WCF Interview questions and answers (reference only, uuber level)

The intent to draft this list of questions is for learning purpose only and at very high level understanding of WCF services. This is not a dump :)   
  
Question 1: **What is the difference WCF and Web services?**   
Answer: -

Web services can only be invoked by HTTP. While Service or a WCF component can be invoked by any protocol and any transport type. Second web services are not flexible. However, Services are flexible. If you make a new version of the service then you need to just expose a new end. Therefore, services are agile and which is a very practical approach looking at the current business trends.

Question 2: **What are different bindings supported by WCF?**   
Answer: -

WCF includes predefined bindings. They cover most of bindings widely needed in day-to-day application. However, just incase you find that you need to define something custom WCF does not stop you. So let us try to understand what each binding provides.

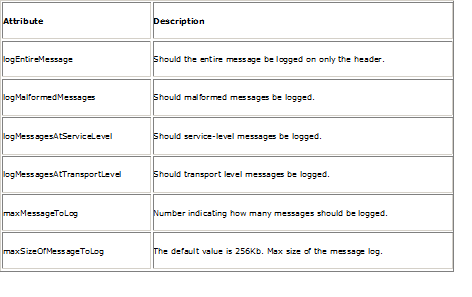
BasicHttpBinding: - This binding is used when we need to use SOAP over HTTP. This binding can also be configured to be used as HTTPS. It can be also configured to send data in plain text or in optimized form like MTOM.   
WsHttpBinding: - It is same like BasicHttpBinding. In short, it uses SOAP over HTTP. But with it also supports reliable message transfer, security and transaction. WS-Reliable Messaging, security with WS-Security, and transactions with WS-Atomic Transaction supports reliable message.   
NetTcpBinding: - This binding sends binary-encoded SOAP, including support for reliable message transfer, security, and transactions, directly over TCP. The biggest disadvantage of NetTcpBinding is that both server and client should be also made in .NET language.   
NetMsmqBinding: - This binding sends binary-encoded SOAP over MSMQ. This binding can only be used for WCF-to-WCF communication.   
NetNamedPipesBinding:-Ths binding Sends binary-encoded SOAP over named pipes. This binding is only usable for WCF-to-WCF communication between processes on the same Windows-based machine.  
  
Note: - An interprocess control (IPC) protocol is used for exchanging information between two applications, possibly running on different computers in a network. The difference between Named pipes and TCP is that named pipes have good performance in terms of communication with in processes. But when it comes to communicate across network TCP holds the best choice. So if you are using WCF to communicate with process it’s the best choice to use in terms for performance. Named pipes do not perform when the traffic is heavy as compared to TCPIP.   
  
Question 3: **What are the main components of WCF?**   
Answer: -

We need to define three main components in WCF:-

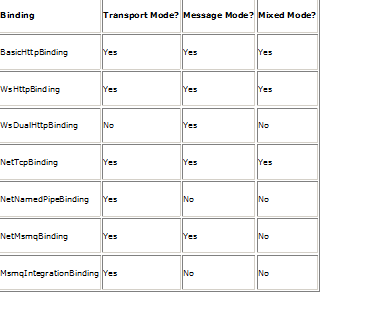
• Service class. • Hosting environment • End point   
  
Question 4: **What are the various ways of hosting a WCF service?**   
Answer: -

There are three major ways to host a WCF service:-

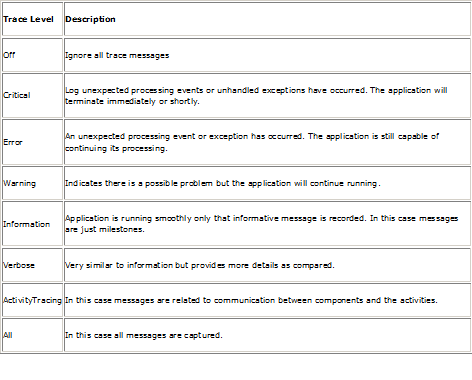
• Self-hosting the service in his own application domain. This we have already covered in the first section. The service comes in to existence when you create the object of Service Host class and the service closes when you call the Close of the Service Host class.   
• Host in application domain or process provided by IIS Server.   
• Host in Application domain and process provided by WAS (Windows Activation Service) Server.   
  
Question 5: **What is a service level message and transport level message?**   
Answer: -   
You can log WCF message at two levels one is service level and the other is transport level. Service level:-In this the messages are logged as they enter the user code or leave the user code. Transport level: - In this the messages are logged as they are ready to be encoded / decoded. All transport level, infrastructure messages and also reliable messaging is logged. You specify the message levels in the diagnostics node as shown in the below code snippet. 

[](http://4.bp.blogspot.com/-BPY-mCr0g1k/Tn-DrkwQB4I/AAAAAAAAFrM/RM_uxOOsPUo/s1600/attributes.png)

**Question 6: For which bindings are transport, message and mixed mode supported?**   
Answer: -   
Below is a table which shows for which binding which mode is supported. We did not discuss the mixed mode. It’s nothing but combination of transport and mixed mode. For instance data encrypted and passed over WsHttp using HTTPS is a mixed mode of security. Encryption is nothing but message security and HTTPS is a transport mode. In a combination they form mixed mode.

[](http://1.bp.blogspot.com/-MYdyKno2GTw/Tn-DSybI3hI/AAAAAAAAFrE/DaQU39H73kk/s1600/binding.png)

Question 7: Which bindings in WCF support the message streaming?   
Answer: -   
Following bindings supports the streaming in WCF:   
1. basicHttpBinding 2. netTcpBinding 3. netNamedPipeBinding   
  
**Question 8: What are the various ways of hosting a WCF Service?**   
Answer: - 1.IIS 2.Self Hosting 3.WAS (Windows Activation Service)  
  
**Question 9: What is the use of ServiceBehavior attribute in WCF ?**   
Answer: -   
ServiceBehaviour attribute is used to specify the InstanceContextMode for the WCF Service class (This can be used to maintained a state of the service or a client too).   
  
There are three instance Context Mode in the WFC PerSession : This is used to create a new instance for a service and the same instance is used for all method for a particular client. (eg: State can be maintained per session by declaring a variable) PerCall : This is used to create a new instance for every call from the client whether same client or different. (eg: No state can be maintained as every time a new instance of the service is created) Single : This is used to create only one instance of the service and the same instance is used for all the client request. (eg: Global state can be maintained but this will be applicable for all clients)   
  
**Question 10: How the concurrency mode is specified in WCF service?**   
Answer: - The concurrency mode is specified using the ServiceBehavior attribute on the class that implements the service. Ex. [ServiceBehavior(ConcurrencyMode=ConcurrencyMode.Single)] Public class ServiceClass : IServiceInterface{ //Implementation Code } There are 3 possible values of ConcurrencyMode enumeration Single Reentrant Multiple.   
  
