

Web Services, un ejemplo práctico

Por [Nicolás Tedeschi](#)

Contenido

[Introducción](#)

[Características de los Web Services](#)

[¿Qué es SOAP?](#)

[Un ejemplo práctico](#)

[Hacia dónde vamos](#)

[Referencias](#)

Introducción

¿Alguna vez pensaste de qué forma poder integrar aplicaciones creadas en lenguajes y plataformas diferentes, a través de Internet o bien en tu propia Intranet basándote en estándares? Bien, si lo pensaste o si no lo has hecho, la respuesta más apropiada a este paradigma son los llamados **Web Services**.

[Principio de la página](#)

Características de los Web Services

Un desarrollador puede incluir en sus sitios **soluciones sentencias**, es decir, instrucciones que consuman Web Services de terceros o propios como por ejemplo aquellos que proporcionan los datos meteorológicos para una localidad determinada, las cotizaciones de determinadas monedas, la cartelera de películas, el calendario o agenda de un especialista médico, etc.

Pensando un poco más en forma comercial, ¿Qué pasaría si por ejemplo estuvieras trabajando en tu procesador de texto en un idioma para el cual no tienes un corrector ortográfico ni sintáctico instalado (quizás no exista para instalar), pero deseas realizar la revisión del documento a toda costa? Bien, perfectamente podría haber una opción en el menú de este procesador que de alguna forma localice un Web Service en Internet que brinde esta funcionalidad, y lo más interesante aún para quien lo haya desarrollado es que puede solicitar al usuario que se suscriba para su uso. Como ves, todos ganan en esta transacción.

El ejemplo anterior muestra una realidad a la que no podemos estar ajenos. Es un replanteo de la estrategia utilizada por los desarrolladores quienes ahora, al realizar una aplicación, no deben pensar únicamente en el lugar físico donde la misma va a ejecutarse sino en que esa aplicación deberá estar interconectada con otras computadoras, corriendo otras aplicaciones quizás en otras plataformas y lenguajes, pero usando protocolos y estándares universales. El intercambio se intensificará muchísimo más y quizás existan por ejemplo “proveedores de dominios de datos” como ser los países, de forma tal que la aplicación que yo realice en lugar de crear toda la lógica para manejar las tablas y el cargado de los datos para el concepto PAIS, se limite a consumir un Web Service que me tome esta información de algún lugar en Internet. Imagino una reutilización aún mayor de funcionalidades y una colaboración e intercambio de lógica a nivel mundial. Quizás sea muy ambicioso en este planteo.

Pasando al terreno más técnico y práctico de este artículo, existen algunas consideraciones y conceptos que es necesario mencionar para comenzar a entender el tema:

- Un Web Service puede ser registrado para poder dejarlo a disposición de otros usuarios y para que los mismos puedan localizarlo. Un mecanismo para registrar estos servicios es por medio de UDDI, sigla que corresponde a **Universal Description , Discovery and Integration**, un “repositorio de Web Services”. Para registrar un servicio tendrás que tener en cuenta que debes suministrar la información de tu empresa, en qué categorías ubicarías tu servicio y la interfaz a utilizar para consumir este servicio.
- El mecanismo utilizado por un Web Service para especificar de qué forma hay que proporcionarle los datos, de manera tal que cualquiera pueda interactuar con el mismo, es por medio de lenguaje **XML**. Esta información se almacena en un archivo llamado WSDL (**Web Services Description Language**), el cual

contiene un documento XML junto con la descripción de ciertos mensajes SOAP y cómo deben intercambiarse, así como también dónde está el recurso del servicio y con qué protocolo debe dialogar quien lo consume.

- El protocolo de comunicación utilizado es el **SOAP** generalmente, el cual es relativamente sencillo de utilizar.
- Los Web Services utilizan protocolos comúnmente conocidos y difundidos tales como el formato XML, TCP/IP como protocolo de transporte y HTTP como protocolo de transferencia de hipertexto.

[Principio de la página](#)

¿Qué es SOAP?

SOAP es un protocolo que define el formato XML para los mensajes de intercambio en el uso de un Web Service. Para aquellos programadores que solían utilizar llamadas del tipo RPC, SOAP también las soporta.

Adicionalmente, es posible mediante [SOAP](#) definir un mensaje HTTP y este punto es de especial interés puesto que el protocolo imprescindible para Internet es HTTP.

