Web Services

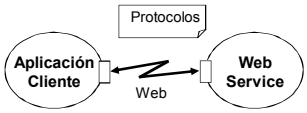
Prof. Javier Echaiz

SOSD
Departamento de Ciencias e Ingeniería de la Computación
je@cs.uns.edu.ar
http://cs.uns.edu.ar/~jechaiz

Definición de Web Service

Una definición simple:

"un Web Service es un programa que es llamado desde otro programa a través de la web empleando protocolos abiertos"



Historia

» Los Web Services son la evolución de tecnologías como RPC, ORPC (DCOM, CORBA, y JAVA RMI).
» Los Web Services se originaron para resolver tres problemas principales:
1. Interoperatividad.
2. Atravesar firewalls.
3. Complejidad.

Interoperatividad

» Los primeros sistemas distribuidos tenían problemas de interoperatividad, cada proveedor implementaba sus propios formatos para el envío de mensajes.
» Aplicaciones DCOM ligadas a Windows.
» Aplicaciones RMI ligadas a Java.

Atravesar Firewalls

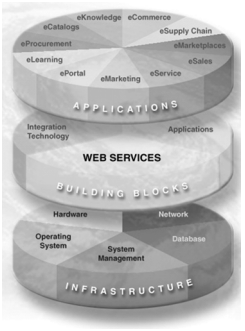
» Proyectos de cooperación entre corporaciones: difícil. CORBA vs. DCOM.
» Los Web Services emplean HTTP como protocolo de transporte y la mayoría de los firewalls permite acceso a través del port 80 ⇨ facilitando la colaboración.

Complejidad

» Las tecnologías para Web Services son "amigables" a los desarrolladores.
» La mayoría de las tecnologías antes mencionadas (RMI, COM, CORBA) involucran una curva completa de aprendizaje.
» Deben aprenderse nuevas tecnologías y lenguajes para implementar estos servicios.

Definición de Web Services (rev.)

- Una definición más precisa:
  - Una aplicación que:
    - Se comunica con otra mediante protocolos abiertos (HTTP, SMTP, etc.)
    - Procesa mensajes XML encapsulados mediante SOAP.
    - Describe sus mensajes empleando XML Schema.
    - Provee una descripción usando WSDL.
    - Se descubre mediante UDDI.

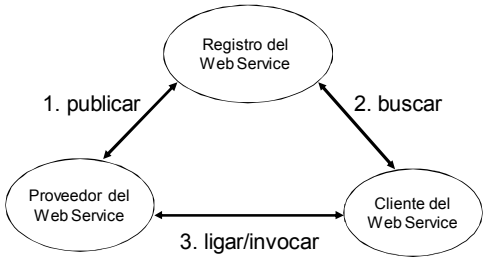


El modelo Web Service

- La arquitectura de Web Services se basa en estos tres componentes:
  - Registro del servicio.
  - Proveedor del servicio.
  - Solicitante del servicio.
- La interacción entre estos componentes involucra:
  - Operaciones de publicación.
  - Operación de búsqueda.
  - Operaciones de ligado (*binding*)/invocación.

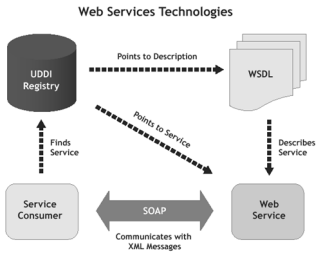
El modelo Web Service (cont.)

El modelo Web Service sigue el paradigma de *publicar*, *buscar*, y *ligar*.