**Question 11: hat the different transaction options ?**   
Answer:-   
  
We can specify transaction in 3 ways in WCF:-   
**TransactionFlowOption.NotAllowed** This is a default option. Using this option no transaction will be propagated across the binding. If any client attempts to call the WCF service in a transaction it will be ignored for this option. This option specifies that client can call this WCF service in a transaction. It’s not compulsory that the service needs to be called in a transaction. You can call without the transaction also. **TransactionFlowOption.Mandatory** This option specifies that client must call the WCF service in a transaction mode. If the WCF service is called without transaction, ‘FaultException’ will be raised.   
  
**Question 12: Wat are the advantages of hosting WCF Services in IIS as compared to self-hosting?**   
Answer: - There are two main advantages of using IIS over self-hosting:-   
  
**Automatic activation** IIS provides automatic activation that means the service is not necessary to be running in advance. When any message is received by the service it then launches and fulfills the request. But in case of self-hosting the service should always be running.   
**Process recycling** If IIS finds that a service is not healthy that means if it has memory leaks etc, IIS recycles the process. Ok let us try to understand what is recycling in IIS process. For every browser instance, a worker process is spawned and the request is serviced. When the browser disconnects the worker, process stops and you lose all information. IIS also restarts the worker process. By default, the worker process is recycled at around 120 minutes. So why does IIS recycle. By restarting the worker process it ensures any bad code or memory leak do not cause issue to the whole system. In case of self-hosting both the above features, you will need to code yourself.   
  
**Question 13: What is trace listener?**   
  
Answer - ‘Tracelistener’ are objects that get tracing information from the trace class and they output the data to some medium. For instance you can see from the figure ‘TraceListener’ how it listens to the trace object and outputs the same to UI, File or a windows event log. There are three different types of ‘tracelistener’ first is the ‘defaulttracelistener’ (this outputs the data to UI), second is ‘textwritertracelistener’ (this outputs to a file) and the final one is ‘Eventlogtracelistener’ which outputs the same to a windows event log. 

[](http://3.bp.blogspot.com/--oIQpIdznLI/Tn-GPHuhrkI/AAAAAAAAFrU/9yzt3Uplug0/s1600/attributes.png)

**Question 14: What is service host factory in WCF?**   
Answer-   
  
1. Service host factory is the mechanism by which we can create the instances of service host dynamically as the request comes in.   
2. This is useful when we need to implement the event handlers for opening and closing the service.   
3. WCF provides ServiceFactory class for this purpose.   
  
**Question 15: What are the advantages of hosting WCF service in WAS?**   
  
Answer- WAS (Windows Activation Service) is a component of IIS 7.0. Following are few advantages : 1. We are not only limited to HTTP protocol. We can also use supported protocols like TCP, named pipes and MSMQ 2. No need to completely install IIS. We can only install WAS component and keep away the WebServer.   
  
**Question 16: In WCF which bindings supports the reliable session?**   
  
Answer- In WCF, following bindings supports the reliable session - 1. wsHttpBinding 2. wsDualHttpBinding 3. wsFederationHttpBinding 4. netTcpBinding   
  
Reference - Prasham on dotnetfunda.com and Pro WCF: Practical Microsoft SOA Implementation -- Chris and Denis Mulder – Apress 2007.

**What are the important principles of SOA (Service oriented Architecture)?**

**What are ends, contract, address, and bindings?**

**Which specifications does WCF follow?**

**What are the main components of WCF?**

**Explain how Ends, Contract, Address, and Bindings are done in WCF?**

**What is a service class?**

**What is a service contract, operation contract and Data Contract?**

**What are the various ways of hosting a WCF service?**

**How do we host a WCF service in IIS?**

**What are the advantages of hosting WCF Services in IIS as compared to self-hosting?**

**What are the major differences between services and Web services?**

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

**What are different bindings supported by WCF?**

**Which are the various programming approaches for WCF?**

**What is one-way operation?**

**Can you explain duplex contracts in WCF?**

**How can we host a service on two different protocols on a single server?**

**How can we use MSMQ bindings in WCF?**

**Can you explain transactions in WCF?**

**What different transaction isolation levels provided in WCF?**

**Can we do transactions using MSMQ?**

**Can we have two-way communications in MSMQ?**

**What are Volatile queues?**

**What are Dead letter queues?**

**What is a poison message?**

**What is WCF?**  
Windows Communication Foundation (WCF) is an SDK for developing and deploying services on Windows. WCF provides a runtime environment for services, enabling you to expose CLR types as services, and to consume other services as CLR types.  
  
WCF is part of .NET 3.0 and requires .NET 2.0, so it can only run on systems that support it. WCF is Microsoft’s unified programming model for building service-oriented applications with managed code. It extends the .NET Framework to enable developers to build secure and reliable transacted Web services that integrate across platforms and interoperate with existing investments.  
  
Windows Communication Foundation combines and extends the capabilities of existing Microsoft distributed systems technologies, including Enterprise Services, System.Messaging, Microsoft .NET Remoting, ASMX, and WSE to deliver a unified development experience across multiple axes, including distance (cross-process, cross-machine, cross-subnet, cross-intranet, cross-Internet), topologies (farms, fire-walled, content-routed, dynamic), hosts (ASP.NET, EXE, Windows Presentation Foundation, Windows Forms, NT Service, COM+), protocols (TCP, HTTP, cross-process, custom), and security models (SAML, Kerberos, X509, username/password, custom).  
  
**What is service and client in perspective of data communication?**  
A service is a unit of functionality exposed to the world. The client of a service is merely the party consuming the service.  
  
**What is endpoint in WCF?** or **What is three major points in WCF?**  
Every service must have Address that defines where the service resides, Contract that defines what the service does and a Binding that defines how to communicate with the service.  
  
In WCF the relationship between Address, Contract and Binding is called Endpoint. The Endpoint is the fusion of Address, Contract and Binding.  
  
**1. Address**: Specifies the location of the service which will be like http://Myserver/MyService.Clients will use this location to communicate with our service.  
  
**2. Contract**: Specifies the interface between client and the server.It’s a simple interface with some attribute.  
  
**3. Binding**: Specifies how the two paries will communicate in term of transport and encoding and protocols.  
 **What is binding and how many types of bindings are there in WCF?**  
A binding defines how an endpoint communicates to the world. A binding defines the transport (such as HTTP or TCP) and the encoding being used (such as text or binary).  
  
