**Windows Communication Service**

**1. What is mean by WCF?**

Windows Communication Foundation (Code named Indigo) is a programming platform and runtime system for building, configuring and deploying network-distributed services. It is the latest service oriented technology; Interoperability is the fundamental characteristics of WCF. It is unified programming model provided in .Net Framework 3.0. WCF is a combined feature of Web Service, Remoting, MSMQ and COM+. WCF provides a common platform for all .NET communication

**2. What is the difference between WCF and Web Service?**

|  |  |  |
| --- | --- | --- |
| **Features** | **Web Service** | **WCF** |
| Hosting | It can be hosted in IIS | It can be hosted in IIS, windows activation service, Self-hosting, Windows service |
| Programming | [WebService] attribute has to be added to the class | [ServiceContraact] attribute has to be added to the class |
| Model | [WebMethod] attribute represents the method exposed to client | [OperationContract] attribute represents the method exposed to client |
| Operation | One-way, Request- Response are the different operations supported in web service | One-Way, Request-Response, Duplex are different type of operations supported in WCF |
| XML | System.Xml.serialization name space is used for serialization | System.Runtime.Serialization namespace is used for serialization |
| Encoding | XML 1.0, MTOM(Message Transmission Optimization Mechanism), DIME, Custom | XML 1.0, MTOM, Binary, Custom |
| Transports | Can be accessed through HTTP, TCP, Custom | Can be accessed through HTTP, TCP, Named pipes, MSMQ,P2P, Custom |
| Protocols | Security | Security, Reliable messaging, Transactions |

**3. What is mean by Endpoint?**

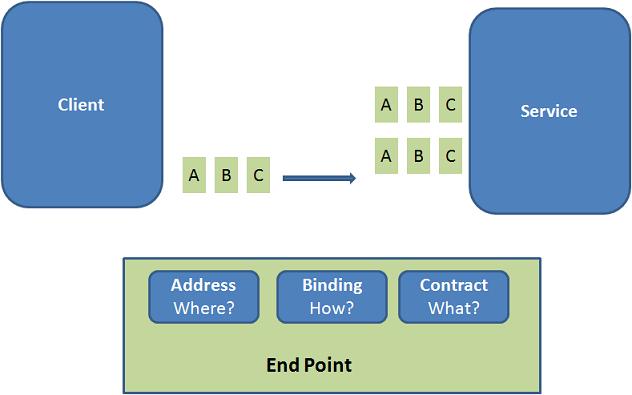
WCF Services exposes a collection of Endpoints, each Endpoint is a portal for communicating with the world. Endpoint is used to identify the service; it is more are like an address for your home. Each endpoint is used to identify the specific service. One service can have multiple endpoints. Client application will use this endpoint for communication with service. All the WCF communications are take place through end point. End point consists of three components. Address, Binding and Contract

Address - Basically URL, specifies **where** this WCF service is hosted .Client will use this url to connect to the service. e.g

http://localhost:8090/MyService/SimpleCalculator.svc

Binding - Binding will describes **how** client will communicate with service. There are different types of protocols available for the WCF to communicate with the Client.e.g: BasicHttpBinding, WSHttpBinding etc

Contract - Contract specifies the **what** are the collection of operations that are exposed to the outside world. Usually name of the Interface will be mentioned in the Contract, so the client application will be aware of the operations which are exposed to the client.



**4. What is mean by Service contract and its use?**

Service contract describes the operation (functionality) that service provides. A Service can have more than one service contract but it should have at least one Service contract.

Service Contract can be define using [ServiceContract] and [OperationContract] attribute. [ServiceContract] attribute is similar to the [WebServcie] attribute in the WebService and [OpeartionContract] is similar to the [WebMethod] in WebService. Attributes mentioned in the service contract are optional.

[ServiceContract(SessionMode=SessionMode.Allowed,

ProtectionLevel=System.Net.Security.ProtectionLevel.EncryptAndSign)]

public interface IMathService

{

[OperationContract]

int Add(int a, int b);

[OperationContract]

int Subtract(int a, int b);

// TODO: Add your service operations here

}

**5. What is mean by Operational contract and its use?**

Operation contract describers the client-callable operations (functions) exposed as service. Only the functions which are decorated with OperationContract will be exposed to outside world. [OpeartionContract] is similar to the [WebMethod] attribute in WebService implementation.

