Capítulo 1

Desarrollos móviles

En este capítulo haremos un repaso general de todas las tecnologías disponibles, los dispositivos del mercado, qué tipo de soluciones podemos desarrollar y sobre cuáles podremos trabajar con las herramientas que nos provee .NET.

DISPOSITIVOS MÓVILES

Antes de comenzar a hablar sobre los desarrollos móviles, comencemos por definir en qué tipo de equipos estamos pensando cuando hablamos de dispositivos móviles. Consideraremos equipos móviles a aquellos dispositivos que los usuarios pueden llevar consigo y que se caracterizan por tener un tamaño reducido, que caben en la palma de la mano y en el bolsillo. Son asistentes personales, pequeñas computadoras y teléfonos celulares, que no llegan al tamaño y robustez de una *notebook*.
Estos equipos tienen ciertas características (la mayoría de ellas, limitaciones) que los hacen muy distintos de lo que conocemos como desarrollos para equipos de escritorio.
En primer lugar consideremos las características de hardware: trabajaremos con pantallas chicas (un promedio de 200 x 200 pixeles), no siempre tendremos teclado para interactuar con el usuario, los procesadores no serán muy poderosos (entre 16 y 500 MHZ), funcionan a batería de limitada duración y no poseen discos duros, sino escasa memoria (entre 2 y 64 MB).
En cuanto a las funciones disponibles, en su mayoría tendremos acceso a protocolos de red, como **TCP/IP**, ya sea a través de Internet móvil (como **GPRS**), de tecnología **Bluetooth**, **Wi-Fi** o de sincronización en un puesto de trabajo. Todos los equipos poseen un sistema operativo reducido en capacidades y ninguno de ellos tiene un sistema de archivos (*file system*) como lo conocemos en equipos de escritorio. Generalmente, se trata de sistemas de almacenamiento de registros de datos que se mantienen latentes en la memoria principal del equipo.
Con todas estas limitaciones nos tendremos que enfrentar a la hora de desarrollar para equipos móviles. Una buena noticia: las plataformas móviles están enfocando su visión hacia lograr una unificación con equipos de escritorio en cuanto a servicios brindados por el sistema operativo o por el lenguaje utilizado. La tendencia es a aprovechar el conocimiento y código que los programadores tenemos en aplicaciones de escritorio; es así que podemos programar en **Java** casi tal cual como si lo hiciéramos con **J2SE** o podemos utilizar **.NET Compact Framework**, que veremos en este libro.
Algo sí es común a todas las plataformas: la tendencia a la programación orientada a objetos y/o eventos. Así, por ejemplo, será posible reutilizar clases ya creadas en .NET en equipos móviles. Veamos entonces las diferentes plataformas existentes en

LOS DIENTES AZULES

Bluetooth es una tecnología de comunicación inalámbrica, pensada especialmente para dispositivos electrónicos de bajo porte. Permite comunicar en un radio de 10 metros, por ejemplo, un celular con un PDAs o con un equipo de escritorio.

la actualidad para luego interiorizarnos en los tipos de desarrollos que podremos realizar en cada una, y analizar sobre cuáles podremos desarrollar utilizando las herramientas que nos provee Microsoft.

PDAs

Los **PDAs** (*Personal Digital Assistants*) o **Asistentes Personales Digitales**, son equipos que caben en una mano y que tienen algunas características en común:

- Pantalla grande en proporción al tamaño del equipo.
- Soporte de pantalla *touch-screen* (sensible al tacto) a través de un *stylus* (lápiz que hace las acciones de un mouse).
- En general, carecen de un teclado físico.
- Permiten el ingreso de texto a través de escritura manual utilizando el stylus.
- Proveen métodos de sincronización con un equipo de escritorio.
- Permiten comunicarse (vía infrarrojo, *Bluetooth* o un cable) con un teléfono celular y acceder a Internet.
- Poseen un sistema operativo con software utilitario y es posible instalarles aplicaciones adicionales.

PalmOS

Los equipos que trabajan con el sistema operativo **PalmOS**, más conocidos como **Palm** (que es una marca registrada, a pesar de lo que se cree comúnmente), son los que actualmente están más difundidos si hablamos de equipos PDAs. Quien comenzó con estos equipos fue una división de la empresa **US Robotics** en la década del 90, que luego se transformó en **3COM**, luego se dividió y se formó **Palm Computing**; y el último cambio (hasta ahora) fue la creación de la empresa **PalmOne** que está a cargo del hardware de los dispositivos, y **PalmSource** que es responsable del desarrollo del sistema operativo.

***Figura 1.** La **Palm m100** fue una adaptación más moderna de la clásica **Palm III**, una de las más vendidas cuando este tipo de dispositivos comenzó a ser popular en el mercado mundial.*

Muchos equipos de otras marcas también utilizan este sistema operativo, como **IBM, Sony, Handera** y **Kyocera**.
Existen diversas versiones del sistema operativo y del hardware de este tipo de dispositivos. La primera versión (**Figura 1**) más popular fue la **3** (todavía actualmente existen muchos dispositivos con esta versión, como ser **Palm III**, **Palm V**, etc.), luego surgió la **versión 4** (la línea **m100**, **m 500**), la **versión 5** conocida como **Garnet** (**Tungsten**, **Treo**, **Zire**) y ya está en el mercado la **versión 6** (**Cobalt**), que incorpora muchas novedades en cuanto a capacidades multimedia y de interconexión.

Los equipos PalmOS siempre se caracterizaron por un bajo consumo de batería, procesadores no muy potentes, memoria disponible limitada y capacidades de sincronización con equipos de escritorio. Ya todas las versiones del sistema operativo incluyen un navegador web y en el mercado existen muchos otros productos que podemos descargar e instalar.

***Figura 2.** Las Palm evolucionaron y la última versión de este equipo ya posee cámara de fotos integrada, kit de software para Internet y mucho más. Las aplicaciones antiguas siguen funcionando.*

Pocket PC

Las **Pocket PC** son la solución PDA que ofreció la plataforma Microsoft desde sus comienzos. Estos dispositivos son similares a los equipos Palm, con la gran diferencia de que siempre han gozado de mejor procesador, mayor cantidad de memoria, mejor resolución de pantalla y, en contrapartida, mayor costo (actualmente se está emparejando) y mayor consumo de batería.
En esta categoría entran equipos de diversas marcas, como ser **HP** con su **iPaq** o **Toshiba**. La ventaja de estos equipos es que proveen un sistema operativo Windows reducido. Esto permite a quienes saben utilizar el Windows de escritorio, acostumbrarse fácilmente a la interfaz de estos equipos. La versión del sistema operativo es conocida como **Windows CE** y ha tenido diferentes versiones durante su evolución.

{ } NOTEBOOKS CON CE

El sistema operativo **Windows CE** fue utilizado en algunos equipos de notebooks ultrafinas con pantallas de 10 ó 12 pulgadas. Sin embargo, no han tenido mucho éxito debido a que el sistema operativo y las aplicaciones disponibles son muy limitados.

La principal característica es que incluye versiones reducidas de aplicaciones muy conocidas en la PC: **Pocket Internet Explorer**, **Pocket Word**, **Pocket Excel** y hasta una versión de **Windows Media** con la que podremos ver contenido local o *streaming*.

***Figura 3.** El clásico menú **Inicio** (**Start**) no podía faltar en una versión reducida de Windows. Recordemos que sobre estos equipos trataremos en la segunda mitad del libro.*

La ventaja en cuanto a desarrollar para estos equipos es que tendremos, como veremos a lo largo de este libro, las mismas herramientas que utilizamos para desarrollar aplicaciones de escritorio.

Handheld PC

Las **Handheld PC** o "*PC de mano*" son una evolución de las PDAs, que no ha tenido gran aceptación en el mercado. Son equipos estilo agenda electrónica, que se abren y poseen la pantalla en la parte superior y un pequeño teclado en la parte inferior. Windows CE fue también el sistema operativo más utilizado en este tipo de dispositivos. La mayor ventaja frente a las Pocket PC es que poseen teclado y pantalla más grandes.

