Introducción a Microsoft .NET

Enrique Hernández Orallo (ehernandez@disca.upv.es)

Recientemente, Microsoft ha presentado su plataforma .NET a bombo y platillo. Como suele ser habitual, Microsoft lo ha presentado como una revolución que va a afectar a la forma de trabajar de los usuarios y las empresas. Pero, ¿qué es exactamente .NET?. No es fácil contestar a esta pregunta, ya que en este caso, Microsoft no está vendiendo un producto con una finalidad concreta (como un sistema operativo, un procesador de textos, etc.), sino que el término engloba una serie de conceptos y tecnologías con el objetivo de cambiar nuestra forma de interactuar con la red.

Pero antes de describir lo que es .NET, voy a poner un ejemplo que puede servir para aclarar porque se llega a esta plataforma. Imaginemos que un usuario quiere ir a ver el concierto de año nuevo en Viena. Hasta la aparición de Internet la forma más fácil de gestionar el viaje era ir a distintas agencia de viajes, consultar precios y posibilidades y seleccionar la que más nos interesara.

Con la aparición de lo que podríamos denominar primera generación de Internet, el usuario podría consultar (antes de ir a la agencia) precios, posibilidades, ofertas, etc. Con esto ahorrábamos el tiempo de recorrer distintas agencias, ya que se iba a la agencia que más nos ha interesado en Internet. Pero en este caso, todavía teníamos que confiar la gestión a la agencia.

En la segunda generación Internet, la red ya nos proporciona una serie de servicios interactivos, como reserva de billetes, hoteles, etc. Siguiendo el ejemplo anterior, el usuario puede consultar la disponibilidad de billetes, hoteles, entradas, etc., y puede reservarlos y comprarlos por Internet. Pero existe un serio problema con este sistema: para ir al concierto necesitamos al menos tres cosas: un vuelo a Viena, el hotel y la entrada. Lo que no tiene sentido es reservar el hotel si no se tiene la entrada. Por tanto, para este caso seguía siendo más seguro realizar la gestión en la agencia.

La tercera generación de Internet, que está apareciendo actualmente, intenta solucionar este tipo de problemas. Usando el ejemplo del viaje, el cliente se conectará al servidor Web de una agencia de viajes y seleccionará lo que quiere hacer (un viaje a Viena, con hotel de 4 estrellas y una entrada al concierto de año nuevo). El servidor Web de la agencia de viajes se encargará, en el momento, de ir contactando (electrónicamente) con los servidores de los hoteles, compañías aeronáuticas y sala de conciertos para averiguar las posibilidades y su disponibilidad, con lo que al cliente le aparecerá directamente una oferta. Si el cliente acepta, se reservará automáticamente todo lo requerido (el vuelo, el hotel y la entrada), se le cargará en la tarjeta de crédito y se le enviará toda la documentación a la dirección que haya proporcionado el usuario.

Como se ve, la tercera generación de Internet va a suponer una mayor integración de los servicios ofrecidos por empresas a otras empresas (lo que se ha denominado *b2b:Business-to-Business*).

El principal cambio que supone esta tercera generación de Internet es que se pasa a hablar de servicios en vez de aplicaciones. El objetivo es por tanto la de proporcionar servicios que resuelvan problemas. Esto servicios los pueden utilizar personas directamente o bien otros sistemas, que a su vez pueden proporcionar sus servicios.

Para facilitar esta integración y el desarrollo de este tipo de servicios, Microsoft ha introducido su plataforma .NET. Además de .NET existen otras arquitecturas que tienen este mismo objetivo como la arquitectura Java J2EE de Sun y una serie de iniciativas para estandarizar esta integración.

PLATAFORMA .NET

La plataforma .NET en realidad no es algo radicalmente nuevo. Es un conjunto de tecnologías dispersas, que en muchos casos ya existían, que Microsoft ha integrado en una plataforma común con el objetivo de facilitar el desarrollo de este nuevo tipo de servicios de tercera generación.

Estos son los pilares de esta nueva plataforma:

- Integración: Proporcionar mecanismos para que una empresa pueda ofrecer servicios a otras empresas o clientes de una forma sencilla y rápida. En general, este tipo de servicios se suelen denominar B2B: *Business to Business* y B2C: *Business to Client*.

- Nuevos dispositivos: La forma más común de acceso a Internet hasta ahora ha sido el ordenador personal con sus limitaciones de movilidad. Pero recientemente han ido apareciendo una serie de dispositivos que permiten el acceso a servicios Internet de forma rápida y directa, como por ejemplo agendas electrónicas, teléfonos móviles, WebTV, videoconsolas, etc. Esto supone un cambio radical en la forma de acceder a este tipo de servicios.

