

BizTalk Server 2010 Developer Immersion

Course Introduction

Application Developers

- 1-2 years of experience developing enterprise applications
- Background systems integration
- Familiarity with Visual Studio and .NET development
- General knowledge of XML
- No prior BizTalk Server experience is expected

Tell us about yourself:

- Your name
- Your company and job role
- Where are you from?
- Years of development experience (.NET, Java, etc.)
- Any BizTalk Server development experience
- Your goals and expectations for this course

Virtual PC lab image

- Microsoft Windows Server 2008
 - .NET Framework 4.0
- Microsoft Visual Studio 2010
- Microsoft SQL Server 2008
- Microsoft BizTalk Server 2010
- Windows SharePoint Services 3.0
- Microsoft Office 2007

Student computer logon

- Logon name: Administrator
- Password: pass@word1

Facility Information

- Class hours
- Building hours
- Parking
- Restrooms
- Meals
- Phones
- Smoking
- Recycling

For the best classroom experience...

- When not speaking, please mute your phone
 - Please do NOT use hold – many phones have hold music
- Please ask questions and provide feedback
 - GoToMeeting chat is NOT the preferred method
 - Best option is to use your voice
- Class starts at 7:00 PST
- Technical support: Monique – 425.256.2573

Agenda

- Module 1: Introduction to BizTalk Server
- Module 2: Creating Schemas
- Module 3: Creating BizTalk Maps
- Module 4: Deploying a BizTalk Project
- Module 5: Routing BizTalk Messages
- Module 6: Creating Pipelines
- Module 7: Integrating with Adapters
- Module 8: Creating BizTalk Orchestrations

Continued...

Agenda

- Module 9: Advanced Orchestration Design
- Module 10: Creating Transactions
- Module 11: Deploying & Managing BizTalk Applications
- Module 12: Integrating with Services
- Module 13: Integrating with Business Rule Policies
- Module 14: Enabling Business Activity Monitoring
- Module 15: Integrating using Parties and EDI

Materials

- Student workbook
- Student PDF
- Course evaluation

Demo: Using Microsoft Virtual PC

Demonstration

- Open the Virtual PC Console
- Start a Virtual Machine
- Logon to a virtual machine
- Use the course PDF

Introduction to BizTalk Server

- Lesson 1: What Is BizTalk Server?
- Lesson 2: What's New in BizTalk Server 2010?
- Lesson 3: The BizTalk Server Development Environment

Lesson 1: What is BizTalk Server?

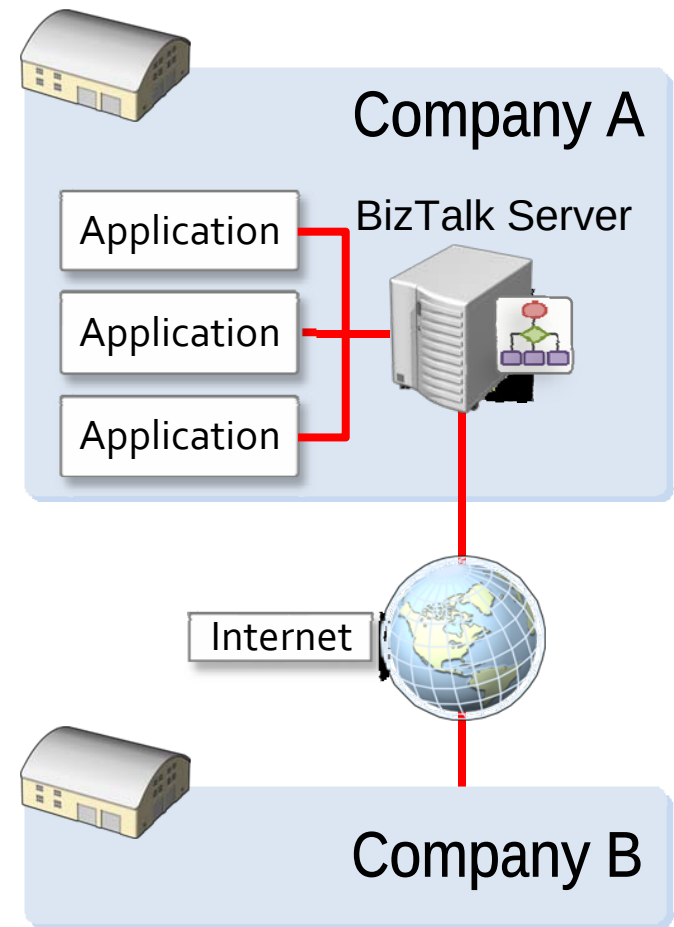
In this lesson, you will learn about...

- BizTalk Integration Services and Tools
- What Problems Does BizTalk Solve?
- Scenario: What Is BizTalk Integration?
- What Is Business Process Automation?
- BizTalk Core Message Flow
- BizTalk Job Roles and Tools
- Available Editions of BizTalk Server

BizTalk Integration Services and Tools

Integration Services and Tools

- Messaging and orchestration
- Application development tools
- Message transformation
- Business rules engine
- Default tracking
- Service integration
- Custom tracking (BAM)
- B2B integration using EDI

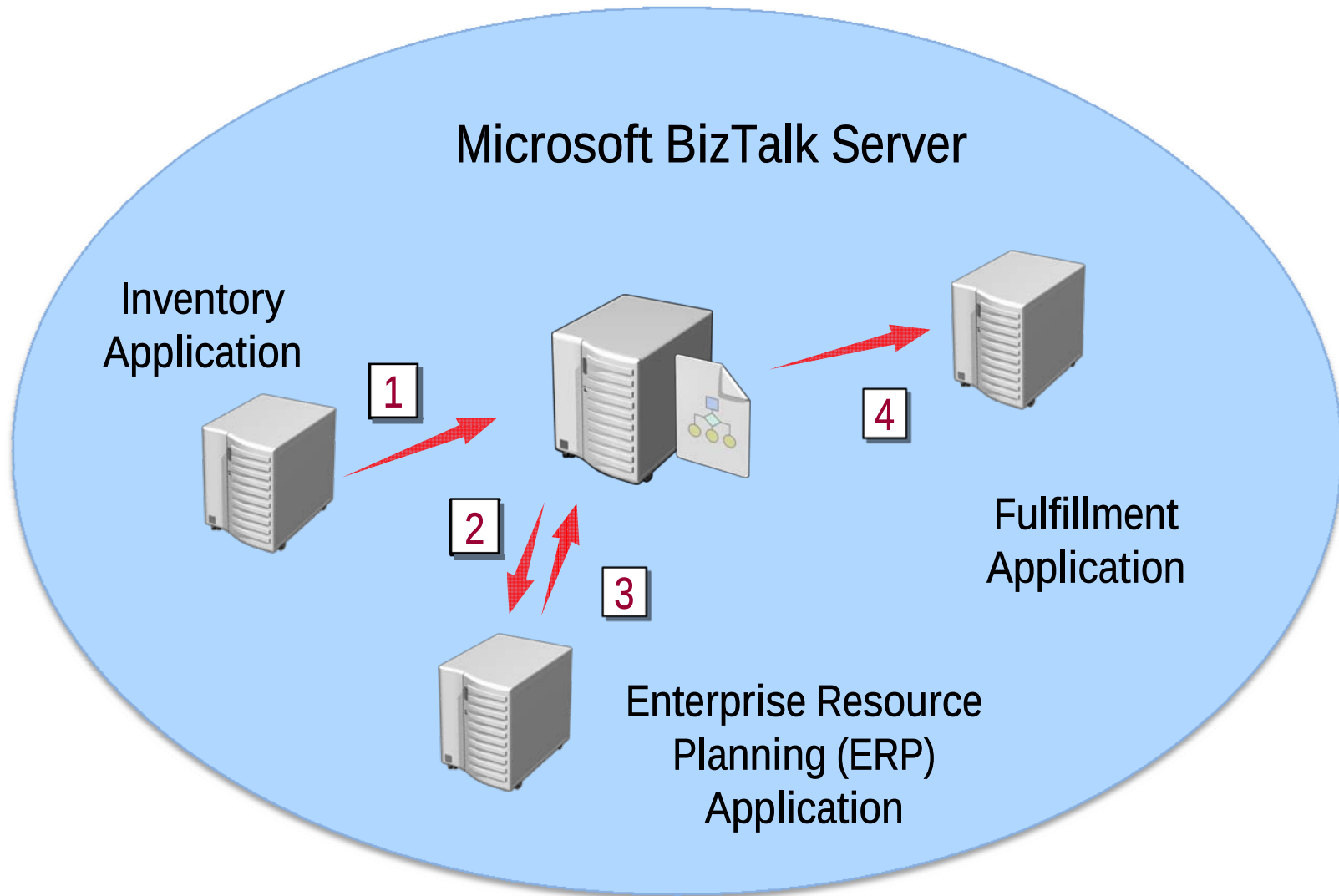


What Problems Does BizTalk Solve?

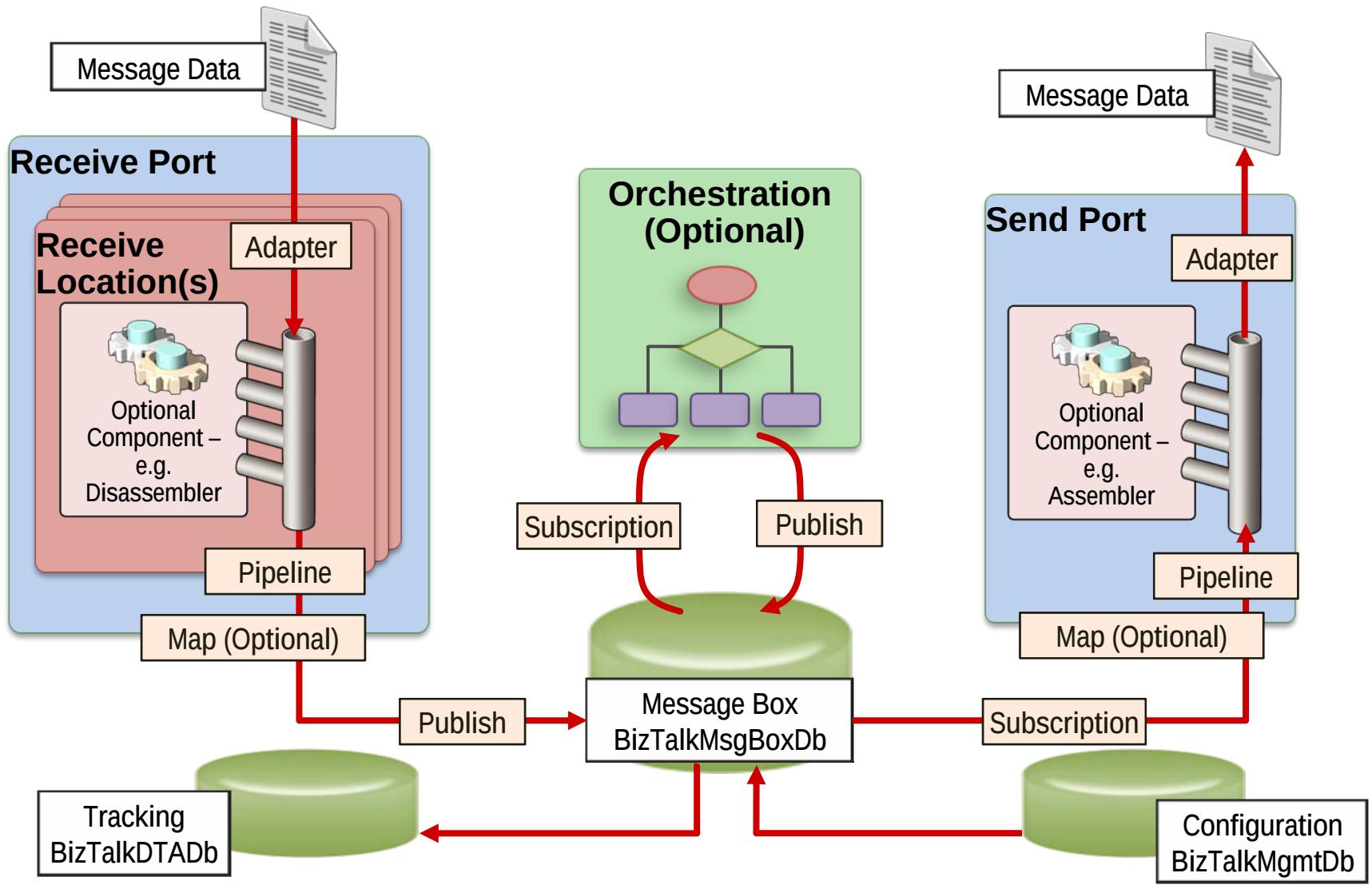
Common Complaints:

- Disparate applications
- Programming overruns
- Time-consuming deployment
- Dissimilar reports
- Modification difficulties
- Lack of set procedures
- Limited tracking
- Changing partners

What Is BizTalk Server Integration? (Scenario)



BizTalk Message Flow



BizTalk Job Roles

Information Workers



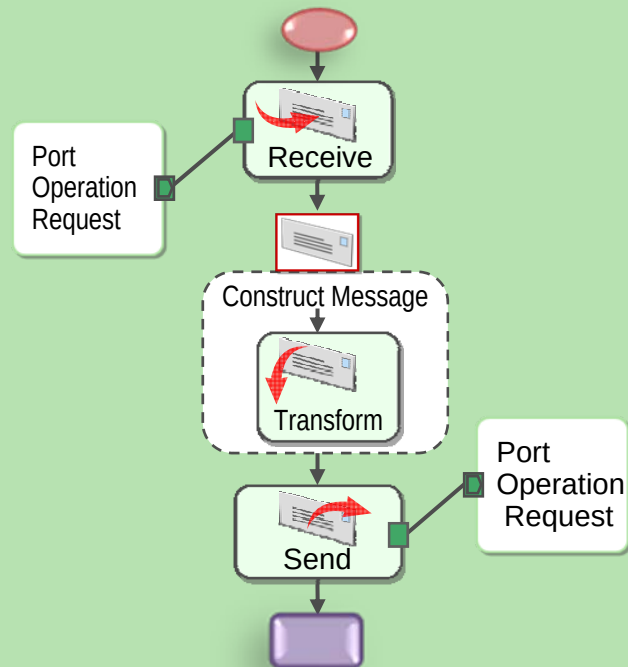
Design business processes, define business rules, and analyze business activity



Developers



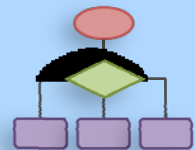
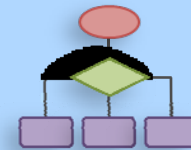
Build and test business process and integration applications



IT Pros



Deploy applications and manage physical environment



BizTalk Server Editions

Edition	Description
Enterprise	• For medium to large organizations • Unlimited internal applications • Multiple processors and service clustered deployments
Standard	• For small organizations • Up to five internal applications • Two processors on a single server
Developer	• For development and test environments • Complete EAI, B2B, and BPM functionality • Deploy from a development environment to a production environment
Branch	• For edge deployments associated with Enterprise BizTalk deployment • Single server with up to two processors • Single MessageBox database • Includes BizTalk RFID and Host Integration Server

For a complete description of editions, please visit: <http://www.microsoft.com/biztalk/howtobuy/default.msp>

Lesson 2: What's New in BizTalk 2010?

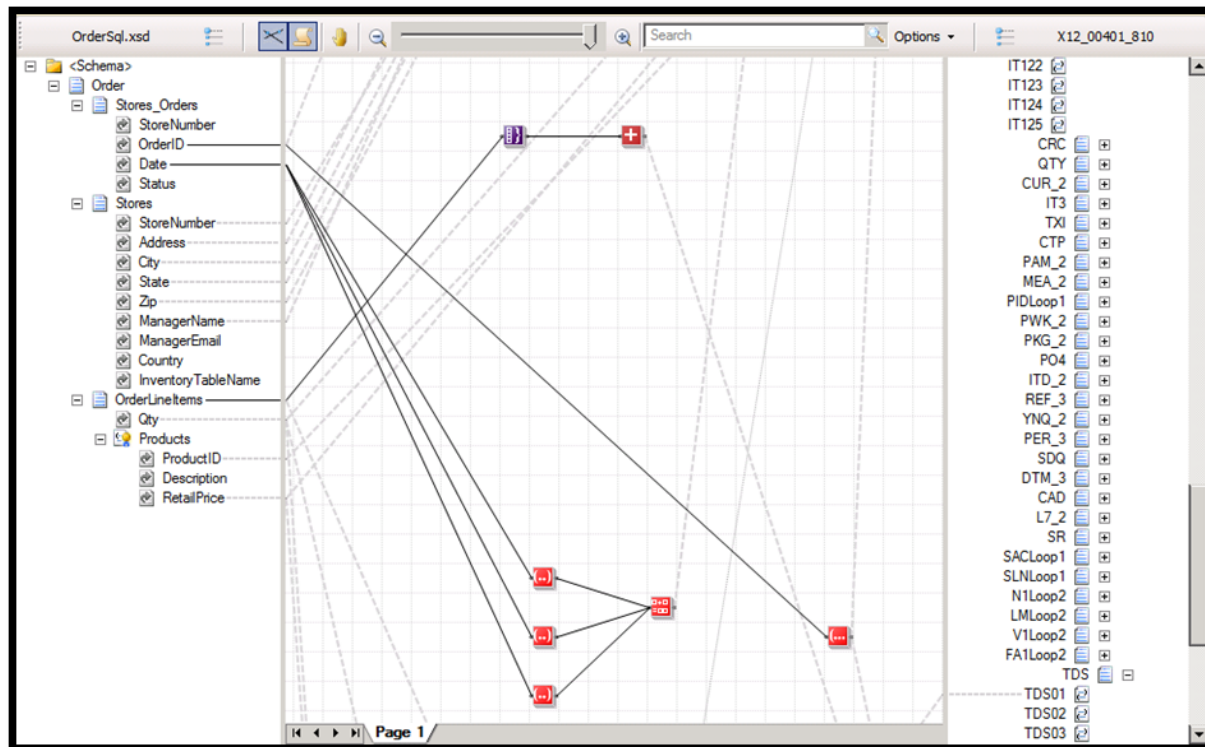
In this lesson, you will learn about...

- Enhanced BizTalk Mapper
- Adapter and Integration Changes
- Improved Trading Partner Management
- BizTalk Server Settings Dashboard
- New OpsMgr Management Pack
- Updated Platform Support
- Demo: Tour Around BizTalk 2010

Enhanced BizTalk Mapper

New Features:

- Copy, Cut, and Paste
- Optimized display of links
- Automatic highlighting and focusing
- Relevance view
- Improved functoid configuration



Adapter and Integration Changes

New FTP Adapter Features

- FTP over SSL (FTPS)
- Receive from read-only locations

WCF-SQL Adapter

- Replaces the SQL adapter

Deprecated Features

- SOAP adapter
- BizTalk Web Services Publishing Wizard

Improved Trading Partner Management

Major redesign of trading partner management

- Parties
- Business Profiles
- Trading Partner Agreements

Continues support for EDI and AS2

Parties and Business Profiles					
			Name		
[-]		1234			
		Supplier			
[-]		1235			
		Supplier			
[-]		1236			
[-]		1237			
[-]		Northwind Traders			
		DefaultProfile			
[-]		Woodgrove			
		Purchaser			
Agreements of : Woodgrove(Purchaser)					
	Agreement	Party1	Party1's Profile	Party2	Party2's Profile
	Purchasing Agreement - 1234	Woodgrove	Purchaser	1234	Supplier
	Purchasing Agreement - 1235	Woodgrove	Purchaser	1235	Supplier

Settings previously configured in many places

- BizTalk Server Administration Console
- Manual configuration of registry settings
- Hacking the BizTalk configuration file
- Manual editing of configuration database

Settings Dashboard provides a single interface

- Improved discoverability of potential settings
- Faster configuration of settings
- Ability to import and export settings
- Holistic view of performance tuning for a BizTalk environment

OpsMgr Management Pack

- Separate views for IT administrators and BizTalk administrators
- Optimized discovery of artifacts
- Fixed the discovery of relationships between artifacts
- Visual representation of health status
- Provides diagnostic support for common BizTalk errors

Supported Operating Systems:

- Windows Server 2008 R2
- Windows Server 2008 SP2
- Windows 7
- Windows Vista SP2

Supported Database Servers

- SQL Server 2008 SP1
- SQL Server 2008 R2

Development Platform

- Visual Studio 2010

Lesson 3: The Development Environment

In this lesson...