***Figura 4.** Las **Handheld PCs** tienen la ventaja de poseer pantallas hasta medio VGA, 640 x 240, lo que permite ver páginas web casi como en una PC.*

Teléfonos

Todos ya nos imaginamos qué son los teléfonos. Son teléfonos celulares "inteligentes", que tienen mayor capacidad digital además de ser usados como canal de voz. Esta categoría de equipos móviles tiene sus coincidencias con los PDAs (en cuanto a las limitaciones), pero también sus propias características que la hacen diferente a la hora de pensar en desarrollar para este tipo de equipos:

- No poseen stylus ni pantallas touch-screen.
- Están pensados para ser manejados con una sola mano.
- La pantalla es, generalmente, mucho más chica que una PDA (entre 90 x 60 y 200 x 150 pixeles).
- No hay sistemas operativos dominantes en el mercado.
- Tienen conectividad a Internet, a través de servicios de datos (como **GPRS** o **EDGE**).

Veamos qué plataformas hay disponibles en esta categoría de dispositivos móviles.

Plataforma propietaria

Éstos son la mayoría de los equipos celulares. Tenemos equipos fabricados por **Nokia**, **Motorola**, **Siemens**, **Samsung**, **Sony Ericsson**, entre otros. Cada uno de ellos desarrolló su propio sistema operativo, su propio navegador y sus propias aplicaciones. Si bien algunos equipos (de gama alta) tienen alguno de los sistemas operativos que veremos a continuación, la mayoría de los teléfonos no lo poseen. La única constante relativa entre todos estos equipos, lo cual nos ayudará a desarrollar aplicaciones, es el soporte de máquina virtual **Java** en muchos de ellos, y el soporte de un navegador de páginas.

La desventaja es que dicho navegador puede soportar diversos lenguajes, **WML**, **HTML**, **cHTML**, **iHTML**, **xHTML** y otros que veremos luego. Tendremos que saber cuál utilizar para cada equipo o, en unas páginas más, ver qué nos ofrece .NET para solucionar este problema.

Figura 5. *Aquí vemos tres teléfonos celulares de marcas distintas, cada uno con su propio sistema operativo y su propia interfaz de uso.*

Symbian OS

Originalmente, también competencia de Palm en PDAs, este sistema operativo se está imponiendo como uno de los más prometedores para usar en equipos celulares.
Actualmente, lo podemos encontrar en equipos celulares de alta gama y tiene la ventaja de poder instalarle diversas aplicaciones con las características propias de un sistema operativo multitarea. Este sistema se encuentra en equipos **Nokia**, **Serie 60** y en equipos de otras marcas. Sobre este sistema operativo es posible instalar, por ejemplo, versiones de **Flash Player**, de **Real One Player** (para video streaming) y del navegador **Opera**.

***Figura 6.** Los celulares con Symbian OS se caracterizan por tener una pantalla muy grande respecto de otros teléfonos celulares.*

Windows Smartphone

Microsoft no se ha quedado atrás en este aspecto y ha creado una versión de Windows CE (actualmente también se lo conoce como **Windows Mobile**), llamada **Smartphone Edition**. Esta versión del sistema operativo es similar a la utilizada por Pocket PC, pero teniendo en cuenta sus diferencias, ya no tendremos soporte de stylus para navegar por la pantalla y tendremos que navegar por los menús y por las aplicaciones con las teclas de cursor y con dos teclas de acción. De esta manera, tendremos disponible ahora el menú **Inicio** del sistema operativo, al pulsar una de las teclas de función del teléfono en uso. La desventaja de este tipo de equipos celulares es que todavía no se encuentra muy difundido entre los usuarios de telefonía celular en nuestra región.

***Figura 7.** Los teléfonos inteligentes con Windows están soportados por varias empresas en el mercado de telefonía celular.*

LINUX MOBILE

El conocido sistema operativo del pingüino está ingresando al mundo de dispositivos móviles de a poco. Ya podemos encontrar algunos equipos PDAs y celulares con **Linux** como sistema operativo.

Otros dispositivos

Los dispositivos móviles se están diversificando cada vez más y podremos encontrar nuevas categorías de este tipo de equipos. Veamos cuáles ya están disponibles, en las que también podremos realizar algún tipo de desarrollo.

Automóviles

Será cada vez más común que un automóvil esté equipado con un dispositivo que permita la ejecución de aplicaciones, la conexión a Internet y la conexión al posicionamiento global (GPS).
En esta categoría, existen diversas soluciones, aunque una de las más difundidas es Windows CE, a través de lo que se conoce como **AutoPC**.

Figura 8. *No debemos olvidarnos de los automóviles a la hora de pensar en qué tipo de dispositivos podemos desarrollar aplicaciones.*

SPOTs

Si bien es un concepto de Microsoft, próximamente veremos otro tipo de plataformas que ofrecen productos similares. Los **SPOTs** (*Small Personal Object Technology*) son pequeños dispositivos (como relojes) con capacidad de conexión inalámbrica.

Actualmente ya existen en el mercado algunos de estos dispositivos que funcionan a través de radiofrecuencias (como un pager) y ofrecen servicios varios de MSN (Hotmail, Messenger, etc.). Su ventaja radica en que son capaces de ejecutar .**NET Byte Code**, por lo que podemos estar atentos y listos para cuando sean masivos.

Figura 9. *Cada vez nos asombramos menos con la tecnología, pero igual no deja de ser admirable ver los mensajes del* ***Messenger*** *de* ***Microsoft*** *en nuestro reloj.*

Tarjetas inteligentes

En esta categoría, entran diversas tecnologías pensadas para tarjetas (de crédito, de débito, de efectivo electrónico, de seguridad, etc.) e incluso otro tipo de tarjetas más pequeñas con fines específicos, como la **SIM Card** de los teléfonos celulares **GSM**. Este tipo de tarjetas, de muy baja capacidad de memoria (64Kb en general) y sin método directo de interfaz con ellas (podemos utilizarlas a través de un cajero automático o el teléfono celular que la posee), nos permite desarrollar pequeñas aplicaciones en ellas. Una de las tecnologías más difundida por ahora es **Java Card**.

***Figura 10.** Si, tampoco esta pequeña tarjeta se salvará de que nuestras manos le programen aplicaciones para ser utilizadas en diversos ámbitos.*

Cámaras digitales

Si bien en primera instancia nos parece extraño pensar en una cámara digital con soporte de algún tipo de desarrollo, pensemos que son equipos digitales con soporte de display LCD color, memoria y teclas de navegación y, ciertamente, ofrecen un pequeño conjunto de aplicaciones y funcionalidades a través de su sistema operativo. No será raro que comiencen a aparecer herramientas de desarrollo para este tipo de dispositivos.

Reproductores multimedia

Ya son un tipo de dispositivo más de los que un usuario aspira a poseer. Los reproductores personales de MP3, video y multimedia en general son equipos con alta capacidad de memoria, pantalla LCD (generalmente pequeña y monocromática) pero que también poseen un procesador y un sistema operativo. Desde pequeños reproductores de MP3 con displays de 2 líneas de texto hasta dispositivos más complejos, como **IPODs** o dispositivos con **Windows Media**, ¿qué nos impide que podamos, por ejemplo, leer un documento de texto o recibir alguna noticia en nuestra pantalla?

SERIE 60

Si bien la denominación **Serie 60** fue originalmente una categorización de los productos de **Nokia**, se ha convertido en un estándar para denominar a equipos celulares con sistema operativo **Symbian**, para lo cual se formó una organización entre varias empresas fabricantes en **www.series60.com**.

