# WCF

#### Difference between WCF and Web API and WCF REST and Web Service

Web Service

* It is based on SOAP and return data in XML form.
* It supports only HTTP protocol.
* It is not open source but can be consumed by any client that understands xml.
* It can be hosted only on IIS.

WCF

* It is also based on SOAP and return data in XML form.
* It is the evolution of the web service(ASMX) and support various protocols like TCP, HTTP, HTTPS, Named Pipes, MSMQ.
* The main issue with WCF is its tedious and extensive configuration.
* It is not open source but can be consumed by any client that understands xml.
* It can be hosted within the applicaion or on IIS or using window service.

WCF Rest

* To use WCF as WCF [Rest service](http://kellabyte.com/2011/09/04/clarifying-rest/) you have to enable webHttpBindings.
* It support HTTP GET and POST verbs by [WebGet] and [WebInvoke] attributes respectively.
* To enable other HTTP verbs you have to do some configuration in IIS to accept request of that particular verb on .svc files
* Passing data through parameters using a WebGet needs configuration. The UriTemplate must be specified
* It support XML, JSON and ATOM data format.

Web API

* This is the new framework for building HTTP services with easy and simple way.
* Web API is open source an ideal platform for building REST-ful services over the .NET Framework.
* Unlike WCF Rest service, it use the full featues of HTTP (like URIs, request/response headers, caching, versioning, various content formats)
* It also supports the MVC features such as routing, controllers, action results, filter, model binders, IOC container or dependency injection, unit testing that makes it more simple and robust.
* It can be hosted with in the application or on IIS.
* It is light weight architecture and good for devices which have limited bandwidth like smart phones.
* Responses are formatted by Web API’s MediaTypeFormatter into JSON, XML or whatever format you want to add as a MediaTypeFormatter.

To whom choose between WCF or WEB API

* Choose WCF when you want to create a service that should support special scenarios such as one way messaging, message queues, duplex communication etc.
* Choose WCF when you want to create a service that can use fast transport channels when available, such as TCP, Named Pipes, or maybe even UDP (in WCF 4.5), and you also want to support HTTP when all other transport channels are unavailable.
* Choose Web API when you want to create a resource-oriented services over HTTP that can use the full features of HTTP (like URIs, request/response headers, caching, versioning, various content formats).
* Choose Web API when you want to expose your service to a broad range of clients including browsers, mobiles, iphone and tablets.

**2. What are WCF Service Endpoints? Explain.**

For **Windows Communication Foundation** services to be consumed, it’s necessary that it must be exposed; Clients need information about service to communicate with it. This is where service endpoints play their role.A **WCF service** endpoint has three basic elements i.e. Address, Binding and Contract.**Address:** It defines “WHERE”. Address is the URL that identifies the location of the service.  
**Binding:** It defines “HOW”. Binding defines how the service can be accessed.  
**Contract:** It defines “WHAT”. Contract identifies what is exposed by the service.

**3. What are the possible ways of hosting a WCF service? Explain.**

For a **Windows Communication Foundation**service to host, we need at least a managed process, a ServiceHost instance and an Endpoint configured. Possible approaches for hosting a service are:

1. Hosting in a Managed Application/ Self Hosting  
a. Console Application  
b. Windows Application  
c. Windows Service  
2. Hosting on Web Server  
a. IIS 6.0 (ASP.NET Application supports only HTTP)  
b. Windows Process Activation Service (WAS) i.e. IIS 7.0 supports HTTP, TCP,  
NamedPipes, MSMQ.

**4. How we can achieve Operation Overloading while exposing WCF Services?**

By default, WSDL doesn’t support operation overloading. Overloading behavior can be achieved by using “Name” property of OperationContract attribute.

[ServiceContract]  
interface IMyCalculator  
{  
[OperationContract(Name = "SumInt")]  
int Sum(int arg1,int arg2); [OperationContract(Name = "SumDouble")]  
double Sum(double arg1,double arg2);  
}

When the proxy will be generated for these operations, it will have 2 methods with different names i.e. SumInt and SumDouble.

