ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Proyecto Fin de Carrera

"Diseño e implementación de servicios web multiplataforma utilizando el framework .NET"

AUTOR: Manuel Ródenas Galián
DIRECTOR: Antonio Javier García Sánchez

Julio/ 2012

Autor	Manuel Ródenas Galián
E-mail del Autor	manuel.rodenas@gmail.com
Director(es)	Antonio Javier García Sánchez
E-mail del Director	antoniojavier.garcia@upct.es
Codirector(es)	
Título del PFC	Diseño e implementación de servicios web multiplataforma utilizando el framework .NET
Descriptores	

Resumen

Con el auge de las plataformas móviles y la necesidad de estar siempre conectado, se hace necesario el desarrollo de servicios web multiplataforma que cumplan con la demanda de los usuarios.

.NET es un framework de Microsoft en el que destaca la transparencia de redes, independientemente de la plataforma de hardware. Su principal característica es que ofrece una manera rápida y económica de desarrollar aplicaciones permitiendo una integración más rápida y eficaz, y simplificando el acceso a información desde cualquier tipo de dispositivo.

En este proyecto se ha realizado el estudio de esta tecnología, proponiendo una aplicación chat para la comunicación tanto en formato texto como en video llamada a través de la conocida aplicación Skype, y desde cualquier dispositivo con conexión a internet, ya sea pc, Tablet o dispositivos móviles.

Titulación	Ingeniería Técnica de Telecomunicaciones, especialidad Telemática
Intensificación	
Departamento	Tecnologías de la Información y las Comunicaciones
Fecha de Presentación	2012

Índice

Índicedefiguras

Capítulo 1
Introducción

1.1 Antecedentes

Durante las tres últimas décadas la programación ha ido evolucionando para satisfacer las necesidades que se iban presentando, como puede ser la reutilización del código. Para ello se han dado tres grandes saltos en lo que avance de la técnica de programación se refiere, empezando por la programación orientada a objetos, la programación orientada al componente y por último la programación orientada a servicio, que ha sido el último gran salto de los paradigmas de la programación. A continuación se muestra un pequeño resumen de cada estilo.

La mayor característica de la programación orientada a objeto es la posibilidad de utilizar clases reusables, así como el uso de varias técnicas como herencia, abstracción, polimorfismo y encapsulamiento. Un objeto se define como la unidad que realiza las tareas de un programa en tiempo de ejecución. La gran ventaja de los objetos es que interactúan unos con otros, a diferencia de la programación tradicional, en la que un programa era simplemente una lista de instrucciones. Fue a mediados de los ochenta cuando la programación orientada a objetos se fue convirtiendo en el estilo de programación dominante

La ventaja de la programación orientada a componentes es que el nivel de abstracción de los componentes es más alto que el de los objetos y se comunican entre ellos por medio del intercambio de mensajes. Un componente es un elemento que ha sido programado para que dé un servicio predeterminado, y que a su vez es capaz de comunicarse con otros componentes. Se podría decir que un componente es como una caja negra, que se desarrolla para que cumpla una función pero sin saber cuándo ni dónde ni quién la va a utilizar.

Y por último, la programación orientada a servicios, tiene como principal ventaja la encapsulación y la posibilidad de carga dinámica, lo que hace posible la total integración con sistemas distribuidos. Esto se consigue mediante la utilización de mensajes para comunicarse. Este tipo de programación ha supuesto un gran avance a la hora de desarrollar sistemas distribuidos que ya no necesitan realizar invocaciones a referencias de objetos, consiguiendo así, la integración del software en un sistema escalable y distribuido.

1.2 Motivación y objetivos

Los servicios web junto con las nuevas tecnologías (smartphones, tablets…) hacen pensar en un gran número de aplicaciones potenciales que facilitarían nuestra vida. El principal objetivo de los desarrolladores actualmente es crear servicios multiplataforma que puedan ser accedidos desde cualquier dispositivo, sin que se vea perjudicada la usabilidad del producto.

Un ejemplo de servicio web sería una agencia de viajes que crea una aplicación para informar al cliente de todo lo relacionado con su compañía. A su vez podría buscar información sobre hoteles, vuelos, de una manera transparente para el usuario, que solo tendría que introducir sus preferencias en el portal web de la empresa de viajes, y sería ésta la que se encargaría de hacer todas las búsquedas necesarias. (ver Figura 1.1).

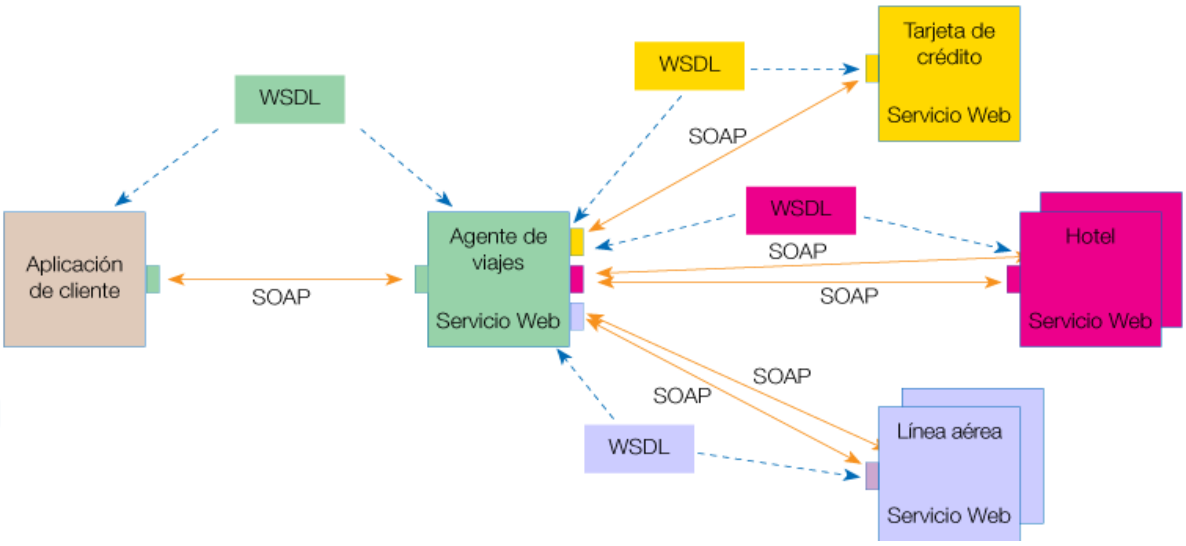

Figura 1.1 *Ejemplo servicio web*

En el departamento de Tecnología de la información y Comunicaciones de la Universidad Politécnica de Cartagena, se trabaja activamente en el desarrollo de servicios web utilizando el framework .NET. El objetivo de este proyecto es el aprendizaje del framework .NET, y a su vez, desarrollar e implementar un servicio web capaz de hacer comunicarse a los usuarios a través de texto o videoconferencia, con la mayor transparencia para el usuario, siendo su única obligación acceder a la página web y disfrutar de los servicios. Una de las mayores ventajas de los servicios web es dicha transparencia. Un usuario sin ninguna noción de informática puede usar una gran cantidad de servicios web interconectados entre ellos, sin percatarse de ello, y sin ninguna dificultad añadida.

El proyecto final de carrera que se presenta en esta memoria tiene como objetivo diseñar, desarrollar e implementar y hacer funcionar el servicio web ChatASP. La aplicación se implementará con las tecnologías que ofrece Microsoft en su Framework .NET, así pues el desarrollo se realizará con el entorno Microsoft Visual Studio, y el lenguaje de programación será C#. Los pasos para realizar dicho trabajo se resumen en los siguientes puntos:

- Estudio del framework .NET y de la tecnología ASP. Con este estudio se pretende obtener una perspectiva del funcionamiento global de .NET y ASP, que permitirá abordar el desarrollo del servicio web objetivo.

- Estudio de los servicios web. Este estudio implica una fase teórica, centrada en entender el funcionamiento de la arquitectura de los servicios web, así como las partes que componen una aplicación y las herramientas necesarias para realizar el trabajo con el Visual Studio.

- Implementación del servicio web ChatASP. El objetivo es realizar un servicio web que haga posible la comunicación entre usuarios de distintas plataformas, tanto por texto como por videoconferencia.

1.3 Herramientas utilizadas

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) para sistemas operativos Windows. Soporta varios lenguajes de programación

tales como Visual C++, Visual C#, Visual J#, ASP.NET y Visual Basic .NET, aunque actualmente se han desarrollado las extensiones necesarias para muchos otros.

Visual Studio permite a los desarrolladores crear aplicaciones, sitios y aplicaciones Web, así como servicios Web en cualquier entorno que soporte la plataforma .NET. Así se pueden crear aplicaciones que se intercomuniquen entre estaciones de trabajo, páginas Web y dispositivos móviles.

Para entender la versión 2003, debemos explicar primero la 2002 porque es la base de la versión 2003 y donde se hace la gran actualización en el Visual Studio y donde se incorpora la plataforma .NET.

En la versión 2002 se produjo un cambio sustancial, puesto que supuso la introducción de la plataforma .NET de Microsoft. .NET es una plataforma de ejecución intermedia multilenguaje, de forma que los programas desarrollados en .NET no se compilan en lenguaje máquina, sino en un lenguaje intermedio (CIL - Common Intermediate Language) denominado Microsoft Intermediate Language (MSIL). En una aplicación MSIL, el código no se convierte a lenguaje máquina hasta que ésta se ejecuta, de manera que el código puede ser independiente de la plataforma (al menos de las soportadas actualmente por .NET). Las plataformas han de tener una implementación de Infraestructura de Lenguaje Común (CLI) para poder ejecutar programas MSIL.

Actualmente se pueden ejecutar programas MSIL en Linux y Mac OS X usando implementaciones de .NET que no son de Microsoft, tal cómo Mono (iniciado por Ximian y actualmente impulsado por Novell tras la adquisición de Ximian) y DotGNU (que forma parte del proyecto GNU con el fin de proporcionar una alternativa libre para la plataforma de desarrollo Microsoft.NET, otras de sus metas son mejorar la compatibilidad con las plataformas diferentes a Windows y a otros procesadores).

Visual Studio .NET 2002 supuso también la introducción del lenguaje C# (utilizado para este Proyecto), un lenguaje nuevo diseñado específicamente para la plataforma .NET, basado en C++ y Java. Se presentó también el lenguaje J# (sucesor de J++) el cual, en lugar de ejecutarse en una máquina virtual de Java, se ejecuta únicamente en el framework .NET. El lenguaje Visual Basic fue remodelado completamente y evolucionó para adaptarse a las nuevas características de la plataforma .NET, haciéndolo mucho más versátil y dotándolo con muchas características de las que carecía. Algo similar se llevó a cabo con C++, añadiendo extensiones al lenguaje llamadas Managed Extensions for C++ con el fin de que los programadores pudieran crear programas en .NET. Por otra parte, Visual FoxPro pasó a comercializarse por separado. Visual FoxPro es un lenguaje de programación orientado a objetos y procedural, un Sistema

Gestor de Bases de datos o Database Management System (DBMS), y desde la versión 7.0 (en 2001), un Sistema administrador de bases de datos relacionales. Visual FoxProofrece a los desarrolladores un conjunto de herramientas para crear aplicaciones de bases de datos para el escritorio, entornos cliente/servidor, tablet PC o para la Web.

Todos los lenguajes se unifican en un único entorno. La interfaz se mejora notablemente en esta versión, siendo más limpia y personalizable.

Visual Studio .NET puede usarse para crear programas basados en Windows (usando Windows Forms en vez de COM), aplicaciones y sitios Web (ASP.NET y servicios Web), y dispositivos móviles (usando el .NET Compact Framework).

Visual Studio .NET 2003 supone una actualización menor de Visual Studio .NET. Se actualiza el .NET Framework a la versión 1.1. También se añade soporte con el fin de escribir aplicaciones para determinados dispositivos móviles, ya sea con ASP.NET o con el .NET Compact Framework. Además el compilador de Visual C++ se mejora paracumplir con más estándares con el Visual C++ Toolkit 2003.

Visual Studio 2010 es la versión más reciente de esta herramienta (y la que se ha utilizado en este proyecto), acompañada por .NET Framework 4.0.

Hasta ahora, uno de los mayores logros de la versión 2010 de Visual Studio ha sido el de incluir las herramientas para desarrollo de aplicaciones para Windows 7, tales como herramientas para el desarrollo de las características de Windows 7 (System.Windows.Shell) y la Ribbon Preview para WPF.

Entre sus más destacables características, se encuentran la capacidad para utilizar múltiples monitores, así como la posibilidad de desacoplar las ventanas de su sitio original y acoplarlas en otros sitios de la interfaz de trabajo.

Además ofrece la posibilidad de crear aplicaciones para muchas plataformas de Microsoft, como Windows, Azure, Windows Phone 7 o Sharepoint. Microsoft ha sido sensible a la nueva tendencia de las pantallas táctiles y con este Visual Studio 2010 también es posible desarrollar aplicativos para pantallas multitáctiles.

Capítulo 2
ASP y .NET

2.1 ASP

ASP es una tecnología que lanzó Microsoft del tipo lado del servidor para la generación de páginas web dinámicamente. A esta tecnología se le conoce actualmente como ASP clásico, y es el precursor del framework .NET.

Al ser una tecnología del tipo lado del servidor, ASP se ejecuta en el servidor web, y una vez finalizado se envía el resultado al cliente a través de Internet en forma de página web. El documento que recibe el cliente es una página en HTML, lo que hace posible su compatibilidad con todos los navegadores y sistemas operativos. Por otra parte, en el lado del servidor, las páginas pueden realizar accesos a bases de datos, conexiones de red, etc. Esto se consigue mediante el uso de scritps, usando Visual Basic Script o JScript (Javascript de Microsoft).

A la hora de usar ASP disponemos de seis objetos nativos, que a diferencia de otros modelos de objetos, no forman una jerarquía, sino que se relacionan entre sí de forma lógica.

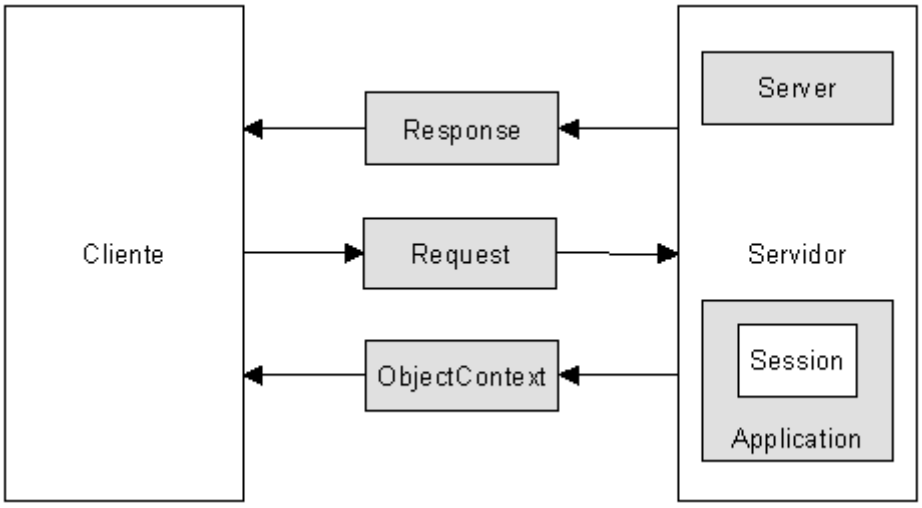

Figura 2.1 *Estructura ASP*

A continuación se procede a mostrar las características más importantes de estos objetos:

• Application. Su principal función es compartir la información de la aplicación entre todos los usuarios conectados a ella. Únicamente existe un objeto Application común a todos los usuarios.