- System Requirements
- Tools for Developers
- BizTalk Administration Console
- Demonstration: The Visual Studio Development Environment

System Requirements

Minimum Requirements

- 1 GHz CPU
- 1 GB of RAM
- 6 GB of available hard disk space

Recommended

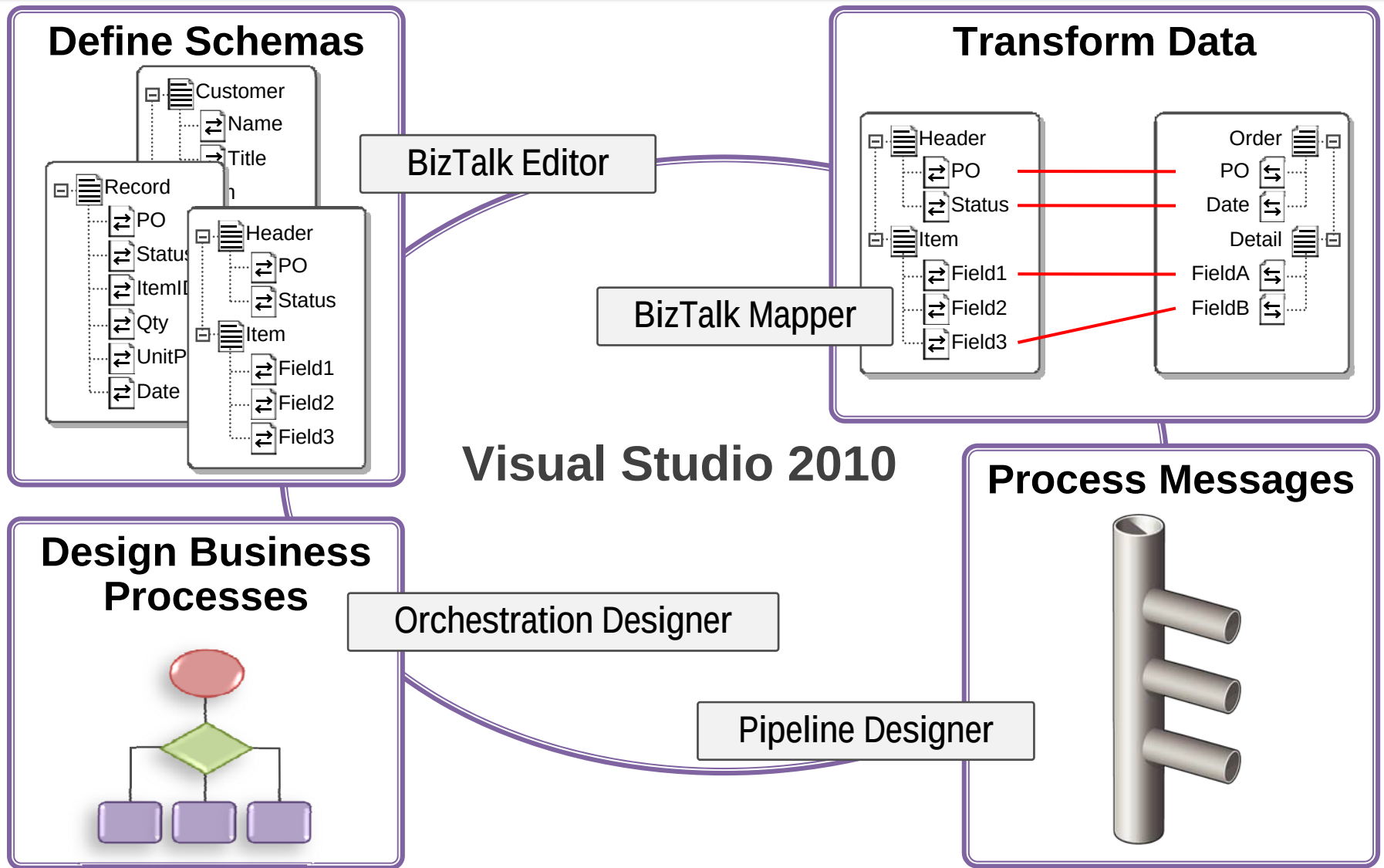
- 2-GHz or faster CPU (64-bit)
- 2 GB or more of RAM
- 60 GB or more of available hard disk space

Developer Workstation



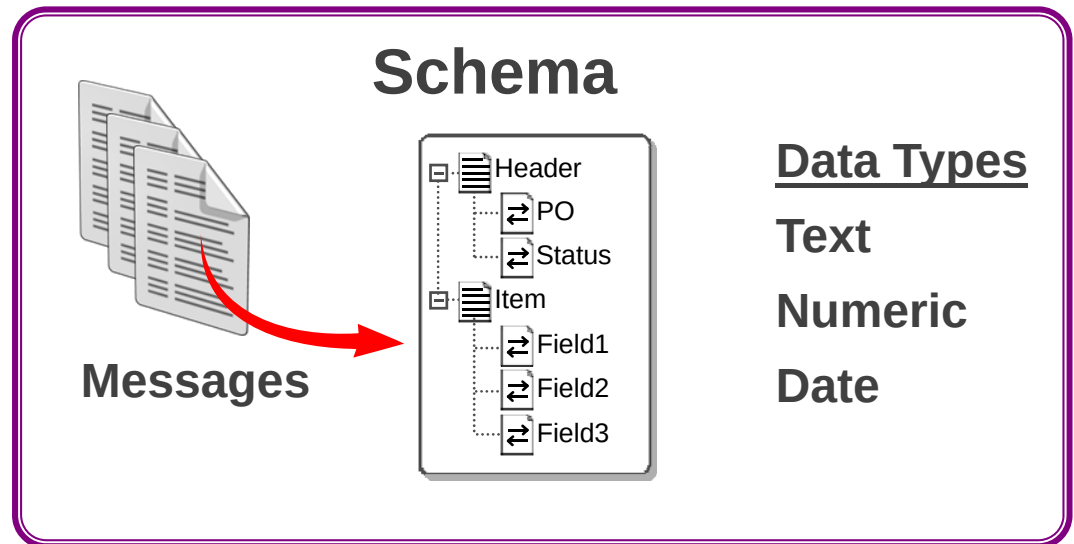
Windows Server 2008 or
Windows 7

Developer Tools



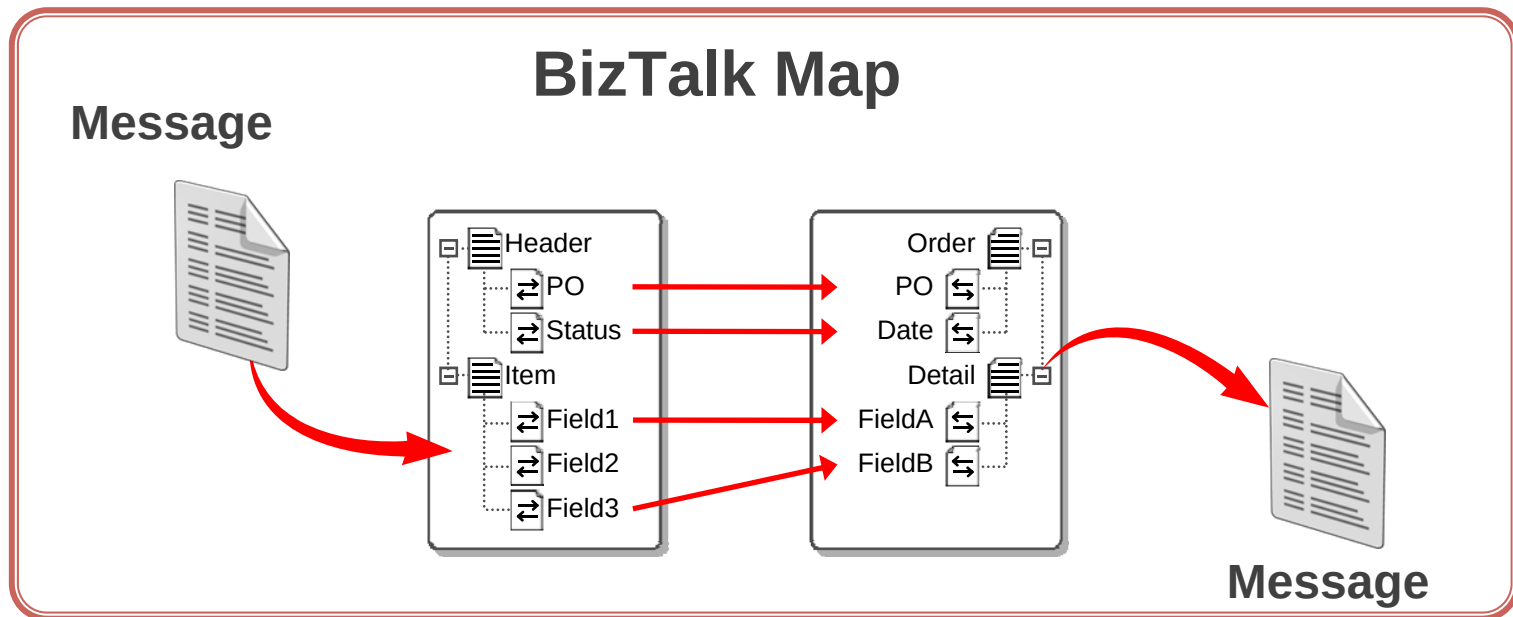
Create a schema to define...

- Elements
- Attributes
- Order of tags
- Mandatory fields
- Repeating fields



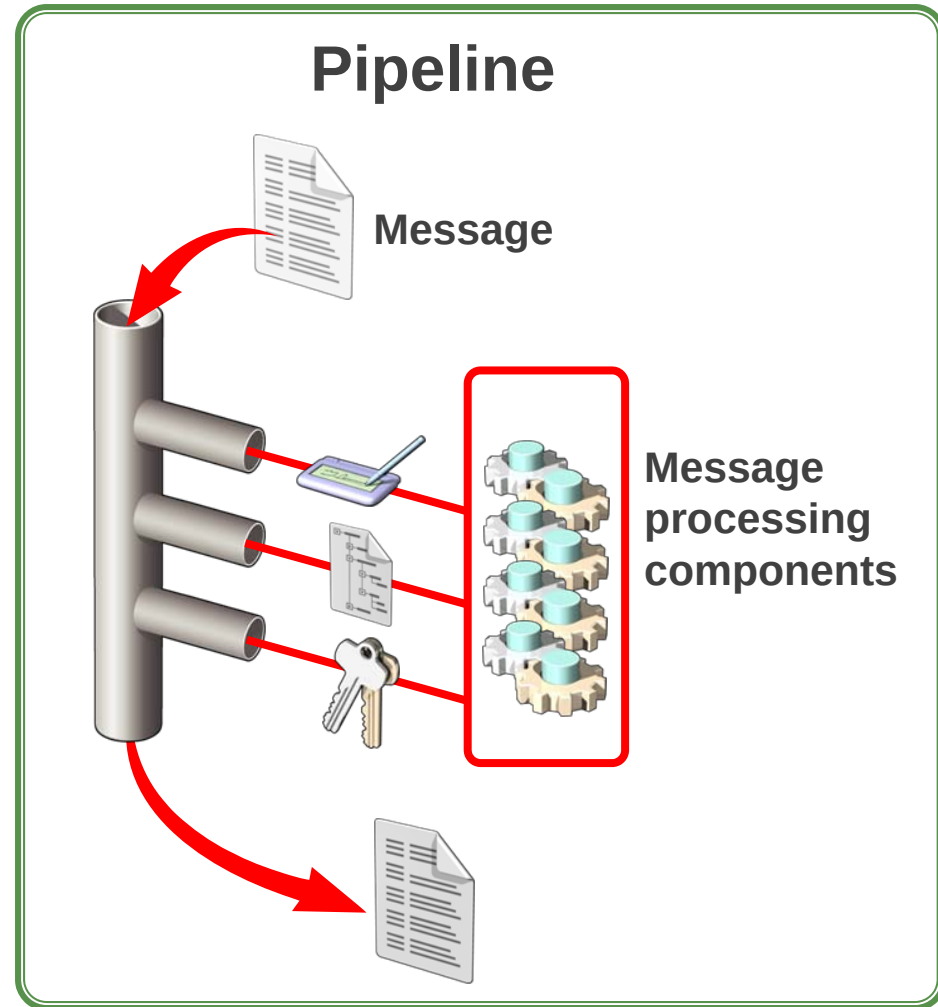
Create a map to...

- Average data from a looping record
- Convert character data to its ASCII format
- Add or subtract data from one or more records



Create a pipeline to...

- Decrypt/encrypt
- Decode/encode
- Validate a document against a schema
- Verify digital signatures

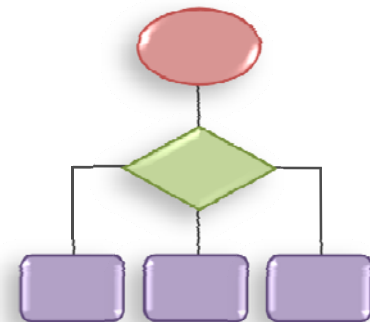


BizTalk Orchestration Designer

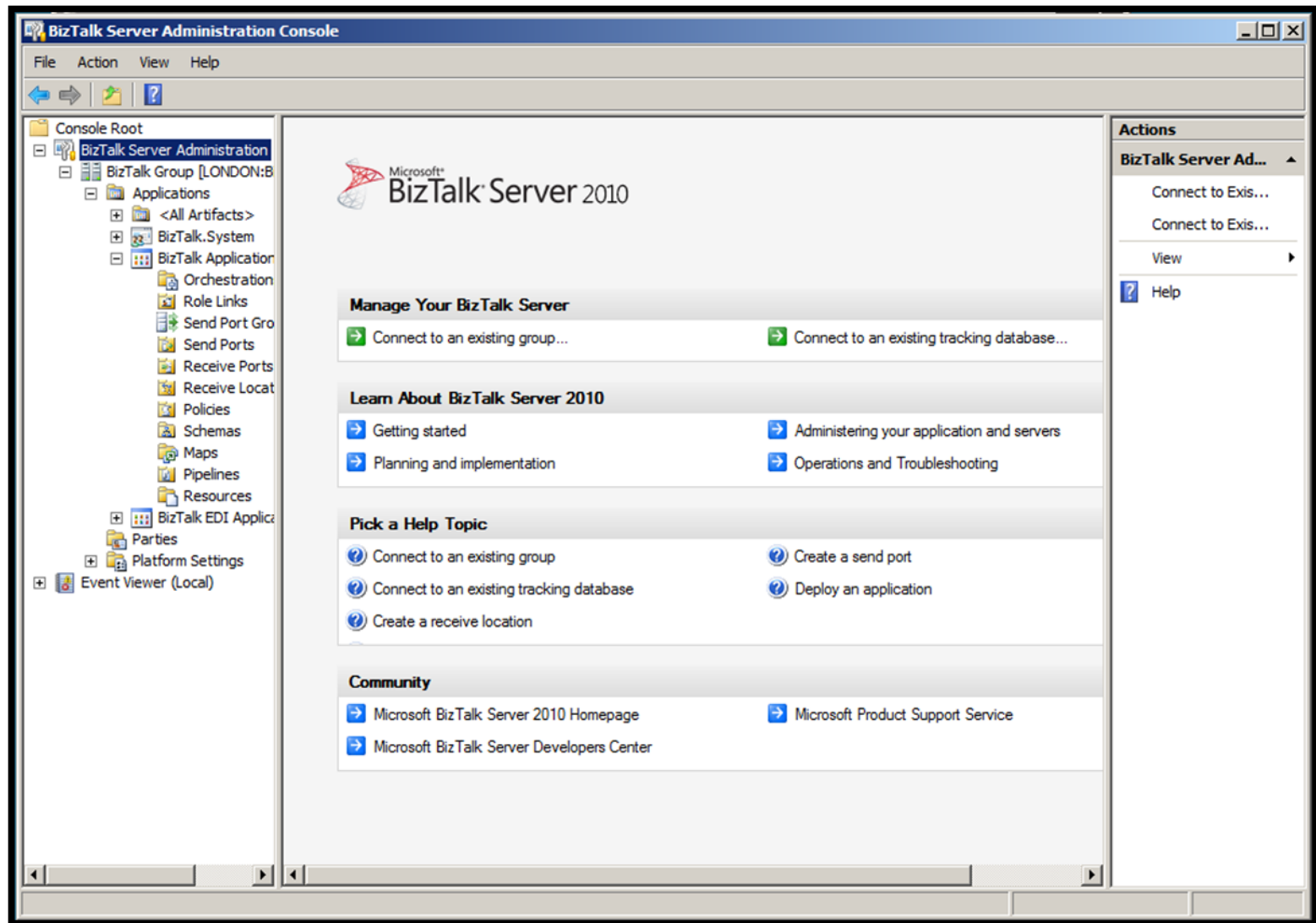
Use the Orchestration Designer to...

- Visually design a business process workflow model
- Specify implementation shapes to modeling the processes
- Save a business process model for later compilation

Design Business Processes



BizTalk Administration Console



This demonstration shows you how to use the...

- BizTalk Schema Editor
- BizTalk Mapper
- BizTalk Pipeline Designer
- BizTalk Orchestration Designer

Lab 1: Examining a BizTalk Application

In this 45 minute lab, you will:

- Examine existing BizTalk assembly artifacts
- Examine an existing BizTalk application
- Test a deployed BizTalk application

Review Questions

- What does BizTalk do?
- What are the new features in BizTalk Server 2010?

Creating Schemas

- Lesson 1: Introduction to BizTalk Schemas
- Lesson 2: Creating XML and Flat File Schemas

Lesson 1: Introduction to BizTalk Schemas

In this lesson, you will learn about...

- Reviewing XML Terminology
- What Are XML Namespaces?
- How Does BizTalk Use XML Namespaces?
- What Is a BizTalk XML Schema?
- Supported BizTalk Schema Types
- Flat File Structures
- What Is Electronic Data Interchange (EDI)?
- Examining an EDI Message

XML Standards

- Namespaces
- Elements
- Attributes
- XML Schema (XSD)
- XML Path Language (XPath)
- Extensible Stylesheet Language Transformations (XSLT)
- Web Services Description Language (WSDL)
- SOAP

What Are XML Namespaces?

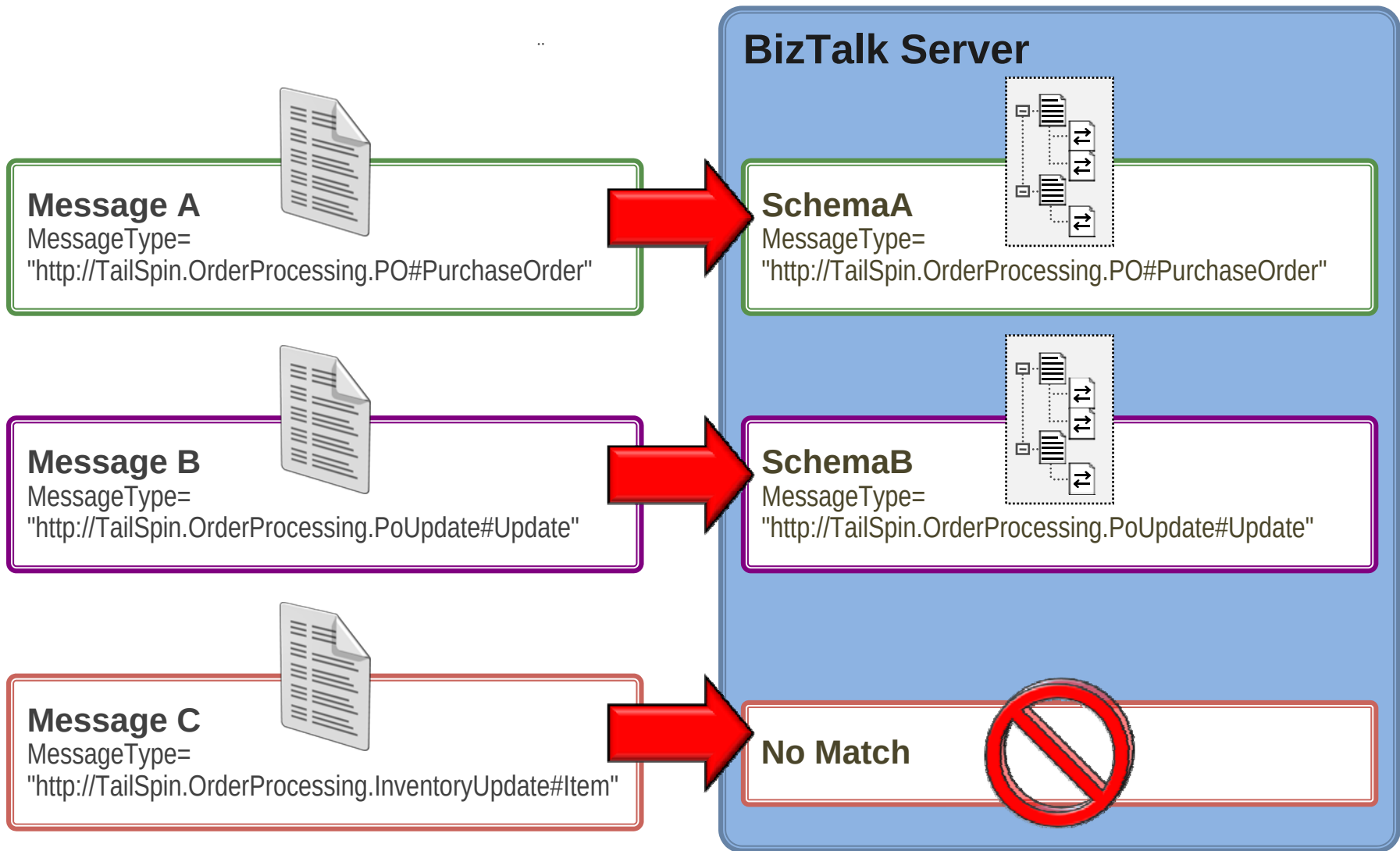
Namespaces

- Provide unique names for elements and attributes
- Prevent naming conflicts with other schemas
- Use the xmlns attribute to declare the namespace to which the element belongs
- A prefix is added to the declaration to remove ambiguity

```
<salesReport xmlns="http://adventure-works.com/salesReport"
xmlns:prod="http://adventure-works.com/products">
  <customer id="Fabrikam">
    <sales>
      <prod:id>widget1004</prod:id>

      <prod:unitsSold>100</prod:unitsSold>
      <prod:price>35</prod:price>
    </sales>
  </customer>
</salesReport>
```

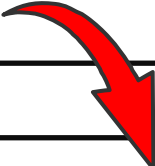
How Does BizTalk Use XML Namespaces?



What Is a BizTalk XML Schema?

XML Schema

```
<?xml version="1.0" encoding="utf-16" ?>
-- <xs:schema xmlns:b=http://schemas.microsoft.com/BizTalk/2003
attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
- <xs:element name="Item">
- <xs:complexType>- <xs:sequence>
  <xs:element name="Description" type="xs:string" />
  <xs:element name="Quantity" type="xs:integer" />
  <xs:element name="UnitPrice" type="xs:decimal" />
  <xs:element name="TotalPrice" type="xs:decimal" />
  <xs:element name="ItemID" type="xs:string" />
</xs:sequence> </xs:complexType>
</xs:element>
</xs:schema>
```

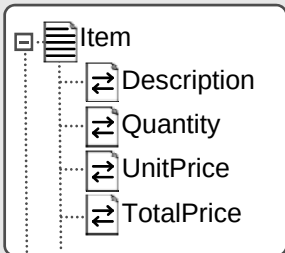


**BizTalk uses the XML
schema definition
language (XSD)**

XML Instance

```
- - -
<Item>
  <Description>Widget</Description>
  <Quantity>10</Quantity>
  <UnitPrice>1.04</UnitPrice>
  <TotalPrice>10.40</TotalPrice>
  <ItemID>A1234</ItemID>
</Item>
```

Supported BizTalk Schema Types

Message Type	Details																												
XML 	• Use when working with XML data • XSD defines the structure of the XML instance to enable parsing and validation																												
Flat File <table><tr><th>Node</th><th>Structure</th><th>Start</th><th>Length</th></tr><tr><td>Description</td><td>Delimited</td><td>0</td><td>50</td></tr><tr><td>UnitPrice</td><td>Delimited</td><td>50</td><td>10</td></tr><tr><td>Quantity</td><td>Delimited</td><td>60</td><td>10</td></tr><tr><td>TotalPrice</td><td>Delimited</td><td>70</td><td>10</td></tr><tr><td>ItemID</td><td>Delimited</td><td>80</td><td>10</td></tr><tr><td>Status</td><td>Delimited</td><td>99</td><td>1</td></tr></table>	Node	Structure	Start	Length	Description	Delimited	0	50	UnitPrice	Delimited	50	10	Quantity	Delimited	60	10	TotalPrice	Delimited	70	10	ItemID	Delimited	80	10	Status	Delimited	99	1	• XSD includes BizTalk specific markup that explains delimited and fixed length element structures • Markup is used by send and receive pipelines for translation
Node	Structure	Start	Length																										
Description	Delimited	0	50																										
UnitPrice	Delimited	50	10																										
Quantity	Delimited	60	10																										
TotalPrice	Delimited	70	10																										
ItemID	Delimited	80	10																										
Status	Delimited	99	1																										
EDI <pre>ISA:00: :00: :01:1515151515 :01:515151 : 041201:1217:U:00403:000032123:0:P:*~GS:CT :99887766 55:1122334455:20041201:1217:128:X:004030~ ST:831:00128001~BGN :00:88200001:20041201~N9:BT:88200001~TR N:1:88200001~AMT:2:10</pre>	• Similar to flat file schemas but for EDIFACT and X12 standard messages • BizTalk includes thousands of provided schemas																												

Delimited elements

- Fields separated by a specified delimiter
- Have a common end-of-record terminator

```
John,Smith,123 Main St,Learning BizTalk Server 2009
```

Positional based elements

- Fields are a fixed length
- Have a common end-of-record terminator

```
John   Smith   123 Main St   Learning BizTalk Server 2009
```


Electronic Data Interchange (EDI)

What is EDI?

