Servicios web

Índice de contenido

Un poco de culturilla

La alta conectividad entre ordenadores ha sido una meta desde que comenzó la informática personal. Con el auge de las redes internas dentro de las empresas llega el deseo de unir máquinas de forma programática. Es decir, un programa en una máquina debería poder llamar a métodos del programa en otra máquina sin la necesidad de intervención humana, para lo que se utilizaron diferentes tecnologías como el Modelo de Objeto de Componente Distribuido (DCOM). El siguiente paso al conectar ordenadores es hacerlo a través de Internet. Mediante ordenadores conectados a través de HTTP y el formato XML se crearon los servicios Web XML.

Los pasos fundamentales del Remoting son:

1. El llamador prensa la pila de llamadas del método local en una cadena que se envía por una conexión. Esto se conoce como **serialización**.
2. El llamador envía la pila de llamadas serializada a través de la conexión
3. El extremo recibe la pila de llamadas serializada y la convierte en una pila de llamadas utilizable en el servidor. Se llama **deserialización**.
4. El extremo procesa la llamada al método
5. El extremo transmite los resultados de vuelta al llamador.

El formato que los llamadores y servicios acuerdan se llamó originalmente Simple Object Access Protocol. Hoy sólo se utilizan sus siglas **SOAP**. El protocolo SOAP es una formalización XML para comunicación basada en mensajes.

ASP.NET manipula los detalles de realizar un servicio Web a través de la clase **System.Web.Services.WebService**.

Los servicios Web ASP.NET viven en un tipo de archivo nombrado con la extensión .**asmx**.

Crear un servicio Web

Para crear un nuevo servicio web creamos un Nuevo proyecto de sitio Web y elegimos servicio Web ASP.NET. (*Ilustración 1*).

En el explorador de soluciones tendremos una estructura como la que se muestra en la Ilustración 2. Renombramos el fichero Service.cs dentro de App_Code a un nombre más significativo.

Haremos lo mismo con Service.asmx

Dentro del código de Service.cs cambiaremos también el nombre de la clase con la utilidad de refactorización de Visual Studio. Para esto seleccionamos Service y pinchando con el botón derecho elegimos refactorizar → Cambiar nombre.

La referencia a la clase en el fichero asmx hay que cambiarla de forma manual.

