

Distributed Enterprise applications

by Josh Benjamin Rowland

FILE	2664_JOSH_BENJAMIN_ROWLAND_DISTRIBUTED_ENTERPRISE_APPLIC ATIONS_94244_1840759064.DOCX (225.25K)		
TIME SUBMITTED	16-JAN-2018 02:55PM (UTC+0000)	WORD COUNT	2519
SUBMISSION ID	81875472	CHARACTER COUNT	14825

Distributed Enterprise Apps/Distributed
and Cluster Comp

P145133

Josh Rowland

BSc Web Development

WCF Concepts

Dr Tim Bashford

Contents

Predominant Rationale.....3

Hosting.....3

Exchange Patterns.....8

Interoperability.....9

Predominant Rationale

The Windows Communication Foundation or WCF is a framework that was built to allow developers to create “service-oriented applications [1].” WCF can be used for situation such as a chat service between two people in real time or a dashboard application that polls several services for data and “presents it in a logical presentation[1].” WCF has many features a developer can make use of such as transaction support using either Systems.Transaction, WS-AtomicTransaction and the Microsoft Distributed Transaction Coordinator [1]. WCF also supports durable messages to ensure that messages are never lost by saving the message to a database which allows the message to continue its path once the connection has been re-established [1]. There are many features to the WCF framework because it incorporates several older methods of building a distributed application such as .NET Remoting, Web services, MSMQ (Message Queueing) and more [2]. .NET Remoting enabled the developer to build widely distributed applications with ease regardless of whether components are all on one device or spread across the world[3]. Due to .NET remoting providing an “abstract approach to the interposes communication [3]” it becomes very flexible and customizable however WCF can use not only HTTP, TCP and SMTP protocols but also named pipes and MSMQ which give the user an increased choice for communication channels compared .NET remoting[4]. Not only does it have more communication channels but also simpler callbacks which are a “complex area in .NET Remoting[4]” because the developer has to use an intermediary shim object in order to successfully receive the event unlike WCF which can use a “Duplex[4]” which is based on a continuous series of one-way interactions to form a “complete conversation” as stated on [4]. WCF has also come to replace web services which was an programmable application logic using standard web protocols such as the SOAP or simple object based protocol[5]. Using WCF web services gives the developer much more flexibility and portability because WCF is designed by Microsoft to unify different distributed programming methods that were developed by Microsoft[5]. MSMQ is a technology that was designed to enable applications that are running at different times to communicate across networks and systems that are currently offline[6]. MSMQ has several ways it can be used such as e-commerce and embedded or hand-help applications[6]. In WCF it expands the usage of MSMQ allowing for scenarios such as failure isolation to stop application failures affecting one and other or load levelling which prevents a receiver from being overwhelmed[7]. COM+ was an improved version of Microsoft Component Object Model and Microsoft Transaction Server which expands application written using COM and MTS[8]. Due to how successful and widely used COM+ was Microsoft had to include it in WCF so that developers could extend their existing logic so that there was no need to re-write it[9]. WCF combines a large number of different technologies into one and improves on the technologies in order to offer developers an easier and more manageable way to develop web services and web service clients[1].

Hosting

WCF doesn't actually come with its own host, the developer instead should use the `ServiceHost` class that allows the developer the ability to host WCF services in their own application[10]. There are several options of hosing such as a WinForms application, Console application, web applications and more however it is not recommended to use other templates but the developer will not be blocked from doing so[10]. There are other ways to host such as self-hosting, hosting in a windows service or hosing in different versions of IIS(internet

