTML MP

El procesamiento de entradas permite al usuario ingresar información que requiere un servicio.

Hay dos clases de elementos de entrada: campos de entrada creados usando un elemento <input> y las listas de selección creadas usando un elemento <select>.

Para especificar cada opción dentro de un elemento <select>, se usan los elementos <option>. Los elementos de opción pueden ser agrupados usando un elemento <optgroup>.

Formularios

Los formularios de XHTML MP, comparados con los del WML, ofrecen a los desarrolladores de servicios mayor control sobre la distribución de las cajas de entrada.
Los desarrolladores pueden definir el tamaño exacto de la caja de entrada así como también el estilo de sus bordes.
Los forms son usados para contener las entradas dadas a un elemento <textarea> y para manejar las selecciones hechas en las listas de opción y selección.

Los formularios XHTML son enviados al servidor usando el botón Submit, a diferencia del WML, el cual usa enlaces que contienen elementos <postfield>[7].

Se puede usar CSS en los formularios para dar posición a los elementos usando padding y propiedades de margen.

Los elementos de un <form> soportados en XHTML MP son <input>, <optgroup>, <select>, <option>, <textarea> y <label>.

XHTML MP no soporta:

- El atributo *accept-charset* en el elemento <form>
- Los atributos *tabindex* y *label* en cualquiera de los elementos
- Los elementos <fieldset>, <button>, y <legend>
- El par atributo-valor *type="button"* en los elementos <input>

[7] Ver ejemplo de formularios WML usando elementos postfield, en el Capítulo 4.

form

El elemento <form> define los atributos *action* y *method* usados para enviar los datos del formulario al servidor Web. El atributo action especifica el URL del servidor a donde los datos son enviados, mientras que el atributo method puede ser *get* o *post*.

Usando el método get se envían los datos del formulario como una cadena de valores de consulta agregados al URL.

```
<form method="get" action="process.cgi">
  Nombre: <input type="text" name="uname"/><br/>
  Password: <input type="password" name="pw"/>
</form>
```

Por ejemplo, el código anterior requerirá una página cuyo URL es http://process.cgi?uname=Joe&pw=xyzzy, como resultado, asumiendo que el usuario ingresó los valores "Joe" y "xyzzy" en los campos de entrada.

La ventaja del método get es permitir al usuario agregar este URL al *bookmark* (favoritos), y de este modo almacenar también los valores del formulario para ser usado nuevamente con los mismos datos. También es posible que el usuario edite esta URL almacenada para cambiar los valores de los campos.
La desventaja del método get es que no puede ser usado para enviar grandes cantidades de datos al servidor (las URLs tienen largo limitado), y los nombres y valores de los campos son visibles. En el ejemplo anterior, el valor del campo password será visible para alguien que examine el bookmark guardado.
Usando el método *post* se envían los datos del formulario embebidos en el header http del request. Esto es útil para enviar cantidades de datos mayores, para hacer más difícil a los usuarios ver los valores, y asegurar que el request del form no pueda ser guardado en el *bookmark*.
Una página que ha sido requerida usando post, siempre será recargada desde el servidor, y no será mostrada desde la cache del navegador, excepto cuando el usuario regresa a la página usando *back*.

Siempre que se retroceda en el historial, las páginas se mostrarán exactamente como el usuario previamente las había visto, a no ser que la página tenga definido *no-cache* con lo cual no será guardada en la cache del browser, o cuando el tiempo de cache haya expirado.

Elementos de entrada

Un elemento de entrada es un campo de edición rectangular. Su tamaño es determinado por el atributo *size*. Además tiene un atributo *type* cuyos valores válidos son *text*, *password*, *checkbox*, *image*, *radio*, *reset*, *submit*, y *hidden*.

• Si se usa el tipo *text*, la entrada es mostrada al usuario en una forma legible, mientras que cuando se usa un tipo *password*, los caracteres que se ingresan son camuflados. El valor por defecto del atributo *type* es *text*.

• Cuando el elemento <input> es de tipo *checkbox*, se muestra un checkbox. *checked* es el atributo que se usa para definir el estado inicial de un checkbox; el valor por defecto es no chequeado. No se insertan saltos de línea antes o después del checkbox. Cuando el checkbox tiene foco, se dibuja un recuadro alrededor de él.

• Cuando el elemento <input> es de tipo *image*, un botón submit gráfico se muestra al usuario. Tampoco se insertan saltos de línea antes o después de un elemento gráfico. Un botón gráfico submit es mostrado como un enlace de imagen.

• Cuando el elemento <input> es de tipo *radio*, se muestra un radio button. Hay un valor checked para establecer el estado inicial de un radio button; el valor por defecto es no chequeado.
Cuando un radio button tiene foco, se dibuja un rectángulo alrededor de él. Si dos o más radio buttons con el mismo nombre en el mismo form tienen el atributo checked, solo el primero declarado estará preseleccionado. Cuando un radio button es activado, todos los demás radio buttons con el mismo nombre en el mismo form serán desactivados.

• Cuando el elemento <input> es de tipo *reset*, se muestra un botón reset. No se insertan saltos de línea ni antes ni después de un botón reset. La etiqueta por defecto es “Reset”.

• Cuando el elemento <input> es de tipo *submit*, se muestra un botón submit. No se insertan saltos de línea ni antes ni después de un botón. La etiqueta por defecto es “Submit”.

Fig.6-8a Elementos de entrada de un formulario: <input type="text">

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD HTML
Mobile 1.0//EN"
    "http://www.wapforum.org/1999/xhtml-
mobile10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Franklin 2</title>
</head>
<body>
<h1>La segunda página de Franklin</h1>
<p>
<form action="process.xhtml" method="get">Nombre:
  <input type="text" name="fname" title="Nombre"
    style="-wap-input-format:ammmmmm"/><br/>
  <input type="submit"/>
</form>
</p>
</body>
</html>
```

Cód.6-4 Elementos de entrada de un formulario: <input type="text">

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD HTML
Mobile 1.0//EN"
    "http://www.wapforum.org/1999/xhtml-
mobile10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>XHTML Training</title>
</head>
<body>
<form action="process.xhtml" method="get">
<p>
  <input type="checkbox" name="wtext"
title="Texto"
    checked="checked"/>Responder con texto<br/>
  <input type="checkbox" name="qtext"
title="Cita"/>
    Responder con cita<br/>
  <input type="submit" value="Seguir"/>
</p>
</form>
</body></html>
```

Cód.6-5 Elementos de entrada de un formulario: <input type="checkbox">

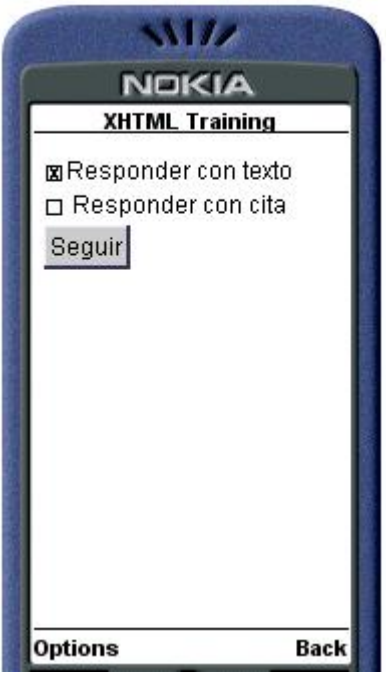


Fig.6-8b Elementos de entrada de un formulario <input type="checkbox">

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD HTML
Mobile 1.0//EN"
    "http://www.wapforum.org/1999/xhtml-
mobile10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>XHTML Training</title>
</head>
<body>
<form action="process.xhtml" method="get">
<p>
  <input type="radio" name="wtext" title="Texto"
    checked="checked"/>Responder con texto<br/>
  <input type="radio" name="qtext" title="Cita"/>
    Responder con cita<br/>
  <input type="submit" value="Seguir"/>
</p>
</form>
</body>
</html>
```

Cód.6-6 Elementos de entrada de un formulario: <input type=”radio”>

Fig.6-8c Elementos de entrada de un formulario <input type=”radio”>

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>Formulario</title></head>
<body>
<h2>Formulario</h2>
  <form action="input.xhtml" method="get">
  <p><input type="text" name="t1" id="t1"
size="10" maxlength="20"
     value="texto"/><br/>
     <input type ="submit" name="Enviar"
value="OK"/>
  </p>
  </form>
  <p><a href="input.xhtml">Siguiente</a></p>
  <p><a href="input.xhtml">Inicio</a></p>
</body>
</html>
```

Cód.6-7 Formularios

La figura 6-9 muestra el resultado del código anterior.

Fig.6-9 Un formulario en colores

El tamaño de las cajas de texto (*input*, *text area*) depende además del tamaño que se le ha fijado, del texto dentro de ellas. Si una caja de texto excede el espacio libre en la actual línea de texto, se inserta un salto de línea antes de ella, y la caja de texto se dibuja en la línea siguiente. Si el texto de entrada es más grande que la caja de texto, el texto es truncado y se agrega una elipsis. El usuario puede ingresar caracteres a un elemento <input> luego de haberlo señalado. Además, puede abrir

la caja de textos en modo edición presionando la tecla de acción (navigation key, en varios dispositivos) o eligiendo Editar desde la lista Opciones, cuando una caja de texto esté seleccionada.
Cuando el usuario está ingresando texto en un campo, las teclas del menú de los dispositivos cambian a Listo (*Done*) y Cancelar (*Cancel*). El usuario confirma su entrada presionando Listo o aborta la edición presionando Cancelar.

Listas de selección

Las listas de selección son elementos que especifican una lista desde la que el usuario podría elegir opciones. Hay dos clases de listas de selección soportadas: listas de selección simple y listas de selección múltiple. El usuario puede seleccionar múltiples opciones si el atributo *multiple* se encuentra en *true* (los ítems seleccionados se marcan con un símbolo de selección). Una lista de selección de ítems en una carta es una caja rectangular de aspecto similar a un elemento <input>. Dentro de la caja de selección, hay una flecha a modo de enlace. Esta flecha se coloca en la parte más a la derecha de la última línea de la caja de selección. La flecha diferencia un cuadro de selección de un cuadro de edición, que se usa para los elementos <input>. El comportamiento de las listas de selección con respecto a las teclas de menú de los dispositivos móviles, es similar al de los elementos de entrada (vea la sección elementos de entrada de este capítulo). Las listas de selección pueden tener un título y un valor predefinido. El título es usado como una etiqueta para la lista de valores de la que los usuarios pueden elegir opciones. Los valores, una vez seleccionados, son mostrados en la caja de selección. Si más de un valor es seleccionado de una lista de selección múltiple, los valores son separados por comas.

Un ejemplo de listas de selección simple podría ser:

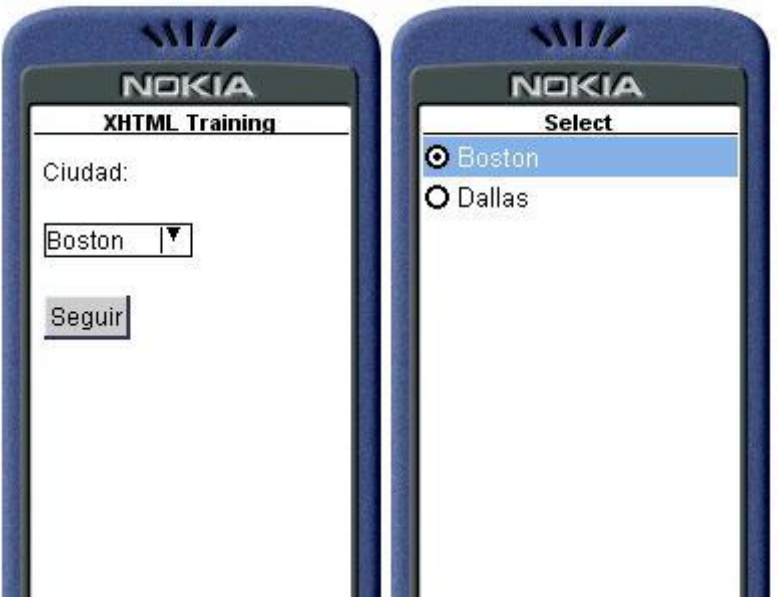


Fig.6-10 Listas de selección

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD HTML
Mobile 1.0//EN  "http://www.wapforum.org
/1999/xhtml-mobile10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>XHTML Training</title></head>
<body>
<form action="process.xhtml" method="get">
<p>Ciudad:<br/>
  <select name="ciudad">
    <option value="Boston">Boston</option>
    <option value="Dallas">Dallas</option>
  </select><br/>
  <input type="submit" value="Seguir"/>
</p>
</form>
</body></html>
```

Cód.6-8 Listas de selección

El siguiente ejemplo muestra la diferencia entre las listas de selección múltiple y las listas de selección simples.

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML
Mobile 1.0//EN" "http://www.wapforum.org
/DTD/xhtml-mobile10.dtd">
<html>
<head><title>Forms XHTML</title></head>
```

```
<body>
<a href="Forms.xhtml">Siguiente</a>
<h3>Forms</h3>
<form action="Forms.xhtml" method="get">
 Multiple:<br/>
 <select name="Select" size="2" multiple="multiple">
 <option selected="selected">Primero</option>
 <option>Segundo</option>
 <option selected="selected">Tercero</option>
 <option>Cuarto</option>
 </select><br/>
 None selected:<br/>
 <select name="Select2">
  <option>Primero</option>
  <option>Segundo</option>
  <option>Tercero</option>
  <option>Cuarto</option>
 </select><br/>
 <input type="hidden" name="Hidden" value="Hidden"/>
 <input type="submit" name="Submit" value="Send"/>
 <input type="reset" name="Reset" value="Clear"/>
</form>
<a href="Forms.xhtml">Siguiente</a> |
<a href="Forms.xhtml">Inicio</a>
</body>
</html>
```

Cód.6-9 Listas de selección simples y múltiples

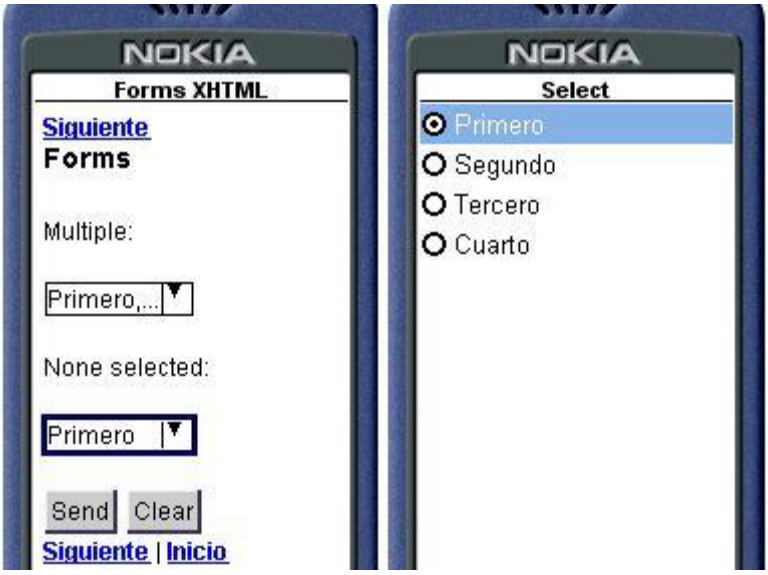


Fig.6-11a Listas de selección simple

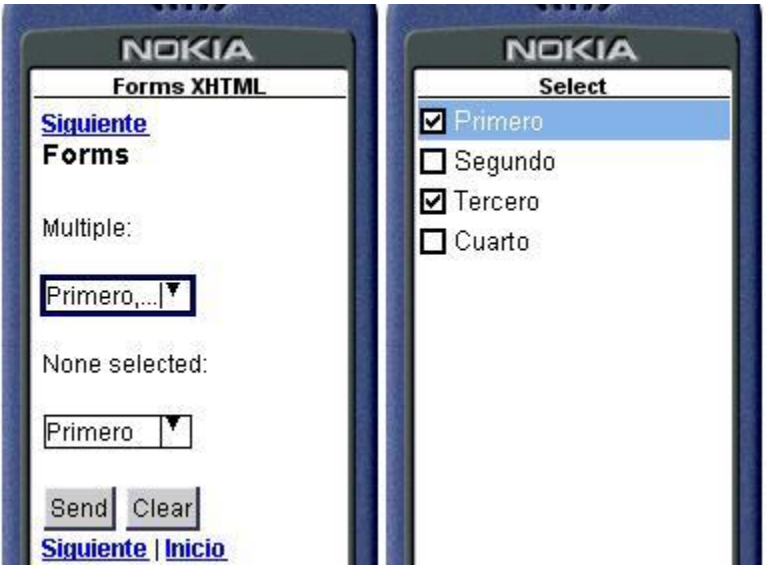


Fig.6-11b Listas de selección múltiple

Las cajas de selección cortan su contenido automáticamente y se extienden a sí mismas a dos o más líneas si es necesario (esto es, si los valores seleccionados no entran dentro del espacio que está disponible en la línea actual de la caja).

Si no se ha hecho ninguna selección, se muestra al usuario el tamaño estándar de la caja de selección. El elemento <option> especifica una opción simple dentro de un elemento <select>. Las opciones pueden ser agrupadas con el elemento <optgroup>, el cual especifica un grupo de ítems seleccionables en un elemento <select>.

Hidden Fields

Se pueden usar elementos de tipo <input type="hidden"/> para enviar valores al servidor. Las entradas *hidden* por supuesto, son ocultas al usuario. Típicamente se usan para alojar un número de sesión o algún valor que debe ser recordado entre requests de páginas. Si es evidente que se están creando demasiados valores *hidden*, entonces se estará manteniendo demasiado "estado" en la página XHTML, y en su lugar, se debería considerar el uso de las características del objeto *session* disponible en la mayoría de los servidores Web para almacenar el estado del usuario en una base de datos en el servidor. Alternativo al uso de un campo hidden para mantener una sesión, o al uso del objeto *session* de los servidores, se pueden usar *cookies* para mantener el ID de la sesión. Estos valores de

cookie se envían con cada request, mientras que los valores *hidden* solo se envían junto con los formularios.

Listas

Las listas ordenadas <ol> organizan ítems por número. Las listas no ordenadas <ul> son similares, pero, en lugar de números usan bullets. Cada una de estas listas contiene uno o más elementos <li>.

Las listas de definiciones <dl> crean ítems y definiciones, típicamente indentados.
Un elemento <dl> contiene pares de elementos <dt>...</dt> <dd>...</dd>.

En XHTML MP, los elementos <ol>, <ul> y <li> no soportan el atributo *type*, en su lugar, se usan estilos.
En el ejemplo siguiente se muestra el uso de listas en XHTML MP.

Fig.6-12 Listas ordenadas y no ordenadas

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD HTML Mobile
1.0//EN"
     "http://www.wapforum.org/1999/xhtml-
mobile10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head>
<title>Listas</title>
</head>
<body>
<h2>de Definición</h2>
  <dl>
    <dt>Átomo</dt>
    <dd>Un objeto se compone de átomos</dd>
    <dt>Protón</dt>
    <dd>Componente del átomo</dd>
  </dl>
<h2>Numeradas</h2>
  <ol>
    <li>Primero</li>
    <li>Segundo</li>
  </ol>
<h2>No numeradas</h2>
  <ul>
    <li>Carne</li>
    <li>Verduras</li>
  </ul>
</body>
</html>
```

Cód.6-10 Listas

Tablas

El elemento tabla, <table>, usado junto con los elementos <tr> y <td> pueden crear un conjunto de filas y columnas de datos, como por ejemplo texto, imágenes, enlaces, etc. Además puede usarse el elemento <caption>. Es posible tener texto, imágenes, y tablas en la misma página. Las celdas se muestran en filas y columnas con borde. Los atributos de presentación de una tabla deberían ser definidos en una hoja de estilo, por ejemplo, el ancho del borde, el color del borde, la alineación del texto, etc. El ancho automático de las celdas y columnas depende del número de columnas y la cantidad de contenido dentro de las celdas. Por ejemplo, si la tabla contiene tres o más columnas, el ancho de una celda de la tabla más su borde es un tercio del ancho de la pantalla, y si la tabla consiste de una única columna, el ancho fijo de una celda de la tabla y su borde es el ancho completo de la pantalla. Si las celdas de la tabla y las columnas son estrechas (solo una pequeña cantidad de contenido en las celdas), es posible tener más de tres columnas en la pantalla. El desarrollador puede también especificar el ancho de las celdas en las hojas de estilo o usando el elemento *style* o los atributos de estilo.

Fig.13a Tablas

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML
Mobile 1.0//EN"
"http://www.wapforum.org/DTD/xhtml-mobile10.dtd">
<html>
<head><title>Clima</title></head>
<body>
<table>
  <caption>París</caption>
  <tr>
   <th>Condiciones Actuales</th>
   <td><img src="sunny.gif" alt="Soleado"/></td>
  </tr>
  <tr>
    <th>Pronóstico</th>
    <td><img src="rain.gif" alt="Lluvia"/></td>
  </tr>
</table>
<h6>
  <a href="index.xhtml">Inicio</a> |
  <a href="clima.xhtml">Clima</a>
</h6>
</body></html>
```

Cód.6-11 Tablas

Aunque se puede producir cualquier diseño deseado, usando los elementos <p>, y del XHTML y luego asignarle propiedades de estilo con WCSS, usando tablas se puede distribuir el contenido de una manera tabular de forma muy eficiente.

Comparado con WML, XHTML MP con WCSS permite una mejor distribución de las tablas. Por ejemplo, en WML los bordes de las tablas no pueden ser modificados o removidos, pero esto si se puede lograr usando XHTML con WCSS. Las tablas pueden usarse para distribuir páginas completas XHTML. Este intento por distribuir páginas es ampliamente usado en la WWW, porque es una manera fácil de alinear contenido. Sin embargo, algunos navegadores móviles no calculan automáticamente el ancho y alto de las celdas basados en sus contenidos, entonces si se quieren tablas con alineamiento exacto, se deben especificar las propiedades *height* y *width* en píxeles.

Las tablas pueden usarse para mostrar contenido en una manera tabular, pero es importante diseñar tablas que quepan en el ancho de la pantalla del dispositivo, ya que algunos dispositivos portátiles no están optimizados para desplazamiento horizontal.

La siguiente es una guía de consejos para tener en cuenta cuando se usan tablas.

• Evitar definir el ancho de tabla que exceda el ancho del display del dispositivo.

• Evitar definir celdas de tabla que sean más altas que la altura de la pantalla. El usuario pude sentirse perdido y perder de vista el contexto de una tabla si las celdas son demasiado grandes.

• Usar el mismo tipo de borde para todas las celdas en una misma tabla. Las variaciones en estilo de borde dentro de una tabla pueden crear confusión cuando leemos el contenido de la misma. Apuntar la atención del usuario directamente al contenido de la tabla en lugar de a sus propiedades.

• Puede ser una buena idea agregar encabezados de columna (<th>) y un título a la tabla (<caption>). Sin embargo, si el contenido de una columna es evidente o

ya está explicada en el texto por encima de la tabla, entonces debe evitarse agregar encabezados de columna ya que requerirá un desplazamiento extra del usuario. Por supuesto, el título y los encabezados de tabla pueden ser obviados cuando usamos tablas simples para distribuir contenido de la página.

El anidamiento de tablas no es soportado, ni los atributos <align>, <cellpading>, <cellspacing> y <border>.

Las propiedades de WCSS usadas junto con XHTML MP permiten estilizar el *padding* de celdas, márgenes, alineación de textos y bordes, con un control individual sobre las propiedades left/right/top/bottom. WCSS también controla el grosor de los bordes y el color, en los dispositivos que tengan soporte.