• ObjectContext. Se utiliza para la gestión de las transacciones.

• Request. Este objeto se encarga de facilitarnos toda la información que un usuario nos manda a través de la cabecera HTTP. Esta información puede llegar en forma de petición GET o POST, en forma de cookies, etc.

• Response. Es el encargado de enviar toda la información del servidor al cliente, como por ejemplo el envío de información directamente al navegador del cliente o el manejo de los valores de las cookies.

• Server. Su función es proporcionar métodos relacionados con el servidor dónde se ejecuta la aplicación. Su principal uso es para crear una instancia de un componente.

• Session. Al acceder a la aplicación, a cada usuario se le asigna un objeto session, siendo esta información diferente para cada usuario. El tiempo de vida de este objeto es de veinte minutos, aunque se puede modificar, y se crea la primera vez que un usuario solicita una página.

La tecnología ASP, aparte del gran avance que supuso a la hora de desarrollar aplicaciones web, también contenía un gran número de inconvenientes que se han subsanado con la aparición de .NET. Entre ellos se puede destacar la dificultad para mantener y actualizar el código escrito debido a que está mezclado con el HTML de la página, complicando la tarea del programador a la hora de introducir novedades, ya que el fragmento de código ha de situarse en el punto exacto donde se quiere que su salida aparezca. Otro de los inconvenientes sería la necesidad de escribir código para mantener el estado de la página ASP, cuando por ejemplo, el usuario quiera volver a ella. Esto provoca la pérdida de información si hubiese algún error al, por ejemplo, rellenar un formulario.

Pero los inconvenientes de ASP no quedan aquí, también es conocido la ineficacia de esta tecnología a la hora de implantar sistemas de cierta envergadura. Debido a sus características se presentan graves problemas de eficiencia cuando, por ejemplo, llegan más peticiones al servidor de las que éste puede atender, aparte de los problemas de depuración que se pueden presentar. Aunque en realidad, muchos de estos

problemas provienen de los lenguajes utilizados en los fragmentos de código en las páginas ASP. En la actualidad, debido a la creciente interoperabilidad de múltiples componentes y la manipulación de datos empleando numerosas fuentes de información se hace más complicado las aplicaciones de esta naturaleza.

Para solucionar este problema aparecieron los servicios web, cuya principal función es atender a diferentes tipos de usuarios y además se introdujo la posibilidad de que los servicios pueden residir en diferentes servidores Web. Esta tecnología fue avanzando hasta la llegada de la tecnología .NET de Microsoft, que se fundamenta en los servicios web para la construcción de aplicaciones basadas en tecnologías de Internet.

2.2. .NET

Microsoft .NET es un framework cuya principal característica es la transparencia de redes, independiente del hardware que se utilice, y que permite un rápido diseño de aplicaciones.

El entorno .NET se puede clasificar en las siguientes partes:

• .NET Framework, que es el entorno de trabajo de la plataforma .NET y que la engloba completamente. Toda la plataforma .NET forma parte de .NET Framework.

• Lenguajes .NET, Soporta más de 20 lenguajes, haciendo posible desarrollar cualquier tipo de aplicación con cualquiera de ellos. Los más usados son C#, Visual Basic .NET, Delphi, etc.

• Common Runtime Language (CRL), es el entorno de ejecución en el que se cargan las aplicaciones desarrolladas en los distintos lenguajes.

• Microsoft Intermedial language (MSIL), se encarga de transformar código nativo intermedio de alto nivel independientemente de la plataforma que lo ejecuta a código de máquina del dispositivo que lo ejecuta.

• Common Language Specification (CLS), su función consiste en determinar las reglas para crear el código MSIL

• ASP.NET, es la evolución del ASP clásico. Su principal característica es su facilidad para la creación de páginas Web dinámicas. Además se encuentra integrado en el entorno .NET.

• Biblioteca de clases .NET, maneja la mayoría de las operaciones básicas que se encuentran involucradas en el desarrollo de aplicaciones

2.2.1. MSIL, CRL y el código controlado

A la hora de escribir un programa se puede hacer en diferentes lenguajes de programación, lenguajes que aunque no se puedan leer como tal, podemos entenderlos con ayuda de los conocimientos adquiridos, mediante un editor de texto. Pero cuando se quiere ejecutar un programa, un ordenador no es capaz de entender este lenguaje, por lo que se tiene que traducir a lenguaje máquina, es decir, el lenguaje que las máquinas son capaces de entender. Esto se hace a través de la compilación, cuya función es traducir el programa del lenguaje que nosotros hemos escrito al lenguaje que la máquina pueda entender, dando como resultado un ejecutable, que si bien la máquina ya puede entender, nosotros no podemos interpretar.

Este lenguaje máquina es diferente dependiendo del sistema operativo, arquitectura del procesador, etc, que se utilice, solo funcionara en la plataforma para la que fue diseñado.

A la hora de desarrollar .NET, se hizo especial hincapié en este tema, ya que el objetivo a conseguir era dejar atrás los problemas de compatibilidad del software dependiendo de la plataforma que se utilizase. Para ello se creó un lenguaje intermedio llamado MSIL. Cuando se programa en .NET, independientemente del lenguaje que se esté utilizando, siempre se compila hacia MSIL (Microsoft Intermediate Language). Una vez que el programa ha sido compilado será interpretado por un intérprete, el CRL. Con esto se consigue, obteniendo el intérprete apropiado, que cualquier programa escrito en .NET funcione en cualquier plataforma.

Cuando se compila a MSIL, el procesador no puede interpretar el programa ya que no está escrito en lenguaje máquina, para ello se utiliza el CRL, cuya función es hacer de intermediario entre el lenguaje de MSIL y el lenguaje máquina.

El CRL es muy parecido al ByteCode de Java, aunque CRL es mejor, además que una de las características de CRL, al contrario que ByteCode, únicamente compila la parte necesario del código en cada momento durante la ejecución.

Así podemos ejecutar nuestro programa sobre cualquier máquina, siempre y cuando exista una versión del .Net FrameWork y del CRL apropiada. Al código que se ejecuta bajo la batuta del CRL se le conoce como código manejado.

La aparición de JAVA en el mercado supuso un gran adelanto, debido a la aparición del concepto de máquina virtual, en el que un lenguaje era compilado a un lenguaje intermedio, que con ayuda de una máquina virtual podía ejecutarse en toda clase de máquinas. Microsoft adopta esta idea en .NET creando CRL, con la diferencia de que no se limita a un único lenguaje. La estrategia de Microsoft en este ámbito es intentar acoger al mayor número de desarrolladores, independientemente del lenguaje que utilicen. De esta manera se asegura que su tecnología .NET pueda ser usada por cualquier desarrollador.

CRL se caracteriza por ser independiente del sistema operativo por tanto se aprecia como Microsoft ve nuevos frentes donde expandirse o protegerse según se mire. El mercado mundial de sistemas operativos orientado al sector de microinformática está dominado por la serie Windows. En años recientes un nuevo competidor GNU ha llegado para quedarse, Linux, una reciente versión de UNIX está comiendo mercado debido a sus ventajas heredadas de UNIX y a las suyas propias. Se puede apreciar como Microsoft podría desembarcar en Linux empleando .NET o bien podría ser un signo de debilidad del sistema Windows ante la llegada de Linux. CRL puede en un futuro implantarse en otros sistemas operativos distintos de la serie Windows como base para el desarrollo de aplicaciones de escritorio. Como se aprecia, esta característica es nueva con respecto a ASP que no da soporte al desarrollo de aplicaciones de escritorio.

2.2.2. Basic Class Library

La librería de clase base es una serie de librerías contenidas en el framework .Net diseñadas para ayudar a los desarrolladores a construir sus aplicaciones. Está formada por cientos de tipos de datos cuya principal función es acceder a los servicios que ofrece el CRL así como el acceso a las funcionalidades más habituales en lo que a escribir programas se refiere. También el desarrollador puede crear clases nuevas que mediante herencia extiendan su funcionalidad.

La ventaja de esta librería es que está escrita en MSIL y por lo tanto se puede usar desde cualquier lenguaje que genere MSIL, es decir, todos los lenguajes que contiene .Net. Una de las particularidades de C# es que carece de librería de clases propia, basándose solo en la BCL.

Debido al gran tamaño de la BCL, se han organizado las clases en espacios de nombres, agrupando las clases con funcionalidades similares. Un ejemplo son:

Espacio de nombres	Utilidad de los tipos de datos que contiene
System	Contiene clases fundamentales y clases base que definen los tipos de datos de valor y de referencia, los eventos y controladores de eventos, los interfaces, los atributos y las excepciones de procesamiento más frecuentes. Proporciona servicios que admiten la conversión de tipos de datos, la manipulación de los parámetros de métodos, matemáticas, la invocación de programas remotos y locales, la administración del entorno de las aplicaciones y la supervisión de aplicaciones administradas y no administradas.
System.Collections	Contiene interfaces y clases que definen varias colecciones de objetos, como listas, colas, matrices, tablas hash y diccionarios. Incluye el espacio de nombres Specialized para colecciones con establecimiento inflexible de tipos.
System.Data	Manipulación de bases de datos. Forman la denominada arquitectura ADO.NET
System.IO	Contiene tipos que permiten la lectura y escritura sincrónicas y asincrónicas en archivos y secuencias de datos.
System.Net	Proporciona una interfaz de programación sencilla para un gran número de protocolos utilizados actualmente en la red. Las clases WebRequest y WebResponse constituyen la base de los "protocolos conectables", una implementación de servicios de red que permite desarrollar aplicaciones que utilizan recursos de Internet sin tener que preocuparse de las particularidades del protocolo utilizado.
System.Reflection	Contiene clases e interfaces que proporcionan una vista administrada de los tipos, métodos y campos cargados, permitiendo la creación y la invocación dinámicas de los tipos.
System.Runtime.Remoting	Proporciona clases e interfaces que permiten a los programadores crear y configurar aplicaciones distribuidas de correspondencia estricta o imprecisa.
System.Security	Proporciona la estructura subyacente del sistema de seguridad del motor de tiempo de ejecución, incluidas en las clases base para los permisos.
System.Threading	Proporciona clases e interfaces que permiten la programación multiproceso. También proporciona clases para la programación de los subprocesos, las notificaciones de espera y la resolución de bloqueo.

System.Web.UI	Proporciona clases e interfaces que permiten crear controles y páginasque constituyen la interfaz de usuario para las aplicaciones Web. Proporciona asimismo clases que incluyen la función de enlace de datos a controles de servidor de formularios Web Forms, permiten guardar el estado de vista de un control o una página determinados y analizar la funcionalidad de los controles programables y Literal.
System.Windows.Forms	Creación de interfaces de usuario y aplicaciones basadas en Windows para aplicaciones estándar.
System.XML	Acceso a datos en formato XML.

2.2.3. Assemblies

Cuando se genera un proyecto .Net, no se genera un ejecutable, sino que se genera un assemblie. Un assemblie es la unidad ejecutable de cualquier programa .NET, pero no es solo eso, también incluye el manifiesto. Un manifiesto es un listado de las librerías y los controles necesarios para que la aplicación funcione. Incluye, entre otras cosas, el número de versión necesario para que la aplicación funcione.

2.2.4. ViewState

El ViewState nos facilita el almacenar la información de estado en las páginas Web ASP.NET entre devoluciones de datos y Post Back. Esta información se serializa en XML para luego codificarla en Base64. Su funcionamiento se limita al procesamiento de la página, a la codificación del estado actual de la página guardando en la página la cadena que se obtiene de forma oculta.

El ViewState es una de las diferencias entre ASP y .NET, ya que permite mantener el estado de los controles de una misma página entre yna ida y venida al servidor. Además tiene muchas otras funciones como poder conocer si es la primera vez que se ejecuta la página, o guardar las variables que se quieran accediendo desde el código.

El problema de ViewState es que al serializar la información en XML y luego codificarla en Base64, puede generar grandes cantidades de información que podría llegar a afectar al rendimiento de la página. Para solucionar este problema se puede deshabilitar

el ViewState para los controles de datos como GrindView,DataList,etc. Para ello se utilizaría la expresión: EnableViewState="false".

2.3. Diferencias entre ASP y .NET

Con la llegada de .NET, Microsoft quiso dar otro enfoque para el desarrollo de software. No se limitó a actualizar ASP, sino que desarrollo una tecnología totalmente nueva para el desarrollo de aplicaciones Web. Las principales prioridades de Microsoft a la hora de desarrollar .NET fueron usar XML como vehículo transmisor de información, incrementar la presencia de modos de tratamiento de la información mediante la librería de datos y utilizar una nueva filosofía de construcción de formularios Web.

El lanzamiento dela tecnología .Net por parte de Microsoft supuso un duro golpe para J2EE ya que lo superaba es varios aspectos. Uno de ellos es el acceso más sencillo y la facilidad de poder emplear una gran variedad de lenguajes de codificación, pero sin perder potencia de diseño ya que los principales lenguajes de programación son orientados a objetos.

Figura 2.2 *Configuración de una aplicación Web con HTML estático*

Antes de la aparición de la Web dinámica, la mayoría de páginas Web se basaban en utilizar documentos HTML estáticos que ofrecían siempre la misma información (Figura 2.1), que aunque puede ser la solución más adecuada cuando nuestra página web

siempre ofrezca la misma información, la evolución de Internet ha llevado a los desarrolladores a crear aplicaciones Web que generan dinámicamente el contenido que finalmente se ofrece a los usuarios. De esta manera podemos seleccionar, filtrar, ordenar y presentar la información de la forma más adecuada en función de las necesidades de cada momento. Aunque esto se podría conseguir con páginas de HTML estático se necesitaría un gran espacio disponible en el disco duro, por lo que la incursión de las aplicaciones Web ha sido fundamental en la evolución de Internet tal y como lo conocemos hoy.

Para ello, la creación de aplicaciones Web requiere la existencia de software ejecutándose en el servidor, cuya función será la generación de los ficheros HTML que le llegan al usuario. Para el usuario no ha supuesto un gran cambio, ya que él sigue realizando la conexión con el servidor a través del protocolo HTTP y recibiendo páginas HTML, que si bien estas han sido creadas dinámicamente, el usuario final no lo puede distinguir. La principal diferencia consiste en que, ahora, el servidor HTTP delega en otros módulos la generación dinámica de las páginas HTML que se envían al cliente (Figura 2.2).