In the below example only Add() and Subtract() methods are decorated with [OperationContract] and So only these two method will be exposed and used by client application.

[ServiceContract()]

public interface IMathService

{

[OperationContract]

int Add(int num1, int num2);

[OperationContract]

int Subtract(int num1, int num2);

int Multiply(int num1, int num2);

int Divide(int num1, int num2);

}

public class MathService : IMathService

{

public int Add(int num1, int num2)

{

return num1 + num2;

}

public int Subtract(int num1, int num2)

{

return num1 - num2;

}

public int Multiply(int num1, int num2)

{

return num1 \* num2;

}

public int Divide(int num1, int num2)

{

return num1 / num2;

}

}

**6. What is mean by DataContract and its use?**

A data contract is a formal agreement between a service and a client that abstractly describes the data to be exchanged. Data contract can be explicit or implicit. Simple type such as int, string etc has an implicit data contract. User defined object are explicit or Complex type, for which we need to define a Data contract using [DataContract] and [DataMember] attribute.

A data contract can be defined as follows:

* It describes the external format of data passed to and from service operations
* It defines the structure and types of data exchanged in service messages
* It maps a CLR type to an XML Schema
* It defines how data types are serialized and deserialized

Example: Create user defined data type called Employee. This data type should be identified for serialization and deserialization by mentioning with [DataContract] and [DataMember] attribute.

[ServiceContract]

public interface IEmployeeService

{

[OperationContract]

Employee GetEmployeeDetails(int EmpId);

}

[DataContract]

public class Employee

{

private string m\_Name;

private int m\_Age;

private int m\_Salary;

private string m\_Designation;

private string m\_Manager;

[DataMember]

public string Name

{

get { return m\_Name; }

set { m\_Name = value; }

}

[DataMember]

public int Age

{

get { return m\_Age; }

set { m\_Age = value; }

}

[DataMember]

public int Salary

{

get { return m\_Salary; }

set { m\_Salary = value; }

}

[DataMember]

public string Designation

{

get { return m\_Designation; }

set { m\_Designation = value; }

}

[DataMember]

public string Manager

{

get { return m\_Manager; }

set { m\_Manager = value; }

}

}

**7. What is mean by FaultContract?**

Fault contract is used to describe the custom exception thrown by service to client. Only if exception class decorated with Fault contract, it can view in client application.

Service that we develop might get error in come case. This error should be reported to the client in proper manner. Basically when we develop managed application or service, we will handle the exception using try- catch block. But these exceptions handlings are technology specific.

In order to support interoperability and client will also be interested not on how and where cause the error, what went wrong?

By default when we throw any exception from service, it will not reach the client side. WCF provides the option to handle and convey the error message to client from service using SOAP Fault contract.

Suppose the service I consumed is not working in the client application. I want to know the real cause of the problem. How I can know the error? For this we are having Fault Contract. Fault Contract provides documented view for error accorded in the service to client. This help as to easy identity the what error has accord.

Example:

You can also create your own Custom type and send the error information to the client using *FaultContract*. These are the steps to be followed to create the fault contract.

* Define a type using the data contract and specify the fields you want to return.
* Decorate the service operation with the FaultContract attribute and specify the type name.
* Raise the exception from the service by creating an instance and assigning properties of the custom exception.

**Step 1**: Defining the type using Data Contract

[DataContract()]

public class CustomException

{

[DataMember()]

public string Title;

[DataMember()]

public string ExceptionMessage;

[DataMember()]

public string InnerException;

[DataMember()]

public string StackTrace;

}

**Step 2**: Decorate the service operation with the FaultContract

[ServiceContract()]

public interface ISimpleCalculator

{

[OperationContract()]

[FaultContract(typeof(CustomException))]

int Add(int num1, int num2);

}

**Step 3**: Raise the exception from the service

public int Add(int num1, int num2)

{

//Do something

CustomException ex = new CustomException();

ex.Title = "Error Funtion:Add()";

ex.ExceptionMessage = "Error occur while doing add function.";

ex.InnerException = "Inner exception message from serice";

ex.StackTrace = "Stack Trace message from service.";

throw new FaultException(ex, "Reason: Testing the Fault contract");

}

**8. What is mean by Hosting WCF service?**

WCF service cannot exist on its own; it has to be hosted in windows process called as host process. Single host process can host multiple services and same service type can be hosted in multiple host process. There are mainly four different way of hosting the WCF service.