- Provides a standard way for organizations to exchange documents such as invoices, bills, and purchase orders
- Designed to be independent of communication and software technologies

```
UNB+IATB:1+6XPPC+LHPPC+940101:0950+1'  
UNH+1+PAORES:93:1:IA'  
MSG+1:45'  
IFT+3+XYZCOMPANY AVAILABILITY'  
ERC+A7V:1:AMD'  
IFT+3+NO MORE FLIGHTS'  
ODI'  
TVL+240493:1000::1220+FRA+JFK+DL+400+C'  
PDI++C:3+Y::3+F::1'
```

Example of an EDI Message

X12 850 Message Sample

```
ST*850*15000077~  
BEG*00*NE*4567890123**20070315~  
ITD*01****10~  
DTM*002*20070315~  
N1*ST*Charlotte*09*0567891230012~  
N3*123 Charlotte Rd~  
N4*CHARLOTTE*NC*12345~  
PO1*001*10*EA*123.45*CT*VN*987654321~  
PID*F****RAW CONCENTRATE~  
CTT*1*123.45~  
AMT*1*123.45~  
SE*1*15000077~
```

EDI will be covered in detail in Module 15

Lesson 2: Creating XML and Flat File Schemas

In this lesson, you will learn about....

- Methods for Creating BizTalk XML Schemas
- Creating a Schema by Using the BizTalk Editor
- Using Multiple Schemas
- Testing a Schema
- Generating Schemas
- Demonstration: Creating and Testing a Schema
- Using the Flat File Schema Wizard
- Demonstration: Creating and Testing a Flat File Schema

Schema creation techniques

- Create from scratch using BizTalk Editor
- Import or include existing types
- Generate from an instance message
- Generate using adapter metadata wizard
- Migrate an older XDR schema to an XSD schema

Creating a Schema by Using the BizTalk Editor

Schema
Tree View

The screenshot displays the BizTalk Editor interface with the 'SalesOrder.xsd' file open. The Schema Tree View on the left shows a hierarchical structure of the schema, including elements like 'SalesOrder', 'Detail', 'StoreNumber', 'Employee', 'OrderNumber', 'OrderType', 'OrderDate', 'CustomerInfo', 'ID', 'CustomerName', 'MonthsAtResidence', 'Residence', and 'Street'. The XSD View on the right shows the XML Schema Definition (XSD) code. A context menu is open over the Schema Tree View, showing options for inserting and modifying schema nodes. The XSD View shows the XML Schema Definition (XSD) code, including namespace declarations and element definitions.

Schema Tree View

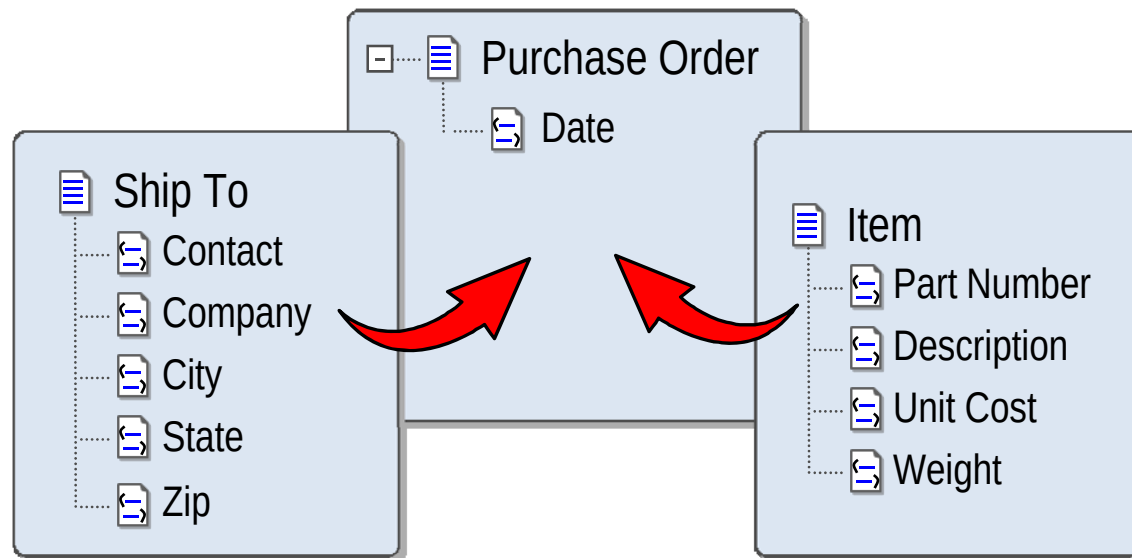
- Insert Schema Node
 - Promote
 - Cut
 - Copy
 - Paste
 - Delete
 - Rename
 - Refresh XSD
 - Expand Schema Node
 - Collapse Schema Node
 - Properties
- Child Record (Ctrl+I, Ctrl+R)
- Child Field Attribute (Ctrl+I, Ctrl+F)
- Child Field Element (Ctrl+I, Ctrl+E)
- Sibling Record (Ctrl+I, Ctrl+S)
- Sibling Field Attribute (Ctrl+I, Ctrl+I)
- Sibling Field Element (Ctrl+I, Ctrl+L)
- Sequence Group (Ctrl+I, Ctrl+G)
- Choice Group (Ctrl+I, Ctrl+H)
- All Group (Ctrl+I, Ctrl+A)
- Attribute Group (Ctrl+I, Ctrl+T)
- Any Element (Ctrl+I, Ctrl+N)
- Any Attribute (Ctrl+I, Ctrl+B)

XSD View

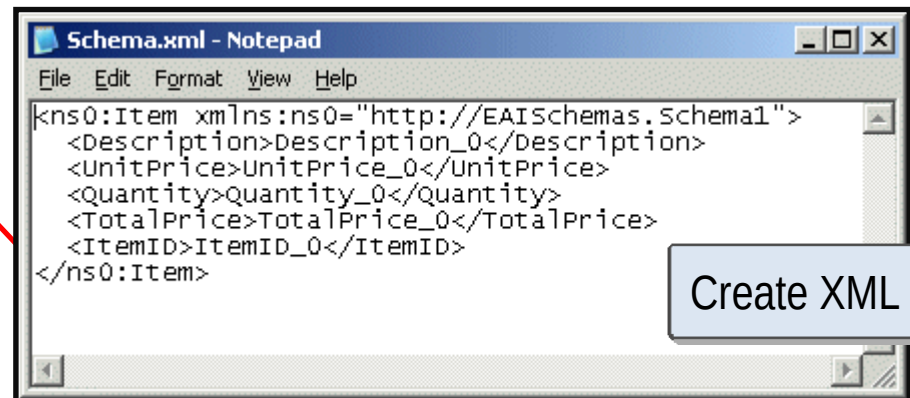
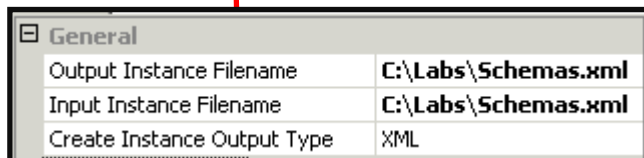
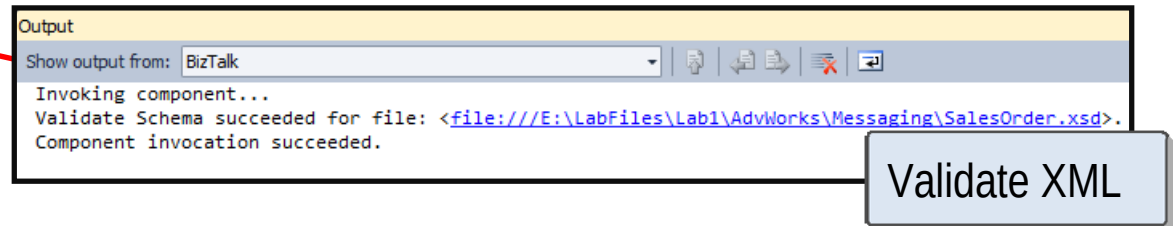
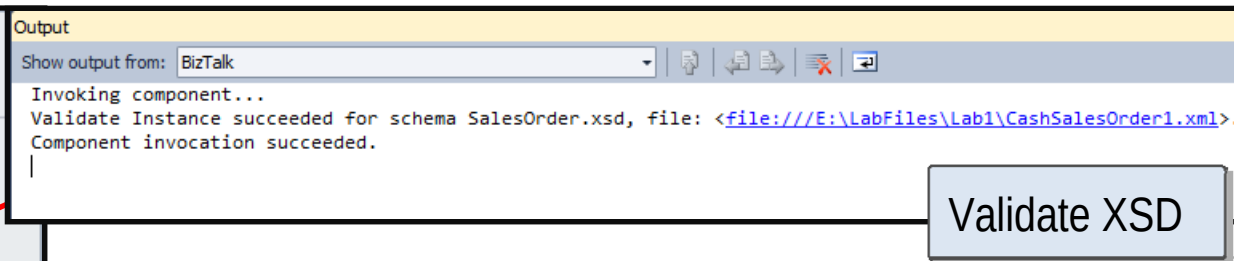
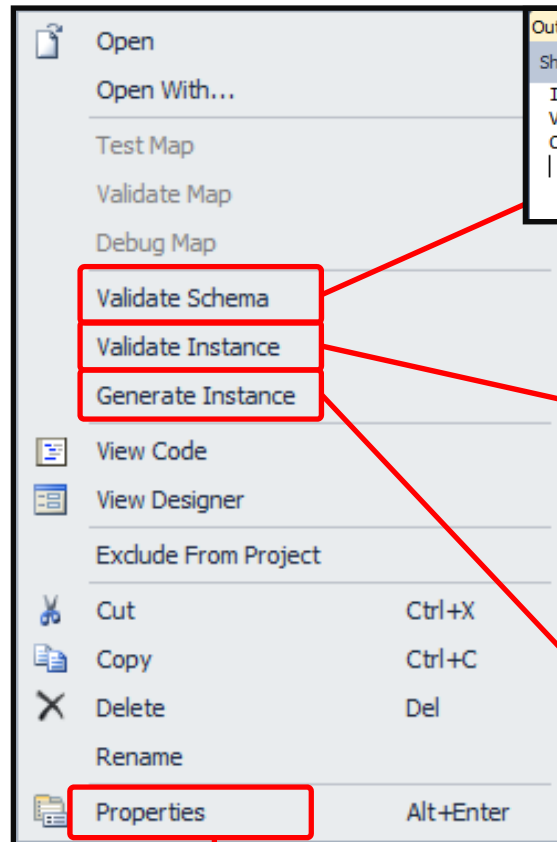
```
<?xml version="1.0" encoding="utf-16" ?>
<xs:schema xmlns="http://AdvWorks.Messaging.SalesOrderv 1.0"
  xmlns:b="http://schemas.microsoft.com/BizTalk/2003"
  xmlns:ns0="http://AdvWorks.Messaging.PropertySchema.PropertySchema"
  targetNamespace="http://AdvWorks.Messaging.SalesOrderv 1.0"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:annotation>
    <xs:appinfo>
      <b:imports xmlns:b="http://schemas.microsoft.com/BizTalk/2003">
        <b:namespace prefix="ns0"
          uri="http://AdvWorks.Messaging.PropertySchema.PropertySchema"
          location=".\\propertySchema.xsd" />
        </b:imports>
      </xs:appinfo>
    </xs:annotation>
    <xs:element name="SalesOrder">
      <xs:annotation>
        <xs:appinfo>
          <b:properties>
            <b:property name="ns0:OrderType" xpath="/*[local-name()
              ='SalesOrder' and namespace-uri()
              =http://AdvWorks.Messaging.SalesOrderv 1.0]/*[local-
              name()='Detail' and namespace-uri()='*/@*[local-name()
              ='OrderType' and namespace-uri()='*/] />
            <b:property name="ns0:OrderTotal" xpath="/*[local-name()
              ='SalesOrder' and namespace-uri()
              =http://AdvWorks.Messaging.SalesOrderv 1.0]/*[local-
              name()='OrderTotal' and namespace-uri()='*/] />
            <b:property distinguished="true" xpath="/*[local-name()
              ='SalesOrder' and namespace-uri()
              =http://AdvWorks.Messaging.SalesOrderv 1.0]/*[local-
              name()='OrderTotal' and namespace-uri()='*/] />
            <b:property distinguished="true" xpath="/*[local-name()
              ='SalesOrder' and namespace-uri()
              =http://AdvWorks.Messaging.SalesOrderv 1.0]/*[local-
              name()='Detail' and namespace-uri()='*/@*[local-name()
              ='OrderType' and namespace-uri()='*/] />
            <b:property distinguished="true" xpath="/*[local-name()
              ='SalesOrder' and namespace-uri()
              =http://AdvWorks.Messaging.SalesOrderv 1.0]/*[local-
              name()='OrderTotal' and namespace-uri()='*/] />
          </b:properties>
        </xs:appinfo>
      </xs:annotation>
    </xs:element>
  </xs:schema>
```

Using Multiple Schemas

Method	Usage
Import	• Accesses and uses types defined in the imported schema • Provides for using types from schemas in other namespaces
Include	• Accesses and uses types defined in the included schema • Provides for using types from schemas in the same namespace
Redefine	• Accesses and uses types defined in the redefined schema • Provides for using types derived from a schema in the same namespace



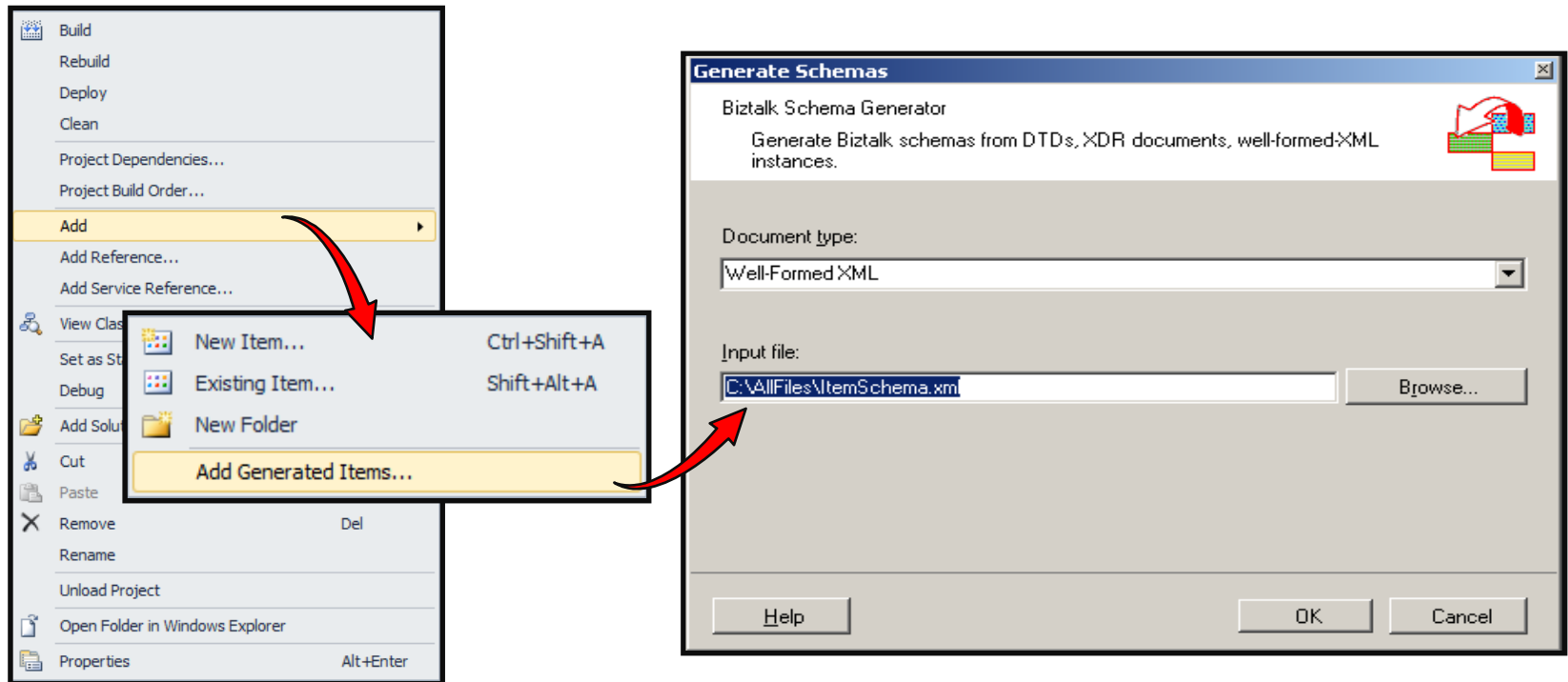
Testing a Schema



Generating Schemas

Automatically create an XSD based on:

- Well-formed XML instance
- XDR schema
- DTD schema



Demo: Creating & Testing a Schema

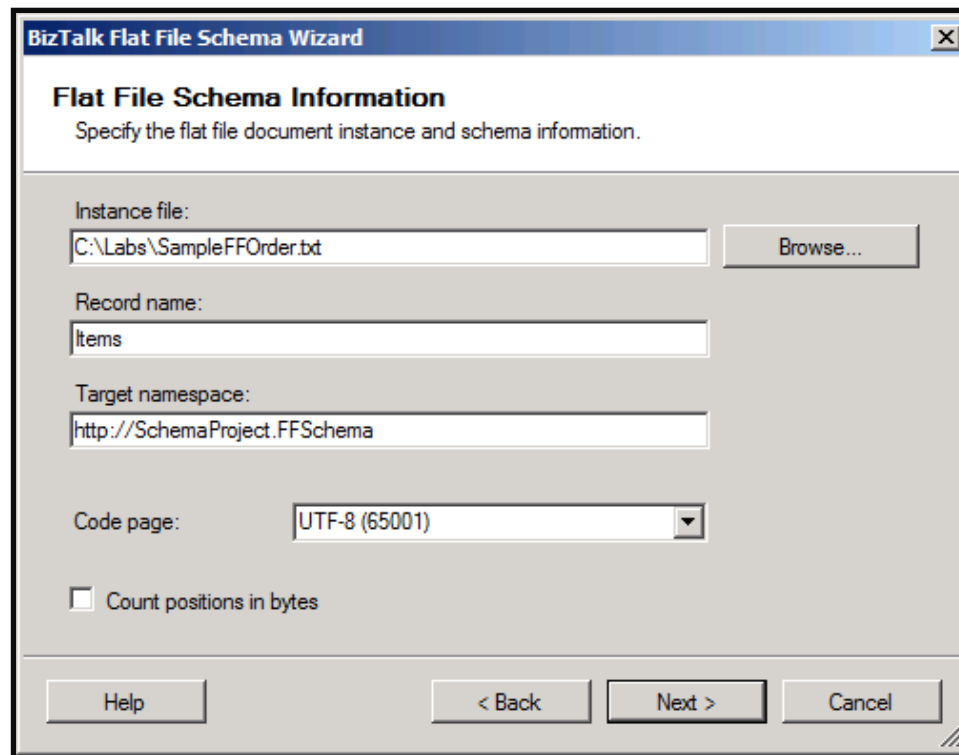
In this demonstration, you will learn about...

- Using the BizTalk Editor to create a schema
- Validating and testing a schema

Using the Flat File Schema Wizard

Flat File Schema Wizard

- Graphical tool for defining flat file schemas
- Can define schemas for delimited or positional flat files



The screenshot shows the 'BizTalk Flat File Schema Wizard' dialog box. The title bar reads 'BizTalk Flat File Schema Wizard'. The main heading is 'Flat File Schema Information' with the instruction 'Specify the flat file document instance and schema information.' Below this, there are several input fields: 'Instance file:' with the text 'C:\Labs\SampleFFOrder.txt' and a 'Browse...' button; 'Record name:' with the text 'Items'; 'Target namespace:' with the text 'http://SchemaProject.FFSchema'; and 'Code page:' with a dropdown menu showing 'UTF-8 (65001)'. There is also an unchecked checkbox labeled 'Count positions in bytes'. At the bottom, there are four buttons: 'Help', '< Back', 'Next >', and 'Cancel'.

Demo: Creating & Testing a Flat File Schema

In this demo, you will learn about....

- Creating a schema by using the Flat File Schema Wizard
- Validating and testing a flat file schema

Lab 2: Creating & Configuring Schemas

In this 45 minute lab, you will...

- Create a New BizTalk Project
- Create an XML Schema
- Create a Flat File Schema
- Generate a Schema from an XML Message Instance

Review Questions

- How does BizTalk Server use schemas?
- What are the ways that you can create XML schemas for use by BizTalk?
- What is the MessageType, and why is it important?

Creating BizTalk Maps

- Lesson 1: Creating a BizTalk Map
- Lesson 2: Configuring Basic Functoids
- Lesson 3: Advanced BizTalk Mapper Features

Lesson 1: Creating a BizTalk Map

In this lesson, you will learn about...

- What Is a BizTalk Map?
- Creating a Map Using the BizTalk Mapper
- Creating Links
- Validating and Testing a Map
- Debugging a BizTalk Map

What Is a BizTalk Map?