Figura 3.3 *Configuración de una aplicación Web con HTML dinámico*

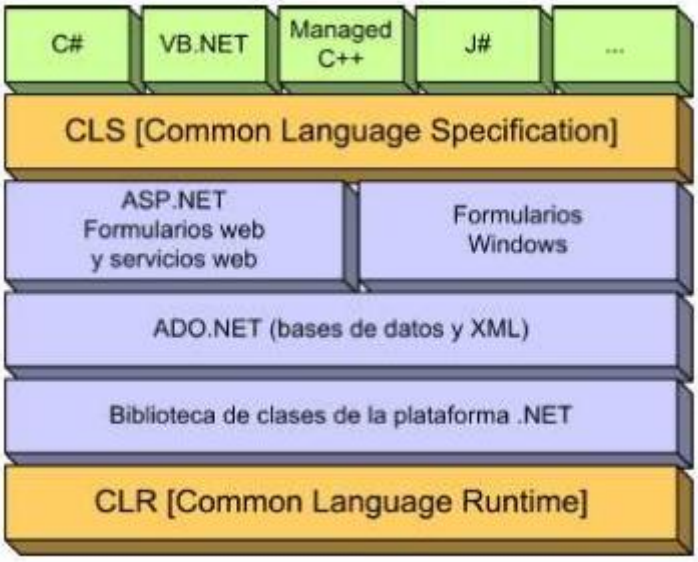

Figura 4.4 *Familia de tecnologías de la plataforma .NET*

Para la construcción de interfaces de usuario en .Net se dispone de dos herramientas, los formularios Web y los formularios ASP.NET (Figura 2.3). El problema de estas dos herramientas es que no son intercambiables, es decir, no existe una forma estándar de crear una interfaz de usuario que funcione tanto para aplicaciones Windows como para aplicaciones Web.

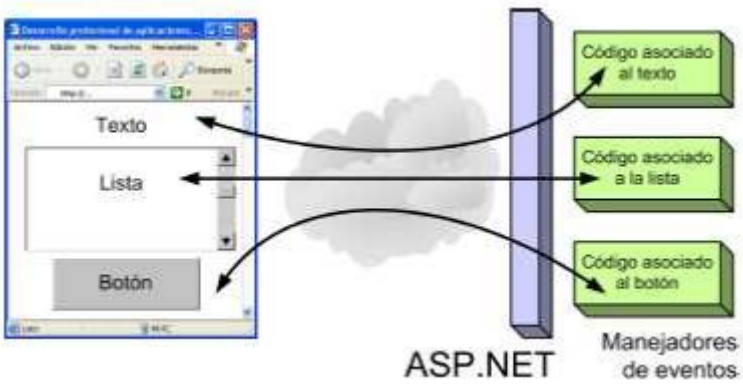

Figura 5.5 *Funcionamiento de los eventos en la plataforma .NET*

Una de las diferencias de ASP y ASP.NET es que éste último en vez de aceptar datos de entrada y generar su salida en HTML, implementa su funcionalidad en fragmentos de código que se ejecutan como respuesta a eventos asociados a los controles de la interfaz con los que puede interactuar el usuario. De esta manera se consigue un mayor nivel de abstracción, necesita menos código y permite crear aplicaciones más modulares, legibles y fáciles de mantener.

Capítulo 3
Servicios Web

3.1 Servicios web

Un servicio web es una tecnología capaz de hacer que las aplicaciones se comuniquen independientemente de la plataforma y del lenguaje de programación. Es una interfaz cuya principal característica es el describir un conjunto de operaciones que pueden ser accedidas por la red a través de mensajería XML estandarizada. Esto lo consigue usando protocolos basados en XML con el objetivo de describir una operación para ser ejecutada o datos con el fin de ser intercambiados con otro servicio web. Es un grupo de servicios web que interactúa de esta forma el que define la aplicación de un servicio web específico en una arquitectura orientada a servicio (SOA).

Desde los inicios de la programación uno de los principales problemas ha sido integrar aplicaciones en distintos sistemas operativos, lenguajes de programación y plataformas hardware, debido al acoplamiento fuerte, en el cual una aplicación que llama una red remota queda fuertemente vinculada a ella por la llamada de función que hace y los parámetros que solicita. Por lo tanto se puede afirmar que antes de la llegada de los servicios web los sistemas tenían una interfaz fija, con poca flexibilidad y adaptabilidad a entornos cambiantes.

Una de las maneras con la que se intentó poner fin a este problema, y se consiguió, fue el uso de XML en los servicios web. De esta forma se puede describir todo tipo de datos de manera independiente de plataforma para el intercambio entre sistemas, permitiendo así el movimiento hacia aplicaciones flojamente acopladas. Pero además los servicios web pueden funcionar a un nivel más abstracto que puede modificar o manejar tipos de datos dinámicamente on demand. Gracias a todas estas funcionalidades, los servicios web han conseguido que se pueda llevar a cabo una comunicación más libre entre software, así como una mayor facilidad para el manejo de datos.

Los servicios web se basan en interfaces definidas universalmente y tareas bien diseñadas, facilitando así la reutilización de las aplicaciones. Ésta reusabilidad de software implica un mayor aprovechamiento de la inversión en software, ya que se pueden producir nuevas aplicaciones con los mismos recursos. El principal objetivo de los servicios web es resolver la integración de datos y aplicaciones y la transformación de funciones técnicas en tareas informáticas orientadas a negocios.

La evolución de los lenguajes de programación ha sido uno de los pilares fundamentales en la aparición de los servicios web tal y como los conocemos. En los primeros lenguajes de programación dominaba la idea de una función en la cual uno proporcionaba unos parámetros, la función ejecuta alguna operación y devuelve un valor.

Con el paso de los años, ese primer concepto evolucionó, dando lugar a objetos que tenían sus propias variables de datos privados, haciendo así más sencillo el desarrollo de aplicaciones. Fue, cuando las aplicaciones empezaron a comunicarse, el punto de partida para el concepto de interfaces universales definidas para objetos, permitiendo así que objetos de distintas plataformas se comuniquen entre sí, aunque no compartan el mismo lenguaje de programación o no operen en el mismo sistema operativo.

Con la llegada de los servicios web, se experimentó una aproximación al concepto de interfaces y comunicaciones definidas en XML, para fusionar aplicaciones de cualquier tipo, además de dar libertad de modificarlas y evolucionar con el paso del tiempo. El gran cambio de produce con la llegada de la versatilidad de XML, ya que permite separar la estructura gramatical (sintaxis) del significado gramatical (semántica), posibilitando así la separación de la forma que cada servicio del entorno procesa y entiende eso. Y es aquí donde se experimenta el mayor cambio, ya que ahora los objetos pueden ser definidos como servicios que se comunican con otros servicios en la gramática definida por XML, donde cada servicio traduce y analiza el mensaje de acuerdo con la implementación local y el entorno. De esta manera, una aplicación conectada en red puede estar compuesta por varias entidades con varios diseños y diferentes construcciones, con la condición de que cumplan las reglas definidas por su arquitectura orientada a servicios.

Los servicios web poseen una serie de funcionalidades que le permiten, entre otras cosas, la interacción entre servicios de cualquier plataforma, así como la conceptualización de funciones de aplicaciones a tareas, permitiendo así una gran abstracción de software. Además permite el acoplamiento flojo, por lo que se puede cambiar el diseño e incluso agregar más servicios sin que se vean afectadas las interacciones entre las aplicaciones.

Dentro de los servicios web se involucran una familia de protocolos relacionados con la descripción y el suministro de servicios web, así como la interactuación entre ellos. Ésta familia se puede dividir en distintos grupos basados en funciones y usos comunes. En el grupo de mensajería y dirección, el protocolo que más destaca es el protocolo simple de acceso a objetos (SOAP), cuya función es posibilitar la llamada entre una aplicación cliente y funciones de objetos remotos.

Una aplicación cliente necesita utilizar una implementación de SOAP, encargada de la serialización y deserialización de los paquetes, así como un servicio necesita una aplicación de servidor que sepa leer y escribir paquetes de mensajes SOAP . La misión principal de un archivo WSDL es comunicarle al componente Web service todo lo necesario para llamar correctamente a métodos del servicio. Por ello una aplicación debe ser capaz de leer y parsear un archivo WSDL en tiempo de ejecución.

Para representar el lenguaje de descripción de servicios web (WSDL) se utiliza una serie de sentencias XML que constituyen la definición de las interfaces de cada servicio. Entre las especificaciones más importantes se encuentran WS-Addressing, que se ocupa de definir cómo identificar los servicios web en una arquitectura distribuida y dar direcciones exclusivas. Otra especificación importante es Web Services Invocation Framework cuya función es hacer posible el definir interfaces WSDL para todo tipo de componentes.

Además de las dos especificaciones visto con anterioridad, también cabe destacar UDDI, que se encarga de definir cómo se anuncian los servicios y se encuentran con otros repartidos por la red. UDDI (Universal Description, Discovery and Integration), es un servicio que permite, tanto a particulares como a empresas, la publicación y búsqueda de servicios web. Funciona de manera independiente de la plataforma usada, describiendo los servicios web y mostrando la información asociada sirviéndose de una estructura estándar. Para conseguir la descripción de las interfaces hacia los servicios web se sirve del lenguaje WSDL, y utiliza SOAP, valiéndose de su interoperabilidad, para conseguir cubrir las necesidades de programación.

Internet como plataforma para los negocios basados en las tecnologías de la información se ha visto mejorado gracias a la llegada de UDDI, ya que antes de su aparición, no era posible localizar o dar a conocer servicios basados en el tratamiento de la información.

A modo de resumen, se puede comprobar que UDDI ha aportado varias mejoras a la hora de localizar el servicio requerido entre todos los que están registrados en la red, así como la manera de describir los servicios y los métodos de negocio de forma automática, en un entorno seguro, sencillo y abierto.

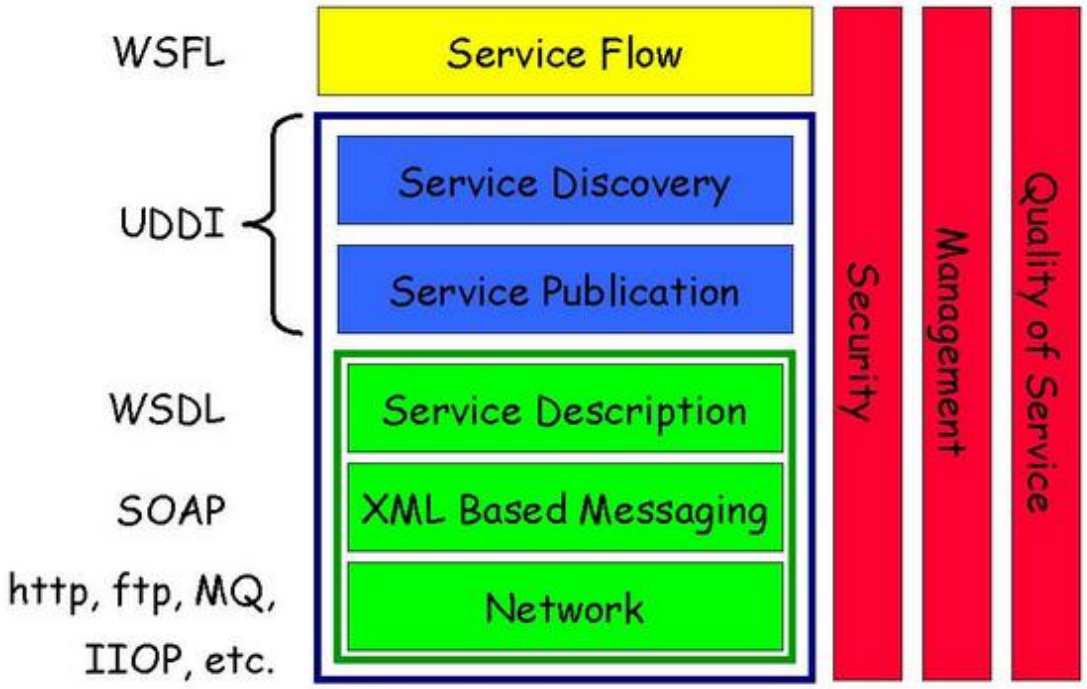

Figura 3.1 *Pila de servicios web*

A la hora de hablar sobre seguridad en los servicios web, se hace necesario mencionar la especificación WS-Security, punto de partida de los protocolos de seguridad, que define una arquitectura basada en señales para comunicaciones seguras. A continuación se muestran las seis principales especificaciones de componentes construidas sobre esta base:

- WS-Policy y sus especificaciones relacionadas, que definen las reglas de políticas sobre la interacción de servicios.
- WS-Trust, que define el modelo confiable para el intercambio seguro.
- WS-Privacy, que define cómo se mantiene la privacidad de las informaciones.
- WS-Secure Conversation, que define cómo establecer una sesión protegida entre servicios para intercambiar datos usando las reglas definidas en WS-Policy, WS-Trust, y WS-Privacy.

- WS-Federation, que define las reglas de identidad distribuida y de la gestión de esa identidad.
- WS-Authorization, que maneja el procesamiento de autorización para acceder a los datos e intercambiarlos.

Cabe destacar que, además del modelo de seguridad, existen las especificaciones de las aplicaciones, como por ejemplo WS-Transaction y WS-Coordination, cuya función es manejar el procesamiento distribuido de transacciones. Actualmente se está desarrollando una especificación para la gestión distribuida de servicios web que trata de la gestión administrativa de software de todos los servicios y de la arquitectura orientada a servicios.

Es el Web Service Interoperability Group (WS-I) el encargado de definir las especificaciones y protocolos para los servicios web, aunque es una tarea ardua, ya que no es posible abarcar todas las situaciones y combinaciones posibles. Su principal tarea fue desarrollar ejemplo de aplicaciones y herramientas de prueba para tener la certeza de que esos estándares y especificaciones funcionaran entre sí, sin importar las implementaciones de productos de los proveedores. WS-I ya ha definido su segundo Basic Profile 2.0 para servicios web, acompañado de casos de ejemplo y herramientas de prueba para evaluar los resultados. También colaboran en la definición de estos estándares la Organización para el Avance de Estándares de Informaciones Estructuradas (OASIS), así como el World Wide Web Consortium (W3C) entre otras.

Como se mencionó anteriormente, las tecnologías de componentes para servicios web son definidas en forma común e interactúan en XML, por lo tanto, ya que el propio XML es independiente del lenguaje usado, los servicios web también lo son. Es por esta razón por la que los servicios web se pueden desarrollar en distintos lenguajes de programación como C#, Basic, java, entre otros.

Uno de los principales objetivos es encontrar una forma mejor para la comunicación y la interactuación de la arquitectura de Internet y la arquitectura de las aplicaciones web. Por ello, la mayoría de los servicios web se basan en programas que operan en entornos en entornos de aplicaciones como WebSphere o Apache, ya que algunas de las mejores herramientas para servicios web están diseñadas para estos entornos.

Debido a la aparición de interfaces universales y más simples, entornos, como pueden ser los móviles, se están beneficiando gracias a la ayuda de los servicios web que están ayudando a mejorar el funcionamiento del modelo de informática. La rápida escalada del software para móviles no solo está adoptando rápidamente el modelo de

comunicaciones de los servicios web, sino que también los propios servicios web se están viendo beneficiados con la mejora de las interfaces.

