WCFConceptsLeahCahill

by Leah Cahill

FILE	2254_LEAH_CAHILL_WCFCONCEPTSLEAHCAHILL_94244_558517049.DOCX (878.38K)		
TIME SUBMITTED	22-JAN-2018 10:25AM (UTC+0000)	WORD COUNT	3229
SUBMISSION ID	82192898	CHARACTER COUNT	19360

2017

WCF Concepts

DISTRIBUTED ENTERPRISE APPLICATIONS

LEAH CAHILL (P140092) – BSC WEB DEV

Contents

Introduction to Windows Communication Foundation 2
Hosting a WCF Service 2
The main classes and the use of the configuration file of WCF services. 2
Self-hosting a WCF service 3
Hosting a WCF service in a Windows service (using transmission control protocol) 3
Hosting a WCF Service in IIS 3
ABCs of WCF 4
The principles and process of metadata exchange in WCF. 5
Main Contract Types 6
Supported exchanged patterns in WCF. 8
Interoperability in WCF 9
References 10

Figure 1 Example code of defining a data contract 6
Figure 2 Example code of defining a service contract 6
Figure 3 Example code of defining an Operation Contract within a Service contract 7
Figure 4 Fault contract general exception example 7

Introduction to Windows Communication Foundation

Windows Communication Foundation (WCF) is a framework for building business and service-oriented systems for Microsoft Windows, WCF was released in 2006 as part of .NET 3.0. Framework. In WCF, the user can host services and data is sent as asynchronous messages from one service endpoint to another; an endpoint consists of an address, a binding and a contract.

WCF is an example of being part of a service orientated system, these systems use inter-process communications technologies. These technologies are required within the computing operating system so data between the systems can be shared; allowing multiple data to be shared and processed concurrently. The first Microsoft model of this kind was the Component Object Model (COM), that was designed to enable components to communicate with the applications that were running on that same computer [1].

The design of COM meant that it didn't particularly work well with components that were remote. DCOM (Distributed Component Object Model) was then released, and was followed by the development of COM+ which allowed the computers components to communicate over a network, it provided additional capabilities and integrated with Microsoft Transaction Server [2].

The .NET Framework was then developed which was an improvement upon the features of COM+ and it resolved issues that COM/DCOM encountered; it also enabled people to use new ways to build networked components and included features such as interoperability support. You can build a web service application using .NET framework on Windows Microsoft visual studios, this runs on the website through the user's web browser rather than the user's computer.

It is important that the software architecture of these service oriented systems is platform independent and isolated, this ensures that changes within one service would not affect the performance or task of another running service. This allows the systems to become scalable and flexible to change without interference [3].

Hosting a WCF Service

When it comes to hosting a WCF service, there are options such as self-hosting in a .NET application, using internet information services or hosting in a Windows service. The options of hosting a WCF service are discussed within this section of the report.

The main classes and the use of the configuration file of WCF services.

There is a class that allows .NET applications to host WCF endpoints within the application to host the services, the class provided is called ServiceHost. It's important for at least one endpoint to be available for the client where it can access or invoke operations through it. [1]

Before you start hosting a service, the contract needs to be defined with the [serviceContract] attribute in the .NET class. The endpoint needs to be defined by using the AddServiceEndpoint method, which specifies the address, binding and contract. This method is used in the ServiceHost class [4].

This ServiceHost class includes a constructor which identifies the service type that is going to be hosted and contains the Uniform Resource Identifiers (URI). These URIs are the base addresses of the WCF framework which requires a configuration file to be created.

An application configuration file is created within the application itself. It is important that every base address is created with a unique schema (as the address is the unique location of the service). The contents of the configuration file can be modified, and the developer should choose to build the solution after any modifications to ensure it works [5].

ServiceHost instances are configured through the configuration file or using code. The user does this by calling Servicehost.AddServiceEndpoint, specifying the binding and then choosing the endpoints that they wish to configure.