Recomendación: Para comenzar a entender este tema es recomendable el uso del [Microsoft SOAP Toolkit Version 3.0](#) (pasaje de COM a SOAP).

[Principio de la página](#)

Un ejemplo práctico

A partir de ahora describiré en unos pocos pasos un ejemplo práctico y sencillo de creación de un Web Service y una muestra de cómo consumirlo desde una aplicación cliente, en este caso una simple planilla de Microsoft Excel:

Paso 1: Lo primero será crear un proyecto **Visual Basic** del tipo **ASP .NET Web Service**, al que llamaremos **DameCotizacion** (Ver [Figura 1](#)):

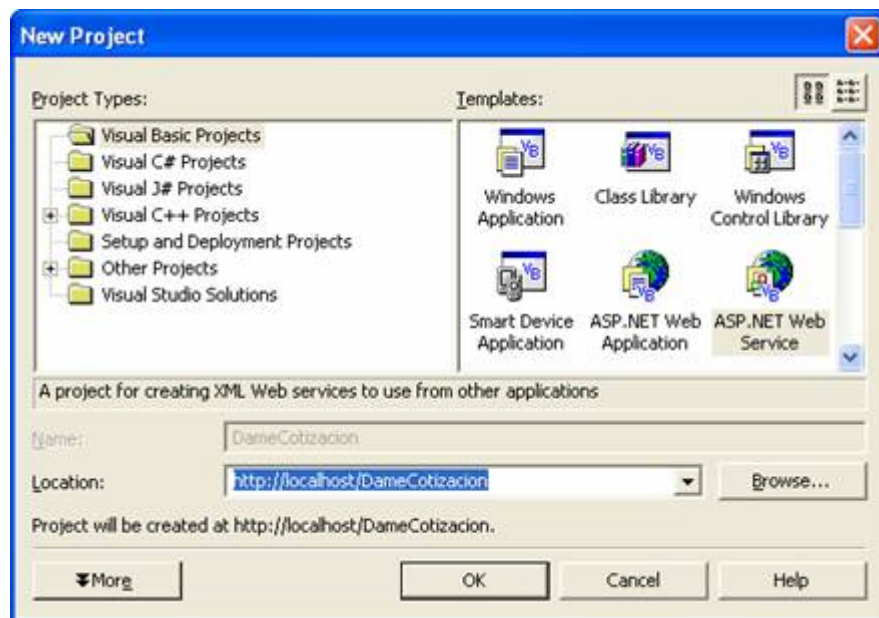


Figura 1. [Volver al texto.](#)

Paso 2: Al archivo **Service1.asmx**, que se crea una vez generado el proyecto, lo renombramos **DameCotizacion.asmx** y lo establecemos como **Página de Inicio del proyecto** (Ver [Figura 2](#)):

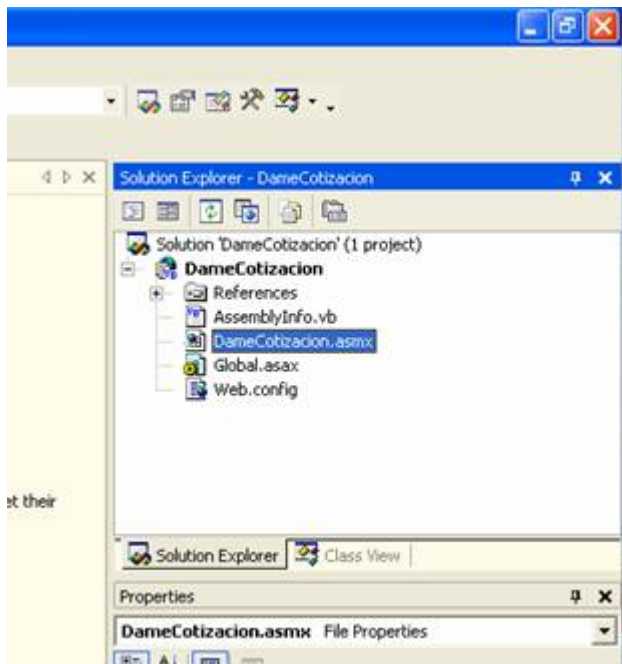


Figura 2. [Volver al texto.](#)