**Important Note:** *Remember that during a technical Interview, interviewer always ask about the latest feature of that particular technology, so be prepare for it also. For latest features series on Windows Communication Foundation v4.5,* [*Click here*](http://www.topwcftutorials.net/2014/03/whats-new-in-wcf-v4-5.html)*.*

**5. What Message Exchange Patterns (MEPs) supported by WCF? Explain each of them briefly.**

1. Request/Response  
2. One Way  
3. Duplex

**Request/Response**  
It’s the default pattern. In this pattern, a response message will always be generated to consumer when the operation is called, even with the void return type. In this scenario, response will have empty SOAP body.  
**One Way**  
In some cases, we are interested to send a message to service in order to execute certain business functionality but not interested in receiving anything back. OneWay MEP will work in such scenarios.  
If we want queued message delivery, OneWay is the only available option.  
**Duplex**  
The Duplex MEP is basically a two-way message channel. In some cases, we want to send a message to service to initiate some longer-running processing and require a notification back from service in order to confirm that the requested process has been completed.

**6. What is DataContractSerializer and How its different from XmlSerializer?**

Serialization is the process of converting an object instance to a portable and transferable format. So, whenever we are talking about **web services**, serialization is very important.

Windows Communication Foundation has DataContractSerializer that is new in .NET 3.0 and uses opt-in approach as compared to XmlSerializer that uses opt-out. Opt-in means specify whatever we want to serialize while Opt-out means you don’t have to specify each and every property to serialize, specify only those you don’t want to serialize.  
DataContractSerializer is about 10% faster than XmlSerializer but it has almost no control over how the object will be serialized. If we wanted to have more control over how object should be serialized that XmlSerializer is a better choice.

**WCF INTERVIEW QUESTIONS**

**What is WCF?**

Microsoft refers WCF as a programming platform that is used to build Service-oriented applications. Windows Communication Foundation is basically a unified programming model for developing, configuring and deploying distributed services. Microsoft has unified all its existing distributed application technologies (e.g. MS Enterprise Services, ASMX web services, MSMQ, .NET Remoting etc) at one platform i.e. WCF. Code name for WCF was Indigo.

**Why to use WCF? or What are the advantages for using WCF?**

* Service Orientation is one of the key advantages of WCF. We can easily build service-oriented applications using WCF.
* If compared with ASMX web services, WCF service provides reliability and security with simplicity.
* As oppose to .NET Remoting, WCF services are interoperable.
* Different clients can interact with same service using different communication mechanism. This is achieved by using service endpoints. A single WCF service can have multiple endpoints. So, developer will write code for service once and just by changing configuration (defining another service endpoint), it will be available for other clients as well.
* Extensibility is another key advantage of WCF. We can easily customize a service behavior if required.

**What are the core components of WCF Service?**

A WCF service has at least following core components.

* Service Class: A service class implementing in any CLR-based language and expose at least one method.
* Hosting Environment: a managed process for running service.
* Endpoint: a client uses it to communicate with service.

**What is the difference between WCF and ASMX Web services?**

The basic difference is that ASMX web service is designed to send and receive messages using SOAP over HTTP only. While WCF service can exchange messages using any format (SOAP is default) over any transport protocol (HTTP, TCP/IP, MSMQ, Named Pipes etc).  
You can find detailed discussion on [WCF Vs ASMX Web services](http://www.topwcftutorials.net/2012/06/wcf-vs-asmx-web-services.html) here.

**What are the different ways to expose WCF Metadata?**

By default, WCF doesn’t expose metadata. We can expose it by choosing one of the following ways:  
1. In configuration file, by enabling metadata exchange as follows:

*<system.serviceModel>*  
*<services>*  
*<service name=”MyService.Service1″*  
*behaviorConfiguration=”MyService.Service1″>*