Map

- Defines correspondence between records and fields in two different schemas

Data Transformation

- Perform computational and other data operations
- Copy data from one message to another

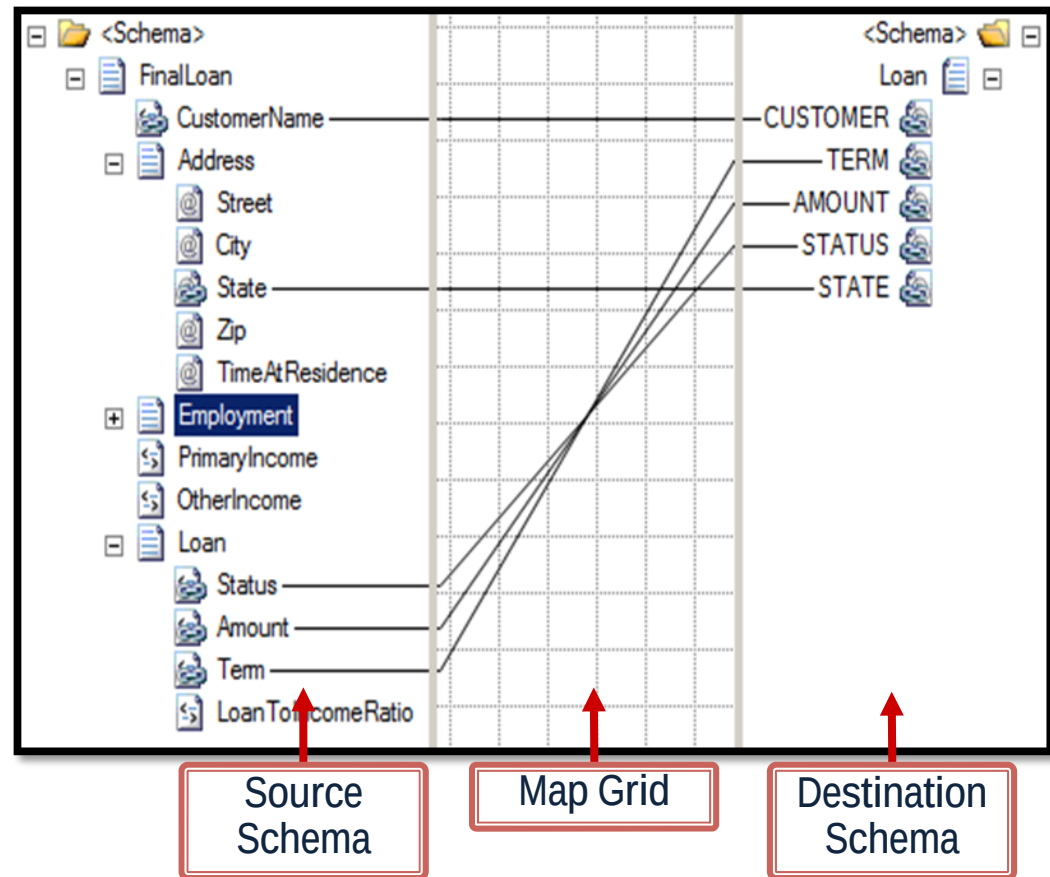
Data Translation

- Change the format of data between messages
- Example: translate between a flat file and an XML file

Creating a Map Using the BizTalk Mapper

BizTalk Mapper

- Integrated within Visual Studio
- Starts when a map is opened or added to a project
- Source and destination schemas must be part of the project or contained in a referenced assembly



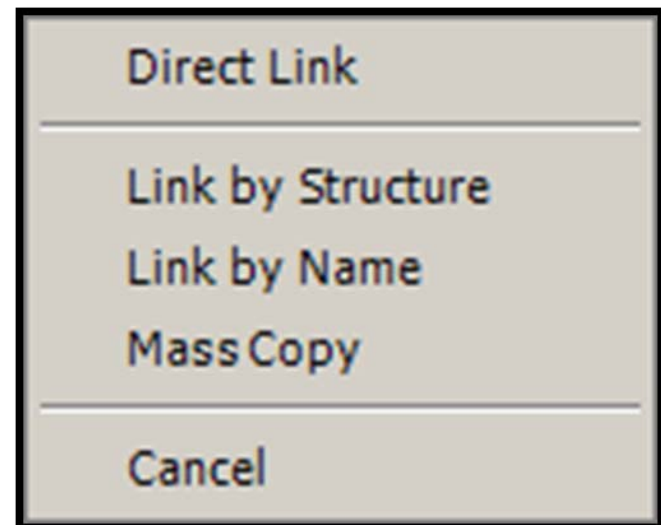
Creating Links

Simple Linking

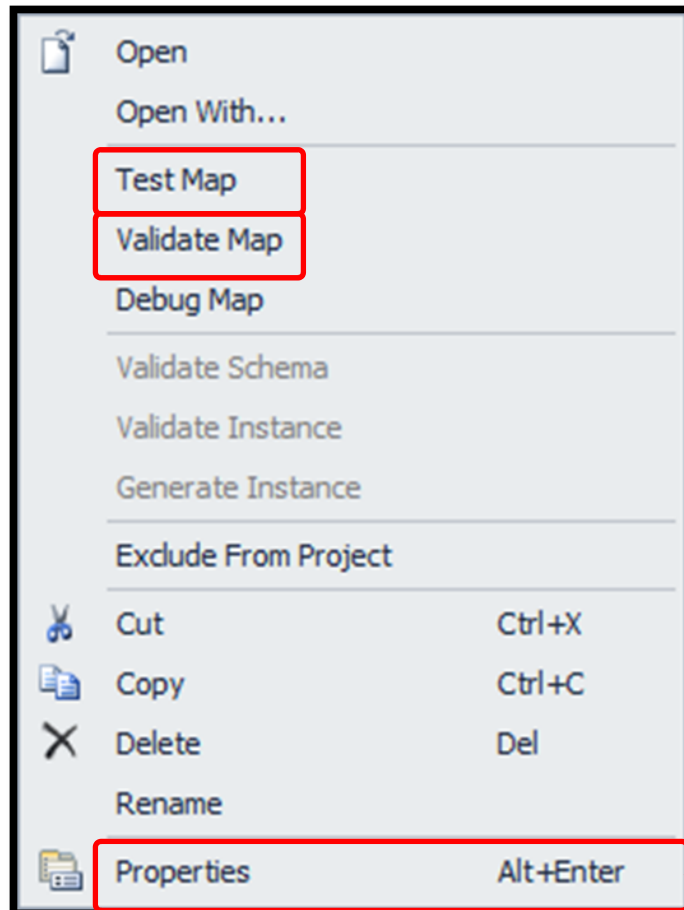
- Enabled by dragging and dropping between two nodes
- Create a single link between the nodes

Automatic Linking

- Enabled by linking two records
- Automatically creates links based on your choice



Validating and Testing a Map



Test Map	
TestMap Input	Generate Instance
TestMap Input Instance	
TestMap Output	XML
TestMap Output Instance	
Validate TestMap Input	True
Validate TestMap Output	True

Debugging a BizTalk Map

Debug the generated XSLT

- Set breakpoints
- Debug scripts, functoids, and custom code



Lesson 2: Configuring Basic Functoids

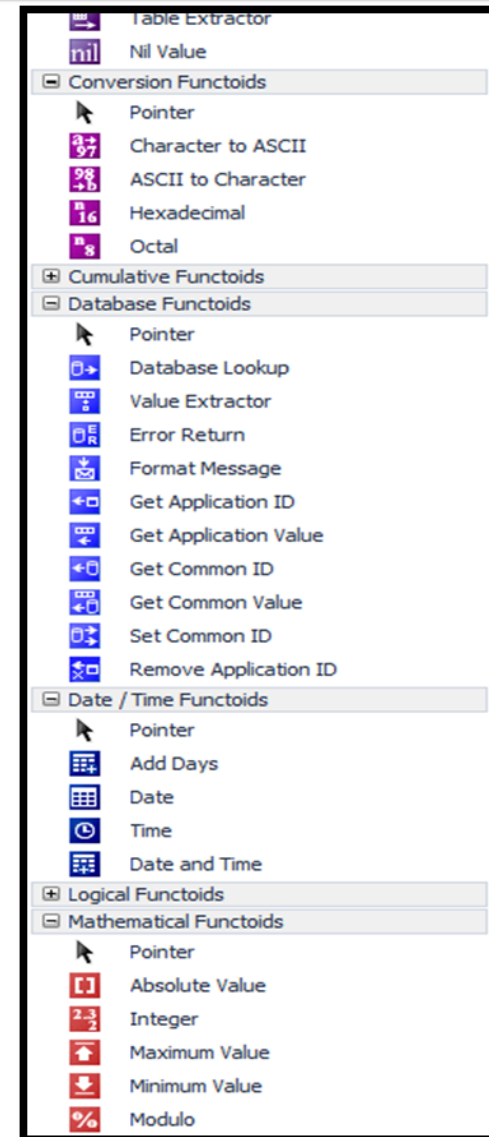
In this lesson, you will learn about...

- Data Manipulation with Functoids
- Basic Functoids Categories
- Adding Functoids to a Map
- Using Map Grid Pages
- Demonstration: Creating a BizTalk Map

Data Manipulation with Functoids

Functoids

- Use to manipulate mapped data
- Dozens of provided functoids
- Develop custom functoids for other behaviors



Basic Functoids Categories

Category	Usage
Conversion	To convert to and from ASCII and between numeric bases, such as hexadecimal and octal
Cumulative	To perform mathematical operations in looping records, such as averages and concatenation
Date and Time	To retrieve the current date and time, and to calculate delta times
Logical	To perform a variety of logical operations, such as greater than and logical existence
Mathematical	To perform a variety of mathematical operations, such as addition and multiplication
Scientific	To perform a variety of scientific operations, such as logarithms and trigonometry
String	To perform a variety of string functions, such as trimming and concatenation

Adding Functoids to a Map

The screenshot displays the Microsoft Dynamics CRM interface with a map titled "SalesOrder_to_Restock.btm". The "Toolbox" on the left lists various functoid categories, with "String Functoids" expanded. The "String Concatenate" functoid is highlighted. The main map area shows a grid with data fields like "StoreNumber", "Employee", "OrderNumber", "OrderType", "OrderDate", "CustomerInfo", "Items", "Qty", "SKU", "Price", "ExtendedPrice", "Comment", "OrderTotal", and "TermOfLoan". A "Schema" pane on the right shows the data structure. A "Configure String Concatenate Functoid" dialog box is open, showing the "Functoid Inputs" tab. The dialog includes a table for inputs and a description of the functoid.

Toolbox

- Advanced Functoids
- Conversion Functoids
- Cumulative Functoids
- Database Functoids
- Date / Time Functoids
- Logical Functoids
- Mathematical Functoids
- Scientific Functoids
- String Functoids**
 - Pointer
 - String Find
 - String Left
 - String Right
 - Uppercase
 - Lowercase
 - Size
 - String Extract
 - String Concatenate**
 - String Left Trim
 - String Right Trim
- General

SalesOrder_to_Restock.btm

Schema: Restock (OrderID, Products, Product)

Map Fields:

- esOrder
- Detail
 - StoreNumber
 - Employee
 - OrderNumber
 - OrderType
 - OrderDate
- CustomerInfo
- Items
 - Item
 - Qty
 - SKU
 - Price
 - ExtendedPrice
- Comment
- OrderTotal
- TermOfLoan

Configure String Concatenate Functoid

Functoid Inputs | Label and Comments

Add/delete/reorder functoid inputs

Name	Value	Type
Input[0]	StoreNumber	
Input[1]	OrderNumber	

This functoid must have between 1 and 100 input parameter(s).

Functoid description

Use the String Concatenate functoid to concatenate a series of input strings. This functoid can accept one to one hundred input parameters.

OK Cancel

Using Map Grid Pages

Grid Pages

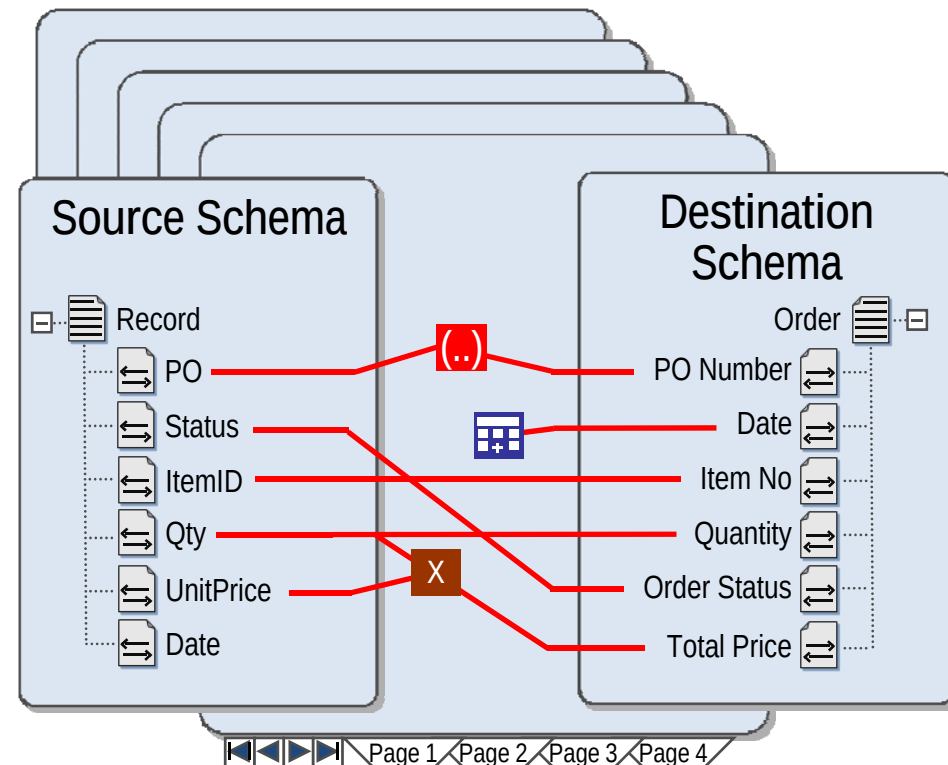
- Create up to 20 pages
- Use to isolate different parts of a map
- Work with parts of a map separately
- Must create connected functoids on the same layer

Grid Preview

- Find and work with a portion of a large map



Use pages to reduce complexity of a map



Demo: Creating a BizTalk Map

In this demonstration, you will see how to...

- Create a map
- Create map links
- Manipulate data using functoids
- Validate and test a map

Lesson 3: Advanced BizTalk Mapper Features

In this lesson, you will learn...

- Using Auto-scroll
- Using Functoid IntelliSense
- Using Relevance View
- Using Indicative Match
- Documenting Maps

Auto-scroll:

- When user clicks an object the view automatically scrolls to bring that object and other related objects into view

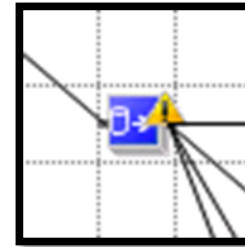
Considerations:

- Auto-scroll is enabled by default but can be toggled by clicking the  button

Using Functoid IntelliSense

Functoid Intellisense:

- Visualization mechanism
- Helps quickly identify problems with a functoid
- Tooltip provides description of error



For example,

- If functoid does not have minimum number of valid inputs
- Tooltip mentions the number of input parameters required

Using Relevance View

Relevance View

- Uses sibling coalescence to collapse non-relevant elements
- Provides a more compact view of the schema
- Option on source and destination schema
- Enabled by clicking the  icon in the Mapper ribbon

When enabled:

- Records without linked children are collapsed
- Successive nodes without links are coalesced and are replaced by the  icon

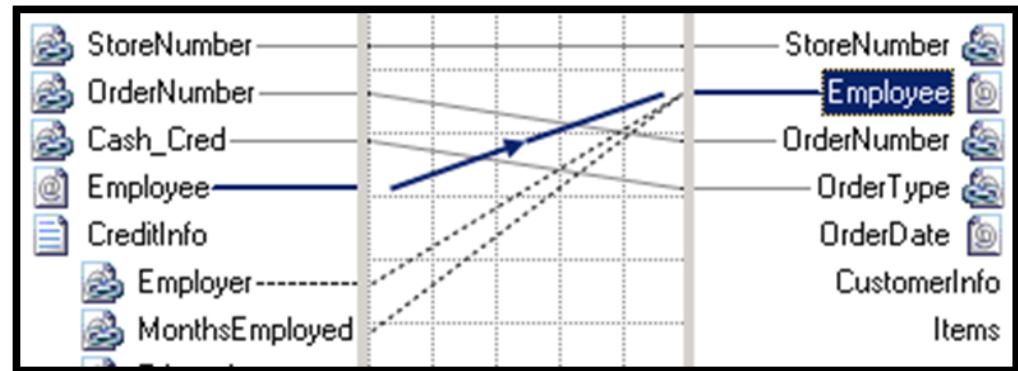
Using Indicative Match

Indicative Match:

- Compares selected node against nodes in opposite schema
- Highlights best suggestions for links

To Use:

1. Select schema node
2. Press SHIFT + SPACE
3. Use arrow keys to cycle through suggestions
4. Press ENTER to create link



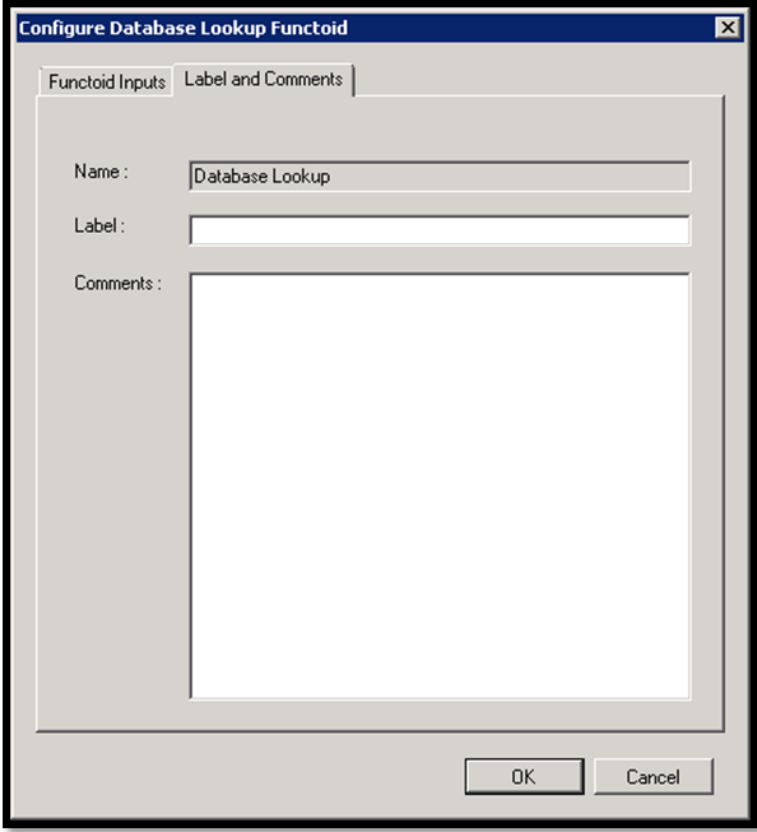
Documenting Maps

Labels

- **Label** property of links change the display name in functoids
- **Label** property of functoids change tooltip

Comments

- Use comments to document functoid usage
- Comments are displayed in tooltip



The screenshot shows a dialog box titled "Configure Database Lookup Functoid". It has two tabs: "Functoid Inputs" and "Label and Comments". The "Label and Comments" tab is selected. Inside the dialog, there are three input fields: "Name:" with the text "Database Lookup", "Label:" which is empty, and "Comments:" which is a large empty text area. At the bottom right, there are "OK" and "Cancel" buttons.

Demo: Advanced Features of the Mapper

In this demonstration, you will see how to...

- Use Auto-scroll
- Use Functoid IntelliSense
- Use Relevance View
- Use Indicative Match
- Document Maps

Lab: Creating a BizTalk Map

In this 45 minute lab, you will...

- Create a map
- Add basic functoids to a map
- Add database functoids to a map

Deploying a BizTalk Project

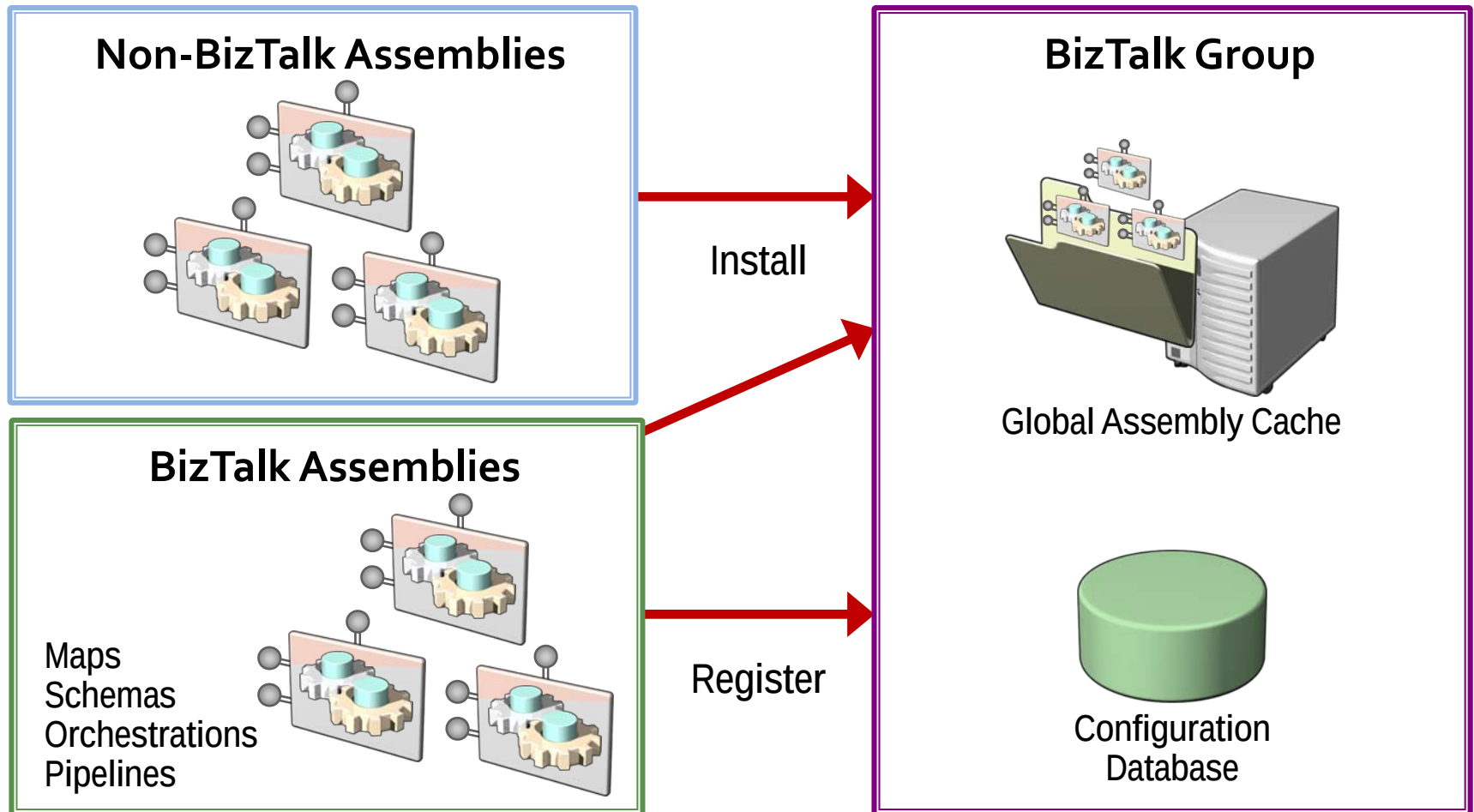
- Lesson 1: Introduction to Deployment
- Lesson 2: Deploying an Assembly

Lesson 1: Introduction to Deployment

In this lesson, you will learn about...