Híbridos

No todo es absoluto en este mundo, y así existen muchos híbridos entre las tecnologías que estuvimos analizando. Podremos encontrar teléfonos celulares con **PalmOS**, equipos **PDAs** con **Symbian** o directamente **PDAs** que poseen conexión celular consigo (como la línea **Palm Treo** o **Pocket PC Phone Edition**). Estos híbridos no son los más comunes aunque, de a poco, serán seguramente la convergencia de la cual se habla desde hace años: un solo dispositivo móvil que sea PDA, celular, cámara digital y reproductor multimedia.

***Figura 11.** Los dispositivos móviles no están exentos de participar de más de una categoría. Aquí vemos un equipo Palm que al mismo tiempo es un teléfono celular con las ventajas de ambos mundos.*

DESARROLLOS: TIPO DE SOLUCIONES

Cuando pensamos en desarrollar una solución para dispositivos móviles, lo primero que hay que considerar es qué tipo de solución será conveniente para la necesidad dada. Esto dependerá de que se requiera conexión o sincronización con un servidor central, de la diversidad o no de equipos de los usuarios del sistema y de la capacidad de interacción con el equipo que precisemos.
Así, podemos escoger entre algunos de los siguientes tipos de soluciones: **soluciones Stand-alone**, **soluciones Online** o soluciones conocidas como **Smart Clients**.

Soluciones Stand-alone

Las aplicaciones **Stand-alone** son aquéllas que se desarrollan para ser instaladas y ejecutadas sobre el equipo móvil en cuestión y que funcionan en forma desconectada de Internet o de un servidor central.
Para desarrollar una solución de este tipo, debemos generar un paquete ejecutable en el formato correcto para el tipo de sistema operativo sobre el cual será instalado, así como también por la versión del mismo.

Cada sistema operativo es diferente y hasta el hardware sobre el que se ejecuta es distinto; es por ello que una aplicación desarrollada para Palm OS es completamente distinta de una para Pocket PC o para Symbian OS.

Las ventajas de este tipo de soluciones son:

- Ejecución veloz.
- Aprovechamiento de características de bajo nivel de cada equipo.
- Uso de todas las herramientas, controles y accesos que ofrece el dispositivo.
- Manejo de memoria.
- Soporte de sincronización con un equipo de escritorio.
- Se puede trabajar sin necesidad de estar conectado.

Las desventajas son:

- Se deben desarrollar diferentes versiones para cada sistema operativo.
- Se deben instalar manualmente en cada equipo.
- No pueden soportar grandes cantidades de información para búsqueda o almacén.
- No pueden consultar o trabajar con centros de datos remotos.

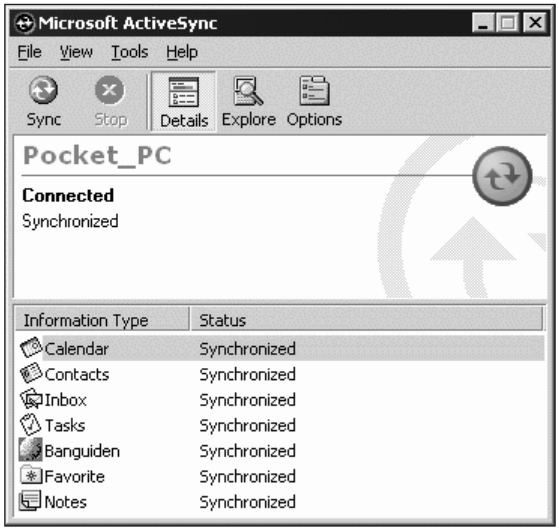


***Figura 12.** Las aplicaciones Stand-alone permiten la sincronización ida y vuelta con un equipo de escritorio y así, actualizar la información que tiene almacenada.*

C++ REY DEL NATIVO

Existen herramientas para desarrollar bajo código nativo para todas las plataformas: **Windows Mobile**, **Palm OS** y **Symbian OS**. Todas tienen oficialmente a **C++** como lenguaje.

Soluciones Online

Una aplicación móvil **Online** o conectada es, en realidad, una solución a través de Internet, utilizando páginas web o wap para la interfaz de la misma en el equipo, y toda la ejecución se realiza en el servidor.
En este caso podemos lograr una mayor compatibilidad que en las soluciones Stand-alone, dado que un mismo lenguaje, por ejemplo **HTML**, es comprendido por varios dispositivos.

No obstante ello, no existe un solo lenguaje actualmente difundido en dispositivos móviles: tenemos **WML**, **HTML**, **cHTML**, **xHTML**, por ejemplo, que tienen sus pequeñas diferencias y, dentro del mismo lenguaje, existen pequeñas variantes en cada modelo de teléfono.

Las ventajas de utilizar este tipo de desarrollos son:

- Mayor compatibilidad con diferentes modelos y sistemas operativos.
- No es necesario distribuir ni instalar ninguna aplicación.
- Podemos utilizar la aplicación en sistemas operativos propietarios que no permiten la instalación de aplicaciones Stand-alone.
- Podemos realizar cálculos y algoritmos complejos dado que la ejecución se realiza en el servidor.
- Se puede trabajar con gran cantidad de información.

Las desventajas son:

- No se puede acceder a capacidades de bajo nivel del equipo.
- Se necesita estar conectado para poder utilizarlo (se debe tener señal en el caso de equipos celulares).
- No se pueden usar todos los controles de ingreso disponibles en el equipo, sólo los propuestos por el lenguaje en cuestión (**HTML** o **WML**, por ejemplo).
- Ejecución más lenta (se debe recargar la información contra el servidor).

EL PROBLEMA DEL LENGUAJE

Como comentamos, los equipos móviles soportan diversa cantidad de lenguajes, según marca y modelo. En los próximos capítulos veremos cómo la tecnología de **ASP.NET** nos permitirá sortear esta barrera muy fácilmente.

Figura 13. *Las aplicaciones Online siempre se ejecutarán en un browser (navegador), ya sea WAP o Web. Así, podremos trabajar con equipos que no tienen otra forma de ejecutar una solución.*

Soluciones Smart Client

Una aplicación **Smart Client** (*cliente inteligente*) junta lo mejor de dos mundos, Stand-alone y Online. Este tipo consta de aplicaciones ejecutables que se distribuyen e instalan en los equipos, pero que también utilizan la conexión para comunicarse e interactuar con un servidor.
La "inteligencia" radica en que la aplicación debe ser capaz de seguir ejecutándose aun cuando el equipo pierda conexión con el servidor (generando *buffers* de información, por ejemplo).

Algunas ventajas de utilizar Smart Clients:

- Junta lo mejor del mundo conectado y del desconectado.
- Permite consultar grandes capacidades de información y hacer uso de funciones de bajo nivel de los equipos.
- Permite seguir trabajando cuando el equipo se desconecta.

Algunas desventajas:

- Es más difícil a la hora de desarrollar las aplicaciones, al pensar de qué manera trabaja la aplicación online u offline sin que el usuario pueda percibir la diferencia.
- Se debe crear el cliente basándose en cada tipo y versión de sistema operativo.
- Se debe distribuir e instalar el cliente en todos los equipos.

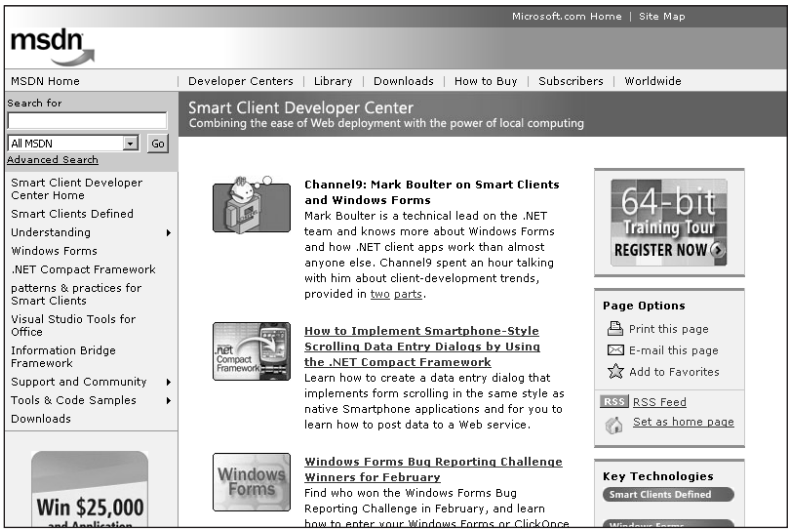


Figura 14. *Microsoft es uno de los propulsores de la idea de construcción de Smart Clients, en su página web* ***www.microsoft.com*** *encontraremos más información.*

TIPO DE CÓDIGO EJECUTABLE

Cuando trabajamos con aplicaciones que se ejecutan directamente sobre el equipo, distinguimos dos tipos de código para generar: **código nativo** y **código manejado**.