* IIS hosting
* Self hosting
* Windows Activation Service
* Windows Service

**9. What are different types of hosting available in WCF?**

WCF service can be hosted in four different ways

* IIS hosting
* Self hosting
* Windows Activation Service
* Windows Service

Multiple hosting and protocols supported by WCF.Microsoft has introduced the WCF concept in order to make distributed application development and deployment simple.

|  |  |
| --- | --- |
| **Hosting Environment** | **Supported protocol** |
| 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 |
| Web server IIS6 | http, wshttp |
| Web server IIS7 - Windows Process Activation Service (WAS) | HTTP,net.tcp,net.pipe,net.msmq |

**10. How will you host the WCF using Windows application?**

WCF service can be hosted in windows application using **"ServiceHost host"** class.

Below example describes the hosting of the CalculatorService in windows application

static void Main(string[] args)

{

//Create a URI to serve as the base address

Uri httpUrl = new Uri("http://localhost:8090/MyService/SimpleCalculator");

//Create ServiceHost

ServiceHost host

= new ServiceHost(typeof(MyCalculatorService.SimpleCalculator), httpUrl);

//Add a service endpoint

host.AddServiceEndpoint(typeof(MyCalculatorService.ISimpleCalculator)

, new WSHttpBinding(), "");

//Enable metadata exchange

ServiceMetadataBehavior smb = new ServiceMetadataBehavior();

smb.HttpGetEnabled = true;

host.Description.Behaviors.Add(smb);

//Start the Service

host.Open();

Console.WriteLine("Service is host at " + DateTime.Now.ToString());

Console.WriteLine("Host is running... Press key to stop");

Console.ReadLine();

}

**11. What are different types of binding supported by WCF?**

* BasicHttpBinding
* WSHttpBinding
* WSDualHttpBinding
* WSFederationHttpBinding
* NetTcpBinding
* NetNamedPipeBinding
* NetMsmqBinding
* NetPeerTcpBinding
* CustomBinding

**12. What is used of different types of binding in WCF?**

The real power of wcf resides in binding; each binding are used in different communication scenario. Example BasicHttpBinding are used to communicate outside the firewall or network. Whereas NetTcpbinding are used to communicate between the systems present inside the Local Area Network. The performance for the communication will be increased if we use the NetTcpBinding inside the network. If we use the BasicHttpBinding inside the network, then the performance will degrade. So selection of the right binding of the right communication is very important.

**13. What is mean by Message exchange endpoint?**

WCF provides rich infrastructure for Exporting, Publishing, retrieving and Importing the metadata (information about the service). WCF uses the Metadata to describe how to interact with the service endpoint. This metadata information of the service is exposed using default endpoint called Message exchange endpoint.

**14. What is use of MEX?**

Message Exchange Endpoint is used by client application to create the proxy of the service using metadata exposed by MEX.

**15. What are different types of bindings supported by MEX?**

mexHttpBinding, mexHttpsBinding, mexNamedPipesBinding, mexTcpBinding

**16. What is mean by Proxy creation?**

The proxy is a class that exposes a single CLR interface representing the service contract. If the service supports several contracts, the client needs a proxy per contract type. The proxy provides the same operations as service's contract, but also has additional methods for managing the proxy life cycle and the connection to the service.

**17. What are different types of Proxy creation?**

There are different methods available to create the proxy class. These methods are explained by creating the proxy class for the MathService with all some mathematical operation exposed as service.