El siguiente ejemplo muestra la construcción de una tabla para contener datos tabulares.

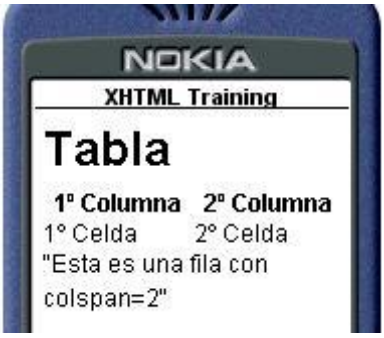


Fig.13b Una tablas para contener datos tabulares

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD HTML
Mobile 1.0//EN"
    "http://www.wapforum.org/1999/xhtml-
mobile10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>XHTML Training</title></head>
<body>
  <h1>Tabla</h1>
  <table>
  <tr>
    <th>1º Columna</th><th>2º Columna</th>
  </tr>
  <tr>
    <td>1º Celda</td><td>2º Celda</td>
  </tr>
  <tr>
```

```
    <td colspan="2">"Esta es una fila con
colspan=2"</td>
  </tr>
  </table>
</body>
</html>
```

Cód.6-12 Una tabla para contener datos tabulares

Este segundo ejemplo muestra una tabla para distribuir imágenes.

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD HTML
Mobile 1.0//EN"
    "http://www.wapforum.org/1999/xhtml-
mobile10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>XHTML Training</title>
<style type="text/css">
table,tr,td { padding:0px; margin:0px;
border:none; }
img { margin:0px; padding:0px; border:none; }
</style>
</head>
<body>
 <p>
   <img src="square.gif" alt="rect" width="14"
        height="14"><br/>
   cellpadding=0, cellspacing=0<br/>
   padding:0px;margin:0px<br/>
 </p>
 <table align="left" width="42" cellpadding="0"
cellspacing="0">
  <tr>
    <td width="14" height="14"><img
src="square.gif" alt="rect" width="14"
height="14"></td>
    <td width="14" height="14"><img
src="square.gif" alt="rect" width="14"
height="14"></td>
    <td width="14" height="14"><img
src="square.gif" alt="rect" width="14"
height="14"></td>
  </tr>
 </table>
</body>
</html>
```

Cód.6-13 Una tabla para manejar la disposición de imágenes

Fig.6-13c Una tabla para manejar la disposición de imágenes

Imágenes

Así como en la WWW, las imágenes son valiosas en los servicios WAP, ya que pueden a menudo "decir" más que las palabras en la misma cantidad de espacio y a su vez "cortar" un bloque monótono de texto. Sin embargo, las imágenes deben ser usadas cautelosamente para evitar reducir la usabilidad de un servicio.

Si una imagen no cabe en el ancho restante de la línea actual, un salto de línea es insertado antes de la imagen. Los usuarios pueden seleccionar en los navegadores si quieren descargar automáticamente las imágenes o no. Cuando hay una imagen visible en la página, puede ser editada, impresa, etc. Las imágenes también pueden guardarse en el álbum de fotos del dispositivo, directamente desde la página XHTML (si el dispositivo posee esta característica). Las imágenes pueden ser usadas dentro de una tabla y hasta como enlaces (ver el apartado *links* de esta misma sección). Si, por alguna razón, una imagen no puede ser descargada (incluso cuando la descarga automática de imágenes está deshabilitada), se muestra al usuario un icono de imagen "rota" y el texto alternativo. El atributo alt es truncado si excede el espacio horizontal disponible para texto en la línea actual.

La siguiente es una guía de recomendaciones a tener en cuenta cuando se usan imágenes en los servicios WAP.

• Usar formatos WBMP o GIF cuando se agregue una imagen a un servicio. La mayoría de los dispositivos del mercado soportan esos formatos.

Una ventaja de usar GIF es que además, los navegadores clásicos de PCs de escritorio lo soportan, mientras que al WBMP, generalmente no.

• Evitar definir imágenes que excedan el tamaño de la pantalla, tanto horizontal como vertical. Las imágenes grandes requerirán desplazamiento y algunos dispositivos no están optimizados para el scrolling horizontal.

• Crear algunos píxeles de espacio en blanco alrededor de las imágenes para prevenir que otras imágenes o texto que la rodeen, se "peguen" a ellas. Se puede usar la propiedad *margin* de WCSS para tal fin.
• Usar la propiedad *float:left* o *float:right* de WCSS para hacer que el texto "flote" alrededor de las imágenes que son más pequeñas que la mitad del ancho de pantalla. Flotar el texto en vez de comenzarlo en una nueva línea resulta en menos scrolling para el usuario.

• Evitar definir bordes alrededor de una imagen ya que hace más difícil al usuario distinguir una imagen-plana de una imagen-enlace. A diferencia de las imágenes planas, las imágenes enlace pueden ser seleccionadas e identificadas por un borde que las rodea (no válido en WML 1.x).

• Usar el atributo *alt* dentro del elemento <img> para especificar texto alternativo, que será mostrado en caso de que la imagen no haya podido ser descargada. Esto mismo debe hacerse también con las imágenes enlace.

• Cuando se crea una imagen enlace (un elemento <a> que contiene inmediatamente un elemento <img>), también se debe especificar el atributo *accesskey* si el enlace es importante para la navegación.
Todos los enlaces que especifican el atributo *accesskey* son automáticamente agregados al menú Opciones.
Se debe tener en cuenta, que para el caso de las imágenes, es el texto alternativo (atributo *alt*) el que se agrega en el menú Opciones y no la imagen en sí. El texto será truncado si excede el ancho del menú Opciones.

• Evitar usar *splash screens*. Un *splash screen* es una página de entrada al servicio, que típicamente muestra el logo del servicio o de la compañía. Los usuarios WAP quieren acceso instantáneo a la información que están buscando, y no tener que esperar (y pagar) por el tiempo que le toma cargar el splash screen. Un pequeño logo en

la primera página es suficiente para proveer una marca y familiarizar un servicio.

• Analizar el grupo al que apunta el servicio, y usar imágenes de acuerdo a esto. Por ejemplo, las animaciones e imágenes divertidas pueden ser apropiadas para un mercado joven, pero no para una aplicación de negocios. En XHTML MP, el elemento <img> no soporta los atributos *align*, *border*, *hspace*, *vspace* o *usemap*.

Comparación entre el WML1.x y XHTML MP + WAP CSS

Ambos, WML 1.x y XHTML MP son lenguajes de marcas óptimos para dispositivos con pequeñas pantallas y capacidades de presentación limitadas. Estos lenguajes pueden ser usados en dispositivos con memoria limitada porque ni el navegador WML ni el navegador XHTML MP requieren demasiada memoria para ejecutarse.

Tanto WML como XHTML MP vigorizan una adhesión estricta a las reglas del lenguaje XML. Las diferencias más significativas entre WML 1.x y XHTML MP con CSS están relacionadas con las capacidades de presentación de contenido y las de portabilidad entre diferentes categorías de interfaces de usuario.

La transición hacia el XHTML provee dos ventajas técnicas comparadas con el ambiente WML existente.

Primero, XHTML es un verdadero estándar de la industria sustentado tanto por la OMA y la W3C, combinando la Web fija y móvil.

Segundo, XHTML soporta una versión de hojas de estilo, la que provee una técnica para separar el contenido de la presentación.

La tabla siguiente resume esta comparación entre el WML 1.x y XHTML MP con CSS.

	WML 1.x	XHTML MP + CSS
Cuerpo de la estandarización	Estándar desarrollado por el Foro WAP	Estándar desarrollado por la W3C y adoptado por la OMA
Contenido mostrado en los dispositivos	El contenido y su disposición están definidos en el mismo documento, y deben ser separados según el dispositivo	El contenido y su disposición son definidos en documentos separados, y el contenido es interpretado de formas distintas usando diferentes hojas de estilo
Codificación de contenido	El contenido necesita ser codificado en binario	No se necesita codificación de contenido
Control de disposición del documento	Básico	Control avanzado de disposición del documento con CSS
Control del soporte de color	Imágenes de un solo color, al igual que el fondo, las fuentes y los bordes	Control total del soporte de color para las fuentes, fondo, bordes, etc., con CSS

XHTML comparado con el HTML

XHTML es estrictamente conforme a las reglas de sintaxis del XML, mientras que el HTML no. Ya que HTML no requiere una adhesión estricta a las reglas de codificación de un lenguaje XML, podría existir inconsistencia al mostrar el contenido en algunos navegadores. Los planteos del XHTML de una codificación XML aseguran una presentación consistente del contenido en un amplio rango de navegadores y dispositivos. Además asegura que todos los documentos están bien formados y son estrictamente conformes al XML, por lo que se podrían usar herramientas estándar XML para manejar, transformar, manipular y analizarlos.

La siguiente tabla muestra ejemplos de reglas XML que son requeridas en XHTML pero no en HTML.

HTML	XHTML Mobile Profile
<UL> <B><I>item 1 <LI>item 2 <A href=document.html>a link	 <li class="i"> item 1 - item 2 a link

El segundo ejemplo usa CSS puro para darle estilo a todos los textos, evitando los elementos y (y asumiendo, además, que las clases CSS i y b están definidas en aluna hoja de estilos aplicada para las propiedades *font-style:italic* y *font-weight:bold*, respectivamente).

En la práctica, XHTML MP no incluye los elementos obsoletos , , <big> y <small> para un conveniente formato de texto.

Note las siguientes diferencias entre HTML y XHTML MP:

• HTML es case-insensitive en los nombres de las etiquetas. XHTML es case-sensitive para todos los nombres de elementos y atributos. Actualmente, todos los nombres de elementos en XHTML MP son en minúscula.

• En la columna del código del ejemplo en HTML, el primer <li> no tiene etiqueta de cierre . Todos los elementos deben cerrarse en XHTML.
• Las etiquetas <I> y <B> están anidadas incorrectamente en el código HTML; anidar de esta forma está prohibido XHTML MP.

• Las comillas no encierran al valor del atributo href en el tag <a> del código HTML. El valor de los atributos en XHTML siempre debe encerrarse entre comillas dobles o simples.

• Todas las etiquetas vacías en XHTML deben ser “autocerrantes”; por ejemplo, <br/> tiene un / de cierre dentro de la etiqueta para indicar que no hay una

etiqueta </br> correspondiente. Esto se aplica a <br/>, <hr/>, <input.../>, y demás.

• Los pares valor-atributo, no pueden ser minimizados como lo son en el HTML. En HTML, cuando un atributo puede tomar un único valor, es permitido omitir este valor. Por ejemplo, en el elemento <form> de HTML del siguiente código:

```
<option value="algúnvalor" selected>
```

el atributo "selected" ha sido minimizado, y el navegador HTML entiende que el atributo selected tomó el único valor posible: *selected*. Sin embargo, XHTML requiere que se ingrese tanto el nombre del atributo como su valor aún cuando se permite un único valor:

```
<option value="algúnvalor" selected="selected">
```

La misma regla se aplica a los siguientes (y a otros) elementos:

```
<input type="radio" ... checked="checked" />
<input type="checkbox" ... checked="checked" />
<dl compact="compact">
```

Los navegadores HTML deben manejar la ambigüedad que es posible en un HTML mal formado. XHTML no admite ambigüedad porque estrictamente sigue las reglas de sintaxis de XML.

• Los campos de entrada de texto soportan la propiedad CSS *wap-input-format* para definir una máscara de entrada.

--wap-input-format

Especifica una máscara para los datos que el usuario ingresa, eliminando así la necesidad de cambiar entre modo texto o modo número.
Ya que algunas primeras versiones de navegadores XHTML aún no soportan esta propiedad pero soportan el atributos *format="..."*, se recomienda especificar ambas propiedades (la propiedad CSS --wap-input-format y el atributo *format*)[8].

[8] Ver "Formato en las entradas del usuario" del Capítulo 4, para ejemplos de máscaras válidas.

CSS, Hojas de estilo en XHTML

La mejora más sustancial que XHTML ofrece por encima de WAP 1.x y otros lenguajes de marcas es el soporte de CSS. CSS está bien documentado en libros y en Internet. En este capítulo, se discutirán las diferencias entre WCSS y el CSS1, en conjunto con algunas sugerencias para usar estilos efectivamente en dispositivos móviles.

WAP CSS o WCSS es un subconjunto propio del CSS1.

El poder de CSS provee un control preciso y la facilidad en la optimización del contenido para su presentación en cualquier dispositivo. Cada aspecto de la apariencia del documento—posición, fuentes, atributos de texto, bordes, alineamiento de márgenes y el flujo—pueden ser definidos usando estilos. Un cambio a cualquier aspecto del documento, necesita ser hecho solamente una vez. De hecho, usando una hoja de estilos compartida externamente, un cambio a cualquier aspecto en todos los documentos de un sitio se logra de una vez.

El control provisto por CSS hace más fácil reformatear el contenido para diferentes dispositivos móviles. A través de CSS, las presentaciones, estilos y navegación son fácilmente reformateados para cada navegador. En la figura 6-15 se ilustra este concepto.

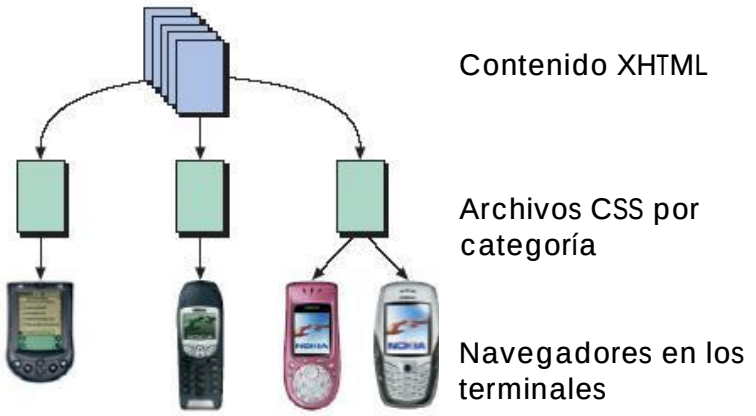


Fig.6-15: Uso de hojas de estilo diferentes para navegadores distintos

En la figura 6-16 la misma página XHTML MP se muestra en dos navegadores diferentes, cada uno con su propia hoja de estilo.

El ejemplo de la izquierda muestra el contenido en un Nokia Mobile Browser 3.0.1, y el de la derecha, muestra el mismo contenido en Internet Explorer 5.0.

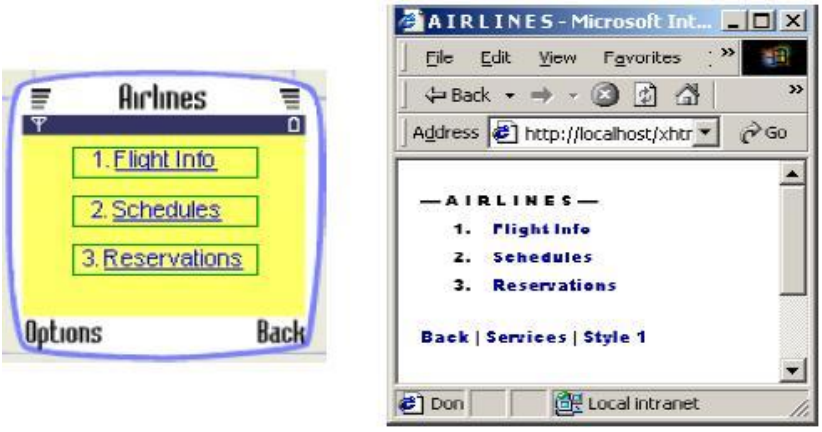


Fig.6-16: Uso de hojas de estilo diferentes para navegadores distintos

La W3C ha promovido activamente el uso de CSS en la Web para uso en navegadores móviles y de escritorio. A través del uso de CSS, los creadores de documentos pueden controlar la presentación de los documentos sin sacrificar la independencia de dispositivo o agregando nuevas etiquetas de lenguajes de marcas, como se hacía con WML 1.x.

WAP CSS (WCSS) es un subconjunto de CSS apropiado para dispositivos pequeños, definido por el Foro WAP, omitiendo propiedades que no son apropiadas para dispositivos muy pequeños.

Usando CSS, el creador del documento puede especificar la presentación completa de una aplicación Web en un solo lugar: la hoja de estilo. Si el creador necesita cambiar la presentación, en cualquier momento, el cambio es hecho una sola vez en la hoja de estilo y la modificación es inmediatamente aplicada a lo largo de todas las páginas en el sitio que hacen referencia a esa hoja de estilo.

CSS separa la presentación del contenido del documento o de la aplicación. Esto le permite a los desarrolladores la fácil creación de versiones específicas para cierto navegador del mismo contenido simplemente creando una hoja de estilo apropiada. Por ejemplo, cuando un usuario requiere un documento con hojas de estilo, el servidor Web podría

identificar el dispositivo que está haciendo la petición y devolverle la versión de hoja de estilo apropiada.

La hoja de estilo es descargada por el navegador una vez y almacenada en memoria cache, para su uso en páginas subsecuentes, lo que aumenta la velocidad que tardan las páginas completas en el sitio en ser mostradas.

Al igual que en HTML, en XHTML MP existen diferentes formas de enlazar hojas de estilo a un documento. A continuación se presenta un guía de estas diferentes técnicas.

• Hojas de estilo Externas. Las hojas de estilo externas están especificadas en el elemento <head> del documento usando una etiqueta <link>. Más de una de hoja de estilo externa puede ser usada en un documento.

La figura 6-17 muestra la etiqueta <link> en un documento XHTML que hace referencia a una hoja de estilo externa.

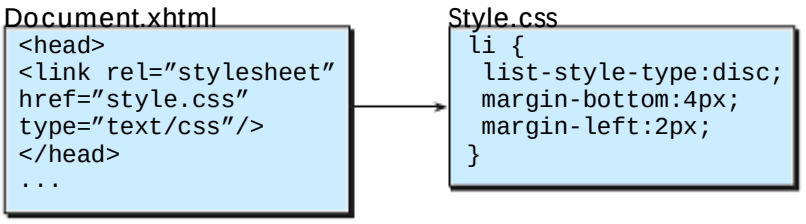


Fig.6-17 Uso de la etiqueta link para enlazar hojas de estilo

• Definición de estilos mediante el elemento <style> en el <head> de un documento. El elemento <style> puede ser usado con el elemento <head> para definir un estilo que se aplique a todo documento.

Fig.6-18 Definición de estilos mediante el elemento <style> en el <head> de un documento

En el ejemplo anterior se definió un elemento <style> en el encabezado del documento (<head>). En el mismo se define el color azul para todos los párrafos de texto.

• Estilos Inline. El atributo style puede ser usado en un elemento en cualquier parte del documento donde este sea necesario. Esto es una extensión del XHTML MP. Permite a un desarrollador de aplicaciones aplicar propiedades de estilo a elementos individuales.

Document.xhtml

```
<body>
<p style="color:red">
  red
</p>
</body>
...
```

Fig.6-19 Definición de estilos inline

Reglas para encadenamiento de elementos de estilo

Los elementos de estilo especificados en primer lugar serán sobrescritos por elementos de estilo idénticos definidos más tarde. La hoja de estilos por defecto de los navegadores tiene la menor prioridad de la jerarquía en cascada, seguida por una hoja de estilos externa, y luego por los elementos de estilo en el encabezado del documento. Los estilos inline tienen la prioridad más alta.

A continuación se muestra una lista de los estilos más comunes, ejemplos de su uso y algunas consideraciones para lograr documentos bien formados usando CSS.

Bordes

El conjunto de estilos para los bordes puede ser aplicado a todos los elementos. Los valores soportados para los estilos de borde son *solid*, *hidden*, y *none*. Aún si se indica el borde como *dashed*, *double* o *dotted*, éste se mostrará siempre como una línea sólida. El valor por defecto para el estilo de borde es *none*. El estilo de borde especifica el estilo para los cuatro bordes de un elemento.
Cuando el estilo del borde es *none* o *hidden* no se dibuja ningún borde. Cuando el estilo de borde es sólido, el

borde es dibujado como una línea continua. Se pueden declarar de uno hasta cuatro estilos de borde en una misma declaración. Si las propiedades *bottom*, *top*, *left* y *right* son declaradas individualmente, solo un valor es permitido por declaración.

Ancho de Bordes

El ancho de bordes puede ser aplicado a todos los elementos. La propiedad width de los bordes puede setearse a *medium*, *thin*, *thick* o a un número particular de píxeles. Anchos negativos no son permitidos y en ese caso, se aplica el valor por defecto (*medium*). Los valores *medium* y *thin* resultan en un borde de un píxel de ancho. La selección de un elemento con borde *medium* o *thin* resultan en un borde de dos píxeles. El valor *thick* resulta en un borde de dos píxeles. La selección de un elemento de borde *thick* resulta en un borde de tres píxeles. Se pueden declarar de uno hasta cuatro anchos de borde en una misma declaración. Si las propiedades para el ancho de los bordes *bottom*, *top*, *left* y *right* son declaradas individualmente, solo un valor es permitido por declaración.

Float

La propiedad *float* puede ser aplicada a todos los elementos excepto a los que se ha dado posición y al contenido generado dinámicamente. La propiedad *float* causa que un elemento sea movido a un lado del área de contenido del elemento contenedor. Los valores soportados por la propiedad float son *left*, *right* y *none*. *left* hace flotar el elemento a la izquierda del contenido del padre, y *right* lo hace flotar a la derecha. El valor *none*, previene que el elemento flote. Hacer que un elemento flote, causa que su propiedad *display* tome el valor *block* excepto si el display original estaba seteado a *none*. En tal caso el elemento no será mostrado, y el floating no tendrá efecto.

```
<html>
<head><title>Pájaro raro</title></head>
<body>
<p>
<img src="bird.gif" style="float:left;margin-
right:4px" alt="ave"/> Este raro ave ha sido
```

```
visto en la costa norte de la Isla Skye. Han
pasado tres años desde su ultimo avistaje.
</p>
</body>
</html>
```

En el ejemplo anterior se muestra el uso de la propiedad float en un estilo inline. La figura 6-20 ilustra el ejemplo.

Fig.6-20 Uso de la propiedad float en un estilo inline

List

El estilo list puede especificarse como sigue:

list-style-image	Define la ubicación de una imagen o bullet para usar en los ítems de la lista. Los valores que acepta son none o un URI. El URI define la ruta de la imagen, y si la imagen no puede ser cargada, se aplica el valor por defecto none, que usa una imagen también por defecto.
list-style-position	Define la ubicación del bullet en relación al contenido del item de la lista. Los valores soportados son *inside* y *outside*, con outside como valor por defecto. *inside* muestra el bullet en la misma línea que el término, y al comienzo del mismo (visualmente el bullet parece ser parte del mismo

contenido). *outside* muestra el bullet fuera del rectángulo que contiene el contenido del item de la lista.

list-style-type — Define el tipo de bullet para los items de la lista. Los valores soportados son: *disc*, *circle*, *square*, *decimal*, *upper-roman*, *lower-roman*, *lower-alpha*, *upper-alpha*, y *none*. El valor por defecto es *disc* que muestra un círculo lleno, *circle* muestra un círculo vacío, *square* muestra un cuadrado, y *decimal* muestra un número decimal con autoincremento iniciando desde uno. *upper-roman* muestra un número romano con autoincremento iniciando en I. *lower-roman* muestra un número romano con autoincremento iniciando en i. *lower-alpha* muestra una letra ASCII empezando en a y *upper-alpha* muestra un carácter ASCII comenzando en A. Si el valor de la propiedad es *none*, no se muestra el bullet.

El siguiente es un ejemplo del uso del estilo *list*. Es un extracto de código que incluye WCSS, y que muestra una lista ordenada con bullets de tipo *upper-roman* y posición *outside*; y una lista no ordenada con bullets de tipo *square*.

