Web Services

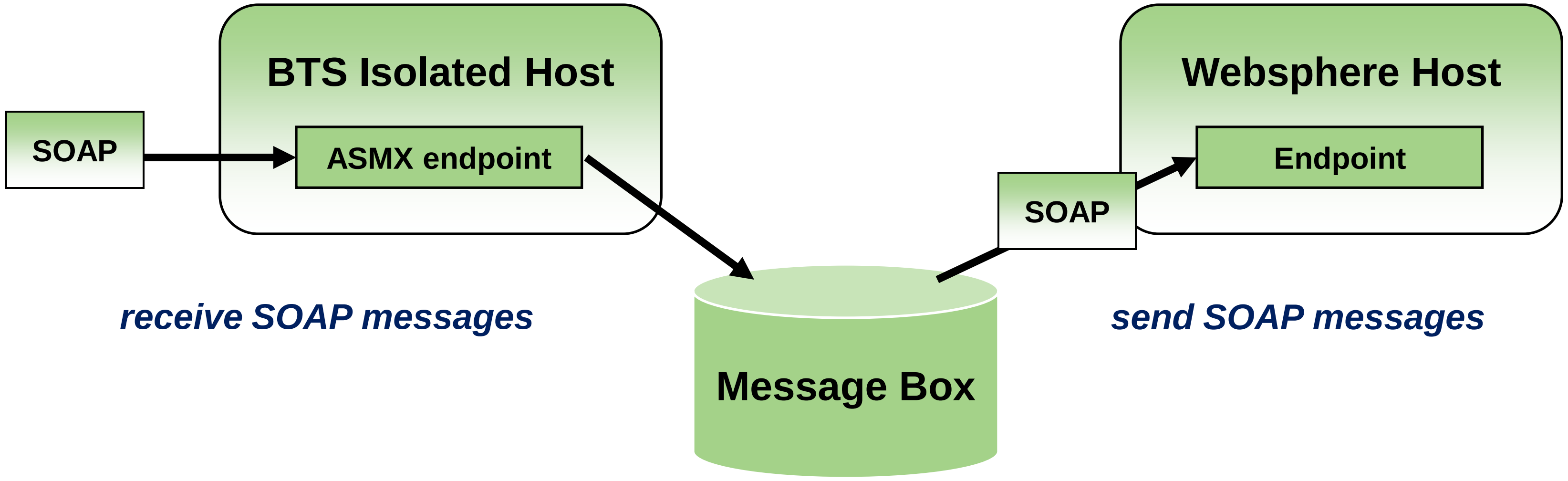
Understanding how to integrate BizTalk with SOAP

Outline

- Web service adapters
- Publishing schemas as Web Services
- Calling Web services from send ports
- Orchestrations and Web Services

Web service adapters

- **BizTalk provides support for Web services through adapters**
  - Adapters provide SOAP integration with the MB
  - Enables MB to send/receive *SOAP message payloads*
- **Support for ASMX, WSE, and WCF**



SOAP (ASMX) adapter*

- **The built-in *SOAP adapter* provides support for ASMX**
  - BTS 2004 supports ASMX 1.1
  - BTS 2006 supports ASMX 2.0
- **Used to send/receive messages via ASMX framework**
  - Receive messages into the MB via ASMX endpoint
  - Interact with external Web services via send ports

**** Deprecated in BizTalk Server 2009***

WCF adapters

- **The built-in *WCF adapters* provides support for**
  - WS-* specifications (WS-Security, WS-RM)
  - Multiple transports (.e.g TCP, HTTP, Named Pipes)
- **Used to send/receive messages via WCF**
  - Receive messages into the MB via WCFendpoint
  - Interact with external Web services via send ports
  - Receive locations can be hosted in IIS (HTTP) or in-process