Componentes de Web Services

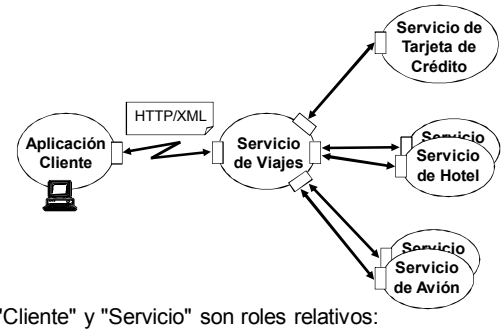
- **XML** – eXtensible Markup Language – Un mecanismo uniforme de representación e intercambio de datos.
- **SOAP** – Simple Object Access Protocol – Un estándar de comunicación.
- **UDDI** – Universal Description, Discovery and Integration specification – Un mecanismo para registrar y localizar una aplicación WS.
- **WSDL** – Web Services Description Language – Un metalenguaje estándar para describir los servicios ofrecidos.



Ejemplo – Un Web Service simple

- Un comprador (cliente) que ordena mercadería/servicio a un servicio vendedor.
- El comprador encuentra el servicio vendedor buscando en el directorio UDDI.
- El servicio vendedor es un Web Service cuya interfaz se define mediante Web Services Description Language (WSDL).
- El comprador invoca el método de orden de compra del servicio del vendedor mediante Simple Object Access Protocol (SOAP) y la definición WSDL para el servicio vendedor.
- El vendedor sabe que esperar en el mensaje de respuesta SOAP, pues fue definido mediante WSDL.

Ejemplo Servicio de Viajes



"Cliente" y "Servicio" son roles relativos:
Servicio podría ser Cliente de otros Web Services.

Conclusión

¿Por qué los Web Services tienen gran potencial?

» Basados en estándares que tienen amplio apoyo de la industria.
» Emplean tecnologías simples y probadas, e.g. HTTP y XML.
» Logran interoperatividad entre sistemas que difieren en software y hardware.

Gracias...

jechaiz@cs.uns.edu.ar

Backup Slides

Links

» http://msdn.microsoft.com/webservices/understanding/webservicebasics/default.aspx
» http://www.w3schools.com
» http://www.w3c.org/TR/soap
» http://www.w3c.org/TR/wsdl
» http://www.uddi.org
» http://www.developer.com/services/article.php/2195981
» http://www.xmethods.com
» …google!

Web Services Def. (W3C)

» *A Web service is a software system*
  » *Identified by a URL, whose public interfaces and bindings are defined and described using XML.*
» *Its definition can be discovered by other software systems*
» *These systems may then interact with the Web service*
  » *using XML based messages conveyed by Internet protocols*

XML

» XML stands for E**X**tensible **M**arkup **L**anguage.
» XML is a **markup language** much like HTML.
» XML was designed to **describe data**.
» XML tags are not predefined. You must **define your own tags.**
» The prefect choice for enabling cross-platform data communication in Web Services.

XML vs HTML

An HTML example:

```
<html>
<body>
   <h2>John Doe</h2>
   <p>2 Backroads Lane<br>
        New York<br>
        045935435<br>
        john.doe@gmail.com<br>
        </p>
</body>
</html>
```


XML vs HTML

» This will be displayed as:

John Doe

2 Backroads Lane
New York
045935435
John.doe@gmail.com

» HTML specifies how the document is to be displayed, and not what information is contained in the document.
» Hard for machine to extract the embedded information. Relatively easy for human.

XML vs HTML

» Now look at the following:

```
<?xml version=1.0?>
<contact>
  <name>John Doe</name>
  <address>2 Backroads Lane</address>
  <country>New York</country>
  <phone>045935435</phone>
  <email>john.doe@gmail.com</email>
</contact>
```

» In this case:
  » The information contained is being marked, but not for displaying.
  » Readable by both human and machines.

SOAP

» SOAP originally stood for "Simple Object Access Protocol" .
» Web Services expose useful functionality to Web users through a standard Web protocol called SOAP.
» Soap is an XML vocabulary standard to enable programs on separate computers to interact across any network. SOAP is a simple markup language for describing messages between applications.
» Soap uses mainly HTTP as a transport protocol. That is, HTTP message contains a SOAP message as its payload section.

SOAP Characteristics

» SOAP has three major characteristics:
  » Extensibility – security and WS-routing are among the extensions under development.
  » Neutrality - SOAP can be used over any transport protocol such as HTTP, SMTP or even TCP.
  » Independent - SOAP allows for any programming model .

