

Windows Communication Foundation

Jim Fawcett
CSE681 – Software Modeling and Analysis
Fall 2017

Primary Code References

- www.ecs.syr.edu/faculty/Fawcett/handouts/CSE681-OnLine/code/WCF_Examples
- Start with these references and use object browser and MSDN docs via F1 key

Tutorials

- [Tutorials on WCF, WPF, and more](#)
- [Getting Started - MSDN](#)

Distributed Computer Communication

- 1964 – Dartmouth Time Sharing System
- 1969 – First link of ARPANET installed
- 1974 – First TCP specification
- 1978 – TCP/IP specification
- 1980 – Ethernet
- 1983 – Berkely sockets released with BSD
- 1990 – CORBA 1.0
- 1991 – OLE
- Early 90's - DCE/RPC
- 1993 – COM
- 1994 – CORBA 2.0
- 1996 – ActiveX, DCOM
- 1997 – Java RMI (Sun jdk 1.1, Java 2.0)
- 2002 – .Net Remoting
- 2006 – WCF (.Net 3.0)

What is WCF?

- Provides software **services** on machines, networks, and across the internet
- Unified programming model for all of these
- Supported natively on Windows 8, 7 and Vista
 - Requires installation on XP
- Not available on other platforms
 - Mono?
 - DotGnu?
 - Mac OSX ?

Integration into .Net Facilities

- One model for distributed systems decomposes into presentation layer, application logic layer, and data layer, all bound together with communication.
- Presentation: WPF
- Data: LINQ – retrieves data
- Communic: WCF – local, network, internet
- Applic Logic: Custom designs

WCF Design Principles

- Boundaries are explicit
 - No attempt to hide communication
- Services are intended to be autonomous
 - Deployed, managed, and versioned independently
- Services share contracts and schemas, not types
 - Contracts define behavior, schemas define data
 - Client needs a reference to contract not to service implementation
- Compatibility is policy-based
 - Policy supports separation of behavior from access constraints

Essential Pieces of WCF

- Contracts for **services, data, and messages**
 - A contract is simply an interface declaration
- Service, Data, and Message definitions
 - Class implementations of the contracts
- Configurations defined programmatically or declaratively
 - .Net class instances versus config files.
- A host process (can be self hosted)
 - IIS, Windows Executable, Windows Service, or WAS
- .Net Framework Classes provide support for all of the above.

Examples

- All code samples for WCF must be run in Visual Studio with administrator privileges. That's because Windows requires administrative privileges to start and stop services.
- You can do that in at least two different ways:
 1. Create a link to Visual Studio (devenv.exe) and right-click, selecting run as administrator.
 2. Search on Visual Studio and right-click on the desktop app and select run as administrator.
- If you forget to do that you will get a crash dialog with lots of text.

Examples

These examples are important. They show you how to build services and proxies that can be self-hosted. The WCF wizard (there is only one) builds an IIS hosted service and that is not what we need for projects.

- BasicHTTP Service – Declarative
- BasicHTTP Service – Programmatic
- WSHttp Service – Declarative
- WSHttp Service – Programmatic