- How Deployment Works
- What Is a BizTalk Application?
- Assembly Deployment Options
- How BizTalk Uses Strong Names
- Assembly Deployment Steps

How Deployment Works



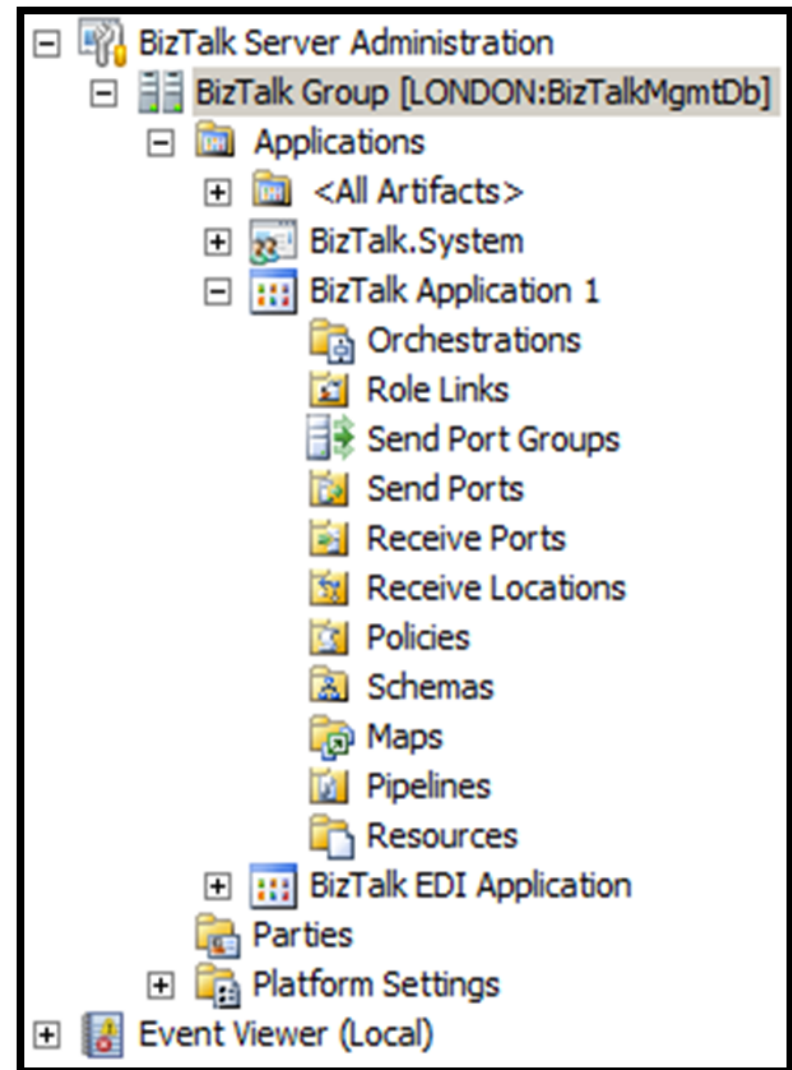
What Is a BizTalk Application?

BizTalk Artifacts

- Schemas
- Maps
- Pipelines
- Orchestrations

Messaging Components

- Receive ports
- Receive locations
- Send ports
- Adapters



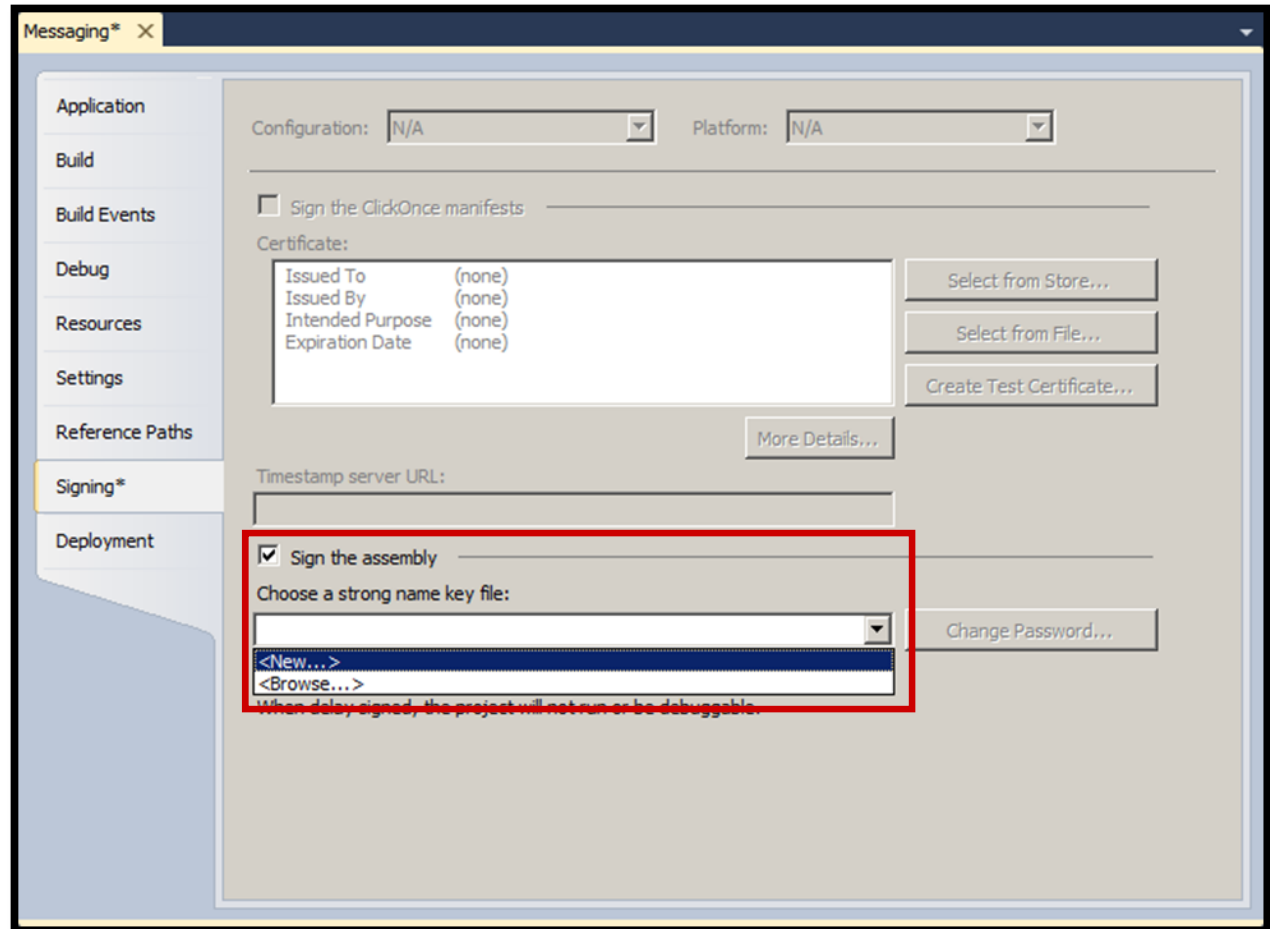
Methods of Deployment

Method	Usage
Visual Studio	• Developer tool (testing) • Use the redeploy option to override same version
BTSTask	• Administrator tool (production) • Deploy assemblies • Import and export bindings • Deploy multiple assemblies to multiple servers
BizTalk Administration Console	• Developer and IT Pro tool • Import and export bindings • Export applications as Windows Installer (MSI) packages

How BizTalk Uses Strong Names

Strong Name

- Part of digital signature
- Provides integrity check
- Guarantees uniqueness
- Prevents version conflict



Assembly Deployment Steps

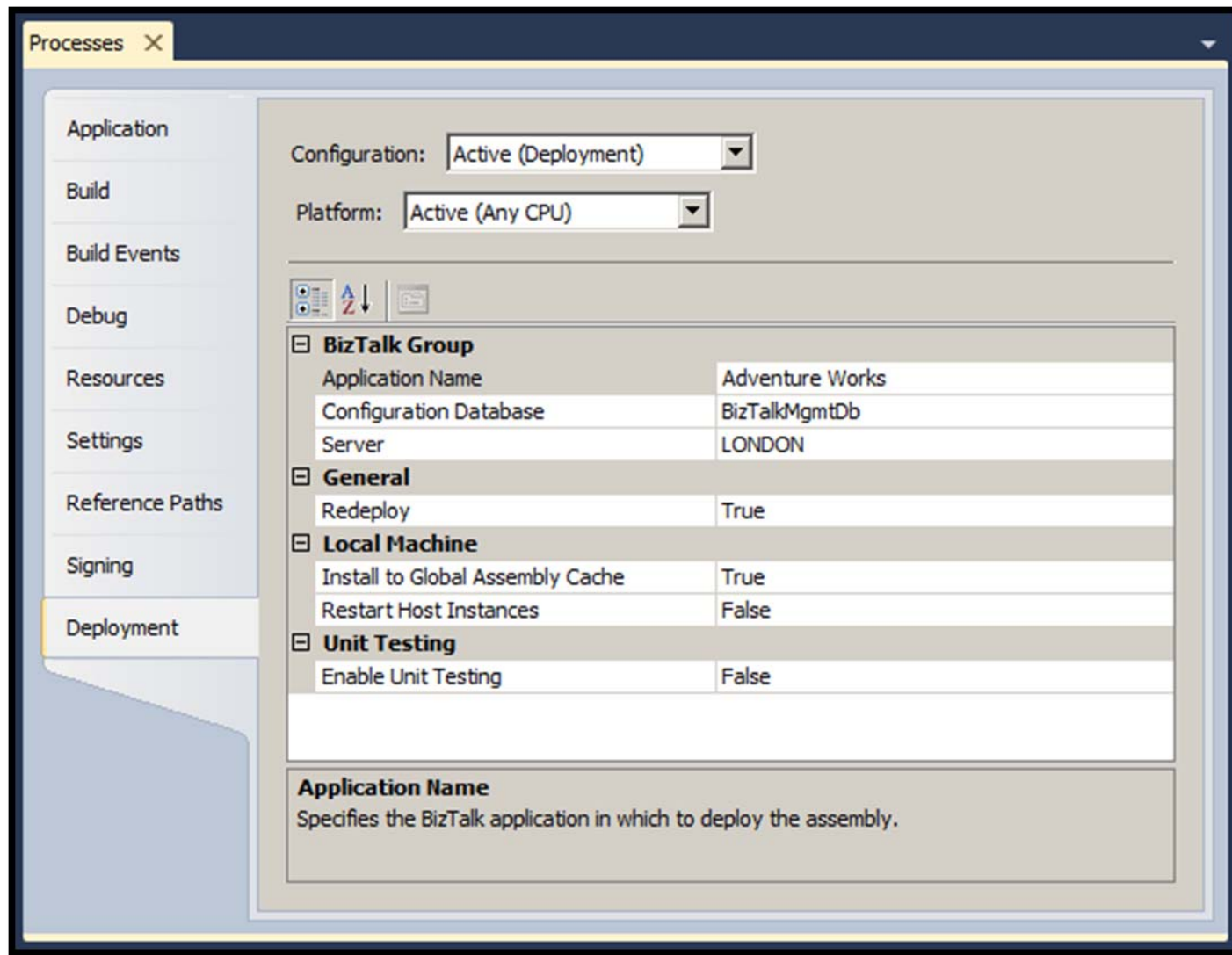
1. Verify assembly versions
2. Assign a strong name the project
3. Build the project to create the assembly
4. Install assembly in the GAC
5. Register assembly in the Configuration database
6. Create physical send ports and receive locations
7. Enable ports and enlist the orchestration

Lesson 2: Deploying an Assembly

In this lesson, you will learn about...

- Deploying Assemblies using Visual Studio
- Demonstration: Deploying the Messaging Solution
- Managing Assembly Versions
- Viewing Deployed BizTalk Assemblies
- Undeploying an Assembly
- Demonstration: Viewing Deployed Assemblies

Deploying Assemblies using Visual Studio



Demo: Deploying the Messaging Solution

In this demonstration, you will learn about...

- Assign a strong name to a project
- Assign an application name to a project
- Build a BizTalk Server project
- Deploy an assembly to the global assembly cache
- Register an assembly in the configuration database

Managing Assembly Versions

Before you deploy

- Compare the existing version with version of assembly to be deployed

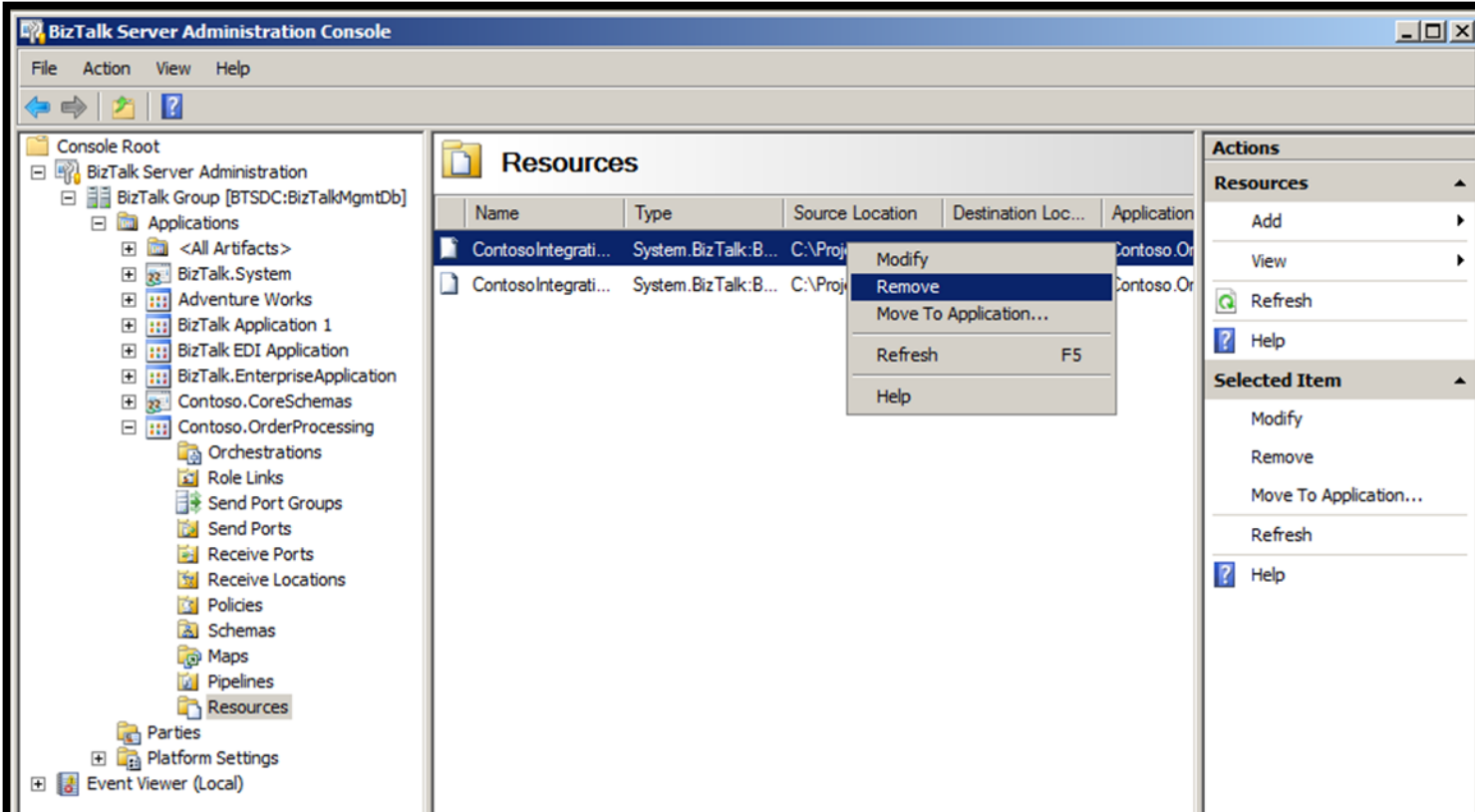
Be aware that...

- Duplicate versions will cause deploy operation to fail
- If versions are different, assembly will deploy
- Versioned schemas must have different message types

To remove an existing assembly

- Use the BizTalk Administration Console
- Run BTSTask from the command line

Undeploying an Assembly



The screenshot shows the BizTalk Server Administration Console. On the left, the tree view is expanded to 'Resources' under 'Applications'. The main pane shows a table of resources with columns: Name, Type, Source Location, Destination Location, and Application. Two resources named 'ContosoIntegrati...' are listed. A right-click context menu is open over the first resource, with 'Remove' selected. The 'Actions' pane on the right shows options for 'Resources' (Add, View, Refresh, Help) and 'Selected Item' (Modify, Remove, Move To Application..., Refresh, Help).

Name	Type	Source Location	Destination Loc...	Application
ContosoIntegrati...	System.BizTalk:B...	C:\Proj...		Contoso.Or
ContosoIntegrati...	System.BizTalk:B...	C:\Proj...		Contoso.Or

```
C:\> BTSTask RemoveResource /ApplicationName:LabA /Server:LONDON
```

Demo: Viewing Deployed Assemblies

In this demonstration, you will learn about...

- Assemblies registered in the BizTalk Configuration database by using the BizTalk Server Administration Console
- Undeploying a BizTalk Server assembly

Lab: Deploying a BizTalk Project

In this 45 minute lab, you will....

- Create and Assign a Strong Name Key to an Assembly
- Build and Deploy a BizTalk Assembly
- View Assemblies using the Administration Console
- Change Assembly Versions
- Deploy the New Version of the BizTalk Assembly
- Undeploy an Assembly using the Administration Console

Review Questions

- Why must BizTalk assemblies be signed with a strong name key?
- Are non-BizTalk “helper” assemblies required to be strong name-signed?
- What two places do BizTalk assemblies have to be “deployed” to?

Routing BizTalk Messages

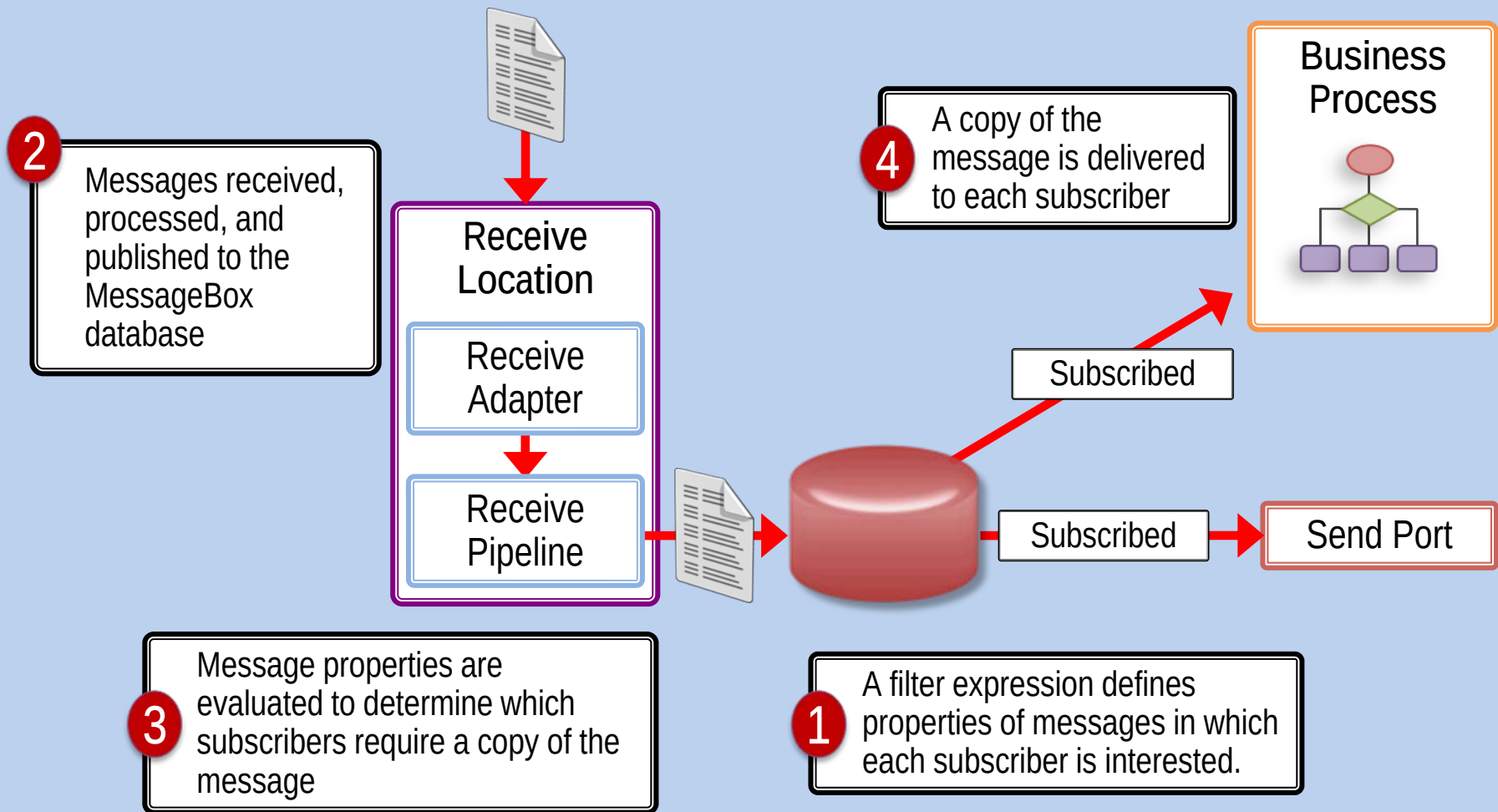
- Lesson 1: Introduction to Message Routing
- Lesson 2: Configuring Message Routing
- Lesson 3: Monitoring Orchestrations and Messages

Lesson 1: Introduction to Message Routing

In this lesson, you will learn about...