La evolución de los servicios web ha ido progresando con el paso de los años. Desde su inicio, concebidos como una familia de protocoles avanzados de comunicaciones que permiten que las aplicaciones de comuniquen, hasta la actualidad, donde la aparición de múltiples herramientas han permitido a los desarrolladores crear servicios web que interactúan entre ellos, haciendo posible el desarrollo de aplicaciones más complejas.

La mayor capacidad que tienen los servicios web es la capacidad de la arquitectura orientada a servicios (SOA). SOA proporciona una metodología y un marco de trabajo para documentar las capacidades de negocio y puede dar soporte a las actividades de integración y consolidación. El principal objetivo de este modelo es la reutilización de la tecnología, produciéndose así una evolución en la forma de diseñar, desarrollar y poner uso a las aplicaciones. A partir de la aparición de SOA, los desarrolladores de software necesitan diseñar sus aplicaciones distribuidas pensando en este modelo, cuya característica principal es el uso de tecnologías para permitir las comunicaciones distribuidas de los servicios, como por ejemplo el uso del bus de servicios empresariales (ESB), que define una red de distribución común para el trabajo con los servicios.

Pero la mayor virtud de SOA y los servicios que lo componen es considerar a éstos como elementos constructivos, consiguiendo que puedan ser montados dentro de aplicaciones completas, dejando a un lado el método de escribir código. En este nivel, los desarrolladores pueden trabar en una arquitectura basada en modelos que les ayuda a crear aplicaciones con mayor exactitud respecto al diseño.

3.2 AJAX

El termino AJAX (Asynchronous JavaScript + XML) fue utilizado por primera vez por Jesse James Garrett, para hacer referencia a un nuevo tipo de aplicación web que estaba apareciendo. AJAX no es una única tecnología, sino que es un conjunto de tecnologías independientes, que se unen creando un sinfín de posibilidades.

Las tecnologías que forman AJAX son:

- XHTML y CSS, para crear una presentación basada en estándares.
- DOM, para la interacción y manipulación dinámica de la presentación.
- XML, XSLT y JSON, para el intercambio y la manipulación de información.
- XMLHttpRequest, para el intercambio asíncrono de información.
- JavaScript, para unir todas las demás tecnologías.

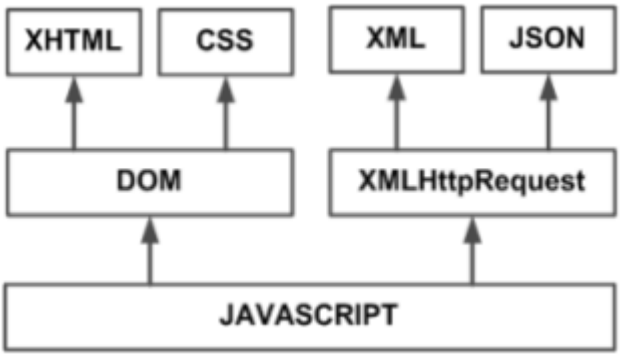

Figura 3.2 *Tecnologías agrupadas bajo el concepto de AJAX*

Antes de la llegada de AJAX, todas las acciones del usuario en una página desencadenaban llamadas al servidor, que se encargaba de procesar la petición y devolver una nueva página HTML. Esto provocaba el refresco de la página cada vez que el usuario, por ejemplo, seleccionaba el valor de una lista, haciendo que la experiencia del usuario no fuese muy agradable.

En el siguiente esquema, la imagen de la izquierda muestra el modelo tradicional de las aplicaciones web. La imagen de la derecha muestra el nuevo modelo propuesto por AJAX:

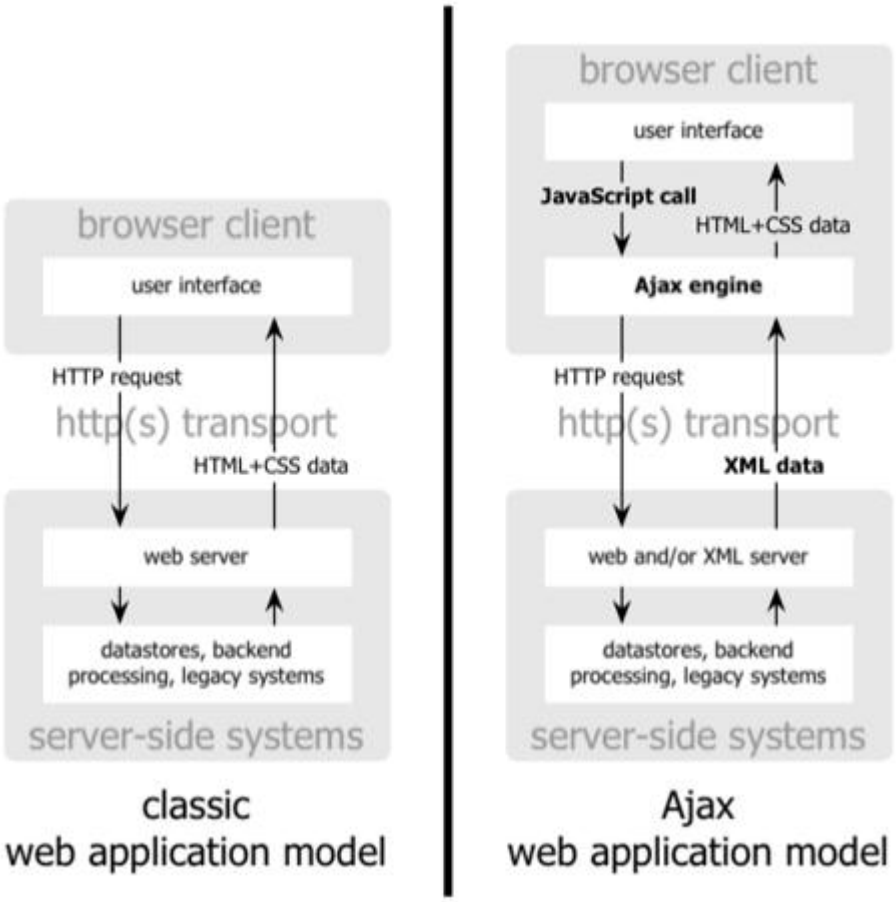

Figura 3.3 *Comparación gráfica del modelo tradicional de aplicación web y del nuevo modelo propuesto por AJAX.*

La principal característica de AJAX es la mejora de la interacción del usuario con la aplicación, haciendo que el intercambio de información con el servidor se produzca en un segundo plano, evitando así las recargas constantes de las páginas. Esto se consigue con la creación de un elemento intermedio entre el cliente y el servidor, una capa intermedia que mejora la respuesta de la aplicación.

El siguiente esquema muestra la diferencia más importante entre una aplicación web tradicional y una aplicación web creada con AJAX. La imagen superior muestra el funcionamiento síncrono propio de las aplicaciones web tradicionales. La imagen inferior muestra la comunicación asíncrona de las aplicaciones creadas con AJAX.

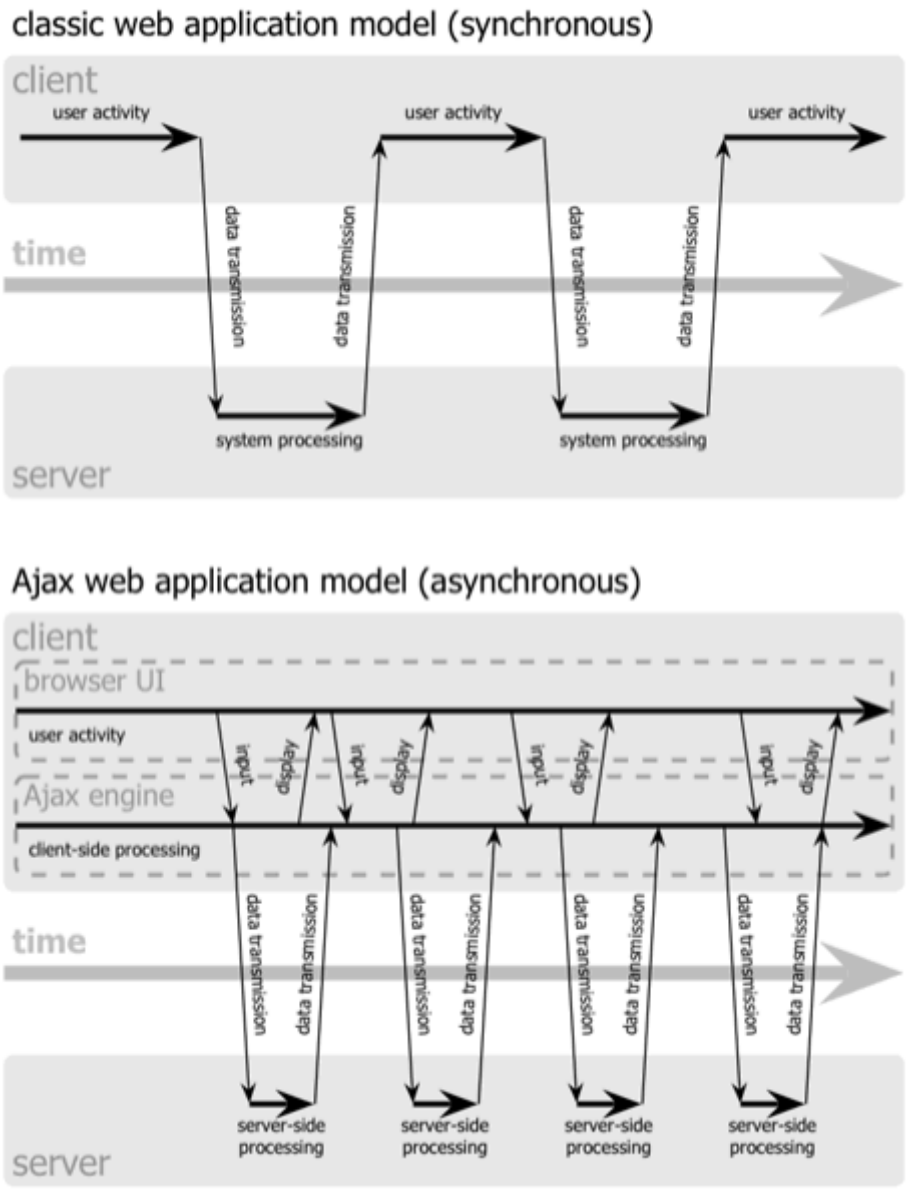

Figura 3.4 *Comparación entre las comunicaciones síncronas de las aplicaciones web tradicionales y las comunicaciones asíncronas de las aplicaciones AJAX*

Con esta nueva técnica, se sustituyen las peticiones HTTP al servidor por peticiones JavaScript al elemento encargado de AJAX. Además, las peticiones más simples no requieren intervención del servidor, haciendo que la respuesta sea inmediata. Aunque si se requiere la respuesta del servidor, la petición se realiza de forma asíncrona mediante AJAX, evitando que la interacción del usuario se vea interrumpida.

3.3 JQuery

JQuery es un Framework de Javascript que nos permite simplificar la manera de interactuar con el árbol DOM del documento HTML con el que estemos trabajando en ese momento. Nos permite manejar eventos, desarrollar animaciones e interactuar con la tecnología AJAX con la página web. JQuery es software libre y nos ofrece una serie de funcionalidades desarrolladas en JavaScript, simplificando al programador su trabajo.

El funcionamiento de JQuery se basa en la función del mismo nombre, JQuery, la cual ofrece una gran variedad de funcionalidades dependiendo de los parámetros utilizados. Además, gracias a que JavaScript toma conceptos del paradigma funcional, la función JQuery cuenta con distintas propiedades y métodos. Con este diseño se pretende evitar llenar el espacio de nombres global con demasiados nombres.

Aparte de JQuery, existen una gran variedad de frameworks, con soluciones similares, que sirven para hacer lo mismo. La ventaja de JQuery es la gran aceptación por parte de los desarrolladores y un grado de penetración en el mercado muy amplio. Además, es un producto serio, estable, bien documentado y con un gran equipo de desarrolladores a cargo de la mejora y actualización del framework, así como una gran comunidad de creadores de plugins que permiten encontrar fácilmente soluciones ya creadas.

3.3.1 Jquery Mobile

JQuery Mobile es un framework JavaScript que permite la creación de sitios web optimizados para dispositivos móviles. Busca subsanar las necesidades de los desarrolladores de dispositivos móviles, agregando una capa más al JQuery tradicional. Antes de la llegada de JQuery Mobile los desarrolladores tenían que programar para cada dispositivo, haciendo poco eficiente y costoso el desarrollo de aplicaciones para dispositivos móviles.

Uno de los mayores avances de JQuery Mobile es la abstracción de la lógica de cada dispositivo, permitiendo así a los desarrolladores concentrar su esfuerzo en el desarrollo de los servicios, dejando a un lado la preocupación por los distintos tipos de dispositivos. También destaca el reducido tamaño de la librería (pesa 12kb comprimida) y su gran soporte para todo tipo de plataformas móviles (Android, iOS, Blackberry, Symbian, Windows Phone…).

3.4 Nuevas tecnologías

En la actualidad, el pc ya no es el único medio para acceder a internet. Cada vez más usuarios lo hacen desde otros dispositivos como móviles o tablets y por ello se hace necesario adaptar los servicios web a toda clase de plataformas para que la experiencia del usuario sea óptima desde cualquier dispositivo desde el que se pueda conectar a internet.

Por encima del resto se encuentran iOS y Android, los sistemas operativos para móviles de Apple y Google respectivamente. Entre los dos tienen una cuota de mercado cerca del 80%, siendo iOS el que sigue ocupando la primera posición, pero con Android cada vez más cerca. A continuación se muestra un gráfico de la progresión de los sistemas operativos móviles en el último año:

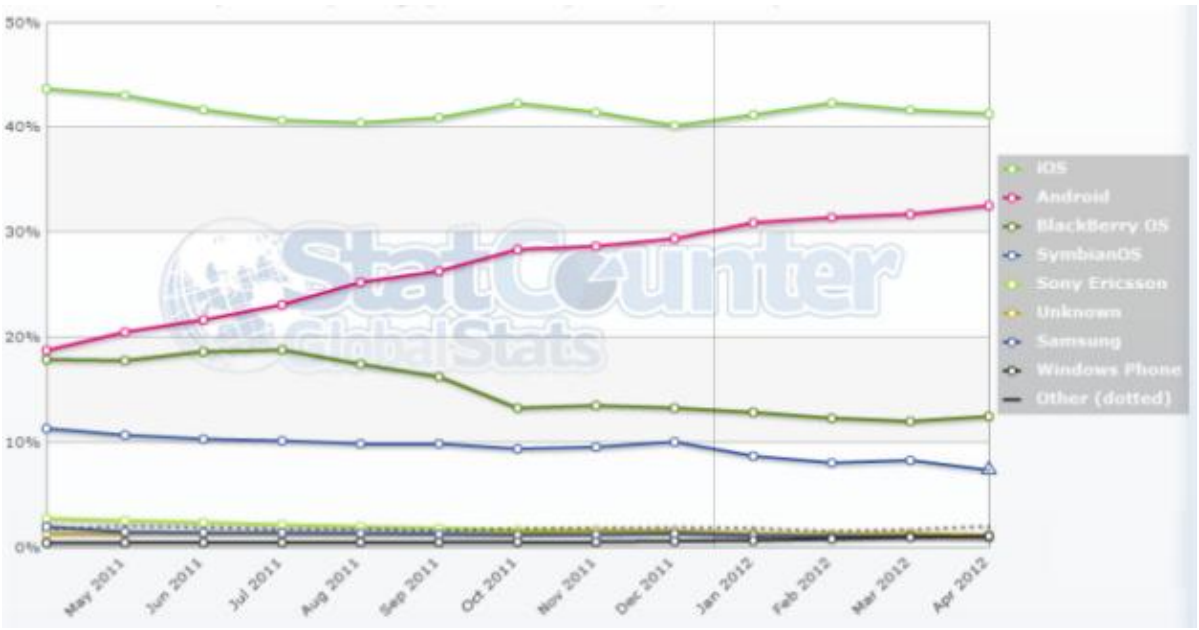

Figura 3.5 *Progresión de los sistemas operativos móviles en 2011/2012*

3.4.1 Introducción Android

Android es la plataforma de software que está revolucionando el mercado global de la telefonía móvil. Se trata de la primera plataforma móvil de código abierto basado en Linux que ha logrado introducirse en los principales mercados del mundo. El sistema permite programar aplicaciones en una variación de Java llamada Dalvik.