A binding can contain binding elements that specify details like the security mechanisms used to secure messages, or the message pattern used by an endpoint.  
  
WCF supports nine types of bindings.  
  
**1. Basic binding :**  
Offered by the BasicHttpBinding class, this is designed to expose a WCF service as a legacy ASMX web service, so that old clients can work with new services. When used by the client, this binding enables new WCF clients to work with old ASMX services.  
 **2. TCP binding :**  
Offered by the NetTcpBinding class, this uses TCP for cross-machine communication on the intranet. It supports a variety of features, including reliability, transactions, and security, and is optimized for WCF-to-WCF communication. As a result, it requires both the client and the service to use WCF.  
  
**3. Peer network binding :**  
Offered by the NetPeerTcpBinding class, this uses peer networking as a transport. The peer network-enabled client and services all subscribe to the same grid and broadcast messages to it.  
  
**4. IPC binding :**  
Offered by the NetNamedPipeBinding class, this uses named pipes as a transport for same-machine communication. It is the most secure binding since it cannot accept calls from outside the machine and it supports a variety of features similar to the TCP binding.  
  
**5. Web Service (WS) binding :**  
Offered by the WSHttpBinding class, this uses HTTP or HTTPS for transport, and is designed to offer a variety of features such as reliability, transactions, and security over the Internet.  
 **6. Federated WS binding :**  
Offered by the WSFederationHttpBinding class, this is a specialization of the WS binding, offering support for federated security.  
  
**7. Duplex WS binding :**  
Offered by the WSDualHttpBinding class, this is similar to the WS binding except it also supports bidirectional communication from the service to the client.  
 **8. MSMQ binding :**  
Offered by the NetMsmqBinding class, this uses MSMQ for transport and is designed to offer support for disconnected queued calls.  
 **9. MSMQ integration binding :**  
Offered by the MsmqIntegrationBinding class, this converts WCF messages to and from MSMQ messages, and is designed to interoperate with legacy MSMQ clients.  
  
**What is contracts in WCF?**  
In WCF, all services expose contracts. The contract is a platform-neutral and standard way of describing what the service does.  
  
WCF defines four types of contracts.  
**1. Service contracts** : Describe which operations the client can perform on the service.  
  
**2. Data contracts** : Define which data types are passed to and from the service. WCF defines implicit contracts for built-in types such as int and string, but we can easily define explicit opt-in data contracts for custom types.  
  
**3. Fault contracts** : Define which errors are raised by the service, and how the service handles and propagates errors to its clients.  
  
**4. Message contracts** : Allow the service to interact directly with messages. Message contracts can be typed or untyped, and are useful in interoperability cases and when there is an existing message format we have to comply with.  
  
**What is address in WCF and how many types of transport schemas are there in WCF?**  
Address is a way of letting client know that where a service is located. In WCF, every service is associated with a unique address. This contains the location of the service and transport schemas.   
  
**WCF supports following transport schemas**  
1. HTTP  
2. TCP  
3. Peer network  
4. IPC (Inter-Process Communication over named pipes)  
5. MSMQ  
  
The sample address for above transport schema may look like  
  
http://localhost:81  
http://localhost:81/MyService  
net.tcp://localhost:82/MyService  
net.pipe://localhost/MyPipeService  
net.msmq://localhost/private/MyMsMqService  
net.msmq://localhost/MyMsMqService  
  
**What is the difference WCF and Web services?**  
**1.** Web services can only be invoked by HTTP. While Service or a WCF component can be invoked by any protocol and any transport type.  
  
**2.** Second web services are not flexible. But Services are flexible. If you make a new version of the service then you need to just expose a new end point. So services are agile and which is a very practical approach looking at the current business trends.  
  
**How can we host a service on two different protocols on a single server?**  
Let’s first understand what this question actually means. Let’s say we have made a service and we want to host this service using HTTP as well as TCP.  
  
You must be wondering why to ever host services on two different types of protocol. When we host a service it’s consumed by multiple types of client and it’s very much possible that they have there own protocol of communication. A good service has the capability to downgrade or upgrade its protocol according the client who is consuming him.  
  
Let’s do a small sample in which we will host the ServiceGetCost on TCP and HTTP protocol.  
  
Once we are done the server side coding its time to see make a client by which we can switch between the protocols and see the results. Below is the code snippet of the client side for multi-protocol hosting  
  
**How does WCF work?**  
Follows the ‘software as a service’ model, where all units of functionality are defined as services.  
  
A WCF Service is a program that exposes a collection of Endpoints. Each Endpoint is a portal (connection) for communication with either clients (applications) or other services.  
  
Enables greater design flexibility and extensibility of distributed systems architectures.  
  
A WCF application is represented as a collection of services with multiple entry points for communications.  
  
**What are the main components of WCF?**  
**1.Service:** The working logic or offering, implemented using any .Net Language©.  
 **2.Host:** The environment where the service is parked. E.g. exe, process, windows service  
  
**3.Endpoints:** The way a service is exposed to outside world.  
  
**Explain transactions in WCF.**  
Transactions in WCF allow several components to concurrently participate in an operation. Transactions are a group of operations that are atomic, consistent, isolated and durable. WCF has features that allow distributed transactions. Application config file can be used for setting transaction timeouts.  
  
**What are different isolation levels provided in WCF?**  
The different isolation levels:  
  
**1. READ UNCOMMITTED: –** An uncommitted transaction can be read. This transaction can be rolled back later.  
 **2. READ COMMITTED :-** Will not read data of a transaction that has not been committed yet  
  
**3. REPEATABLE READ: –** Locks placed on all data and another transaction cannot read.  
  
**4. SERIALIZABLE:-** Does not allow other transactions to insert or update data until the transaction is complete.  
  
**How do I serialize entities using WCF?**  
LINQ to SQL supports serialization as XML via WCF by generating WCF serialization attributes and special serialization specific logic during code-generation. You can turn on this feature in the designer by setting serialization mode to ‘Unidirectional’. Note this is not a general solution for serialization as unidirectional mode may be insufficient for many use cases.  
  
**What is End point ?**  
Every service must have Address that defines where the service resides, Contract that defines what the service does and a Binding that defines how to communicate with the service. In WCF the relationship between Address, Contract and Binding is called Endpoint.  
  
The Endpoint is the fusion of Address, Contract and Binding.