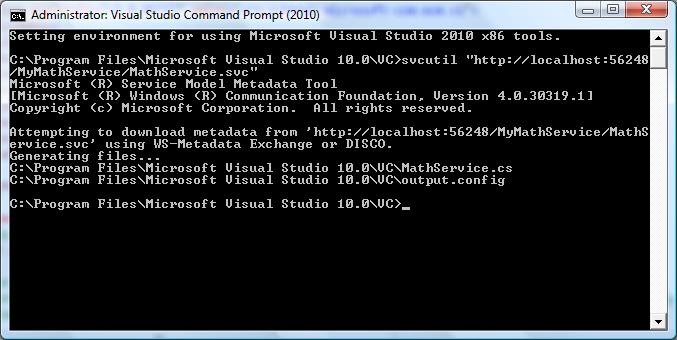
**Method 1:**

Using service utility command in visual studio command prompt, we can generate the proxy

Example:

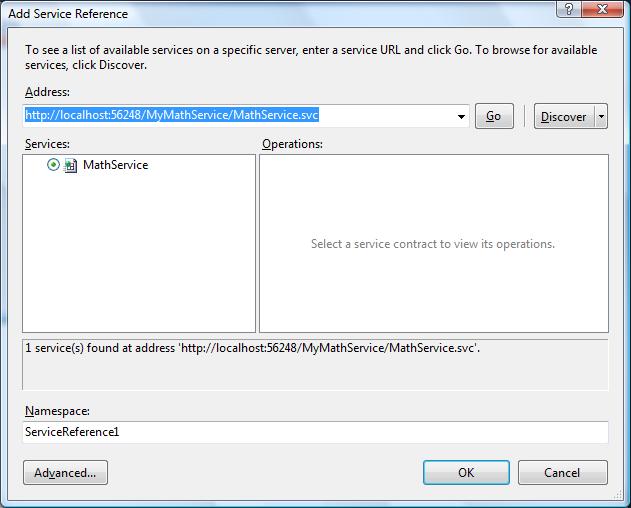
Svcutil "http://localhost:56248/MyMathService/MathService.svc"

Proxy files and its configuration files are generated in default directory of the command prompt MathService.cs and output.config



**Method 2:** Add Service Reference to the project file

Right click the project file and select "Add Service Reference". Specify the service endpoint address and click "Go" and Math service will be identified and listed. Click OK to create proxy class inside the project application.



**Method 3:**

By Inheriting ClientBase class

Below example describes the creating the proxy class using Contract information and Channel property of the ClientBase

[ServiceContract]

public interface IMathService

{

[OperationContract]

int Add(int a, int b);

[OperationContract]

int Subtract(int a, int b);

}

public class MyMathServiceProxy : ClientBase

{

public int Add(int a, int b)

{

return this.Channel.Add(a, b);

}

public int Subtract(int a, int b)

{

return this.Channel.Subtract(a, b);

}

}

**18. Can we able to call the WCF service without creating Proxy class?**

Yes, using "ChannelFactory" we can call the service operation

Example:

private ChannelFactory cf;

private IHelloChannel client;

private void btnSayIt\_Click(object sender, EventArgs e)

{

cf = new ChannelFactory("\*");

client = cf.CreateChannel();

MessageBox.Show(client.SayHello(txtName.Text));

client.Close();

}

**19. What are different types of Behaviors available in WCF?**

WCF defines two types of service-side behaviors

* ServiceBehaviors - It is used to configure service behavior and it will affect all endpoins of the service

Example: Below service behavior will enable the exposing of service metadata

<serviceBehaviors>

<behavior name="ServiceBehavior">

<serviceMetadata httpGetEnabled="true"/>

</behavior>

</serviceBehaviors>

* EndpointBehaviors - It is used to configure specific endpoint behavior

**20. How will you implement function overloading in WCF?**

By default WCF will not provide option to overload a function. But method overloading can be achieved using "Name" attribute of the OperationContract.

Below examples describes implementation of function overloading.

[ServiceContract]

public interface IMathService

{

[OperationContract]

int Add(int a, int b);

[OperationContract(Name = "AddDecimal")]

decimal Add(decimal a, decimal b);

}

If we don’t specify the "Name" attribute for the OperationContract. Following error will be displayed while the running the service

**Server Error in '/MyMathService' Application.**

*Cannot have two operations in the same contract with the same name, methods Add and Add in type IMathService violate this rule. You can change the name of one of the operations by changing the method name or by using the Name property of OperationContractAttribute.*

**21. Will WCF service returns the dataset?**

Yes, dataset are serializable object, so WCF service will be able to return the result in dataset format.