Código nativo

Desarrollar una aplicación en código nativo implica que el archivo ejecutable que instalaremos en el equipo está expresado en código ensamblador entendible por el sistema operativo y por el procesador del equipo. Esto no quiere decir que nosotros desarrollemos bajo ensamblador (*assembler*), simplemente el compilador que utilicemos para desarrollar realizará la traducción a dicho código.
Las ventajas de desarrollar bajo código nativo son:

III CÓDIGO MIXTO

Por medio de la utilización de las tecnologías de **Microsoft**, estamos en condiciones de generar aplicaciones de **código mixto**, donde tienen parte desarrollada en código manejado y ciertas funciones de bajo nivel desarrolladas en código nativo.

- Mayor rapidez de ejecución.
- Código más compacto.
- Acceso al 100% de las capacidades del equipo.
- No se requiere la instalación de ningún agregado para la ejecución.
- Se posee acceso directo a memoria y de bajo nivel.

Las desventajas son:

- En cada sistema operativo o hardware, se necesita recompilar el proyecto y generar ejecutables distintos.
- Si trabajamos con dos equipos distintos, debemos tener en cuenta diferencias de hardware en algunas funciones.
- Se posee acceso directo a memoria (esto puede traer problemas).

Código manejado

El código manejado surgió como solución a los problemas que traía el código nativo. Cuando trabajamos con este tipo de código, lo que se genera al compilar el proyecto no es código nativo entendible por el hardware y el sistema operativo del equipo, sino que es un código que es entendible por un aplicativo intermedio entre nuestro programa y el hardware, llamado **Máquina Virtual**. Esta máquina virtual interpreta el código manejado y lo convierte en tiempo real (*Just in Time*) a código nativo subordinado al sistema y hardware en que se encuentra.

Las ventajas del código manejado son:

- Con un solo proyecto y compilación podremos ejecutar nuestra aplicación en diversos sistemas y hardware.
- No se posee acceso directo a memoria (ya que la Máquina Virtual lo administra automáticamente).
- Generamos un solo paquete de instalación para todos los equipos.

III .NET EN PALM Y SYMBIAN

A través de un producto comercial llamado **CrossFire** es posible desarrollar aplicaciones ejecutables para **Palm OS**, **Symbian OS** y algunos celulares propietarios utilizando **Visual Studio.NE**T con lenguaje **Visual Basic** o **C#**.

Las desventajas del código manejado son:

- No se accede al 100% de los recursos del equipo, sólo a lo que se definió como parte del estándar de la máquina virtual.
- No se tiene acceso de bajo nivel a recursos o a memoria.

Los más utilizados en el mundo móvil en código manejado son: **J2ME** para celulares y PalmOS y **.NET Compact Framework** para equipos con distintas versiones de **Windows**, que analizaremos en la segunda parte de este libro.

Matriz de lenguajes

Ahora veamos en una tabla los distintos lenguajes que podemos utilizar para desarrollar en cada una de las plataformas mencionadas anteriormente.

PLATAFORMA	JAVA	C++	C# - VISUAL BASIC (.NET)
Pocket PC	No	Ejecutables	Ejecutables
		Código Nativo	Código Manejado
			Aplicaciones Online
Windows Smartphone	No	Ejecutables	Ejecutables
		Código Nativo	Código Nativo
			Aplicaciones Online
Palm OS	Ejecutables	Ejecutables	Aplicaciones Online
	Código Manejado	Código Nativo	
Symbian OS	Ejecutables	Ejecutables	Aplicaciones Online
	Código Manejado	Código Nativo	
Celulares propietarios	Ejecutables	No	Aplicaciones Online
	Código Manejado		
Otros Dispositivos	Smart Cards	No	Aplicaciones Online
			SPOTs

***Tabla 1.** Los lenguajes disponibles para desarrollar en las plataformas mencionadas sin productos de terceros.*

{ } FLASH MOBILE

La herramienta de **Macromedia Flash**, conocida por sus animaciones para la Web, permite desarrollar aplicaciones con su lenguaje **Action Script** para algunos teléfonos celulares y PDAs (todavía en forma limitada). Más información en **www.macromedia.com/mobile.**

SOLUCIONES MÓVILES DE MICROSOFT

Veamos qué tipo de soluciones nos permiten desarrollar las herramientas de desarrollo de **Microsoft** y en qué tipo de equipos podemos ejecutarlo.

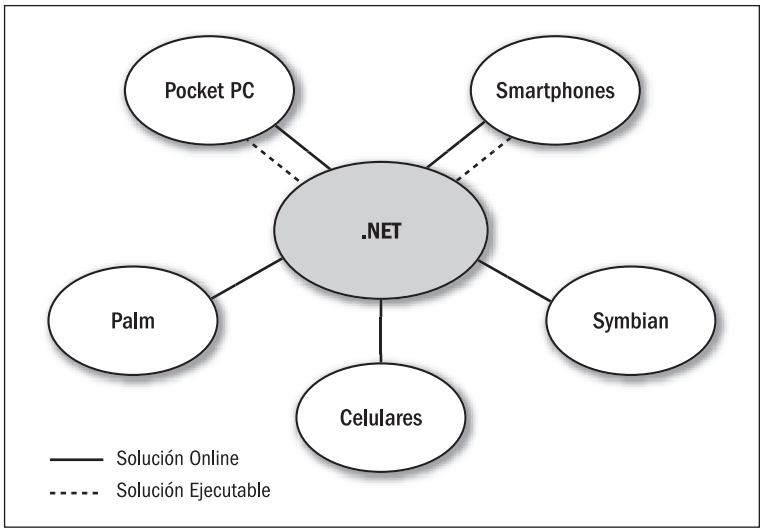


***Figura 15.** Con .NET podemos acceder a toda la gama de equipos móviles en el mercado, ya sea con aplicaciones online o aplicaciones ejecutables.*

Soluciones Online

Con la salida de .NET, Microsoft ofreció su nueva plataforma para desarrollos web: **ASP.NET**. Junto con esta plataforma, un poco más tarde, apareció **ASP.NET Mobile**. Esta tecnología ofreció una de las primeras (y actualmente la más robusta) soluciones al desarrollo de aplicaciones móviles Online que nos resuelve uno de sus mayores problemas: la incompatibilidad de lenguajes.

Además de todas las ventajas que tiene **ASP.NET** como plataforma para aplicaciones Online, que repasaremos más adelante, en cuanto a la solución móvil nos ofrece, mediante el uso de formularios web móviles, la posibilidad de generar conteni-

POR QUÉ ASP.NET Y NO OTRO

¿Podemos desarrollar aplicaciones móviles con **PHP**, **JSP** u otro lenguaje? Por supuesto que sí, pero ninguno de estos lenguajes nos ofrece una solución integral especialmente diseñada para equipos móviles. En el resto de los lenguajes debemos realizar todo a mano.

do estático y dinámico sin importar qué lenguaje necesita el equipo receptor de la página. Automáticamente, el motor de **ASP.NET Mobile** generará contenido **WML** si es necesario, **cHTML**, **XHTML** o **HTML 3.2**, según el equipo que solicite la página. Esto produjo un gran avance, dado que hasta ahora el programador debía realizar este tipo de conversión de un lenguaje a otro en forma manual, desarrollando versiones alternativas para cada tipo de dispositivo.