SOAP Building Blocks

A SOAP message is an ordinary XML document containing the following elements:
  » A required Envelope element that identifies the XML document as a SOAP message.
  » An optional Header element that contains header information.
  » A required Body element that contains call and response information.
  » An optional Fault element that provides information about errors that occurred while processing the message.

SOAP Request

```
POST /InStock HTTP/1.1
Host: www.stock.org
Content-Type: application/soap+xml; charset=utf-8 Content-Length: 150

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle=http://www.w3.org/2001/12/soap-encoding">

    <soap:Body xmlns:m="http://www.stock.org/stock">
        <m:GetStockPrice>
            <m:StockName>IBM</m:StockName>
        </m:GetStockPrice>
    </soap:Body>
</soap:Envelope>
```


SOAP Response

```
HTTP/1.1 200 OK
Content-Type: application/soap; charset=utf-8
Content-Length: 126

<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-
envelope" soap:encodingStyle="http://www.w3.org/2001/12/soap-
encoding">

    <soap:Body xmlns:m="http://www.stock.org/stock">
        <m:GetStockPriceResponse>
            <m:Price>34.5</m:Price>
        </m:GetStockPriceResponse>
    </soap:Body>
</soap:Envelope>
```


SOAP Security

- SOAP uses HTTP as a transport protocol and hence can use HTTP security mainly HTTP over SSL.
- But, since SOAP can run over a number of application protocols (such as SMTP) security had to be considered.
- The WS-Security specification defines a complete encryption system.

WSDL

- WSDL stands for Web Services Description Language.
- WSDL is an XML vocabulary for describing Web services. It allows developers to describe Web Services and their capabilities, in a standard manner.
- WSDL specifies what a request message must contain and what the response message will look like in unambiguous notation. In other words, it is a contract between the XML Web service and the client who wishes to use this service.
- In addition to describing message contents, WSDL defines where the service is available and what communications protocol is used to talk to the service.

The WSDL Document Structure

- A WSDL document is just a simple XML document.
- It defines a web service using these major elements:
  - **port type** - The operations performed by the web service.
  - **message -** The messages used by the web service.
  - **types -** The data types used by the web service.
  - **binding -** The communication protocols used by the web service.

WSDL Document

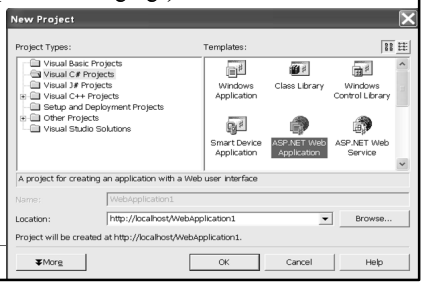
```
<message name="GetStockPriceRequest">
    <part name="stock" type="xs:string"/>
</message>
<message name="GetStockPriceResponse">
    <part name="value" type="xs:string"/>
</message>

<portType name="StocksRates">
    <operation name="GetStockPrice">
        <input message="GetStockPriceRequest"/>
        <output message="GetStockPriceResponse"/>
    </operation>
</portType>
```

UDDI

- UDDI stands for Universal Description, Discovery and Integration.
- UDDI is a directory for storing information about web services , like yellow pages.
- UDDI is a directory of web service interfaces described by WSDL.

Step by Step – using a web service

1. Inside Visual Studio .NET Choose File > New > Project.
2. Choose Visual C# Projects (or Visual Basic Projects if you prefer this language).
3. Choose ASP.NET Web Application as your template



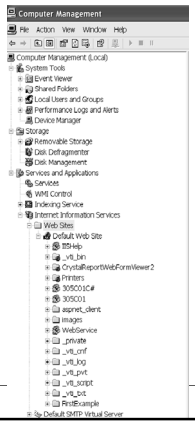
Step by Step – using a web service

- Inside the Location text box enter the name of your project after the prefix : http://localhost/YourProjectName
- Press OK.
- This makes The Internet Information Services installed on your computer create a new directory on the default path: C:\Inetpub\wwwroot\FirstExample

Step by Step – using a web service

- You can open IIS by typing compmgmt.msc /s in the run command and then choosing Services And Application > Internet Information Services.
- Inside this node you can choose Web Sites node and then Default Web Site to see all the web sites on your computer.