Una de las características más importantes de Android es que se pueden crear aplicaciones completas para aprovechar los recursos disponibles en el mercado ya que no existen diferencias entre las aplicaciones incorporadas y las creadas con el SDK.

La Figura 2.1 muestra la relación entre Android y el hardware sobre el que se ejecuta. Puede que lo más notable de Android sea que su naturaleza de código abierto no proporciona un entorno sofisticado, pero como la plataforma es abierta se puede modificar. Del mismo modo, se pueden obtener codecs multimedia de terceros y no es necesario depender de la empresa creadora (Google) para disfrutar de nuevas funciones.

Figura 3.6 *Núcleo Android.*

3.4.2 Introducción iOS

iOS (anteriormente denominado iPhone OS) es un sistema operativo móvil de Apple desarrollado originalmente para el iPhone, siendo después usado en el iPod Touch e iPad. Es un derivado de Mac OS X, que a su vez está basado en Darwin BSD. El iOS tiene 4 capas de abstracción: la capa del núcleo del sistema operativo, la capa de "Servicios Principales", la capa de "Medios de comunicación" y la capa de "Cocoa Touch". Todo el sistema se encuentra en la partición "/root" del dispositivo, ocupa poco menos de 500 megabytes.

Características y especificaciones actuales:

Interfaz de usuario intuitiva, basada en una pantalla multitáctil y un conjunto de componentes hardware internos (acelerómetros y giroscopios) que permiten interactuar con el s.o. realizando gestos comunes como mover el aparato para deshacer o rehacer, rotarlo para girar la imagen, deslizar el dedo para moverse por los diferentes menus y aplicaciones, etc..

Una pantalla principal (llamada "SpringBoard") donde están ubicados los iconos de las aplicaciones.

Una pantalla de estado situada en la parte superior para mostrar datos, tales como la hora, el nivel de batería o la intensidad de la señal.

Soporte para mensajería SMS y MMS

Cliente de correo (Mail)

Navegador web (Safari)

Soporte para videoconferencia

Soporte para la mayoría de los formatos multimedia estándar. Aunque cabe destacar que iOS no soporta Adobe Flash y Java.

Soporte para HTML5

Soporte multitarea únicamente para aplicaciones por defecto del sistema para prevenir el consumo excesivo de batería y mantener el rendimiento. A partir de la versión 4 se permite el uso de siete API's multitarea para aplicaciones de terceros: audio en segundo plano, VOIP, localización en segundo plano, notificaciones push, notificaciones locales, completado de tareas y cambio rápido de aplicaciones.

Capítulo 4
Funcionamiento del programa

4.1 Introducción

En este capítulo se mostrará el funcionamiento del servicio web "Chat ASP", cuya principal función es hacer posible la comunicación entre usuarios de distintas plataformas, tanto por texto como por videoconferencia.

Para cada dispositivo se ha adaptado la página para que la experiencia del usuario sea la más cómoda, independientemente de la plataforma desde la que se conecte. Para acceder a la aplicación basta con introducir la dirección en cualquier navegador de cualquier dispositivo.

A continuación se explicará todo lo referente al funcionamiento de la aplicación desde el punto de vista de la codificación y del funcionamiento.

4.2 Clases

Para la construcción de una sala de chat interactiva se requiere mantener a los usuarios actualizados tanto con los mensajes como con el estado de los demás usuarios. Para ello son necesarias las clases que se muestran a continuación:

4.2.1 ChatUser

```
public class ChatUser:IDisposable
      {

            #region Members
            public string UserID;
            public string UserName;
            public bool IsActive;
            public DateTime LastSeen;
            public int LastMessageReceived;
            #endregion
```

```
        #region Constructors
        public ChatUser(string id,string userName)
        {
                this.UserID=id;
                this.IsActive=false;
                this.LastSeen=DateTime.MinValue ;
                this.UserName=userName;
                this.LastMessageReceived=0;
        }
        #endregion

        #region IDisposable Members
        public void Dispose()
        {
                this.UserID="";
                this.IsActive=false;
                this.LastSeen=DateTime.MinValue ;
                this.UserName="";
                this.LastMessageReceived=0;
        }
        #endregion
}
```

Esta clase representa el usuario que acaba de conectarse. Se hace uso de ella para dar de alta a los nuevos usuarios inicializando los datos que se necesitaran más adelante para llevar a cabo la relación entre usuarios. Las propiedades más importantes son:

- IsActive es la variable que define si un usuario se encuentra en el sistema o si por el contrario ha salido de la aplicación. Ésta variable se pondrá a true al ejecutar la función JoinRoom, y a false con la función LeaveRoom.
- LastSeen que contiene el valor con la hora del último movimiento del usuario, ésto es útil para poder desconectar del chat a aquellos usuarios que se encuentren inactivos durante el tiempo que se estime.
- LastMessageReceived muestra el último mensaje recibido por el usuario.

4.2.2 ChatEngine

ChatEngine.cs

```
    #region Methods

        public static void CleanChatRooms(object state)
```

```
{
            lock (Rooms)
            {
                foreach (object key in Rooms.Keys)
                {
                    ChatRoom room = Rooms[key.ToString()];
                    if (room.IsEmpty())
                    {
                        room.Dispose();
                        Rooms.Remove(key.ToString());
                    }
                }
            }
        }
             public static ChatRoom GetRoom(string roomID)
             {
                    ChatRoom room=null;
            lock (Rooms)
            {
                if (Rooms.ContainsKey (roomID))
                    room = Rooms[roomID];
                else
                {
                    room = new ChatRoom(roomID);
                    Rooms.Add(roomID, room);
                }
            }
                    return room;
             }
             public static void DeleteRoom(string roomID)
             {
            if (!Rooms.ContainsKey(roomID))
                return;
            lock (Rooms)
            {
                ChatRoom room = Rooms[roomID];
                room.Dispose();
                Rooms.Remove(roomID);
            }
             }


             #endregion
    }
```

La principal función de esta clase es el mantenimiento de las salas del chat. Para ello se dispone de los métodos:

- DeleteRoom que se encarga de cerrar las salas en las que no haya ningún usuario

- getRoom devuelve la ID de la sala de chat o la crea si no existe. Es una de las funciones más usadas ya que su llamada se hace necesario para cualquier tipo de acción que se quiera realizar sobre una sala, ya sea mandar un mensaje o salir de una sala, siempre es necesario conocer la ID de la sala en la que se encuentra el usuario para poder actuar en consecuencia.

4.2.3 ChatMessage

ChatMessage.cs

```
public class Message
      {
            #region Members
            public string user;
            public string msg;
            public MsgType type;
            #endregion


            #region Constructors
            public Message(string _user, string _msg, MsgType _type)
            {
                  user = _user;
                  msg = _msg;
                  type = _type;
            }
            public  Message(string  _user,  MsgType  _type)  :  this(_user,  "",
_type) { }
            public Message(MsgType _type) : this("", "", _type) { }
            #endregion

            #region Methods
            public override string ToString()
            {
                  switch(this.type)
                  {
                        case MsgType.Msg:
                              return this.user+" says: "+this.msg;
                        case MsgType.Join :
                              return this.user + " has joined the room";
                        case MsgType.Left :
                              return this.user + " has left the room";
```

```
            return "";
        }
        #endregion

    }
```

Esta clase se encarga de los mensajes que muestra el servidor en la sala de chat. Estos mensajes son los que muestra el chat para diferentes acciones de los usuarios, como por ejemplo al entrar al chat o al salir de él. A continuación se muestra una imagen donde se ve el uso de esta función.

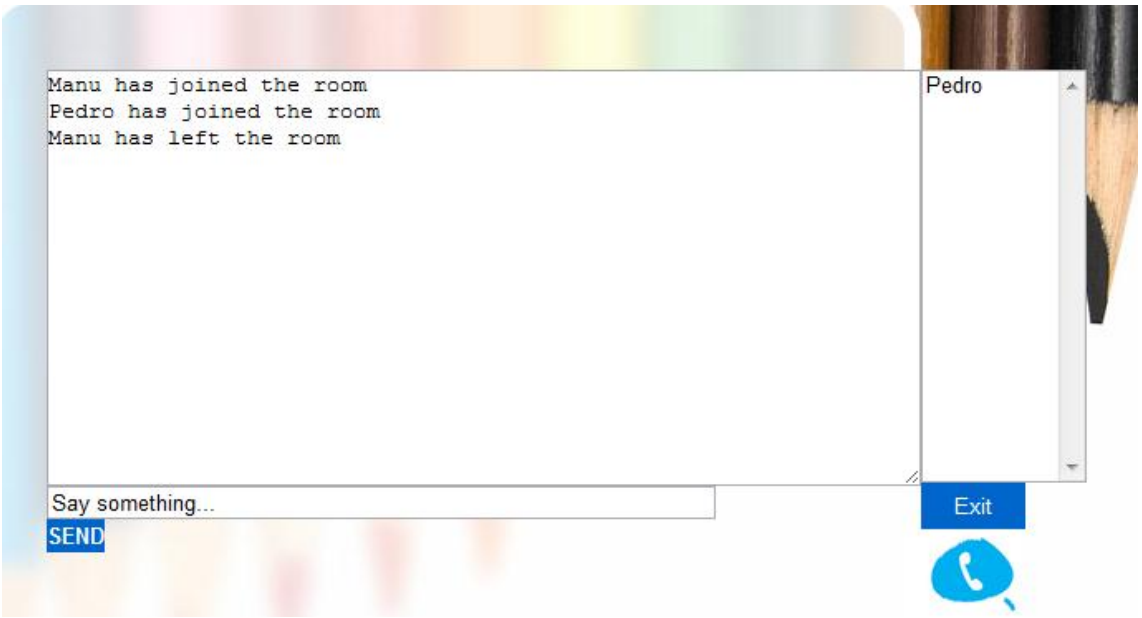

Figura 4.1 *Ejemplo de los mensajes de la clase ChatMessage*

4.2.4 ChatRoom

ChatRoom.cs

```
public string UpdateUser(string userID)
```

```
	{
			ChatUser user=this.GetUser(userID);
			int lastMsgID;
			List<Message>previousMsgs=this.GetMessagesSince(
user.LastMessageReceived,out lastMsgID);
			if (lastMsgID!=-1)
				user.LastMessageReceived=lastMsgID;
			string res=this.GenerateMessagesString(previousMsgs);
			return res;
		}

 public List<Message> GetMessagesSince(int msgid,out int lastMsgID)
		{
			lock(messages)
			{
				if ((messages.Count) <= (msgid+1))
					lastMsgID=-1;
				else
					lastMsgID=messages.Count-1;
				return  messages.GetRange(msgid+1  ,  messages.Count  -
 (msgid+1));
			}
		}
```

Estas funciones sirven para mantener a los usuarios actualizados. Su objetivo es recuperar todos los mensajes enviados en la sala de chat. Además esta clase contiene las funciones JoinRoom y LeaveRoom, cuya función es conectar al usuario a la sala de chat y desconectarlo.

4.2.5 Javascript

La página chat.aspx contiene un javascript cuya función es llamar a los métodos en el servidor usando AJAX. Para llamar a un método en el servidor asíncronamente en AJAX se tiene que poner este método en la página aspx como un método estático y publico aplicándole el atributo WebMethod.

Hay cuatro peticiones asíncronas que se realizan desde el javascript hasta el servidor, los dos primeros se envían periódicamente utilizando un temporizador Javascript, la solicitud updateUser se utiliza para obtener todos los mensajes que fueron enviados por los otros usuarios y actualiza el área de texto con estos mensajes. La solicitud updateMembers se utiliza para obtener todos los usuarios conectados al chat y actualiza el cuadro de la lista de usuarios de tal manera que los miembros que abandonan

el chat se eliminan de la lista y los miembros recién incorporados se agregan a ella. El código Javascript para estas dos peticiones se muestra a continuación.

```
                    function startTimers()
                    {
                           msgTimer = window.setInterval("updateUser()",3000);
                           membersTimer                                          =
window.setInterval("updateMembers()",10000);
                    }

                    function updateUser()
                    {
                        PageMethods.UpdateUser($get("hdnRoomID").value,
UpdateMessages);
                    }
                    function updateMembers()
                    {
                        PageMethods.UpdateRoomMembers($get("hdnRoomID").value,
UpdateMembersList);
                    }



                    function UpdateMessages(result)
                    {
                        $get("txt").value=$get("txt").value+result;
                        $get("txt").doScroll();
                    }
                    function UpdateMembersList(result)
                    {
                       // alert(result);
                        var users=result.split(",");
                       // alert(users.length);
                        var i=0;


                $get("lstMembers").options.length=0;

                while (i < users.length)
                {
                    if (users[i]!="");
                    {
                        var op=new Option(users[i],users[i]);

$get("lstMembers").options[$get("lstMembers").options.length]= op;
                    }
                    i+=1;
}
```

La clase PageMethods es generada por el AJAX *script manager* y ofrece un proxy para todos sus métodos de modo que se puede llamar a los métodos pasándole los parámetros y el nombre de la función. Esta función será llamada cuando el resultado del método llegue desde el servidor.

4.3 Funcionamiento

Cuando se ejecuta la aplicación, el usuario accede a la página Default.aspx donde se le pedirá que introduzca el nombre que va a utilizar en la sala de chat.