Fig.6-21 <ol> y <ul> usando hojas de estilo.

```
ol { list-style-type:upper-roman ;list-style-
position:outside }
ul { list-style-type:square }
h3 { color:white }
.b { font-weight:bold }
.u { text-decoration:underline }
```

Cód.6-14 Estilo de listas

Padding

La propiedad padding puede ser aplicada a todos los elementos que se pueden mostrar. Los valores soportados son: length, inherit, y percentage. Para percentage, el padding es calculado respecto del ancho del bloque que contiene al elemento. El valor por defecto es cero. Los cuatro lados de un elemento pueden ser rellenados a la vez usando la propiedad padding, o pueden ser rellenados individualmente definiendo los valores de padding top, right, bottom, y left. Un valor negativo no es permitido; en tal caso, se aplica el valor por defecto. Las propiedades *length* y *percentage* podrían ser usadas en conjunto.

Text

Los estilos de texto pueden ser especificados como sigue:

text-align — La propiedad text-align se aplica a todos los niveles de elementos y establece la forma en que un bloque de texto es alineado. Los valores soportados para esta propiedad son *left*, *right*, y *center*. El valor por defecto es *left*. El valor *left* alinea el bloque a la izquierda. El valor *right*, hacia la derecha y el valor *center* lo centra.

text-decoration — La propiedad text-decoration se aplica a todos los elementos incluso a los inline. Los valores soportados para esta propiedad son *none*, *underline*, y *blink*. El valor por defecto es *none*. El valor *underline* causa que los elementos sean subrayados con una línea simple. El valor *blink* causa que el elemento parpadee. El valor de parpadeo es de un destello por segundo.

```
.p { text-decoration:blink }
```

El anterior es un ejemplo de decoración de textos usando WCSS en el que cualquier texto dentro de un párrafo <p> parpadeará.

Migrando desde el WML1.x a XHTML MP

Las especificaciones de los lenguajes de marcas y los protocolos de transporte pueden evolucionar independientemente. Para asegurar una migración "suave", y para soportar la compatibilidad hacia atrás, debe haber una transición gradual desde la tecnología WML al HXTML MP. Los navegadores duales garantizan esta transición.

La figura 6-22 muestra la evolución independiente del WAP 2.0 y las capas de transporte que le dan soporte a esta tecnología.

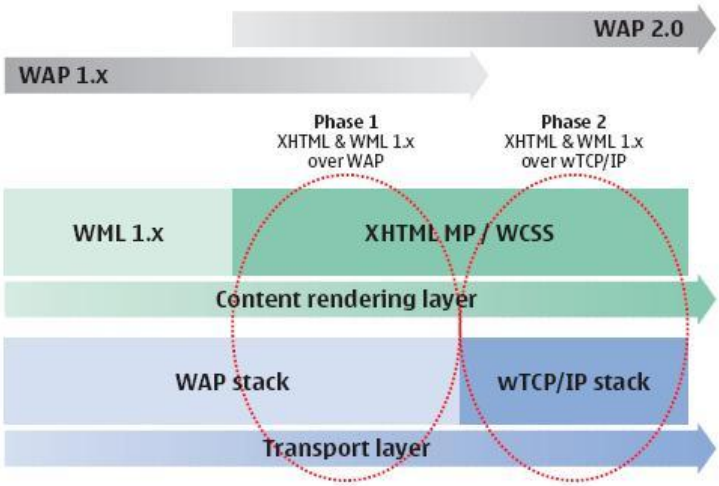


Fig.6-22 Evolución independiente del WAP 2.0 y las capas de transporte

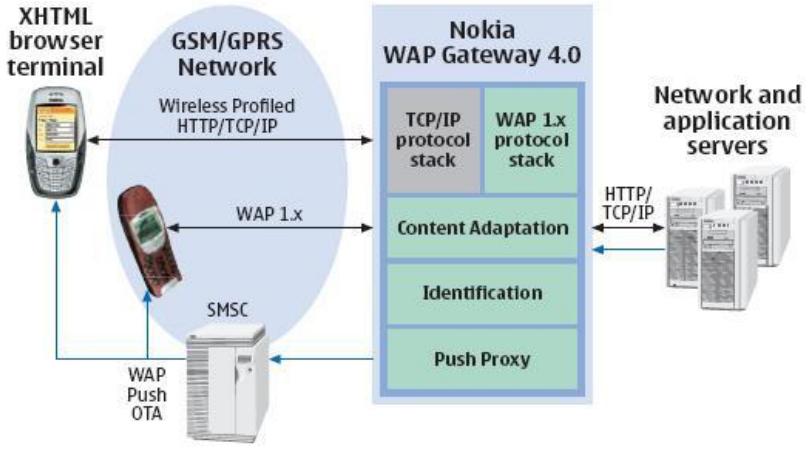


Fig.6-23 Gateway WAP

En la gama de lenguajes de marcas, la tecnología XHTML MP y WCSS son la clave del nuevo desarrollo de la especificación. XHTML MP introduce gráficos full color y control sofisticado de la disposición de contenido accedido por un navegador móvil, tal como se trató en éste capítulo.

Los navegadores duales, son necesarios para asegurar que los dispositivos con soporte XHTML MP puedan acceder contenido existente WML.

Si el dispositivo no está equipado con un browser dual, entonces el Gateway WAP debería ser capaz de manejar la conversión de contenido. La siguiente figura muestra un Gateway WAP (como el mostrado en la figura 6-23) manejando los enlaces entre un servidor de contenidos y un dispositivo móvil, a partir de los lenguajes que éste soporta. En algunos casos, el enlace será directo, mientras que en otros, existirá una conversión de contenidos de por medio.

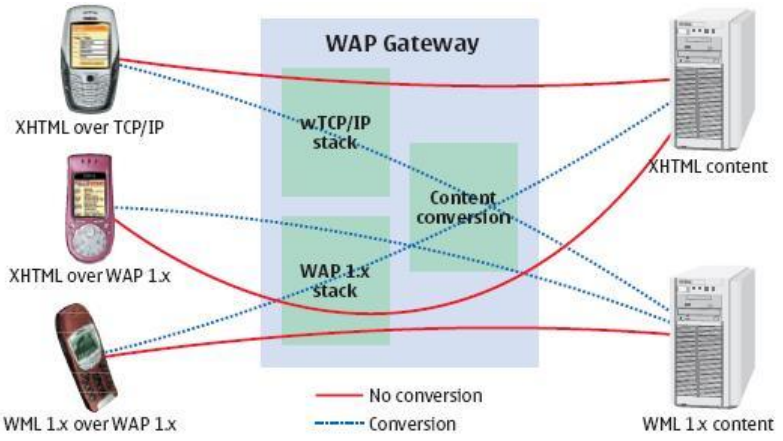


Fig.6-24 Manejo de conversiones XHTML/WML en un Gateway WAP

En la capa de transporte, la industria móvil migraría desde el uso de la pila de protocolos WAP al Wireless Profiled TCP/IP (wTCP/IP) como una forma de mejorar la eficiencia del transporte y la velocidad de grandes archivos multimedia, así como traer verdadera seguridad punto a punto. wTCP/IP es compatible con el estándar TCP/IP, pero incluye configuraciones optimizadas para aumentar la performance sobre los enlaces inalámbricos.

La performance se ve incrementada en gran proporción cuando ambos, cliente y servidor implementan las optimizaciones inalámbricas perfiladas.

Hasta el WAP 2.0, el WML era usado para la creación de contenido para aplicaciones móviles. Desde el punto de vista de los desarrolladores, es importante crear aplicaciones que no estén atadas a ningún lenguaje específico, anticipándose a un eventual cambio desde el WML al XHTML.

La tabla siguiente describe qué necesita hacer un desarrollador WML cuando comienza a desarrollar contenido para XHTML MP/CSS.

Se consideran tres casos diferentes:

a. Un desarrollador con experiencia en crear contenido nuevo en WML (el desarrollador puede usar herramientas y editores primitivos).

b. Un desarrollador con experiencia en modificar contenido existente en WML para adaptarlo a XHTML.

c. Un desarrollador con nivel en desarrollo de aplicaciones del lado del servidor y capaz de modificar una aplicación WML/XML para soportar XHTML.

	a. Nuevo contenido XHTML	b. WML estático a XHTML	c. WML dinámico a XHTML
Herramientas	La misma	La misma	La misma
Modificaciones al código	Hacer código dinámico, los datos no se alteran	El código que incluye datos debe ser cambiado manualmente	La salida debe ser cambiada, el resto, sin cambios
Cambios en UI	Con CSS	El código se debe cambiar manualmente	Con CSS

En el primer caso, al crear nuevo código, los desarrolladores pueden usar las mismas herramientas para crear código XHTML MP como hacían antes con el código WML. Es recomendado crear código dinámico XHTML porque es más fácil de presentar el mismo contenido en varios dispositivos usando hojas de estilo. El soporte CSS en XHTML MP hace más fácil la optimización de las presentaciones para diferentes interfaces de usuario.

En el segundo caso, al cambiar contenido estático WML a XHTML, los desarrolladores pueden usar las mismas herramientas para crear código XHTML MP como hacían antes con el código WML. El cambio de contenido estático WML a XHTML debe ser hecho manualmente. Para hacer cambios en la UI, el código WML debe ser cambiado manualmente al XHTML MP, que permite el control de la disposición y la presentación mediante CSS.

En el tercer caso, al cambiar contenido dinámico WML a XHTML, los desarrolladores pueden usar las mismas herramientas para crear código XHTML MP como hacían antes con el código WML. En este caso, los datos ya habrán sido almacenados en alguna base de datos o, por ejemplo, en formato XML, que puede ser fácilmente migrado a XHTML MP. Con código dinámico es más fácil definir UIs de diferente estilo usando CSS.

El DTD

Validar XML simplemente significa que el documento usa el lenguaje correctamente—que incluye solo etiquetas definidas, y que cada etiqueta es usada en el contexto correcto. Por ejemplo, en XHTML una etiqueta <row> solo es válida dentro de una tabla (<table>...</table>), ya que una fila no tiene sentido fuera de una tabla. El *Document Type Definition* (DTD) define las reglas para el XHTML (o cualquier lenguaje XML válido).

El DTD (o la dirección en donde puede ser accedido) aparece antes de la cabecera (<head>) de cada documento XHTML. La cabecera forma la sección inicial de un documento Web y contiene información que no es parte del contenido actual de la página. También contiene controles para el documento en general.

De acuerdo a la W3C, el DTD facilita las producciones y presentaciones libres de inconsistencia. El DTD identifica qué significa cada etiqueta y especifica cómo deben ser tratadas. En breve, provee las reglas usadas por los editores y otras herramientas de creación y transformación para validar documentos, lo que asegura que un documento puede ser mostrado en cualquier navegador XHTML. Esto es particularmente importante para la Internet móvil, donde los dispositivos ofrecen un amplio rango de formatos de pantalla y capacidades.

Siguiendo estrictamente las reglas XHTML, la memoria cache en los navegadores móviles puede ser reducida, permitiendo menor uso de memoria y mayor performance en la eficiencia, lo que es esencial en los dispositivos móviles. Si se mira al tamaño de los navegadores Web estándar de escritorio, mucha de la complejidad tiene que ver con el manejo de las ambigüedades del HTML.

En la sección DTD del capítulo 4 se puede encontrar más información sobre los DTDs. En el Apéndice B se incluye un conjunto de DTDs estándar para desarrollar aplicaciones WAP, tanto en WML como en XHTML MP.

Protocolos de Transmisión

La mayoría de los sitios móviles están configurados para reconocer y procesar contenido escrito en WML 1.x. WML requiere un Gateway WAP para codificar el contenido WML y WMLScript antes de ser enviado por el aire y hacia el terminal del cliente. El contenido se codifica en una forma binaria llamada WBXML.

A diferencia del contenido WML, el contenido XHTML no es codificado por el gateway, sino que es pasado sin cambios, al igual que el HTML, a través de un servidor Web. Esto se ilustra en la figura 6-25.

Fig.6-25 Transmisión de contenido WML

Sin embargo, ya que el contenido WML es aún útil y lo será por un tiempo, los navegadores que soportan el estándar WAP 2.0 puede mostrar tanto contenido XHTML MP como WML 1.x, como también se muestra en la figura 6-26. Estos navegadores se denominan navegadores duales.

Fig.6-26: Los navegadores duales soportan tanto WML como XHTML

Cuestiones acerca de los navegadores

Navegadores Web estándar

XHTML MP funciona en la mayoría de los navegadores Web y los más recientes dispositivos móviles; usar, previsualizar, y testear el mismo contenido en un amplio rango de navegadores es fácil. Hay sin embargo, unas pocas diferencias pero que no hacen a mayores complicaciones.

XHTML MP no soporta los atributos *cellpadding* o *cellspacing* en el elemento <table>, lo que significa que no hay un valor por defecto en los navegadores WAP 2.0 para estos atributos. Para hacer que las tablas luzcan igual tanto en un navegador Web y en un navegador WAP, se debe definir a las tablas como sigue:

```
<table cellpadding="0" cellspacing="0">
```

Los navegadores WAP ignoran estos atributos, por lo que es aceptable usarlos incondicionalmente en documentos que estén destinados para Web y WAP.

Si se escribe código XHTML tanto para un navegador Web y un navegador móvil, el siguiente podría ser un truco útil. Mientras que la mayoría de los elementos HTML vienen en pares, esto es, tienen etiquetas de apertura y cierre separadas (por ejemplo, o), los elementos <hr>, <img>, <br>, <input> (además de otros pocos), no. Por esta razón, no se puede cerrar estos elementos usando etiquetas de cierre como </img> o </br>, lo que es inválido tanto para un parser HTML o XHTML. De ahí, que para complacer a ambos parsers, simplemente se puede insertar un espacio en blanco antes de el caracter / en un elemento de etiqueta autocerrante; por ejemplo, <br /> en vez de <br/>. Un intérprete HTML ignora el espacio en blanco y el caracter / siguiente, mientras que un intérprete XHTML removerá el espacio e interpretará el elemento correctamente como <br/>.

Internet Explorer

XHTML MP se muestra bien en Internet Explorer. Los test que se han realizado son sobre la versión 5 en adelante. El código que se muestra al final de esta sección se puede probar tanto en un browser WAP 2.0 como en Internet Explorer.

El siguiente estilo hace que IE muestre código XHTML de manera similar al Nokia Mobile Browser 3.0.1:

```
td, body {font-family:arial; font-size:x-small}
```

Sin embargo, el tamaño de fuente *x-small* podrá mostrarse muy pequeño en algunos dispositivos, así que el envío de este estilo debería estar condicionado a requests que lleguen desde navegadores Web estándar.

Una manera de detectar si el request llega desde un navegador Web estándar, es leer el valor de HTTP_USER_AGENT en busca de la subcadena "Mozilla". A continuación se muestra un ejemplo de esto, usando Active Server Pages (ASP) con Microsoft Internet Information Server:

```
<style>
<%
If
InStr(Request.ServerVariables("HTTP_USER_AGENT"),
"Mozilla") Then
Response.Write("td, body {font-family:arial;
font-size:x-small}")
End If
%>
</style>
```

Netscape Navigator

Las versiones de Netscape Navigator anteriores a la 6.0 tenían muchos problemas con su implementación de CSS. Para lograr que una fuente tenga una apariencia razonable, se debe definir el siguiente estilo:

```
td, body, p {font-family:arial; font-size:small}
```

Los bordes de tabla, colores de fondo y otros estilos tienen también muchos errores. Se recomienda usar Internet Explorer o Netscape Navigator 6+ para testear estilos.

Nokia Mobile Browser 3.0.1

Nokia Mobile Browser 3.0.1 presenta las siguientes características:

• Soporte completo del WML 1.3 excepto el elemento <fieldset>.

• Soporte completo de los elementos y atributos del XHTML MP.

• Soporte completo WAP CSS, excepto la propiedad opcional
-wap-accesskey (en su lugar, se pude usar el atributo accesskey: <a accesskey="1" href="...">)

• En Nokia Mobile Browser, todos los enlaces y botones Submit que contienen un atributo accesskey son automáticamente agregados al menú Opciones.

Los elementos <base> y <fieldset> de XHTML MP no están soportados en Nokia Mobile Browser 3.0.1.

Herramientas de validación de código

Existen varias herramientas y utilidades para la validación de código bien formado disponibles en Internet. Algunas de ellas son:

W3C XHTML Validation Tool

LA W3C provee un servicio de validación gratuito para código XHTML [XHTMLValidator]: se envía el documento XHTML, usualmente proveyendo su URL, y el servicio retorna una notificación de su validez. Si el documento es inválido, se retornan los errores. Esta herramienta está disponible en: http://www.validator.w3c.org/.

W3C Tidy

HTML [TIDY] es una utilidad para convertir código HTML en código XHTML bien formado. Hay varias versiones de GUI disponibles, o su código fuente en UNIX a modo de filtro para procesar archivos HTML.

CSS Validator

La W3C tiene una herramienta de validación [CSSValidator] que puede verificar si un CSS es correcto. Aún no tiene capacidades de verificar si el código está en conformidad con las restricciones del WCSS.

Configurando un servidor para brindar contenido XHTML/WML

Tipos MIME

Se debe configurar un servidor Web para que reconozca el tipo MIME para cada nuevo tipo de archivo.
Es típicamente un parámetro de configuración o una tabla.
Las dos tablas siguientes muestran los tipos MIME que se deben configurar para los contenidos WML y XHTML respectivamente.

Tipos MIME para contenido WML 1.x

Tipo MIME	Descripción	Extensión
text/vnd.wap.wml	Código WML	.wml
application/vnd.wap.wmlc	WML Compilado	.wmlc
text/vnd.wap.wmls	Código WMLScript	.wmls
application/vnd.wap.wmlsc	WMLScript Compilado	.wmlsc
image/vnd.wap.wbmp	Imagen WBMP	.wbmp

Tipos MIME para contenido XHTML

Tipo MIME	Descripción	Extensión
text/html application/xhtml+xml application/vnd.wap.xhtml+xml	Código XHTML	.xhtml, .htm, .html
image/gif	Imagen GIF	.gif
text/css	Código CSS	.css

Note que una de las ventajas de usar el tipo MIME `text/html` para el código XHTML es que estos archivos podrán ser cargados por navegadores Web estándar. Los navegadores Web de PCs de escritorio, podrían no reconocer los nuevos tipos MIME `application/*`.

Capítulo 7. Construyendo Aplicaciones WAP

Reglas de Usabilidad generales
Organizando el sitio WAP
Displays grandes y displays chicos

Principios de navegación

Dos ejemplos de aplicaciones WAP

Capítulo 7. Construyendo Aplicaciones WAP

Reglas de Usabilidad generales

Cuando estamos decidiendo que información incluir en las diferentes aplicaciones en un dispositivo móvil, tenemos que pensar que información va a ser relevante en las situaciones en que un dispositivo móvil va a ser usado. Debido a la practicidad de una pantalla pequeña, el usuario podría primeramente usar el dispositivo móvil en donde no haya acceso a Internet desde una PC.

Creemos que los usuarios de aplicaciones móviles, en principio, están interesados en información breve y rápida. Por ejemplo, un acceso rápido a la agenda de vuelos, podría ser relevante para un usuario que está camino al aeropuerto y necesita chequear la partida de un vuelo. Similarmente, varios flashes de pequeñas noticias son más relevantes que artículos más largos. El acceso rápido a la información del clima, podría ser usado en una situación de un viajante que desea conocer la temperatura de su destino. Sin embargo, es menos común navegar en Internet en un teléfono móvil que hacerlo en la PC del hogar o la oficina.

En cuanto a lo que respecta a la tecnología, debemos tener presente que los *roundtrips* en la red agregan un mínimo de dos segundos de demora en el tiempo de respuesta. De ahí que el número de *roundtrips* debe ser minimizado. El tiempo de respuesta de una primera página tiene un fuerte impacto sobre la percepción del usuario, por lo que la optimización vale la pena.

Los tiempos muertos (*latency*) entre el WAP Gateway y el servidor de contenido también deberían ser minimizados. Cada consulta viaja por este enlace dos veces, una como *request* y otra como *response*.

Este WAP Gateway debería ser ubicado cerca de la red inalámbrica (*GPRS, Global Packet Radio Station*). "Cerca" en este contexto hace referencia tanto al *latency* como a la probabilidad de pérdida de paquetes. Los mensajes perdidos causan retrasos adicionales debido a la retransmisión HTTP.

Organizando el sitio WAP

• Páginas de Bienvenida. No es recomendado iniciar el sitio usando una página de inicio, quizás mostrando un logo, que no sirva a otro propósito que dar la bienvenida al visitante. Es mejor si el usuario puede ir directamente al servicio.

• Login. La entrada de información de login puede consumir más tiempo en un dispositivo móvil. Esta información debería ser pedida solo una vez por sesión, y no cada vez que se inicia la navegación desde la página principal. El chequeo *case-sensitive* de nombres de usuario y passwords debería evitarse.

• Detección del Browser. Es posible detectar tanto el navegador como el lenguaje y proporcionar el contenido correcto transparentemente, sin la interacción del usuario. Los navegadores de los dispositivos móviles envían un campo *user-agent* en el encabezado WSP hacia el WAP Gateway y este puede ser usado para definir el tipo de navegador y entonces construir la lógica en el servidor para dar un contenido más apropiado.

• Optimizando el tamaño. Las páginas interactivas deberían ser pequeñas, las páginas informativas, largas. Demasiadas páginas cortas y muchas descargas es peor que tener información cercana y enlazada en una página más larga, pero especialmente las páginas interactivas con formularios no deberían ser demasiado largas, ya que los usuarios comienzan a sentir que pierden el control si el formulario toma más de dos pantallas completas. Las filas vacías pueden causar confusión, ya que el usuario podría creer que no hay más contenido debajo.
Si la página contiene una historia o instrucciones, la historia completa debería estar en una página. Esta podría contener subtítulos-enlaces que llevaran al usuario a ciertos puntos en la página, ayudando a la navegación de páginas informativas largas.
Es difícil decir cuántos links pueden proveerse en una página. Si los links claramente están relacionados y son fáciles de navegar (una línea por link, en orden alfabético o lógico), es mejor dar 30 links en una página que dar 5, en seis páginas.
En lo que respecta al tiempo total de descarga, algunos estudios le ponen un límite máximo para un retraso aceptable en un navegador de PC, de 10 a 15 segundos,

incluyendo todas las imágenes. Es altamente recomendado tener un tiempo de respuesta (excluyendo el tiempo de conexión) de menos de 10 segundos en el ambiente móvil.

• La Primera Pantalla. La primera pantalla es la más importante. Todos los links navegacionales, campos de búsqueda, pantallas de login, y la mayor parte de la información debería residir aquí. El usuario puede continuar navegando antes de que la página esté totalmente cargada, incluso si aún no tiene los *scrollers*.
Desperdiciar el tope de la página para banners de publicidad o gráficos no informativos debe ser evitado. Es mejor colocar las publicidades en los márgenes derecho o izquierdo.

• Imágenes y Tablas. La descarga de imágenes toma tiempo, y varios usuarios podrían elegir desactivar esta descarga de imágenes para incrementar la velocidad. Los autores deberían tratar de optimizar el tamaño de las imágenes. Si hay grandes imágenes en el sitio, los autores podrían considerar usar vistas en miniatura para el índice de imágenes. El color debería ser probado en varios displays, por cuestiones de visibilidad. Usualmente la misma imagen no puede ser usada en pantallas color y monocromáticas.
Se debería dar un texto alternativo a las imágenes que complementen la información (usando el atributo *alt* del tag <img>). Un texto alternativo nulo podría ser usado para las que no complementen información, o sirvan a propósitos decorativos solamente.
El ancho o el alto de la tabla no debería exceder el tamaño de la pantalla, ya que la vista desplazable hace a la tabla menos legible (por ejemplo, el título de las columnas no se muestran cuando se desplaza la vista hacia abajo).