**22. What is the use of ServiceKnownType attribute?**

ServiceKnownType attribute is used to describe the inherited class of the DataContract of the service. Consider a base class and derived class which are decorated with DataContract, when we use base class inside the wcf service, service will be it will able to understand and serialize only the Base class not the derived class. In order to service to know the derived class has to be serialized we need to mention the class name using ServiceKnowType attribute.

Example:

[DataContract]

class Contact

{...}

[DataContract]

class Customer : Contact

{...}

[DataContract]

class Person : Contact

{...}

[ServiceContract]

[ServiceKnownType(typeof(Customer))]

[ServiceKnownType(typeof(Person))]

interface IContactManager

{

[OperationContract]

Contact[] GetContacts( );

}

**23. If you use [DataMember] for the private property of the class, will it be exposed to the client application?**

No, only public member of the DataContract, which is decorated with DataMember attribute will be exposed to the client.

**24. How will you use Enum in WCF?**

Enum can be mention in WCF using *[EnumMember]* attribute

[DataContract]

enum ContactType

{

[EnumMember]

Customer,

[EnumMember]

Vendor,

//Will not be part of data contract

Partner

}

**25. What are different levels of service Instance can be managed in WCF?**

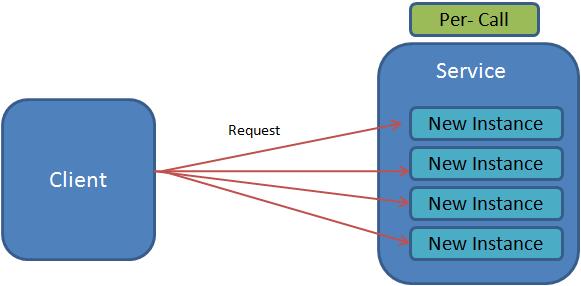
Instance management refers to the way a service handles a request from a client. Instance management is set of techniques WCF uses to bind client request to service instance, governing which service instance handles which client request.

Basically there are three instance modes in WCF:

* Per-Call instance mode
* Per-Session instance mode
* Singleton Instance Mode

**26. What is the mean by Per-Call session?**

When WCF service is configured for Per-Call instance mode, Service instance will be created for each client request. This Service instance will be disposed after response is sent back to client.



Example :

[ServiceContract()]

public interface IMyService

{

[OperationContract]

int MyMethod();

}

[ServiceBehavior(InstanceContextMode = InstanceContextMode.PerCall)]

public class MyService : IMyService

{

public int MyMethod()

{

...

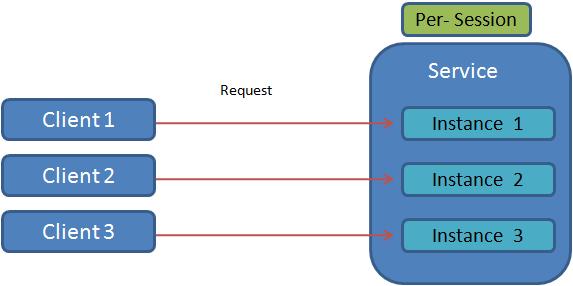
}

}

**27. What is mean by Per-Session instance?**

When WCF service is configured for Per-Session instance mode, logical session between client and service will be maintained. When the client creates new proxy to particular service instance, a dedicated service instance will be provided to the client. It is independent of all other instance.

Following diagram represent the process of handling the request from client using Per-Session instance mode.



Example :

[ServiceContract()]

public interface IMyService

{

[OperationContract]

int MyMethod();

}

[ServiceBehavior(InstanceContextMode = InstanceContextMode.PerSession )]

public class MyService : IMyService

{

public int MyMethod()

{

...