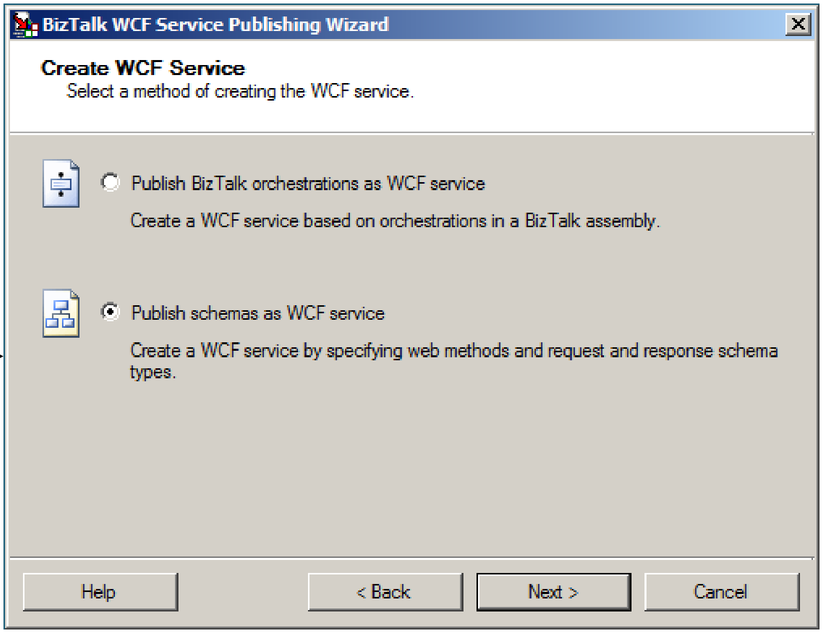
Using the WCF adapter to receive

- **Specify the correct WCF adapter as the receive location transport**
  - In-process options include TCP, MSMQ and Named Pipes
  - HTTP hosting is handled by IIS
  - For HTTPAddress (URI) properties, specify the virtual directory plus .svc file name (/Orders/Orders.svc)
- **The adapter functionality is handled in WCF code**
  - WCF implements SOAP protocol
  - Adapter publishes the body to the MB

Generating the SVC endpoint

- **You can generate the SVC endpoint for the WCF HTTP adapters**
  - Use the *BizTalk WCF Service Publishing Wizard*
- **Wizard allows you to generate SVC from two starting points**
  - An *XSD schema*
  - A BizTalk orchestration

Publish schema as Web service



Create WCF Service
Select a method of creating the WCF service.

( ) Publish BizTalk orchestrations as WCF service
Create a WCF service based on orchestrations in a BizTalk assembly.

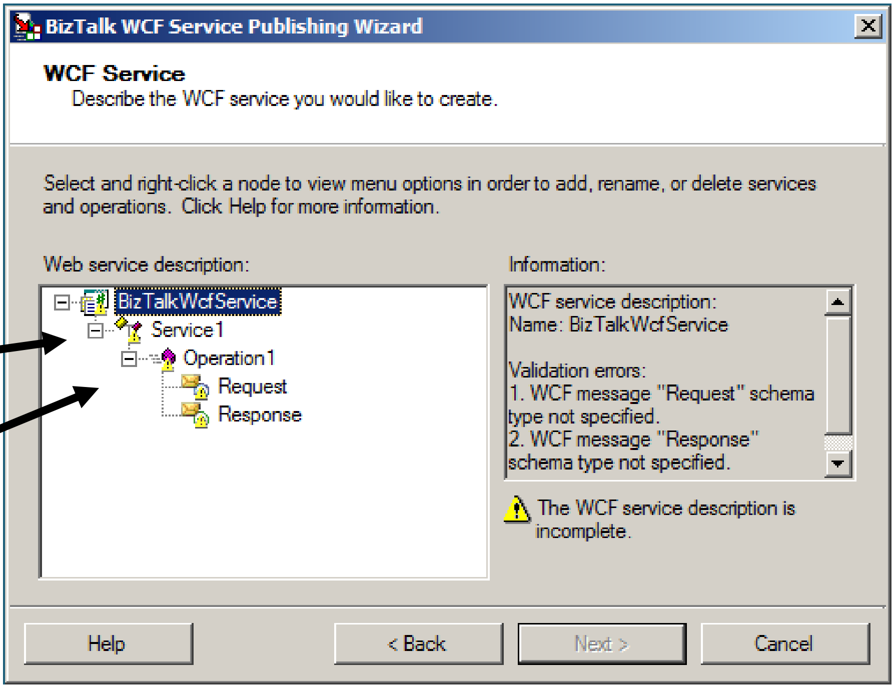
(●) Publish schemas as WCF service
Create a WCF service by specifying web methods and request and response schema types.

