Web Services

Understanding how to integrate BizTalk with SOAP

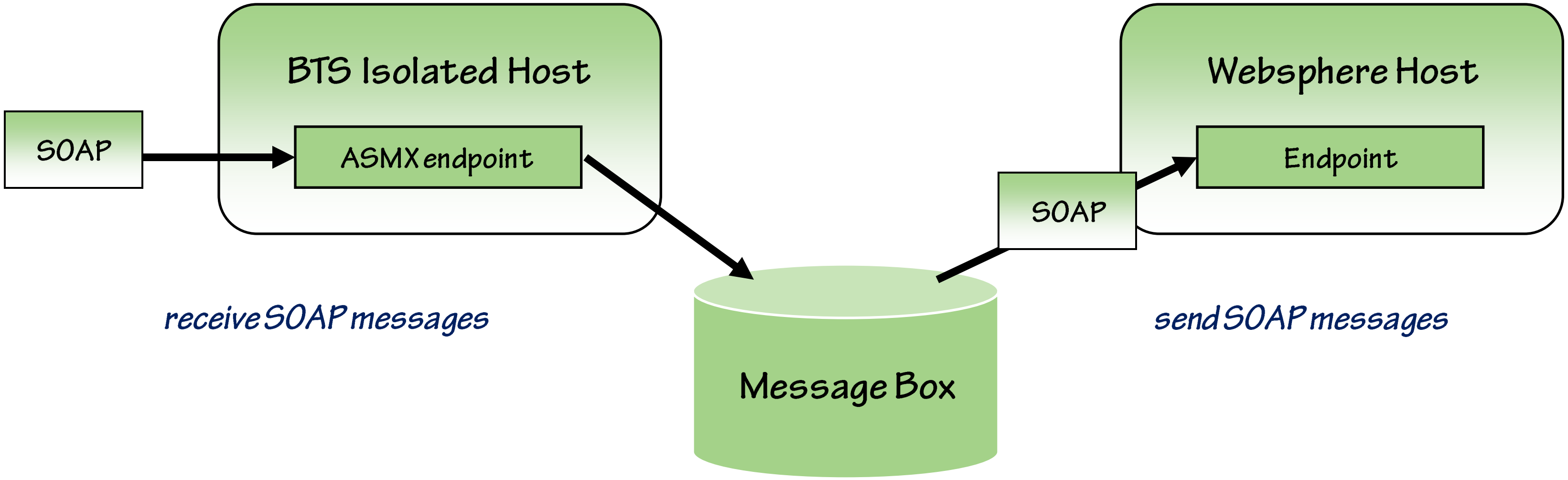
pluralsight
see what you can learn

Outline

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

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**



receive SOAP messages

send SOAP messages

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

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

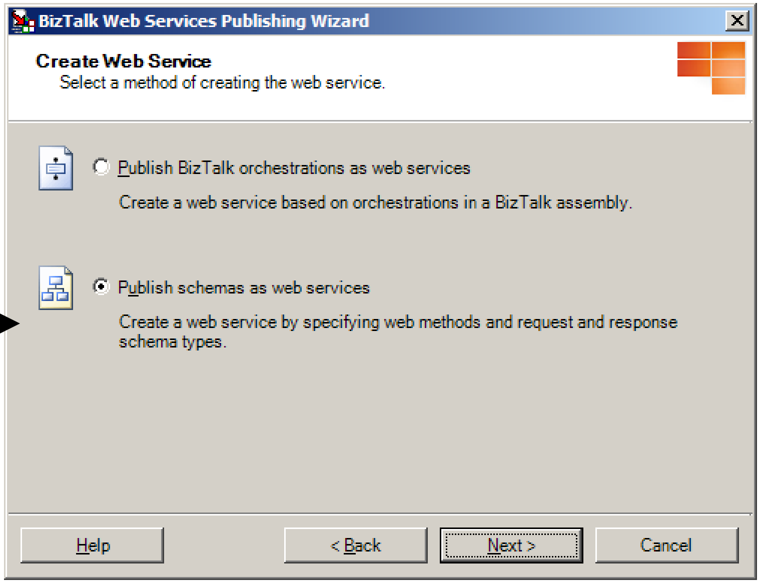
Using the SOAP adapter to receive

- **Specify the SOAP adapter as the receive location transport**
  - This tells BizTalk you'll be receiving messages via SOAP
  - For the Address (URI) properties, you specify the virtual directory plus .asmx file name (/Orders/Orders.asmx)
- **The adapter functionality is provided by the *ASMX code***
  - ASMX code implements SOAP protocol
  - Then it publishes the body to the MB
  - Assumes its on the same server as MB