- The Publish and Subscribe Architecture
- What Is the MessageBox Database?
- What Is Message Routing?
- What Is a Port?
- BizTalk Message Flow
- What Is Property Promotion?
- Demonstration: Promoting a Property

The Publish and Subscribe Architecture



What Is the MessageBox Database?

MessageBox Database

- Stores subscription information
- Delivers messages to subscribers
- Stores the queues and state tables for each BizTalk host
- Stores messages and message properties

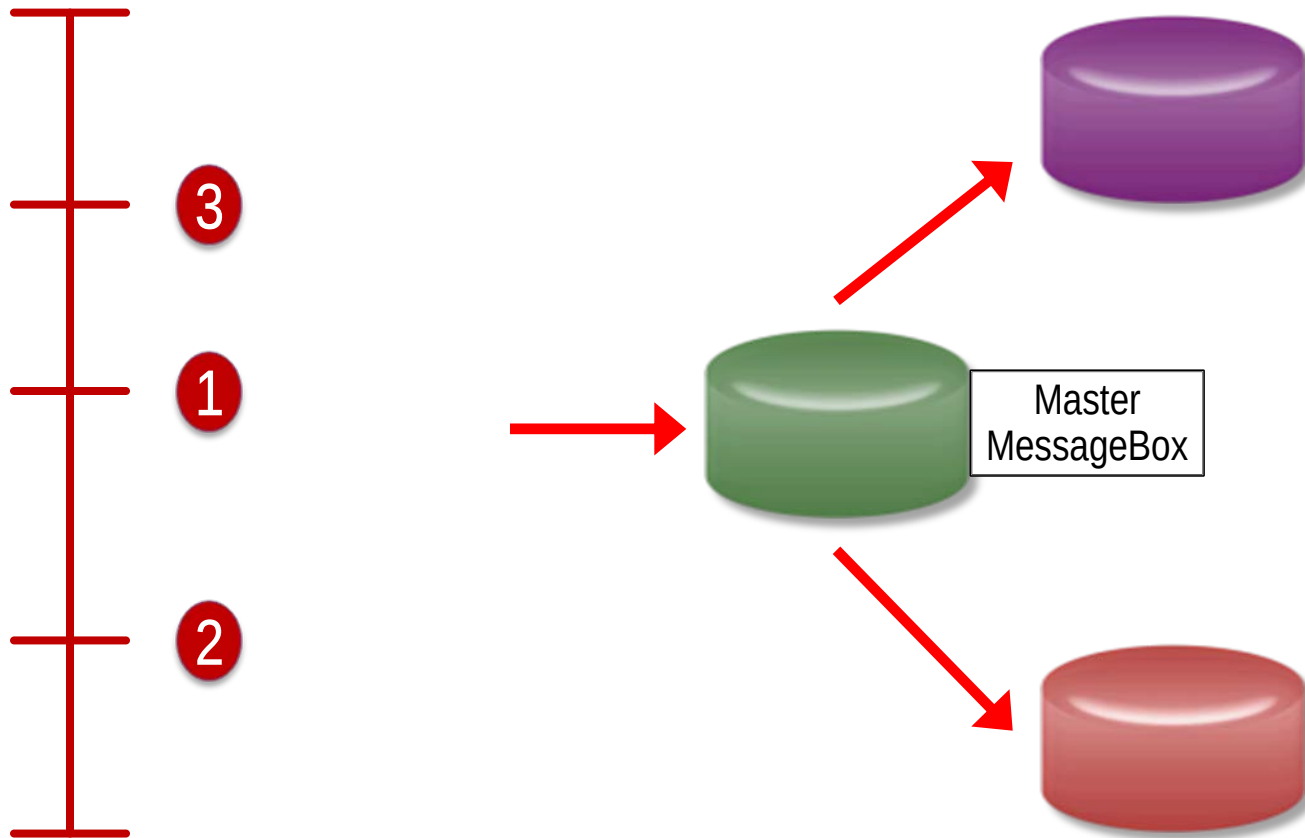


Administration Tasks

- Adding a new MessageBox database
- Disabling new message publication
- Deleting a MessageBox from a BizTalk Group



Multiple MessageBox Databases

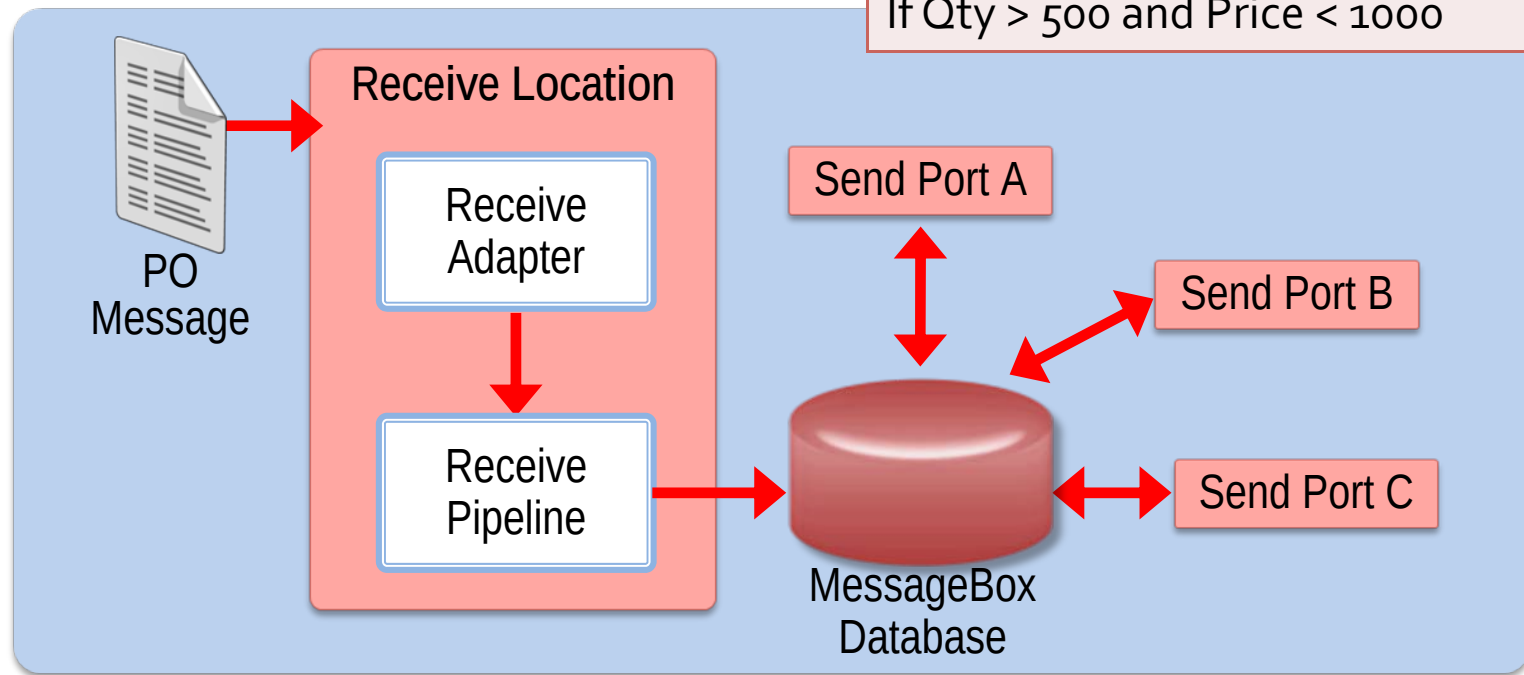


What Is Message Routing?

Message Routing

- Routes directly to a port
- Based on a filter expression
- Configured on the send port

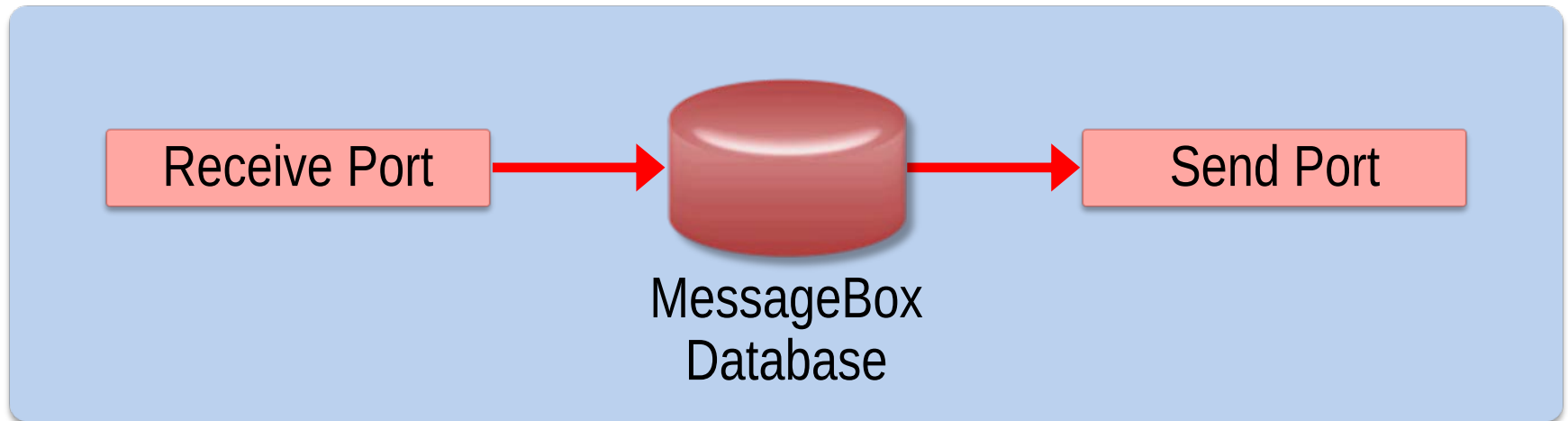
Filter Expression	Port
If CustomerName = Contoso	A
If Price > 1000	B
If Qty > 500 and Price < 1000	C



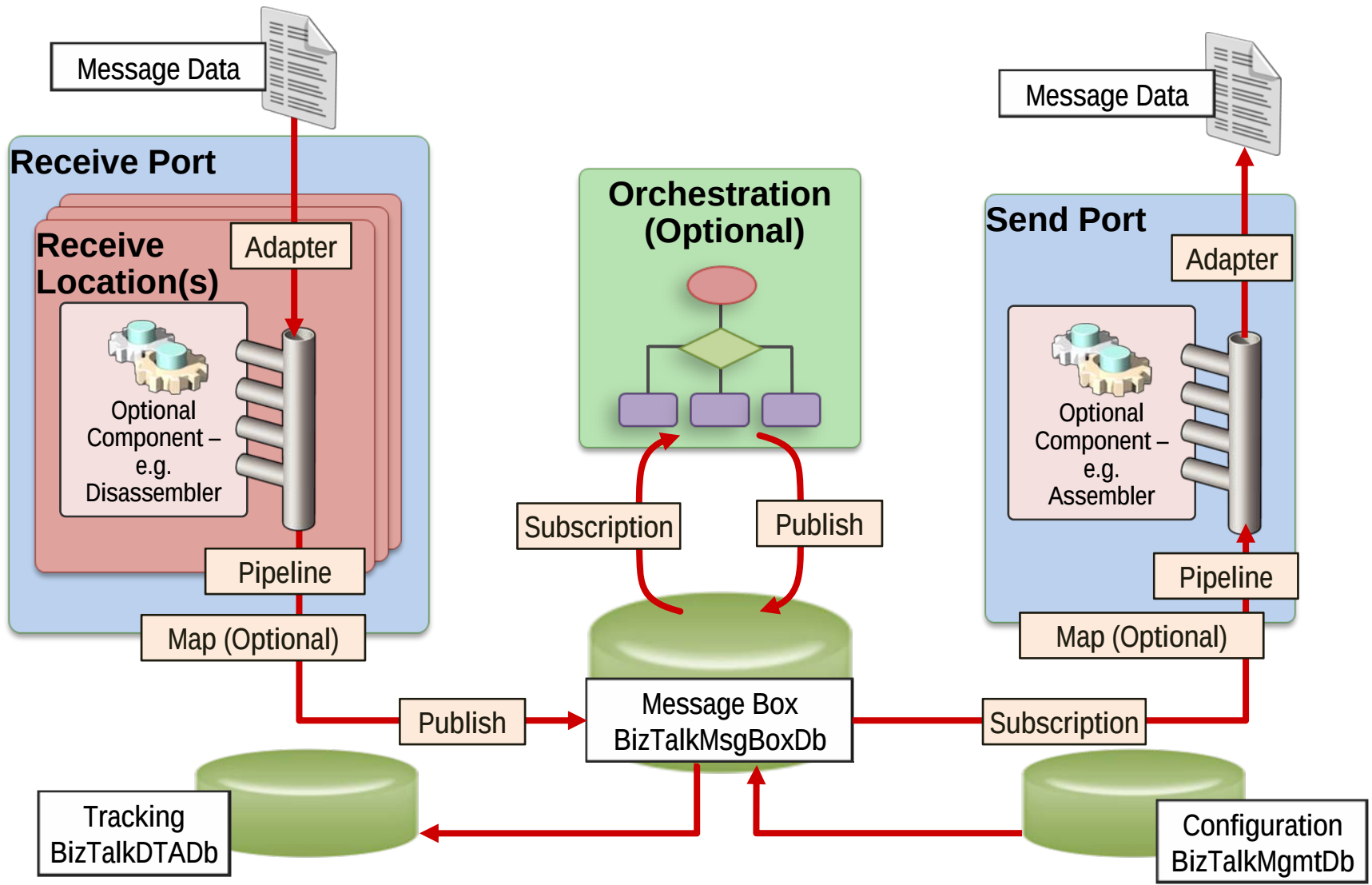
What Is a Port?

Ports specify:

- How messages are sent and received by BizTalk
- The communication direction and communication pattern
- The location where messages are received or sent



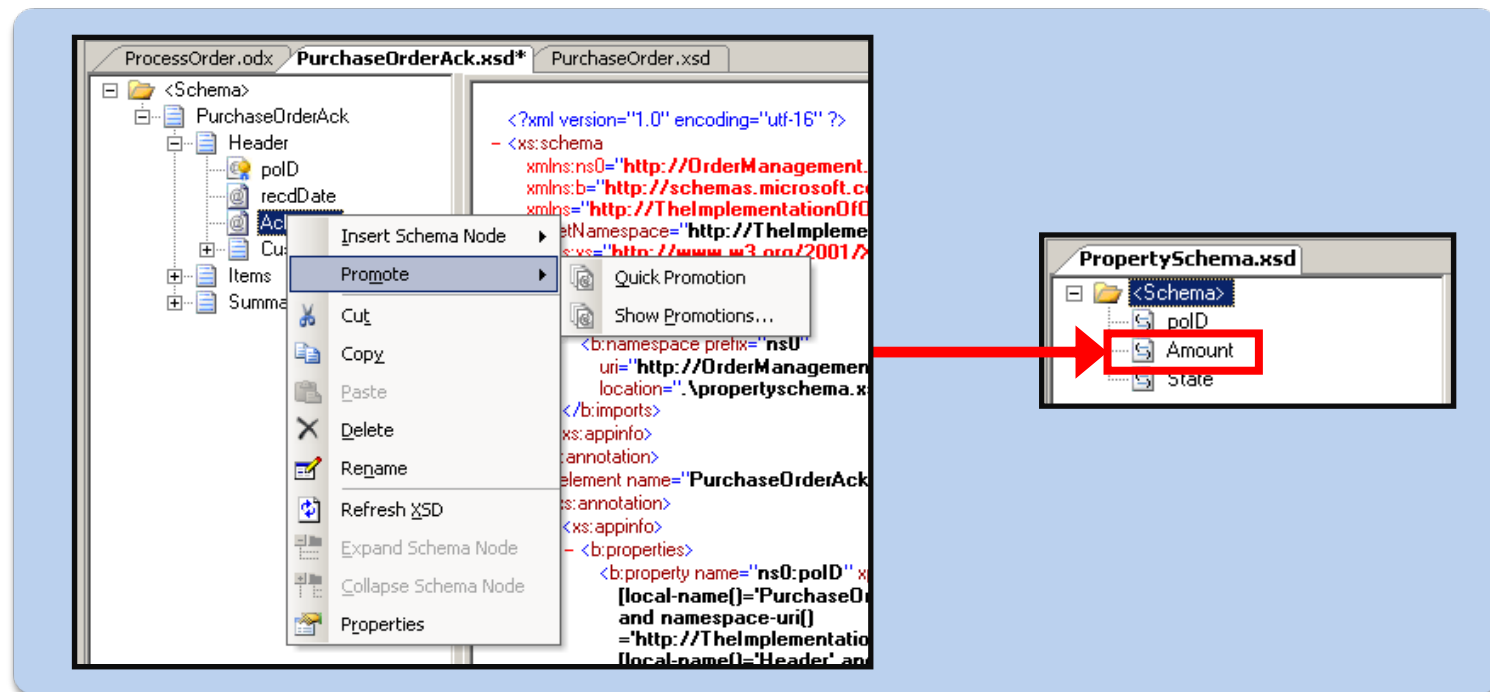
BizTalk Message Flow



What Is Property Promotion?

Property promotion

- Enables content-based routing
- Exposes message content to the MessageBox database
- Uses BizTalk specific “property schemas”



Demo: Promoting a Property

In this demonstration, you will see....

- Promoting a schema property
- Configuring QuickPromotion

Lesson 2: Configuring Message Routing

In this lesson, you will learn about...

- Steps for Enabling Message Routing
- Configuring a Receive Port and Receive Location
- Demonstration: Configuring a Receive Port and a File Receive Location
- Creating and Configuring a Send Port
- Configuring Send Ports
- Using a Send Port Group
- Enlisting and Starting a Send Port
- Demo: Configuring a Send Port and Send Port Group

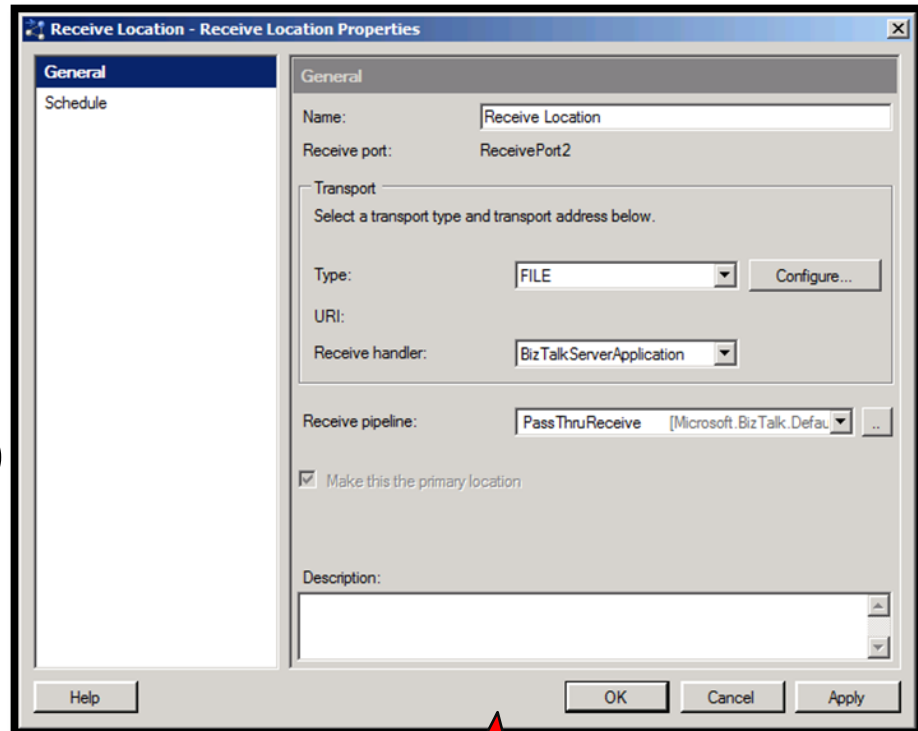
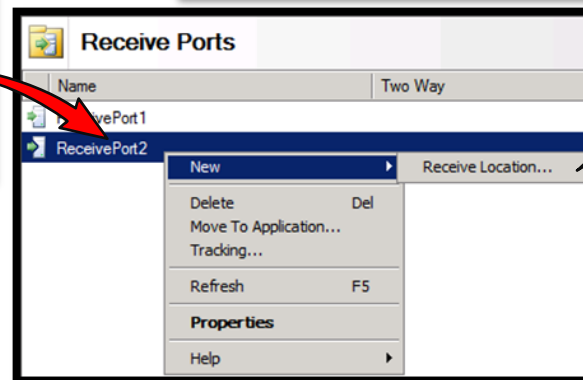
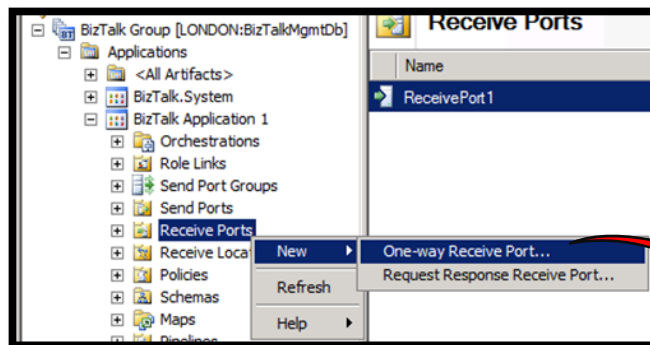
Steps for Enabling Message Routing

1. Promote fields to be filtered
2. Create receive port and receive location
3. Create send ports or send port groups
4. Create and configure send port filters
5. Start and enable the newly created ports

Configuring a Receive Port & Receive Location

Receive location properties:

- Receive handler (host)
- Enabled or disabled
- Start/stop receiving (date)
- Operating window (time period)



Demo: Configuring a Receive Port & Location

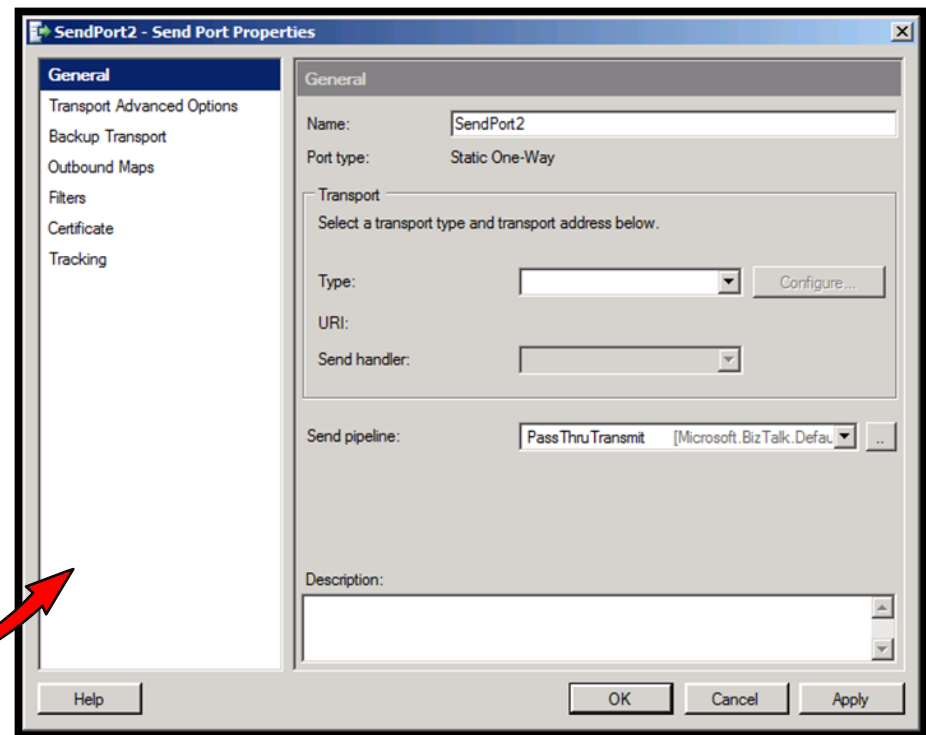
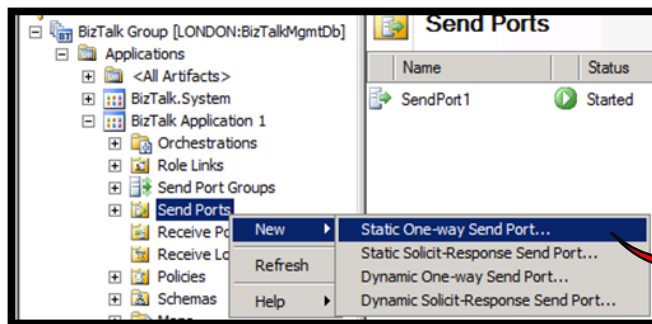
In this demonstration, you will see...