Con estos objetivos, Microsoft .NET es una plataforma para construir, ejecutar y experimentar la tercera generación de aplicaciones distribuidas, que consiste en los siguiente elementos:

- Un modelo de programación pasado en XML.

- Un conjunto de servicios Web XML, como *Microsoft .NET My Services* para facilitar a los desarrolladores integrar estos servicios.

- Un conjunto de servidores que permiten ejecutar estos servicios (como *.NET Enterprise Servers*).

- Software en el cliente para poder utilizar estos servicios (como Windows XP, agendas electrónicas, etc.)

- Herramientas para el desarrollo como *Visual Studio.NET*.

En la figura 1 se muestran los elementos que pueden componer la plataforma .NET.

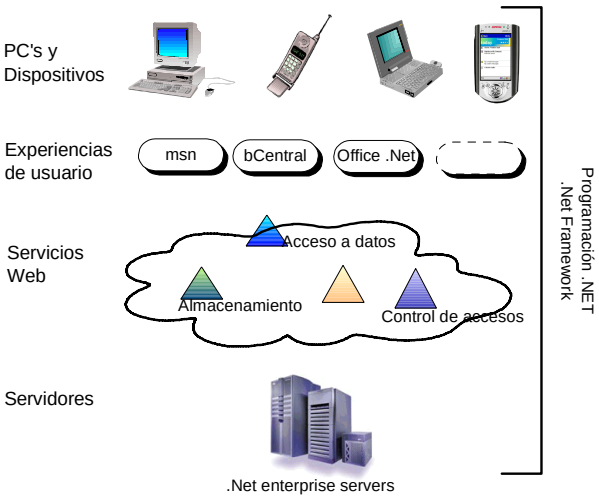

Figura 1: Elementos de la plataforma .NET

Una parte importante de esta plataforma es el software de los dispositivos clientes y servidores, que ha sido el mercado habitual de Microsoft. Para los dispositivos clientes, Microsoft planea integrar .NET en cualquier dispositivo imaginable, como PCs con Windows, agendas electrónicas con Pocket PC, teléfonos móviles, su consola de videojuegos X-Box, en WebTV, etc. Esto supone para las empresas aumentar el numero de potenciales clientes que puedan utilizar su servicios (ya no están limitados al PC).

Para poder ejecutar estos servicios, Microsoft introduce una serie de software englobado dentro de los .NET Enterprise Servers, como es el *Application Center*, *Commerce Server*, etc.

Estos servicios se ofrecerán al cliente a través de distintos canales, lo que Microsoft ha denominado *Experiencias de Usuarios*. Así, Microsoft ha pensado que MSN sea el canal para clientes domésticos y *bCentral* es el canal de comercio electrónico para empresas.

SERVICIOS WEB

Simplificando, un servicio Web es un programa que se puede acceder a través de Internet utilizando protocolos estándar [1]. Estos servicios se ejecutarán en un servidor Web, no en los PCs, permitiendo que los dispositivos que los utilicen sean más simples (simplemente se necesita un navegador Web) [2] .

Para implementar un servicio Web es necesario resolver varios problemas:

- Representación de los datos. Para poder compartir datos entre distintas organizaciones se necesita un estándar de representación de datos. Este estándar es XML [3] (ver tabla 1).

- Utilización del servicio. Se necesita un protocolo para definir cómo acceder y utilizar el servicio. Para ello se utiliza SOAP.

- Definición del servicio. Dado un servicio, para poder utilizarlo se necesita saber qué operaciones ofrece y cómo utilizarlas. Para esto, se utiliza el protocolo WSDL

- Publicación del servicio. Las empresas que proveen servicios y los clientes que quieran utilizarlos necesitan un mecanismo para que se conozcan, es decir, algo parecido a las paginas amarillas. Este el objetivo del protocolo UDDI.

Todo estos mecanismos son los que utiliza la plataforma .NET para implementar sus servicios Web, que en muchos casos denomina "Servicios Web XML".

Los servicios Web son componentes débilmente acoplados, lo cual quiere decir que se puede modificar la implementación en cada lado de conexión (sin modificar el interfaz) y el conjunto seguirá funcionando.

Estándares: Dado que el objetivo de los servicios Web es la integración entre empresas es necesario que se definan una serie de estándares para que esta comunicación se realice sin problemas. Para ello el consorcio W3C está realizando un proceso de estandarización de los protocolos que utiliza .NET, que están detallados en la tabla 1.