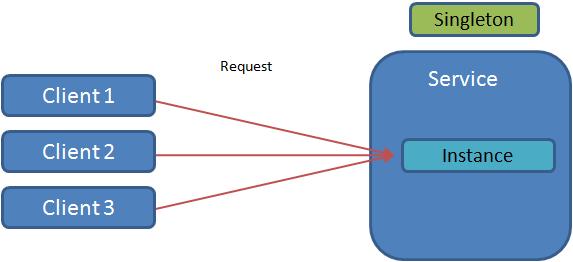
}

}

**28. What is mean by Singleton instance?**

When WCF service is configured for Singleton instance mode, all clients are independently connected to the same single instance. This singleton instance will be created when service is hosted and, it is disposed when host shuts down.

Following diagram represent the process of handling the request from client using Singleton instance mode.



Example:

[ServiceBehavior(InstanceContextMode = InstanceContextMode.Single )]

public class MyService : IMyService

{

public int MyMethod()

{

...

}

}

**29. What is the default Instance Management supported by WCF?**

Per-Session instance management

**30. What are the different ways to deactivate the service Instance?**

When service is configured to use Per-Session instance, WCF provides the option of disposing and recreating the instance within same session.

*ReleaseInstanceMode* property of the OberationalBehavior attribute used to control the instance in relation to the method call.

Followings are the list Release mode available in the *ReleaseInstanceMode*

* RealeaseInstanceMode.None
* RealeaseInstanceMode.BeforeCall
* RealeaseInstanceMode.AfterCall
* RealeaseInstanceMode.BeforeAndAfterCall

Example:

[ServiceContract()]

public interface ISimpleCalculator

{

[OperationContract()]

int Add(int num1, int num2);

}

[OperationBehavior(ReleaseInstanceMode=ReleaseInstanceMode.BeforeCall]

public int Add(int num1, int num2)

{

return num1 + num2;

}

**31. What is mean by Durable service?**

Durable services are WCF services that persist service state information even after service host is restarted or Client. It means that durable services have the capability to restore their own state when they are recycled. It can use data store like SQL database for maintain instance state. It is new feature in .Net 3.5

**32. How will you implement the durable service?**

Durable service can be created using following steps

1. Add "DurableService" attribute to the service class

[DurableService()]

public class SimpleCalculator : ISimpleCalculator

{

}

1. Use "wsHttpContextBinding" binding for the endpoint address

  <endpoint address="" binding="wsHttpContextBinding"

          bindingConfiguration="browConfig" contract="ISimpleCalculator">

1. Add <persistenceProvider> tag in web.config, it is used to configure the persistence provider

<persistenceProvider

  type="System.ServiceModel.Persistence.SqlPersistenceProviderFactory,

       System.WorkflowServices, Version=3.5.0.0, Culture=neutral,

        PublicKeyToken=31bf3856ad364e35" connectionStringName="DurableServiceStore"

                            persistenceOperationTimeout="00:00:10"

                            lockTimeout="00:01:00"

                            serializeAsText="true"/>

**33. What is mean by Throttling in WCF?**

WCF throttling provides some properties that you can use to limit how many instances or sessions are created at the application level. Performance of the WCF service can be improved by creating proper instance.

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

**34. Will the Callback operation is supported in WCF?**

Yes, Service can also call the functions located at the client side.