# Interview Questions Answers – Windows Communication Foundation (WCF)



[**Adarsh Chaurasia (Consultant|Mentor|Tech Savvy)**](http://www.codeproject.com/script/Membership/View.aspx?mid=10590527), 14 Apr 2014 [CPOL](http://www.codeproject.com/info/cpol10.aspx)

|  |  |
| --- | --- |
| http://dj9okeyxktdvd.cloudfront.net/script/Ratings/Images/stars-fill-lg.png  http://dj9okeyxktdvd.cloudfront.net/script/Ratings/Images/stars-empty-lg.png | 4.74 (15 votes) |
|  | |  |  |  | | --- | --- | --- | | Rate this: | [vote 1vote 2vote 3vote 4vote 5](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication) |  | |

In this article, I have tried to answer all the frequently asked WCF questions in interviews. It helps you not only to crack interviews but to refresh the concepts of WCF programming. What is service ? A service is a unit of functionality exposed to the world. Service orientation (SO) is an abstract

In this article, I have tried to answer all the frequently asked WCF questions in interviews. It helps you not only to crack interviews but to refresh the concepts of WCF programming.

## What is service ?

A service is a unit of functionality exposed to the world. Service orientation (SO) is an abstract set of principles and best practices for building service-oriented applications.

## What do you mean by client ?

The client of a service is the program unit consuming its functionality. The client can  
be literally anything—for instance, Console application, Windows Forms, WPF, or Silverlight class, anASP.NET page, or another service.

## What is WCF ?

* Stands for Windows Communication Foundation.
* Its code name is “Indigo”.
* It is a framework for building, configuring and deploying interoperable distributed services.
* It enables you to write more secure flexible services without any code change (using configuration).
* It also provide built-in support for logging. You can enable/disable logging using configuration.

WCF = Web Service **+** Remoting **+** MSMQ **+** COM+

or

WCF = ASMX **+** .Net Remoting **+** WSE + Messaging **+** Enterprise Services

## What are the transport schemes supported by WCF ? Give example of address for each scheme.

Following are the transport schemes supported by WCF:

* HTTP/HTTPS - http://localhost:8001/MyService
* TCP - net.tcp://localhost:8002/MyService
* IPC - net.pipe://localhost/MyPipe
* Peer network
* MSMQ - net.msmq://localhost/private/MyQueue
* Service bus - sb://MyNamespace.servicebus.windows.net/

## What is Contract ? What are the types of Contract ?

It is the agreement between client and service which specifies:

* [ServiceContract] - which services are exposed to the client.
* [OperationContract] - which operations the client can perform on the service.
* [DataContract] – which data types are passed to and from the service.
* [MessageContract] - allow the service to interact directly with messages. Message contracts can be typed or untyped and are useful in interoperability cases when another party has alreadydictated some explicit (typically proprietary) message format.
* [FaultContract] -which errors are raised by the service and how the service handles andpropagates errors to its clients.

## What is the difference between Web Service and WCF Service ?

|  |  |  |
| --- | --- | --- |
| **Features** | **Web Service** | **WCF** |
| Hosting | It can be hosted in IIS. | It can be hosted in IIS, WAS (windows activation service), Self-hosting or Windows service |
| Programming | Apply [WebService] attribute to the class to be exposed as a service. | Apply [ServiceContract] attribute to the class to be exposed as a service. |
| Model | Apply [WebMethod] attribute to the method exposed to client. | Apply [OperationContract] attribute to the method exposed to client. |
| Supported Operations | One-way and Request- Response. | One-Way, Request-Response and Duplex. |
| Logging | Needs custom implementation. | No custom implementation needed. Can be configured in service config file. |
| Serialization | System.Xml.serialization namespace is used. | System.Runtime.Serialization namespace is used. |
| Supported Encoding | XML 1.0, MTOM(Message Transmission Optimization Mechanism), DIME, Custom. | XML 1.0, MTOM, Binary, Custom. |
| Supported Transports | Can be accessed through HTTP, TCP and Custom. | Can be accessed through HTTP, TCP, Named pipes, MSMQ,P2P(Peer to Peer) and Custom. |
| Service Types | ASMX, .Net Remoting | .ASMX, .Net Remoting, WSE(WS\* Protocols), MSMQ and Enterprise Services |
| Supported Protocols | Security | Security, Reliable messaging, Transactions |

## How can we host WCF service ?

Every WCF service needs to be hosted in a windows process called the host process. A single host process can host multiple services, and the same service type can be hosted in multiple host processes.

WCF services can be hosted in many ways:

**IIS 5/6 Hosting**

WCF services can be hosted in IIS. It is similar to hosting a ASMX web service. To host a WCF service, you need to create a .svc file similar to this example and put it in virtual directory of IIS:

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

<%@ ServiceHost Language = "C#" Debug = "true" CodeBehind = "˜/App\_Code/MyService.cs" Service = "MyService" %>

Instead of defining a .svc file, you can create and host the service in .config file as below:

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

<system.serviceModel>

<serviceHostingEnvironment>

<serviceActivations>

<add relativeAddress = "MyService.svc" service = "WcfService.MyService"/>

<add relativeAddress = "MyOtherService.svc" service = "MyOtherService"/>

</serviceActivations>

</serviceHostingEnvironment>

<services>

<service name = "WcfService.MyService">

...

</service>

<service name = "MyOtherService">

...

</service>

</services>

</system.serviceModel>

**Self Hosting**

Self-hosting is the technique in which the developer is responsible for providing and  
managing the lifecycle of the host process. You can host WCF service inside any Windows process, such as a Windows Forms application, a WPF application, a Console application, or a Windows NT Service.

You can also host your WCF service as in-proc. In-process (or in-proc) hosting is the hosting technique where the service resides in the same process as the client. By definition, the developer provides the host for the in-proc hosting.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

var host = new ServiceHost(typeof(MyService));

host.Open();

Console.WriteLine("Press any key to stop service");

Console.ReadKey();

host.Close();

**WAS (Windows Activation Service) Hosting**

Microsoft provides a generalpurpose hosting engine called the Windows Activation Service (WAS). WAS is a system service available with Windows Vista, Windows Server 2008, and Windows 7 (or later). The WAS is a true general-purpose hosting engine. It can host websites (in fact, IIS 7 will host its websites in the WAS by default), but it can just as easily host your services, allowing you to use any transport, such as TCP, IPC, or MSMQ. You can install and configure the WAS separately from IIS 7. Hosting a WCF service in the WAS is designed to look just like hosting in IIS 5/6. You need to either supply an .svc file, just as with IIS 5/6, or provide the equivalent information in the config file.