XML	*Extensible Markup Language*, XML es un metalenguaje de marcas que permite definir cómo es la información que se transmite. Esto permite una comunicación de datos entre distintos sistemas.
SOAP	*Single Object Access Protocol*. Este protocolo define como un cliente se comunica con un servicio usando HTTP y XML como mecanismo de intercambio de información.
WSDL	*Web Service Description Language*. Este protocolo basado en XML ha sido desarrollado conjuntamente por Microsoft e IBM. WSDL es un lenguaje en formato XML que define las operaciones que proporciona un servicio.
UDDI	*Universal Description, Discovery and Integration.* El UDDI es un directorio universal de Servicios Web, basado en XML que permite publicar, localizar y utilizar los servicios Web.

Tabla 1: Estándares de los servicios Web

Además, Microsoft ha desarrollado un conjunto de servicios Web orientados fundamentalmente al usuario, denominados *.NET My Services*. Basicamente, *.NET My Services* permite hacer llegar información personal a un servicio Web. Esto permite proporcionar información personal (siempre de un modo seguro y con el consentimiento explícito) a estos servicios. Esto puede ser útil, por ejemplo, a la hora de proporcionar nuestra dirección, datos bancarios, preferencias, etc.

Por supuesto, este es un tema conflictivo, ya que estamos tratando de un tema de confidencialidad de datos y que tiene que adaptarse a las normativas vigentes (principalmente la LOPD: Ley Orgánica de Protección de Datos).

ARQUITECTURA .NET

Una definición general de la arquitectura .NET podría ser la siguiente [4]: "Una plataforma independiente del lenguaje para el desarrollo de servicios Web".

La arquitectura .NET (*.NET Framework*) es el modelo de programación de la plataforma .NET para construir y ejecutar los servicios .NET. El objetivo de esta arquitectura es la de reducir la complejidad en el desarrollo de este tipo de aplicaciones, permitiendo a

los desarrolladores centrarse en escribir la lógica específica del servicio a desarrollar.

Esta arquitectura está compuesta por librerías y un ejecutivo tal como muestra la figura 2.

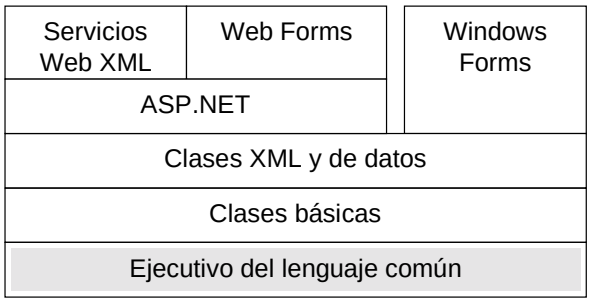

Figura 2: Componentes de la arquitectura .NET

El ejecutivo del lenguaje común (*CLR: Common Languaje Runtime*) es un soporte que permite ejecutar los servicios .NET en cualquier máquina que lo disponga. Esta basado en la idea de Java, que también tiene un módulo de ejecución independiente del sistema operativo donde se vaya a ejecutar. La gran diferencia con Java es que este ejecutivo es multilenguaje, esto es, no está limitado a un único lenguaje como Java. Esto permite al desarrollador utilizar una amplia variedad de lenguajes como C++, Visual Basic y C#.

Las librerías básicas proporcionan una serie de funcionalidades que son necesarias a la hora de desarrollar los servicios Web. Las *clases básicas* gestionan las operaciones más básicas como las comunicaciones, entrada/salida, seguridad, etc. Las *clases XML y de datos* gestionan el acceso a base de datos y la gestión de datos en XML.

El objetivo de las librerías *Servicios Web XML* es la de dar soporte para el desarrollo de aplicaciones distribuidas que ofrezcan servicios XML a otras entidades. Las *Web forms* permiten desarrollar la parte gráfica de una aplicación para la Web, mientras las *Windows Forms* están orientadas a implementar la parte gráfica de las aplicaciones clásicas para Windows.

En la figura 3 se muestra el modelo de ejecución de los programas .NET (que es muy parecida a la de Java).