Self-hosting a WCF service

Self-hosting is a flexible way to host a service, it supports all bindings, transports and all you need is the WCF runtime and a .NET application to host ServiceHost to do this.

The user must implement at least one instance of ServiceHost when they are self-hosting, this calls ServiceHost.Open. There is no limit to the number of instances that the user can create of the class ServiceHost. Self-hosting allows the user to create, open and configure the instances of ServiceHost and this ay of hosting may be chosen if the developer doesn't want to use HTTP as the protocol [6].

Hosting a WCF service in a Windows service (using transmission control protocol)

To begin creating this type of WCF service, the user needs to configure the endpoints to enable them to use Transmission Control Protocol (TCP) and the base address is set at this stage.

The Windows service is modified so it can host the WCF service and then the user should install Windows Service. The user must then create a Windows Form application where they wish to test the WCF service that they are creating. To test, the user needs to add a WCF service reference by right-clicking on the project (which would be the Windows Form application that they just created) and select "Add Service Reference". The final stage is to test that the WCF service runs without a problem [7].

Hosting in a windows service have the advantages of binding flexibility and built in support to restart when failure occurs. However, there is the disadvantage of the features being quite limited on Windows.

Hosting a WCF Service in IIS

Hosting a WCF service in IIs is sometimes referred to as managed hosting. These are the stages for a user to deploy an internet information services-hosted WCF service [8]:

- Install and register IIS, ASP.NET, MVC and the WCF activation component.
- Create new IIS application.
- A .svc file is created as the WCF requires it (ServiceHost is created based on the .svc).

- At this stage, the service implementation to the IIS application is deployed.
- WCF services are configured at this stage, these are configured in the applications Web.config file.

The benefits to hosting a service in IIS is that the services are managed like an ASP.Net application or any other IIS application and it improves manageability and security [4]. The user does not always have to create a new IIS application to host a WCF service, they are able to reuse any of their existing ASP.NET applications if they wish to do so.

ABCs of WCF

An endpoint consists of the ABC's of WCF: address, binding and contract.

The address is the unique location of the service which is identified by a URI, an address could be the URL or IP address (for example). The address is made up of a scheme, domain and port. There are different types of address such as HTTP addresses, IPC addresses and TCP addresses where the address contains the transport protocol e.g. HTTP addresses will use Http for transport. The user decides the transport protocol for themselves when creating the WCF service, the transport protocol is decided based upon the project specifically, for example Http is used for the web and TCP is more suitable when high performance communication within the application is preferred [9].

The binding defines how the client and the service communicates, WCF provides a list of predefined bindings available for the developer to use, for example 'basicHttpBinding' or 'netMsmqBinding', each standard/default binding allows the user to do something specific such as exposing services in a certain format such as Http or allowing the binding to communicate a certain way. The user can define their own through a configuration file or in the code and are able to set multiple bindings within the application [10].

When hosting or creating a host, the binding is specified when configuring the endpoints in the ServiceHost class and the contract is specified when AddServiceEndpoint is called [6]. The address is set through a configuration file or in the code within a class called EndpointAdress.

Lastly, the 'C' is the contract, this provides the functions and operations that are available to the client exposed by an endpoint [11], and there are different types of contracts available such as the data contract and fault contract, which are discussed later in this report.

The principles and process of metadata exchange in WCF.

WCF provides structure to export, publish, import and retrieve service Metadata. A service can publish metadata by using a configuration file or by using code (which uses a dedicated endpoint), there are different steps for the two ways.

There are many steps to publishing metadata using an application configuration file. It begins with creating a <behaviours> element and then including a <serviceBehaviours> within it. Once that ServiceBehaviour element has been added, another <behaviour> element must be added to it with a specified value to the attribute 'name'. This type of service uses HTTP GET request and this is implemented by adding a <serviceMetadata> element and ensuring that the attribute httpGetEnabled is set to true so the service is enabled. The last few steps include adding a behaviourConfiguration attribute, adding endpoints with the IMetadataExchange set as the contract, and then the user should finally specify the binding and addresses of these endpoints. Once these steps are followed then the application can be built, and metadata can be published [12].