**Custom Hosting in IIS/WAS**

When using IIS 5/6 or WAS, you have no direct access to the host. To overcome this hurdle, WCF provides a hook called a host factory. Using the Factory tag in the .svc file, you can specify a class you provide that creates the host instance.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

<%@ ServiceHost Language = "C#" Debug = "true" CodeBehind = "˜/App\_Code/MyService.cs" Service = "MyService" **Factory = "MyServiceFactory"** %>

You can also specify the host factory in the config file when not using an .svc file explicitly:

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

<serviceActivations>

<add relativeAddress = "MyService.svc" service = "MyService" factory = "MyServiceFactory" />

</serviceActivations>

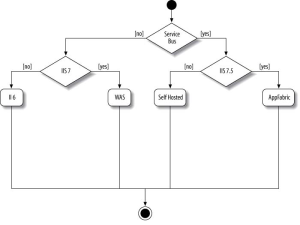
Note: The host factory class must derive from the ServiceHostFactory class and override the CreateServiceHost() virtual method.

**Windows Server AppFabric**

AppFabric is an extension to the WAS. It is geared more toward WF services, which require support for persistence and state management correlation. Windows Server AppFabric adds items for managing and monitoring the services, as well as WCF and WF configuration items, to the IIS 7 management console. Windows Server AppFabric provides a dashboard for monitoring the running instances of WCF or WF services, and is reminiscent of the MTS or COM+ Component Services Explorer of old. Windows Server AppFabric provides health monitoring and custom diagnostics, as well as some troubleshooting features for analyzing why a service call has failed.

Windows Server AppFabric also supports scripting of all the options available in the user interface. Windows Server AppFabric offers its own events collecting system service, which stores the events in a SQL Server database. You can provide Windows Server AppFabric with tracing and tracking profiles at various verbosity levels.

## How do you choose the hosting for WCF internet service ?

[](http://synvistech.com/blogs/wp-content/uploads/2014/04/HowToChooseHost.png)

## What are the protocols supported by WCF hosting environment ? What are their advantages and disadvantages ?

WCF support multiple ways in which you can host your WCF service.

|  |  |
| --- | --- |
| **Hosting Environment** | **Supported protocol** |
| IIS6 | http, wshttps |
| IIS7 – WAS (Windows Process Activation Service) | http,net.tcp,net.pipe,net.msmq |
| Windows console and form application | http,net.tcp,net.pipe,net.msmq |
| Windows service application (formerly known as NT services) | http,net.tcp,net.pipe,net.msmq |

Below is the feature summary of hosting environments:

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Self-Hosting** | **IIS Hosting** | **WAS Hosting** |
| Executable Process/ App Domain | Yes | Yes | Yes |
| Configuration | App.config | Web.config | Web.config |
| Activation | Manual at startup | Message-based | Message-based |
| Idle-Time Management | No | Yes | Yes |
| Health Monitoring | No | Yes | Yes |
| Process Recycling | No | Yes | Yes |
| Management Tools | No | Yes | Yes |

## What is binding ?

A binding is the set of configurations regarding the transport protocol, message encoding, communication pattern, reliability, security, transaction propagation, and interoperability.

## What are the types of bindings supported by WCF ? What are their advantages and disadvantages ?

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Binding** | **Feature** | **Suitable For** | **Transport** | **Message encoding** | **Security Mode** | **Resource Manager** | **Transaction Flow** |
| BasicHttpBinding | Not secure by default. | Communication with WS-Basic Profile conformant Web Services like ASMX. | HTTP | Text | None | X | X |
| WSHttpBinding | Secure, Interoperable. | Non-duplex service contracts. | HTTP | Text | Message | Disabled | WS-Atomic |
| WSDualHttpBinding | Secure, Interoperable. | Duplex service contracts or communication through SOAP intermediaries. | HTTP | Text | Message | Enabled | WS-Atomic transaction |
| WSFederationHttpBinding | Secure, Interoperable. | Supports the WS-Federation protocol, enabling organizations that are in a federation to efficiently authenticate and authorize users. | HTTP | Text | Message | Disabled | WS-Atomic transaction |
| NetTcpBinding | Secure, Optimized. | Cross-machine communication between WCF applications. | TCP | Binary | Transport | Disabled | Ole transaction. |
| NetPeerTcpBinding | Secure. | Multi-machine communication. | P2P | Binary | Transport | X | X |
| NetNamedPipesBinding | Secure, Reliable, Optimized. | On-machine communication between WCF applications. | Named Pipes | Binary | Transport | X | Ole transaction. |
| NetMsmqBinding |  | Cross-machine communication between WCF applications. | MSMQ | Binary | Message | X | X |
| MsmqIntegrationBinding | Does not use a WCF message encoding – instead it lets you choose a pre-WCF serialization format. | Cross-machine communication between a WCF application and existing MSMQ applications. | MSMQ | Pre-WCF format | Transport | X | X |

## What is Endpoint in WCF ?

## or

## What is ABC in WCF ?

Endpoint = Address (**A**) + Binding (**B**) + Contract (**C**)

Address specifies **where** the services is hosted.

Binding specifies **how** to access the hosted service. It specifies the transport, encoding, protocol etc.

Contract specifies **what** type of data can be sent to or received from the service.

Eg:

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

<endpoint name="BasicHttpGreetingService" address="http://localhost:5487/MyService/GreetingService.svc" binding="basicHttpBinding" contract="MyNamespace.MyService.IGreetingService" />

## Can you explain Address in detail ?

It is the url which specifies the location of the service. Client can use this url to connect to the service and invoke the service methods.

Eg: http://localhost:5487/MyService/GreetingService.svc

## Can you explain Binding in detail ?

It specifies how to access the hosted service. There are following characteristics of binding:

**Transport** defines the communication protocol to be used to communicate between service and client. It may be HTTP, TCP, MSMQ, NamedPipes etc. It is mandatory to define transport.

**Encoding** defines the technique used to encode the data before communicating it from one end to the other.

**Protocol** defines the configurations like reliability, security, transaction, timouts, message size etc.

## What is binding configuration ?

You can customize the binding used by endpoint using config file. For example, you can enable/disable transaction for the binding used by endpoint. All you need is to configure binding element in config file similar as below:

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

<bindings>

<netTcpBinding>

<binding name = "TransactionalTCP" transactionFlow = "true" />

</netTcpBinding>

</bindings>

## What is default endpoints ?

If the service host does not define any endpoints (neither in config nor programmatically) but does provide at least one base address, WCF will by default add endpoints to the service. These are called the default endpoints. WCF will add an endpoint per base address per contract, using the base address as the endpoint’s address. WCF will infer the binding from the scheme of the base address. For HTTP, WCF will use the basic binding. Note that the default bindings will affect the default endpoints. WCF will also name the endpoint by concatenating the binding name and the contract name.

