Fundamentals of WCF

Steven Bucksbaum, March 24, 2011

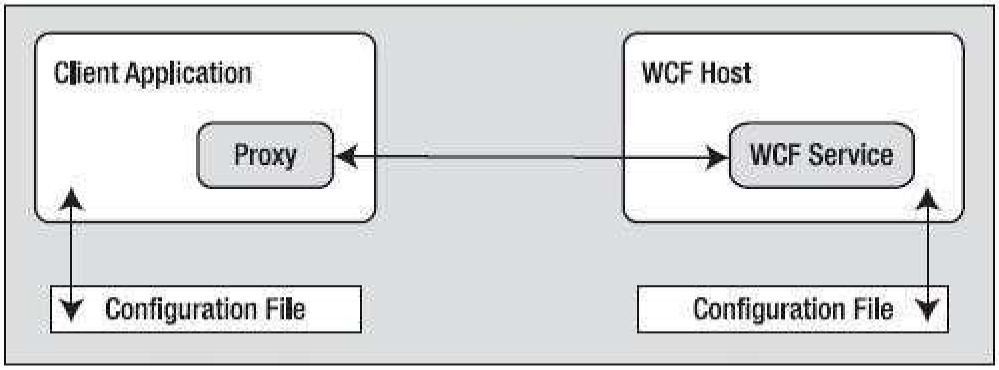
Fundamentals of Windows Communication Foundation (WCF)

Overview of WCF Features

The WCF is a unified messaging platform. The WCF supports TCP (sockets), HTTP, UDP, or MSMQ.. Interoperability and integration of diverse APIs are only two (very important) aspects of WCF. In addition, WCF provides a rich software fabric that complements the remoting technologies it exposes. Consider the following list of major WCF features:

1. Support for strongly typed *as well as* untyped messages. This approach allows .NET applications to share custom types efficiently
2. Support for several *bindings* (raw HTTP, TCP, MSMQ, and named pipes)
3. Support for session like state management techniques, as well as support for one-way stateless messages

Typical WCF Application:



Purpose of WCF

Web services are just one technology that you can use to create distributed applications for Windows. Others already mentioned include Enterprise Services and .NET Framework Remoting. Another example is Microsoft Message Queue (MSMQ). If you are building a distributed application for Windows, which technology should you use, and how difficult would it be to switch later if you need to? The purpose of WCF is to provide a unified programming model for many of these technologies, enabling you to build applications that are as independent as possible from the underlying mechanism being used to connect services and applications together (note that WCF applies as much to services operating in non-Web environments as it does to the World Wide Web)

The ABCs Of WCF

Hosts and clients communicate with each other by agreeing on the ABCs, a friendly mnemonic for remembering the core building blocks of a WCF application, specifically address, binding, and contract

Address : The location of the service
Binding : WCF ships with a number of different bindings the specify network protocols
Contract : A description of each method exposed from the WCF service.

Contract

The notion of a *contract* is the key to building a WCF service. While not mandatory, the vast majority of your WCF applications will begin by defining a set of .NET interface types that are used to represent the set of members a given WCF type will support. Specifically, interfaces that represent a WCF contract are termed *service contracts*. The classes (or structures) that implement them are termed *service types*.

WCF service contracts are adorned with various attributes, the most common of which are defined in the System.ServiceModel namespace. When the members of a service contract contain only simple data types (such as numerical data, Booleans, and string data) you can build a complete WCF service using nothing more than the [ServiceContract] and [OperationContract] attributes. However, if your members expose custom types, you will need to make use of types in the System.Runtime.Serialization namespace of the System.Runtime.Serialization.dll assembly. Here you will find additional attributes (such as [DataMember] and [DataContract]) to fine-tune the process of defining your interface types.

Metadata

Client applications require access to the metadata of a service so that they can determine the operations the service implements