- Creating a receive port
- Configuring a receive port to use a BizTalk map
- Creating a receive location
- Configuring a receive location
- Enabling a receive location

Creating and Configuring a Send Port

Send port properties:

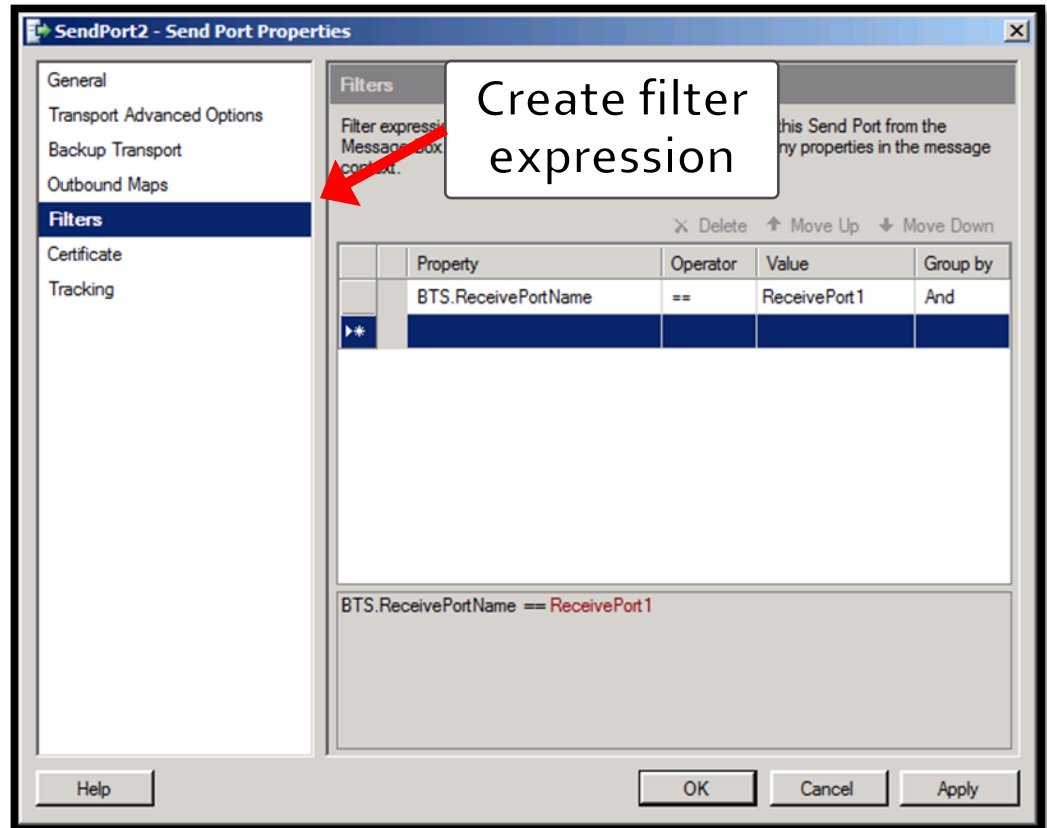
- Send port name
- Primary and secondary transport
- Address (URI)
- Retry count
- Service window enabled or disabled



Configuring Send Ports

Filtering

- Filter messages based on an expression
- Route to a send port
- Use to subscribe to failed messages



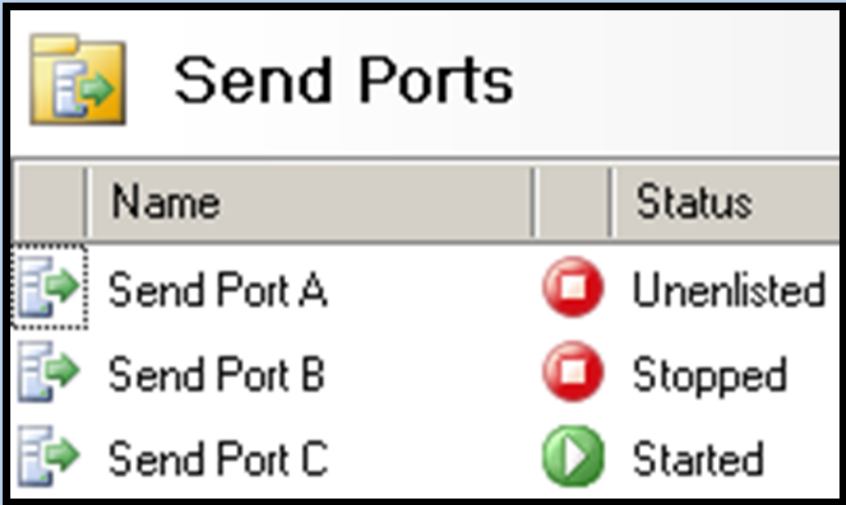
Send port group:

- A named collection of send ports
- Used to send the same message to multiple destinations
- Can be used to route based on message content by using a filter expression

Enlisting and Starting a Send Port

Port states:

- Unenlisted – no subscription (default)
- Stopped – subscription exists, messages queued
- Started – subscription exists, messages processed



	Name	Status
	Send Port A	 Unenlisted
	Send Port B	 Stopped
	Send Port C	 Started

Demo: Configure a Send Port & a Send Port Group

In this demo, you learn about...

- Creating a send port
- Configuring a send port
- Creating a send port filter
- Creating a send port group
- Changing the state of a send port

Lesson 3: Monitor Messages & Orchestrations

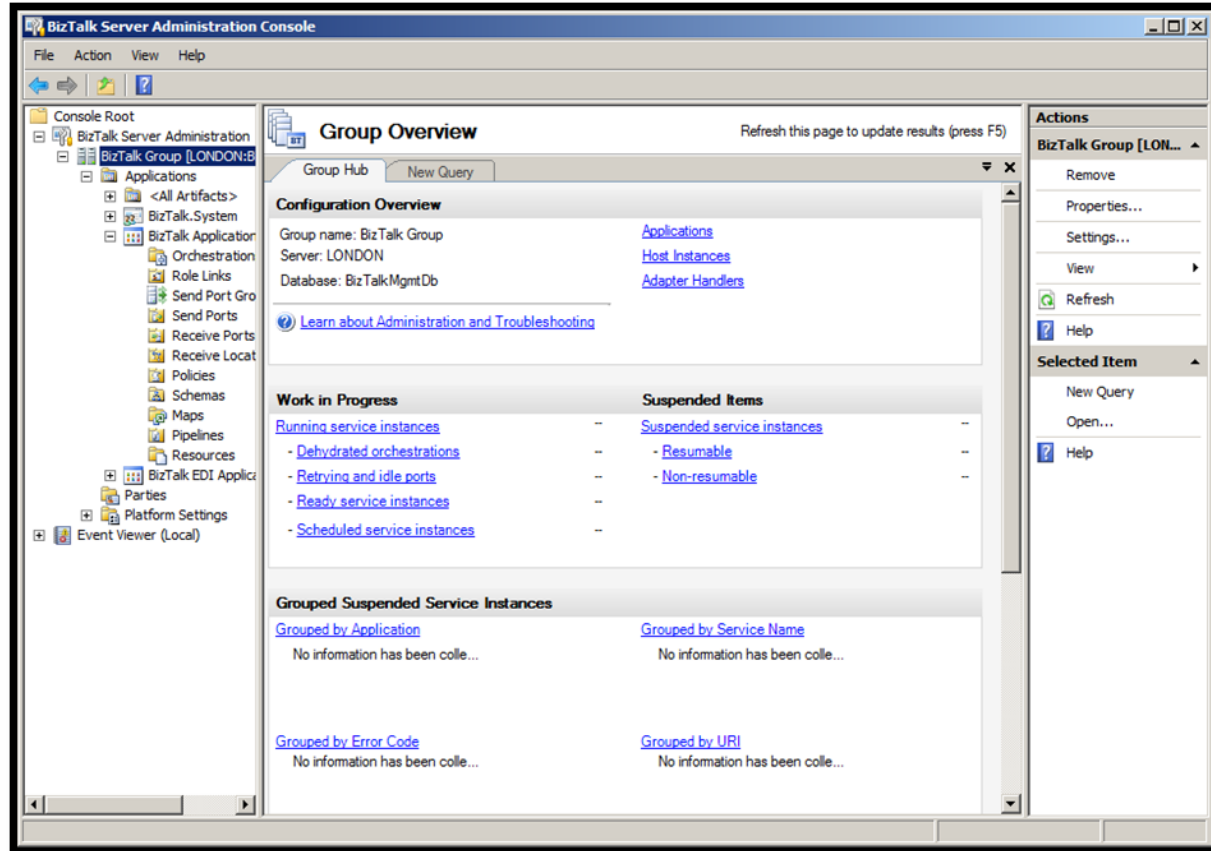
In this lesson, you will learn about...

- What Is the BizTalk Group Hub Page?
- Running a Query in the Group Hub
- What Is Health and Activity Tracking (HAT)?
- Identifying Events and Data that Can Be Tracked
- Viewing and Tracking Message Activity
- Demonstration: Tracking and Viewing Message Activity

What Is the BizTalk Group Hub Page?

Group Hub

- Configuration overview
- Work in progress
- Suspended items
- Grouped suspended service instances



Running a Query in the Group Hub

The screenshot displays the BizTalk Group Overview interface. On the left, the 'Console Root' tree shows the hierarchy: BizTalk Server Administration > BizTalk Group [LONDON:Biz] > Applications > BAM Lab. The main area is titled 'Group Overview' and contains a 'Group Hub' tab and a 'New Query' button. Below this is a 'Query Expression' table with the following data:

Field Name	Operator	Value
Search For	Equals	Running Service Instances
Service Class	Equals	Orchestration
Instance Status	Equals	Dehydrated
Maximum Matches	Equals	50

To the right of the table are three buttons: 'Run Query' (highlighted with a green border), 'Save As...' (highlighted with a purple border), and 'Open Query...'. Below the query expression is a table showing 'Query results (29 items were found):'

Service Name	Application
TheImplementati...	BAM Lab
TheImplementati...	BAM Lab
TheImplementati...	BAM Lab
TheImplementati...	BAM Lab
TheImplementati...	BAM Lab
TheImplementati...	BAM Lab
TheImplementati...	BAM Lab
TheImplementati...	BAM Lab
TheImplementati...	BAM Lab
TheImplementati...	BAM Lab

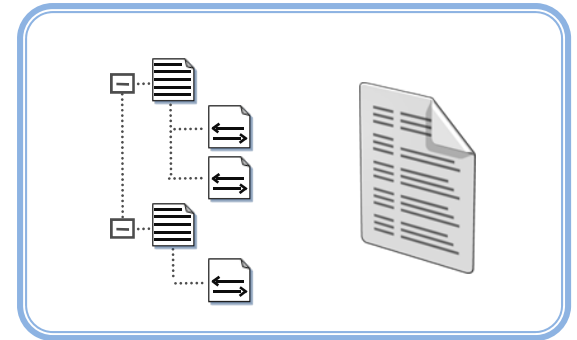
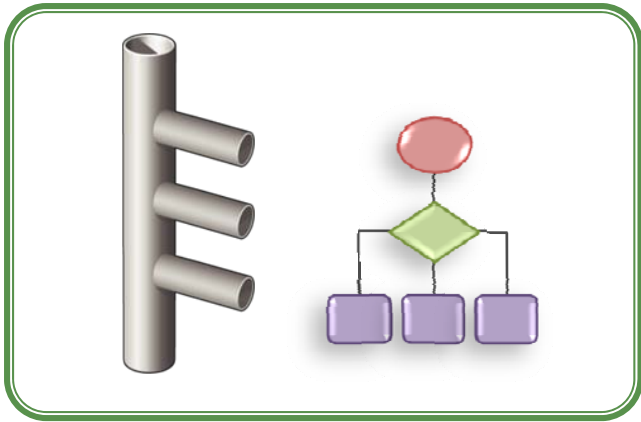
On the right, a 'Message Details' window is open, showing the message type 'http://TheImplementationOfOrderMgmt.PurchaseOrder#PurchaseOrder' and ID '045350c4-b...'. The 'body' tab is selected, displaying the following XML:

```
<po:PurchaseOrder xmlns:po="http://TheImplementationOfOrderMgmt.PurchaseOrder">
  <Header poID="PO_13" poDateTime="2005-04-20T10:48:31Z">
    <ShipTo Name="Unlimited Ltd." Address="1 Main Square" City="Portland" State="OR" Zip="97032" />
    <BillTo Name="Unlimited Ltd." Address="1 Main Square" City="Portland" State="OR" Zip="97032" />
  </Header>
  <Item Product="Flash MC" Quantity="64" Price="3.20" />
  <Summary Amount="204.80" />
  <Timing EvalTime="2" FulfillmentTime="1" RejectionTime="6" />
</po:PurchaseOrder>
```

Configuring Event and Data Tracking

Event Tracking

- Start and end of a service
- Sending or receiving of a message
- Start and end of a pipeline
- Start and end of an orchestration
- Execution of each orchestration shape



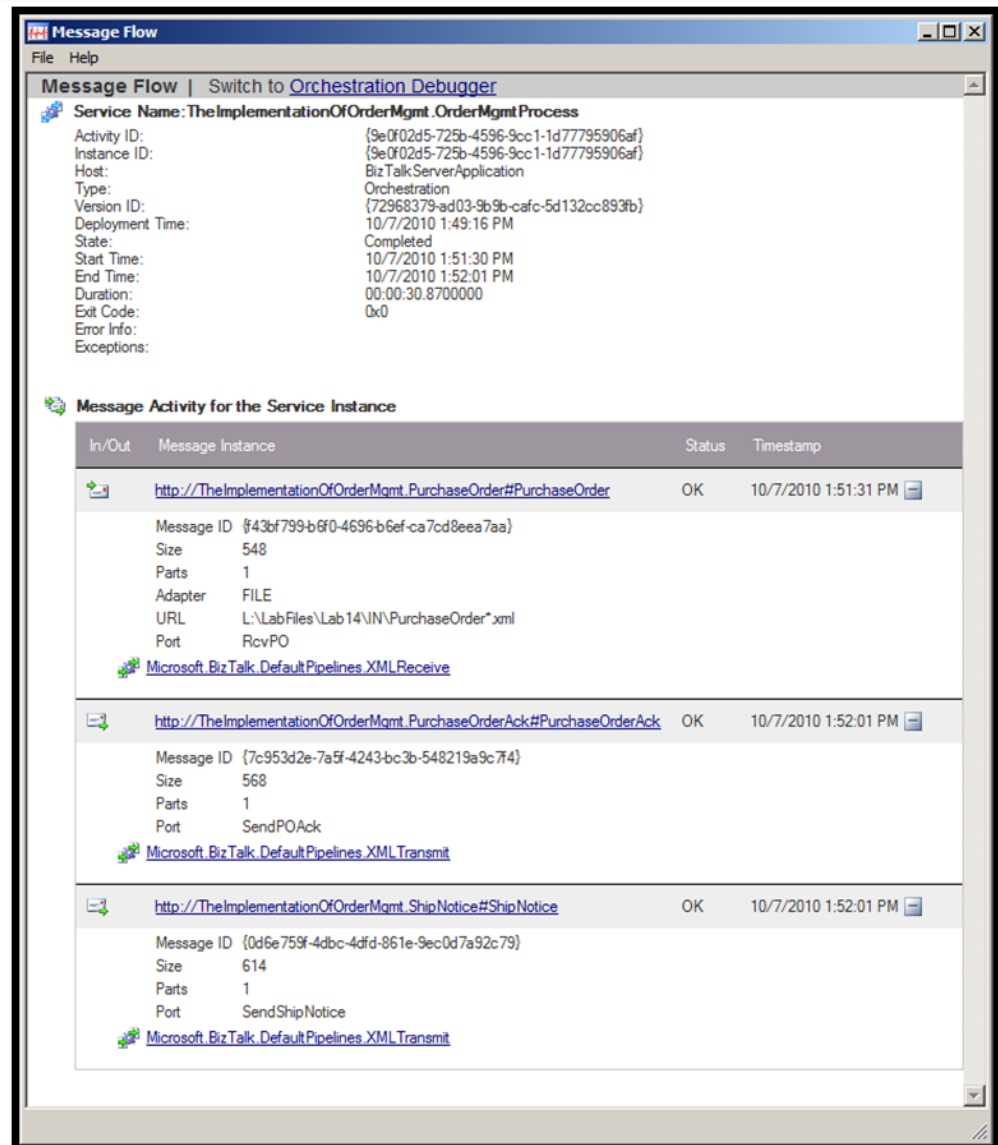
Data Tracking

- Message contents
- Promoted properties

Viewing and Tracking Message Activity

View Message Activity

- Track path of completed instances
- Launch the Orchestration Debugger
- Accessed from Group Hub queries



The screenshot displays the 'Message Flow' window in the Orchestration Debugger. The window title is 'Message Flow' and it includes a 'File' menu and a 'Help' button. The main content area is titled 'Message Flow | Switch to Orchestration Debugger'.

Service Name: TheImplementationOfOrderMgmt_OrderMgmtProcess

Activity ID: {9e0f02d5-725b-4596-9cc1-1d77795906af}

Instance ID: {9e0f02d5-725b-4596-9cc1-1d77795906af}

Host: BizTalkServerApplication

Type: Orchestration

Version ID: {72968379-ad03-9b9b-cafc-5d132cc893fb}

Deployment Time: 10/7/2010 1:49:16 PM

State: Completed

Start Time: 10/7/2010 1:51:30 PM

End Time: 10/7/2010 1:52:01 PM

Duration: 00:00:30.8700000

Exit Code: 0x0

Error Info:

Exceptions:

Message Activity for the Service Instance

In/Out	Message Instance	Status	Timestamp
In	http://TheImplementationOfOrderMgmt_PurchaseOrder#PurchaseOrder Message ID {43bf799-b6f0-4696-b6ef-ca7cd8eea7aa} Size 548 Parts 1 Adapter FILE URL L:\LabFiles\Lab 14\IN\PurchaseOrder*.xml Port RcvPO Microsoft.BizTalk.DefaultPipelines.XMLReceive	OK	10/7/2010 1:51:31 PM
Out	http://TheImplementationOfOrderMgmt_PurchaseOrderAck#PurchaseOrderAck Message ID {7c953d2e-7a5f-4243-bc3b-548219a9c7f4} Size 568 Parts 1 Port SendPOAck Microsoft.BizTalk.DefaultPipelines.XMLTransmit	OK	10/7/2010 1:52:01 PM
Out	http://TheImplementationOfOrderMgmt_ShipNotice#ShipNotice Message ID {0d6e759f-4dbc-4dfd-861e-9ec0d7a92c79} Size 614 Parts 1 Port SendShipNotice Microsoft.BizTalk.DefaultPipelines.XMLTransmit	OK	10/7/2010 1:52:01 PM

Demo: Tracking & Viewing Message Activity

In this demonstration, you will see how to....

- View subscriptions using the Group Hub
- View running and suspending instances using the Group Hub
- Examine message flow for completed instances

Lab: Routing BizTalk Messages

In this 45 minute lab, you will....

- Add an Existing Schema and Map to the Project
- Promote Schema Properties
- Create a Receive Port and a Receive Location
- Create a Send Port for All Orders
- Create a Send Port for Cash Orders
- Create a Send Port for Credit Orders
- Test the Port Configuration

Review Questions

- In what format is the message while it is in the MessageBox database?
- At which stage of processing does content of the message become context?