information services) which will be discussed further into the report[10]. Self-hosting is the most flexible and easiest way to host the WCF service because with only a small amount of code will have the service running also the developer can also control the service lifetime using the Open() and Close() methods however it has limited availability because it is only reachable when the application is running and it is not suited for large scale operations and should be used only for development and demonstrations[10]. Using IIS to host means the developer needs to understand sites, applications and virtual directories[10]. Sites are bound to a certain scheme, network address and port combination which allows the developer to host multiple applications under the same site that use the same scheme, network address and port but that's depending on the version of FTP(File transfer protocol), NNTP(Network news transfer protocol), SMTP(Simple mail transfer protocol) and HTTP(Hypertext transfer protocol)[10]. There are three basic concepts to WCF, these are the address binding and contract concepts and all three are key to hosting and using a WCF service[11]. Put simply the WCF address is the exact location for a service, it is used to specify the path where messages are sent and all WCF services require an address to receive incoming requests[11]. The binding concept is used to specify the transport, protocol and encoding details that are required for the clients and services to communicate with one and other, also the application must have at least one transport and encoding binding with any number of protocol binding elements[11]. The contract concept is a set of specification that were made to define the interfaces that are used with WCF services and are split into different contract types as seen in[11]:

- The service contract – It is simply the interface of a WCF service which tells the user what the service can do while also allowing the developer to define methods or service operations for tasks and it must contain at least one service contract so that it can service the requests.
- The operation contract – This contract must be defined inside of the service contract and defines the return type and parameters of a specific operation. It can only take primitive data such as integers or messages. This contract must be implemented for the WCF service to function. It can also define settings such as the transaction flow, direction of the operation and the fault contract.

An example of the service and operation contract can be seen below. The example shows the [ServiceContract] is declaring the ICustomer interface to allow the user to access the interface. The [OperationContract] is then declaring the function inside the interface for the client to call and it is key that the contract tag is used as show below or the function will not run correctly.[12].

Figure 1. [13]

```

1  [ServiceContract]
2  interface ICustomer
3  {
4      [OperationContract]
5      Response AddNewCustomer(string customername);
6      Response DeleteCustomer(int customerID);
7  }
```

- The message contract – This is used when an operation contract needs to either pass or return a message. The contract defines the message elements and settings such as security level and of an element should go into the body or head.
- The data contract – This contract is used for data types of all the WCF services. This contract is used by the operation contract as a parameter and return type or even by a message contract to define elements. There is no need for this contract if the WCF service only uses primitive data types.

An example of the data contract as seen below shows the [DataContract] class Customer with the [DataMember] information inside of the class for the client to access however ContactNumb has been left **unsterilized** to show that it will not be given access unless the tag is used[12].

Figure 2. [14]

```

1 [DataContract]
2 public class Customer
3 {
4     [DataMember]
5     public int ID {get; set;};
6
7     [DataMember]
8     public string Name {get; set;};
9
10    public string ContactNumb {get; set;};
11 }

```

- The fault contract – Error types are defined as fault contracts which are used in any WCF service to return an error. An operation can have any number of fault contracts involved with it.

An example of the fault contract as seen below shows the declaration of the fault contract inside of the service contract.

Figure 3. [15]

```

1 [ServiceContract]
2 public interface IService1
3 {
4     [OperationContract]
5     [FaultContract(typeof(ServiceData))]
6     ServiceData TestConnection(string strConnectionString);
7 }

```

Once it has been declared in the above code the fault contract can then be put to use as displayed in the code below which shows its use with a try and catch functions to display the error messages depending on the issue[12].

Figure 4. [16]

```

1 namespace FaultContractSampleWCF
2 {
3
4     public class Service1 : IService1
5     {
6         public ServiceData TestConnection(string StrConnectionString)
7         {
8             ServiceData myServiceData = new ServiceData();
9             try
10             {
11                 SqlConnection con = new SqlConnection(StrConnectionString);
12                 con.Open();
13                 myServiceData.Result = true;
14                 con.Close();
15                 return myServiceData;
16             }
17             catch (SqlException sqlEx)
18             {
19                 myServiceData.Result = true;
20                 myServiceData.ErrorMessage = "Connection can not open this " +
21                     "time either connection string is wrong or Sever is down. Try later.";
22                 myServiceData.ErrorDetails = sqlEx.ToString();
23                 throw new FaultException<ServiceData>(myServiceData, sqlEx.ToString());
24             }
25             catch (Exception ex)
26             {
27                 myServiceData.Result = false;
28                 myServiceData.ErrorMessage = "unforeseen error occurred. Please try later.";
29                 myServiceData.ErrorDetails = ex.ToString();
30                 throw new FaultException<ServiceData>(myServiceData, ex.ToString());
31             }
32         }
33     }
34 }

