"Windows Communication Foundation"

Ajinkya D. Machale

Department of Computer Science and Engineering, College of Engineering

Third Year Engineering Student

DOT NET provides much facility to application developer for developing various applications. Using .NET we can develop many kind of application such as web based, windows based or it may be console based application. Let us consider .NET architecture, using that application developer develop many distributed application. It includes web services, .Net Remoting, Message Queuing (MSMQ), COM Services. All these services have their own development environment. As per client requirement application developer can develop application in any kind of services which having different environment to develop.

So it was big challenge for Microsoft to maintain functionality of .NET i.e. interoeratability. That's why Microsoft includes new unified communication framework of window in .NET framework 3.0 i.e. **Windows Communication Foundation (WCF)** which unify all existing communication services into one communication model. Hence now it becomes default choice for connecting application. All services come under one umbrella. So application developer became much flexible with .NET by developing code for any one of service and merges it with WCF. Microsoft maintained Dot Net functionality in upcoming versions like 3.5,4 and enhanced more.

Basically WCF is unified programming model for building service oriented application. WCF is single model for development of distributed application. Services oriented means performing task that client application can communicate with message to call web service. The client passes the message we call it as request which consist Extensible markup language (XML). It returns response message get back to client. It also consists of XML. For communication purpose WCF uses HTTP & TCP protocol. Real life example is Bank transaction which uses web services.

Windows Communication Foundation (WCF) is Microsoft's unified programming model for building service-oriented applications. It enables application developers to build secure, reliable, transacted solutions that integrate across platforms and interoperate with existing investments.

We shall discuss in the forthcoming paragraph what is new in WCF 4.5

1) Task-based Async Support

By default, Add Service Reference generates Task-returning async service operation methods. This is done for both synchronous and asynchronous methods. This allows you to call the service operations asynchronously using the new task based async programming model. When you call the generated proxy method, WCF constructs a Task object to represent the asynchronous operation and returns that Task to you. The Task completes when the operation completes. When implementing an async operation you can implement it as a task-based async operation.

2) Contract-First Development

WCF now has support for contract-first development. The svcutl.exe has a ServiceContract switch which allows you to generate service and data contracts from a Web Service Description Language (WSDL) document.

3) Simplified Generated Configuration Files

When you add a service reference in Visual Studio or use the SvcUtil.exe tool, a client configuration file is generated. In previous versions of WCF these configuration files contained the value of every binding property even if its value is the default value. In WCF 4.5 the generated configuration files contain only those binding properties that are set to a non-default value.

4) WCF Configuration Validation:

As part of the build process within Visual Studio, WCF configuration files are now validated for attributes defined within the project. A list of validation errors or warnings is displayed in Visual Studio if the validation fails.

5) Generating a Single WSDL Document

Some third-party WSDL processing stacks are not able to process WSDL documents that have dependencies on other documents through a xsd: import. WCF now allows you to specify that all WSDL information be returned in a single document. To request a single WSDL document append "? SingleWSDL" to the URI when requesting metadata from the service.

6) Configuring WCF Services in Code

WCF allows developers to configure services using configuration files or code. Configuration files are useful when a service needs to be configured after being deployed. When using configuration files, an IT professional only needs to update the configuration file, no recompilation is required. Configuration files, however, can be complex and difficult to maintain. There is no support for debugging configuration files and configuration elements are referenced by names which makes authoring configuration files error-prone and difficult. WCF also allows you to configure services in code. In earlier versions of WCF (4.0 and earlier) configuring services in code was easy in self-hosted scenarios, the ServiceHostclass allowed you to configure endpoints and behaviors prior to calling ServiceHost.Open. In web hosted scenarios, however, you don't have access to the ServiceHost class. To configure a web hosted service you were required to create aServiceHostFactory that created the ServiceHost and performed any needed configuration. Starting with .NET 4.5, WCF provides an easier way to configure both self-hosted and web hosted services in code.

Why WCF?

It is the latest service oriented technology. Interoperability is the fundamental characteristics of WCF Distributed application development supports. It is the next version of several existing product – ASMX, .NET Remoting, Enterprise service, Messaging.

WCF Fundamentals

WCF is a runtime and a set of APIs for creating systems that send messages between services and clients. The same infrastructure and APIs are used to create applications that communicate with other applications on the same computer system or on a system that resides in another company and is accessed over the Internet.

Messaging and Endpoints:

WCF is based on the notion of message-based communication, and anything that can be modeled as a message (for example, an HTTP request or a Message Queuing (also known as MSMQ) message) can be

represented in a uniform way in the programming model. This enables a unified API across different transport mechanisms.

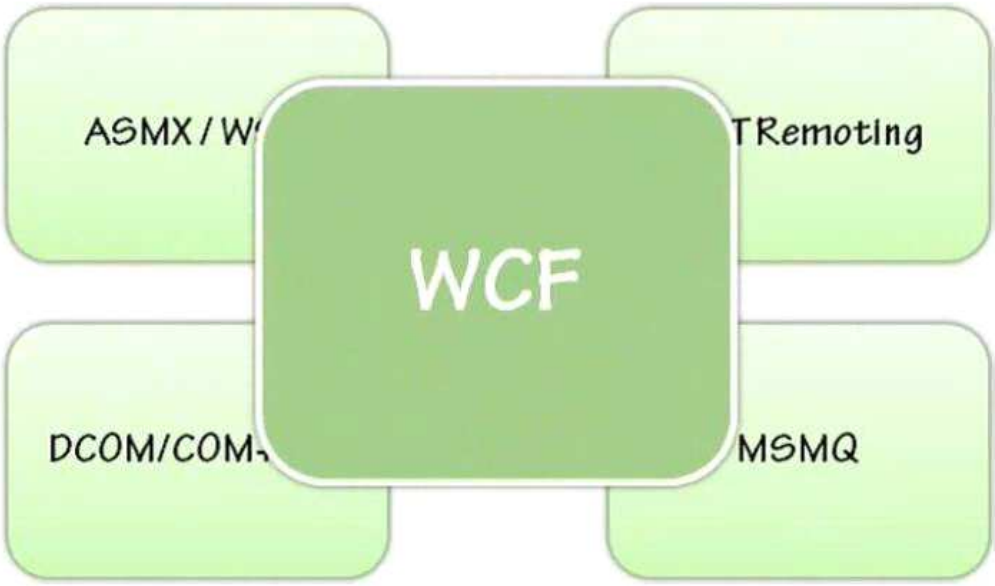
The model distinguishes between *clients*, which are applications that initiate communication, and *services*, which are applications that wait for clients to communicate with them and respond to that communication. A single application can act as both a client and a service. Messages are sent between endpoints. *Endpoints* are places where messages are sent or received (or both), and they define all the information required for the message exchange. A service exposes one or more application endpoints (as well as zero or more infrastructure endpoints), and the client generates an endpoint that is compatible with one of the service's endpoints.

Communication Protocols

One required element of the communication stack is the *transport protocol*. Messages can be sent over intranets and the Internet using common transports, such as HTTP and TCP. Other transports are included that support communication with Message Queuing applications and nodes on a Peer Networking mesh. More transport mechanisms can be added using the built-in extension points of WCF.

Message Patterns

WCF supports several messaging patterns, including request-reply, one-way, and duplex communication. Different transports support different messaging patterns, and thus affect the types of interactions that they support. The WCF APIs and runtime also help you to send messages securely and reliably.



Conceptual Representation of WCF

References: (1) Microsoft Developer Network (https://msdn.micrsoft.com)

(2) C# book (www.c-sharpcorner.com/ebooks)