Las soluciones que desarrollemos en ASP.NET Mobile serán compatibles con **todos** los tipos de dispositivos móviles del mercado, aunque no sean de Microsoft, o no tengan un sistema operativo de esta empresa. El contenido será generado para el tipo de contenido que necesite el dispositivo. Por eso la primera parte de este libro está dedicada a esta plataforma de desarrollo.

Soluciones Stand-alone en código nativo

Desde la salida de los primeros equipos Pocket PC, Microsoft ofreció herramientas de desarrollo para esta plataforma. Estas herramientas están basadas principalmente en **C++** (también las hubo para **Visual Basic**) y generan código nativo para cada tipo de dispositivo Pocket PC, en sus diferentes versiones.
Las herramientas actualmente disponibles para realizar este desarrollo son gratuitas y se pueden descargar del sitio web **www.microsoft.com/mobile**. Están: **Embedded C++** (conocido como **eC++**), **Embedded Tools** y el discontinuado **Embedded VB** (**eVB++**) que no genera código ejecutable, sino código de *script*. Estas herramientas sólo desarrollan aplicaciones para equipos con Windows CE (Pocket PC, Handhelds PC, etc.).

Soluciones Stand-alone y Smart Clients

Con la salida de .NET y el principio de la programación en código manejado en cuanto a herramientas Microsoft, surgió la necesidad de recrear el mismo concepto para los equipos móviles. Así surgió **.NET Compact Framework**, un subset del **.NET Framework** conocido en aplicaciones de escritorio y aplicaciones web. Las aplicaciones bajo este paradigma se conocen como **Smart Device Application**.

{} EL FIN DE EC++

Con la salida de **Visual Studio.NET 2005**, el soporte para desarrollo nativo en **C++** para **Pocket PC** y **Smart Phones** vendrá incluido en esta herramienta y se espera que no se ofrezca más soporte a la herramienta gratuita **Embedded C++**.

Este **Framework**, junto con una máquina virtual que se instala en cada equipo, permite la ejecución de aplicaciones en código manejado desde cualquier equipo que lo soporte.
Los nuevos equipos Pocket PC y Smart Phone ya tienen el Framework instalado en su **ROM**; para los más antiguos el Framework puede descargarse gratuitamente e instalarse sobre los equipos. Recordemos que como se trata de código manejado, es necesario este intérprete para poder ejecutar la aplicación.
En la segunda parte de este libro nos dedicaremos a la generación de aplicaciones Stand-alone y Smart Clients ejecutables para Pocket PC y Windows Smart Phones utilizando la tecnología de .NET Compact Framework.
Es importante aclarar que los únicos dispositivos que actualmente brindan soporte para la ejecución de este Framework son aquéllos que vienen provistos con versiones de Windows, a partir de la **Pocket PC 2000**.

FRAMEWORK VS. FRAMEWORK

Cuando trabajemos con ASP.NET utilizaremos el **.NET Framework** completo, dado que la ejecución se realizará en el servidor y no en el equipo móvil. Cuando trabajemos con desarrollo ejecutable, usaremos el **.NET Compact Framework** dado que la ejecución se realiza en el equipo móvil. Veamos las diferencias entre ambos.

.NET Framework

Con el surgimiento de .NET, Microsoft quiso revolucionar el modo de desarrollar bajo su plataforma. Ahora podemos desarrollar aplicaciones en múltiples lenguajes con la garantía de compatibilidad entre ellos y con un mismo Framework de clases detrás que respalda a toda la funcionalidad que ofrece la plataforma.
En el caso de aplicaciones móviles, utilizaremos el **.NET Framework** para nuestras aplicaciones **ASP.NET Mobile**. Salvo aquellas clases que están pensadas para aplicaciones de escritorio, tendremos todo el Framework a nuestra disposición.

ASP.NET MOBILE ES COMPATIBLE CON TODOS

La ventaja de **ASP.NET Mobile** es que no trabaja sólo con los dispositivos de sistema **Windows Mobile**, puede generar contenido para equipos **PalmOS**, **Symbian** o teléfonos celulares de cualquier operador.

La última versión estable al momento de escribir este libro es la **1.1 Service Pack 1** y sobre ella trabajaremos la mayor parte del libro.

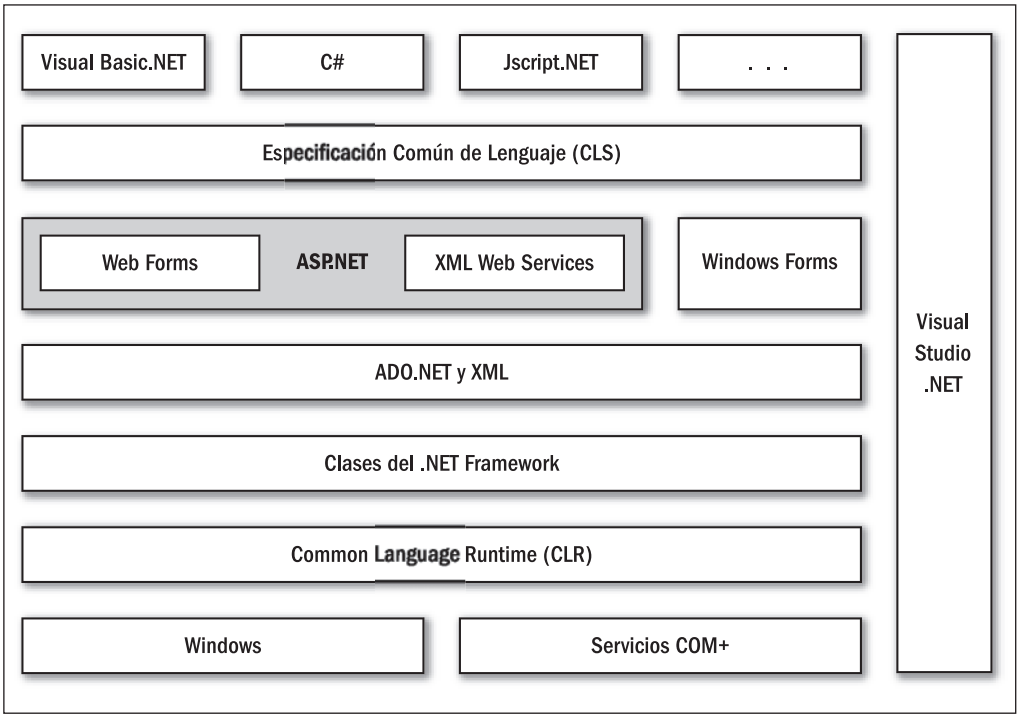


Figura 16. *Aquí vemos la arquitectura de .NET Framework. En este libro, trabajaremos con ASP.NET como interfaz.*

.NET Compact Framework

Desde la salida de **Visual Studio 2003**, surgió esta versión compacta y reducida del .NET Framework. Consta de un subset de clases (y cada clase, un subset de métodos y propiedades) del .NET Framework mayor, e incorpora algunos *namespaces* dedicados específicamente al trabajo sobre equipos móviles.

III NET FRAMEWORK 2.0

Ya está en Beta y próximamente disponible **.NET 2.0**, con él vendrán algunos cambios en ASP.NET. En cuanto a Mobile, se sigue recomendando toda la base vista de 1.1 en este libro.

La última versión es la **.NET Compact Framework 1.0 Service Pack 2**. Esta versión compacta ocupa generalmente entre 2 y 3 MB en memoria del equipo, contra los casi 24 MB de la versión mayor.