The IMetadataExchange interface (which is mentioned above in one of the stages of publishing data) represents the MEX endpoint, MEX stands for metadata exchange endpoint and it is an industry standard for metadata exchange [13].

The second way to publish metadata is through code on a WCF service. To begin publishing metadata, a WCF service needs to already be in place which will help a user test the code and make sure it is working. A ServiceHost object needs to be instantiated and (if one doesn't exist already) then a new instance of MetadataBehavior needs to be created here. Like the part in publishing metadata through a configuration file, the HttpGetEnabled must be set to equal true.

There is a property called PolicyVersion which the user should be set to Policy15. This property will be found within the MetadataExplorer property which is found within ServiceMetadataBehaviour. After these steps, the user should go to the part of code where the collection of the service host behaviours is located and add an instance of ServiceMetadataBehaviour. The last few steps before the user should build and run the solution are to add the Metadataexchange endpoint and add the application endpoint to the service host [14].

Main Contract Types

As previously mentioned, contracts are the 'C' in the ABCs of WCF; there are four main defined contract types, which are:

- Data contract
- Fault contract
- Service contract
- Operation contract

Data Contract

In WCF the service client and the server exchanges data. Before doing so, an agreement needs to be made of the exact data that is going to be exchanged in the service. The data contract represents and defines this data [15]. This type of contract also allows the user to serialize data by the formatter class: DataContractserializer, .NET already supports the serialization of data, which allows WCF to make use of with the formatter mentioned [3].
An example of defining a data contract using the [DataContract] attribute below:

```
20      [DataContract]
21      public class SystemInformation
22      {
23      }
```

Figure 1 Example code of defining a data contract

Service Contract

The service contract is the interface for WCF as it defines the operations of that service to the outside world, a WCF application can have more than one service contract. This type of contract also defines what kind of message exchange pattern is going to take place, and this contract can be defined using the attribute [ServiceContract] which is added to an interface or a class. A simple example of this is shown below [16]:

```
15      [ServiceContract]
16      public interface IService
17      {
18      }
```

Figure 2 Example code of defining a service contract

Operation Contract

An operation contract describes the functionality and exposes the methods. This contract is defined within the servicecontract using the [OperationContract] attribute, and an operation contract can also define the fault contract [16].

```
[ServiceContract]
public interface IService
{
    [OperationContract]
    List<example> GetExample(string Val);
}
```

Figure 3 Example code of defining an Operation Contract within a Service contract

Fault Contract

The fault contract defines and provides the user a documented view when there is an error in the system [17]. There are different types of errors that the user can come across when they are creating the service and these errors throw different exceptions. For example, a CommunicationException shows an error within the communication where as an error caused in the service call will throw the errors as FaultException [17]. An example of the fault contract throwing a general exception [18]:

```
[ServiceContract()]
public interface IAddMenuItems
{
    [OperationContract()]
    string AddMenuItem(string item1, string item2);
}

public class Menu : IAddMenuItems
{

    public int AddMenuItem(string item1, string item2)
    {

        throw new Exception("Error while adding item");

    }

}
```

Figure 4 Fault contract general exception example

Supported exchanged patterns in WCF.

There are three main message exchange patterns that are used in WCF which are going to be discussed in this section, the default exchange pattern in WCF is always the Request-Response pattern. The Request-Response pattern always generates a response when a service operation has been called.

There is also the One-way pattern. One-way is an exchange pattern where a message is sent from the client application to the service application and does not need a reply, unlike the Request-Response pattern which always generates a reply [9].

However, in this pattern, the message may not always reach the service that it was intended due to the out-of-band confirmation of a successful delivery. The service operation is set as a one-way pattern and represented when the attribute OperationContract is IsOneWay=True [3].