Debido a las restricciones de las pantallas, en la mayoría de los dispositivos móviles, el uso de tablas no es recomendado, la figura 7-1 muestra la desventaja del uso de tablas en una pantalla pequeña: Una tabla de 2x2 es mostrada como una columna de 4 filas.

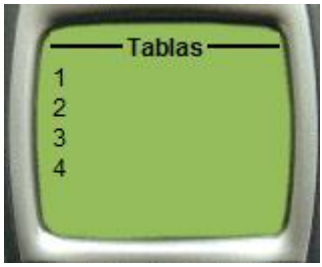


Fig.7-1 Visualización de tablas

• Navegación eficiente. Los proveedores de servicio deben responsabilizarse por la navegación como un todo. Se deberían proveer junto con el servicio, métodos de navegación consistentes. Cada página debería contener enlaces a la página principal del sitio, y a posibles subsecciones de la página principal.
La función "atrás" (*Back*) también podría se provista como un enlace en la página. Sin embargo la función atrás no debería usarse en páginas que llevarían al usuario hacia, por ejemplo, la confirmación de una orden de compra.
Si se usan las teclas de acceso rápido, se podrían definir consistentemente atajos para los links principales, de este modo el "1" podría llevar al usuario a la página principal del servicio.

• Búsqueda. Una forma para hacer más accesible la información del sitio es organizarla en una jerarquía, o conduciendo a una búsqueda. En el caso ideal, ambos deberían ser provistos. Los árboles jerárquicos ayudan a usuarios de poca experiencia a ver la estructura del servicio con mayor claridad, pero podría ser demasiado lento para usuarios con experiencia.
Los resultados de la búsqueda deberían ser optimizados para incluir unos pocos items, y si la búsqueda produce demasiados resultados, se le debería dar al usuario la posibilidad de ajustar el criterio de búsqueda.

• Títulos y Etiquetas de Elementos. El título describe el contenido de la pantalla, por lo que es recomendable usarlo en WML. Los títulos ayudan al usuario a navegar en la aplicación porque actúan como un recordatorio de su ubicación en el sitio. Podría ser una buena idea empezar el título con el nombre del servicio y mantener corto el largo total de este. Un item previamente seleccionado por el usuario debería determinar el texto del encabezado. Por ejemplo, el título "Favoritos" (*Bookmarks)*

le indica al usuario que la pantalla contiene una lista de enlaces favoritos en la aplicación y que la opción previamente elegida fue Favoritos.

Las fuentes proporcionales son usadas en los textos de encabezado, y si el encabezado es demasiado largo, es cortado automáticamente. Este tipo de títulos es mejor que las abreviaciones, ya que el usuario podría confundirse con abreviaturas no familiares.
Aunque se recomiendan palabras cortas para etiquetas de elementos, las abreviaciones que no son conocidas por el grupo de usuarios, no deberían ser usadas. La misma etiqueta debería siempre ser usada para la misma cosa; especialmente con las etiquetas de las funciones Borrar, Eliminar, Limpiar y Destruir.

• Mensajes de Error y Confirmaciones. Especialmente, en los servicios móviles, es irritante que se le presente un mensaje de error a un usuario y se le pida corregir una cierta entrada. Los servicios deberían ser mucho más flexibles para interpretar entradas, por ejemplo, aceptar varios formatos para las fechas. Si hay una necesidad real de mostrar un mensaje de error, debería ser de asistencia y lo más cordial posible. Los mensajes de error deberían explicar por que no se pudo ejecutar la acción, y proveer una forma fácil al usuario de corregir esta situación.

Luego de que el usuario haya elegido una acción de destrucción de información o transferencia de dinero, es esencial proveer una página de confirmación antes de ejecutar la acción. Por ejemplo, cuando se borra un item, es mejor pedir confirmación primero, y luego informar al usuario que la eliminación se ejecutó satisfactoriamente.

• El uso de links. Debido a que las características de los enlaces son conocidas de Internet, los usuarios intuitivamente sabrán que cliqueando un enlace, tendrán acceso a más información. Los links son una herramienta poderosa para enlazar porciones diferentes de información y prevenir largas páginas de texto.
Se recomienda el uso de los links teniendo en mente que las jerarquías profundas innecesarias deberían ser evitadas.

• Terminología consistente. Es importante mantener una terminología consistente. Una terminología inconsistente a menudo confunde al usuario innecesariamente y hace a una navegación compleja. Una terminología consistente

hace posible pensar en términos de pantallas, y reconocer cada una cuando navegamos en ella. El siguiente es un ejemplo del uso de terminología consistente:

Fig.7-2 Terminología consistente

En la figura 7-3 se muestra una inconsistencia entre el link *"Home"* y la página de inicio de la CNN.

Fig.7-3 Terminología inconsistente

• Abreviaturas y palabras largas. La longitud total de las palabras puede causar problemas en pantallas pequeñas. De ahí que se recomienda especial atención en la elección de palabras cortas en el display.
Tratar de usar abreviaturas es posible, pero el significado de abreviaturas no familiares podría ser difícil de comprender para el usuario.

• Textos de ayuda. El contexto de los textos de ayuda debería ser especialmente considerado. Agregar textos de ayuda solo si el contenido del texto es relevante para alguna situación particular de uso. El texto debería se relevante para la página que se trata de explicar.
Es recomendado no tener cualquier item de ayuda en la lista de opciones porque el contenido de esta ayuda puede ser muy general para la aplicación y no podrá ayudar al usuario. Por ejemplo, se debería evitar tener textos de ayuda que expliquen la forma en que se usa un

botón, porque puede asumirse que el usuario lo sabía de antemano cuando se conectó a la aplicación. Sin embargo, esto puede se bueno para explicar algo relevante para alguna página particular, visible al usuario como un item Ayuda ubicado en la página actual como un link.

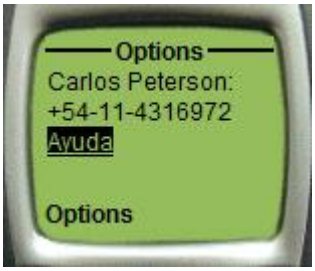


Fig.7-4 Visualización de un item de ayuda como un link

• Test de usabilidad. Siempre es bueno realizar un test a la aplicación. Las personas que no hayan estado involucradas en el diseño o desarrollo de una aplicación tienden a encontrar potenciales problemas de usabilidad a veces no tan obvios para aquellos que conocen el diseño de memoria. Los test de usabilidad deben ser realizados tan pronto como sea posible en el proceso de desarrollo. Cualquier necesidad de cambio resultante de los tests puede ser implementada dentro de la agenda de desarrollo. Personas representativas de los futuros usuarios finales, podrían encargarse de los test, y estos deberían ser conducidos, al menos en una escala pequeña si la agenda no permite testeos más extensos.

Displays grandes y displays chicos

Las pequeñas pantallas en los dispositivos móviles, limitan la mayoría de la información que puede ser mostrada en cada vista en comparación con la PC. Las computadoras de escritorio proveen más espacio para animaciones e información extra. En los teléfonos celulares, se debe poner especial atención y priorizar la información que se muestra por pantalla. Como se dijo antes, pensar en que información es relevante para el usuario.

• El largo de los textos en las pantallas pequeñas. Se debe poner especial atención en el largo de los textos en aplicaciones para dispositivos móviles. Cuando se lee un

texto, sólo cuatro líneas pueden ser mostradas de una vez. Cuando se desplaza un texto, existe siempre el riesgo de que el usuario pierda el rastro en la página de texto, si la página es muy larga. El botón de scroll hace que el desplazamiento del texto sea más simple, pero las pantallas pequeñas limitan la cantidad de texto con la que un usuario se sentiría cómodo. Se recomienda que el texto esté limitado en largo a un máximo aproximado de cuatro o cinco pantallas (=16 o 20 líneas).

• El uso de bucles. El uso de bucles se refiere a la forma en la que el último item en una lista es automáticamente seguido del primero en la misma lista. Si el usuario se desplaza hacia abajo cuando el último item se está mostrando al fondo de la pantalla, el próximo item será automáticamente el primero.
Mientras que las cartas con solamente texto, o cartas con texto y algunos links o solamente links, y las listas de selección simple o múltiple no proveen un mecanismo de bucle, las listas de opción (*option lists*) sí producen bucles automáticamente.

• Uso de gráficos. La mayoría de los dispositivos móviles soportan el uso de gráficos, desde algunos muy simples, monocromáticos en formato WBMP, hasta los formatos comúnmente usados en la Web. El máximo tamaño del gráfico depende del área de la pantalla, generalmente pequeña.
Las imágenes pueden ser usadas por ejemplo para mostrar un logo corporativo cuando se inicia el servicio. Las imágenes que caben dentro del display son centradas automáticamente, mientras que las que exceden en ancho el tamaño del display son alineadas a la izquierda y se trunca el excedente. Así mismo, las que exceden en alto el tamaño del display son alineadas arriba y la parte inferior se trunca.
No puede haber texto a los lados de la imagen, la imagen siempre comenzará su propia línea, como se muestra en la siguiente figura:

Fig.7-5 Uso de gráficos

El usuario puede desplazar una imagen línea por línea, pero cuando la imagen quede ubicada al tope de la pantalla, desaparecerá por completo en el próximo desplazamiento. Esto se debe a que la imagen es tratada como una única línea. Las imágenes siguientes ilustran esta situación:

Fig.7-6 Desplazando imágenes

• Utilización del espacio de pantalla. Deben evitarse los espacios excesivos entre los items, ya que esto disminuye la sensación de fácil navegación que percibe el usuario, y no mejora la usabilidad de la aplicación.

Principios de navegación

Es una regla general que los usuarios deban tener una manera fácil de retroceder en el historial de la navegación. El usuario también debe poder cancelar comandos. Esto es particularmente relevante en situaciones en donde el usuario es meramente curioso por explorar la aplicación, y necesita una forma de escape fácil. La satisfacción con la aplicación depende sobre todo en la sensación de facilidad de navegación que percibe el usuario.

Otro principio importante de la navegación es siempre dar al usuario más de una forma de realizar acciones importantes. En otras palabras, el usuario puede tanto seleccionar un item en una página mediante un botón para tal fin o activarlo desde una lista de links que el mismo navegador provee, como es el caso más general de los teléfonos celulares.

La jerarquía de una aplicación determina la estructura de navegación. Mientras más capas sean agregadas o percibidas por el usuario, más compleja se vuelve la navegación. Existe un riesgo de que el usuario pierda foco del punto inicial y experimente una pérdida de control. A la mayoría de los usuarios les gusta tener un lugar que les es familiar y al que reconozcan mientras naveguen por la aplicación. Cuando más profunda se vuelva la navegación y más se aleje al usuario de ese lugar familiar, más se preocupará este en como volver a su lugar seguro. En la siguiente secuencia de imágenes se muestran dos ejemplos diferentes de estructurar la navegación para el mismo contenido. El primero es recomendado.

Fig.7-7 Una jerarquía profunda de links se evita permitiéndole al usuario desplazar las vistas de una página de resultados

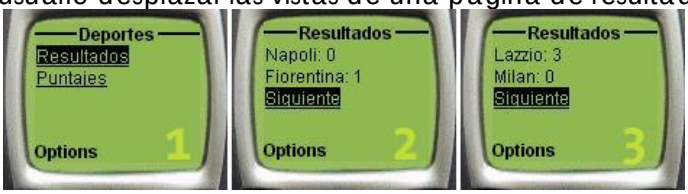


Fig.7-8 Una jerarquía profunda de links podría causar que un usuario se perdiera

Dos ejemplos de aplicación WAP

• El siguiente es un ejemplo de un sitio WAP que tiene como propósito el envío de email. La aplicación está construida en PHP para generar contenido dinámico WML.

Mazo 1 (email1.wml):
Envía el encabezado propio al browser WAP.
Pregunta al usuario la casilla de Email destino.
Pregunta al usuario el tema.
Pregunta al usuario el mensaje.
Pasa información al segundo mazo, titulado email2.wml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML
1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
    <card id="email1" title="Email Example">
     <p>
      <do type="accept">
       <go href="codi7-2.php" method="post">
        <postfield name="Email" value="$email"/>
        <postfield name="Message"
value="$message"/>
        <postfield name="Subject"
value="$subject"/></go>
      </do>
      Dirección:
      <input title="Email" name="email"/>
      Tema:
      <input title="Subject" name="subject"/>
      Mensaje:
      <input title="Message" name="message"/>
     </p>
    </card>
</wml>
```

Cód.7-1 email1.wml

La siguiente secuencia de imágenes ilustra el ejemplo anterior.

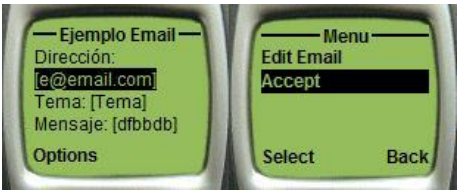


Fig.7-9 Visualización del ejemplo de la aplicación WAP (email1.wml)

Mazo 2 (email2.wml):
Envía el mensaje al recipiente.
Informa al usuario si el mensaje fue enviado o no.

```
<?php
$Email=$_REQUEST["Email"];
$Subject=$_REQUEST["Subject"];
$Message=$_REQUEST["Message"];
header("Content-type: text/vnd.wap.wml");
print "<?xml version=\"1.0\"?>";
echo "<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML
1.1//EN\"" . " \"http://www.wapforum.org
/DTD/wml_1.1.xml\">";
?>
<wml>
<card id="email2" title="Send Email">
        <p>
        <?php
        if (mail($Email,$Subject,$Message,
"From:PHP-enabled WAP site")) :
            print "El mensaje ha sido enviado a
$Email";
        else :
            print "No se pudo enviar el mensaje a
$Email";
        endif;
        ?>
        </p>
</card>
</wml>
```

Cód.7-2 email2.wml

En la figura 7-10 se muestra el resultado del código de email2.wml.

Fig.7-10 Pantalla de confirmación de envío de email (email2.wml)

• A continuación se presenta otro ejemplo que ilustra una aplicación WML con acceso a una base de datos. La aplicación también está construida en PHP, y la base de datos es accedida a través de MySQL.

El ejemplo involucra la extracción de información desde una base de datos MySQL para mostrar en una carta WML. Supóngase que se quiere mostrar los resultados de los partidos de una serie de fútbol.

Se podría tener una base de datos creada apropiadamente como sigue:

```
mysql>create table soccer_scores (
    >team1 char(20),
    >score1 int,
    >team2 char(20),
    >score2 int );
```

Y con los datos como se muestra a continuación, en una tabla:

Equipo1	Goles1	Equipo2	Goles2
Francia	2	Holanda	3
Inglaterra	2	Rumania	3
Italia	2	Suiza	1
Yugoslavia	3	España	4

Se podría crear un simple script capaz de obtener esta información y luego mostrarla en un navegador WAP como se muestra debajo:

Fig.7-11 Resultado de una consulta sobre la tabla anterior

Descripción del código:

Se envía el header adecuado al browser WAP.
Se comienza un nuevo mazo WML.
Se conecta al servidor MySQL (“www.zend.com”)
Se selecciona la base de datos “sports”

Se selecciona información de la tabla “soccer_scores”
Si existen datos en la tabla, la salida se produce usando el formato Equipo1 Goles1 Equipo2 Goles2
Si los datos no existen, se muestra un mensaje apropiado

```
<?php
header("Content-type: text/vnd.wap.wml");
echo "<?xml version=\"1.0\"?>";
echo "<!DOCTYPE wml PUBLIC \"-
//WAPFORUM//DTD WML 1.1//EN\"" . " \"http://www.w
apforum.org/DTD/wml_1.1.xml\">";
?>
<wml>
    <card id="card1" title="Soccer Results">
     <p>
     <?php
     $scores = mysql_pconnect("www.zend.com", "sp
orts","scores");
     $db = mysql_select_db("sports");
     $query = "select team1, score1,
                         team2, score2
                         from soccer_scores";
     $result = @mysql_query($query);
     print "<b>Soccer Results:</b> <br/>";
     if (mysql_num_rows($result) > 0) :
     while ($row = mysql_fetch_array($result)) :
print "$row[team1] $row[score1] $row[team2] $row[
score2]<br/>";
     endwhile;
     else:
          print "No current results.";
     endif;
     ?>
     </p>
    </card>
</wml>
```

Cód.7-3 Accediendo a información desde una base de datos

Nota

La comunicación entre un servidor Web común y un dispositivo móvil WAP, en principio, es muy parecida a la comunicación entre un servidor Web y un browser basado en PC tradicional, pero con un paso adicional.
Este paso extra involucra el paso de información a través de un gateway WAP. Los gateways WAP sirven como intermediarios entre el browser móvil del cliente y el servidor de información.

El proceso se muestra en la figura siguiente:

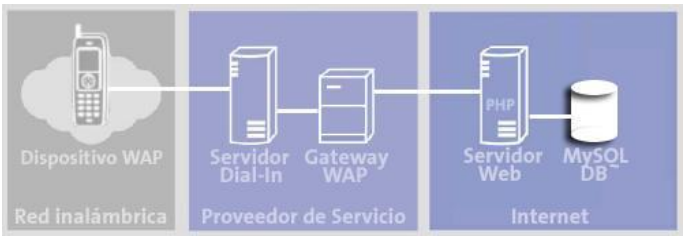


Fig.7-12 Comunicación entre un dispositivo móvil y un servidor Web

Considere un ejemplo típico del proceso de comunicaciones, como se ilustró en la figura 7-12: Se asume que se han definido dos mazos de dos cartas. Un dispositivo móvil de un usuario requiere un mazo (documento). La secuencia de eventos que tomaría lugar como resultado de esta petición sería como sigue:

• La petición es enviada al gateway WAP bajo el protocolo WAP. El gateway WAP, bajo el control del dispositivo WAP realiza un request típico de URL, usando el protocolo http.

• El request URL es transmitido a través de Internet a la dirección IP del dispositivo WAP (el operador móvil asigna al dispositivo una dirección IP).

• El request alcanza a su destino final—este es un servidor Web. El servidor, lee el encabezado y procesa el request del documento WAP. El código PHP contenido dentro de este documento es compilado y formateado según sea necesario.

• El documento WAP (mazo) localizado (y procesado) es ruteado de nuevo a través del gateway WAP, donde su contenido se comprime en datos binarios y es enviado al dispositivo WAP.

Capítulo 8. UAProf, Adaptación de contenidos WAP

El desafío
Arquitectura
Adaptando contenidos
Atributos
Los Beneficios
Estandarización

Capítulo 8. UAProf, Adaptación de contenidos WAP

El desafío

Lograr la máxima experiencia del usuario, a través de los diferentes dispositivos existentes con capacidad WAP, es más un desafío que una tarea de ingeniería como lo es el diseño o la programación de sitios WAP.

Con el paso del tiempo, la evolución de la tecnología ha ido progresivamente incorporando capacidades a los dispositivos portátiles que acortan la brecha que existía entre éstos y los dispositivos de escritorio. Debido a este fenómeno de crecimiento tecnológico, los proveedores de servicio debieron ir modificando tanto su infraestructura como la arquitectura subyacente para poder brindar servicio a un gran número de dispositivos móviles con diferentes características y capacidades.

Así mismo, los usuarios, son tentados día a día con nuevos servicios y el acceso a una inmensa cantidad de contenidos, desde los tradicionales—como la mensajería de texto y la navegación Web—hasta los más recientes, como la descarga de archivos de video, música o el mismo soporte de aplicaciones Java online.

La figura 8-1 muestra un primer enfoque de la arquitectura WAP más reciente.

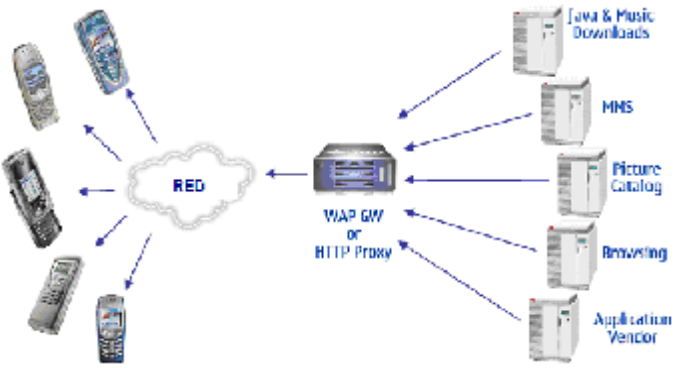


Fig.8-1 Enfoque de la más reciente arquitectura WAP

Entre los proveedores de servicio y los usuarios, está el contenido WAP. Supongamos que para cada dispositivo (de todos los existentes) un programador debiera ajustar el

contenido, es decir, debiera ajustar el tamaño del texto y las imágenes para que estas encajen en pantalla, o debiera agregar funciones a los botones de cada dispositivo específico. Sería imposible asegurar que el cien por ciento de los terminales estarían cubiertos, y esto significa que varios usuarios quedarían fuera del amplio espectro que planteó el Foro WAP cuando diseñó el protocolo.

Para solucionar este inconveniente, se propuso crear un Perfil de Agente Usuario (*UAProf, User Agent Profile*) para cada dispositivo. Este UAProf es un documento en formato XML que está alojado en un servidor público a manera de repositorio.

Un UAProf, provee información de las capacidades de los distintos dispositivos con el propósito de dar formato al contenido. Esto permite el mejor ajuste posible entre el contenido y las capacidades de un tipo de dispositivo específico. De esta forma, UAProf elige el contenido más apto para un cierto dispositivo, y lo adapta a las capacidades de este terminal.

Arquitectura

Cómo se realiza la adaptación de contenidos, es quizá más simple de lo que se podría presuponer: el proceso completo se realiza en el servidor de contenidos, luego de importar el perfil desde la cache local o del repositorio.

Un gateway WAP con soporte de reenvío de encabezados UAProf (*UAProf header forwarding*) lee la información del encabezado enviado por el dispositivo para indicarle al servidor de contenidos qué perfil deberá importar.

El resto es un simple mapeo de información, desde un formato genérico a otro específico del terminal.

La figura 8-2 muestra los elementos de una arquitectura WAP con soporte UAProf.

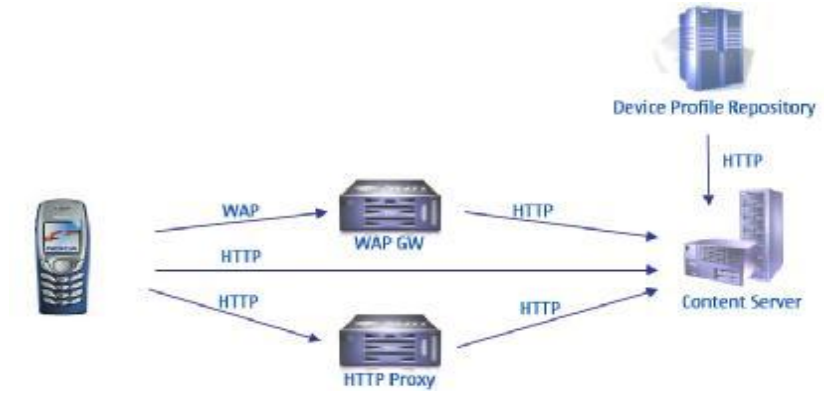


Fig.8-2 Arquitectura WAP con soporte UAProf

Adaptando contenidos

En resumen, los perfiles de agente usuario hacen accesible la información de las capacidades de los dispositivos por las diferentes partes, para lograr el mejor ajuste posible entre el contenido y estas capacidades diferentes.

Esta adaptación permite:

• Elegir entre contenido XHTML o WML según el soporte del dispositivo.

• Ajustar el tamaño de una imagen y su mapa de color para una determinada pantalla.

• Reducir el tamaño de un archivo de imagen, video o música mediante *resampling*.

Como se mencionó anteriormente, existen repositorios públicos disponibles, en donde se almacenan los distintos perfiles de usuario. Estos repositorios no son más que servidores HTTP propios de cada fabricante de donde los proveedores de servicio pueden descargarse los perfiles como documentos XML.

El siguiente, es un ejemplo de un dispositivo enviando en el encabezado de sesión el URL del repositorio de perfiles (Figura 8-3) junto con un fragmento del perfil correspondiente a un dispositivo, en este caso, un Nokia 6100.

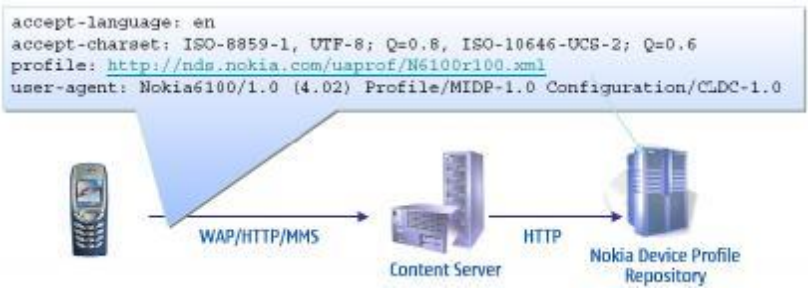


Fig.8-3 Identificación de dispositivos