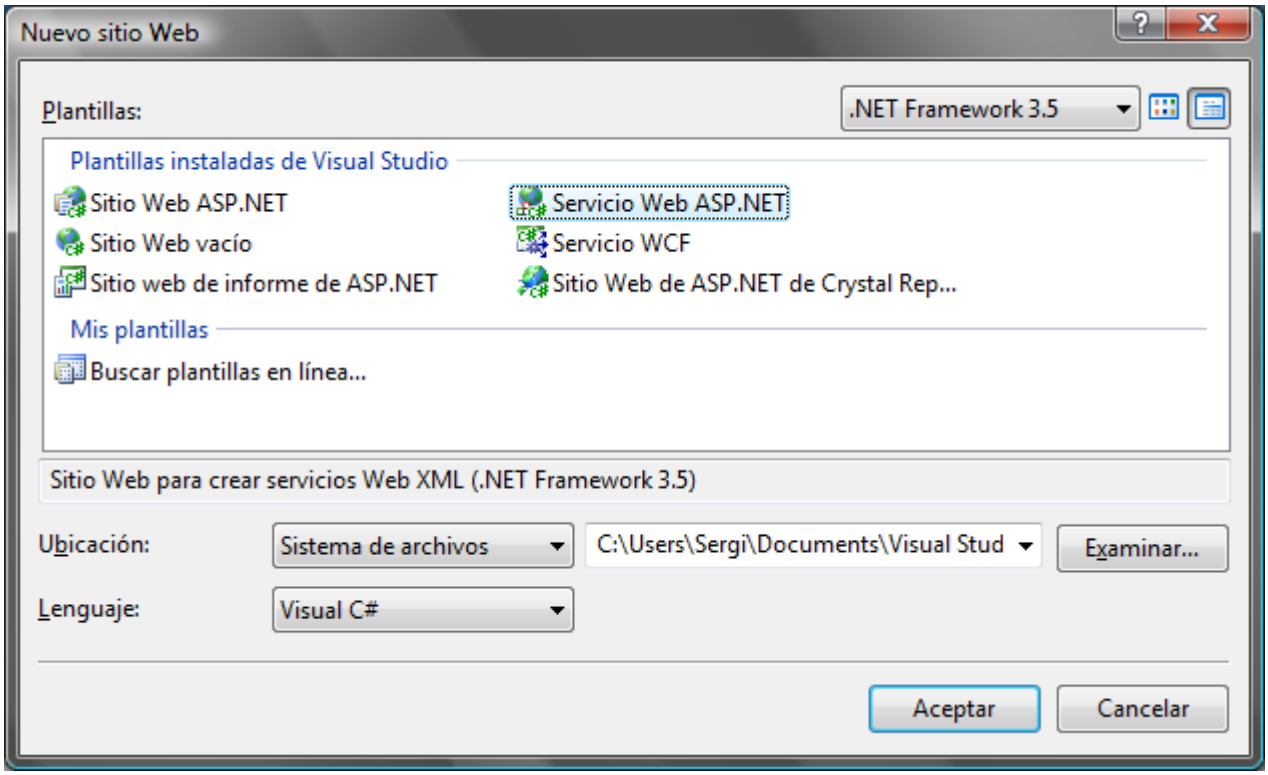


Ilustración 1: Nuevo servicio web

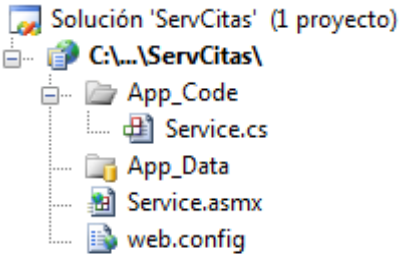


Ilustración 2:

En el ejemplo se ha cambiado Service por ServCitas.

El fichero ServCitas.asmx queda de la siguiente manera:

```
<%@ WebService Language="C#" CodeBehind="~/App_Code/ServCitas.cs"
Class="ServCitas" %>
```

Y el servicio web llamado ahora ServCitas.cs así:

```
using System;
using System.Linq;
using System.Web;
using System.Web.Services;
using System.Web.Services.Protocols;
using System.Xml.Linq;

[WebService(Namespace = "http://tempuri.org/")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
// Para permitir que se llame a este servicio web desde un script, usando
// ASP.NET AJAX, quite la marca de comentario de la línea siguiente.
// [System.Web.Script.Services.ScriptService]
public class ServCitas : System.Web.Services.WebService
{
    public ServCitas () {

        //Eliminar la marca de comentario de la línea siguiente si utiliza los
        //componentes diseñados
        //InitializeComponent();
    }

    [WebMethod]
    public string HelloWorld() {
        return "Hello World";
    }
}
```

Y el explorador de soluciones como muestra la ilustración 3.

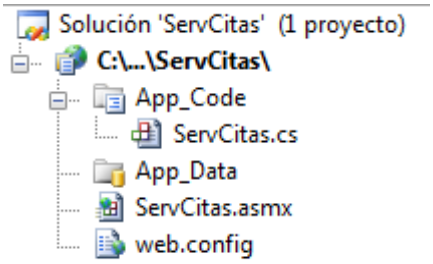
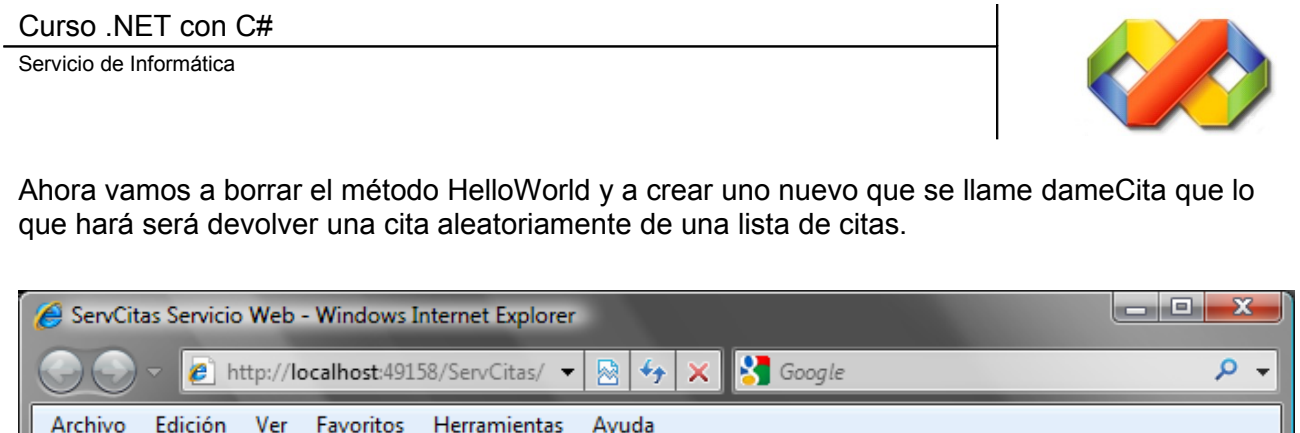


Ilustración 3:

Si navegamos ahora al archivo ServCitas.asmx para ver lo que nos muestra un HTTP GET predeterminado obtendríamos lo siguiente (*Ilustración 4*) .