```

.As of .NET 4.0 onwards the developer can use the file-less approach also known as svc-less activation service to specify which service that the website should host[11]. The developer will also need to edit the web.config file to host the chosen WCF service. As seen below in figure one the image shows the web.config edited to have the system.serviceModel node that allows the application to use specific WCF services and edit their settings[11]. To host in IIS the developer needs a file with an svc extension so the IIS can create the ServiceHost which means there is no need for the developer to start the ServiceHost themselves but hosting in IIS means that the developer will need to configure the web.config file to show where the service should be hosted an example of which can be seen in figure 6 which shows the web.config file for a TradeService service[10]. The address attribute in figure 6 is empty because the svc file determines the base address of the service however the developer can set the endpoints address relative to the svc file[10].

Figure 5. [17]

```

<?xml version="1.0"?>
<!--
  For more information on how to configure your ASP.NET
  application, please visit
  http://go.microsoft.com/fwlink/?LinkId=169433
-->
<configuration>
  <system.web>
    <compilation debug="true" targetFramework="4.5"/>
    <httpRuntime targetFramework="4.5"/>
  </system.web>

  <system.serviceModel>
    <serviceHostingEnvironment >
      <serviceActivations>
        <add
          factory="System.ServiceModel.Activation.ServiceHostFactory"
          relativeAddress="./HostExpressServer/HelloWorldService.svc"
          service="HelloWorldService.HelloWorldService"/>
      </serviceActivations>
    </serviceHostingEnvironment>
    <behaviors>
      <serviceBehaviors>
        <behavior>
          <serviceMetadata httpGetEnabled="true"/>
        </behavior>
      </serviceBehaviors>
    </behaviors>
  </system.serviceModel>
</configuration>

```

Figure 6. [18]

```

<?xml version="1.0"?>
<configuration xmlns="http://schemas.microsoft.com/.NetConfiguration/v2.0">
  <system.serviceModel>
    <services>
      <service
name="QuickReturns.StockTrading.ExchangeService.TradeService"
behaviorConfiguration="tradeServiceBehavior">
        <endpoint name="basicHttpBinding"
address=""
binding="basicHttpBinding"
contract="QuickReturns.StockTrading.ExchangeService.?
Contracts.ITradeService"/>
        <endpoint name="mexHttpBinding"
contract="IMetadataExchange"
binding="mexHttpBinding"
address="mex" />
      </service>
      <service
name="QuickReturns.StockTrading.ExchangeService.TradeServiceInline"
behaviorConfiguration="tradeServiceBehavior">
        <endpoint name="basicHttpBinding"
address=""
binding="basicHttpBinding"
contract="QuickReturns.StockTrading.ExchangeService.?
Contracts.ITradeService"/>
        <endpoint name="mexHttpBinding"
contract="IMetadataExchange"
binding="mexHttpBinding"
address="mex" />
      </service>
    </services>
    <behaviors>
      <serviceBehaviors>
        <behavior name="tradeServiceBehavior" >
          <serviceMetadata httpGetEnabled="true" />
        </behavior>
        <behavior name="returnFaults"
returnUnknownExceptionsAsFaults="true"/>
      </serviceBehaviors>
    </behaviors>
  </system.serviceModel>
</configuration>

```

All WCF services will contain metadata that is used to describe the characteristics of the service that an external entity needs so that it can communicate with said service[11]. Metadata includes information such as the XML schema that defines the data contract of not only the service but the WSDL documents that describe the methods in the service[11]. It is possible to hide the metadata from external entities but the developer will need to pass the metadata to the client another way but in doing so adds an extra layer of security[11]. WCF will generate the metadata for the service contract and using certain code can control the way in which the meta data will be generated using properties such as the “Name” and “Namespace” and using other attributes in the code can signal the WCF runtime on how to generate the metadata and WSDL that is needed for finding supported operations plus managing runtime calls from clients[19].