*<endpoint address=””*

*binding=”wsHttpBinding”*  
*contract=”MyService.IService1″>*  
*<identity>*  
*<dns value=”localhost”/>*  
*</identity>*  
*</endpoint>*  
*<endpoint address=”mex” binding=”mexHttpBinding”*  
*contract=”IMetadataExchange”/>*  
*</service>*  
*</services>*  
*<behaviors>*  
*<serviceBehaviors>*  
*<behavior name=”MyService.Service1″>*  
*<serviceMetadata httpGetEnabled=”true”/>*  
*<serviceDebug includeExceptionDetailInFaults=”false”/>*  
*</behavior>*  
*</serviceBehaviors>*  
*</behaviors>*  
*</system.serviceModel>*

2. ServiceHost can expose a metadata exchange endpoint to access metadata at runtime.

*using (ServiceHost host = new ServiceHost(typeof(MyService)))*  
*{*  
*ServiceMetadataBehavior behavior = new ServiceMetadataBehavior();*  
*behavior.HttpGetEnabled = true;*  
*host.Description.Behaviors.Add(behavior);*  
*host.Open();*  
*Console.WriteLine(“My Service here……….”); Console.ReadLine();*  
*host.Close();*  
*}*

**What is mexHttpBinding in WCF?**

In order to generate proxy, we need service metadata and mexHttpBinding returns service metadata.

If we look into our configuration file, service will have an endpoint with mexHttpBinding as follows:  
***<endpoint address=”mex” binding=”mexHttpBinding” contract=”IMetadataExchange”/>***

and service metadata behavior will be configured as follows:  
***<serviceMetadata httpGetEnabled=”true”/>***

Before deployment of application to production machine, it should be disabled.  
In order to support other protocols, related bindings are mexHttpBinding, mexHttpsBinding, mexTcpBinding.

**What is a Service Proxy in Windows Communication Foundation?**

A service proxy or simply proxy in WCF enables application(s) to interact with WCF Service by sending and receiving messages. It’s basically a class that encapsulates service details i.e. service path, service implementation technology, platform and communication protocol etc. It contains all the methods of service contract (signature only, not the implementation). So, when the application interact the service through proxy, it gives the impression that it’s communicating a local object.  
We can create proxy for a service by using Visual Studio or SvcUtil.exe.

**What are the different ways to generate proxy in WCF?**

Generating proxy using Visual Studio is simple and straight forward.

* Right click References and choose “Add Service Reference”.
* Provide base address of the service on “Add Service Reference” dialog box and click “Go” button. Service will be listed below.
* Provide namespace and click OK.

Visual studio will generate a proxy automatically.

We can generate proxy using svcutil.exe utility using command line. This utility requires few parameters like HTTP-GET address or the metadata exchange endpoint address and a proxy filename i.e. optional.  
***svcutil http://localhost/MyService/Service1.svc /out:MyServiceProxy.cs***

If we are hosting the service at a different port(other than default for IIS which is 80), we need to provide port number in base address.  
***svcutil http://localhost:8080/MyService/Service1.svc /out:MyServiceProxy.cs***  
For parameter details regarding svcutil, please follow the MSDN link  
<http://msdn.microsoft.com/en-us/library/aa347733.aspx>

**Difference between using ChannelFactory and Proxies in WCF?**

A ChannelFactory creates a kind of Channel used by clients to communicate with service endpoints.

If we have control over Server and Client, then ChannelFactory is a good option because it relies on having local interfaces that actually describes the service i.e. service contract.  
On the other hand, If we don’t have control over server and only have WSDL/URL, then it’s better to generate proxy using Visual Studio or SvcUtil.  
SvcUtil is better option as compared to Visual Studio because we have more control in case of SvcUtil.

**What is a fault contract?**

Normally, by default, when some exception occurs at a WCF service level, it will not expose as it is to client. Reason is that WCF exception is a CLR exception and it doesn’t make sense to expose it outside CLR because it contains internal details of service code like stack trace. So, WCF handles and returns error details to client using Fault Contract.“So, fault contract is a contract that contains the details of possible exception(s) that might occur in a service code.”

*[ServiceContract]*  
*public interface IService1*  
*{*  
*[OperationContract]*  
*[FaultContract(typeof(MyFaultDetails))]*  
*int MyOperation1();*  
*}*

*[DataContract]*  
*public class MyFaultDetails*  
*{*  
*[DataMember]*  
*public string ErrorDetails { get; set; }*  
*}*

In implementing service…..