Paso 3: Ingresando a la ventana de código del archivo **DameCotizacion.asmx**, en la zona del **<webMethod(>**, agregamos el siguiente código:

```
<WebMethod(>
    Public Function GetCotizacion(ByVal strmoneda As String) As String
        'Objetivo: Devolver cotización para una moneda en pesos uruguayos
        '           Obviamente esto es un ejemplo por lo que esta info que
        '           se presenta estática se tomaría de una base de datos
        'Acepta: strmoneda - un id para la moneda de dos caracteres.
        'Devuelve: cotizacion de dicha moneda en pesos uruguayos

        Select Case UCase(Trim(strmoneda))
            Case "DO"
                'dolar
                Return "30"
            Case "RE"
                'real
                Return "9.9"
            Case "EU"
                'Euro
                Return "33"
        End Select
    End Function
```

Para verificar el correcto funcionamiento de esta aplicación, vamos a ejecutarla. Para ello apretamos **F5** y el resultado esperado se verá en **Internet Explorer**. Como puedes ver, se ofrece el método **GetCotizacion()** definido en el código anterior (Ver [Figura 3](#)):

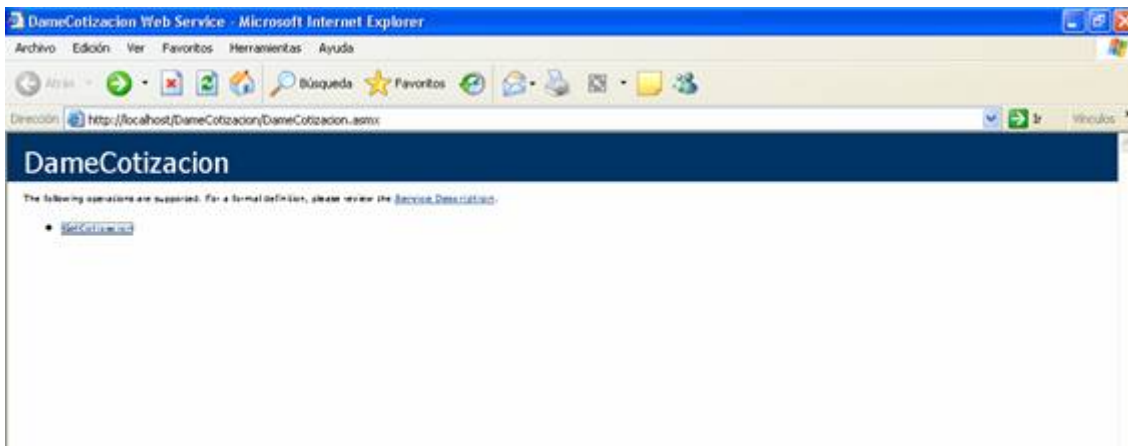


Figura 3. Volver al texto.

Si cliqueamos sobre este método, podremos ver la especificación del mismo y la definición del tipo de intercambio de mensajes (Ver Figura 4):