There is the Duplex pattern. The duplex pattern is where the user defines two interfaces, which allows both the client and the service to start communications between one another, whereas the One-way and Request-Response patterns are where the clients begin the communication. The advantage to implementing a duplex pattern is that the operation can request specific information from the client rather than the client initially sending too little or too much information in the exchange.

These exchange patterns mentioned can be implemented as an asynchronous operation using a message exchange. The contract and service implementation control the asynchronous operations and the message exchange patterns are specified in the WCF attributes. Asynchronous operations allow multiple methods to run concurrently without interrupting the system [19]. An asynchronous operation can be implemented by using a custom ISyncResult implementation.

Lastly, there is the Peer-to peer-pattern. Peer-to-peer is a popular term when it comes to distributed computing as it enables a group of computers to work together on the same network, the type of peer-to-peer application can be one-to-one, one-to-many and many-to-many. Distributed computing has many advantages as this allows users to collaborate on their projects and share each other's files within the network, this type of pattern enables distributed processing which is helpful for teams as it then breaks down larger tasks into smaller more manageable sized tasks for the individual to work on [20].

This pattern relates to WCF, as a peer-to-peer network is a mesh network that allows a continuous connection, it involves a unique ID (also known as peer ID) on each node and this unique peer ID is part of an endpoint along with a communication protocol and port number. The data is routed between the nodes in this mesh network by using an endpoint. A user can configure a network on windows to allow Peer-to-Peer and/or optionally choose to install Windows P2P networking stack [20].

Interoperability in WCF

WCF supports interoperability; this allows the integration of different applications to work together in a system. It also allows the exchange of data over multiple machines and is able to communicate this data/information over different hardware and software.

WCF allows interoperability with ASP.NET and the WCF web services, Web services enhancement and it also integrates with COM and COM+ applications; the COM+ applications can be exposed as WCF services whereas COM applications are able to call them. Interoperability also allows WSE 3.0 Web services and ASP.NET Web services to migrate to a WCF service.

Interoperability with Web Services is achieved in WCF when they are configured to support the specifications that are industry standard Web service specifications. For example, the services implemented within the ASP.NET and WCF web services must conform to the WS-I Basic Profile 1.1 specification to enable interoperability, and the standard binding of the BasicHttpBinding also provides interoperability by conforming to these specifications. [10].

Users can carry out interoperability analysis on their systems to identify any issues within the interoperability. However, there has been a study [21] to identify difficulties of performing interoperability analysis. The study concluded a couple of main points from the surveys that were performed in the study, it was felt that concrete interoperability analysis approaches should be developed, and criteria should be identified to resolve the issues. Another point of the conclusion was that developed guidelines should be set when it comes to applying interoperability analysis.

References

[1] M. Bustamante, Learning WCF: A Hands-on Guide, O'Reilly Media, Inc, 2007.

[2] J. Sharp, Windows Communication Foundation 4 Step by Step, Pearson Education, 2010.

[3] S. Klein, Professional WCF Programming: .NET Development with the Windows® Communication Foundation, John Wiley & Sons, 2007.

[4] S. R. C. B. Richard Crane, Essential Windows Communication Foundation (WCF): For .NET Framework 3.5, Pearson Education, 2008.

[5] M. M. N. W. Craige McMurtry, Microsoft Windows Communication Foundation: Hands-on!, Sams Publishing, 2006.

[6] J. Flanders, RESTful .NET: Build and Consume RESTful Web Services with .NET 3.5, O'Reilly Media, Inc, 2008.

[7] "How to: Host WCF in a windows Service Using TCP," Microsoft, [Online]. Available: https://msdn.microsoft.com/en-us/library/ff649818.aspx#_Step_7:_Create. [Accessed 12 December 2017].