Exchange Patterns

There are several exchange patterns that are used in WCF such as the request-response pattern also known as the request-reply pattern, it is the most used pattern that has every call returned by a message[20]. There is also the one-way interaction which is simply the

application sending a message to the WCF service, but the service does not send a reply such as logging in and out[21]. Or the developer can use the duplex pattern allows both the client and the service to initiate the communication unlike the one-way pattern and it is used when the service is needed to send information to the client after the client has called the service[21]. There is also the asynchronous two-way pattern which is very similar to the one-way pattern however this pattern has the extra requirement that there needs to be a function to correctly associate the response message with the correct request and both sides must have a web service to receive the messages[22]. Lastly there is the peer-to-peer exchange which consists of a networks of computers and does not rely on any form of server instead all communication is directly from computer to computer to form a group to increase the total computing power[19]. In WCF it is normally used as a peer channel for message based services and has any number of participants so if a message is sent by one participant it will be received by all of the other participants on the channel (e.g. Facebook group chat)[19]. It is also possible to send a message to only part of the mesh using a custom resolver and once the node is resolved it can accept or decline a connection[19].

Interoperability

Interoperability is of massive importance as it means for systems and components to exchange data and work together with no restriction[23]. Unfortunately there were a number of limitations with interoperability such as if you try to use a COM+ application to communicate over firewalls or organisational boundaries there will be significant difficulty in succeeding as most enterprises work on different technology[19]. It can be used in WCF by ASP.NET web services if the service implemented uses the WS-I basic profile 1.1 specification and the WCF client uses system-providing binding[24]. The developer should use ASP.NET 2.0 to add the WebService and WebMethodAttribute to an interface because it can then be reused with other classes as seen in figure two below[24]. Using WS-I (web services interoperability) allows seamless communication with other software on other operating systems and because the core of all web services are in WSDL, WCF and speak the language of the latest web services[19]. An example is that several SOAP messages are sent from different sources with different operating systems to the WCF service it can interact with all these different technologies and provide reliable messaging even if the WCF application runs on Windows vista can easily interact with a java application running on the IBM mainframe[19]. WCF also supports non SOAP XML, JSON, RSS and binary formats for communication to external sources if it is HTTP or HTTPS connection[25]. This would remove the issues that was previously discussed where it is difficult to achieve interoperability between separate technologies used by separate enterprises.

Figure 7. [24]

```

[WebService]
public interface IEcho
{
    [WebMethod]
    string Echo(string input);
}

public class Service : IEcho
{
    public string Echo(string input)
    {
        return input;
    }
}

```

- [1] "What Is Windows Communication Foundation | Microsoft Docs." [Online]. Available: <https://docs.microsoft.com/en-us/dotnet/framework/wcf/whats-wcf>. [Accessed: 08-Dec-2017].
- [2] "Introduction to Windows Communication Foundation." [Online]. Available: <https://msdn.microsoft.com/en-us/library/dd936243.aspx>. [Accessed: 08-Dec-2017].
- [3] ".NET Remoting Overview." [Online]. Available: <https://msdn.microsoft.com/en-us/library/kwdt6w2k%28v=vs.71%29.ASPX>. [Accessed: 20-Dec-2017].
- [4] "From .NET Remoting to the Windows Communication Foundation (WCF)." [Online]. Available: [https://msdn.microsoft.com/en-us/library/aa730857\(v=vs.80\).aspx](https://msdn.microsoft.com/en-us/library/aa730857(v=vs.80).aspx). [Accessed: 20-Dec-2017].
- [5] S. Orientation, "What's the Difference between WCF and Web Services? - CodeProject," 2000.
- [6] "Message Queuing (MSMQ)." [Online]. Available: [https://msdn.microsoft.com/en-us/library/ms711472\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/ms711472(v=vs.85).aspx). [Accessed: 20-Dec-2017].
- [7] "Queues in Windows Communication Foundation | Microsoft Docs." [Online]. Available: <https://docs.microsoft.com/en-us/dotnet/framework/wcf/feature-details/queues-in-wcf>. [Accessed: 20-Dec-2017].
- [8] "COM+ (Component Services) (COM+)." [Online]. Available: [https://msdn.microsoft.com/en-us/library/windows/desktop/ms685978\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/ms685978(v=vs.85).aspx). [Accessed: 20-Dec-2017].
- [9] "Integrating WCF with COM+." [Online]. Available: <https://msdn.microsoft.com/en-us/library/bb735856.aspx>. [Accessed: 20-Dec-2017].
- [10] "Hosting and Consuming WCF Services." [Online]. Available: <https://msdn.microsoft.com/en-us/library/bb332338.aspx>. [Accessed: 23-Dec-2017].
- [11] M. Liu, *WCF multi-layer services development with Entity Framework : create and*