```
<!--DOCTYPE rdf:RDF SYSTEM "uaprof.dtd"-->
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/
02/22-rdf-syntax-ns#" xmlns:prf="http://www.
wapforum.org/profiles/UAPROF/ccppschema-
20010430#">
  <rdf:Description rdf:ID="Nokia6100">
    <prf:component>
      <rdf:Description rdf:ID="HardwarePlatform">
        <rdf:type rdf:resource="http://www.
wapforum.org/profiles/UAPROF/ccppschema-
20010430#HardwarePlatform"/>
        <prf:BitsPerPixel>1</prf:BitsPerPixel>
        <prf:ImageCapable>Yes</prf:ImageCapable>
        <prf:InputCharSet>
          <rdf:Bag>
            <rdf:li>iso-8859-1</rdf:li>
            <rdf:li>utf-8; q=0.8</rdf:li>
            <rdf:li>iso-10646-ucs-2;
q=0.6</rdf:li>
          </rdf:Bag>
                          ...
```

Atributos

Los perfiles UAProf, están compuestos por varias secciones, cada una de ellas hace referencia a una característica o atributo en especial de un cierto dispositivo. Entre los distintos atributos se pueden encontrar:

- HardwarePlatform. Contiene propiedades del hardware del dispositivo: capacidad de color, tamaño de pantalla, número de teclas, aspecto del píxel, capacidad de sonido, etc.

• SoftwarePlatform. Contiene propiedades del ambiente de aplicación del dispositivo, sistema operativo y software instalado: OS, capacidad de descarga de software, tipos de contenido (MIME), conjunto de caracteres (código de página), etc.

• BrowserUA. Contiene atributos relacionados con el navegador del dispositivo: nombre del navegador, versión de XHTML, módulos XHTML, soporte de frames y tablas, soporte JavaScript™, etc.

• NetworkCharacteristics. Contiene propiedades que describen el ambiente de red: Banda portadora, datos de sesión, bandas portadoras disponibles, mecanismos de seguridad y encriptación disponibles, etc.

• WapCharateristics. Contiene propiedades del ambiente WAP: tamaño del mazo, versión de WML, librerías de WMLScript, librerías WTAI, disponibilidad de descargas de la OMA, etc.

• PushCharacteristics. Contiene propiedades de las capacidades WAP Push: tipo de contenidos Push, tipo de aplicaciones Push, conjunto de caracteres Push, tamaño del mensaje Push, etc.

Los Beneficios

A simple vista, los beneficios que se pueden percibir cuando se piensa en el funcionamiento de los perfiles de agente de usuario, se reducen a una mejor adaptación de contenido a los diferentes dispositivos existentes en el mercado.

Esto es cierto, al menos hasta cierto punto. También se puede enumerar una lista de beneficios para todas las partes que intervienen en el proceso de interacción del usuario con el contenido WAP:

• Mejor experiencia del usuario y aumento en la satisfacción del cliente ofreciendo servicios sutiles y de buen funcionamiento.

• Mejores servicios implican mayor uso y por lo tanto más rédito para los proveedores de contenido.

• Mejores servicios implican mayor uso y por lo tanto incremento en las suscripciones de usuarios para los operadores.

• Mejores servicios implican mayor uso y por lo tanto aumento en las ventas de dispositivos para los fabricantes.

Estandarización

El estándar de perfiles de usuario, UAProf, está definido por la OMA (*Open Mobile Alliance*).

Actualmente, los perfiles definen:

• Un formato para describir los atributos y características de los dispositivos.

• Un vocabulario central para definir atributos específicos de los dispositivos.

• Un protocolo para que el cliente pueda informar a los servidores de sus atributos (extensión del HTML).

• Reglas para determinar como los servidores resuelven los perfiles a partir de la información enviada por los clientes.

En el Apéndice C, se incluyen los UAProfs para la mayoría de los dispositivos móviles del mercado.

Para más información acerca de los perfiles UAProf, puede consultar en http://www.w3.org/Mobile/CCPP[9].

[9] User Agent Profile (UAProf), o Composite Capabilities/Preference Profile (CC/PP) son sinónimos.

Capítulo 9. El Futuro del WAP

La industria móvil

De dónde venimos
Los Escalones previos: de 1G a 3G
La primera generación (1G):
Sistemas Celulares Analógicos
La segunda generación (2G):
Sistemas Digitales Múltiples
La tercera generación (3G)

Hacia dónde vamos

WAP en el ambiente competitivo

Capítulo 9. El Futuro del WAP

La industria móvil

El enorme surgimiento de interés y desarrollos en el área de la información móvil en los tiempos que corren, han causado que los operadores de red, la infraestructura, y los fabricantes de terminales colaboren en una escala sin precedentes, en un área notoria por la diversidad de estándares y protocolos. Los esfuerzos de colaboración del Foro WAP, han proyectado y continuado el desarrollo de un conjunto de protocolos para proveer un ambiente común para el desarrollo de servicios telefónicos avanzados y acceso a Internet para el mercado inalámbrico. Si los protocolos WAP fueran tan exitosos como el TCP/IP, el boom en las comunicaciones móviles sería fenomenal. De hecho, un navegador WAP haría para la Internet móvil lo que Netscape hizo por Internet.

Como se mencionó anteriormente, los participantes de la industria—desde desarrolladores de contenido hasta operadores—pueden explorar las vastas oportunidades que presenta WAP. Internet ha probado ser exitosa, llegando a millones de hogares en todo el mundo como una tecnología de red fija. Sin embargo, los usuarios móviles hasta ahora han sido forzados a aceptar niveles de funcionalidad relativamente básicos, sobre y bajo comunicaciones de voz y, de a poco, han comenzado a demandar a la industria, mudarse desde un ambiente fijo a uno móvil, llevando consigo la funcionalidad y los éxitos de este ambiente fijo.

En este continuo proceso de evolución, la industria atravesó diferentes etapas tecnológicas, de desarrollo, y de amoldamiento, como la mayoría de las tecnologías que se incorporaron en la historia de la humanidad. Entender cómo la industria llegó hasta donde hoy está y poder mirar más allá, y pensar qué vendrá hace necesario que esta evolución sea recorrida, ubicándonos a principios de los '90.

Era muy probable esperar que los fabricantes de terminales móviles experimentaran un cambio significativo como resultado de la tecnología WAP—un cambio que impactara sobre el *look and feel* del hardware que producen. El primer tema que enfrentó este brazo de la industria, tiene que ver con el tamaño de los dispositivos, fuentes de energía,

tamaños de pantalla, usabilidad, y el rol que cumplen los terminales móviles en la vida de un usuario.

Con casi el 75 por ciento de los fabricantes de dispositivos móviles involucrados en el Foro WAP, era también muy probable que en poco tiempo ya no quedaran usuarios fuera de la nueva era de las comunicaciones: dispositivos compatibles con un microbrowser embebido, capaz de interpretar WML y mostrar contenido interactivo al usuario, lentamente comenzaron a inundar el mercado.

La especificación abierta del WAP prometía a los usuarios una naturaleza de servicios de un amplio rango, la similaridad con el ya establecido y aceptado modelo de Internet, y la simplicidad de los lenguajes WML/WMLS con los que las aplicaciones serían escritas. La información estaría disponible mediante funciones *push-and-pull*, con la habilidad de interactuar con los servicios tanto vía voz como vía interfaces de datos. La navegación Web, así como la conocía un usuario de PC de escritorio, sin embargo, no iba a ser seguramente el mayor atractivo del WAP, debido a las restricciones de tiempo y procesamiento para ese entonces.

Aplicaciones de tiempo real y servicios demandando una pequeña cantidad de información clave serían el combustible que llevarían al éxito del WAP en el mercado móvil. La cotización de las acciones, noticias, el informe meteorológico, e información sobre viajes son algunas de las tantas áreas en las que el WAP proveería servicios para usuarios móviles. Esencialmente, la estrategia de la aplicación WAP consiste en tomar los servicios ya existentes que son comunes entre sí del ambiente fijo y conducirlos al prometedor y amigable ambiente móvil.

De dónde venimos

Los Escalones previos: de 1G a 3G

Las ondas electromagnéticas fueron descubiertas como un medio de comunicación a finales del siglo 19. Los primeros sistemas ofreciendo servicio telefónico móvil fueron introducidos a finales de los ’40 en los Estados Unidos y en los principios de los ’50 en Europa. Estos sistemas tempranos de celda única estaban severamente restringidos en movilidad, la poca capacidad, un servicio limitado y pobre calidad de audio. El equipo era pesado, voluminoso, caro, y susceptible a

interferencias. Debido a estas limitaciones, el sistema registraba menos de un millón de suscriptores a principios de los '80.

La primera generación (1G): Sistemas Celulares Analógicos

La introducción de los sistemas celulares en los finales de los '70 y principios de los '80, representaron un salto en las comunicaciones móviles (especialmente en la capacidad y la movilidad). La tecnología de los semiconductores y microprocesadores, hicieron a los terminales más pequeños, livianos y más sofisticados, y por lo tanto, una realidad más práctica para muchos más usuarios.

Estos sistemas transmitían solo voz analógica. Los más prominentes sistemas celulares de 1G son AMPS (*Advanced Mobile Phone System*), NMT (*Nordic Mobile Telephone*) y TACS (*Total Access Communication System*).

Con la introducción de 1G, el mercado de las comunicaciones móviles mostró tasas de crecimiento anual de entre 30 y 50 por ciento, llegando a casi 20 millones de suscriptores en 1990.

La segunda generación (2G): Sistemas Digitales Múltiples

El desarrollo de los sistemas de segunda generación fue guiado por la necesidad de mejorar la calidad de transmisión, capacidad del sistema y cobertura. Los avances en tecnología de semiconductores y dispositivos microonda acercaron las transmisiones digitales a las comunicaciones móviles. La transmisión de audio aún domina las autopistas aéreas, pero la demanda de fax, mensajes cortos y transmisión de datos crece rápidamente.

Los servicios suplementarios, como prevención de fraude y encriptación en los datos del usuario, se han convertido en características estándar comparables con las de una red fija, y esto es lo que anima a más usuarios a sumarse al sistema.

Los sistemas celulares 2G incluyen GSM, D-AMPS (*Digtial AMPS*) y CDMA (*Code Division Multiple Access*)/TDMA (*Time Division Multiple Access*). Actualmente, múltiples sistemas estándar 1G y 2G se usan en las comunicaciones móviles alrededor del mundo.

Los diferentes estándares sirven diferentes aplicaciones con diferentes niveles de movilidad, capacidad, y área de servicio (sistemas de *paging*, teléfonos inalámbricos, canales de radio privados, sistemas celulares y satelitales móviles). Además, varios de estos estándares son usados en una misma región.

Actualmente, en el mundo existen más de 450 millones de suscriptores al sistema celular, de los cuales 250 millones son suscriptores del sistema GSM (*Global System for Mobile communication*), divididos en 140 países y 400 redes.

La tercera generación (3G)

La Fase 1 de la estandarización de GSM se completó en el ETSI (*Instituto Europeo de Estándar de Telecomunicaciones*) en 1990 e incluyó todas las definiciones para las operaciones de la red GSM. Este estándar fue mejorado en la Fase 2 (1995) para incorporar una gran variedad de servicios suplementarios comparables con los del estándar de las redes fijas ISDN (*Integrated Services Digital Network*). En 1996, el ETSI decidió mejorar aún más al GSM en lo que se conoce como Fase 2+ que definitivamente define la evolución de 2G a 3G.

En la figura 9-1 se muestra la evolución en la cantidad de suscriptores al sistema de comunicaciones móviles, desde la aparición de los sistemas primitivos hasta la actualidad.

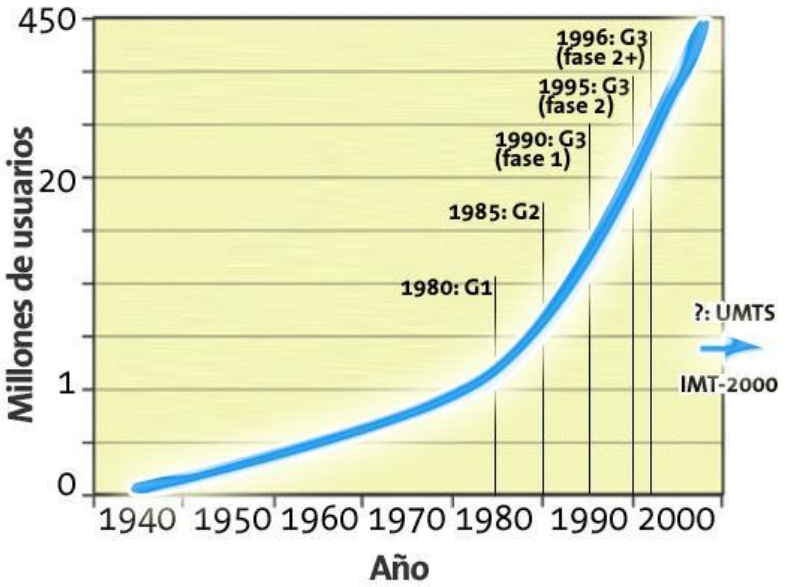


Fig.9-1 Evolución de los estándares y cantidad de suscriptores.

Hacia dónde vamos

Todo hace pensar que la evolución de las comunicaciones móviles no termina en nuestra era. Por el contrario, ya existe en desarrollo un sucesor del estándar 3G, el UMTS (*Universal Mobile Telecommunication System*).

Este estándar completará la última fase de los sistemas 3G y promete ser como una única familia para todas las aplicaciones y países. Responde a un conjunto de requerimientos definido por la ITU (*International Telecommunications Union*) y se conoce como IMT-2000. IMT por *International Mobile Telecommunications* y 2000 porque ese es el año en que se comenzó su desarrollo.

La propuesta más importante de IMT-2000 es el estándar UMTS (*o W-CDMA, wide-band CDMA*) como sucesor de las tecnologías existentes hasta ese momento, como una forma de conducir a los estándares previos hacia la meta final: IMT-2000.

UMTS permite introducir mucho más aplicaciones al mundo de los usuarios de sistemas móviles y provee un vínculo vital entre los actuales sistemas y el IMT-2000. La nueva red, además predice un aumento en la demanda de aplicaciones móviles para las nuevas capacidades en el sobrepoblado mundo del cielo de las comunicaciones móviles. UMTS aumentaría la velocidad de transmisión a 2 Mbps.

Se prevé que hacia finales del 2005, UMTS será el estándar que domine a las comunicaciones móviles. Para mayor información sobre el tema se pueden consultar los documentos de la IEC (*International Engineering Consortium*) en Internet: http://www.iec.org

WAP en el ambiente competitivo

En la actualidad, se podría decir que la competencia del WAP podría venir desde varias fuentes:

- SIM (Suscriber Identity Module) toolkit. El uso de SIMs o *smart cards* en dispositivos móviles está ampliamente difundido.

- Windows CE. Es un sistema operativo multitarea y *multithreaded* de Microsoft, diseñado para ser incluido o embebido en dispositivos móviles y/o de tamaños restringidos.

- JavaPhone™. Sun Microsystems está desarrollando las API de Personal Java™ y JavaPhone™, que se embeberían en una JVM (*Java™ virtual machine*) en el dispositivo. Los fabricantes de estos terminales podrían construir dispositivos con capacidad para descargar características y funciones extra desde Internet, así, los

clientes no tendrían que comprar nuevos dispositivos para sacar provecho de las mejoras en la funcionalidad.

Las ventajas que el WAP puede ofrecer sobre estos otros métodos son:

- WAP es un estándar abierto, independiente de proveedores y fabricantes.

- WAP es independiente de los estándares de red.

- WAP es un mecanismo de transporte—optimizado para bandas portadoras de datos inalámbricos.

- WAP propone descargar aplicaciones desde un servidor, permitiendo la creación e introducción de servicios más rápidamente, al contrario del software embebido.

Conclusión

Al menos 200 millones de personas usan Internet hoy para comunicarse con otros, buscar información, o hacer algún tipo de operaciones comerciales. En contraste, hay 450 millones de usuarios de teléfonos celulares alrededor del mundo.

Como resultado de las nuevas tecnologías como SMS y WAP, los usuarios de teléfonos celulares se han unido a la revolución de Internet –como lo ilustran los 2 billones de emails que se enviaron usando sus teléfonos celulares en Octubre de 1999 o los 7 billones en Marzo de 2003, o la venta de teléfonos con soporte de navegación Web y el emergente segmento del mercado de comercio electrónico móvil, que se proyecta crecer desde u$s 313 millones en 1998 hasta u$s 23.7 billones en 2003.

Si creemos en los fabricantes de teléfonos celulares, habrá 1 billón de usuarios de móviles en el 2005. De acuerdo con Nokia y Ericsson, la mitad de estos, con soporte para navegación en Internet, creando una plataforma de Internet móvil con cientos de millones de usuarios.

Haciendo móviles a las aplicaciones Web

Las implicaciones sobre el crecimiento y la evolución de la tecnología celular antes citada son alarmantes. Compañías de productos y agresivos "punto coms" deberán planear brindar soporte al creciente número de clientes móviles para poder acceder al contenido y los servicios de sus aplicaciones Web. Esto debe hacerse dentro de las restricciones del ambiente más hostil de los dispositivos portátiles, que son menos amigables que las PCs.

Los dispositivos portátiles, particularmente los teléfonos celulares, están limitados por los requerimientos de bajos consumos de energía, los CPUs de baja capacidad de procesamiento, ancho de banda nominal, y conexiones inestables. Peor aún, estos dispositivos tienen escasamente doce teclas en su teclado para dar soporte a la navegación y funciones de entrada de datos.

A pesar de los desafíos técnicos y de diseño de interfaces, el soporte inalámbrico es una eventualidad natural para las aplicaciones Web. Recibir información de que su vuelo ha sido cancelado antes de partir hacia el aeropuerto es algo perfectamente adecuado para un dispositivo Web móvil. El

desafío es, para los desarrolladores, de encontrar una forma para reconstruir sus aplicaciones existentes para que "hablen el lenguaje inalámbrico".

XML sortea las dificultades al mostrar contenido en pantallas pequeñas.

