WCF vs ASP.NET Web Services

**Messaging**

ASP.NET Web Services use SOAP (Simple Object Access Protocol) for sending and receiving data. XML Schema defines the structure of message.

Windows communication foundation can send message in any format. By default it use SOAP for communication. It use any transport protocol for message transporting.

**Data Representation**

ASP.NET Web Services use XMLSerializer to translate complex data to XML data for transporting messages. .NET framework classes can be serialized to and from XML. XMLSerializer is used for this. These classes can be written manually and used command line support utility xsd.exe for defining XML Schema.

Following point must be remembered when .NET framework class serialized to and from XML.

* Only public fields and properties of .NET framework can translated into XML.
* Instances of collection classes can be serialized into XML only if the classes implement either the IEnumerable or ICollection interface.
* Classes that implement the IDictionary interface, such as Hashtable, cannot be serialized into XML.
* The great many attribute types in the System.XML.Serialization namespace can be added to a .NET Framework class and its members to control how instances of the class are represented in XML.

The WCF can be made to use same .NET framework as ASP.NET do. The WCF DataContractAttribute and DataMemberAttribute can be added to .NET framework type to serialize into XML, and which particular field to be serialized either private or public.

[DataContract]

public class CompositeType

{

[DataMember]

private string stringValue="Demo";

int item1;

int item2;

[DataMember]

public int Item1

{

get { return item1; }

set { item1 = value; }

}

[DataMember]

public int Item2

{

get { return item2; }

set { item2 = value; }

}

}

The **DataContract attribute** signifies that class has types to be serialized. The **DataMember attribute** indicate which property or field need to be serialized, it can be private or public. These are serialized into XML using **DataContractSerializer.**

**Service Development**

To develop web service using ASP.NET we need to add WebService attribute and WebMethod attribute to expose methods.

[WebService]

[System.ComponentModel.ToolboxItem(false)]

public class WebService : System.Web.Services.WebService

{

[WebMethod]

public string HelloWorld()

{

return "Hello World";

}

}

ASP.NET 2.0 introduces the option of adding the attribute WebService and WebMethod attribute to an interface. This is useful because the interface with the **WebService attribute** constitutes a contract for the operations performed by the service that can be reused with various classes that might implement that same contract in different ways.

A WCF service is provided by defining one or more WCF endpoints. The endpoint is defined by an address, a binding and a contract. The address defines where service located. The way of communication is defined by binding. The operation of service is defined by service contract.

The **ServiceContract attribute** specifies that the interface defines a WCF service contract and **OperationContract attribute** define methods of service contract. A class implements a service contract is a service type in WCF.

Next step is binding which specifies set of protocol for communication with the application. Mode communication can be on HTTP, TCP or many more.

The internal behavior of service can be adjusted using the properties of the family if classes called behaviors. The ServiceBehaviorAttribute class is used to specify that the service type is to be multithreaded.

**Hosting**

ASP.NET complies into class of class library assembly. Service class file extension is .asmx having **@service** directive that identified class that class contains the code for service and in which it is located. This service is hosted into Internet Information Services (IIS).

WCF services can readily be hosted within IIS 5.1 or 6.0, the Windows Process Activation Service (WAS) that is provided as part of IIS 7.0, and within any .NET application. To host a service in IIS 5.1 or 6.0, the service must use HTTP as the communications transport protocol.

**Message Representation**

The headers of the SOAP messages sent or received on ASP.NET Web Services can be customized. A class is derived from **SoapHeader class** define the structure of header, and then the SoapHeader Attribute is used to indicate the presence of the header.

The WCF provides MessageContract attribute, MessageHeader attribute, and MessageBodyMember attribute to describe the structure of the SOAP messages sent and received by a service.

**Exception Handling**

In ASP.NET Web Services return unhandled exception as SOAP fault to client. Instance of **SoapException class** can throw explicitly to get more control on error transmission.

WCF provide more control in exception handling. The unhandled exception is not transferred as SOAP fault. Generic type exception’s instance, **FaultException** is throwing to client. You can add **FaultContract attribute** to specify the fault that the operation might yield.

[DataContract]

public class MathFaultException

{

[DataMember]

public string operation;

[DataMember]

public string problemType;

}

[ServiceContract]

public interface IWcfAddtion

{

[OperationContract]

string GetData(int value);

[OperationContract]

int GetDataUsingDataContract(CompositeType composite);

[OperationContract]

[FaultContract(typeof(MathFaultException))]

CompositeType AddValue(int item1, int item2);

}

You can catch exception in client using this way.

try

{

}

catch (FaultException exception)

{

}

**State Management**

The **HttpContext object** can be used to update and retrieve application state information by using its Application property, and can be used to update and retrieve session state information by using its Session property.

ASP.NET provides considerable control over where the session state information accessed by using the Session property of the **HttpContext** is actually stored. It may be stored in cookies, in a database, in the memory of the current server, or in the memory of a designated server. The choice is made in the service’s configuration file.

The WCF provide extensible object for state management which are implemented from IExtensibleObject. Examples of extensible objects are ServiceHostBase and InstanceContext. ServiceHostBase allows you to maintain state that all of the instances of all of the service types on the same host can access, while InstanceContext allows you to maintain state that can be accessed by any code running within the same instance of a service type.

**Security**

ASP.NET Web Services use IIS securities which are used in any IIS application. WCF is capable to run in any .NET executable, so it needs independent security capabilities. WCF has many security options, Claim-based Authorization is an example. It support for authorizing access to protected resources based on claims. Authorization based on claims is accomplished by comparing a set of claims to the access requirements of the operation and, depending on the outcome of that comparison, granting or denying access to the operation. In WCF, you can specify a class to use to run claims-based authorization, once again by assigning a value to the ServiceAuthorizationManager property of **ServiceAuthorizationBehavior.**

|  |
| --- |
| **Resource** [MSDN:Comparing ASP.NET Web Services to WCF Based on Development](http://msdn.microsoft.com/en-us/library/aa738737.aspx) |
|  |

Source: <http://www.dotnetpal.com/wcf/webservicewcf.htm>