deploy complete solutions with WCF and entity framework. Packt Pub, 2014.

- [12] K. Behera, "Various Types of WCF Contracts." [Online]. Available: <http://www.c-sharpcorner.com/UploadFile/87b416/various-types-of-wcf-contracts/>. [Accessed: 30-Dec-2017].
- [13] K. Behera, "ServiceOperationContract." .
- [14] K. Behera, "DataContract." .
- [15] K. Behera, "FaultContractDeclaration." .
- [16] K. Behera, "FaultContract." .
- [17] M. Liu, *web.config*. 2014.
- [18] "TradeService web.config." .
- [19] N. Pathak, "Pro WCF 4: Practical Microsoft SOA Implementation, Second Edition."
- [20] P. Cibraro, K. Claes, F. Cozzolino, and J. Grabner, *Professional WCF 4 : Windows Communication Foundation with .NET 4*. Wiley, 2010.
- [21] "Message Patterns in WCF Services." [Online]. Available: <https://msdn.microsoft.com/en-us/library/ff395349.aspx>. [Accessed: 27-Dec-2017].
- [22] M. Ticknor and International Business Machines Corporation. International Technical Support Organization., *IBM WebSphere Application Server V8 concepts, planning, and design guide*. IBM Corp. International Technical Support Organization, 2011.
- [23] "Definition: Interoperability." [Online]. Available: <http://interoperability-definition.info/en/>. [Accessed: 30-Dec-2017].
- [24] "Interoperability with ASP.NET Web Services | Microsoft Docs." [Online]. Available: <https://docs.microsoft.com/en-us/dotnet/framework/wcf/feature-details/interop-with-aspnet-web-services>. [Accessed: 30-Dec-2017].
- [25] "Introducing Windows Communication Foundation in .NET Framework 4." [Online]. Available: <https://msdn.microsoft.com/library/ee958158.aspx>. [Accessed: 02-Jan-2018].

Distributed Enterprise applications

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

Instructor

60/100

PAGE 1

PAGE 2

PAGE 3



Comment 1

access

QM

Contraction

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

PAGE 4



Comment 2

This is quite a bit too close to the source material for comfort.

PAGE 5



Comment 3

I don't think that's the word you meant :-)

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

RESEARCH-INFORMED LITERATURE



Generally good use of literature - a few sources are incomplete, and possibly a little heavily-skewed towards web sources, even considering the context of this technology. Be careful with how you construct your work - you tread the line a little close to plagiarism in a few places. Make sure your work is entirely in your own words.

6

KNOWLEDGE AND UNDERSTANDING



Good evidence of knowledge and understanding, with only a few small gaps throughout the report. For example, it would've been nice to get into the nuance of fault contracts - what impact does a fault contract have on debugging?

15

ANALYSIS



Some evidence of synthesis of material, but a little bit too adherent to the idea of "one idea, one source". Try to read a little more around the topic to combine into multiple sources, citing each [1][2][3][4] - we'll have a chat about this to help you on the next assignment. It'd be nice to see some conclusions in your work - would definitely help the report to flow better.

12

PRACTICAL APPLICATION AND DEP.



Good discussion of practical application - nice to see some relevant code snippets.

22

SKILLS FOR PROFESSIONAL PRAC.



A few minor issues, noted throughout the report. You have a tendency to write enormous paragraphs - attempt to split your narrative up a little better, to make it easier to read. The work is generally nicely presented - well done Josh.

5