[8] M. W. M. H. L. L. y. Dotnet bot, "Deploying an Internet Information Services-Hosted WCF Service," Microsoft, 30 March 2017. [Online]. Available: https://docs.microsoft.com/en-us/dotnet/framework/wcf/feature-details/deploying-an-internet-information-services-hosted-wcf-service. [Accessed 8 December 2017].

[9] M. L. Bustamante, Learning WCF, O'Reilly Media, Inc., 2007.

[10] J. G. P. C. F. C. Kurt Claeys, Professional WCF 4: Windows Communication Foundation with .NET 4, Wrax, 2010.

[11] C. Patel, Developing Service-Oriented Applications Using the Windows Communication Foundation (WCF) Framework, IGI Global, 2017.

[12] M. j. M. W. M. H. dotnet bot, "How to: Publish Metadata for a Service Using a Configuration File," Microsoft , 03 March 2017. [Online]. Available: https://docs.microsoft.com/en-us/dotnet/framework/wcf/feature-details/how-to-publish-metadata-for-a-service-using-a-configuration-file. [Accessed 28 December 2017].

[13] J. Lowy, Programming WCF Services, 3rd Edition, O'Reilly Media, Inc., 2010.

[14] M. W. M. j. D. B. Matt Hoffman, "How to: Publish Metadata for a Service Using Code," Microsoft, 30 March 2017. [Online]. Available: https://docs.microsoft.com/en-us/dotnet/framework/wcf/feature-details/how-to-publish-metadata-for-a-service-using-code. [Accessed 8 January 2018].

[15] S. Cheng, Microsoft Windows Communication Foundation 4.0 Cookbook for Developing SOA Applications, Packt Publishing Ltd, 2010.

[16] M. M. Juval Lowy, Programming WCF Services: Design and Build Maintainable Service-Oriented Systems, O'Reilly Media, Inc., 2015.

[17] J. Lowy, Programming WCF Services, O'Reilly Media, Inc., 2007.

[18] "Fault contract," WCF Tutortial, [Online]. Available: http://wcftutorial.net/fault-contract.aspx. [Accessed 22 January 2018].

[19] A. C. R. S. K. L. Richard Blewett, Pro Asynchronous Programming with .NET, Appress, 2014.

[20] N. Pathak, Pro WCF 4: Practical Microsoft SOA implementation, Second Edition, Appress, 2011.

[21] D. R. Hadil Abukwaik, "Software Interoperability Analysis in Practice: A Survey," *Proceeding EASE'17 Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering,* pp. 12-20, 2017.

WCFConceptsLeahCahill

GRADEMARK REPORT

FINAL GRADE

79/100

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

PAGE 3

Comment 1

Hyphenate

QM

Contraction

Don't use contractions in academic writing. E.g. you should use "do not" instead of "doesn't".

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

Comment 2

Code looks OK - but think you're missing a namespace include.

PAGE 9

PAGE 10

PAGE 11

PAGE 12

GRADING FORM: SMU GENERIC_GJ

LEAH CAHILL

79

RESEARCH-INFORMED LITERATURE

An excellent number of sources used, and cited correctly. As mentioned previously, maybe a little heavily weighted towards books and the web, but this doesn't significantly detract from the work presented overall. Well done, Leah.

8

KNOWLEDGE AND UNDERSTANDING

Good evidence of knowledge and understanding - you obviously spent a lot of time researching this work.

22

ANALYSIS

Some reasonable synthesis of material - in places you possibly hang too much on a single reference. It'd be better to cite multiple sources to back up arguments where possible. Consider including some concluding sections - these'd really help tie your work together.

17

PRACTICAL APPLICATION AND DEP.

Very good consideration of practical implementation, with appropriate code snippets presented.

24

SKILLS FOR PROFESSIONAL PRAC.

You write very well - I enjoyed reading this report. The document is nicely-presented - might actually benefit from a few additional figures in places. Overall, an extremely good piece of work - really well done, Leah. We can have a chat about the (relatively minor) things you can do to aim for an even higher mark in the second assignment.

8