La página de chat consiste en un *text area* que muestra todos los mensajes de la sala de chat y una *listbox* donde aparecen todos los usuarios conectados. Se ha implementado la opción de que el usuario pueda mandar sus mensajes tanto al hacer click sobre el botón SEND, como al presionar la tecla enter. Para ello se ha utilizado la siguiente línea de código:

```
txtMsg.Attributes.Add("onkeypress", "return clickButton(event,'btn')");
```

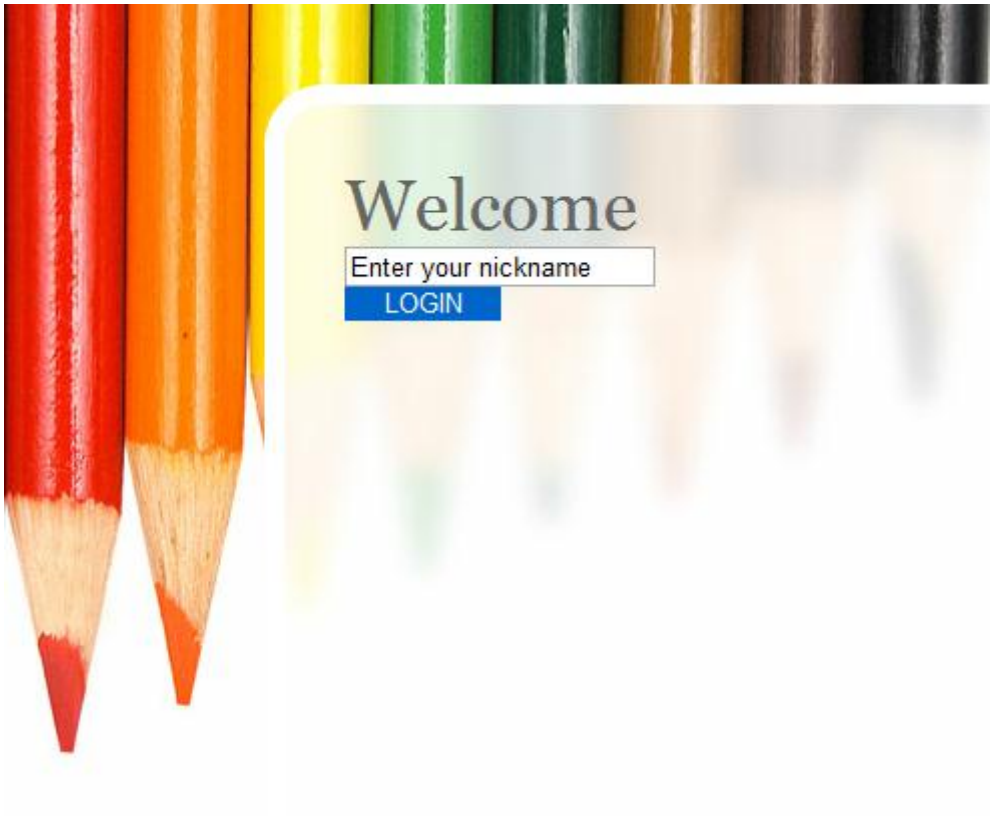

Figura 4.2 *Página principal de ChatASP*

Figura 4.3 *Sala de chat*

4.4 Funcionamiento (versión móvil)

Cuando el usuario accede al servicio web, lo puede hacer desde cualquier dispositivo que integre un navegador web. El problema surge a la hora de saber desde qué dispositivo esta accediendo el usuario para mostrarle la versión acorde a su plataforma. Para ello se ha implementado una solución en la cual el servicio web comprueba el dispositivo y muestra la versión del servicio web que se necesite. Para ello se ha utilizado el siguiente código:

```
    string u = Request.ServerVariables["HTTP_USER_AGENT"];
    Regex b = new
Regex(@"android.+mobile|avantgo|bada\/|blackberry|blazer|compal|elaine|fennec|hip
top|iemobile|ip(hone|od)|iris|kindle|lge |maemo|midp|mmp|netfront|opera
m(ob|in)i|palm(
os)?|phone|p(ixi|re)\/|plucker|pocket|psp|symbian|treo|up\.(browser|link)|vodafon
e|wap|windows (ce|phone)|xda|xiino", RegexOptions.IgnoreCase |
RegexOptions.Multiline);
```

```
Regex v = new Regex(@"1207|6310|6590|3gso|4thp|50[1-6]i|770s|802s|a
wa|abac|ac(er|oo|s\-
)|ai(ko|rn)|al(av|ca|co)|amoi|an(ex|ny|yw)|aptu|ar(ch|go)|as(te|us)|attw|au(di|\-
m|r |s )|avan|be(ck|ll|nq)|bi(lb|rd)|bl(ac|az)|br(e|v)w|bumb|bw\-
(n|u)|c55v)|vm40|voda|vulc|vx(52|53|60|61|70|80|81|83|85|98)|w3c(\-|
)|webc|whit|wi(g |nc|nw)|wmlb|wonu|x700|xda(\-|2|g)|yas\-|your|zeto|zte\-",
RegexOptions.IgnoreCase | RegexOptions.Multiline);
    if ((b.IsMatch(u) || v.IsMatch(u.Substring(0, 4)))) {
        Response.Redirect("MobileLogin.aspx");
    }
```

Una vez que se sabe desde qué dispositivo se está accediendo se muestra la versión indicada.

A continuación se muestra una captura de la versión móvil del servicio web:

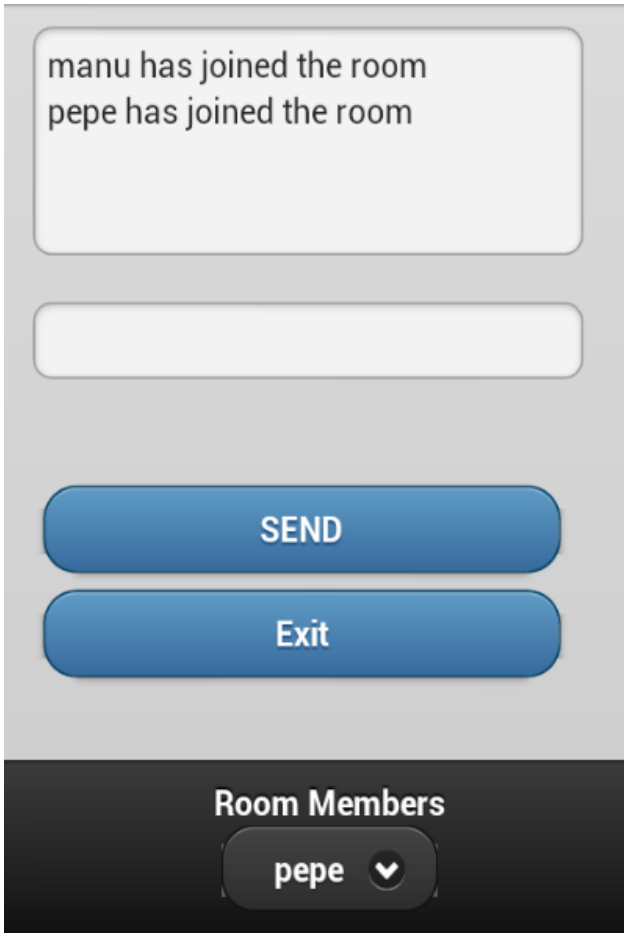

Figura 4.4 *Versión móvil ChatASP*

4.5 Videoconferencia

Una videoconferencia , es la comunicación simultánea bidireccional de audio y vídeo, permitiendo mantener reuniones personales o con grupos de personas situadas en lugares alejados entre sí. Adicionalmente, los equipos pueden tener la capacidad de mostrar o compartir presentaciones, archivos y videos y/o conectarse entre múltiples ubicaciones (multipunto) a la vez, resultando una comunicación total entre ellos.

El mercado de la videoconferencia está en constante crecimiento, debido sobre todo a la facilidad de su uso, y que permite la comunicación a un coste reducido, siendo solo necesaria una conexión a internet.

Para este proyecto se ha hecho uso de Skype como herramienta para tal fin, ya que es la aplicación por excelencia en el campo de la videoconferencia. Skype es un programa (una red de telefonía entre pares por Internet) que te permite llamar gratis a cualquier otro usuario de Skype, en cualquier parte del mundo. Sus principales características son su sencillez a la hora de su instalación y posterior utilización, así como su gran calidad tanto de audio como de video, y su seguridad. También cabe destacar su capacidad como aplicación multiplataforma.

Para realizar una llamada, basta con seleccionar al usuario al que se quiere llamar en el *listbox* y presionar el botón azul que se muestra a continuación:

Figura 4.5 *Botón Skype*

La decisión de implementar la videoconferencia a través de Skype es, aparte de por las ventajas citadas anteriormente, por su gran impacto en el mercado. La mayoría de usuarios tienen instalado Skype en sus PCs, y se considera una gran ventaja a la hora de analizar el servicio que se iba a utilizar.