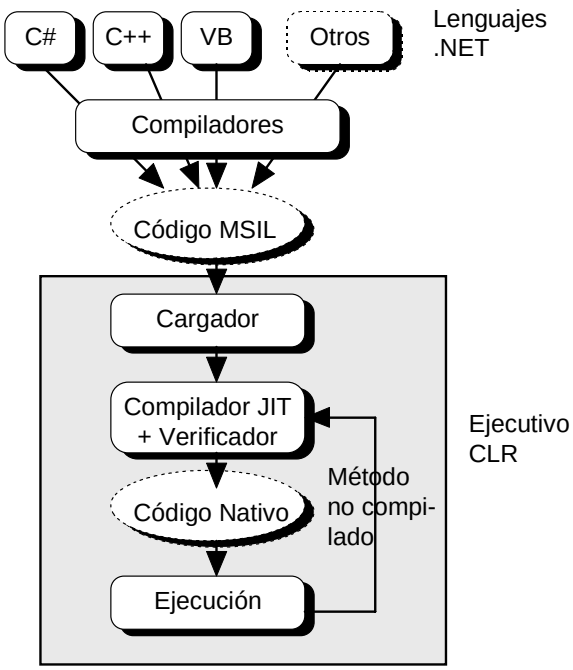

Figura 3: Modelo de ejecución .NET

Los compiladores producen código MSIL (*MicroSoft Intermediate Language*), que es un lenguaje intermedio que se puede ejecutar en la máquina virtual. Este código no es interpretado por el ejecutivo, sino que es compilado de nuevo en tiempo de ejecución (JIT: *Just in Time*) al código nativo de la máquina. Este código compilado no se ejecuta independientemente sino dentro de este ejecutivo. Esto se denomina código manejado, lo cual permite que el ejecutivo controle ciertos aspectos de la aplicación que ejecuta como son seguridad, gestión de memoria, compartición de datos, etc.

Aparte de Microsoft, existe actualmente un proyecto de Software Abierto para implementar toda esta arquitectura en Linux que se denomina MONO [5]. El objetivo es portar el ejecutivo CLR a Linux e implementar un compilador C#. Esto es muy interesante, ya que rompería uno de los objetivos de Microsoft, que era que la plataforma .NET sólo se ejecutase en sus sistemas operativos Windows.

VISUAL STUDIO .NET

Para poder desarrollar servicios para este tipo de arquitectura, Microsoft ha lanzado el nuevo entorno de desarrollo denominado *Visual Studio .NET*. El objetivo principal de este entorno de desarrollo es la de simplificar el desarrollo de aplicaciones Windows y servicios Web permitiendo la elección del lenguaje de

programación más adecuado (Visual Basic, C++ o C#).

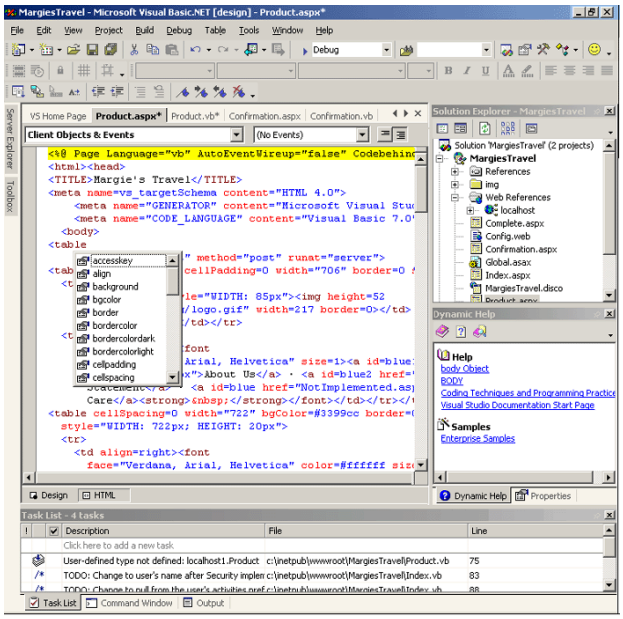

Figura 4: Visual Studio .NET

Aparte de poder elegir el lenguaje de programación hay que decidir qué tipo de aplicación se va a desarrollar, y en este caso se distingue ya entre las aplicaciones Windows tradicionales y los servicios Web, todo en el mismo entorno.

Visual Studio .NET sigue teniendo muchas de las características de versiones anteriores, como el entorno de edición, compilación y depuración integrado, gestión de proyectos complejos, diseño con notación UML. Pero con lo que respecta a la plataforma .NET se enumeran las principales características o mejoras al desarrollo que proporciona este entorno de desarrollo:

- Ejecutivo común: Como se ha comentado antes todos los lenguajes en la arquitectura .NET utilizan un módulo de ejecución común con librerías comunes. Con esto se termina con los distintos módulos de ejecución para cada lenguaje (como vbrun.dll para Visual Basic o msvc42.dll para Visual C++).

- Clases unificadas: Hasta ahora, cada lenguaje tenía su conjunto de clases o librerías para poder desarrollar programas Windows (las MFC en C++, VB Framework en Visual Basic). Esto implicaba que para cambiar de lenguaje, era necesario, aparte de conocer la sintaxis del lenguaje, conocer las librerías a utilizar. En la nueva plataforma .NET estas librerías o clases son comunes para todos los lenguajes, con lo que los desarrolladores no tienen que aprender una nueva librería cuando cambian de lenguaje.