Step by Step – using a web service



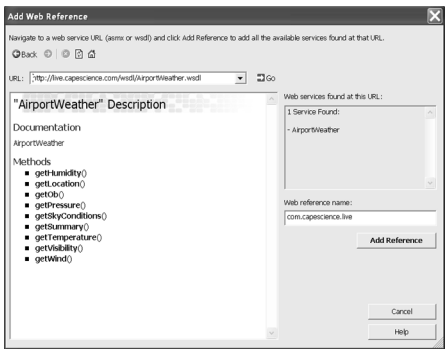
Step by Step – using a web service

- In the new project you opened in VS.NET Move to the Solution Explorer.
- Right Click on the References folder and Choose Add Web References.
- This Opens the Add Web Reference Dialog Box.

Step by Step – using a web service

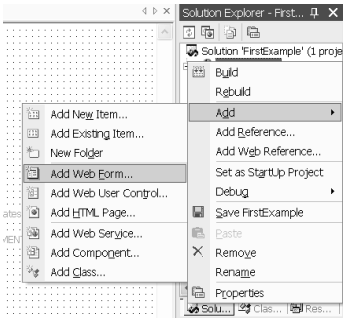
» Type the Web Service URL and Press Go.
» It takes a couple of seconds to find the Web services and finally all it's methods appear in the list box.
» The Web Reference name is shown in the Dialog Box.
» Press Add Reference to complete the process.

Step by Step – using a web service



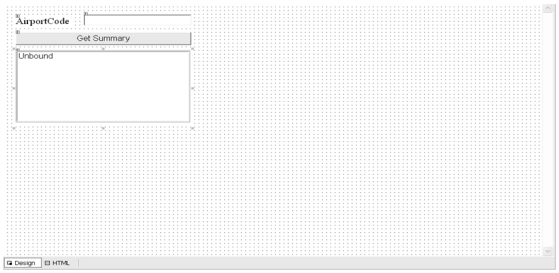
Step by Step – using a web service

» Add a new Web Form.



Step by Step – using a web service

» Add the following Controls to the Web Form



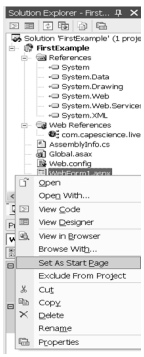
Step by Step – using a web service

» Double Click on the button and insert this code to it's OnClick event handler.

```
private void btnGetSummary_Click(object sender, System.EventArgs e)
{
    com.capescience.live.AirportWeather aw = new
        com.capescience.live.AirportWeather();
    com.capescience.live.WeatherSummary ws = aw.getSummary(txtCode.Text);

    lbResults.Items.Clear();
    lbResults.Items.Add(ws.location);
    lbResults.Items.Add("Temprature: " + ws.temp);
    lbResults.Items.Add("Visibility: " + ws.visibility);
    lbResults.Items.Add("Wind: " + ws.wind);
}
```


Step by Step – using a web service

1. Set the Web Form as the Start Page.
2. Build and Run the Program.
3. Try to use the Web Application.

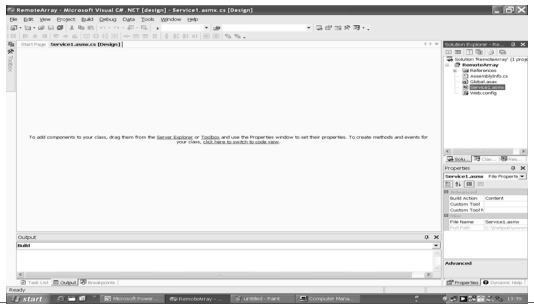


Step By Step – Creating a Web Service

» In this Step I will create a new Web Service and write a Simple Program that uses it.
» The program will perform various operations on an array.
» The client program will be a simple dialog box that activates those opeartions.