Generating the ASMX code

- You can generate the ASMX code to use as a SOAP adapter
  - Use the *BizTalk Web Services Publishing Wizard*
- Wizard allows you to generate ASMX from two starting points
  - An *XSD schema*
  - A BizTalk orchestration

Publish schema as Web service



BizTalk Web Services Publishing Wizard

Create Web Service

Select a method of creating the web service.

○ Publish BizTalk orchestrations as web services

Create a web service based on orchestrations in a BizTalk assembly.

● Publish schemas as web services

Create a web service by specifying web methods and request and response schema types.

BizTalk isolated hosts

- **SOAP/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 ASMX services to run in their own IIS *AppPool***
  - Create a new Application Pool in IIS 6.0
  - 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

SOAP adapter for sending

- **In BTS 2004, you had to use orchestrations to invoke services**
  - No support for messaging-only Web service integration
- **With 2006, you can invoke services directly from *send ports***
- **You configure the SOAP adapter with the following**
  - Web Service URL
  - Authentication type and credentials
  - Proxy server details
  - Assembly containing a *proxy class* for calling the service

WCF adapters for sending

- **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

BizTalk orchestrations

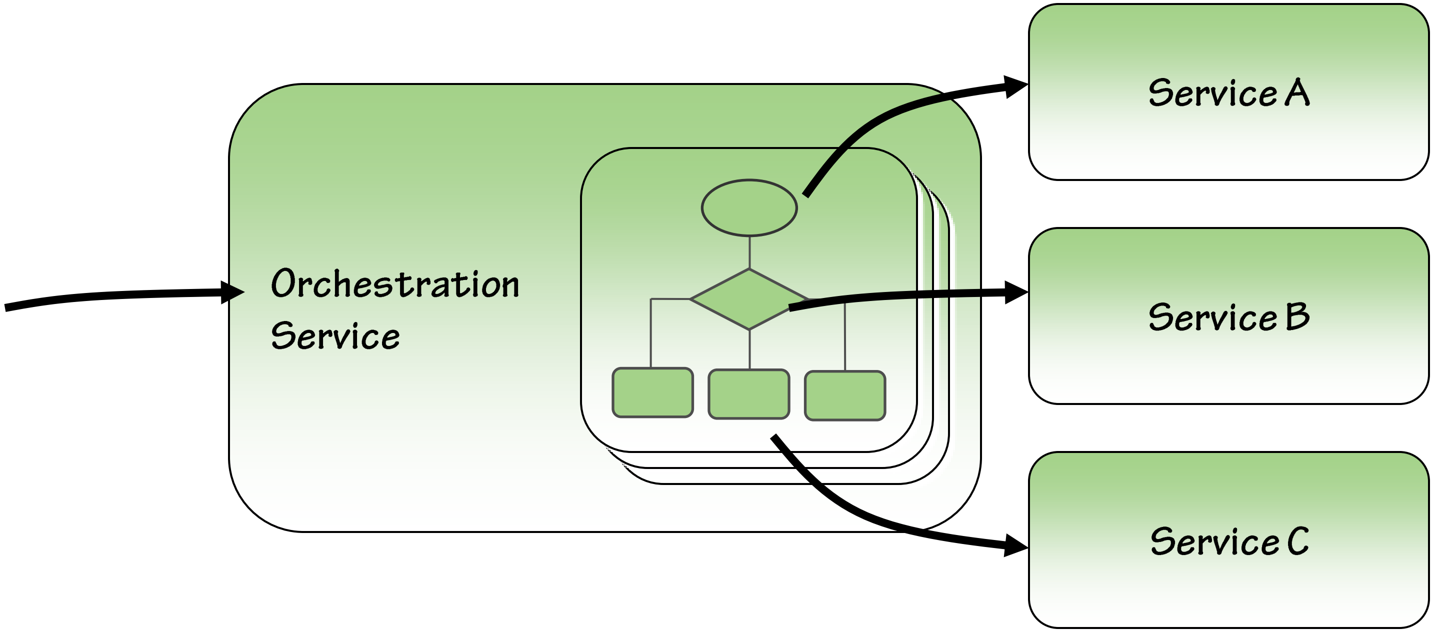
- **Orchestrations need the ability to consume Web services**
  - The orchestration can then become an aggregate service
- **You can easily publish an orchestration as a Web service**
  - Orchestrations can be consumed by any WS-aware app
- **Understanding how orchestrations map to WSDL is vital**