Publishing schema as a service

- **When you publish a schema as a service**
  - There isn't enough information in the schema to produce a full service contract (service name, operation names, etc)
  - You must provide these additional details via the wizard

Specify service name

Specify operation name and exchange pattern, map to schema types



WCF Service
Describe the WCF service you would like to create.

Select and right-click a node to view menu options in order to add, rename, or delete services and operations. Click Help for more information.

WCF service description:
Name: BizTalkWcfService

Validation errors:
1. WCF message "Request" schema type not specified.
2. WCF message "Response" schema type not specified.

The WCF service description is incomplete.

BizTalk isolated hosts

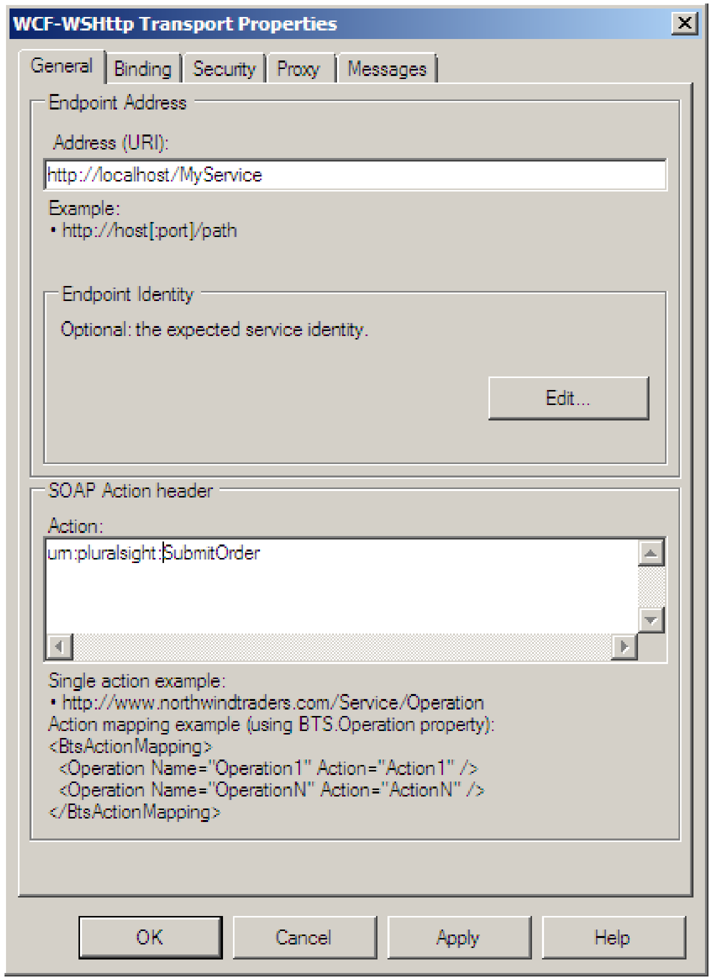
- **HTTP adapters must run in a *BizTalk Isolated Host***
  - An external process not managed by BizTalk (e.g., IIS processes)
  - Offers isolation protection and security
- **Configure WCF services to run in their own IIS *AppPool***
  - Create a new Application Pool in IIS
  - Define it to run under an identity with limited privileges
  - Ensure that the user is a member of BizTalk Isolated Host Users
  - Assign the new virtual directory to run in the new app pool
  - Restart IIS and you should be set