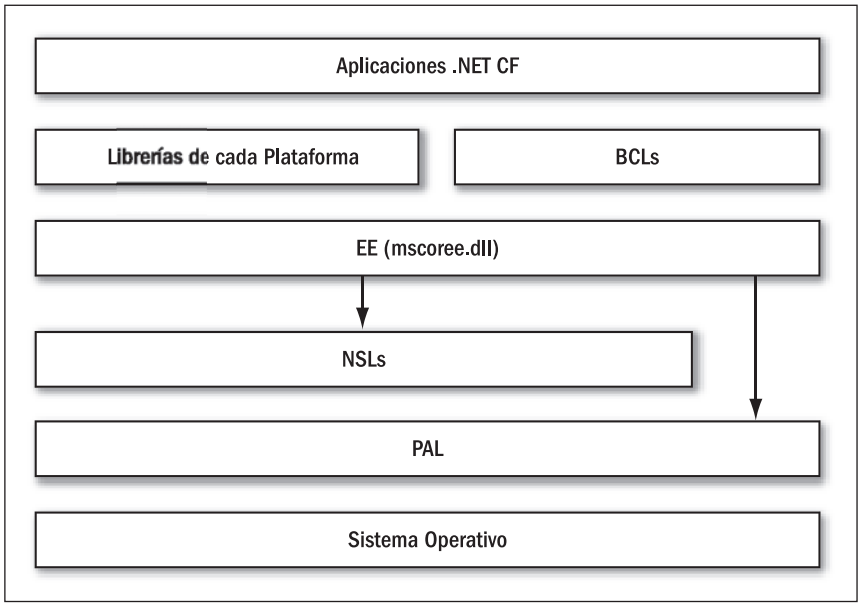


***Figura 17.** Aquí vemos la arquitectura del **Compact Framework** de Microsoft. Vemos que es un poco más simple que la del **.NET Framework**.*

Las clases exclusivas de .NET Compact Framework son:

- **System.Data.SqlServerCE**: administra la funcionalidad contra la versión compacta de SQL Server CE (ahora conocida como Mobile).
- **Microsoft.WindowsCE.Forms**: administra la funcionalidad para controlar el *input panel*, y la comunicación con código nativo.
- **System.Net**: incorpora diversas clases para el manejo de puertos infrarrojos, como **IrDAEndPoint**, **IrDAClient**, **IrDADeviceInfo**, **IrDAListener**.

III .NET COMPACT FRAMEWORK 2.0

También la versión compacta espera ver su segunda versión a la luz. Soportará todas las mejoras de **C# 2.0**, mejoras de rendimiento, mayor cantidad de controles disponibles, **IPv6** y hasta la posibilidad de embeber un browser.

Más adelante, haremos un detalle más intensivo de esta versión del Framework. Éstas son algunas de las diferencias más importantes entre ambos Frameworks:

TEMA	DIFERENCIA EN .NET COMPACT FRAMEWORK
ASP.NET	No trae soporte de ASP.NET dado que no está pensado para que pueda ejecutarse un servidor en el equipo móvil.
Interoperabilidad COM	No se tiene soporte para comunicarse con objetos COM. Sí es posible hacerlo a través de una invocación en la plataforma (PInvoke)
Datos	Posee un subset de ADO.NET, no está soportado el name space OleDb y se incorpora el name space SqlServerCE.
Input / Output	Debido a limitaciones en el sistema operativo, algunas operaciones de entrada / salida no están disponibles.
Redes	Provee soporte adicional para puertos infrarrojos.
Serialización	No se provee serialización de objetos a través de Binary Formatter o Soap Formatter. Sí la hay en el uso de Web Services.
Gráfico	Hay diversos cambios en los controles de formulario, que veremos luego.
XML	No soporta validación del XML Schema, ni consultas Xpath.

***Tabla 2.** Diferencias entre los **Framework**.*

Lenguajes

Tanto para desarrollos ASP.NET Mobile, como para desarrollos Smart Devices (es el nombre que le da Microsoft al desarrollo para .NET Compact Framework), podremos elegir realizar desarrollos en cualquiera de los siguientes lenguajes:

- **Visual Basic.NET**
- **C#.NET**

HERRAMIENTAS DE DESARROLLO

A continuación analizaremos todas las herramientas que vamos a necesitar durante todo el libro para ambos tipos de desarrollo.

En primer término hay que definir el entorno de desarrollo donde trabajaremos para realizar nuestros desarrollos móviles con .NET. Para el caso de desarrollos para SD (otra abreviatura muy utilizada, de Smart Devices o dispositivos inteligentes) la única opción será **Visual Studio.NET** (VS.NET). Para el caso de aplicaciones ASP.NET Mobile podremos elegir entre **VS.NET** o **Web Matrix**.

Web Matrix

Este producto es una herramienta de desarrollo exclusivamente pensada y diseñada para ASP.NET y es distribuida en forma gratuita por Microsoft. Puede descargarse desde la web **www.asp.net** y pesa solamente unos 1.3 MB (debemos instalar con anterioridad el .NET Framework).
Web Matrix ofrece a los programadores las siguientes ventajas:

- Edición en forma visual o código **ASP.NET** y **HTML**.
- Soporte para conexión a bases de datos **SQL Server** y **Access** desde el entorno.
- Soporte para **VB.NET**, **C#**, **J#** y **JScript.NET**.
- Examinador de clases, permitiendo el acceso a sus métodos, atributos y eventos de manera bastante sencilla.
- Soporte para **XML Web Services**.
- Soporte para desarrollos móviles.
- Soporte de **FTP**.
- Incluye una versión personal del servidor web, con la cual poder probar los desarrollos sin necesidad de tener **Internet Information Server** instalado.

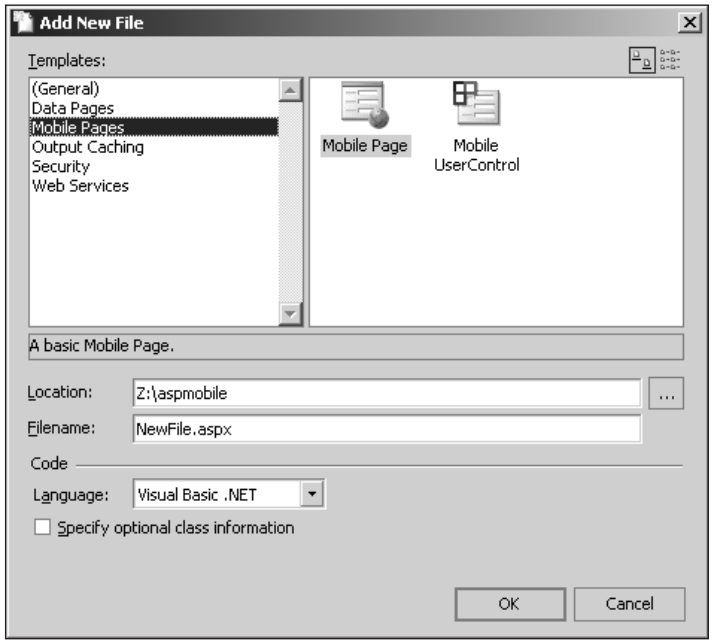


***Figura 18. Web Matrix** nos permite crear tanto páginas web para móviles como controles de usuario destinados a este tipo de equipos.*

En el próximo capítulo explicaremos la manera de aprovechar este aplicativo gratuito con la finalidad de desarrollar aplicaciones móviles.

Visual Studio .NET

Este producto es el más completo orientado a crear desarrollos en ASP.NET y es el único disponible de Microsoft para el desarrollo sobre Pocket PC. Se trata de exactamente el mismo entorno que utilizamos para desarrollar aplicaciones de Windows. Visual Studio, desde la versión 2003, permite realizar desarrollos móviles bajo el mismo formato y características.
Dentro de las ventajas con respecto a, por ejemplo, Web Matrix, está la posibilidad de administrar un proyecto, de tener acceso a mejor administración de los componentes, documentación más detallada, aumento del rendimiento al desarrollar y mejores herramientas para la codificación, *debugging* y compilación.

Visual Studio.NET 2003 provee herramientas de completado automático de código **HTML** o **ASP.NET** (*Intellisense*), ayuda dinámica (mientras escribimos código), integración con **MSDN** (*Microsoft Developer Network*), provee de soporte para crear páginas web y formularios móviles en modo visual. También permitirá compilar los proyectos enteros, realizar debugging y generar paquetes de distribución sobre el mismo entorno.