ASP.NET nos muestra los nombres de los metodos disponibles cuando se hace un GET hacia el archivo ASMX. Podemos ver el método HelloWorld (creado por Visual Studio). Si queremos probar a ejecutar el método sólo tenemos que pinchar en HelloWorld. (*Ilustración 5*)

Si en la primera página pinchamos en el vínculo **descripción de servicios** veremos el WSDL, que es un chorizo de XML para fliparlo, pero no está hecho para que lo leamos nosotros, sino un cliente proxy.



Ahora vamos a borrar el método HelloWorld y a crear uno nuevo que se llame dameCita que lo que hará será devolver una cita aleatoriamente de una lista de citas.

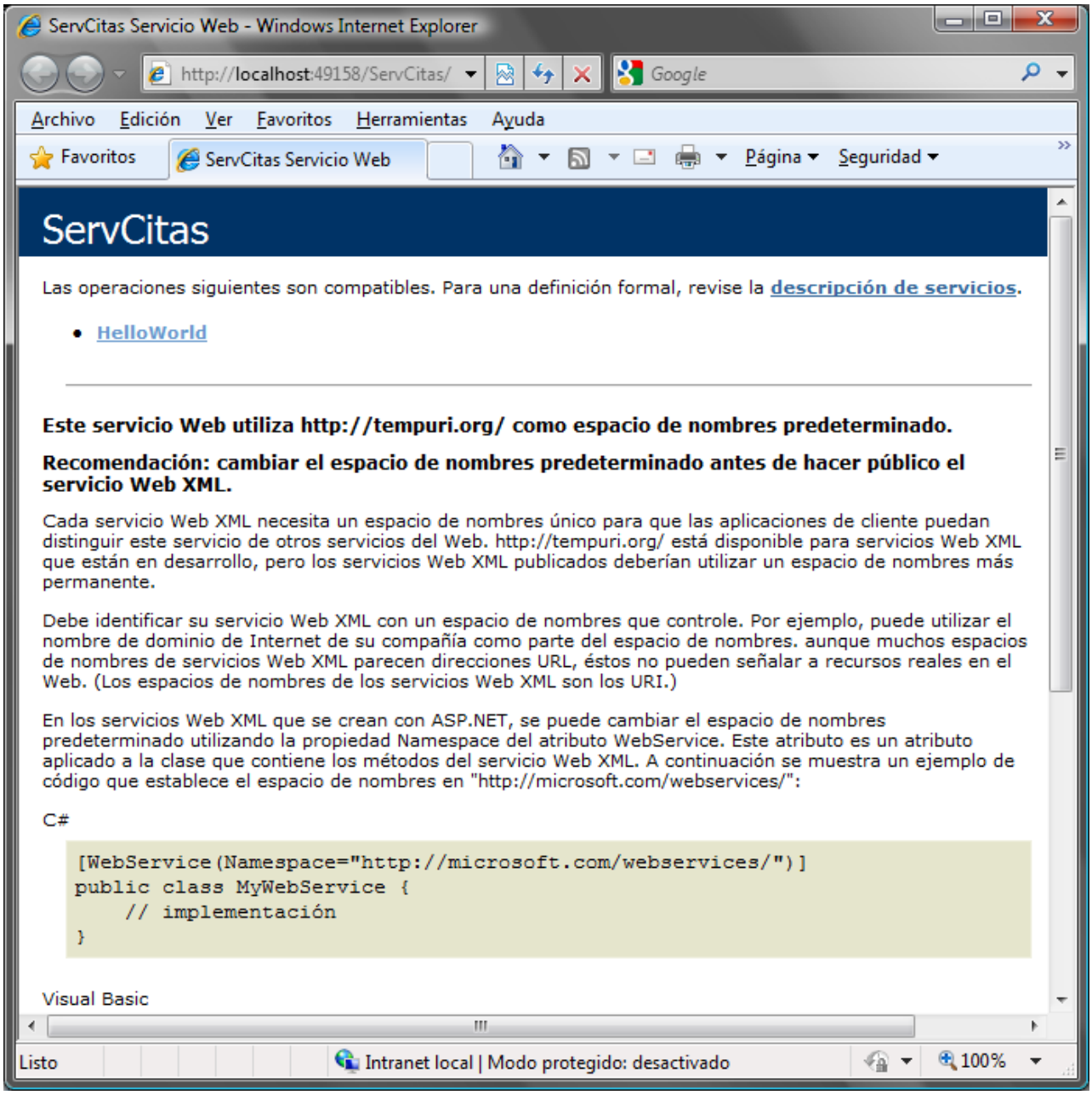


Ilustración 4: Navegar a ServCitas.asmx

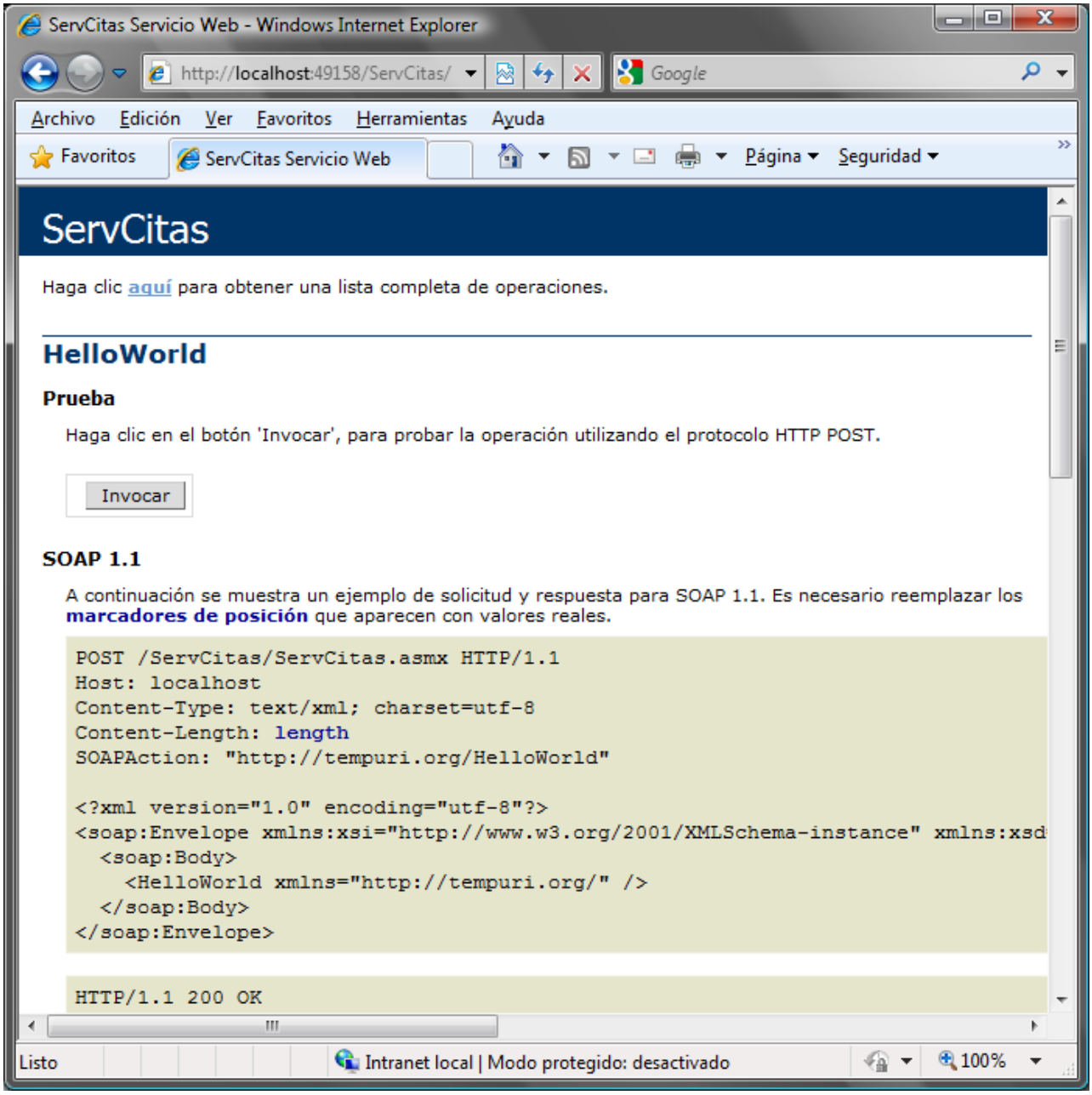


Ilustración 5: Ejecutar HelloWorld

El código del metodo dameCita sería:

```
[WebMethod]
public String dameCita() {
    String [] citas = {
       "No hay mal que por bien no venga",
       "Por un perro que maté, mata perros me llamaron",
       "No por mucho madrugar amanece mas temprano"};
    Random random = new Random();
    int numCita = random.Next(0,3);
    return (citas[numCita]);
}
```

Si lo ejecutamos podremos ver que obtenemos una respuesta como la siguiente:

```
<?xml version="1.0" encoding="utf-8" ?>
<string xmlns="http://tempuri.org/">No hay mal que por bien no
venga</string>
```

Variando la cita cada vez que la ejecutemos.

Este es un método simple pero sirve de ejemplo para ver como funcionaría un servicio que sacara datos de una base de datos.