WCF adapter for sending

- **In BTS 2004, you had to use orchestrations to invoke services**
  - No support for messaging-only Web service integration
- **Since BizTalk 2006, invoke services directly from *send ports***
- **You configure the WCF adapter with the following**
  - Web Service URL
  - Authentication type and credentials / include WS-Security
  - Proxy server details
  - Binding and behavior configuration
  - Message processing directives

Configuring WCF transport for sending

Specify address and action



WCF-WSHttp Transport Properties

General | Binding | Security | Proxy | Messages

Endpoint Address

Address (URI): http://localhost/MyService

Example:
• http://host[:port]/path

Endpoint Identity

Optional: the expected service identity.

Edit...

SOAP Action header

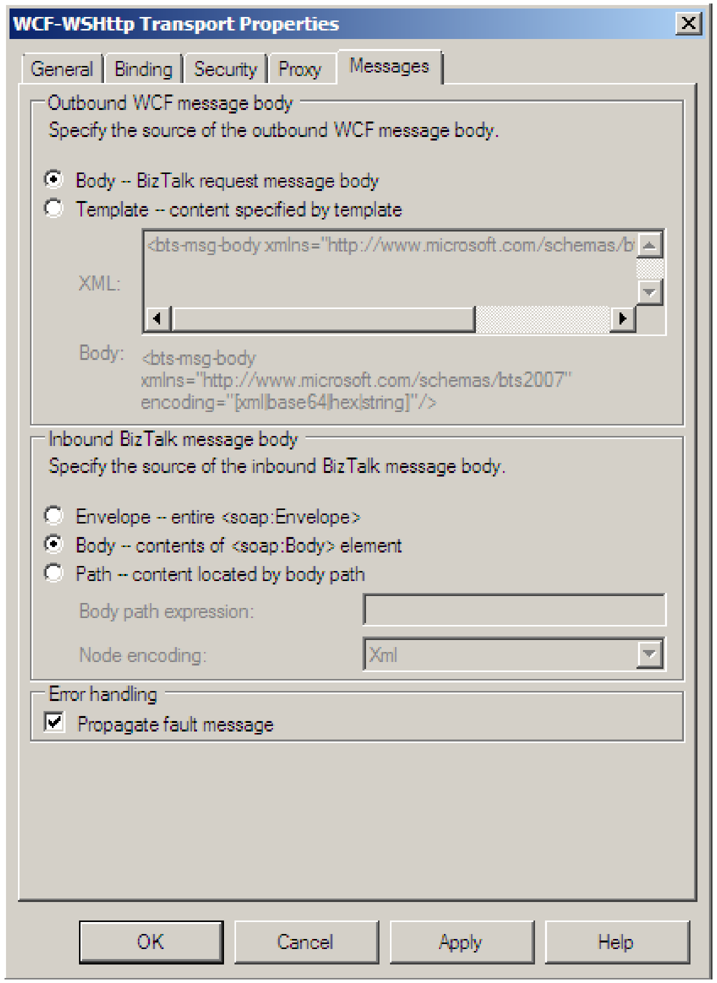
Action: urn:pluralsight:SubmitOrder

Single action example:
• http://www.northwindtraders.com/Service/Operation
Action mapping example (using BTS.Operation property):

```
<BtsActionMapping>
  <Operation Name="Operation1" Action="Action1" />
  <Operation Name="OperationN" Action="ActionN" />
</BtsActionMapping>
```

OK | Cancel | Apply | Help

Specify message details



WCF-WSHttp Transport Properties

General | Binding | Security | Proxy | Messages

Outbound WCF message body

Specify the source of the outbound WCF message body.