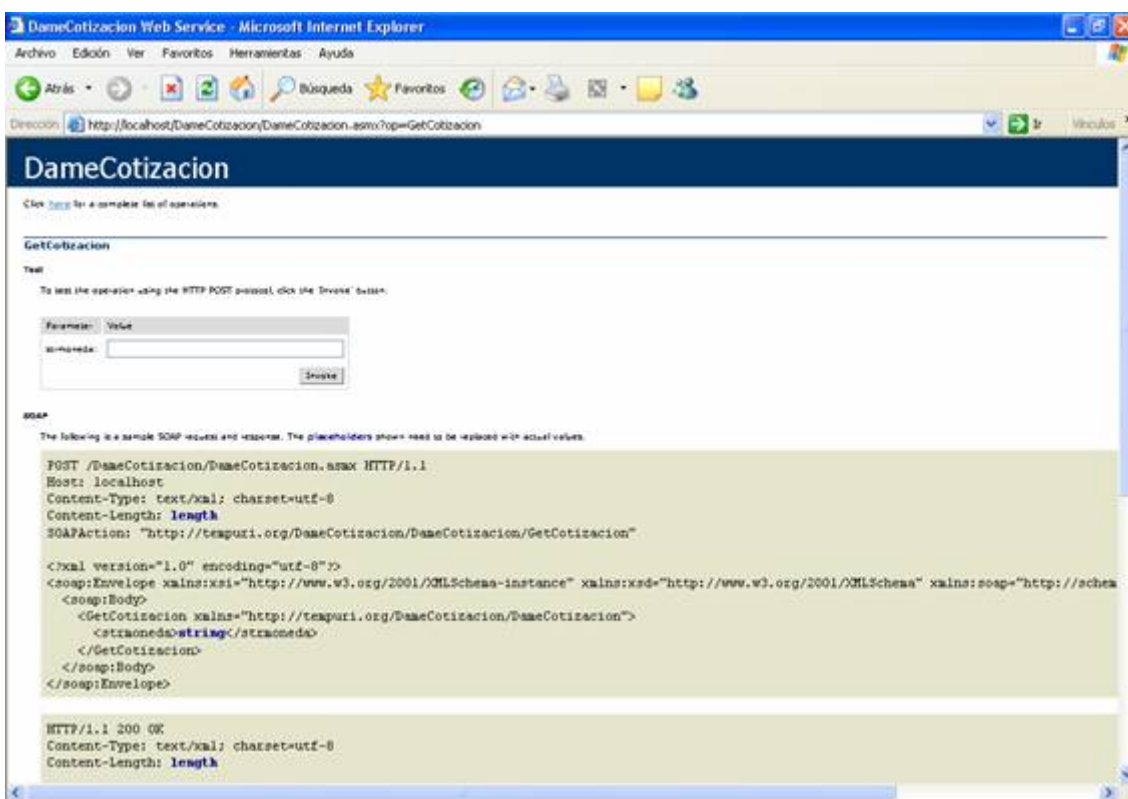


Figura 4. Volver al texto.

Paso 4: Podemos comprobar su funcionamiento colocando el valor **EU** (código que establecimos para el Euro) y cliqueando en **Invoke**. El valor esperado por el método es del tipo **string** y deberá ser uno de los tipo de monedas (**DO**, **RE** ó **EU**) definido en nuestro método.

El resultado debería ser un mensaje en XML como se muestra en la Figura 5, mostrando el valor definido para la moneda de código EU:

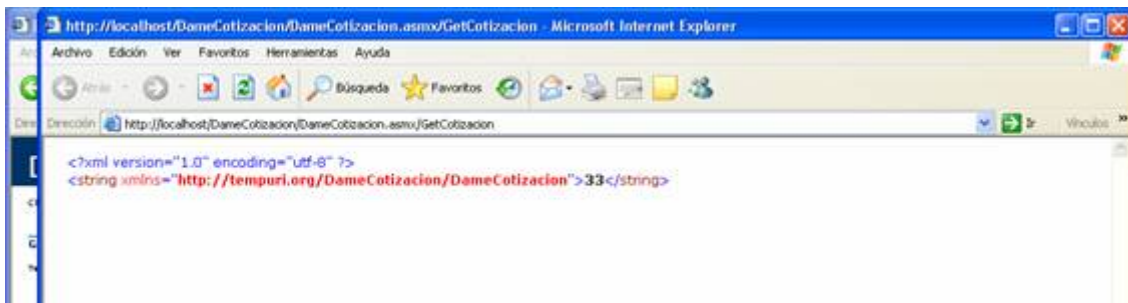


Figura 5. Volver al texto.

Pongamos a trabajar nuestro Web Service en una aplicación práctica. Supongamos que tenemos una planilla de Microsoft Excel donde tenemos artículos cuyos precios están en su moneda original y queremos que aparezca su valor en **Pesos Uruguayos**. Para esto consumiremos el Web Service creado en los pasos anteriores (DameCotizacion) que proporcionando el código de la moneda me devuelve la cotización correspondiente.

Nota: Para este ejemplo debemos tener instalado el paquete **Microsoft Web Services Toolkit**.

Paso 5: Crearemos una planilla de Microsoft Excel como se muestra en la Figura 6, donde agregaremos un botón cuyo nombre y *Caption* será **Cotizar**:

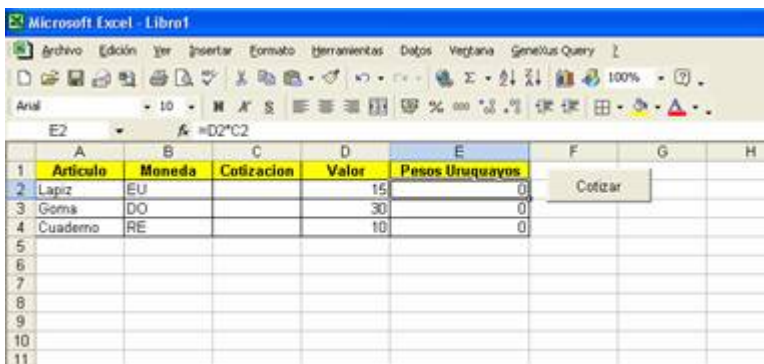


Figura 6. Volver al texto.