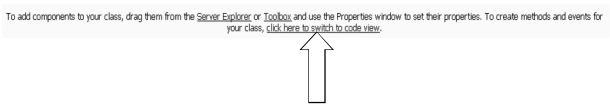
Step By Step – Creating a Web Service

» Create a new Visual C# project with the name RemoteArray. The following screen appears.



Step By Step – Creating a Web Service

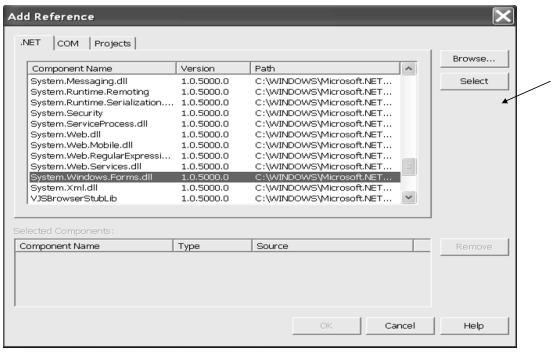
» To see the code Press on the following hyperlink.



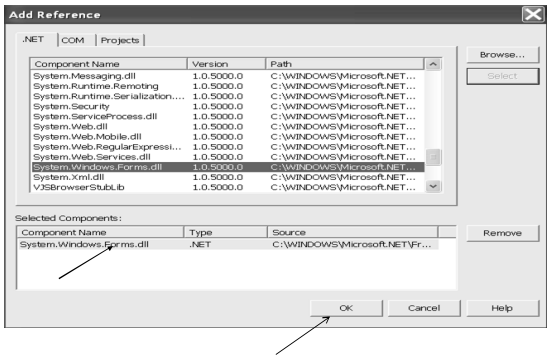
Step By Step – Creating a Web Service

» Right Click on the References folder and choose add Reference.
» Insert the System.Windows.Forms.dll option in to this folder.

Step By Step – Creating a Web Service



Step By Step – Creating a Web Service



Step By Step – Creating a Web Service

» Insert the following code to the .asmx file you've created.

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Web;
using System.Web.Services;
using System.Windows.Forms;

namespace RemoteArray
{
    /// <summary>
    /// Summary description for Service1.
    /// </summary>
    [WebService( Namespace = "http://localhost/RemoteArray",
          Description = "A simple web service that sorts, reverses, binary search and"
              +" displays an array of ints." )]
    public class RemoteArray : System.Web.Services.WebService
    {

        public RemoteArray()
        {
            //CODEGEN: This call is required by the ASP.NET Web Services Designer
            InitializeComponent();
        }
```


Step By Step – Creating a Web Service

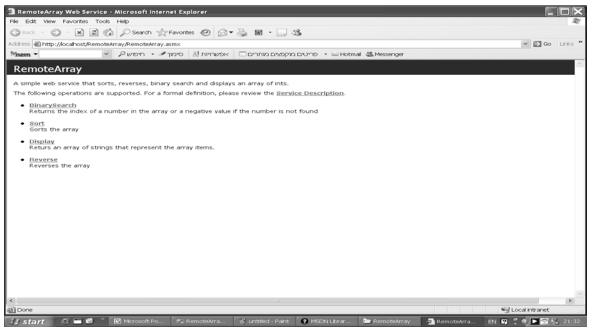
```
        Component Designer generated code

        [WebMethod ( Description = "Sorts the array")]
        public int[] Sort(int [] array)
        {
            Array.Sort(array,0,array.Length);
            return array;
        }

        [WebMethod ( Description = "Reverses the array" )]
        public int []  Reverse(int [] array)
        {
            Array.Reverse(array,0,array.Length);
            return array;
        }

        [WebMethod ( Description = "Returns the index of a number in the array or a negative"
             +" value if the number is not found")]
        public int BinarySearch (int [] array,int whatToSearch)
        {
            return Array.BinarySearch(array,0,array.Length,whatToSearch);
        }
```


Step By Step – Creating a Web Service