# How can we enable/disable metadata publishing of our WCF service ?

You can enable enable meta data publishing for a WCF service two ways:

* Configure metadata publishing for a service that uses default endpoints. Specify the ServiceMetadataBehavior in the configuration file but do not specify any endpoints.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

<configuration>

<system.serviceModel>

<behaviors>

<serviceBehaviors>

<behavior name="CustomServiceBehavior">

<serviceMetadata httpGetEnabled="True" />

<serviceDebug includeExceptionDetailInFaults="False" />

</behavior>

</serviceBehaviors>

</behaviors>

</system.serviceModel>

</configuration>

* Configure metadata publishing for a service that uses explicit endpoints. Specify the ServiceMetadataBehavior in the configuration file and a mex endpoint.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

<configuration>

<system.serviceModel>

<services>

<service name="MyNamespace.MyService.GreetingService" behaviorConfiguration="CustomServiceBehavior">

<endpoint address="" binding="wsHttpBinding" contract="MyNamespace.MyService.IGreetingService" />

<endpoint address="mex" binding="mexHttpBinding" contract="IMetadataExchange" />

</service>

</services>

<behaviors>

<serviceBehaviors>

<behavior name="CustomServiceBehavior">

<serviceMetadata httpGetEnabled="True" />

</behavior>

</serviceBehaviors>

</behaviors>

</system.serviceModel>

</configuration>

Supported bindings for mex endpoint are mexHttpBinding, mexHttpsBinding, mexNamedPipeBinding andmexTcpBinding.

## How can you generate proxy class and configuration file for WCF service ?

WCF provides an utility **svcutil.exe** which can be used to generate proxy class and configuration file. Eg:

SvcUtil http://localhost:8002/MyService/ /out:Proxy.cs /noconfig

## Is there any tool provided by Microsoft for editing configuration file ?

Yes. Microsoft provides an utility “**SvcConfigEditor.exe**” that can edit any configuration file.

## How can you test your new WCF service without writing any client application ?

Microsoft provides a tool which can be used to test any WCF service. To use this tool, open visual studio command prompt and execute the command “**wcftestclient.exe**“. It will open a window where you can add many WCF services and test. You can also provide values for input parameters of WCF methods.

## Which bindings support reliability and message ordering ?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Binding name** | **Supports** **reliability** | **Default** **reliability** | **Supports ordered** **delivery** | **Default Ordered** **delivery** |
| BasicHttpBinding | No | N/A | No | N/A |
| NetTcpBinding | Yes | Off | Yes | On |
| NetNamedPipeBinding | No | N/A(On) | Yes | N/A(On) |
| WSHttpBinding | Yes | Off | Yes | On |
| NetMsmqBinding | No | N/A | No | N/A |

## How can you configure reliability using .config file ?

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

<bindings>

<netTcpBinding>

<binding name = "ReliableTCP">

<**reliableSession enabled = "true"**/>

</binding>

</netTcpBinding>

</bindings>

## How can you implement operation overloading in WCF service ?

We can implement operation overloading using “Name” property of OperationContract attribute. For eg:

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

[ServiceContract]

interface ICalculator

{

[OperationContract(**Name = "AddInt"**)]

int Add(int arg1,int arg2);

[OperationContract(**Name = "AddDouble"**)]

double Add(double arg1,double arg2);

}

## What is Known Types ?

By default, you can not use a subclass of a data contract class instead of its base class. You need to explicitly tell WCF about the subclass using the KnownTypeAttribute. For eg

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

[DataContract]

[**KnownType(typeof(SubClass))**]

class BaseClass

{...}

[DataContract]

class SubClass : BaseClass

{...}

## What is ServiceKnownType ?

Instead of using the KnownType attribute on the base data contract, you can apply the  
ServiceKnownType attribute on a specific operation on the service side. Then, only that  
operation (across all supporting services) can accept the known subclass.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

[OperationContract]

[**ServiceKnownType(typeof(SubClass))**]

void AddContact(BaseClass baseObject);

## How can we create and host a WCF service in IIS ?

Please read [Create RESTful WCF Service API Using GET & POST: Step By Step Guide](http://synvistech.com/blogs/create-restful-wcf-service-api-using-get-post-step-by-step-guide/)

## How to create a service contract and operation contract ? Can you give an example ?

We can create a contract using an interface by applying [ServiceContract] and [OperationContract] attributes on Interface and Methods respectively.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

[ServiceContract]

public interface IGreetingService

{

[OperationContract]

string GreetMe(string userName);

}

public class GreetingService : IGreetingService

{

public string GreetMe(string userName)

{

return string.Format("Welcome {0}", userName);

}

}

If we do not want to create interface then we can apply the attributes on a class itself.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

[ServiceContract]

public class GreetingService

{

[OperationContract]

public string GreetMe(string userName)

{

return string.Format("Welcome {0}", userName);

}

}

## Can you give an example of DataContract ?

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

[DataContract]

public enum Color

{

[EnumMember]

Red,

[EnumMember]

Green,

[EnumMember]

Blue

}

[DataContract]

public class Shape

{

[DataMember]

public string Name { get; set; }

[DataMember]

public Color FillColor { get; set; }

[DataMember]

public double Area { get; set; }

}

## What is MessageContract ? Can you give an example ?

WCF uses SOAP messages to transfer information from client to server and vice-versa. It converts data contract to SOAP messages. SOAP message contains Envelope, Header and Body. SOAP envelope contains name, namespace, header and body element. SOAP Header contains important information which are related to communication but not directly related to message. SOAP body contains information which is used by the target.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

SOAP Envelope = Name + Namespace + Header + Body

However there are some cases when you want to have control over the SOAP messages. You can achieve this using MessageContract.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

[MessageContract]

public class Shape

{

[MessageHeader]

public string ID;

[MessageBodyMember]

public string Name;

[MessageBodyMember]

public double Area;

}

In above example, ID will be added as header, Name and Area as body in SOAP envelope.

When you use MessageContract then you have control over the SOAP message. However some restrictions are imposed as below:

* You can have only one parameter for a service operation if you are using MessageContract.
* You can return either void or MessageContract type from service operation. Service operation can not return DataContract type.
* Service operation can accept and return only MessageContract type.

Some important points about MessageContract:

* You can mention the MessageHeader or MessageBodyMember to be signed or Encrypted using ProtectionLevel property.
* The order of the body elements are alphabetical by default. But you can control the order, using Order property in the MessageBody attribute.