Paso 6: En la celda E2, la fórmula para calcular el valor del artículo en Pesos Uruguayos es **=D2*C2**. La columna **Cotización** será alimentada una vez que se oprima el botón **Cotizar**, el cual disparará un evento que consumirá el Web Service **DameCotizacion** y retornará en cada celda **Cotización** el valor correspondiente.

Paso 7: Haciendo doble clic sobre el botón **Cotizar** ingresaremos a la ventana de código Visual Basic posicionados en el evento **Click** de dicho botón.

Previo a esto último, relacionaremos nuestro Web Service a nuestra planilla mediante el uso de la herramienta Microsoft Web Services Toolkit (**Paso 8**). Para esto, desde el menú Herramientas de la ventana de código Visual Basic, seleccionamos la opción "Web Service References ..." (Ver Figura 7):

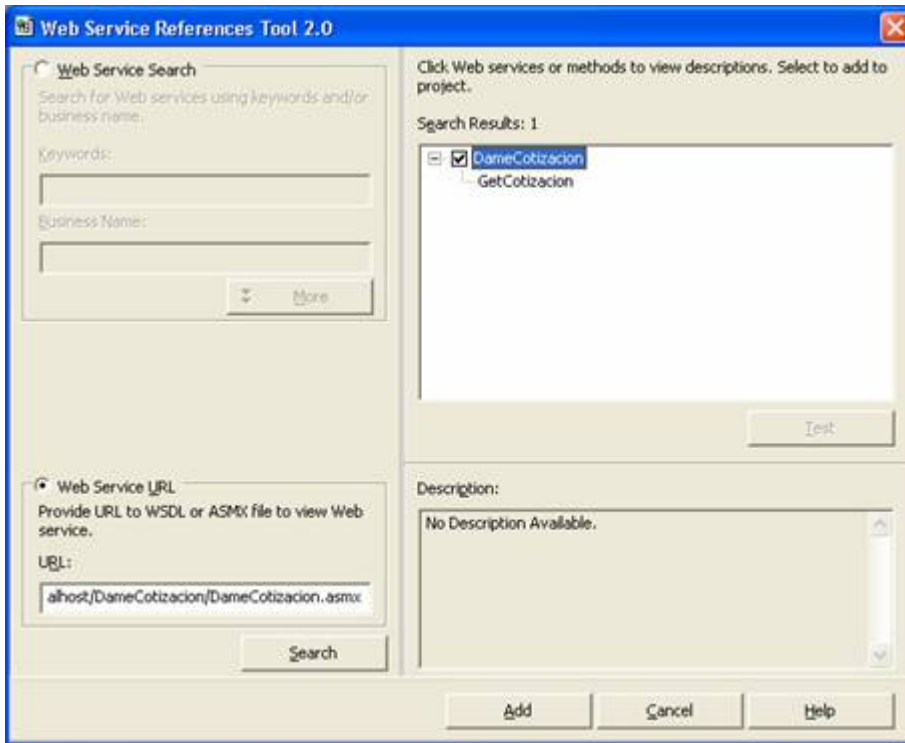


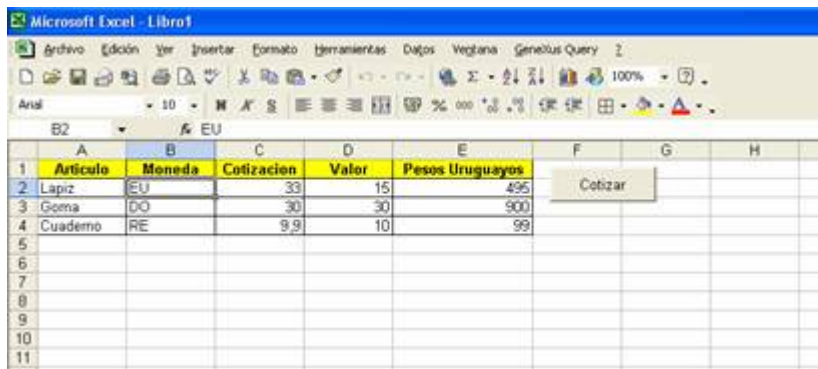
Figura 7. [Volver al texto.](#)

En esta ventana seleccionamos **Web Service URL** y colocamos "http://localhost/DameCotizacion/DameCotizacion.asmx" en el cuadro de texto **URL**, y apretamos el botón **Search**. Esta acción deberá traer como resultado nuestro Web Service DameCotizacion en la sección **Search Results**, el cual seleccionaremos, donde podrá verse que está disponible nuestro método **GetCotizacion()**. Cliquharemos entonces **Add**.