Hay dos desafíos para las aplicaciones Web en un mundo inalámbrico: la presentación y la conectividad. La presentación se refiere a la muestra de contenidos y a la interacción del usuario con las aplicaciones móviles. Por supuesto, mientras más actúe un dispositivo móvil como un navegador, mejor.

La respuesta de la industria al HTML es el lenguaje de marcas WML, una derivación del HTML en formato XML, que a diferencia del HTML, aprovecha al máximo las ventajas de la experiencia del usuario móvil.

La pantalla se describe como un mazo, separado en cartas, para encargarse de un área de visualización restringida. El teclado del teléfono se considera como el conjunto de controles por defecto, en vez del teclado QWERTY estándar.

Sumado al WML, los microbrowsers de los teléfonos celulares tienen un lenguaje de scripting, WMLScript, que ayuda en la creación de sitios más dinámicos, orientados a los dispositivos móviles, e incluso permiten agregarle lógica a las cartas WML. Sin embargo, no cubren la adaptación de contenido a los diferentes dispositivos.

En un ambiente de estrategias tecnológicas estrepitosas, hay más de una forma de lograr el soporte para una gran variedad de pantallas. Una tecnología para realizar transformaciones XML-a-HTML/WML es la especificación de hojas de estilo XSL, usando UAProf o CC/PP. Mediante una hoja de estilos XSL apropiada, el servidor Web o servidor de aplicaciones, puede formatear los datos XML para aparecer correctamente en las pantallas destino.

La clave para una aplicación exitosa en el futuro, será su habilidad para crear un documento XML bien formado, interpretado con precisión por cualquier navegador, móvil o de escritorio.

Los gateways WAP proveen conectividad

Generar WML es sencillo, pero una vez hecho, ¿cómo entregarlo a un dispositivo móvil? El hecho es que el HTTP no es apropiado para los dispositivos portátiles. La tecnología destinada para sobrellevar las restricciones de los recursos y la comunicación de los dispositivos móviles es WAP.

WAP, como está definido por el Foro WAP, provee la presentación y la entrega de información y servicios a teléfonos móviles y otras aplicaciones de Internet.

¿Cómo se logra que las aplicaciones Web que requieren mucho ancho de banda "negocien" con los dispositivos inalámbricos? La respuesta es el gateway WAP. WAP es el HTTP del mundo inalámbrico. Un gateway WAP está diseñado con el propósito de entregar el contenido y los servicios existentes del lado del servidor al lado de los clientes.

Un gateway WAP traduce de una pila, HTTP, a la otra, WAP, codificando y decodificando contenido Web en una forma compacta para una red de datos inalámbrica. Los gateways WAP son esenciales, ayudando a aislar los mecanismos propios de cada compañía para acceder a los teléfonos de sus clientes y viceversa.

Un ojo crítico sobre las aplicaciones existentes

Si bien están dadas todas las condiciones para que se forme un estándar sobre qué y cómo deberían ser las aplicaciones WAP, y con el crecimiento sorprendente en el número de los usuarios móviles como argumento, en la actualidad, la mayoría de las aplicaciones existentes para el mundo WAP, ya sea las creadas exclusivamente para el ambiente móvil, o las que fueron migradas a este desde la Web, presentan ciertas carencias que no hacen a una buena estructura de la aplicación y por lo tanto no están optimizadas para un ambiente WAP; en otras palabras, no están bien formadas.

Como resultado del análisis de los sitios WAP citados en la sección Estado del Arte del prefacio, y teniendo en cuenta las conclusiones a las que se arriban luego de esta tesis, podemos enumerar los puntos que deberían realizarse siguiendo un criterio puramente objetivo.

• Poseen líneas en blanco, que pueden hacer que el usuario piense que no hay más información debajo.

• Los links no están ordenados alfabética o lógicamente para facilitar la navegación. Además existen links de más de una línea que no son recomendables.

• Pantallas de información demasiado largas, que pueden causar desorientación en los usuarios.

• No hacen un uso de títulos: el usuario no sabe en que parte del sitio se encuentra en un determinado momento, lo que complica la navegabilidad.

• No proveen links al home de la página o a algún sitio familiar para el usuario.

• No hacen uso de las teclas de acceso rápido.

• No utilizan el atributo *alt* de las etiquetas <img> para proveer un texto alternativo a la imagen.

Mirando hacia adelante

La migración de los servicios basados en la Web actuales a la red inalámbrica de datos puede ser un tanto dura. Sobre todo, es solo una cuestión de cooperación entre los estándares de la industria y los esfuerzos Open Source.

Usualmente, una nueva industria está atestada de incompatibilidades y disputas políticas. Ciertamente algunas de estas aún existen.

Pero el campo de juego creado por la combinación de los estándares de facto de Internet, y las tecnologías como WAP, XML, WML y DOM, serán la excusa de varias batallas inútiles por la migración al mundo inalámbrico.

Nuestro aporte

En Argentina, por diferentes cuestiones, no podemos disfrutar por completo los beneficios que brinda el protocolo WAP. Debido a esto, no hemos tenido la oportunidad de hacer uso del mismo, y por lo tanto, no hemos podido confeccionar una conclusión acerca de sus capacidades.

Principalmente, y en relación con lo antes dicho, nuestro aporte principal radica en un conjunto de puntos a tener en cuenta, a la hora de desarrollar una aplicación para un ambiente móvil.

No es un secreto que en Argentina el atraso tecnológico es quizás el factor limitante de la mayoría de los desarrollos destinados a mejorar la calidad de vida y el confort de los usuarios finales.

Durante nuestro estudio de la tecnología wireless, el protocolo WAP y los lenguajes de aplicación, encontramos una problemática que excede nuestro "poder cambiar las cosas y hacer un mundo mejor". ¿Cómo desarrollar aplicaciones pensando en todas las ventajas de las que disfruta el mundo desarrollado como Europa y Japón y ahora EEUU y Canadá, y que en nuestro país están tan lejos de ser realidad?

Fundamentalmente, la red GSM provee la seguridad y ancho de banda suficiente para el transporte de comunicaciones de voz, mensajería de texto y multimedia; por supuesto, en nuestro país las redes son de una generación anterior. Esto y los dispositivos de última generación hacen que el WAP, en Argentina, se parezca a la Internet de los 90s.

Es bueno sí, tener en cuenta los consejos y las "buenas prácticas" propuestas en este trabajo, pensando en desarrollar aplicaciones a futuro, si es posible utilizando XHTML y XML, pero también tenemos que tener presente que los celulares en el mercado soportan más características de las que están disponibles, y por eso, el limitante es como dijimos, tecnológico. Tecnológico y social, como pasa siempre con cualquier tecnología nueva: al principio nos mostramos reticentes, y no es hasta que se vuelve una necesidad imperiosa o se transforma en una costumbre, que nos decidimos a adoptarla. No obstante, y con más esperanza que visión de la realidad futura, la tecnología evolucionará y seguramente permitirá que los dispositivos puedan usar todas sus características.

He ahí nuestro aporte, una guía a futuro para el desarrollo de aplicaciones móviles, abiertas a los nuevos lenguajes y con compatibilidad hacia atrás, teniendo en mente las limitaciones existentes y las ventajas que vendrán; y por ello, en lugar de

usar el atraso tecnológico como excusa proponemos una solución.

En nuestro caso pretendíamos desarrollar una aplicación que integrara un sistema de mensajería Web entre alumnos y profesores con el ambiente WAP y así permitir el acceso a este sistema mediante dispositivos móviles con la comodidad que esto trae aparejado.

El principal problema que se nos presentó fue que en nuestro área de cobertura celular la navegación WAP no está disponible. A decir verdad, en la mayoría de las ciudades no hay soporte de navegación WAP, salvo en aquellas que concentran la mayor cantidad de habitantes. A este inconveniente se lo pudo resolver haciendo uso de emuladores del protocolo WAP.

Otra complicación que presenta el desarrollo de estas aplicaciones es que el protocolo WAP 2.0 así como fue diseñado no se encuentra implementado en nuestro país, al menos no completamente. Esto implica que los servidores WAP, los proxies o gateways WAP y los servidores de perfiles UAProf, o bien no existen o son unas primeras implementaciones, pero en todo caso no son gratuitos. La solución a esto es utilizar servidores Web adaptados para brindar contenido móvil junto con una lógica de aplicación para simular la adaptación de contenidos.

Como se puede imaginar, tampoco están disponibles las características de SMS de manera directa, esto es, disponer de un iniciador PUSH de manera libre y gratuita para dar lugar a los mensajes que manejaría el sistema propuesto; sin embargo, la mensajería SMS puede accederse a través de las diferentes operadoras del servicio celular, por lo que los mensajes tendrán un costo sujeto a las disposiciones de estas empresas. Este obstáculo impide que el servicio de mensajería que brinda nuestro sistema sea en todos los casos gratuito. Cosa que se hubiese podido lograr si usáramos nuestro propio servidor SMS.

Como si fuera poco, no todos los lenguajes de marcas son soportados por la gran mayoría de los dispositivos en el mercado. En la actualidad el lenguaje predominante, y que asegura la mayor compatibilidad, es el WML 1.1, por eso decidimos adoptarlo para nuestro sistema.

En lo que respecta a la construcción del sitio no encontramos mayores problemas, primero diseñamos los modelos del dominio, los requerimientos, el modelo de casos de

uso y el diagrama de clases del modelo del diseño, siguiendo el Proceso de Desarrollo Unificado de Rational (RUP)[10].

Para la implementación del sistema decidimos utilizar PHP como lenguaje principal para agregar lógica del lado del Servidor y para generar contenido dinámico, así como para la adaptación de contenido según el dispositivo que lo requiera, y así entregar código HTML y JavaScript para los navegadores de escritorio y WML y WMLScript para los dispositivos móviles.

La base de Datos que utiliza el sistema es mantenida y accesible mediante el motor MySQL.

Como se puede observar, todas las tecnologías utilizadas son de uso libre.

Las ventajas que se pueden enumerar acerca de los lenguajes orientados a la tecnología WAP tienen que ver con la similitud que presentan con los lenguajes tradicionales orientados a la Web fija, y por ello, la simplicidad para producir código móvil. Un lenguaje simple, bien estructurado y estricto, combinado con la dinámica y la potencia de los lenguajes de programación orientados a la Web y una integración directa con un motor de base de datos, como es el caso del WML, es la combinación perfecta para el desarrollo de aplicaciones rápidas, dinámicas y confiables.

El código de la aplicación se puede encontrar en el Apéndice E del CD-ROM que acompaña a esta publicación.

Más información y otros aspectos de los servicios y la utilización del sistema están descritos en el material de ayuda que incluye el sistema, provisto en el CD-ROM y accesible también en http://www.siuwap.ing.unlpam.edu.ar (para ambos protocolos, HTTP y WAP).

[10] El proceso de desarrollo para la aplicación propuesta en este trabajo de investigación como caso de estudio puede encontrarse en el Apéndice E de esta publicación.

Apéndice A. SMS

Definición
Introducción
Beneficios del SMS
Elementos de Red y Arquitectura
Funcionamiento
Aplicaciones SMS

Apéndice A. SMS

Definición

El servicio SMS (*Short Message Service*) es un servicio móvil aceptado globalmente, que permite la transmisión de mensajes alfanuméricos entre usuarios de sistemas celulares y sistemas externos—como por ejemplo correo electrónico, paging y sistemas de correo de voz.

Introducción

SMS apareció en la escena del mundo de los dispositivos portátiles hacia 1991 en Europa, donde la tecnología inalámbrica digital tuvo sus primeras raíces.

El estándar para esta tecnología se conoce actualmente como GSM, que incluye el servicio SMS desde sus principios. En 1998, cuando en América se finalizó la implementación de las redes PCS—basadas en GSM—y los métodos CDMA y TDMA cobraron popularidad entre los proveedores de servicios digitales móviles, el SMS pudo finalmente gozar de sus completas y ventajosas características, también en el nuevo continente.

El servicio SMS punto a punto, provee un mecanismo para transmitir mensajes cortos entre terminales portátiles. Este servicio hace uso de un SMSC (*Short Message Service Center*) que actúa como un sistema de almacenamiento y reenvío (*store and forward system*) de mensajes cortos. La red inalámbrica provee el transporte de estos mensajes entre los SMSCs y los terminales.

En contraste con los servicios de transmisión de texto preexistentes—como el paging—el servicio SMS está diseñado para proveer entrega garantida de mensajes de texto al destino. De este modo, las fallas temporales en la red son detectadas y el mensaje es almacenado hasta que el destino sea alcanzable.

Beneficios del SMS

En el mundo competitivo de hoy en día, la diferenciación[11] es un factor muy significativo en el éxito de un proveedor de servicios. Ya que los servicios básicos—como la telefonía de voz—es algo muy común, otros servicios—como el SMS—proveen un medio muy poderoso para este propósito.

Los beneficios que el SMS brinda a un proveedor de servicios son:

- Complementa las comunicaciones inalámbricas de voz clásicas con un potente sistema de mensajería instantánea de texto.

- Es un servicio alternativo al paging.

- Permite a los usuarios corporativos el acceso a datos de manera inalámbrica.

- Provee de servicios de valor agregado—como la integración con e-mail, correo de voz y fax-mail; servicio de recordatorios; cotizaciones y banca digital; reservas aéreas y de hotelería; etc.

- Provee de servicios de administración clave—como aviso de cargos, descarga de contenidos y provisión de servicios.

Todos estos beneficios son fácilmente accesibles con un mínimo incremento en el costo, y típicamente las inversiones se recuperan en menos de seis meses.

Por otro lado, los beneficios que SMS aporta a los clientes giran en torno a la conveniencia, flexibilidad e integración de servicios de datos y mensajes. Desde esta perspectiva, el beneficio es poder usar un dispositivo portátil como una extensión de la computadora. SMS también elimina la necesidad de dispositivos distintos para diferentes tipos de mensajería, ya que permite la integración de todos ellos en un mismo dispositivo inalámbrico—el terminal móvil.

---

[11] *Diferenciación*, en este contexto, se refiere a la posibilidad que tiene una empresa prestadora de servicios, de distinguirse de otras, justamente por los servicios que presta.

En la figura B-1 se muestran los elementos de la red relacionados con la arquitectura de los servicios SMS.

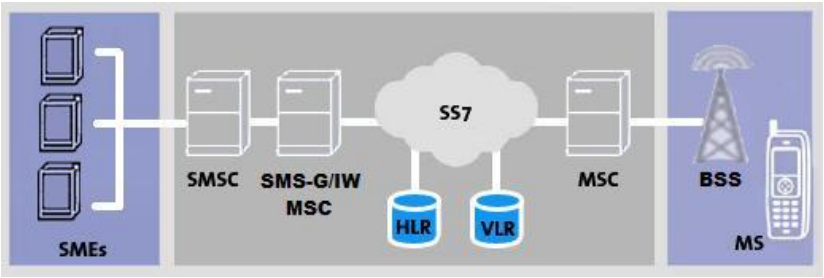


Fig.B-1 Elementos de la red y arquitectura del servicio SMS

SME (*Short Messaging Entities*). Un SME es una entidad que puede enviar o recibir mensajes. El SME puede estar localizado dentro de la red fija, en una estación móvil, o en un centro de servicios—como por ejemplo, un SMSC.

SMSC (*Short Message Service Center*). Un SMSC es responsable del manejo y *store-and-forwarding* de mensajes entre un SME y una estación móvil.

SMS-G/IWMSC (*SMS Gateway/Interworking Mobile Switching Center*). Un SMS-GWMSC es un MSC capaz de recibir un mensaje de un SMSC, interrogar a un HLR (para cuestiones de ruteo) y entregarlo al MSC que aloja al recipiente de mensajes del terminal destino.

HLR (*Home Location Register*). El HLR es una base de datos usada para almacenamiento permanente y manejo de suscripciones de usuarios y perfiles de servicio. Cuando es interrogado por un SMSC, el HLR provee la información de ruteo para el usuario indicado. El HLR además indica a un SMSC que no había logrado enviar mensajes pasados a un determinado terminal, cuando una estación móvil sea reconocida y esté accesible.

MS (*Mobile Station*). Una MS es un terminal capaz de recibir y originar mensajes cortos así como llamadas de voz.

VLR (*Visitor Location Register*). El VLR es una base de datos que contiene información temporal sobre usuarios. Esta

información es necesaria para el MSC cuando brinda servicio a usuarios visitantes[12].

MSC (*Mobile Switching Center*). El MSC hace las funciones de *switching* de sistemas y controla las llamadas de y hacia otros teléfonos y sistemas de datos.

BSS (*Base Station System*). Las funciones relacionadas con el sistema de radio son realizadas en las BSS. Una estación base consiste en BSCs (*base station controllers*, o controladores de estación base) y de BTSs (*base-transceiver station*, o transceptores de estación base), y su responsabilidad principal es de transmitir tráfico de voz y datos entre estaciones móviles.

SS7 (*Signaling System 7*). La red inalámbrica tiene una infraestructura basada en SS7, un estándar publicado por la *Telecommunication Industry Association* [IS-41] que ofrece las operaciones básicas necesarias para proveer servicios de información de ruteo, entrega de mensajes punto a punto, indicación de mensajes en espera y alertas al SMSC—como por ejemplo un HLR que indica a un SMSC que no había logrado enviar mensajes pasados a un determinado terminal, que la estación móvil fue reconocida y está accesible.