## What is FaultContract ?

In most of the cases you would like to convey the details about any error which occurred at service end. By default, WCF exceptions do not reach client. However you can use FaultContract to send the exception details to client.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

[DataContract()]

public class CustomError

{

[DataMember()]

public string ErrorCode;

[DataMember()]

public string Title;

[DataMember()]

public string ErrorDetail;

}

[ServiceContract()]

public interface IGreetingService

{

[OperationContract()]

[FaultContract(typeof(CustomError))]

string Greet(string userName);

}

public class GreetingService : IGreetingService

{

public string Greet(string userName)

{

if (string.IsNullOrWhiteSpace(userName))

{

var exception = new CustomError()

{

ErrorCode = "401",

Title = "Null or empty",

ErrorDetail = "Null or empty user name has been provided"

};

throw new FaultException<CustomError>(exception, "Reason : Input error");

}

return string.Format("Welcome {0}", userName);

}

}

## How is the service instance created ? How can you manage or control WCF service instance creation ?

Client request can be served by using single service instance for all users, one service instance for one client, or one instance for one client request. You can control this behavior by using the technique called Instance Management in WCF.

There are three instance modes supported by WCF:

* **Per-Call**: Service instance is created for each client request. This Service instance is disposed after response is sent back to client.
* **Per-Session (default):** Service instance is created for each client. Same instance is used to serve all the requests from that client for a session. When a client creates a proxy to particular service, a service instance is created at server for that client only. When session starts, context is created and when it closes, context is terminated. This dedicated service instance will be used to serve all requests from that client for a session. This service instance is disposed when the session ends.
* **Singleton:** All client requests are served by the same single instance. When the service is hosted, it creates a service instance. This service instance is disposed when host shuts down.

You can configure instance mode using [ServiceBehavior] attribute as below:

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

[ServiceBehavior(**InstanceContextMode=InstanceContextMode.Single**)]

public class MyService:IGreetingService

## Can you control when the service instance is recycled ?

Yes, we can control when the service instance is recycled using the ReleaseInstanceMode property of theOperationBehavior attribute. You can control the lifespan of your WCF service. You can set the value ofReleaseInstanceMode property as one of the following:

* **RealeaseInstanceMode.*None***: No recycling behavior.
* **RealeaseInstanceMode.BeforeCall:**Recycle a service object before an operation is called.
* **RealeaseInstanceMode.AfterCall:**Recycle a service object after an operation is called.
* **RealeaseInstanceMode.BeforeAndAfterCall:**Recycle a service object both before and after an operation is called.

## Can you limit how many instances or sessions are created at the application level ?

Yes, you can limit how many instances or sessions are created at the application level. For this, you need to configure throttling behavior for the service in its configuration file. Some of these important properties are:

* **maxConcurrentCalls** limits the total number of calls that can currently be in progress across all service instances. The default is 16.
* **maxConcurrentInstances** limits the number of InstanceContext objects that execute at one time across a ServiceHost. The default is Int32.MaxValue.
* **maxConcurrentSessions** limits the number of sessions a ServiceHost object can accept. It is a positive integer that is 10 by default.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

<behaviors>

<serviceBehaviors>

<behavior name="ServiceBehavior">

<serviceThrottling **maxConcurrentCalls**="500" **maxConcurrentInstances** ="100" **maxConcurrentSessions** ="200"/>

</behavior>

</serviceBehaviors>

</behaviors>

## What are session modes in WCF ? How can you make a service as sessionful ?

ServiceContract attribute offers the property SessionMode which is used to specify the session mode. There are three session modes supported by WCF:

* **Session.Allowed(default):** Transport sessions are allowed, but not enforced. Service will behave as a per-session service only if the binding used maintains a transport-level session.
* **Session.Required:** Mandates the use of a transport-level session, but not necessarily an application-level session.
* **Session.NotAllowed:** Disallows the use of a transport-level session, which precludes an application-level session. Regardless of the service configuration, the service will always behave as a per-call service.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

[ServiceContract(**SessionMode = SessionMode.NotAllowed**)]

interface IMyContract

{...}

[ServiceBehavior(**InstanceContextMode = InstanceContextMode.PerCall**)]

class MyService : IMyContract

{...}

## What is the instance mode as a product of the binding, contract configuration, and service behavior ?

|  |  |  |  |
| --- | --- | --- | --- |
| **Binding** | **Session mode** | **Context mode** | **Instance mode** |
| Basic | Allowed/ NotAllowed | PerCall/ PerSession | PerCall |
| TCP, IPC | Allowed/ Required | PerCall | PerCall |
| TCP, IPC | Allowed/ Required | PerSession | PerSession |
| WS (no Message security, no reliability) | NotAllowed/ Allowed | PerCall/ PerSession | PerCall |
| WS (with Message security or reliability) | Allowed/ Required | PerSession | PerSession |
| WS (with Message security or reliability) | NotAllowed | PerCall/ PerSession | PerCall |

## What is MEP (Message Exception Pattern) in WCF ?

MEP describes the way in which Client and Server communicates. It describes how client and server would be exchanging messages to each other. There are three types of message exchange patterns:

* **Request- Replay (default):** When client makes a request to the WCF service, it waits to get response from service till receiveTimeout expires. If client does not get any response from the service before receiveTimeout expires, TimeoutException is thrown.
* **One-Way:** When client makes a request to the WCF service, it does not wait for reply from the service. Service does not send any response to the sender, even if any error occurs in the communication. It does not support **out** or **ref** parameters. It does not return value to an operation.
* **Duplex/Callback:** Client and service can sends messages to each other by using One-way or request-reply messaging. This MEP is supported by only bidirectional-capable bindings like as WS Dual, TCP and IPC bindings.To make a duplex contract, you must also define a callback contract and assign the typeof that callback contract to the CallbackContract property of your service contract’s ServiceContract attribute.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

public interface IMyDuplexServiceCallback

{

[OperationContract(IsOneWay = true)]

void Progress(string status);

}

[ServiceContract(CallbackContract = typeof(IMyDuplexServiceCallback))]

public interface IMyDuplexService

{

[OperationContract(IsOneWay = true)] *//One-Way*

void SaveData();

[OperationContract] *//Request-Reply.*

string GetData();

}

For Duplex MEP, you need to specify the one of the binding which supports bi-directional like wsDualHttpBinding as in below example:

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

<services>

<service name="MyWCFServices.DuplexService">

<endpoint address ="" binding="**wsDualHttpBinding**" con-tract="MyWCFServices.IDuplexService">

</endpoint>

</service>

</services>

You can configure MEP using **IsOneWay** property of **OperationContract** attribute as below:

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

[OperationContract(**IsOneWay = true**)]

## What is transaction and committed transaction in WCF ?

A transaction is a collection or group of one or more units of operation executed as a whole. It provides way to logically group multiple pieces of single work and execute them as a single unit. In addition, WCF allows client applications to create transactions and to propagate transactions across service boundaries.

A transaction that executes successfully and manages to transfer the system from the consistent state A to the consistent state B is called a committed transaction.