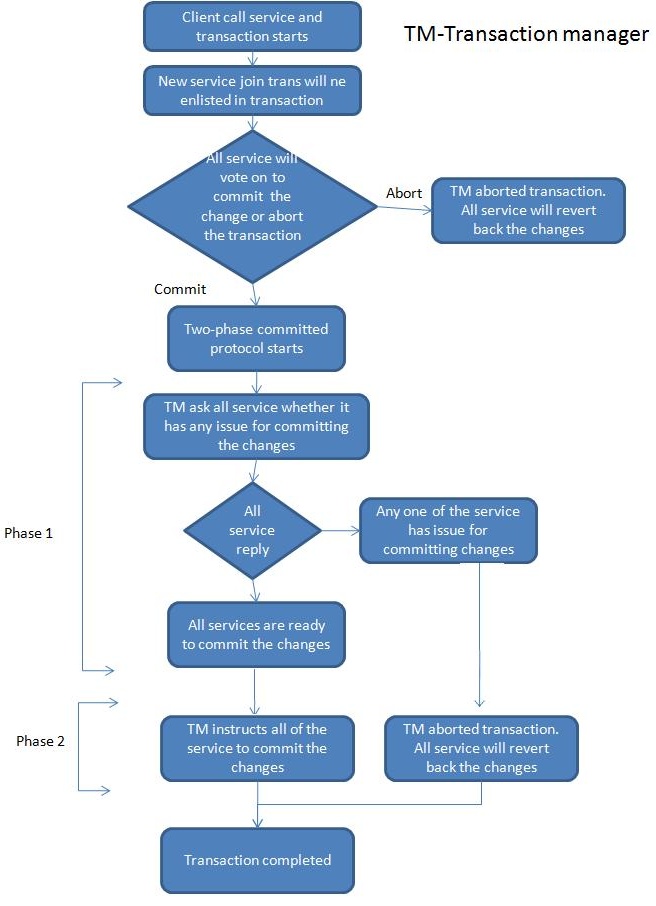
**35. How will you implement Call back service in WCF?**

Call back service can be implemented using *"wsDualHttpBinding"* binding and *CallbackContract* property in the*ServiceContract* attribute.

**36. What are mean by Two-phase committed protocol?**

Consider the scenario where I am having single client which use single service for communication and interacting with single database. In which service starts and manage the transaction, now it will be easy for the service to manage the transaction.

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.



In order to overcome these situations, WCF come up with distributed transaction using two way committed protocol and dedicated transaction manager.

Transaction Manager is the third party for the service that will manage the transaction using two phase committed protocol.

**37. What are different types of Transaction protocol?**

There are three different types of transaction protocol

1. Lightweight protocol
   * This protocol is used to manage the transaction within same Application Domain
   * Best performance compare to other
2. OleTx protocol
   * This protocol is used to manage the transaction in an intranet and in a windows environment
3. WS-Atomic Transaction (WSAT) protocol
   * It can propagate the transaction across the firewalls
   * Primarily used in Internet with multiple transaction managers are involved.

**38. What are different types of Transaction manager?**

* Lightweight Transaction Manager (LTM)
* Kernel Transaction Manager (KTM)
* Distributed Transaction Coordinator (DTC)

**39. What is the use of Lightweight Transaction Manager (LTM)?**

LTM can only manage a local transaction; that is, a transaction inside a single app domain. The LTM uses the lightweight transaction protocol to manage the two-phase commit protocol.

**40. What is the use of Kernel Transaction Manager (KTM)?**

The KTM can be used to manage transactional kernel resource managers (KRM) on Windows Vista, specifically the transactional filesystem (TXF) and the transactional registry (TXR). Transaction can involve at most one service, as long as that service does not propagate the transaction to other services.

**41. What is the use of Distributed Transaction Coordinator (DTC)?**

The DTC is the transaction manager used when transactions flow across the service boundary. The DTC is a system service available by default on every machine running WCF. Each DTC will communicate with other DTC in different machine to maintain the transaction across the network.

**42. What are different levels of Currency mode supported in WCF?**

Concurrent access to the service instance is governed by the ConcurrencyMode property of the ServiceBehavior attribute. There are three different types of currency mode

* Single - Only one caller is allowed to access the service instance
* Multiple - Concurrent calls are allowed on the service instance
* Reentrant - concurrent calls on the same instance are never allowed but the reentrant service calls out to another service are allowed.

**43. What is the use of dispatcher in WCF?**

Dispatcher will receive the message from channel and converts the message to a stack frame and calls the service instance for processing.

**44. What is Volatile Queued Communication?**

In queued communication, the client communicates to the service using a queue. More precisely, the client sends messages to a queue. The service receives messages from the queue. The service and client therefore, do not have to be running at the same time to communicate using a queue.

When you send a message with no assurances, MSMQ only makes a best effort to deliver the message, unlike with Exactly Once assurances where MSMQ ensures that the message gets delivered or, if it cannot be delivered, lets you know that the message cannot be delivered.

In certain scenarios, you may want to send a volatile message with no assurances over a queue, when timely delivery is more important than losing messages.