● Body -- BizTalk request message body
○ Template -- content specified by template

XML: <bts-msg-body xmlns="http://www.microsoft.com/schemas/b

Body: <bts-msg-body xmlns="http://www.microsoft.com/schemas/bts2007" encoding="[xml|base64|hex|string]"/>

Inbound BizTalk message body

Specify the source of the inbound BizTalk message body.

○ Envelope -- entire <soap:Envelope>
● Body -- contents of <soap:Body> element
○ Path -- content located by body path

Body path expression:

Node encoding: Xml

Error handling

☑ Propagate fault message

OK | Cancel | Apply | Help

Adding a Service Reference

- **Add a service reference to generate message schema**
  - *Add Generated Items / Consume WCF Service* on project menu
  - Enter the URL for the WSDL you wish to consume
- **This imports the types found in the referenced WSDL and XSD**
  - Orchestration created with port and message types
  - Schemas generated for messages
  - Binding file created for send port
  - Local definitions stored in *Reference.map*

BizTalk orchestrations

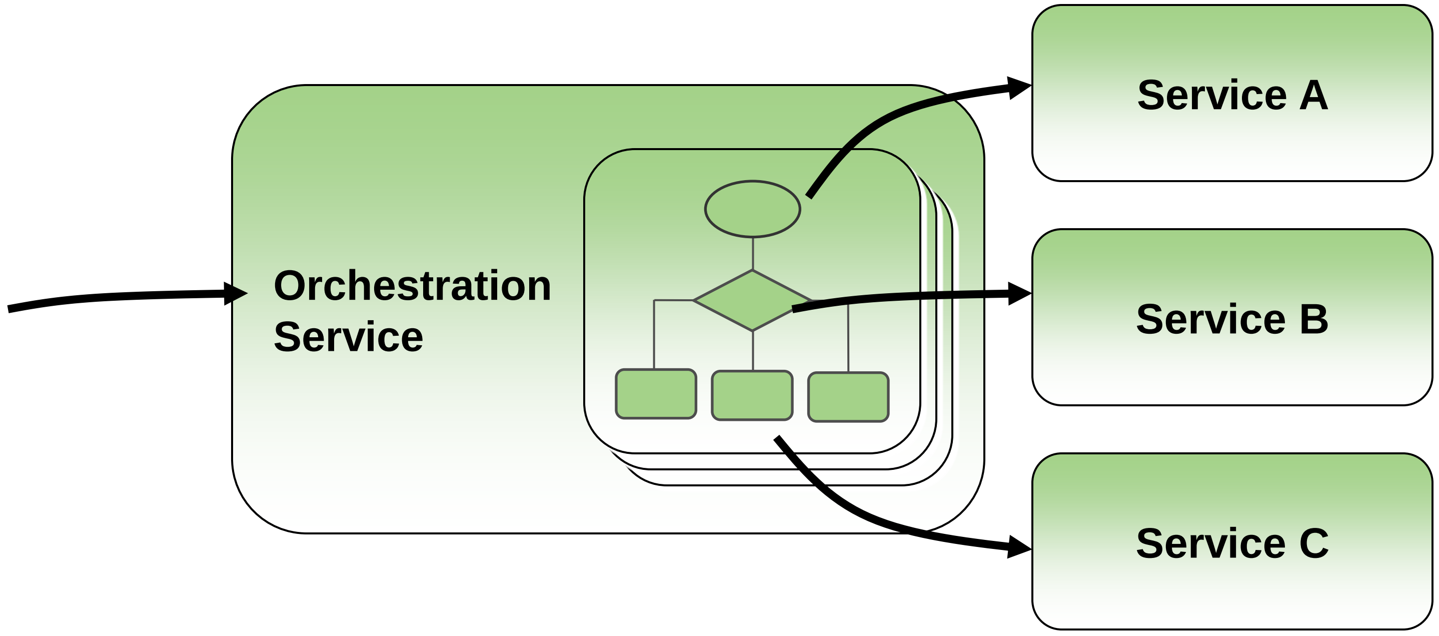
- **Orchestrations need the ability to consume Web services**
  - The orchestration can then become an aggregate service
- **You can easily publish an orchestration as a Web service**
  - Orchestrations can be consumed by any WS-aware app
- **Understanding how orchestrations map to WSDL is vital**