Funcionamiento

Un SMS puede originarse en un MS y tener como destino otro MS, siempre pasando a través de un SME. O, por otro lado, puede originarse en un SME y tener como destino un MS. Por lo tanto, se puede afirmar que siempre es un SME quien envía mensajes y estos mensajes tienen siempre como destino un MS.

Para ilustrar el funcionamiento del servicio SMS y la interacción de los distintos elementos que conforman su arquitectura, ilustramos a continuación una secuencia típica de un mensaje SMS, primero desde un MS hasta un SME y luego desde ese SME hasta un segundo MS (este es, como se dijo anteriormente, un caso típico de envío de un SMS entre dos terminales móviles).

[12] Un usuario visitante o usuario viajero es aquel que se encuentra en un área no local para su terminal, es decir, que está en una red diferente a su red local.

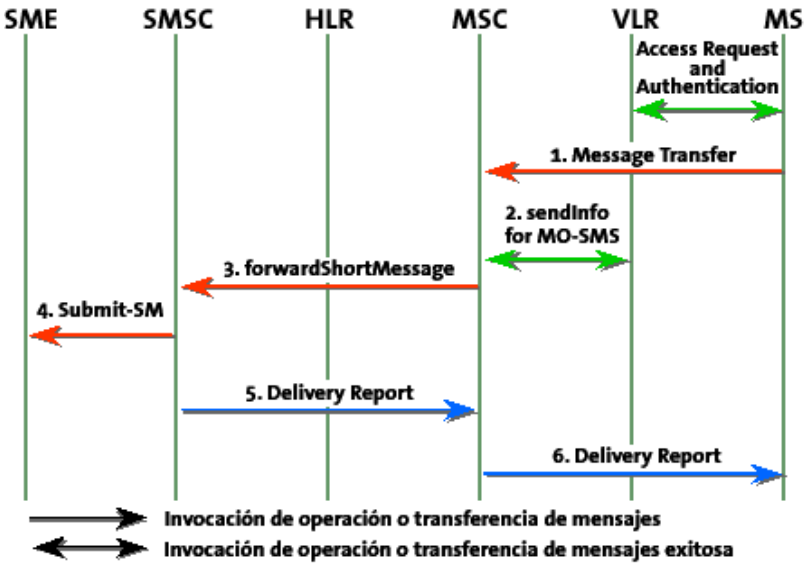


Fig.B-2 Ejemplo de un SMS saliente

Escenario de un mensaje originado en una estación móvil (*MO-SM, Mobile Originated-Short Message*).

- El MS transfiere el mensaje al MSC.
- El MSC interroga al VLR para verificar que la transferencia del mensaje no viola los servicios suplementarios invocados o las restricciones impuestas.
- El MSC envía el mensaje al SMSC usando la operación *forwardShortMessage*.
- El SMSC entrega el mensaje al SME.
- El SME informa al MSC del resultado de la operación *forwardShortMessage*.
- El MSC le retorna el resultado de la operación al MS.

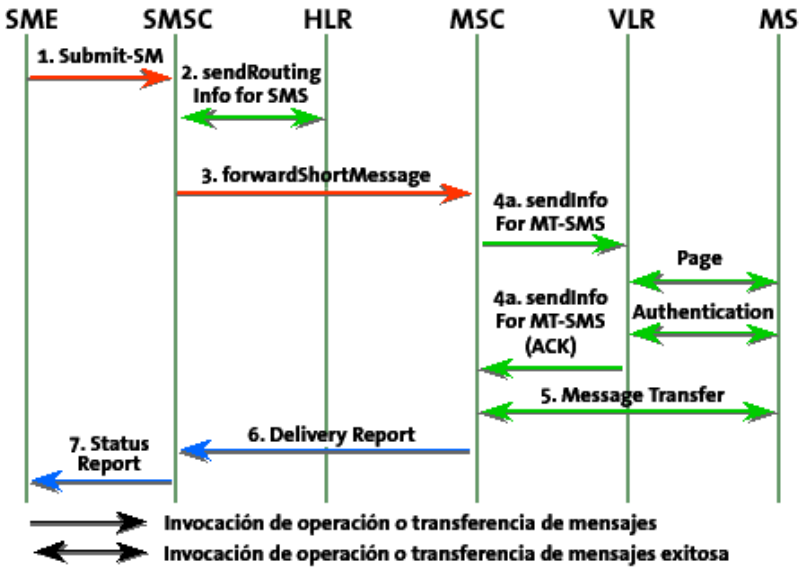


Fig.B-3 Ejemplo de un SMS entrante

Escenario de un mensaje terminado en una estación móvil (MT-SM, Mobile Terminated-Short Message).

- El mensaje es enviado desde el SME al SMSC
- Finalizado su procesamiento interno, el SMSC interroga al HLP y recibe la información de ruteo para el destino solicitado.
- El SMSC envía el mensaje al MSC usando la operación *forwardShortMessage*.
- El MSC lee la información del VLR. Esta operación podría incluir procedimientos de autenticación.
- El MSC transfiere el mensaje al MS.
- El MSC retorna el resultado de la operación *forwardShortMessage* al SMSC.
- Si el SME lo solicita, el SMSC le retorna un reporte de estado del envío del mensaje.

Aplicaciones SMS

SMS fue inicialmente diseñado para soportar mensajes de tamaño limitado, mayormente notificaciones y páginas alfanuméricas. Mientras que estas aplicaciones son y continuarán siendo ampliamente usadas, hay algunas otras facetas a las que el SMS puede actualmente sacarle provecho.

Pequeñas ráfagas de datos son el corazón de varias aplicaciones que estaban restringidas al mundo de las redes de datos de terminales fijas a una red local o una WAN (*wide-area network*). Sin embargo, muchas de estas aplicaciones representarían un mayor servicio en conjunto con un terminal con capacidad móvil. Así, un mozo podría cargar los gastos a la tarjeta de crédito de un cliente desde la misma mesa, en cualquier momento, en vez de ir hasta una terminal POS y hacer perder tiempo al cliente.

Además, la habilidad de rastrear un bien en circulación—como un camión o su carga, es muy valioso, para ambos, proveedor y cliente. Esta aplicación, nuevamente, necesita un simple intercambio de información mínima, en este caso, la latitud y longitud en un determinado momento del día, y quizás otros parámetros como la temperatura o humedad.

Esta aplicación no necesariamente requiere que la entidad monitoreada esté en movimiento. Los requerimientos son básicamente menores: datos capaces de ser transmitidos por ráfagas, y un área con cobertura digital. Por ejemplo, en un barrio, sería más rápido, fácil y barato, conducir un camión de la compañía eléctrica que interrogue a medidores inteligentes acerca de sus lecturas (para determinar el consumo de los clientes) y luego, reenviarlos vía SMS a una central de procesamiento para generar la facturación.

Otra familia de aplicaciones que pueden usar SMS como transporte de datos es el de la banca electrónica (*home-bankinkg*). No es un secreto que los cajeros electrónicos y las transacciones por Internet son menos costosas que las transacciones manuales en un banco. Las transacciones por Internet, son aún más baratas. De aquí que habilitar a los usuarios de dispositivos móviles el acceso al saldo de sus cuentas, a la transferencia de fondos entre ellas, el pago de sus facturas o su tarjeta de crédito, es también muy valioso, tanto para un cliente como para las instituciones.

El entretenimiento también es una de las buenas aplicaciones del SMS. Ejemplos de esto son los mensajes de texto entre dos partes (*texting*) o entre múltiples participantes

(*chat*). A esto, se le suma la entrega de información sobre eventos de espectáculo, promociones, descuentos, etc. que representen una proposición atractiva para los usuarios de terminales móviles.

La navegación inalámbrica, es una de las industrias en mayor evolución. Permite al usuario navegar en búsqueda de información sin las restricciones físicas de una PC. Revisar nuestro correo electrónico o saber qué película proyectarán esta semana en el cine sin tener que ir hasta un cibercafé o encender la computadora del estudio, suele ser una de las ventajas más apreciables, especialmente si no disponemos del tiempo para hacerlo.

El correo electrónico sigue siendo por lejos la aplicación más usada en el ambiente de los terminales móviles. Sin embargo, los dispositivos están evolucionando tan rápidamente y están incluyendo más y más funcionalidad para soportan nuevas aplicaciones, al mismo tiempo que aumentan su aspecto amigable. Probablemente, el próximo gran suceso detrás de la Web inalámbrica sea el *Internet shopping* y otras aplicaciones de *e-commerce* como cupones electrónicos, publicidad, etc.

Apéndice B. Glosario y Referencias

Términos y abreviaturas
Referencias

Apéndice B. Glosario y Referencias

Términos y abreviaturas

API
Application Programming Interface.

ASCII
American Standard Code for Information Interchange. ASCII es un estándar desarrollado por el *American National Standards Institute* (ANSI) para definir valores para los caracteres usados en textos. El conjunto ASCII de 128 caracteres incluye letras del alfabeto Inglés mayúsculas y minúsculas, números, signos de puntuación, y 33 códigos de control (como *tab*, *bell*, *carriage return*). ASCII usa 7 bits para representar cada caracter. Se podrán ver a los caracteres ASCII identificados por un número decimal desde 0 a 127. Los conjuntos de caracteres más grandes, conocidos como *extended ASCII* o *high ASCII*, usan todos los 8 bits, permitiendo así ser definidos otros 128 caracteres adicionales. Numerosas extensiones al *ASCII* han sido ideadas y unos pocos se han convertido en estándares nacionales o internacionales. Entre ellos, se destaca la familia de estándares internacionales, ISO-8859, que define extensiones apropiadas para ciertos grupos de lenguajes que ASCII solo no puede soportar. El miembro más importante de este grupo es el ISO-8859-1, conocido como ISO Latin-1, que fue diseñado para los lenguajes del oeste de Europa.

Bandwidth (Ancho de Banda)
El ancho de banda es la capacidad que tienen los medios de telecomunicaciones para transportar datos. Para las comunicaciones analógicas o de voz, el ancho de banda es medido como la diferencia entre las frecuencias de transmisión altas y bajas, expresada en ciclos por segundos, o Hertz [Hz]. Para las comunicaciones digitales, el ancho de banda y la velocidad de transmisión son usualmente tratados como sinónimos y medidas en bits por segundo [bps]. La velocidad actual o tiempo de transmisión de cualquier mensaje o archivo desde el origen hasta el destino dependen en un número de factores. La mayoría de las transmisiones en Internet viajan a muy alta velocidad en líneas de fibra óptica la mayor parte del camino. El ruteo (*switching*), pequeños anchos de banda en bucles locales en ambos extremos, y el tiempo de procesamiento de los servidores suman al tiempo total de la transmisión.

Bearer (Portadora)
Un servicio que permite transmisiones de datos digitales por aire. Las portadoras soportan diferentes protocolos basados en los formatos de dato, como SMS (*Short Message Service*), CSD (*Circuit Switched Data*), etc.

Bearer Adapter (Adaptador de Portadora)
Un módulo de software (una implementación del WDP) que le permite al WPS (*Wireless Protocol Stack*) trabajar en conjunto con una portadora en particular.

Byte
Una secuencia de bits consecutivos es tratada como una unidad. En casi todas la computadoras modernas, un byte está compuesto por 8 bits, aunque un se pueden encontrar otros valores. Para evitar ambigüedad, el término *octeto* se usa en lenguaje de los estándares internacionales para referirse a una unidad de 8 bits. Grandes cantidades de memoria son indicados en términos de kilobytes (1,024 bytes), megabytes (1,048,576 bytes), y gigabytes (aproximadamente 1 billón de bytes). Un disco que puede almacenar 1.44 megabytes, por ejemplo, es capaz de almacenar aproximadamente 1.4 millones de caracteres ASCII, o alrededor de 3000 páginas de información.

Bytecode (Código de bytes)
Una codificación de contenido, donde el contenido es típicamente un conjunto de operaciones (*opcodes*) de bajo nivel, por ejemplo instrucciones, y operandos para un hardware determinado (o virtual).

Card (Carta)
Es simplemente una unidad navegacional WML e interface de usuario. Una carta puede contener información para presentar al usuario o instrucciones para encargarse, por ejemplo, de las entradas de usuario.

CC/PP
Composite Capabilities/Preferences Profile

Client, Cliente
Dispositivo o aplicación que inicia un *request* de conexión a un servidor.

Common Gateway Interface (CGI)
Lenguaje de programación que permite el uso de formularios en un sitio Web.

Content Dispatcher
El *Content Dispatcher* es un modulo de propósito general diseñado para despachar mensajes a los manejadores de contenido destino (*WML, WMLScript, Multipart, SI, y SL*). En el inicio, el *Content Dispatcher* se registra con el *Application Dispatcher* para todos los mensajes cuyo tipo de contenido es soportado por el WAE.

Content Encoding (Codificación de contenido)
Usado como verbo, *content encoding* indica la acción de convertir datos de un objeto de un formato a otro. Típicamente el formato resultante requiere menos espacio físico que el original, es fácil de procesar o guardar, y es encriptado. Usado como un sustantivo, *content encoding* especifica un formato particular o codificación estándar o proceso.

Content Generator (Generador de contenido)
Servicio que genera o da formato a contenido. Típicamente, los generadores de contenido residen en los servidores origen.

CSD
Circuit Switched Data.

Deck (Mazo)
Colección de una o más cartas WML contenidas en un archivo de tipo wml. Un mazo WML es también un documento XML.

Device (Dispositivo)
Una entidad de red capaz de enviar y recibir paquetes de información que tiene una dirección de dispositivo única. Un dispositivo puede actuar tanto como un cliente o un servidor dentro de un contexto dado, o a lo largo de varios contextos. Por ejemplo, un dispositivo puede servir a un número de clientes (como servidor) y mientras tanto ser cliente de otro servidor.

Document Type Definition (DTD, definición de tipo de documento)
La definición del tipo de documento dice que elementos pueden estar anidados dentro de otros. Un DTD define:
• Los nombres y contenidos de todos los elementos que son permisibles en un cierto documento.
• La frecuencia con la que un elemento podría aparecer.
• El orden en que los elementos deben aparecer.
• Si la etiqueta de inicio o fin podrían ser omitidas.
• El contenido de todos los elementos, esto es, los nombres de los otros identificadores genéricos que están permitidos aparecer dentro de ellos.
• Los atributos y sus valores por defecto.

• Los nombres de los símbolos de referencia que podrían ser usados.

ECMA
European Computer Manufacturer's Association. Ver http://www.ecma.ch/ para más información.

ECMAScript
Lenguaje de script desarrollado por ECMA y definido en la especificación ECMA-262. ECMAScript está basado en JavaScript. Ver http://www.ecma.ch/ para más información.

Element (Elemento)
Entidad en XML que describe la estructura y el contenido de la información. Los elementos podrían contener una etiqueta de inicio, contenido, y una etiqueta de cierre, así como uno o más atributos.

Extensible Markup Language (XML)
El XML es un estándar de la World Wide Web Consortium (W3C) para los lenguajes de marcas de Internet, de entre los cuales WML es uno. XML es un subconjunto restringido de SGML.

GUI
Graphical User Interface.
HTTP (Hypertext Transfer Protocol)
HTTP es el protocolo subyacente usado por la World Wide Web. HTTP define como son formateados y transmitidos los mensajes, y que acciones deben realizar los servidores Web y los navegadores en respuesta a varios comandos. Por ejemplo, cuando se ingresa una URL en el navegador, un comando http es enviado al servidor Web dirigiéndolo a recuperar y transmitir la página Web solicitada.

IPC
Interprocess Communication.

JavaScript™
Un lenguaje estándar de facto que puede ser usado para agregar comportamiento dinámico a los documentos HTML.

MIME
Multipurpose Internet Mail Extension.

MMS (Multimedia Messaging Service)
Capacidad de un cliente WAP para recibir y procesar mensajes multimedia como sonido y video.

MTM (Message Type Module)
Módulo agregado que contiene un conjunto de componentes que soportan un protocolo específico de mensajes. Usado en el framework de mensajería multiprotocolo introducido en *EPOC Release 5*, el MTM maneja todas las interacciones con protocolos de comunicación de bajo nivel como el TCP/IP.

Multipart User Agent (Multipart Handler)
El encargado de procesar contenido multiparte.

Origin Server (Servidor de origen)
Servidor en el cual reside un servicio dado (o será creado), a menudo referido como *Web Server* o *HTTP Server*.

Push Access Protocol (PAP)
Un protocolo para especificar información de control y contenido que será "empujado" (*pushed*) a un cliente mediante un *Push Proxy Gateway* (PPG).

Push Framework
Término que describe el sistema *WAP Push* completo y abarcando los protocolos, interfaces de servicio, y entidades de software usadas para "empujar" contenido a un cliente WAP.

Push Initiator (Iniciador Push)
Servidor origen que envía contenido al *Push Framework* para la entrega a un *user agent* en un cliente WAP.

Push OTA Protocol
El protocolo *Push "Over the Air"*, es el protocolo usado para encaminar contenido *push* desde un *Push Proxy Gateway*.

Push Proxy Gateway (PPG)
Servidor responsable de aceptar contenido push desde un *Push Initiator* y transmitirlo a un cliente WAP.

Push Session (Sesión Push)
Una sesión WSP capaz de conducir operaciones *push*.

Rendering
Formateo y presentación de la información en la pantalla de un dispositivo móvil.

Resource (Recurso)
Objeto de datos en red o un servicio que puede ser identificado por una URL.

Server (Servidor)
Un dispositivo o (aplicación) que pasivamente espera pedidos de conexión de uno o más clientes. Un servidor podría aceptar o rechazar estos pedidos de conexión.

Service Indication (SI, Indicación de servicio)
Uno de los dos tipos de contenido usados para desarrollar un mensaje *push.*

Service Loading (SL, Carga de servicio)
Uno de los dos tipos de contenido usados para desarrollar un mensaje *push.*

Short Message Service (SMS)
Protocolo definido en un bearer que permite transmisiones punto a punto de mensajes cortos. Un mensaje corto es de 140 bytes (o 160 caracteres) de largo y usa un conjunto de caracteres especial de 7 bits; además mediante la concatenación de mensajes, puede ser usado para transmitir hasta 31 KB de texto binario o datos.

Standardized Generalized Markup Language (SGML)
Lenguaje de propósito general para especificar el dominio de un lenguaje de marcas. SGML está definido en el estándar ISO8879.
Tag (Etiqueta)
Una etiqueta es un término genérico para un elemento descriptor de un lenguaje. WML consiste de contenido rodeado de etiquetas de formato. Cada etiqueta es encerrada en un par de corchetes angulares: < y >. Las etiquetas son generalmente usadas en pares, una para comenzar el elemento y otra para terminarlo.

Terminal
Dispositivo que provee al usuario las capacidades de un User Agent, incluyendo la habilidad de requerir y recibir información. También llamado terminal móvil, dispositivo móvil o estación móvil.

UAPROF (User Agent Profile)
La especificación UAPROF define los mecanismos para describir y transmitir información sobre las capacidades de los dispositivos del cliente y las preferencias de los usuarios de tales dispositivos. Permite el flujo punto a punto de un *User Agent*

Profile (también referido como *Capability and Preference Information*, o *CPI*) entre el cliente WAP, puntos intermedio de la red, y el servidor origen.

Unicode