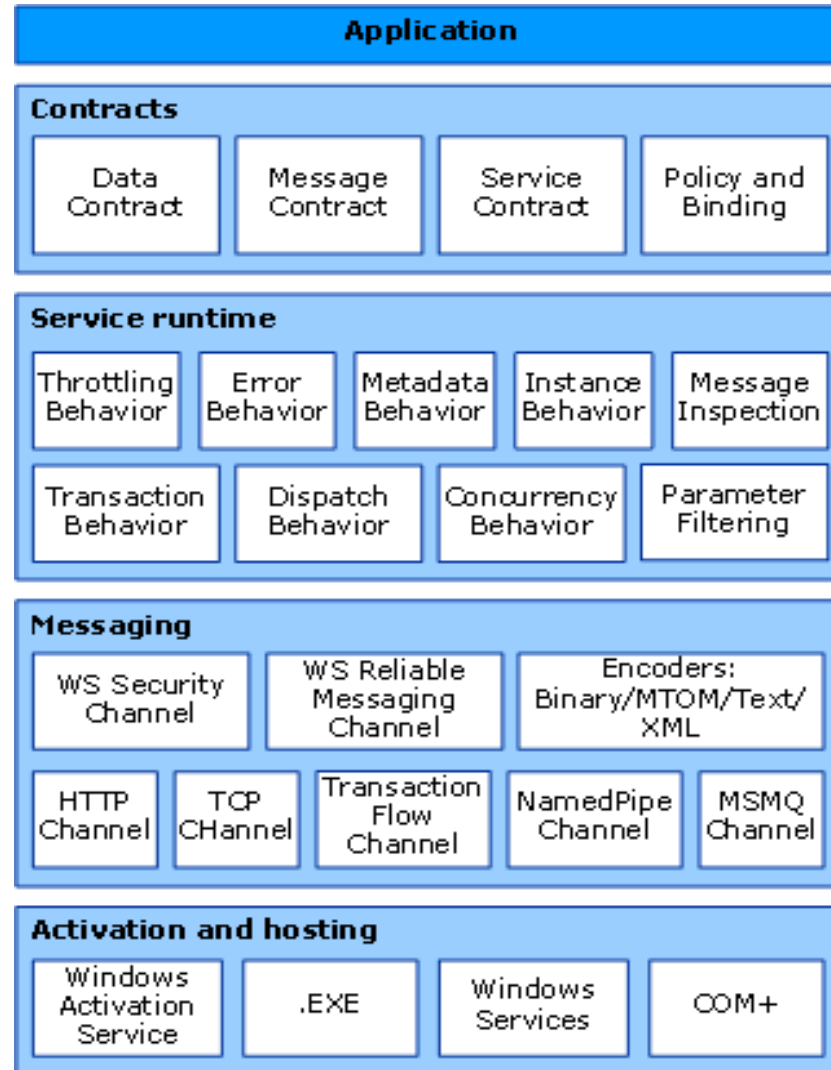
You will find these in the code folder in WCF Examples.

Look and Feel of WCF

- Convergence of programming models
 - Just like web services
 - Similar to .Net Remoting
 - Sockets on steroids
 - Hosting for local, network, and web
- Communication models
 - Remote Procedure Call (RPC) with optional data models
 - Message passing
 - One way, request and (callback) reply, synchronous call (wait for return)

WCF Architecture

<https://docs.microsoft.com/en-us/dotnet/framework/wcf/architecture>



ServiceModel Namespace

- Bindings, Channels, Endpoints, Messages, Serialization
- Activation, Concurrency, Hosting, Security, Sessions
- Queuing, Transactions
- Exceptions

Principle Parts of a WCF Service

- **Contract**
 - An interface defining services rendered
 - Service, Data, Message
- **Endpoints**
 - Address: <http://localhost/Calculator/service>
 - Binding: WSHttpBinding
 - Contract: ICalculator
- **Implementation**
 - One or more classes that implement the contract interfaces. May also include hosting code.

WCF Service Files

- **IService.cs**
 - Interface(s) that define a service, data, or message contract
- **Service.cs**
 - Implement the service's functionality
- **Service.svc**
 - Markup file (with one line) used for services hosted in IIS
- **Configuration files that declare service attributes, endpoints, and policy**
 - App.config (self hosted) contains service model markup
 - Web.config (hosted in IIS) has web server policy markup plus service model markup, as in App.config

Service serviceModel Markup

- ```
<system.serviceModel>
 <services>
 <service name="mySvcName" behaviorConfiguration="...">
 <endpoint address="" binding="wsHttpBinding"
 contract="myNamespace.myInterface" />
 <!-- can expose additional endpoints here -->
 <endpoint address="mex" binding="mexHttpBinding"
 contract="IMetadataExchange" />
 </service>
 </services>
 <behaviors>
 <serviceBehaviors>
 <behavior name="myNamespace.mySvcNameBehavior">
 <serviceMetaData httpGetEnabled="true" />
 <serviceDebug includeExceptionDetailInFaults="false" />
 </behavior>
 </serviceBehaviors>
 </behaviors>
</system.serviceModel>
```