Mapping orchestrations to WSDL

Name	WSDL	Orchestration
Types	WSDL documents import XSD types in order to define the messages that will define the service inputs and outputs.	BizTalk orchestrations reference XSD types in order to define the input and output messages bound to send/receive ports.
Messages	WSDL messages define the particular elements from the imported XSD that will flow in and out of the service.	BizTalk message variables specify exactly which public element from the referenced XSD will be used by the send/receive ports.
Port Types	WSDL portType elements define groups of logical operations, which consist of input and output messages (interface).	BizTalk Port Types define groups of logical operations, which consist of input and output messages (interface).
Bindings	WSDL binding elements define transport and protocol details for a portType.	BizTalk orchestrations are "bound" to physical ports, which include adapter and pipeline configurations to control communication details.
Services	WSDL service elements define ports. Ports define the endpoints (addresses) for communication. Each port is mapped to a portType and binding.	BizTalk orchestrations define ports. BizTalk Server ports define the logical and physical I/O mechanisms of an orchestration, Each logical port is associated a portType and is bound.

BizTalk and Web services

- **BTS automates the mapping with support for Web services**
  - Orchestrations can *consume* external Web services
  - Orchestrations can be *published* as Web services



Consuming Web services

- **You can consume a Web service from an orchestration**
  - *Add Service Reference*
  - Create port instances from port types
  - Define message variables for the required messages
  - Construct the messages needed to call the service
  - Add Send/Receive shapes and connect to ports
- **Service types imported via their WSDL definitions**
  - BizTalk automates conceptual mapping

Create and configure a Port

- **A *Port* represents a connection to a service endpoint**
  - Right click on the Port Surface and select *New Configured Port*
  - Choose *Use an existing Port Type*
  - Select the Port Type imported using Consume Service Reference
  - Finish the wizard
- **Later you'll connect Send/Receive shapes to the Web Port**

Calling the service

- ***"Calling the service*“ consists of sending/receiving messages**
  - You send them to and from the Web Port
- **Before you can do this, you need *message variables***
  - They should map to an imported Message Type
  - Using the imported message types ensures conformity
- **Then simply use Send/Receive shapes to use the Web Port**

Dealing with void and one-way

- **If the service operation takes no parameters**
  - You must still construct the required empty message
  - Use *Construct Message* without a transform/assignment shape
- **If the service operation is truly one-way (no response message)**
  - Don't use a Receive shape
- **However, if the operation returns void but it's not one-way**
  - You'll still need a Receive shape (consider SOAP faults)

Handling SOAP faults

- **Web services return exceptions via *SOAP fault* elements**
  - Represents anything that goes wrong during message exchange
- **Handle SOAP faults by using orchestration *Scope* shapes**
  - Add an exception handler to catch the fault exception type
  - Use *System.Web.Services.Protocols.SoapException*
- **Possible to catch typed faults as well**
  - FaultContract used in WCF service
  - Add Exception handler for the custom fault type
  - Modify send port properties to extract fault message using Path

Throwing SOAP faults

- **You can return SOAP faults to the orchestration consumers**
  - The infrastructure does most of the work
- **Create a fault message on the operation**
  - Right=click the operation and choose “Add Fault Message”
- **Send a message to the *Fault* on the Web Port**
  - The message you send will be transmitted in the *fault detail*
  - You can use a complex messages or just simple strings
  - Fault will not appear on one-way operations
- **Update receive location properties (messages tab)**
  - Check the box to *Include exception details in fault*