```
        [WebMethod ( Description = "Returns an array of strings that represent the array items." )]
        public string[] Display(int [] array)
        {
            string []output = new string[array.Length];

            for (int i = 0 ; i < array.Length ; i++)
                output[i] = "array[" + i + "] = " + array[i].ToString();

            return output;
        }
    }
}
```

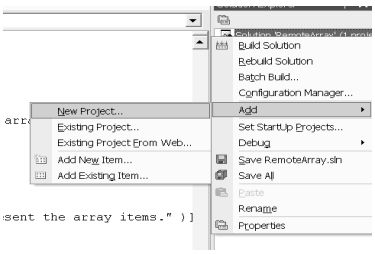

Step By Step – Creating a Web Service

» Press Ctrl + F5 to Run the Web service.



Step By Step – Using Remote Array

» Add a new project to RemoteArray Solution



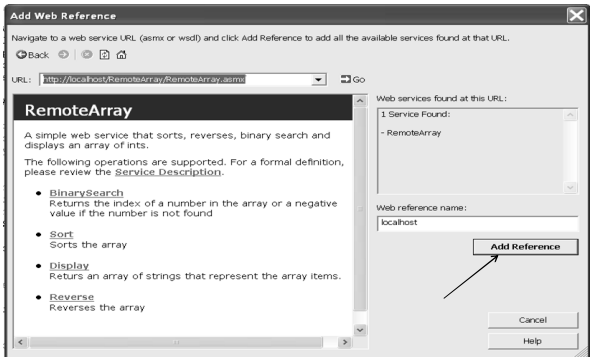
Step By Step – Using Remote Array

» Choose Windows Application from the templates.

» Add a web reference for the Remote Array Web Service.

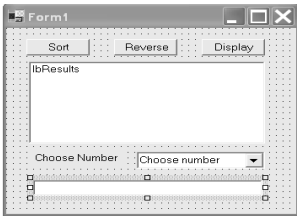
» Remember that it's inside an asmx file.

Step By Step – Using Remote Array



Step By Step – Using Remote Array

» Add the following elements to the Form



Step By Step – Using Remote Array

» Create a private RemoteArray object and a private int array object to the Form.

» Insert this code after the Initialize component part.

```
public Form1()
{
    InitializeComponent();

    ra = new localhost.RemoteArray();
    Random r = new Random();
    array = new int [20];
    for (int i = 0 ; i < 20 ; i++)
        array[i] = r.Next(20);
    for (int i = 0 ; i < 20 ; i++)
        cbNumber.Items.Add(i.ToString());
}
```

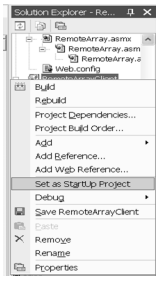

Step By Step – Using Remote Array

» Insert the following code to controls handler

```
private void btnSort_Click(object sender, System.EventArgs e)
{
    array = ra.Sort(array);
}

private void btnDisplay_Click(object sender, System.EventArgs e)
{
    string [] text = ra.Display(array);
    lbResults.Items.Clear();
    for (int i = 0 ; i < text.Length ; i++)
        lbResults.Items.Add(text[i]);
}

private void btnReverse_Click(object sender, System.EventArgs e)
{
    array = ra.Reverse(array);
}

private void cbNumber_SelectedIndexChanged(object sender, System.EventArgs e)
{
    int i = Int32.Parse(cbNumber.SelectedIndex.ToString());

    int t = ra.BinarySearch(array,i);

    if ( t >= 0 )
        txtResult.Text = "The item you requested is in index " + t.ToString();
    else
        txtResult.Text = "The number you chose doesn't exist";
}
```


Step By Step – Using Remote Array

» Set the Windows Application project as the Startup



Step By Step – Using Remote Array

» Compile and run the application.

» This is an example that an XML Web application can be used over Windows and not only with ASP.NET