# Channels

- Channels are the vehicles that transport messages. They provide:
  - Transport protocols via bindings
    - Http, WSHttp, Tcp, MSMQ, named pipes
  - Encoding and Encryption
  - Reliable sessions
  - Communication modes
    - Simplex, duplex, send and wait
  - Security modes

# WCF Bindings

| Binding                     | Interoperability                                                | Mode of Security (Default)        | Session (Default)                   | Transactions | Duplex |
|-----------------------------|-----------------------------------------------------------------|-----------------------------------|-------------------------------------|--------------|--------|
| BasicHttpBinding            | Basic Profile 1.1                                               | (None), Transport, Message, Mixed | None, (None)                        | (None)       | n/a    |
| WSHttpBinding               | WS                                                              | None, Transport, (Message), Mixed | (None), Transport, Reliable Session | (None), Yes  | n/a    |
| WS2007HttpBinding           | WS-Security, WS-Trust, WS-SecureConversation, WS-SecurityPolicy | None, Transport, (Message), Mixed | (None), Transport, Reliable Session | (None), Yes  | n/a    |
| WSDualHttpBinding           | WS                                                              | None, (Message)                   | (Reliable Session)                  | (None), Yes  | Yes    |
| WSFederationHttpBinding     | WS-Federation                                                   | None, (Message), Mixed            | (None), Reliable Session            | (None), Yes  | No     |
| WS2007FederationHttpBinding | WS-Federation                                                   | None, (Message), Mixed            | (None), Reliable Session            | (None), Yes  | No     |
| NetTcpBinding               | .NET                                                            | None, (Transport), Message, Mixed | Reliable Session, (Transport)       | (None), Yes  | Yes    |
| NetNamedPipeBinding         | .NET                                                            | None, (Transport)                 | None, (Transport)                   | (None), Yes  | Yes    |
| NetMsmqBinding              | .NET                                                            | None, Message, (Transport), Both  | (None)                              | (None), Yes  | No     |
| NetPeerTcpBinding           | Peer                                                            | None, Message, (Transport), Mixed | (None)                              | (None)       | Yes    |
| MsmqIntegrationBinding      | MSMQ                                                            | None, (Transport)                 | (None)                              | (None), Yes  | n/a    |

# Interoperability

- Channel protocols determine interoperability with other platforms:
  - BasicHttpBinding → universal interoperability
  - WSHttpBinding → platforms that use ws extensions
  - NetTcpBinding → .Net on both ends
  - MSMQ → WCF to pre WCF windows platforms

# Service Behaviors

- Instancing:
  - **Singleton**: one instance for all clients
  - **Per call**: one instance per service call
  - **Private session**: one instance per client session
  - **Shared session**: one instance per session shared between clients
- Concurrency models for instances:
  - **Single**: one thread at a time accesses instance
  - **Multiple**: more than one thread may enter instance
  - **Reentrant**: threads make recursive calls without deadlock

# Other Service Behaviors

- Throttling:
  - Limits on number of messages, instances, and threads a service can process simultaneously
- Error handling:
  - Options to handle, let framework handle, and report to client
- Metadata:
  - Services can be self-describing, providing MEX endpoints
- Lifetime:
  - Can specify session duration, service operations to initiate sessions and others to terminate sessions
- Security:
  - Can specify message confidentiality, integrity, authentication, authorization, auditing, and replay detection.

# Structuring Service Code

| Service           | Class     | Attribute                                                                     |
|-------------------|-----------|-------------------------------------------------------------------------------|
| Service contract  | interface | [ServiceContract]                                                             |
| Service operation | method    | [OperationContract]                                                           |
| Implementation    | class     | [ServiceBehavior]<br>Derive from contract<br>interface                        |
| Implementation    | method    | [OperationBehavior]                                                           |
| Data Contract     | class     | [DataContract] class<br>[DataMember] member                                   |
| Message Contract  | interface | [MessageContract] interface<br>[MessageHeader] member<br>[MessageBody] member |

# Building Clients

- There are three ways to build a client's proxy:
  - Hand craft a proxy using `ClientBase<ServiceType>` as we did in **BasicService-Declarative**
  - Create a proxy programmatically with the `ChannelFactory<ServiceContract>`, as we did in **SelfHosted\_StringsService**
  - Create a proxy with `SrvcUtil`. It also uses `ClientBase<ServiceType>` but adds a lot of other stuff as well.