Ahora creamos un nuevo método llamado citaPropia que recibe un parámetro y lo devuelve concatenado a una cadena.

```
[WebMethod]
public String citaPropia(String laCita)
{
    return ("Esta es tu cita: " + laCita);
}
```

El funcionamiento es igual que el anterior pero si lo ejecutamos nos pedirá el valor del parámetro. (*Ilustración 6*)

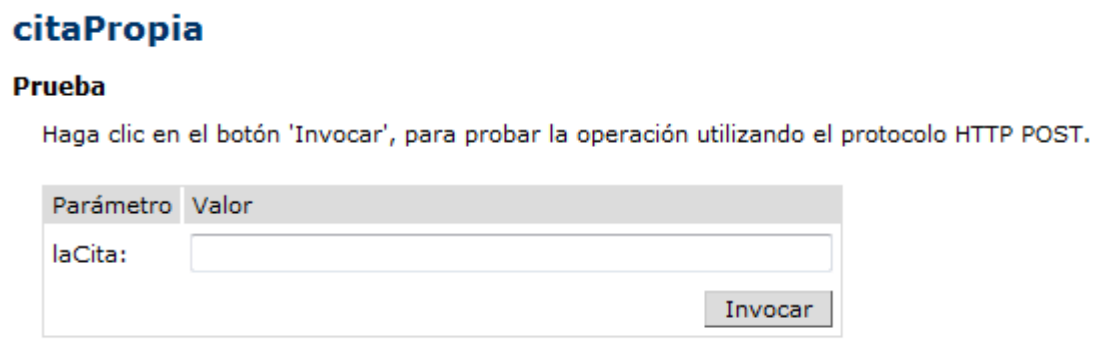


Ilustración 6: Servicio con parámetros

Usar servicios web

Usar servicios web desde Visual Studio es casi tan fácil como crearlos. No hay que olvidar que los servicios web XML son una plataforma independiente, y las plataformas informáticas más modernas soportan el uso de servicios web XML.

Vamos a crear un sitio web que utilice el servicio antes creado.

Agregamos un nuevo sitio web a la solución anterior pulsando con el botón derecho en la solución dentro del Explorador de Soluciones y eligiendo Agregar → Nuevo Sitio Web.

Una vez lo tenemos creamos una referencia de servicio pulsando con el botón derecho en el proyecto dentro del Explorador de Soluciones y eligiendo **Agregar referencia de servicio**. Nos aparecerá un cuadro de diálogo como el que se muestra a continuación. (*Ilustración 7*).