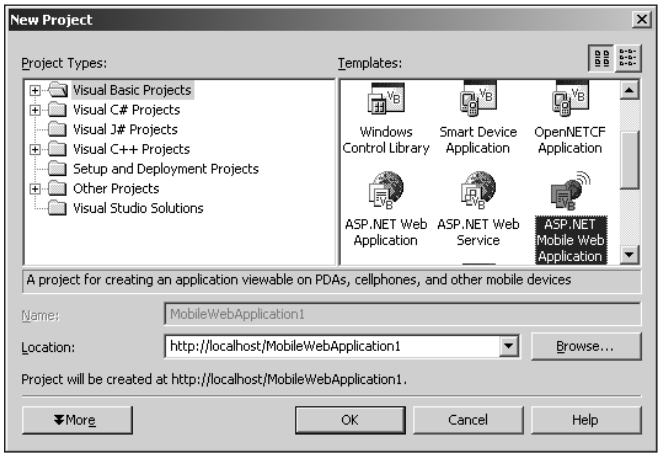


***Figura 19.** Con **VS.NET** podremos crear **Smart Device Applications** (aplicaciones con Compact Framework) y **ASP.NET Mobile Web Application**, desde el mismo entorno.*

III SQL EXPRESS

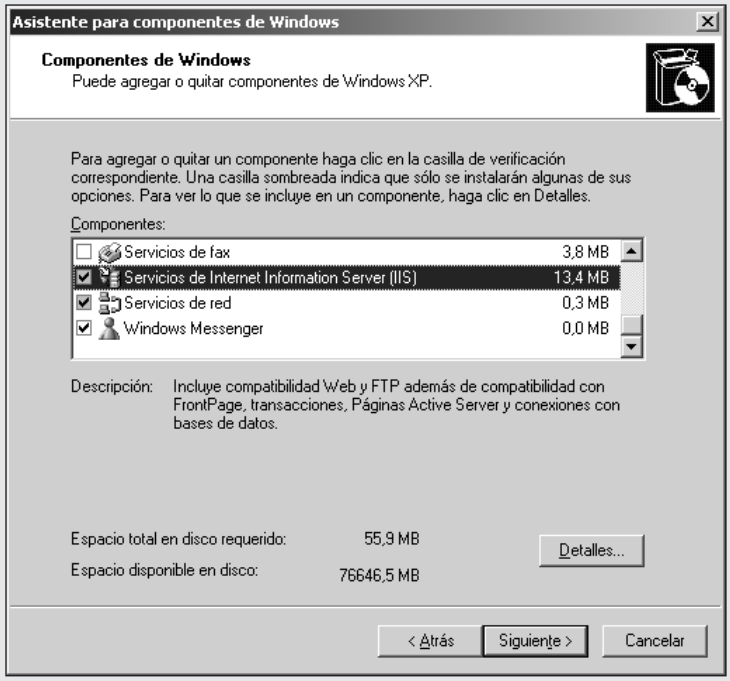
La próxima versión del **MSDE** parece ser la que actualmente se conoce como **SQL Server Express 2005** y que se ofrece gratuitamente (la beta) en el sitio web de Microsoft. Así, incorpora todas las mejoras de **SQL Server 2005**.

SERVIDORES

Sólo para el caso de trabajar con ASP.NET Mobile necesitaremos tener instalado **Internet Information Server** (*IIS*) como servidor en producción y, en desarrollo, si utilizamos **Web Matrix** no nos hará falta.
El servidor **Internet Information Server** forma parte de la instalación de **Windows 2000**, **XP**, **2003** y superior. En caso de no tenerlo instalado, se lo puede instalar teniendo a mano el CD de instalación realizando los siguientes pasos:

Instalar IIS — PASO A PASO

1. Ingrese a **Panel de control**.
2. Ingrese a **Agregar o quitar programas**.
3. Entre en la opción **Agregar o quitar componentes de Windows**.
4. Seleccione la opción **Servicios de Internet Information Server (IIS)**.



5. Pulse **Siguiente**.

Este servidor no es para nada necesario en el caso de desarrollar aplicaciones móviles para dispositivos con **.NET Compact Framework**.

MSDE o SQL Server

Para el caso de querer trabajar con SQL Server como base de datos, en ASP.NET o en proyectos Smart Device, necesitaremos tener instalado este servidor en nuestro equipo. Si está en el equipo de desarrollo, podemos utilizar **SQL Server Desktop Engine**, que es la versión gratuita de **SQL Server** para equipos de escritorio. Con ella, podremos realizar todas las pruebas necesarias durante el desarrollo de una aplicación .NET. Se puede conectar sin problemas con **Web Matrix**, **Visual Studio**.
El producto es conocido como **MSDE** y se encuentra disponible para descargar en forma gratuita en español. Ocupa 45 MB de descarga y la dirección para descargarlo es **www.microsoft.com/sql/msde/downloads/download.asp**.

SQL Server Mobile

Sólo para el caso de aplicaciones Smart Device, tendremos la opción de utilizar **SQL Server Mobile** (hasta ahora conocido como **SQL Server CE**). Éste es un pequeño motor que corre directamente sobre un equipo Windows CE (Pocket PC u otro) al que podremos acceder desde nuestra aplicación.
El producto no requiere licencia, salvo que esta versión compacta se comunique y sincronice con un servidor SQL mayor, para lo cual sí se necesitan las licencias.
Más adelante veremos más información sobre este tema, mientras tanto podemos descargarlo de **www.microsoft.com/sql/ce/downloads**.

EMULADORES

Para desarrollar aplicaciones Smart Devices necesitaremos probar nuestros desarrollos en un equipo **Pocket PC**, **Handheld PC** o **SmartPhone**. Si no disponemos de

III WINDOWS XP EMBEDDED Y TABLET PC

Existen dos versiones de Windows XP, llamadas **Embedded** y **Tablet PC Edition**, que están destinadas a equipos un poco más grandes que los que usamos en este libro, pero que no llegan a ser equipos de escritorio. Para estas versiones rigen las "mayores" de las herramientas de desarrollo.

él, o si lo tenemos (pero es más cómodo no usarlo para debugging), podemos hacer uso de los símiles que nos provee Microsoft, que emulan el sistema operativo y la funcionalidad del mismo en nuestra PC de escritorio.
En el caso de aplicaciones móviles, utilizaremos diferentes emuladores de distintas marcas para probar cómo se ven nuestras páginas móviles en cada uno de los diferentes browsers.

Windows

Visual Studio.NET ya viene incluido con emuladores básicos de cada una de las plataformas. Más allá de eso, podremos descargar en forma gratuita de la página de Microsoft diversas imágenes de nuevos equipos o versiones del sistema operativo para poder probar nuestras aplicaciones ejecutables o nuestras aplicaciones **ASP.NET Mobile** a través de su **Pocket Internet Explorer**.

***Figura 20.** Recordemos que los emuladores tienen incluido todo el sistema operativo. En este caso usamos la versión **Pocket Internet Explorer** para acceder a Google.*

Éstos son los únicos emuladores que nos serán de utilidad a la hora desarrollar aplicaciones para dispositivos Smart Devices.

Palm OS

En el caso de que deseemos probar la manera en que se desempeñará nuestra aplicación móvil en un browser de Palm OS, éste es el emulador que tenemos que instalar en nuestra computadora.
Tenemos la posiblidad de descargar emuladores para diferentes versiones del sistema operativo Palm OS, bajando las ROMs correspondientes a cada una.

III EMULADORES

Existen emuladores para distintos tipos de procesadores, no sólo celulares y Palms. Hay emuladores de consolas de juego, como GameBoy, Playstation o equipos antiguos, como Commodore 64.