# Building Clients

- Build proxy with svcutil
  - Visual Studio will do that if you add a service or web reference
  - Proxy code is in a subdirectory under the project.
- Add proxy code to project
- Add “using System.ServiceModel” to client code
- Build and run

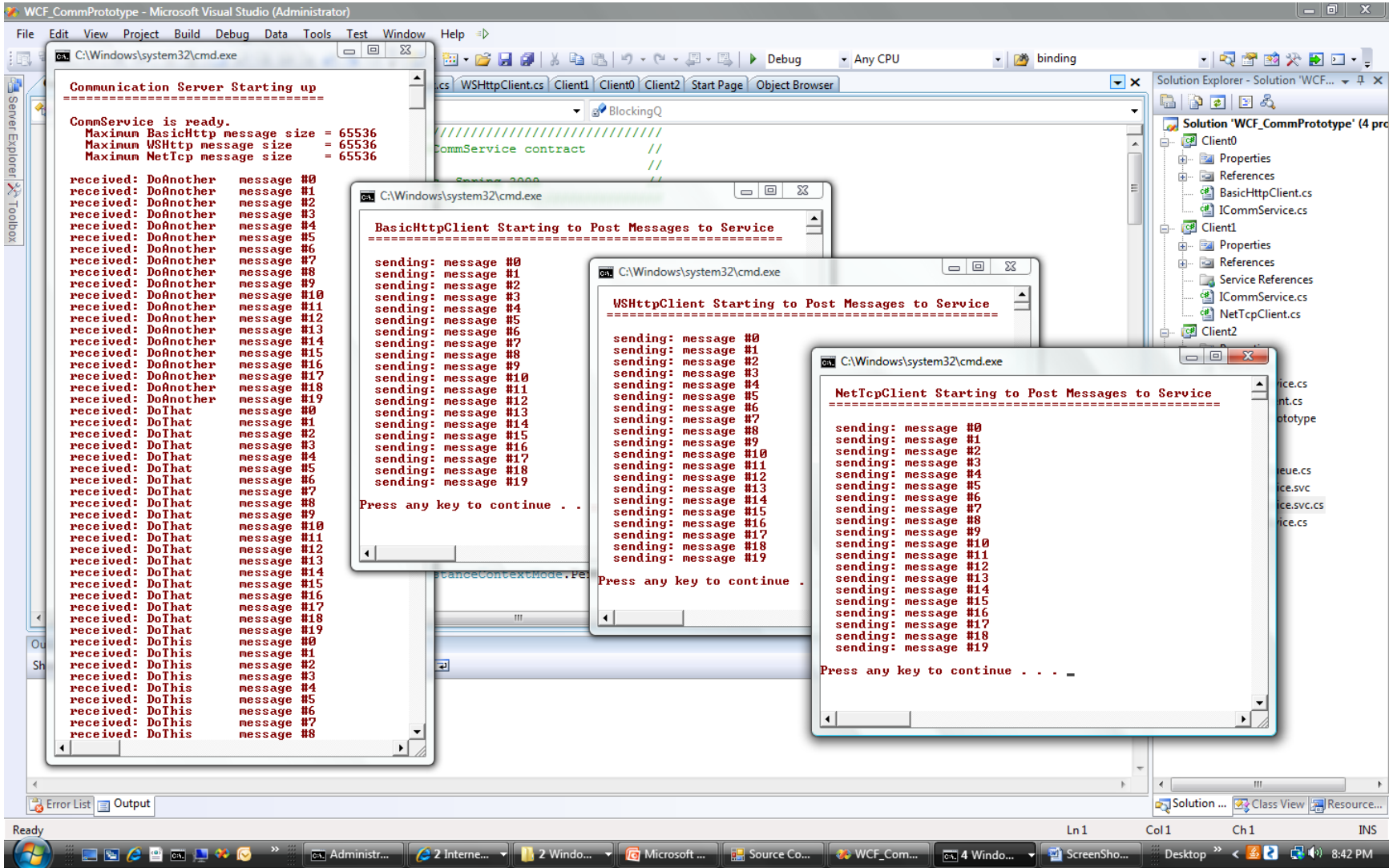


# Generating Proxies

- SvcUtil.exe generates code:
  - from a mex endpoint:  
svcutil <http://localhost/myService>
  - from WSDL or XSD files:  
svcutil myService.wsdl
- SvcUtil.exe generates WSDL and XSD files from a service library:
  - svcutil myService.dll

# Building Proxy Programmatically

- `WSHttpBinding binding = new WSHttpBinding();`
- `Uri address = http://Odysseus:4040/ICommService;`
- `ICommService proxy =  
ChannelFactory<IContract>.CreateChannel(binding, address);`
- `Message msg = new Message();`
- `msg.text = "a message";`
- `proxy.PostMessage(msg);`