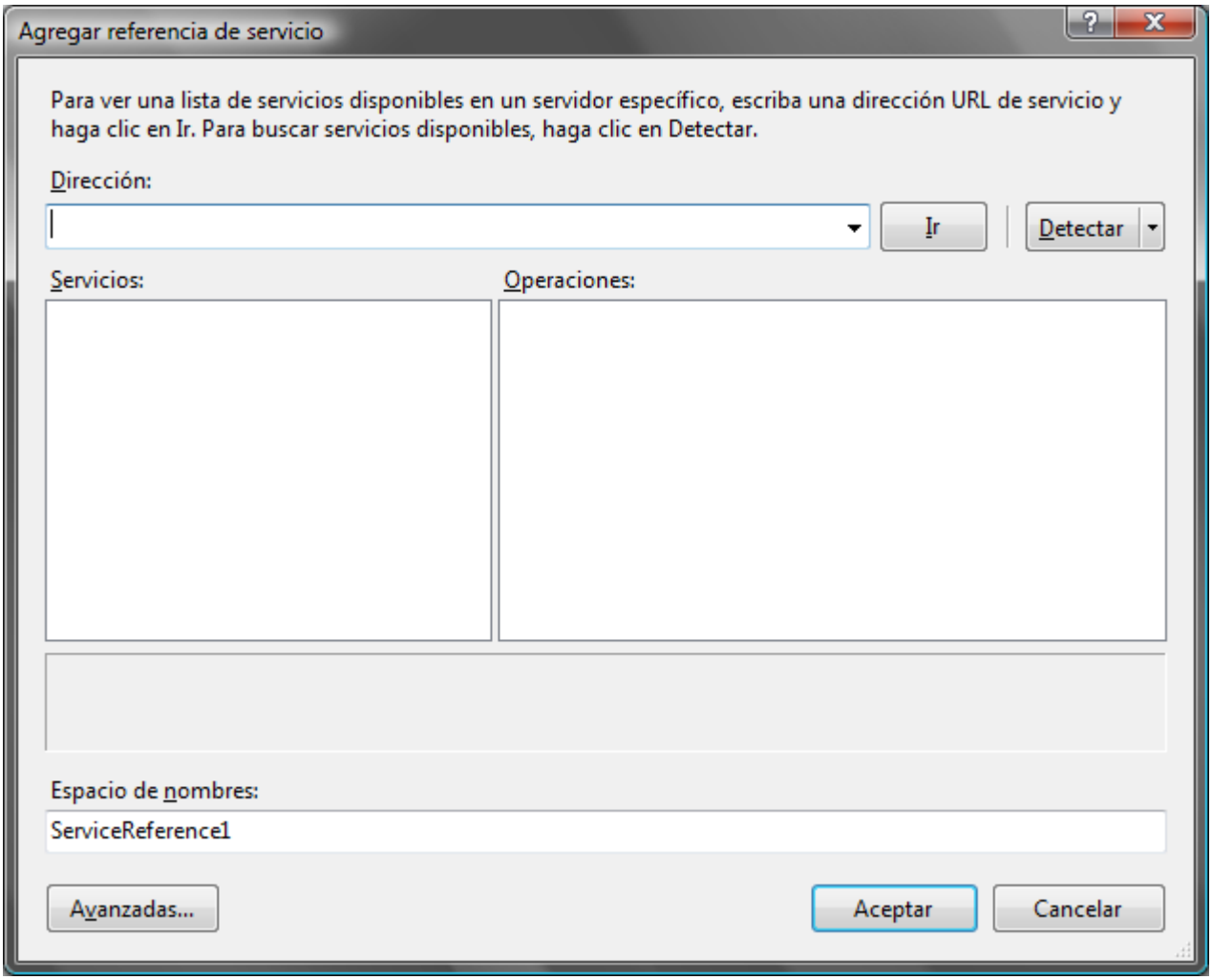


Ilustración 7: Agregar referencia de servicio

Pulsamos en el botón Detectar y obtenemos una lista con los servicios disponibles. Expandimos el árbol pulsando en el más. Podremos ver los métodos creados anteriormente en el cuadro Operaciones.(*Ilustración 8*)

También cambiaremos el espacio de nombres a uno mas significativo.