La descarga debe efectuarse desde la dirección web **www.palmos.com/dev/dl**.
Podremos descargar el **Emulador**, cuya tarea sonsiste en realizar una emulación de hardware y del sistema operativo, o un **Simulador** (para el caso de equipos Palm sobre chips Intel), que no lleva a cabo la emulación de hardware.

***Figura 21.** Los emuladores de Palm también permiten definir el skin con el cual podremos ver el equipo Palm exactamente como es en realidad.*

■ Descargar un emulador de PalmOS — PASO A PASO

1. Descargue un emulador o simulador.
2. Descargue las ROMs de las versiones que queremos probar del sistema operativo.
3. Si la versión descargada no incluye un browser, podemos descargar algún browser de **www.handango.com** o **www.palmfreeware.com**.
4. Según la configuración de la versión descargada, es necesario que defina la configuración de la red del emulador para que el equipo Palm pueda hacer uso del protocolo **TCP/IP** utilizando la conexión de su PC.

Symbian OS

No existe gran cantidad de emuladores **Symbian OS** disponibles para utilizar, que además tengan uso de su navegador web. Un buen punto de partida es el sitio web

{ } OPENWAVE BROWSER

Muchos fabricantes de celulares, como **Siemens**, utilizan los browsers de **OpenWave**, por lo que podemos suponer que en muchos de ellos nuestros desarrollos se verán de la misma forma.

oficial **www.symbian.com/developer** o alguno de los emuladores de **Nokia** que veremos enseguida que pueden emular equipos Series 60 con Symbian OS.

Nokia

Nokia es una de las empresas que ofrece mayor cantidad de emuladores para los desarrolladores. En el caso de emuladores que traen incorporado el sistema operativo completo, incluyendo el web browser que no requiere de configuraciones adicionales, podemos destacar el **Nokia Series 40 MIDP Concept SDK** que se descarga gratuitamente de **www.forum.nokia.com**. Una vez descargado e instalado el emulador, se lo puede ejecutar desde **Inicio/Programas** y podremos ejecutar el browser de dos formas:

1. Abriéndolo como si estuviéramos en el teléfono real buscándolo en el menú.
2. Utilizando la opción **File/Open** del emulador e ingresando la dirección web (incluyendo **http://**) que queremos abrir.

Figura 22. *Aquí vemos la versión* ***Mobile*** *de Google desde el emulador de* ***Nokia Series 40*** *utilizando la opción File/Open.*

OpenWave

Desde el sitio **www.openwave.com/us/products/mobile/developer_products** podremos descargar el **OpenWave Mobile SDK** que trae consigo un simulador de teléfono (**Phone Simulator**) con un browser que podemos utilizar para probar nuestros desarrollos. Este browser está distribuido en la mayoría de los modelos de diferentes marcas de teléfonos celulares.

Opera

El navegador web de escritorio **Opera** tiene además versiones para distintas plataformas móviles. La ventaja que tenemos es que si poseemos la versión de escritorio de Opera (es *adware*, se puede descargar gratuitamente de **www.opera.com**), podremos ver cómo se verían nuestras páginas en la versión móvil del navegador, ya que incluye el motor móvil con la versión "grande".
Cuando ingresamos a una página web que queremos probar en su versión para móvil a como se vería desde **Opera**, sólo presionamos **SHIFT+F11** y veremos el contenido adaptado a la pantalla de un equipo celular.

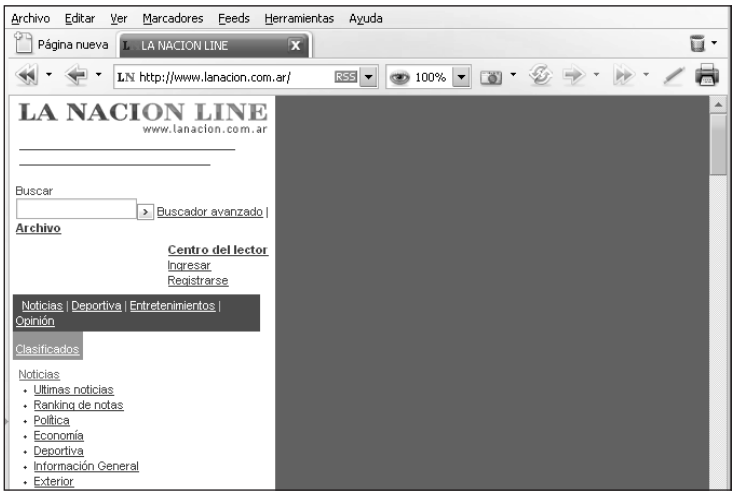


Figura 23. *Cualquier página web, Opera la convierte en una versión para móvil presionando* ***SHIFT+F11*** *cuando la tenemos abierta en el navegador.*

Motorola Browser ADK

Motorola nos ofrece este aplicativo también de descarga gratuita de su sitio para desarrolladores **www.motocoder.com**.

Este emulador nos ofrece la visión de un browser **Motorola** (en su versión genérica) para nuestra página web móvil.

La ventaja más importante que trae consigo este browser es que al mismo tiempo que nos muestra la pantalla de un teléfono virtual, en otra ventana nos muestra el código fuente del proyecto que estamos desarrollando.

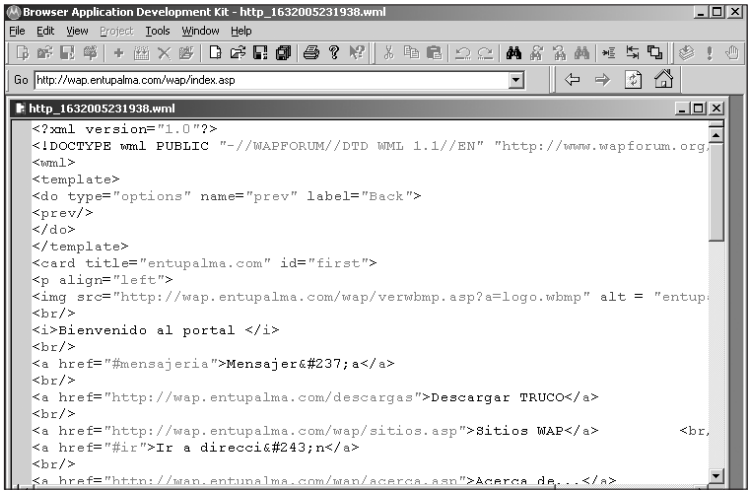


Figura 24. *El browser de* ***Motorola*** *nos muestra el código fuente, en este caso* ***WML****, mientras estamos navegando en el teléfono virtual.*

Otros emuladores

Existe infinidad de otros emuladores **WAP**, **WML**, **cHTML**, **XHTML**, **iHTML** que podemos utilizar para probar nuestros desarrollos móviles. Basta con buscarlos en sitios como **www.download.com** y probarlos.

Figura 25. *En el sitio download.com encontrará muchísimas herramientas que le serán de utilidad al momento de desarrollar aplicaciones móviles.*

RESUMEN

En este capítulo hicimos un repaso de todos los dispositivos móviles existentes en el mercado, cómo podemos desarrollar para cada uno de ellos y dónde .NET nos ofrece soluciones, ya sea móviles o ejecutables para distintos tipos de equipos. También vimos todas las herramientas que necesitaremos a lo largo del libro para nuestros desarrollos.

ACTIVIDADES

EJERCICIOS PRÁCTICOS

- ✔ Instale todos los aplicativos y acostúmbrese a su uso.
- ✔ Pruebe diferentes portales y sitios para móviles con los emuladores para habituarse a las aplicaciones móviles.
- ✔ Investigue el mercado de las aplicaciones móviles y encuentre algún servicio que todavía no se esté ofreciendo.
- ✔ Piense en cinco aplicaciones móviles y decida qué será mejor para cada una: utilizar soluciones online, ejecutables o smart clients.
- ✔ Investigue en la Web acerca de los smart clientes y las técnicas que se utilizan.
- ✔ Verifique qué equipos se distribuyen actualmente en el mercado (celulares y PDAs), e identifique el sistema operativo y la versión que emplean.