*public int MyOperation1()*  
*{*  
*Try{ //Do something…… }catch()*  
*{*  
*MyFaultDetails ex = new MyFaultDetails();*  
*ex.ErrorDetails = “Specific error details here.“;*  
*throw new FaultException(ex,“Reason: Testing…..“);*  
*}*  
*}*

**A user has a service with a one-way operation that includes a fault contract, and he gets an exception when he tries to host the service. Why?**

This is true, because, to return faults, the service requires some form of a two-way communication channel, which is not there with one-way operations.

**What are the core security concepts supported by WCF?**

There are four core security Features

**Confidentiality**: It’s a confirmation about the recipient. Only the valid recipient can read the message when it passed between service and client.  
**Integrity**: is to ensure that message received is not being tempered or changed during exchange.  
**Authentication**: is a way for the parties (sender and receiver) to identify each other.  
**Authorization**: ensures that what actions an authenticated user can perform?

**Difference between Message Level security and Transport Level security?**

Security can be configured at different levels in Windows Communication Foundation.

1. Transport Level Security
2. Message Level Security

**What is SOA (Service Oriented Architecture) and how WCF supports it?**

SOA is basically an architectural model that dictates few principles for building business applications in the form of independent, loosely coupled and interoperable services. These services are well defined, self-contained and can work together to achieve certain business functionality without depending on context or state of other services.

WCF supports almost all those principles dictated by Service Oriented Architecture for developing services; those are independent, loosely coupled and interoperable also. Please visit for detailed discussion on [WCF and SOA](http://visualstudiomagazine.com/articles/2011/06/01/pcnet_wcf-and-soa.aspx).

**What is ESB in SOA environment?**

In Service Oriented Architecture environment, ESB (Enterprise Service Bus) acts as a single interface for all messaging between applications and services in a loosely coupled manner. ESB is capable to call and subscribe different service provider’s methods and subscriptions respectively.

**What is Transaction Propagation? And how WCF support it?**

Transaction propagation is the ability to propagate transaction across the boundaries of a single service. Or in other words, we can say that a service can participate in a transaction that is initiated by a client.  
In a SOA environment, transaction propagation becomes a key requirement. As we know that WCF supports SOA, so it provides support for transaction propagation as well.  
To enable transaction propagation, we need to set the value of *TransactionFlow* property of the binding being used. This can be done programmatically as follows:

*WSHttpBinding bindingBeingUsed = new WSHttpBinding();*  
*bindingBeingUsed.TransactionFlow = “true”;*

Or It can be done declaratively by updating configuration file as follows:

*<bindings>*  
*<wsHttpBinding>*  
*<binding name=”binding1”*  
*transactionFlow=”true” />*  
*</wsHttpBinding>*  
*</bindings>*

Default value for *TransactionFlow* property is “False”.

**WCF Vs ASMX Web Services**

1. ASMX web services can be hosted only in IIS while WCF service has all the following hosting options:  
   a. IIS  
   b. WAS (Windows Process Activation Services)  
   c. Console Application  
   d. Windows NT Services  
   e. WCF provided Host
2. ASMX web services support is limited toHTTP while WCF supports HTTP, TCP, MSMQ, NamedPipes.
3. ASMX Security is limited. Normally authentication and authorization is done using IIS and ASP.NET security configuration and transport layer security.For message layer security, WSE can be used.  
   WCF provides a consistent security programming model for any protocol and it supports many of the same capabilities as IIS and WS-\* security protocols, additionally, it provides support for claim-based authorization that provides finer-grained control over resources than role-based security.WCF security is consistent regardless of the host that is used to implement WCF service.
4. Another major difference is that ASMX web services uses *XmlSerializer* for serialization while WCFuses *DataContractSerializer* which is far better in performance than XmlSerializer.  
   Key Issues with XmlSerializer in serializing .NET types to xml are:  
   a. Only public fields or properties of the .NET types can be translated to Xml.  
   b. Only the classes that implement IEnumerable can be translated.  
   c. Classes that implement IDictionary, such as Hashtable cannot be serialized.