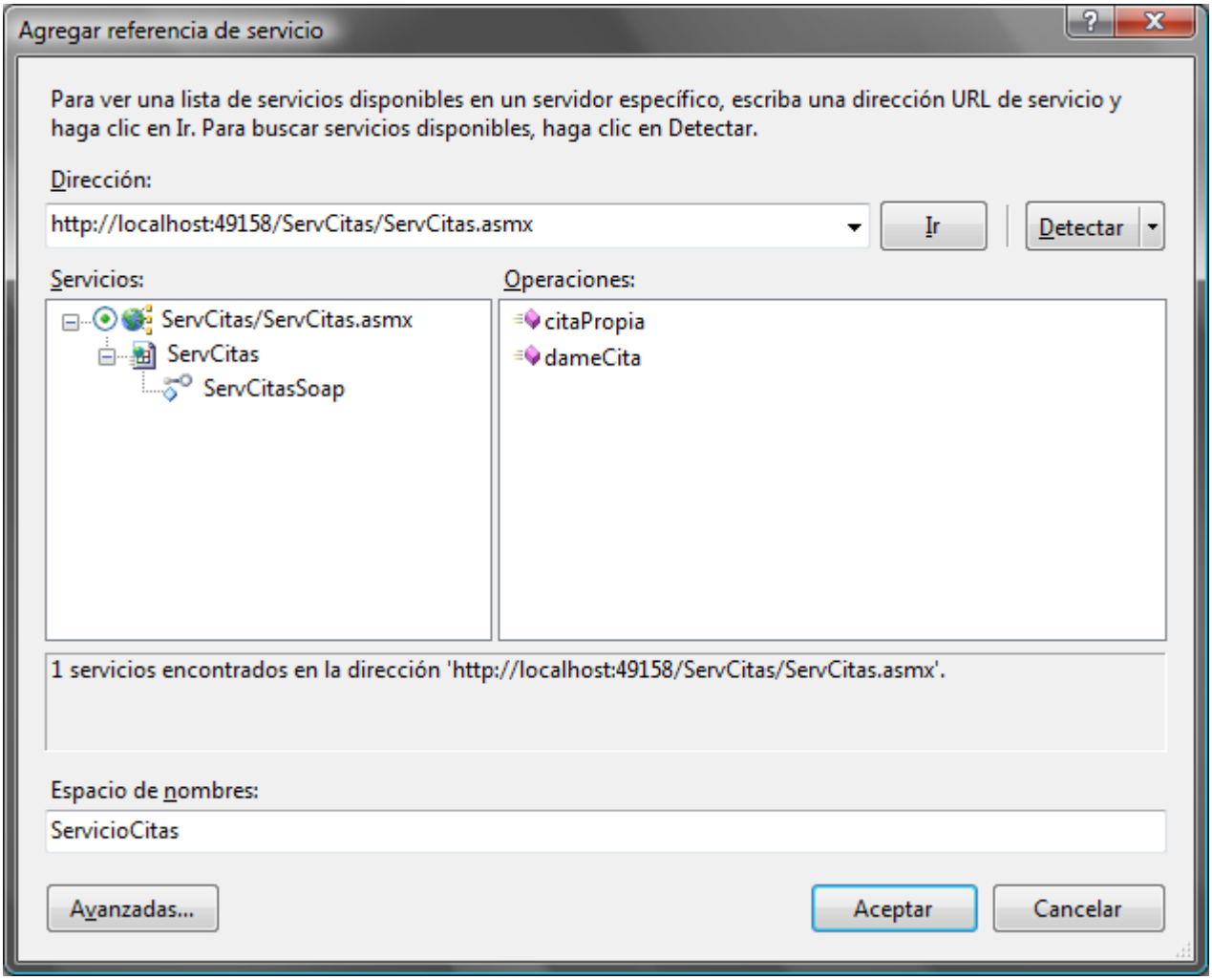


Ilustración 8: Servicios disponibles

Si pinchamos en el botón Avanzadas... podremos especificar más opciones, como la de Generar operaciones asíncronas. Marcamos esta casilla (*Ilustración 9)*.

Ahora ya podemos llamar al servicio web a través del proxy. El nombre del proxy ServicioCitas es ServicioCitasSoapClient. Lo instanciamos como si fuese una clase más.

Cuando llamamos a métodos, el proxy empaquetará la llamada en un envoltorio SOAP y enviará la petición al destino especificado dentro del proxy.

Vamos a imprimir en nuestra web las citas devueltas por el servicio web.