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**
  1. Add a *Web reference* to your orchestration project
  2. Create a *Web Port* for the referenced Web service
  3. Define message variables for the required messages
  4. Construct the messages needed to call the service
  5. Add Send/Receive shapes and connect to Web Port
- **Service types imported via their WSDL definitions**
  - BizTalk automates conceptual mapping

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*

Configuring SOAP transport properties

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

```
SubmitCustomerMsg(SOAP.ClientConnectionTimeout) = 20000;
```

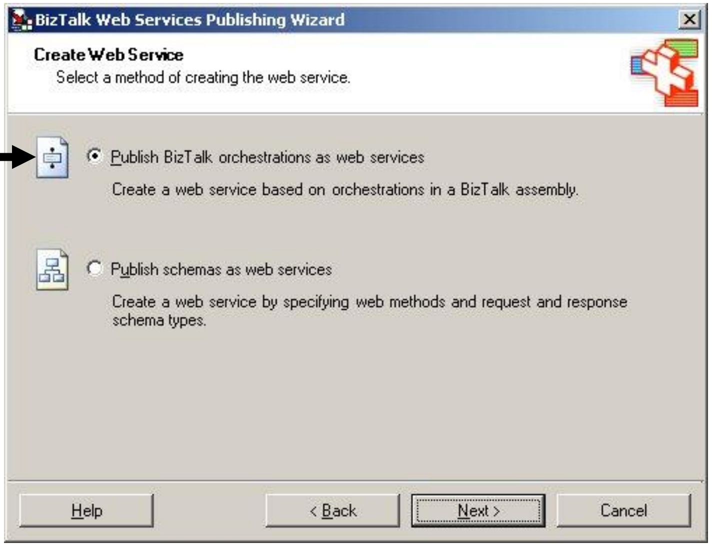
- **You can set the Web service address dynamically**

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

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



BizTalk Web Services Publishing Wizard

Create Web Service

Select a method of creating the web service.

● Publish BizTalk orchestrations as web services

Create a web service based on orchestrations in a BizTalk assembly.

○ Publish schemas as web services

Create a web service by specifying web methods and request and response schema types.

Publishing wizard

- **The Web Services Publishing Wizard produces the following**
  - □ Creates an ASMX endpoint/operation for each ***public*** receive port
  - □ 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 SOAP receive location manually**
  - □ Create a new receive location, specify SOAP transport
  - □ And specify the service endpoint details

Throwing SOAP faults

- **You can return SOAP faults to the orchestration consumers**
  - The infrastructure does most of the work
- **Simply 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

Summary

- BizTalk provides sophisticated Web services integration
- The SOAP adapter integrates with ASMX framework
- Allows you to receive SOAP messages into the MB, and to invoke services using messages being sent out
- Orchestrations consume ASMX Web services via Web Ports
- BizTalk provides a wizards for generating the service endpoints
- R2 release includes WCF adapters with increased standards support

References

- **BizTalk Server 2004 and Web Services Whitepaper**
  - http://msdn.microsoft.com/library/default.asp?url=/library/en-us/BTS_2004WP/html/5cab05ab-6848-4f6c-8d11-9abc4dd1d1fa.asp
- **BizTalk Support for Web Services FAQ**
  - http://blogs.msdn.com/biztalk_server_team_blog/archive/2006/04/12/575207.aspx
- **WCF Adapters in BizTalk Server R2**
  - http://www.microsoft.com/downloads/details.aspx?familyid=a976dc7d-2296-4f88-be4d-0d314fca9e59&displaylang=en&tm
- **Pluralsight's BizTalk Wiki**
  - http://pluralsight.com/wiki/default.aspx/Aaron/TheBiztalkWiki.html