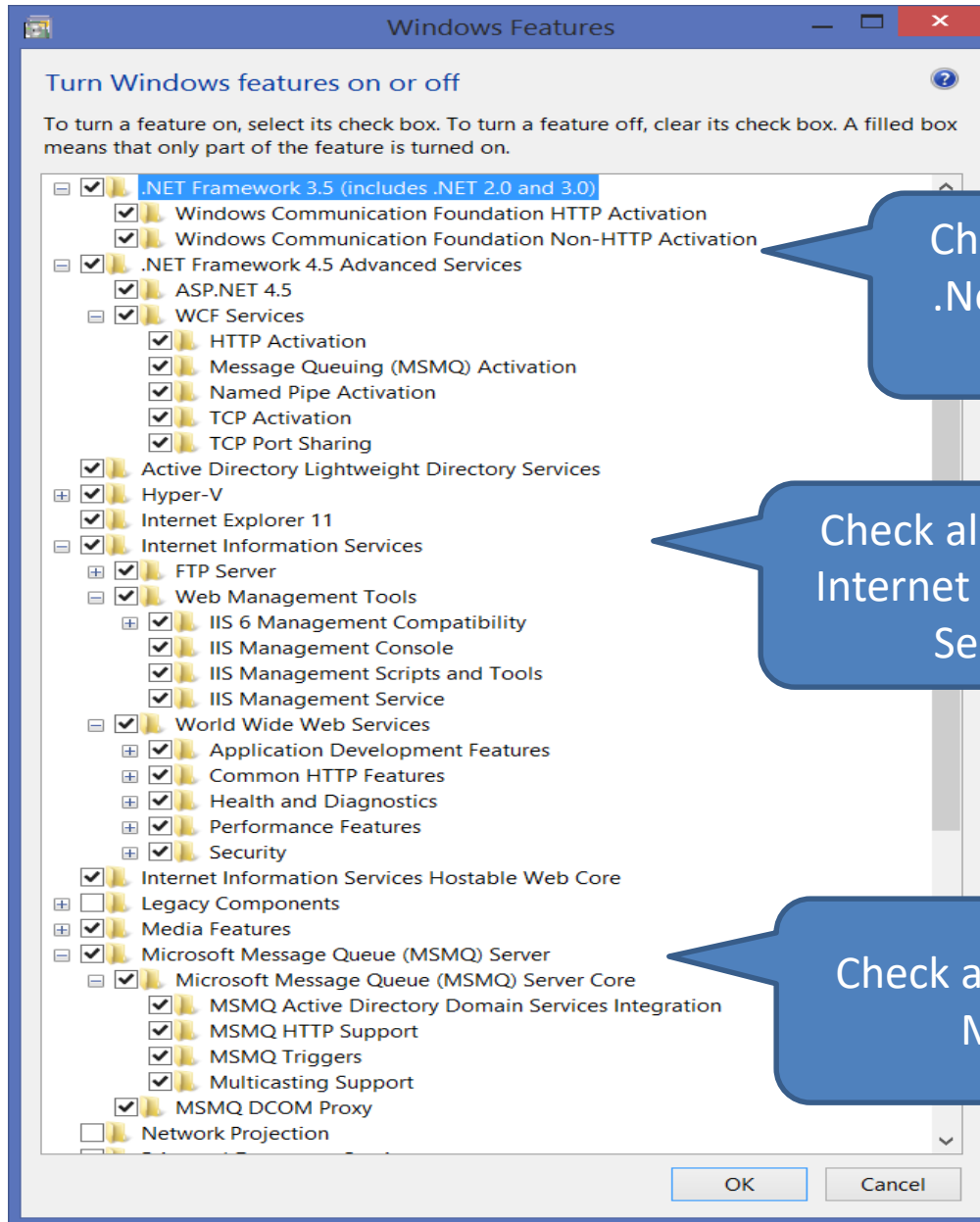


# References

- Setting up Reference Code
- Web References
- Tutorials

# Microsoft WCF Samples are useful, but NOTE !!!

- WCF Samples won't run unless you:
  - Install a digital certificate (can be self-signed), part of setup
  - Run Visual Studio as Administrator (in Win8, Win7, Vista)
- Setup steps
  - [One-Time Setup for WCF](#)
  - IIS7 doesn't include a folder of scripts used by setup so you have to add IIS6 compatibility which creates the script folder
  - There are a number of Windows features you need to turn on to make the samples work – see next slide.



Check all features in .Net Framework 3.5 and 4.5

Check all features in Internet Information Services

Check all features in MSMQ

# Primary Book References

- Pro C# 6.0 and the .Net 4.6 Platform, Andrew Troelsen, Apress,
  - Excellent introduction
- Programming WCF Services, 2<sup>nd</sup> Edition, Juval Lowy, O'Reilly, 2009
  - More up-to-date details than many of the others
- Inside Windows Communication Foundation, Justin Smith, Microsoft Press, 2007
  - Covers some of the plumbing, e.g., messaging and service host details
- Microsoft Windows Communication Foundation, Step by Step, John Sharp, Microsoft Press, 2007
  - Practical discussion with examples for securing communication

# Primary Web References

- [MSDN WCF Root Page](#)
- [WCF Feature Details](#)
- [MSDN WCF Architecture Overview](#)
- [Microsoft WCF Samples Setup](#)