Paso 9: El código del evento **Cotizar_Click()** es el siguiente:

```
Private Sub Cotizar_Click()
    Dim clsCotizacion As clsws_DameCotizacion
    Dim monedas As Range
    Dim moneda As Range
    Dim cotizacion As String
    clsCotizacion = New clsws_DameCotizacion
    monedas = Range(Range("b2"), Range("b65536").End(xlUp))
    Application.ActiveSheet.Range("b2").Activate()
    For Each moneda In monedas
        cotizacion = clsCotizacion.wsm_GetCotizacion(moneda)
        moneda.Offset(0, 1).Value = Val(cotizacion)
    Next moneda
End Sub
```

Si todo sale como es de esperar, el resultado de oprimir el botón **Cotizar** deberá ser el que se muestra en la [Figura 8](#):



Artículo	Moneda	Cotización	Valor	Pesos Uruguayos
Lapiz	EU	33	15	495
Goma	DO	30	30	900
Cuaderno	RE	9.9	10	99

Figura 8. [Volver al texto.](#)

Como verás, este es un simple ejemplo que muestra cómo consumir un Web Service desde una aplicación cliente, en este caso Microsoft Excel, que ilustra dos puntos interesantes: la facilidad de implementación del mismo y la potencia que nos brinda. Basta conocer un proveedor de un servicio de cotizaciones de moneda para mantener nuestra planilla al día con las últimas cotizaciones del mercado bursátil.

Para aquellos desarrolladores que ya hacían uso de incluir referencias a objetos COM en sus herramientas quizás esto no sea muy novedoso, pero en el caso de los objetos COM, los mismos debían estar físicamente en la computadora cliente. En el caso de los Web Services, estamos hablando de compartir recursos que habiten en la Intranet corporativa o más aún, en Internet y en sitios bien dispersos en el mundo.

Para los que quieran hacer números y le quieran sacar un beneficio económico, ¿Qué ocurriría si tú fueras un proveedor de Web Services y solicitaras la suscripción para el uso de los mismos a tus clientes a lo largo y ancho del planeta? Interesante, ¿No es así?

Más aún, no necesariamente el escenario se limita a una aplicación cliente consumiendo un Web Service sino que a su vez un Web Service podría consumir otro Web Service para poder armar la información de respuesta a retornar al cliente. No hace falta imaginar un escenario de este tipo pues esto ya es posible.

[Principio de la página](#)

Hacia dónde vamos

Si bien se ha avanzado mucho al respecto y hay infinidad de desarrolladores trabajando en este tema, existen aspectos a mejorar para catapultar aún más esta funcionalidad. Algunas características a mejorar pasan por temas relacionados con la seguridad (autorización, autenticación y cifrado) en el intercambio de mensajes, manejar el modelo transaccional y poder confirmar la entrega efectiva de los mensajes que se intercambian a través de los Web Services.

Adicionalmente, se continúa trabajando en la estandarización de los principales actores, como ser el WDSL y SOAP. Muchos fabricantes seguirán contribuyendo, elaborando herramientas para facilitar el manejo y elaboración de Web Services como es el caso de Microsoft y su Web Services Toolkit para Office 2003, que actualmente esta en su versión 2.01.

Otros elementos claves que no entran en análisis de este artículo pero igual los menciono por si es de interés del lector ahondar en los mismos, son los relacionados a las especificaciones de WS-Security, WS-Routing y DIME, para lo cual puedes encontrar mas información en la herramienta Microsoft WSDK Technology Preview o en Internet.

[Principio de la página](#)

Referencias

Desarrollo y consumo de un Web Services con Microsoft Visual Studio .Net

<http://www.desarrolloweb.com/articulos/1718.php?manual=54>

Nicolás Tedeschi es Analista de Sistemas de la Facultad de Ingeniería del Uruguay y se encuentra desarrollando tecnología .net y SQL Server. Desde 1995 trabaja con tecnologías



Microsoft de tres capas (HTML-ASP, COM+ y SQL Server). Es además Desarrollador 5 Estrellas de Microsoft MSDN y ha obtenido 3 Estrellas de ese programa.

[Principio de la página](#)

© 2019 Microsoft