4.6 Diagrama de secuencia

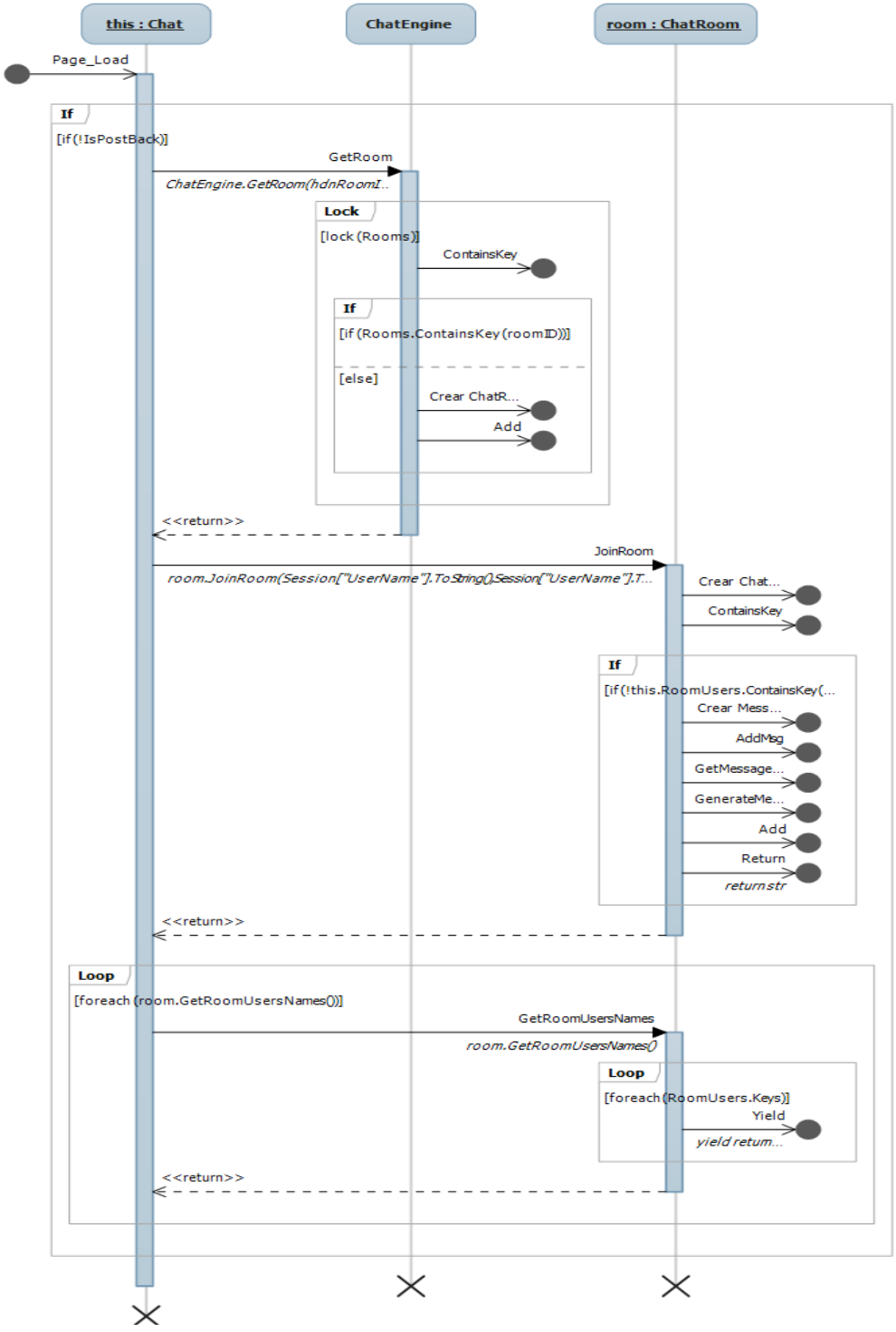

Figura 4.6 *Diagrama de secuencia (parte 1)*

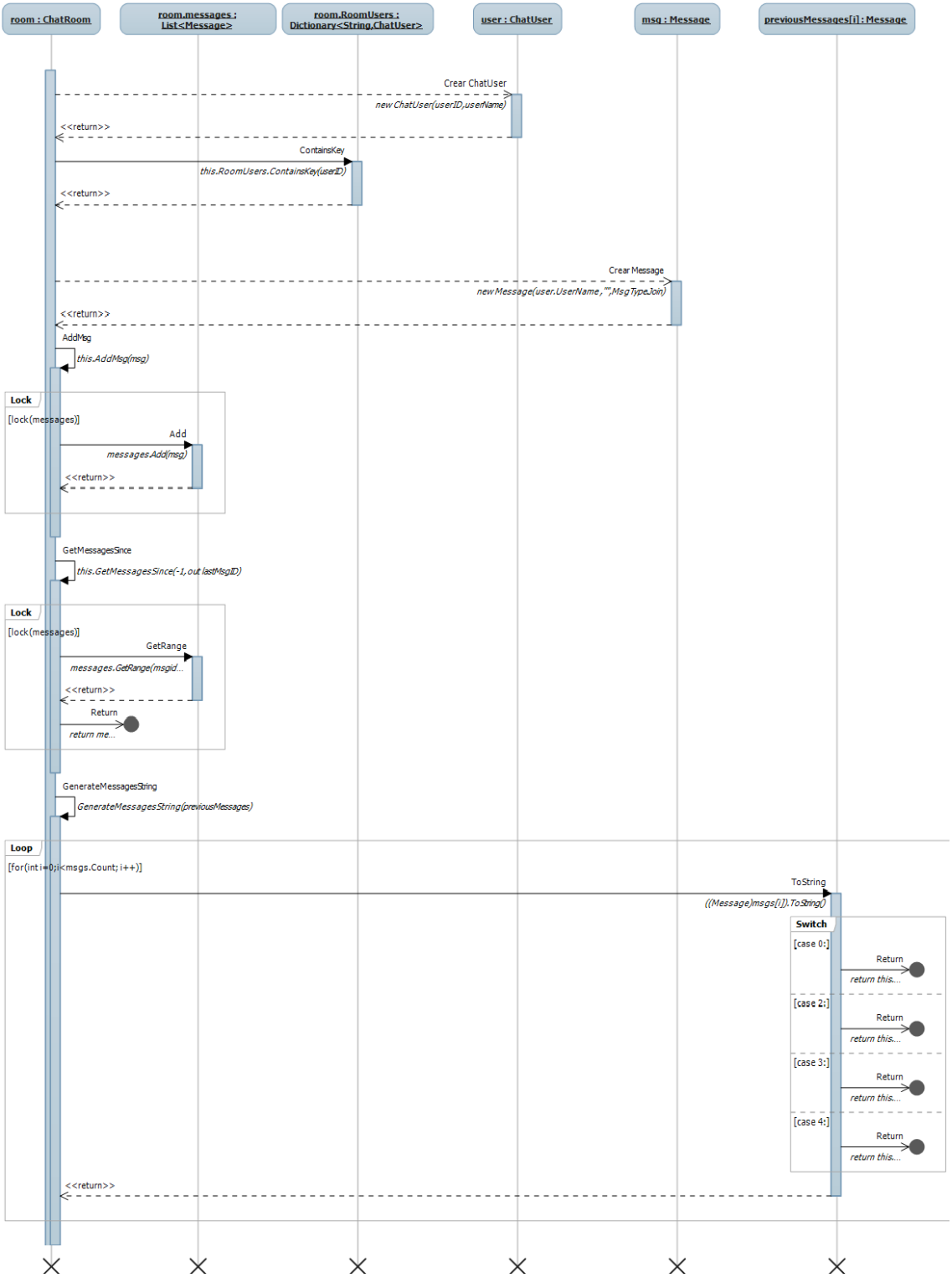

Figura 4.7 *Diagrama de secuencia (parte 2)*

4.7 Diagrama UML

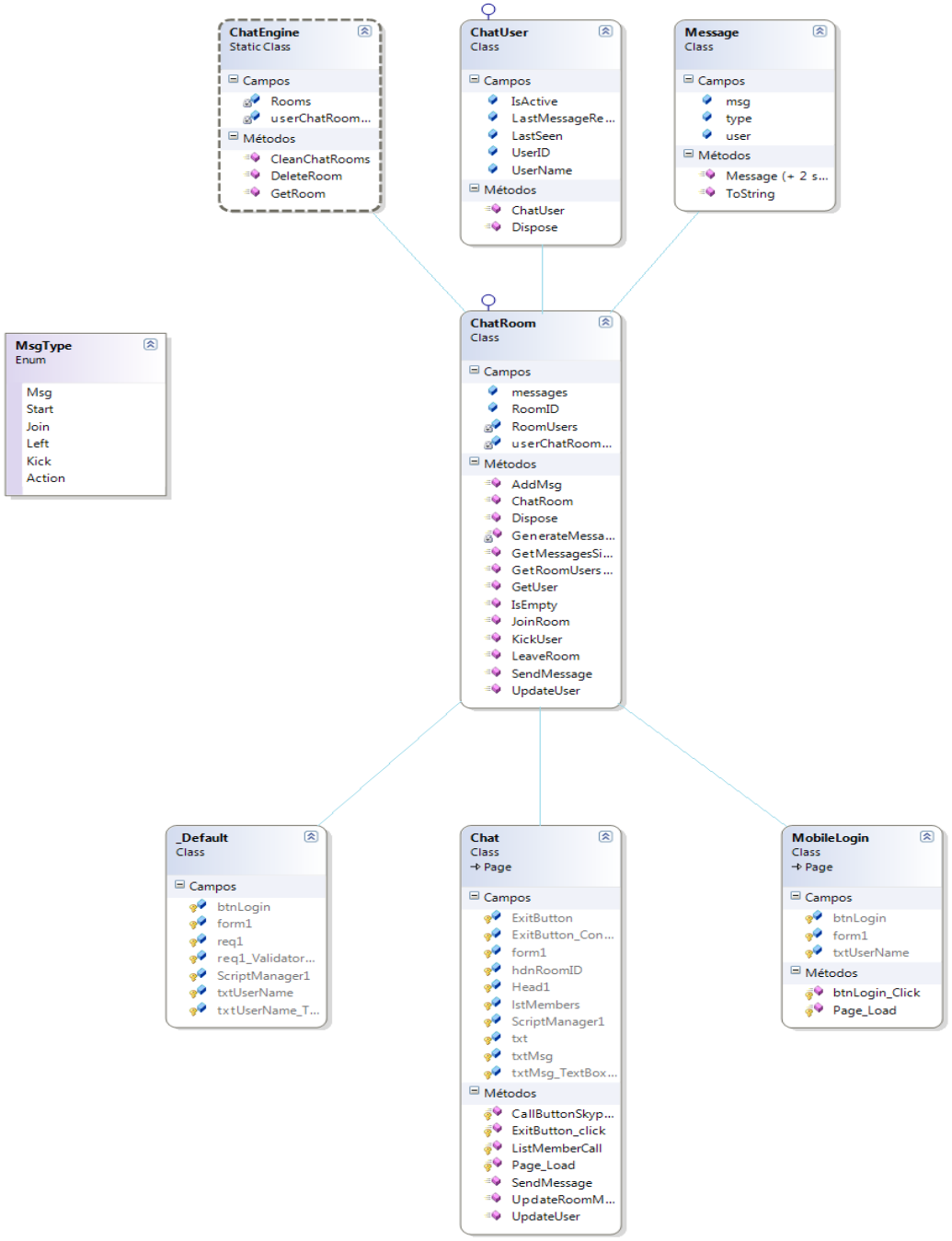

Figura 4.8 *Diagrama UML*

Capítulo 5
Caso de uso

A continuación se va a exponer un caso de uso del servicio web desarrollado en este proyecto. Al acceder al servicio web, éste se encarga de comprobar desde qué tipo de dispositivo se conecta el usuario. Para ello utiliza lo expuesto en el apartado 4.4, eligiendo así la página que va a mostrar. A continuación se muestra la pantalla de inicio para la versión pc:

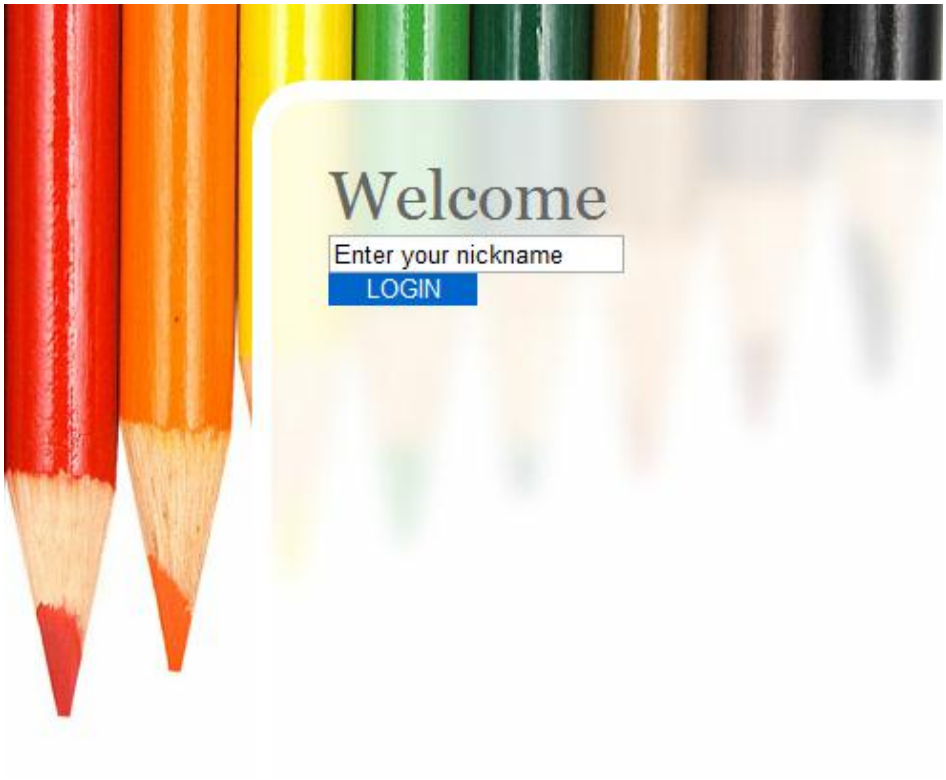

Figura 5.1 *Login versión pc*

En esta primera pantalla se tienen una serie de añadidos para hacer más agradable la experiencia del usuario, utilizando para ello la colección AJAX Control Toolkit. Se trata de un proyecto de código compartido construido sobre ASP.NET AJAX para escribir

controles y extensiones reusables y extensibles que aumenten la funcionalidad de los controles ASP.NET.

- En primer lugar se tiene un TextBoxWatermarkExtender, que se utiliza para añadir texto a los textboxs, pero que desaparece en el momento en el que el usuario hace click en él. Esto es muy útil para indicar al usuario dónde tiene que escribir.

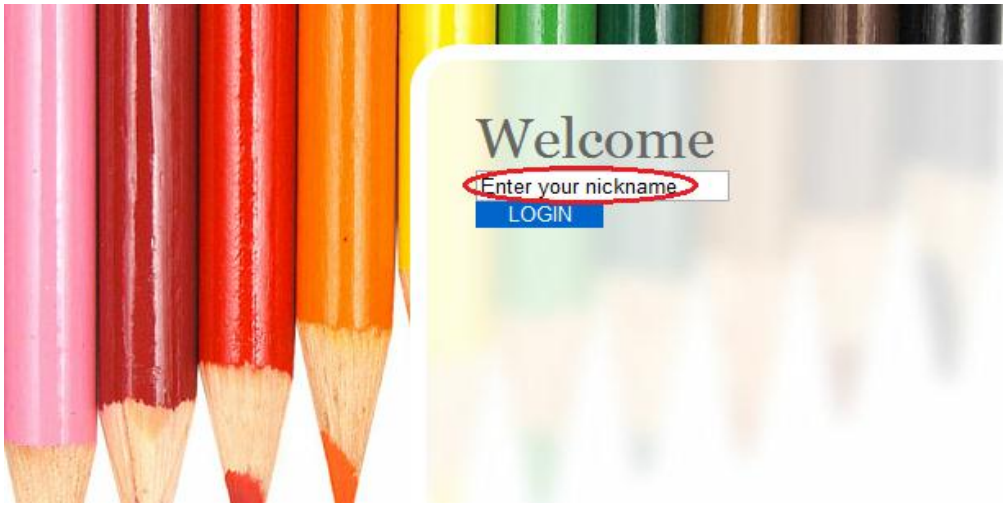

Figura 5.2 *TextBox Watermark Extender*

- También se han usado controles de validación para capturar lo tecleado por el usuario y comprobar si los datos proporcionados son correctos de acuerdo con la información solicitada. En este caso se ha hecho uso del control RequiredFieldValidator, usado para asegurarse de que el usuario no ha dejado el campo txtUserName en blanco. Pero además se ha utilizado un componente del AJAX Control Toolkit para mejorar visualmente el error a mostrar. Para ello se ha hecho uso del componente ValidatorCalloutExtender que muestra una pequeña ventana informando del error al usuario.

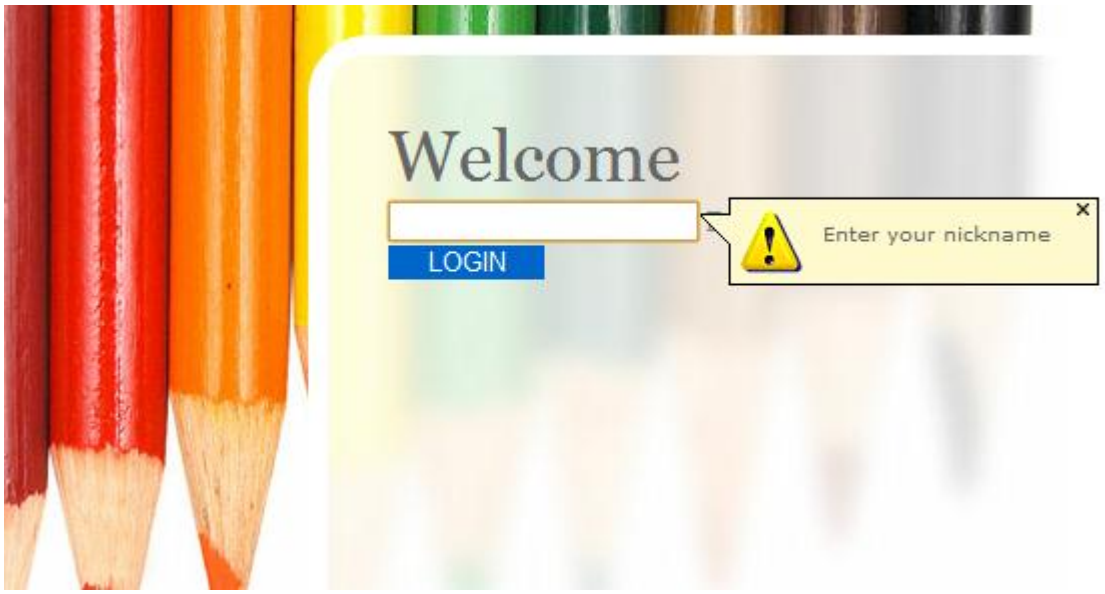

Figura 5.3 *Required Field Validator*

Una vez introducido un Nick valido, la aplicación guarda el nombre del usuario en una variable del tipo Session y redirige al usuario a la página chat.aspx, mediante la instrucción *Response.Redirect("Chat.aspx")*. Una vez en la página Chat.aspx, se ejecuta el procedimiento Page_Load, el cual, es el encargado de realizar las tareas necesarias para añadir al nuevo usuario a la sala, llevando a cabo las instrucciones que se muestran a continuación:

```
if (!IsPostBack)
    {
             hdnRoomID.Value = Request.QueryString["rid"];
             ChatRoom room = ChatEngine.GetRoom(hdnRoomID.Value);
            string prevMsgs=room.JoinRoom(Session["UserName"].ToString(),Session["UserName"].ToString()
            txt.Text = prevMsgs;
            foreach (string s in room.GetRoomUsersNames())
            {
                lstMembers.Items.Add(new ListItem(s, s));
            }

    }
```

Se puede observar cómo se solicita en primer lugar la ID de la sala, para posteriormente utilizar el método JoinRoom presente en la clase ChatRoom. Es en éste

método donde realmente se conecta al nuevo usuario, es decir, una vez llamado al método JoinRoom se puede asegurar que el usuario está totalmente operativo. Por último se añade el nuevo usuario al listbox.