- [Self-signed certificates, finding keys, ... Petar Vucetin blog](#)
- [Makecert - Certificate Creation Tool](#)
- [Distributed .NET Learn ABCs Of Programming WCF](#)
- [Service Station Serialization in Windows Communication Foundation](#)
- [Service Station WCF Messaging Fundamentals](#)
- [Windows Communication Foundation Glossary](#)
- [Windows Communication Foundation Tools](#)
- [Tutorials on WCF, WPF, and more](#)
- [A Performance Comparison](#)
- [Security in WCF](#)



# References – Activation & Channels

- [How to Access WCF Services with One-Way and Request-Reply Contracts](#)
- [How to Access Services with a Duplex Contract](#)
- [Sessions, Instancing, and Concurrency](#)
- [WCF Essentials Instance Management](#)
- [WCF Essentials One-Way Calls, Callbacks, Events](#)

# References – Web Model

- [How to Create a Basic Web-Style Service](#)
- [WCF Web Programming Model Overview](#)
- [Web Programming Model](#)
- [WCF Web Programming Object Model](#)
- [WCF Syndication HTTP Programming with WCF and the .NET Framework 3.5](#)
- [Creating WCF AJAX Services without ASP.NET](#)
- [Creating WCF Services for ASP.NET AJAX](#)

# References – Building Clients

- [Accessing Services Using a Client](#)
- [Clients](#)
- [Client Architecture](#)
- [Client Configuration](#)
- [Call WCF Service Operations Asynchronously](#)
- [A Performance Comparison](#)
- [Security in WCF](#)
- [File-streaming with a few comments about file chunking](#)

***End of Presentation***