- Integración multilenguaje: Además de los puntos anteriores, se incluye la posibilidad de llamada a métodos de otros objetos desarrollados en otros lenguajes e incluso su herencia. Esto permite desarrollar objetos en el lenguaje más apropiado para el problema a solucionar.

- ASP.NET: Esta librería proporciona un nuevo modelo para la creación de aplicaciones Web. Esto permite crear gráficamente páginas Web utilizando una serie de controles (desde el tipo campo de edición, hasta calendarios). Estos servicios se compilarán en el servidor y al cliente se le genera en tiempo de ejecución la página HTML apropiada para el navegador que utilice.

- ADO.NET: Esta librería proporciona un acceso común a los datos, ya sea en bases de datos o XML.

- Plataforma abierta: A este entorno de desarrollo se le pueden añadir herramientas o nuevos lenguajes de programación, de tal forma que estén perfectamente integrados en Visual Studio. De esta forma se van a poder utilizar distintos lenguajes de programación como Eiffel, Perl, Java e incluso lenguajes tan venerables como Cobol y Fortran.

Además de todo este soporte, Microsoft ha desarrollado un lenguaje nuevo de programación basado en C++ denominado C# (*C sharp*). El concepto de C# es muy parecido a Java, un lenguaje que elimina las complicaciones innecesarias del C++ pero manteniendo su potencia. El principal objetivo de C# es eliminar el uso de Java y C++, con el objetivo de reducir el coste de desarrollo de los servicios .NET.

Con esta nueva versión de Visual Studio, el lenguaje Visual Basic también ha sufrido un fuerte cambio en su sintaxis (lo que suele ser habitual en un lenguaje propietario). Esto implicará una conversión de la aplicaciones desarrolladas en versiones anteriores de Visual Basic.

Aparte de Microsoft, Borland ha anunciado que

Delphi y C++ Builder adoptarán la plataforma .NET con lo que no estamos limitados a utilizar Visual Studio para desarrollar servicios Web en .NET.

CONCLUSIONES

En este artículo se ha hecho una breve introducción de la plataforma Microsoft .NET, sus consecuencias y su arquitectura.

Microsoft .NET pretende imponer un cambio radical en la forma de desarrollar y utilizar las aplicaciones en la red. Se cambia del concepto de aplicaciones a servicios. Estos servicios pueden utilizarse tanto para dar soporte a personas, como para gestionar negocios, o incluso ser ensamblados para construir aplicaciones más complejas.

Para implementar esta nueva arquitectura, Microsoft ha utilizado una serie de tecnologías y estándares que junto a la herramienta Visual Studio .NET permiten desarrollar servicios Web de forma eficiente.

Dado el poder que tiene Microsoft y lo acertado de la propuesta, no es difícil predecir que esta plataforma va a tener éxito. Pero también es cierto que éste no será rápido porque implica un cambio drástico en el desarrollo (incluso las aplicaciones existentes desarrolladas con herramientas de Microsoft requieren una compleja adaptación). Los aspectos más negativos de esta plataforma es la creación del lenguaje C# como respuesta a Java. Ninguno de los argumentos utilizados para crear este lenguaje son convincentes, ya que tiene los mismos objetivos de un lenguaje que ya existe y que es ampliamente utilizado como es Java. Como suele ser habitual en Microsoft, en este caso han primado las decisiones comerciales sobre las técnicas.

También existen otras plataformas que solucionan los mismo problemas y que ya están en el mercado (como la plataforma Java J2EE), con lo que el mercado para este tipo de soluciones (afortunadamente) se repartirá.

REFERENCIAS

1. Mary Kirtland, "A Platform for Web Services", Microsoft Developer Network http.//msdn.microsoft.com
2. Vaughan-Nichols, "Web Services: Beyond the Hype", IEEE Computer, Febrero 2002
3. Ramón Montero, "XML, el lenguaje universal", Manual formativo Acta 13, 1999.
4. Meyer, B. ".NET is coming", IEEE Computer, Agosto 2001.
5. de Icaza, B. Jepson, "Mono & .NET Framework: Una alternativa Open Source". Dr Dobb's España. Nº1, Marzo 2002.
6. Microsoft .NET white papers. http://www.microsoft.com/net/
7. "Microsoft .NET: Construyendo la 3ª generación de Internet", Monográfico Byte Nº 3, MKM Publicaciones, 2001.