Un esquema de codificación para caracteres y texto. A diferencia del ASCII, que usa 7 bits para la codificación, usa 16 bits, lo que significa que puede representar más de 65000 caracteres únicos, un enorme incremento sobre la capacidad de códigos del ASCII de 128 caracteres. Fue creado y es mantenido por el *Unicode Consortium*, un grupo de corporaciones e instituciones involucradas en la computación internacional. Corresponde al ISO 10646, como estándar internacional *Universal Character Set (UCS).* En particular, los primeros 256 caracteres son idénticos a los definidos en ISO 8859-1 extendidos a 16 bits. Los valores Unicode son mostrados como cuatro dígitos hexadecimales precedidos por U+. Por ejemplo, U+0041 es la A mayúscula latina.

URI (Uniform Resource Identifier)
Un URI identifica un recurso en la Web como un documento, una imagen, un archivo descargable, un servicio, casillas de correo electrónico, u otros recursos. Un URI puede referir a un *Uniform Resource Locator (URL)* o a un *Uniform Resource Name (URN)*.

URL (Uniform Resource Locator)
Un URL especifica la dirección de un recurso en la Web. La sintaxis de un URL consiste de tres elementos:
• El protocolo, o el lenguaje de comunicación que usa la URL.
• El nombre de dominio, o nombre exclusivo que identifica a un sitio Web.
• El nombre de la ruta del archivo a ser recuperado.

User (Usuario)
Un usuario es una persona que interactúa con un *user agent*, para ver, oír o de alguna forma, usar un recurso.

User Agent (UA)
Un user agent es cualquier componente de software ejecutándose en un dispositivo móvil que interpreta contenido WAP identificado por el tipo de contenido que soporta, como WML, WMLScript, etc.

User Interface (UI, Interface del Usuario)
La capa de presentación de un sistema de software/hardware a través de la cual un usuario interactúa con el sistema, por ejemplo, para la entrada o recepción de datos.

vCAL (vCalendar)
Tipo de contenido WAP que representa un Calendario Electrónico (*Electronic Calendar*) y Formato de Agenda (*Scheduling Format*).

vCARD
Tipo de contenido WAP que representa una Tarjeta Electrónica de Negocios (*Electronic Business Card*).

VM (Virtual Machine)
Intérprete de código de bytes (*bytecode*).

WAE (Wireless Application Environment)
Especifica un ambiente de propósito general para aplicaciones basadas fundamentalmente en tecnologías y filosofías World Wide Web. WAE especifica un ambiente que permite a los operadores y proveedores construir aplicaciones y servicios que pueden alcanzar una amplia variedad de diferentes plataformas. WAE es parte del *Wireless Application Protocol*.

WAP (Wireless Application Protocol)
Especifica un framework de aplicación y protocolos de red para los dispositivos móviles. Las especificaciones del Glosario WAP extienden las tecnologías de trabajo en red (como *digital data networking standards*) y tecnologías de Internet (como XML, URLs, scripting, y varios formatos de contenido).

WBXML (Wireless Binary XML Content Format)
La especificación WBXML define una representación binaria compacta del XML. Está diseñado para reducir el tamaño de los documentos XML (como cartas WML) para una transmisión más eficiente a través de la red inalámbrica. El formato binario codifica la interpretación física de un documento XML, esto es, la estructura y el contenido del documento.

WDP (Wireless Datagram Protocol)
La capa WDP yace directamente encima de la capa bearer y de sus servicios, los cuales son soportados por varios tipos de red. WDP provee una interfaz común para los protocolos superiores como los de las capas de seguridad, transacción y sesión, que les permite funcionar independientemente de la red inalámbrica subyacente.

Web server (Servidor Web)
Servidor en el cual reside un servicio dado (o será creado), a menudo referido como *Origin Server* o *HTTP Server*.

Wireless Identity Module (WIM)
Un WIM es un modulo de hardware que se agrega a los dispositivos móviles cliente con el propósito de almacenar y procesar identificaciones de usuario y autenticar datos como claves privadas permanentes.

WML (Wireless Markup Language)
Es un lenguaje de marcas basado en WML y está destinado para su uso en la especificación de contenido e interfaces de usuarios para dispositivos de pequeño ancho de banda.

WMLScript
Un lenguaje de script usado para programar dispositivos móviles. WMLScript es un conjunto extendido del lenguaje JavaScript.

WPS
Wireless Protocol Stack (incluye manejo de sesión y todas las capas: WPS, WTP, WTLS y WDP)

WSP (Wireless Session Protocol)
Provee a las capas de aplicación WAP de nivel superior de una interface consistente para dos servicios de sesión. El primero es un servicio orientado a la conexión que opera sobre los protocolos de la capa de transacción, y el segundo es un servicio sin conexión que opera sobre un servicio de transporte de datagramas seguro o no-seguro.

WTA (Wireless Telephony Application)
WTA especifica un framework para soportar aplicaciones de telefonía inalámbrica: estas son aplicaciones con interface entre un Agente Usuario embebido en un dispositivo WTA y la infraestructura de la red de telefonía.

WTAI (Wireless Telephony Application Interface)
WTAI provee una interface que consiste de funciones de librería usadas por los User Agent WTA para conducir sesiones de telefonía (como llamadas y control de recursos), y que están disponibles tanto a través de WML como de WMLScript.

WTP (Wireless Transaction Protocol)
WTP es un protocolo liviano orientado a las transacciones que es apropiado para ser implementado en clientes livianos (dispositivos móviles). Opera eficientemente sobre redes inalámbricas y provee los servicios necesarios para el intercambio request-response entre un cliente y un servidor, operación que se conoce como transacción.

WTLS (Wireless Transport Layer Security)
WTLS es un protocolo a nivel de seguridad en la arquitectura WAP. La capa WTLS yace sobre la TLP y provee a las capas superiores del protocolo WAP un servicio de transporte seguro. Provee una interface para manejar conexiones seguras a través de las aplicaciones.

XML (Extensible Markup Language)
Un estándar de la World Wide Web Consortium (W3C) para los lenguajes de marcas de Internet, entre los cuales WML es uno. XML es un subconjunto restringido del SGML.

Referencias

http://allnetdevices.com	aND, allNetDevices
http://www.w3.org/Style/CSS	CSS
http://www.w3.org/Mobile/CCPP	CC/PP (w3c)
http://www.cnet.com/webbuilding	HTML to WML conversion
http://www.ecma.ch	ECMA
http://www.mobilegprs.com	GPRS-related documentation
http://www.iec.org	Internet Engenieering Consortium
http://forum.nokia.com	Introduction to WAP
http://buzz.builder.com	Jakob Nielsen
http://www.projectliberty.org	Liberty Alliance
http://www.openmobilealliance.org	Open Mobile Alliance
http://www.openwave.com	Openwave Systems Inc
http://www.w3.org/Mobile/CCPP	RDF Framework (UAProf, CC/PP)
http://www.w3.org/RDF	Resource Description Framework (w3c)
http://www.oasis-open.org	The XML Cover Pages
http://www.waphosting.com.sg	W@Phosting
http://www.wapforum.org	WAP Forum
http://www.wapforum.org	WAP Push specifications
http://www.phnet.fi/public/jiikoo	WAPDraw
http://www.wapit.com	Wapit Ltd.
http://www.wapmore.com	Wapmore
http://alphaworks.ibm.com	Wapsody
http://www.waptop.net/waptor	WAPtor (WAPtop.net)

http://www.waptunnel.com	waptunnel.com
http://wapulous.com	wapulous.com
http://www.trantor.de/wbxml	WBXML
http://www.wapforum.org	WAE specifications
http://www.wapforum.org	Wireless Profile CSS specification
http://forum.nokia.com	WML to XHTML Migration
http://www.w3c.org	World Wide Web Consortium
http://www.w3.org/TR/xhtml-basic	HTML Basic Specification
http://www.wapforum.org	XHTML Mobile Profile Specification
http://www.w3.org/XML	XML (w3c)
http://www.w3.org/TR/xpath	XPath (w3c)
http://www.w3.org/Style/XSL	XSL/XSLT (w3c)

Apéndice C. Código y archivos[13]

códigos fuente

Contiene el código fuente de los ejemplos que se muestran en el desarrollo del documento.

dtd

Contiene los DTDs (*Document Type Definitions*) especificados por la OMA y el Foro WAP.

Especificaciones WAP

Contiene la especificación WAP completa, incluyendo las versiones 1.x y 2.0, según el Foro WAP.

SMS

Contiene un ejemplo/aplicación de la función SMS (*Short Message Service*), incluye el código fuente en Java™.

uaprof_repository

Contiene los UAProfs (*User Agent Profiles*) para todos los dispositivos con soporte, según la OMA y el Foro WAP.

WAPpush

Contiene un ejemplo/aplicación de la tecnología WAP y la característica Push, incluye el código fuente en Java™.

WML Validator

Contiene un validador de código WML implementado a través de un parser XML en JavaScript, y los DTDs (*Document Type Definitions*) en sus versiones 1.1 y 1.3.

ZOOInfoPage

Contiene un ejemplo/aplicación usando SMS (*Short Message Service*), incluye el código fuente en Perl y JSP.

---

[13] Este Apéndice se encuentra en el CD-ROM que se adjunta con esta publicación.

Apéndice D. Siu WAP[14]

Siu WAP

Contiene el código fuente del Sistema de Información Universitario para dispositivos WAP realizado como resultado de este trabajo de investigación.

[14] Este Apéndice se encuentra en el CD-ROM que se adjunta con esta publicación.

Apéndice E. El RUP aplicado

Introducción

Modelo del dominio
Diagrama de clases
Glosario

Requerimientos y Modelo de casos de Uso
Requerimientos funcionales
Requerimientos no funcionales
Diagrama de casos de uso

Modelo del Análisis
Diagramas de clase

Modelo Navegacional
Mapa Navegacional

Modelo del Diseño
Diagrama de clases

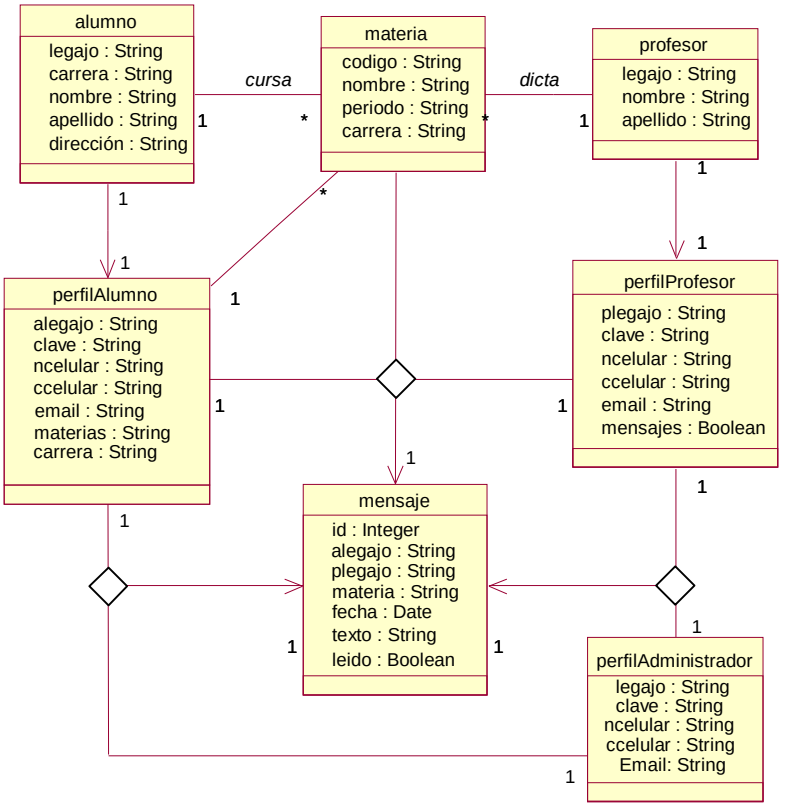
Apéndice E. El RUP aplicado

El Proceso Unificado de Rational, o RUP (Rational Software, Inc.) describe un proceso para el desarrollo de software, incluyendo las aplicaciones Web.

El Lenguaje Unificado de Modelado, o UML es el lenguaje gráfico para visualizar, especificar, construir y documentar los artefactos de un sistema con gran cantidad de software.

En esta sección se proponen algunos de los modelos utilizados en la construcción y el desarrollo de la aplicación que complementa a este trabajo de investigación.

Diagrama de Clases



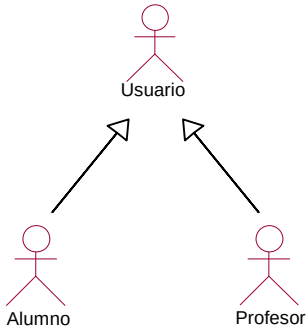
Glosario

Un usuario del sistema es o bien un alumno o un profesor. Para el sistema, una persona física se transforma en usuario una vez que completó el proceso de registración y se haya creado su perfil.

Se considera alumno a una persona física que se encuentra inscripto en una carrera (y por lo tanto tiene un legajo como alumno). Un alumno es un usuario que tiene un perfil de alumno.

Se considera profesor a una persona física que dicta alguna materia (y por lo tanto tiene un legajo como profesor). Un profesor es un usuario que tiene un perfil de profesor.
Un profesor podría ser también alumno. En tal caso, para el sistema, sería tratado como dos usuarios diferentes: un usuario alumno y un usuario profesor, dependiendo de cómo se registre cada vez que inicie su sesión.

Un administrador es la persona física que administra el sistema y los mensajes entre los usuarios, y que se encarga del mantenimiento del sistema.



Perfil, o perfil de usuario hace referencia a los datos personales y preferencias de un usuario, que lo identifican en el sistema.

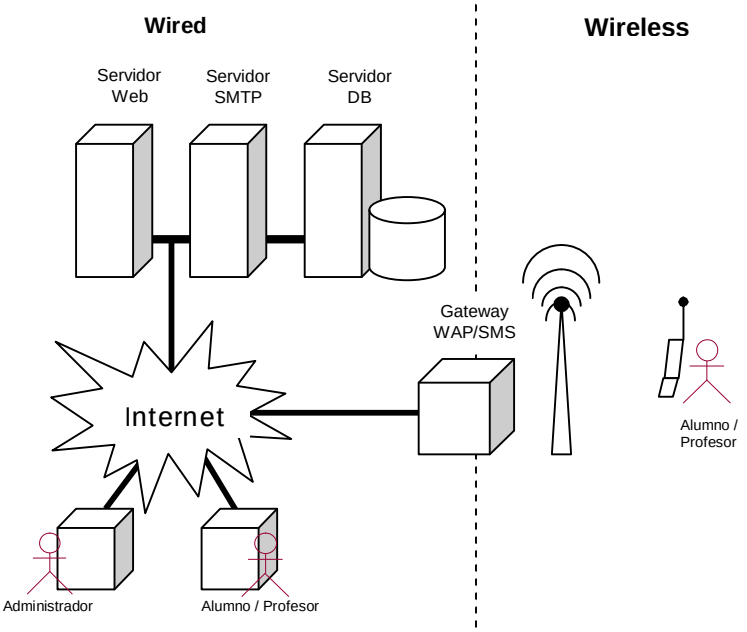
Un examen es una evaluación de cualquier tipo, por ejemplo: primer parcial, segundo parcial, recuperatorio del primer parcial, segundo final, etc. Para el sistema, un examen está compuesto por el código de la materia, el tipo de examen (primer parcial, segundo final, etc.) y la fecha.

Una nota es la puntuación que se le asigna a una evaluación. Para el sistema, una nota está compuesta por

el legajo del alumno, código de la materia, código de examen, fecha y puntuación.

Un mensaje es un texto dirigido a un usuario, y que puede contener una fecha de examen, una nota de examen, o algún aviso especial. Los dos primeros van dirigidos a alumnos y son generados por el sistema a pedido del profesor o el administrador, mientras que el último podría ir dirigido a cualquier usuario y generado solo por el administrador.

Las redes existentes, involucran al sistema y la tecnología subyacente que soporta las comunicaciones y las transferencias de información que realiza este sistema. Estas pueden ser diferentes a lo largo de toda una interacción usuario/sistema, dependiendo de la forma en la que estos lo accedan.



Requerimientos Funcionales

• El sistema deberá proveer acceso Web/WAP a alumnos y profesores como usuarios de un sistema de información.

• El sistema deberá proveer a un administrador único, una interface Web para la administración del sistema.

• El sistema deberá permitir a un alumno registrarse como usuario del sistema WAP, ingresando sus datos y preferencias, también mediante una interface Web.

• El sistema deberá permitir a un alumno modificar su perfil de usuario.

• El sistema deberá permitir a un alumno la consulta de mensajes de las materias en las que tenga interés, previo login.

• El sistema deberá permitir a un profesor registrarse como usuario del sistema WAP, ingresando sus datos y preferencias, también mediante una interface Web.

• El sistema deberá permitir a un profesor modificar su perfil de usuario.

• El sistema deberá permitir a un profesor la carga de fechas de examen de las materias que dicta, previo login.

• El sistema deberá permitir a un profesor realizar la carga de notas de examen de una materia que dicta, previo login.
• El sistema deberá permitir a un administrador ver los mensajes en el sistema.

• El sistema deberá permitir a un administrador eliminar mensajes en el sistema.

• El sistema deberá permitir a un administrador enviar mensajes especiales a cualquier conjunto de usuarios del sistema WAP.

• El sistema deberá permitir encolar mensajes en la cola de sistema para luego enviarlos de forma automática a

los alumnos. El sistema deberá mantener una base de datos con estos mensajes.

Requerimientos no funcionales

• Usabilidad
El usuario debe contar con un servicio de telefonía celular y con un dispositivo móvil con soporte SMS o e-mail WAP para recibir los mensajes que el sistema le envíe. Alternativamente, un usuario que no cumplimente los requisitos, podría optar por recibir los mensajes a una casilla de correo electrónico convencional.
El sistema deberá ser accesible por cualquier navegador Web estándar o un dispositivo móvil con soporte de navegación WAP (y el servicio contratado).
El sistema deberá contar con una interfaz de usuario intuitiva, de fácil uso, y óptima para la navegabilidad con la menor cantidad de interacción por parte del usuario.

• Performance
La navegación WAP deberá estar optimizada para minimizar tiempos (y costos) de la interacción usuario-sistema.
El sistema deberá enviar los mensajes de la cola con la mayor prontitud posible, y los mensajes deberán tener un formato optimizado para garantizar el menor uso del canal.

• Confiabilidad
Los tiempos de retraso y la confiabilidad en el envío de mensajes dependen de las redes existentes sobre las que funciona el sistema. La seguridad sobre los datos que maneja el sistema está garantizada por la base de datos que los mantiene.

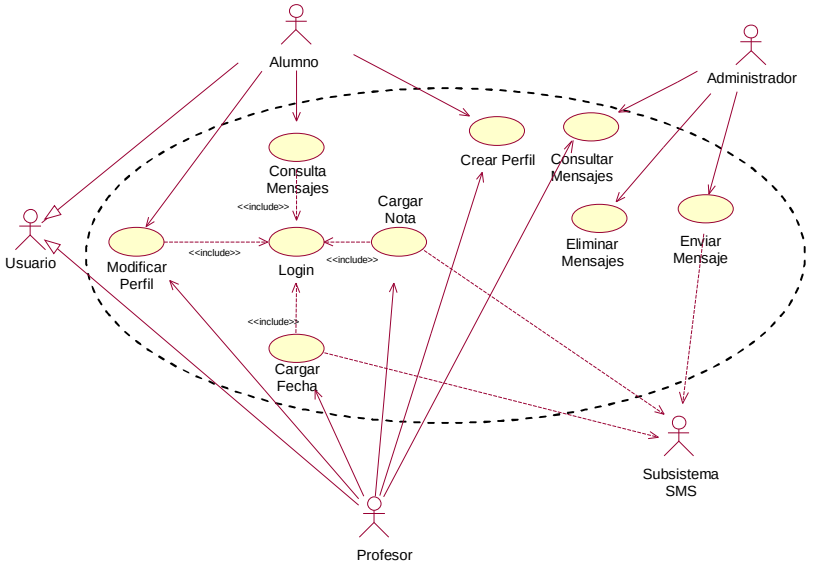
• Seguridad
El sistema debe garantizar la seguridad en cuanto a los perfiles de usuario (y los datos que estos contienen), así como los mensajes que este maneja.
El resto de la seguridad, esto es, en la transacción envío de mensaje, es ajena al sistema, y depende de la seguridad de las redes que conforman el sistema. No obstante, las redes existentes presentan un alto grado de seguridad.
El acceso al sistema también está controlado mediante la validación de los usuarios.

• Hardware

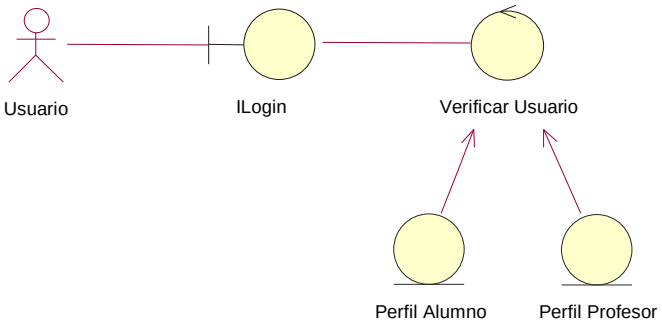
Los clientes deberán contar con un dispositivo móvil con soporte SMS/e-mail WAP para recibir los mensajes de sistema y opcionalmente soporte de navegación WAP para navegar el sitio. Alternativamente la navegación se puede efectuar desde una PC de escritorio y a través de un browser estándar, así como recibir los mensajes en una casilla de e-mail convencional, o consultarlos a través del mismo navegador Web.

El hardware mínimo requerido por el servidor del sistema es una PC con conexión a Internet y capacidades de servidor (servidor Web, servidor de e-mail/SMS, y servidor de Base de Datos). El rendimiento del sistema es dependiente del volumen de información y cantidad de transacciones que se generen sobre el servidor.

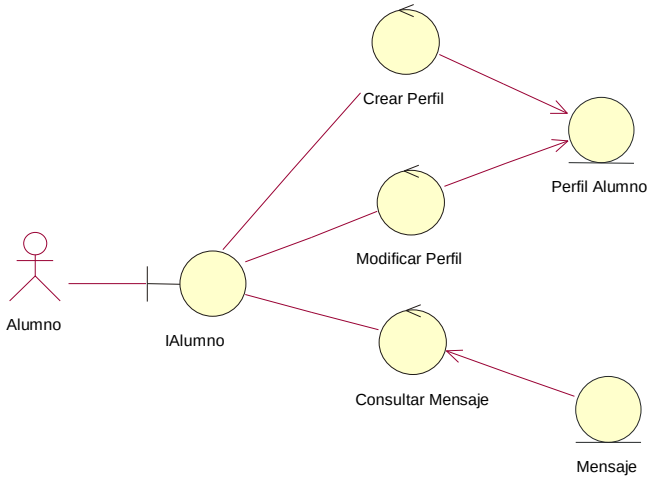
Diagrama de Casos de Uso



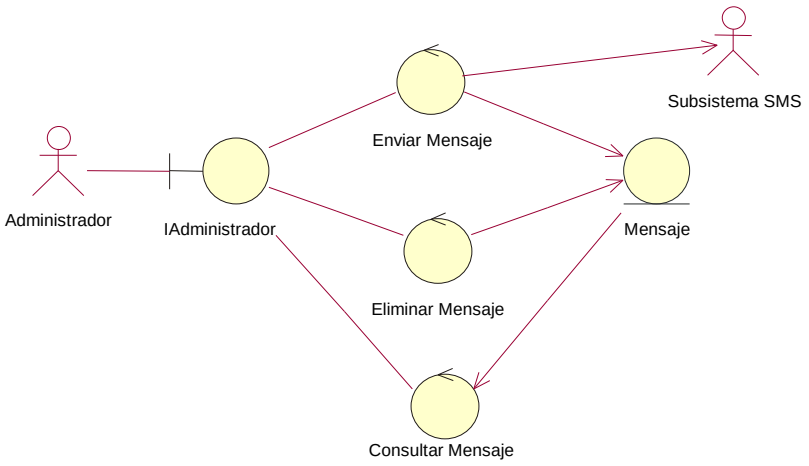
Usuario: Login



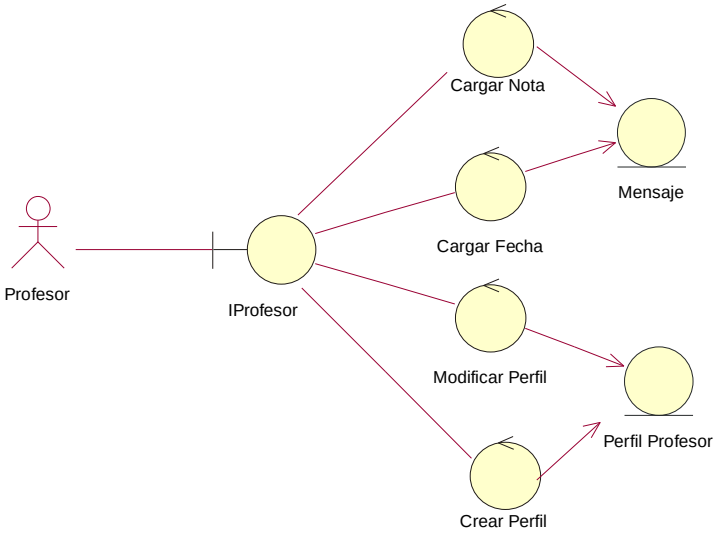
Alumno: Crear Perfil, Modificar Perfil y Consultar Mensaje



Administrador: Enviar, Consultar y Eliminar Mensajes



Profesor: Cargar Nota, Cargar Fecha, Crear y Modificar Perfil



Mapa Navegacional

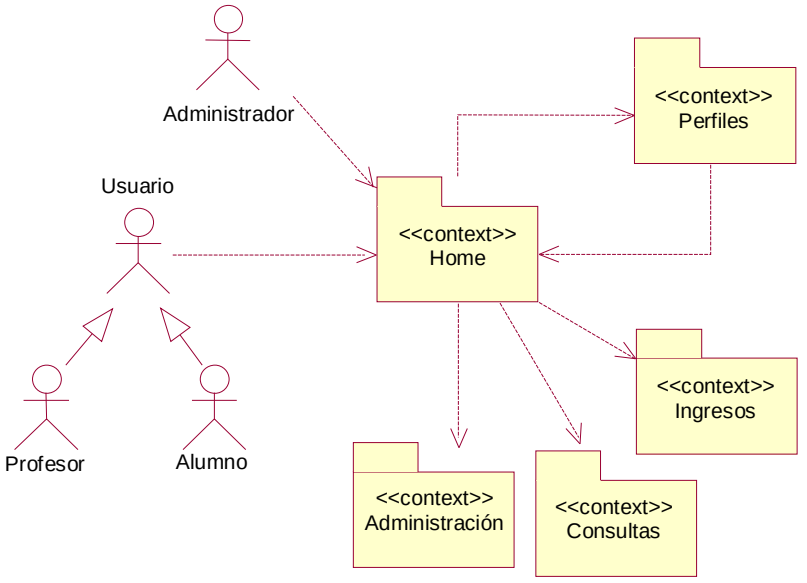


Diagrama de Clases (A1)

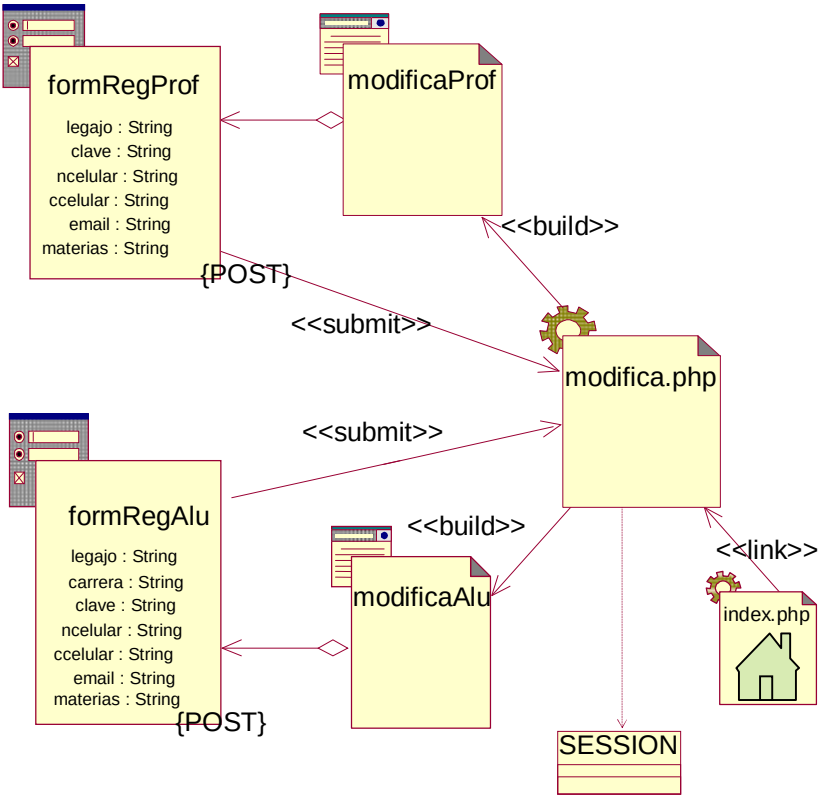


Diagrama de Clases (A2)

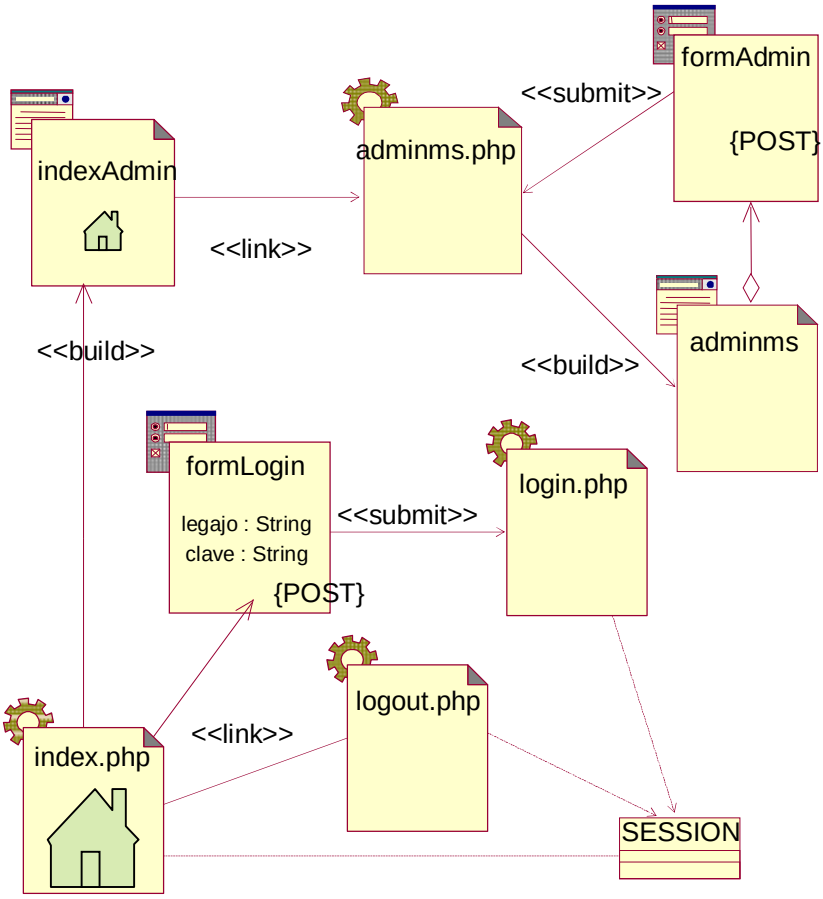


Diagrama de Clases (B1)

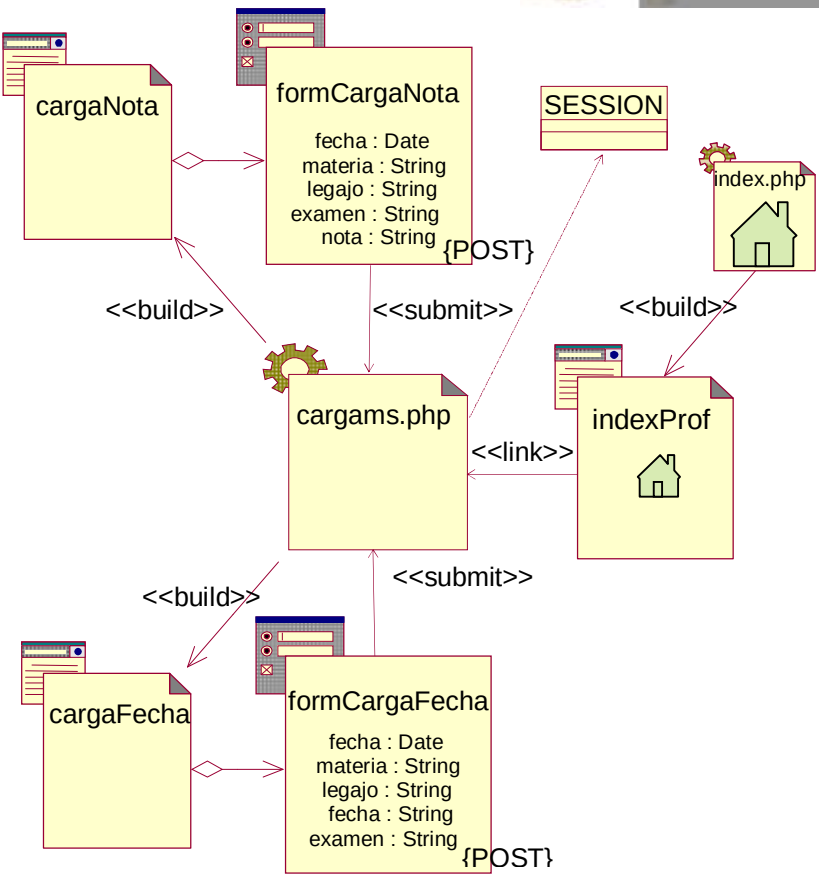


Diagrama de Clases (B2)

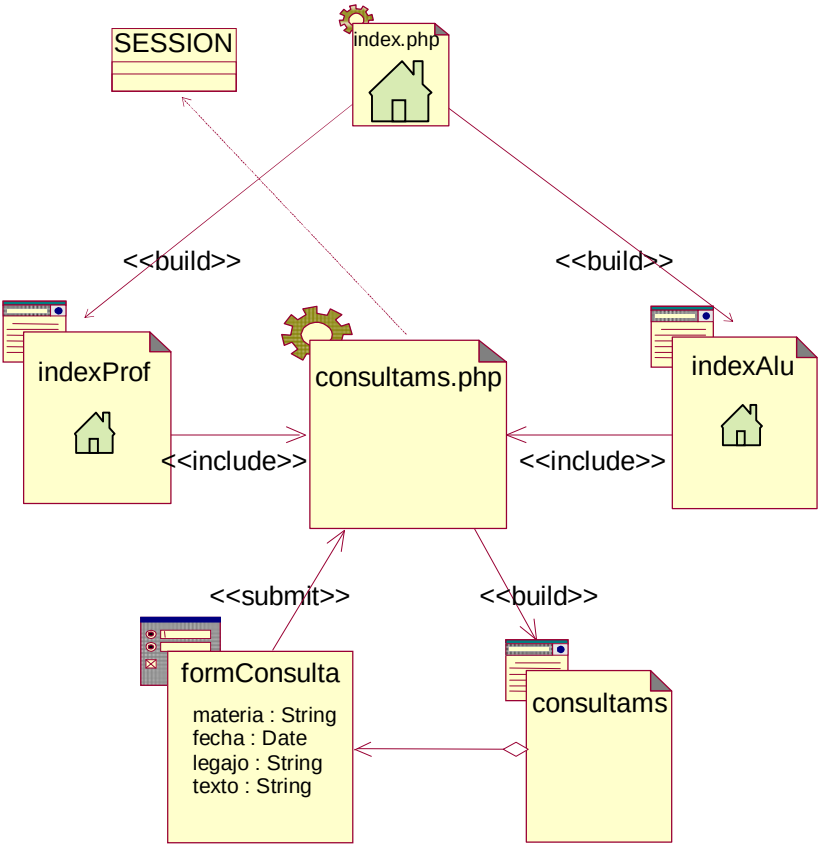
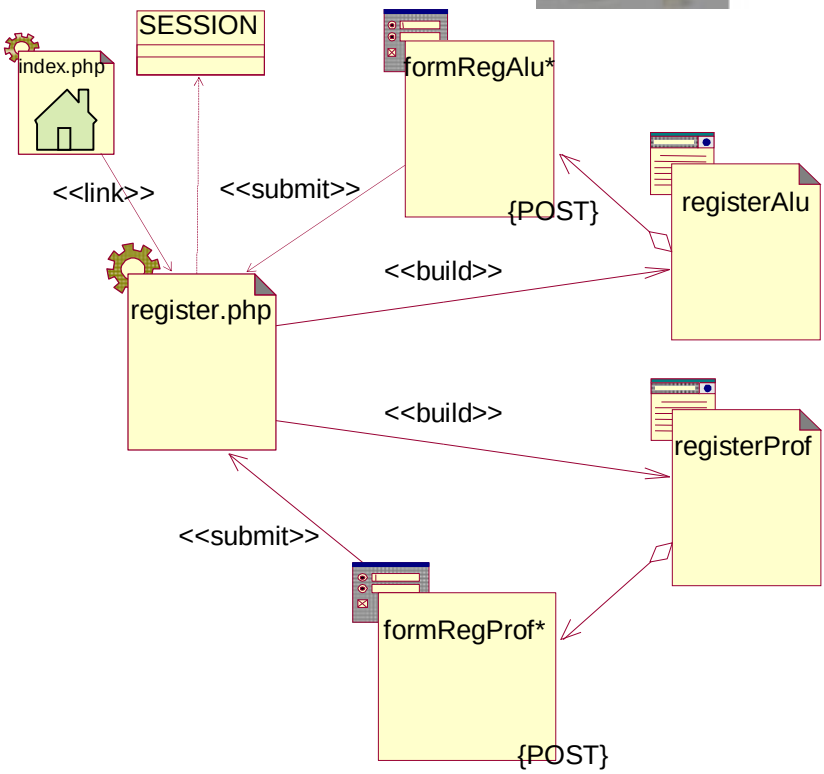


Diagrama de Clases (B3)



General Pico, La Pampa
Argentina
septiembre de 2003