Configuring WCF transport properties

- **The WCF adapters provide message context properties**
  - These can be used to configure aspects of WCF behavior
  - You set these on the message themselves, before sending
  - Most properties can only be used on *Dynamic* send ports

```
SubmitCustomerMsg(WCF.OpenTimeout) = "00:01:00";
SubmitCustomerMsg(WCF.Action) = "urn:Pluralsight:SubmitCustomer";
```

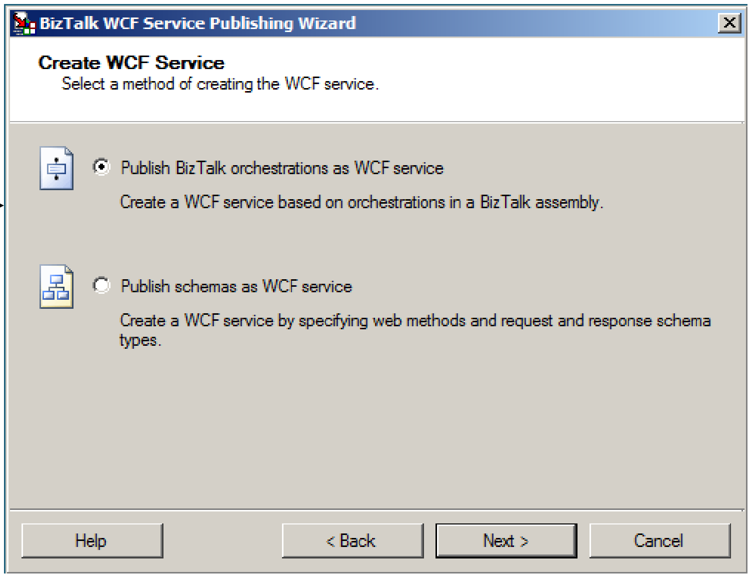
- **You can set the Web service address dynamically**

```
CustomerService(Microsoft.XLANGs.BaseTypes.Address) =
  "http://PayrollCompany/RegisterEmployee.asmx";
CustomerService(Microsoft.XLANGs.BaseTypes.TransportType) =
  "WCF-BasicHttp";
```

Publishing orchestrations as services

- **You can *publish* orchestrations as Web services**
  - Makes them available to any SOAP-aware app
- **Use the *Web Services Publishing Wizard* to generate ASMX**
- **Use the WCF Service Publishing Wizard to generate SVC**

Publish orchestration as Web service



Create WCF Service
Select a method of creating the WCF service.

[✓ Publish BizTalk orchestrations as WCF service]
Create a WCF service based on orchestrations in a BizTalk assembly.

Publish schemas as WCF service
Create a WCF service by specifying web methods and request and response schema types.

Publishing wizard

- **The WCF Service Publishing Wizard produces the following**
  - Creates an SVC endpoint for each ***public*** receive port (can be combined)
  - Automatically maps orchestration Port Types to WSDL/XSD
  - Creates a virtual directory and deploys project/code
  - Can also create the receive location automatically
- **If you need to configure the WCF receive location manually**
  - Create a new receive location, specify correct WCF transport
  - Specify the service endpoint details
- **Create metadata for in-process hosted WCF endpoints**
  - Manually create receive location
  - Use wizard to create metdata endpoint for service

Summary

- BizTalk provides sophisticated Web services integration
- The WCF adapters integrate with WCF framework
- Allow you to receive SOAP messages into the MB, and invoke services
- Orchestrations consume WCF services via Ports
- BizTalk provides a wizard for generating the service endpoints

References

- **Using the WCF adapters in BizTalk Server Whitepaper**
  - http://msdn.microsoft.com/en-us/library/bb967002(BTS.10).aspx
- **WCF Adapters in BizTalk Server**
  - http://msdn.microsoft.com/en-us/library/bb246032(BTS.10).aspx
- **WCF Adapter FAQ**
  - http://msdn.microsoft.com/en-us/library/dd560703(BTS.10).aspx