Con la instrucción *if (IsPostBack)* se consigue que este trozo de código solamente se ejecute la primera vez que se accede al procedimiento Page_Load, es decir, cuando se accede por primera vez a la página Chat.aspx

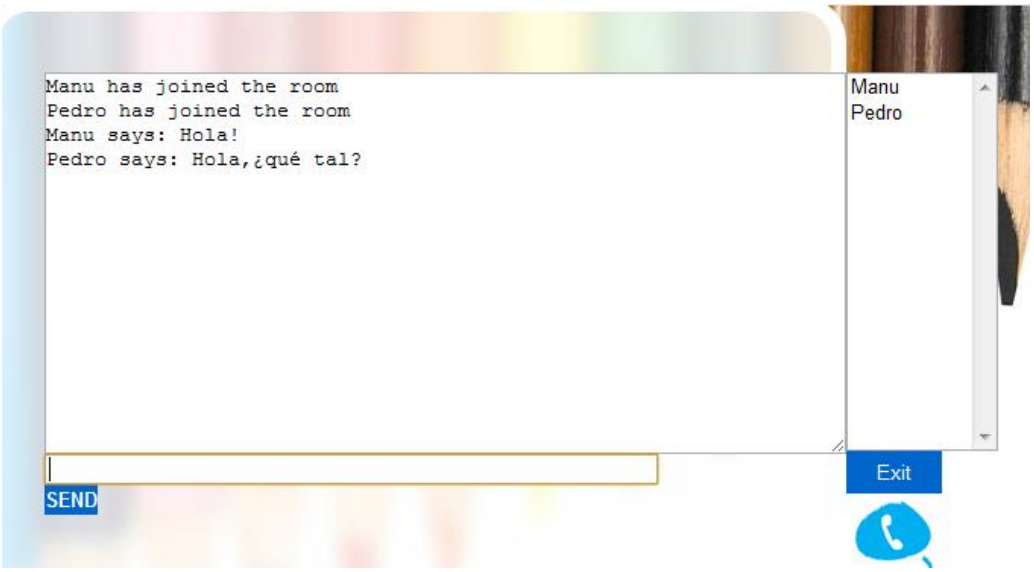

Figura 5.4 *Chat versión pc*

Cuando se quiere mandar un mensaje por el chat, el usuario escribe en el textbox destinado para ello y se inicia una secuencia que se explica a continuación:

- Al presionar Enter o pulsar el botón SEND, se activa la función de la clase scripts.js *button_clicked()*. Con esto se consigue llamar al método del servidor *SendMessage()* de una manera transparente para el usuario gracias a la clase PageMethods, que se encarga de crear un proxy para no entorpecer la experiencia del usuario.
- Ésta función llama a su vez al método SendMessage de la página chat.aspx, cuya función es conseguir la ID de la sala mediante el método GetRoom, y llamar al método SendMessage que se encuentra en la clase ChatRoom.
- El método SendMessage de la clase ChatRoom es el que realmente se encargar de mostrar por pantalla los mensajes de los usuarios.

A la hora de elaborar la sala de chat se ha optado por la sencillez para mejorar la experiencia del usuario. En la actualidad, la mayoría de páginas y servicios web optan por

la sencillez en sus diseños para favorecer el manejo de sus servicios, eliminando todo lo superfluo, y abarcar así a la mayoría de usuarios, facilitando su uso incluso para los usuarios menos expertos.

También es posible realizar una videollamada con otro usuario que esté conectado al chat. Para ello basta con seleccionar al usuario en la lista y pulsar el botón skype. A continuación se abrirá el programa skype, llamando directamente al usuario escogido. Para ello se utiliza el siguiente código:

```
string callperson = lstMembers.SelectedItem.ToString();
string a = "skype:" + callperson + "?call";
System.Diagnostics.Process.Start(a);
```

En esta página también se han añadidos controles del AJAX Control Toolkit, como el anteriormente usado TextBoxWatermarkExtender para indicar al usuario dónde tiene que escribir. Pero además se ha incorporado un control de seguridad a la hora de cerrar la aplicación. Para ello se ha utilizado el control ConfirmButtonExtender, cuya función es mostrar una ventana preguntando al usuario si realmente quiere cerrar la sesión. Con el uso de este control se evita salir de la aplicación accidentalmente. A continuación se muestra éste control en funcionamiento:

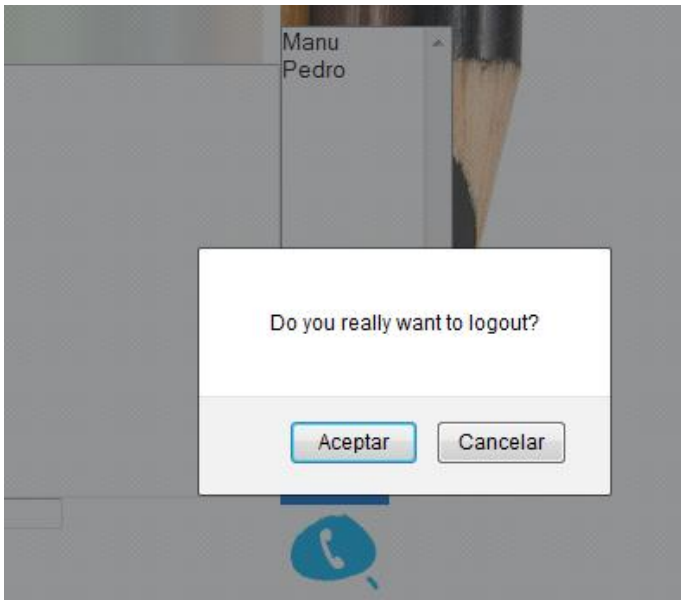

Figura 5.5 *Confirm Button Extender*

Si el usuario acepta, comienza una secuencia de acciones que se describen a continuación:

- Cuando el usuario hace click en el botón aceptar se activa el método *ExitButton_click()*, que, como se muestra a continuación, utiliza el método *GetRoom()* para conseguir la ID de la sala y llama a la función *LeaveRoom*, presente en la clase *ChatRoom*. Además pone la variable *Session* del usuario a null y redirige al usuario a la página principal.

```
protected void ExitButton_click(object sender, EventArgs e)
{

    ChatRoom room = ChatEngine.GetRoom(hdnRoomID.Value);

    room.LeaveRoom(Session["UserName"].ToString());


    Session["UserName"] = null;
    Response.Redirect("Default.aspx");

}
```

- La función LeaveRoom presente en la clase ChatRoom se encarga de marcar al usuario como inactivo, asignando a la variable IsActive el valor false y llamando a la clase Message para indicar por pantalla que el usuario se ha desconectado. Además, si la sala está vacía, se elimina.

```
public void LeaveRoom(string userID)
{
    ChatUser user=this.GetUser(userID);
    if (user == null)
        return ;
    user.IsActive=false;
    user.LastSeen=DateTime.Now;
    this.RoomUsers.Remove(userID);

    Message msg = new Message(user.UserName ,"",MsgType.Left);
    this.AddMsg(msg);

    if (IsEmpty())
        ChatEngine.DeleteRoom(this.RoomID);

}
```

La versión móvil del servicio web utiliza las mismas clases y métodos que la versión para pc, únicamente cambia la interfaz, que se hace más intuitiva, sobre todo para

pantallas táctiles. Para ello se ha utilizado el framework JQuery Mobile, que permite el desarrollo de interfaces web destinadas a móviles, gracias a la abstracción de la lógica de cada dispositivo, permitiendo así al desarrollador centrarse en el desarrollo del servicio.

Al acceder a la versión móvil aparece la pantalla de login, con un textbox para introducir el Nick, y un botón login para acceder al servicio. Al presionar el botón login se desencadena la misma secuencia descrita para la versión pc, con la única diferencia que ahora la aplicación nos redirige a la página MobileChat.aspx, creada especialmente para dispositivos móviles.

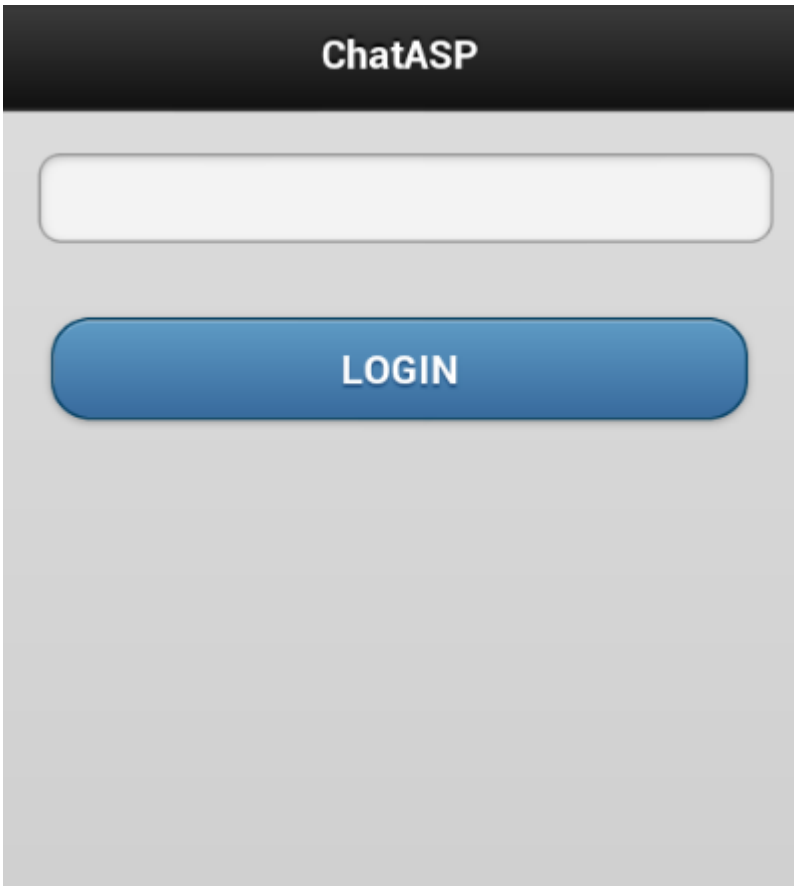

Figura 5.6 *Login versión móvil*

Una vez dentro de la página MobileChat.aspx, el usuario se encuentra con un textbox donde aparecerán las conversaciones que se lleven a cabo en la sala, aparte de otro textbox para escribir. También se tienen los botones SEND y Exit, para mandar los mensajes y para salir de la aplicación respectivamente. En la parte de debajo de la página aparece un botón que al presionarlo despliega una lista con los usuarios que se encuentren conectados en ese momento.

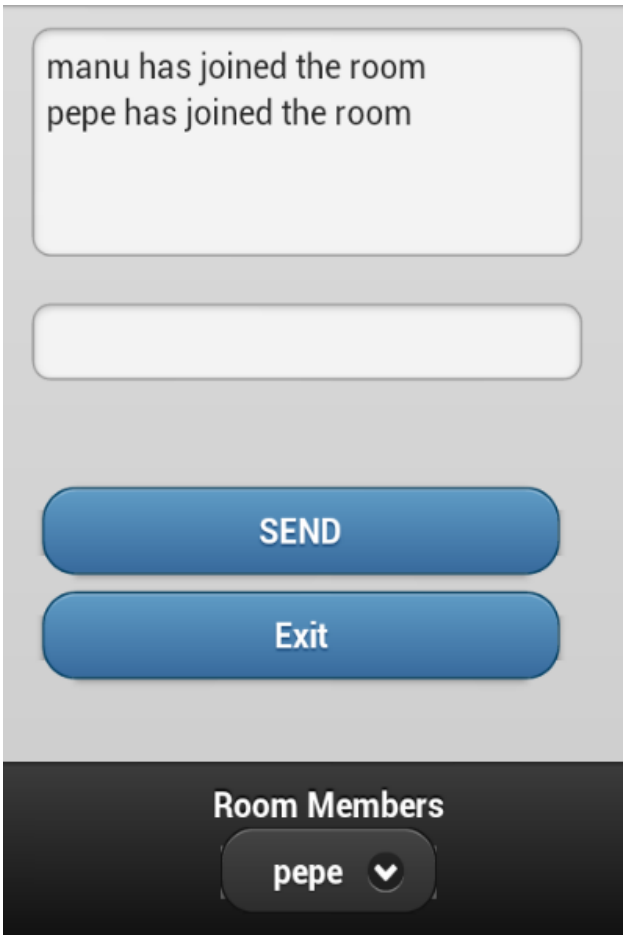

Figura 5.7 *Chat versión móvil*

Como se ha dicho anteriormente, todas las acciones que se realizan en la versión móvil utilizan los mismos métodos y clases de la versión pc, unificando así a todos los usuarios que accedan al servicio.

Capítulo 6
Conclusiones y líneas futuras

6.1 Conclusiones

En este proyecto se ha desarrollado un servicio web con el framework .NET y la herramienta Microsoft Visual Studio, dando como resultado el servicio web ChatASP, diseñado con el objetivo de permitir a usuarios de distintas plataformas comunicarse entre sí. Se ha conseguido gracias al gran potencial de los servicios web, una tecnología cada vez más utilizada entre las empresas y particulares para conseguir un entorno web con más utilidades y más fácil para el usuario.

El trabajo de este proyecto se ha dividido en cuatro apartados:

Estudio del framework .NET, así como de la tecnología ASP, precursora de los servicios web. Este estudio ha permitido una visión global del funcionamiento de .NET, permitiendo así conocer con gran detalle todas las características de este framework. También se ha podido comprobar las diferencias entre ambas tecnologías, y comprobar por qué es mucho más interesante el uso de .NET.

Estudio del entorno de desarrollo integrado Microsoft Visual Studio, así como del lenguaje de programación C#, llevándose a cabo mediante el aprendizaje de su arquitectura, características y funcionalidades.

Estudio de los servicios web, gracias al conocimiento de sus características y su utilidad para intercambiar datos entre aplicaciones.

Desarrollo del servicio web ChatASP para todo tipo de plataformas, realizando las tareas de codificación, depuración y pruebas. El objetivo del servicio web es la comunicación entre usuarios de distintas plataformas, consiguiendo la máxima transparencia y dando las mayores facilidades posibles para el usuario final.

6.2 Líneas futuras

El desarrollo y codificación de aplicaciones es una tarea larga y laboriosa, y una de sus principales características es la continua dedicación a la realización de

actualizaciones, corrección de errores o adicción de nuevas funcionalidades. Por ello, el futuro de este proyecto y más concretamente del servicio web "ASPChat" pasa por:

Desarrollar una aplicación para cada una de las plataformas móviles más importantes en la actualidad (Android, iOS, Windows Phone). En el desarrollo de este proyecto se ha optado por hacer una versión móvil del servicio web ASPChat, que, aunque se reduce ligeramente la integración con la plataforma, se consigue abarcar una mayor cuota de mercado, ya que se puede acceder al servicio fácilmente desde cualquier plataforma. La segmentación presente en el mercado de los smartphones hace que los desarrolladores tengan que hacer un esfuerzo doble (o incluso triple con la llegada de Windows Phone) al desarrollar sus aplicaciones, o arriesgarse a perder una cuota de mercado muy significativa.

Añadir nuevas funcionalidades tales como:

- Implementar una base de datos para almacenar los usuarios de nuestro servicio y evitar así la repetición de nicks y poder identificar un nombre con un usuario.
- A partir del punto anterior, desarrollar una cuenta admin, que tenga privilegios como por ejemplo desconectar a usuarios conflictivos.
- Utilizar Google Maps para situar la localización de los usuarios.
- Indicar cuándo un usuario está conectado desde un pc o un dispositivo móvil.

Referencias

[1] José Antonio González Seco. El lenguaje de programación C#.

[2] .NET Application Architecture Guide v2.0

[3] Charles Petzold, "Programming Windows with C# ". Ed. Microsoft Press, 2002. ISBN: 0735613702.

[4] Mridula Parihar, "ASP.NET". Ed. Anaya, 2002. ISBN: 8441513856

[5] Simon Robinson, "Professional C#",Wrox, 2003, Gerald Brose, Andreas

[6] Wikipedia.org - http://es.wikipedia.org/wiki/Microsoft_Visual_Studio

[7] Desarrolloweb.com - http://www.desarrolloweb.com

[8] Microsoft.com - http://msdn.microsoft.com

[9] Jquery – http://jquery.com

[10] Jquery mobile – http://jquerymobile.com