## What is Two-phase commit protocol in WCF ? Why is it needed ?

Consider for example client calling multiple service or service itself calling another service, this type of system are called as Distributed Service-oriented application. Now the questions arise that which service will begin the transaction? Which service will take responsibility of committing the transaction? How would one service know what the rest of the service feels about the transaction? Service could also be deployed in different machine and site. Any network failure or machine crash also increases the complexity for managing the transaction. This problem is resolved by using two phase protocol.

All the transactions in WCF complete using two phase commit protocol. It is the protocol which enables transactions in a distributed environment. This protocol mainly consist of two phases:

* **Prepare phase:** In this phase the client application performs the operations of a WCF service. WCF service determines whether the requested operation will be successful or not and notify the client about the same.
* **Commit Phase:** In the commit phase the client checks for the responses it got from the prepare phase and if all the responses are indicating that the operation can be carried out successfully the transaction is committed. If the response from any one of the operations indicates failure then the transaction will be rolled back. The actual operation on the service end will happen in the commit phase.

WCF service will have to send the notification of whether the operation will succeed or fail to the client application. It means that the One way operations can never support transactions. The operations that support transactions have to follow the Request-Response MEP. Also the applied binding should support WS-Atomic Transaction protocol like wsHttpBinding.

## What is Transaction Propagation in WCF ? Explain with example.

Suppose that there are two services CreditService and DebitService. CreditService has operation Credit(int accountId, double amount) and DebitService has operation Debit(int accountId, double amount). If you want to transfer amount from one account to another account, you need to call both the services. You also need to ensure that both the services should either succeed or fail together. You can achieve this by propagating the transaction of first service call to the second service call. Transaction Propagation is supported by WCF. You can propagate transaction across the service boundaries. It enables multiple services to participate in same transaction.

You can enable/disable transaction propagation using configuration as below:

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

<bindings>

<netTcpBinding>

<binding **transactionFlow="true"**></binding>

</netTcpBinding>

</bindings>

Above configuration ensures that transaction can be propagated. However it does not force the transaction propagation until you specify for particular operation. You need to enable transaction flow for the operations whom you want to be part of transaction as below:

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

[OperationContract]

**[TransactionFlow(TransactionFlowOption.Allowed)]**

bool Credit(int accountId, double amount);

[OperationContract]

**[TransactionFlow(TransactionFlowOption.Allowed)]**

bool Debit(int accountId, double amount);

Note:You can have single operation as which can do credit and debit. However I have separated as two for illustrating about transaction.

transactionFlow and TransactionFlowOption together enables the transaction flow for particular operation. If you enable only one of these two, transaction flow can not be enabled.

There are 3 possible values for TransactionFlowOption:

* **TransactionFlowOption.Mandatory:** specifies that this function can only be called within a transaction.
* **TransactionFlowOption.**Allowed: specifies that this operation can be called within a transaction but its not mandatory.
* **TransactionFlowOption.**NotAllowed: specifies that this operation can not be called within a transaction.

## How to create WCF transaction ?

There are some steps you need to follow to enable transaction for a WCF service as below:

**Step 1:** Decorate the operation contract with TransactionFlow attribute for enabling the transaction.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

<code>[OperationContract]

**[TransactionFlow(TransactionFlowOption.Allowed)]**

bool Debit(int accountId, double amount);

**Step 2:** Create the service class which implements the service contract and set the operation behavior withTransactionScopeRequired = true. This attribute is used to enable the service transaction when the client transaction is not available.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

**<code>[OperationBehavior(TransactionScopeRequired = true)]**

public bool Debit(int accountId, double amount)

{

*// Debit logic goes here*

}

**Step 3**: Enable transaction flow in configuration file.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

<bindings>

<wsHttpBinding>

<binding name="myTransactionBinding" **transactionFlow="true"** ></binding>

</wsHttpBinding>

</bindings>

## What is Restful service ?

* REST stands for Representational State Transfer.
* REST is an architectural style for building distributed applications. It involves building Resource Oriented Architecture (ROA) by definding resources that implement uniform interfaces using standard HTTP verbs (GET, POST, PUT, and DELETE), and that can be located/identified by a Uniform Resource Identifier (URI).

Any Service which follows the REST architecture style is called as RESTful service.

Characteristics of RESTful services:

* We can load the server information using web url in the browser.
* We can access/modify the server resource using Url.
* It allows the client, written in different language, to access or modify the resource in the server using URL.
* It uses the http protocol for its communication and it is stateless.
* It can transfer the data in XML,JSON,RSS,ATOM.

## How can you control if and when concurrent calls are allowed ? Or What is concurrency modes in WCF ? Why do we use it ?

We can use Concurrency Modes in WCF to control if concurrent calls to the context are allowed or not and if yes then when concurrent calls to the instance/context should be allowed.

There are three possible values for Concurrency Modes:

**ConcurrencyMode.Single(default)-** only one caller at a time is allowed.

WCF will provide automatic synchronization to the service context and disallow concurrent calls by associating the context containing the service instance with a synchronization lock. If there are multiple concurrent callers to the same context while the lock is locked, all the callers are placed in a queue. Once the context is unlocked waiting callers are allowed to lock the context in the order of queue. If a call times out while blocked, WCF will remove the caller from the queue and the client will get a TimeoutException.

**ConcurrencyMode.Multiple -** multiple callers at a time are allowed.

WCF will not synchronize access to the service context. It means that the service context is not associated with any synchronization lock. In this case, you must manually synchronize access to the service instance state using .NET locks such as Monitor or a WaitHandle-derived class.

**ConcurrencyMode.Reentrant -** multiple callers at a time are allowed only if it is reentrant.  
WCF associates the service context with a synchronization lock, so concurrent calls on the same instance are never allowed. However, if the reentrant service calls another service or a callback, and that call chain somehow winds its way back to the service instance that call is allowed to reenter the service instance. For example, suppose there are three services A, B and C. Service A calls service B, service B calls service C, and service C calls service A then service C is allowed to call service A because it is reentrant service.

You can set the concurrency mode for the service by setting the ConcurrencyMode property of ServiceBehavior attribute.

http://www.codeproject.com/images/minus.gif Collapse | [Copy Code](http://www.codeproject.com/Articles/759331/Interview-Questions-Answers-Windows-Communication)

[ServiceBehavior(InstanceContextMode = InstanceContextMode.Single, ConcurrencyMode = ConcurrencyMode.Multiple)]

Hope it helps you in understanding the basic concepts of WCF and answer related questions. Please comment your suggestion to make this article more useful. You can write your question, if any, as comment and I shall answer it.