Incluimos el espacio de nombres antes creado con:

```
using ServicioCitas;
```

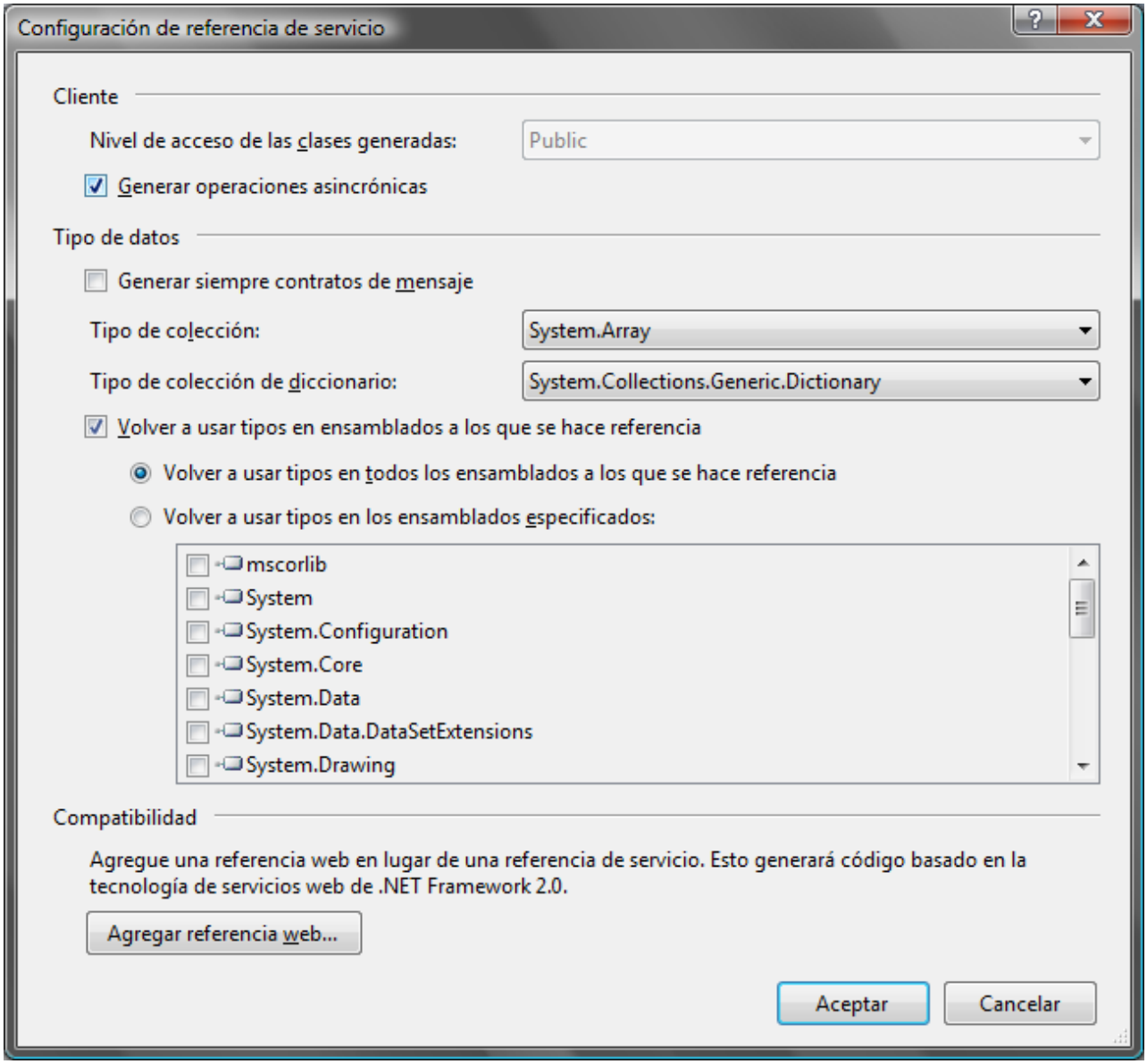


Ilustración 9: Configuración de referencia de servicio

El código de nuestro sitio web es:

```
using System;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;
using ServicioCitas;
```

```
public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
     ServCitasSoapClient servicioCitas = new ServCitasSoapClient();
     String citaDeHoy = servicioCitas.dameCita();
     String miCita = servicioCitas.citaPropia("Al pan pan y al vino vino");

     Response.Write("<br />La cita automática de hoy es: '" + citaDeHoy + "'");
     Response.Write("<br />Mi cita es: '" + miCita + "'");
    }
}
```

Si visualizamos en el navegador obtendremos lo siguiente (*Ilustración 10*).

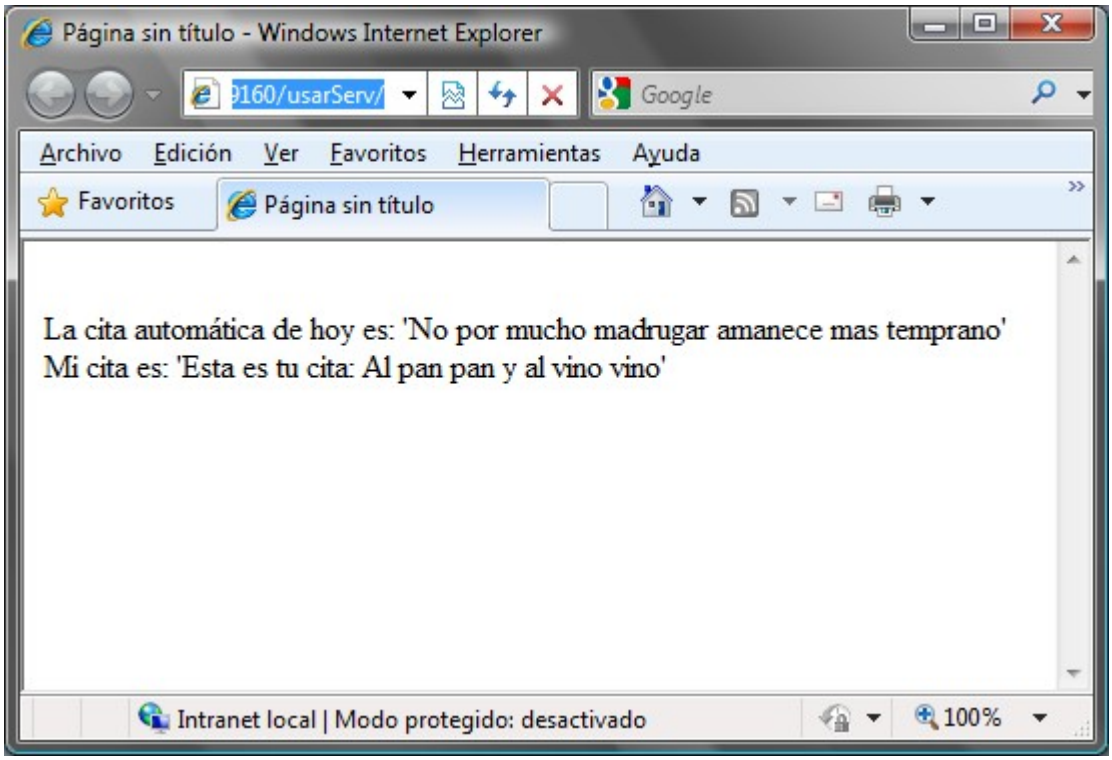


Ilustración 10: Uso de servicio web

No hay que olvidar que este es un ejemplo muy sencillo que sólo devuelve y recibe cadenas de texto. Pero se puede extender a cualquier tipo de objeto

C'est finit !!