Creating Pipelines

- Lesson 1: Introduction to Pipelines
- Lesson 2: Building a Pipeline

Lesson 1: Introduction to Pipelines

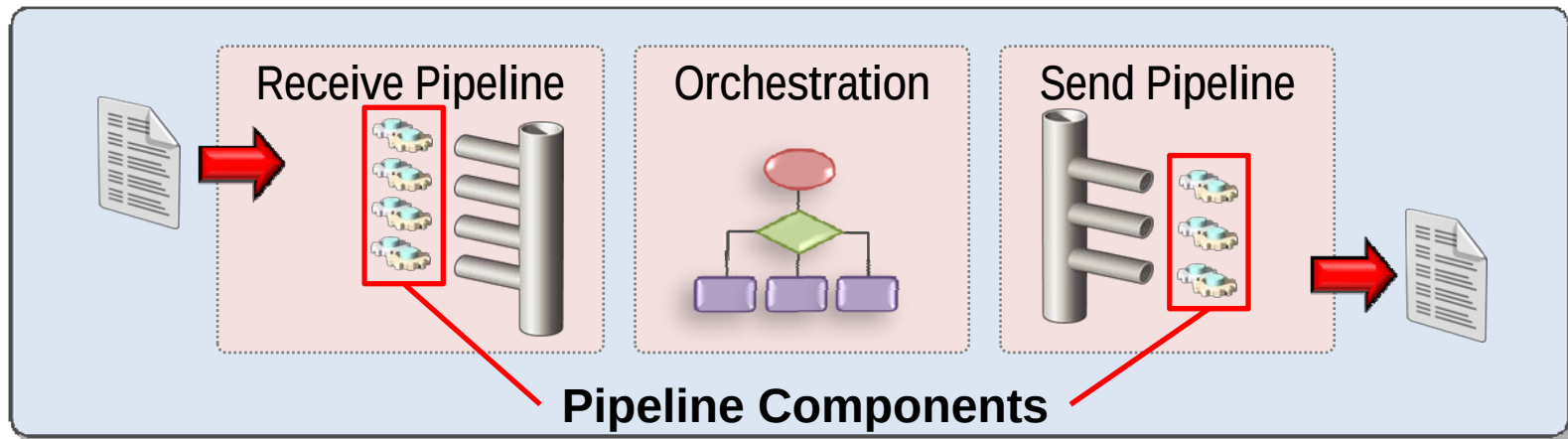
In this lesson, you will learn about....

- What Is a Pipeline?
- Pipeline Scenarios
- Receive Pipeline Stages
- Send Pipeline Stages
- Standard Pipeline Components
- What Are the Default Pipelines?
- What Are Custom Pipeline Components?

What Is a Pipeline?

Pipelines:

- Perform data translation during the receipt or transmission of messages
- Have a number of stages
- Each stage is designed for specific types of components
- Components provide translation behaviors



Receive Pipelines:

- Decrypt inbound messages from trading partners
- Split batched messages
- Validate messages against known schemas
- Verify the sender of a message for non-repudiation

Send Pipelines:

- Encrypt outbound messages to trading partners
- Digitally sign outbound messages
- Provide a wrapper for outbound messages
- Validate messages against known schemas

Receive Pipeline Stages

Decode

- Used to decode or decrypt messages
- MIME/SMIME, ZIP, PGP, etc.

Disassemble

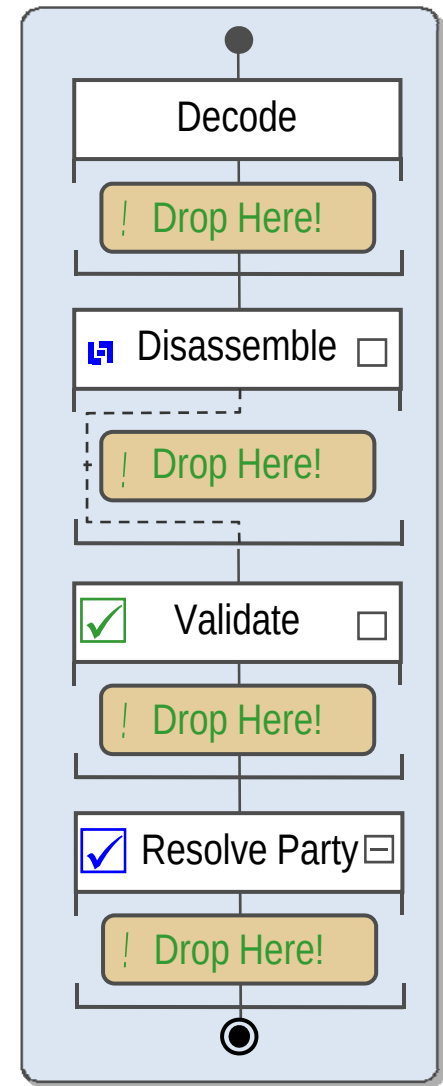
- Translates native message to XML
- Resolves message schema
- Creates promoted properties
- Enables de-batching

Validate

- Used to validate the XML message
- Will run for each message created by disassembler

Resolve Party

- Used to verify the send of the message



Send Pipeline Stages

Pre-assemble

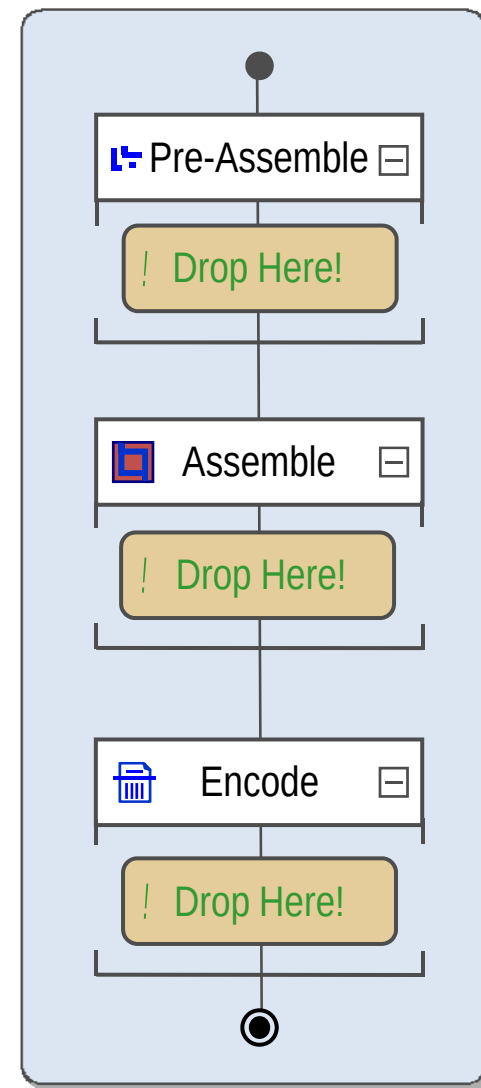
- Designed for custom components
- Last chance to modify the message

Assemble

- Used to create the message's "final form"
- Add envelopes, headers, and trailers
- Convert from XML to native format

Encode

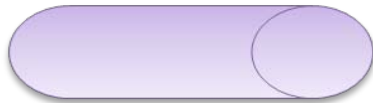
- Used to encode or encrypt messages
- MIME/SMIME, PGP, ZIP, etc.



Standard Pipeline Components

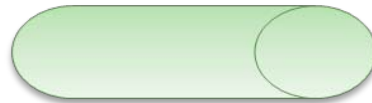
Receive Pipeline

Decode



MIME/SMIME
Decoder

Dissassemble

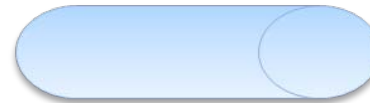


XML
Disassembler

Flat File
Disassembler

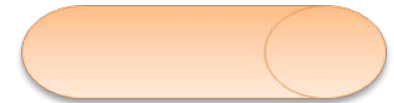
EDI
Disassembler

Validate



XML Validator

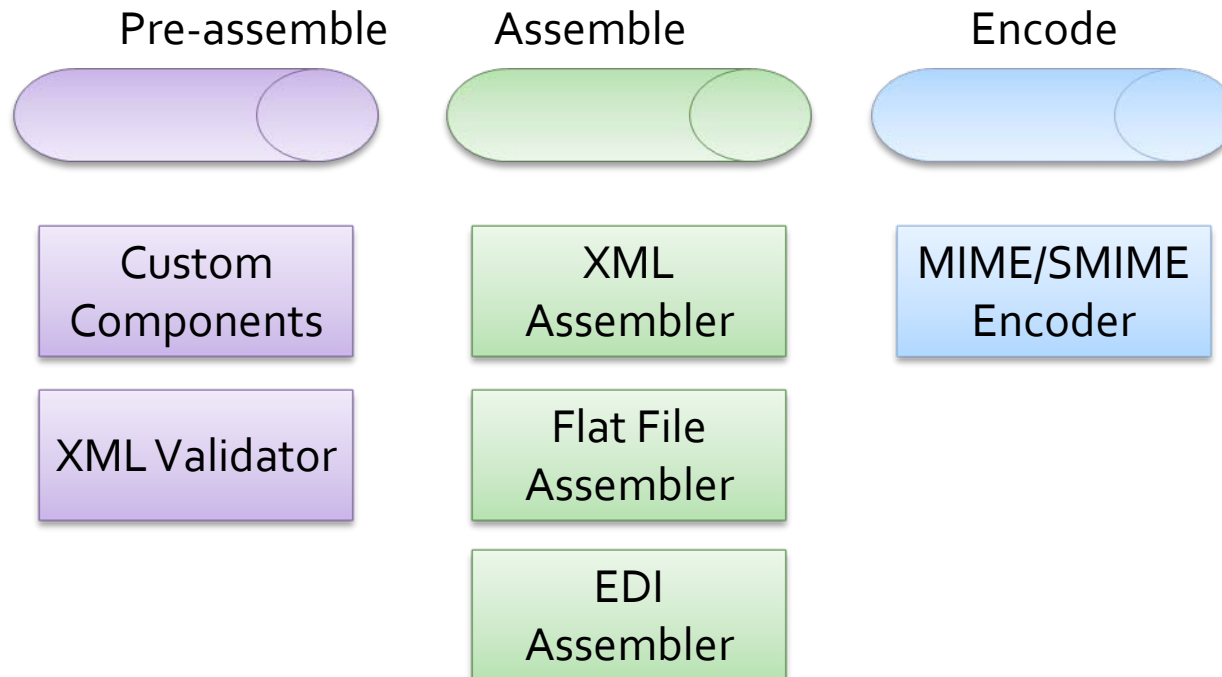
Resolve Party



Party
Resolution

Standard Pipeline Components

Send Pipeline



What Are the Default Pipelines?

Pipeline	Usage
Pass-Through Receive (<i>PassThruReceive</i>)	• Contains no components • No decoding, disassembling, or validation required
Pass-Through Send (<i>PassThruTransmit</i>)	• Contains no components • No encoding or assembling required • Use for basic message routing
XML Receive (<i>XMLReceive</i>)	• Contains the XML disassembler component • Party resolution component for security • No decoding or validation stage
XML Send (<i>XMLTransmit</i>)	• Contains the XML assembler component • Other stages empty by default
<i>EDI and AS2 pipelines are discussed in detail in Module 15</i>	

What Are Custom Pipeline Components?

Custom Pipeline Components

- Use to extend message processing capabilities
- Use when standard pipeline components are not sufficient

Use Scenarios

- Non-XML validations
- Non-supported decryption algorithms
- Non-supported signature formats
- Custom data conversions
- Advanced routing requirements

Check the BizTalk community before building your own

Lesson 2: Building a Pipeline

In this lesson, you will learn about...

- Using the Pipeline Designer
- Creating Schemas for Batched Messages
- Configuring Disassembler Properties
- Pipeline Test Tools
- Demonstration: Creating and Testing a Pipeline

Using the Pipeline Designer

The screenshot displays the BizTalk Pipeline Designer interface with the following components:

- Toolbox:** Located on the left, it contains a list of BizTalk Pipeline Components. A label "Toolbox" is placed over the bottom section of this window.
- Design Surface:** The central area where the pipeline is built. It shows a sequence of steps: Decode, Disassemble, Validate, and ResolveParty. A "Drop Here!" placeholder is visible between Decode and Disassemble. A label "Design Surface" is placed at the bottom of this window.
- Solution Explorer:** Located on the right, it shows the project structure. The file "ReceiveSalesOrder_Batch.btp" is highlighted with a red box. A label "Pipeline File (.btp)" is placed over this box.
- Properties Window:** Located at the bottom right, it displays the properties of the selected pipeline file. A label "Properties Window" is placed over this window.

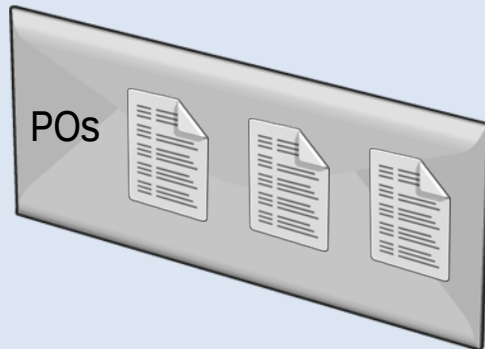
The Properties Window shows the following details:

Microsoft.VisualStudio.BizTalkProject.CBtsFile	
Advanced	
Build Action	BtsCompile
Fully Qualified Name	AdvWorks.Pipelines.Receive
Namespace	AdvWorks.Pipelines
Type Name	ReceiveSalesOrder_Ba
Misc	
File Name	ReceiveSalesOrder_Ba
Full Path	E:\LabFiles\Lab1\AdvWorks

Creating Schemas for Batched Messages

Flat Files

Sender: Contoso
Recipient: AdvWorks



Header_FF.xsd (Company, address, batch#)



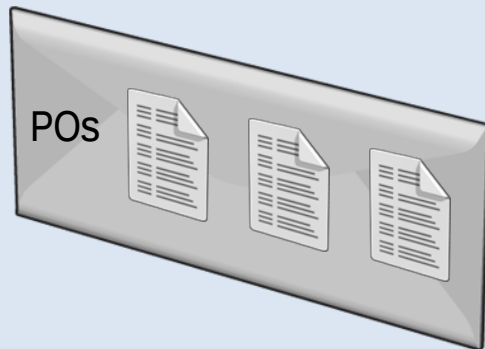
PurchaseOrder_FF.xsd



Trailer_FF.xsd

XML Files

Sender: Contoso
Recipient: AdvWorks



Envelope.xsd (Envelope attributes)

<Any>



PurchaseOrder.xsd

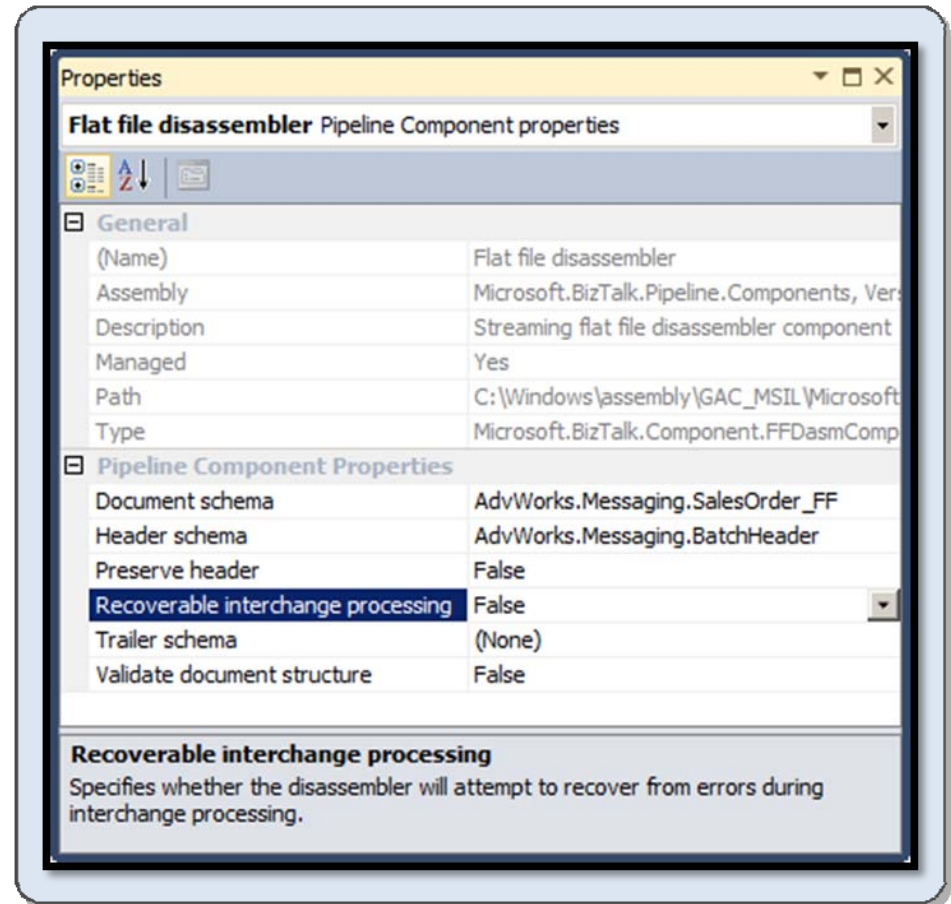
Configuring Disassembler Properties

May Require Multiple Schemas

- Header (Envelope)
- Body
- Trailer

Interchange Processing

- Standard
- Recoverable



Pipeline Test Tools

Tools	Usage
Pipeline.exe	• Accepts XSD schemas and input message • Produces output document
FFAsm.exe	• Executes the flat-file assembler component
FFDasm.exe	• Executes the flat-file disassembler component
XMLAsm.exe	• Executes the XML assembler component
XMLDasm.exe	• Executes the XML disassembler component

Demo: Creating & Testing a Pipeline

In this demonstration, you will learn about...

- Creating and configuring a receive pipeline
- Creating and configuring a send pipeline
- Using the pipeline command-line tools

In this 60 minute lab, you will....

- Configure a Send Pipeline to Encrypt Outgoing Messages
- Configure a Send Port to Use the Encryption Pipeline and Certificate
- Configure a Receive Pipeline to Disassemble a Message Interchange
- Configure a Receive Location to Use the Pipeline
- Submit Test Messages
- Enable and Test Recoverable Interchange

Review Questions

- What does the parameterized pipeline configuration enable you to do?
- Can you think of some reasons you'd need a custom pipeline component?

Integrating with Adapters

- Lesson 1: Introduction to BizTalk Adapters
- Lesson 2: Configuring a BizTalk Adapter

Lesson 1: Introduction to BizTalk Adapters

In this lesson, you will learn about...

- What Is an Adapter?
- Sources of Adapters
- What Is the BizTalk Adapter Framework

What Is an Adapter?

Adapters

- .NET or COM software components
- Connect BizTalk to specific applications and systems
- Can use standard protocols (HTTP, FTP, POP3)
- Can use proprietary communication mechanisms
- Support one-way or two-way communications
- Add metadata to data stream of received messages

Adapter Sources

- Standard BizTalk Server adapters
- Microsoft add-on adapters
- Third-party adapters
- Custom adapters

BizTalk Adapter Framework

- APIs for access to BizTalk shared services
- Includes samples and a template for creating custom adapters
- Can use standard BizTalk administration tools to manage and configure the adapter

WCF LOB Adapter SDK

- Provides a consistent and repeatable way to build LOB adapters for the .NET platform
- Use WCF to create adapters for endpoints other than BizTalk Server

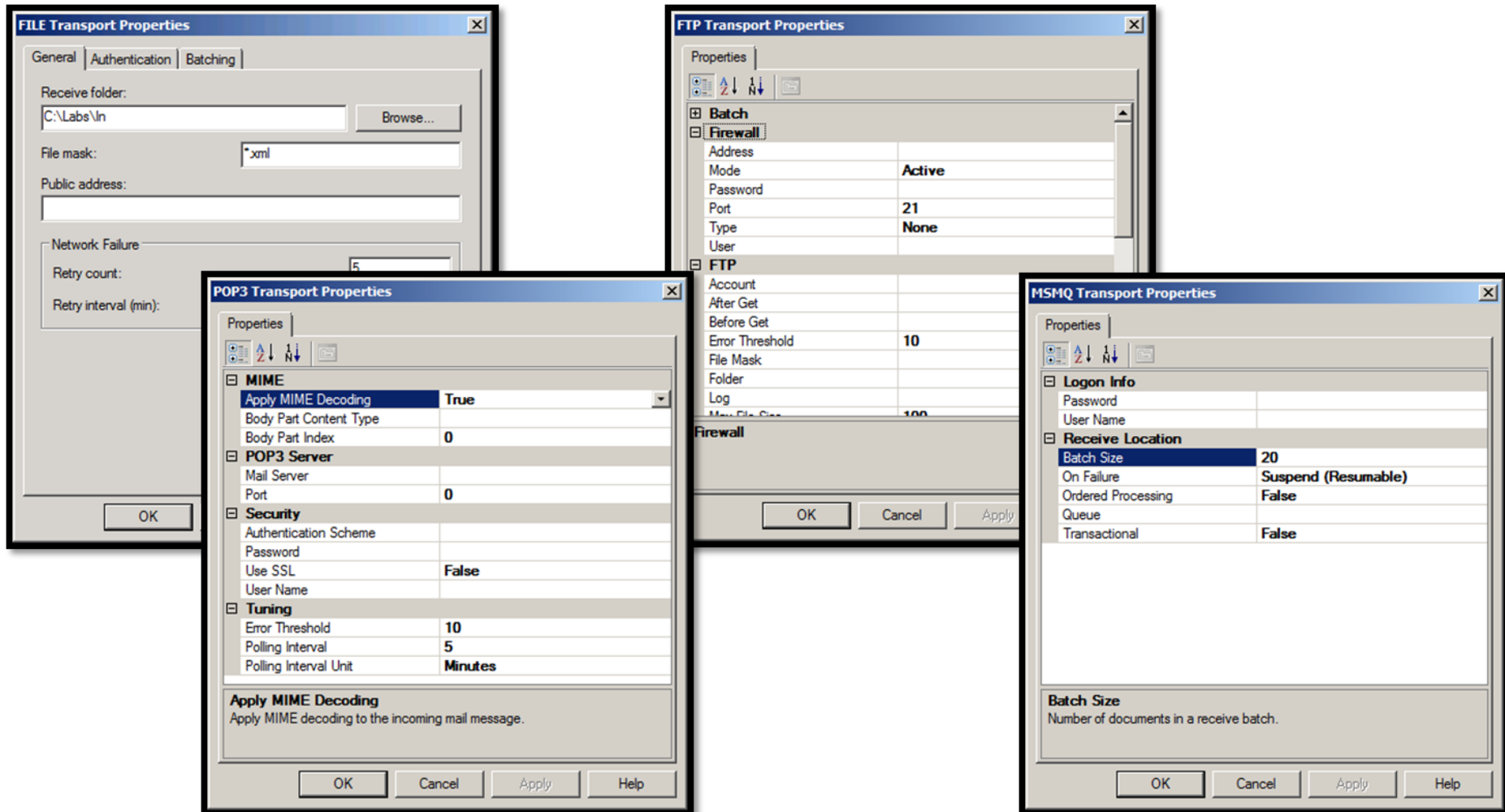
Lesson 2: Configuring a BizTalk Adapter

In this lesson, you will learn about...

- Configuring Adapters
- Integrating with Protocol Adapters
- Demonstration: Configuring a Protocol Adapter
- Integrating with Data Adapters
- Demonstration: Configuring a Data Adapter
- Integrating with Application Adapters
- Demonstration: Integrating with SharePoint
- Integrating with WCF Applications

Configuring Adapters

Properties vary from one adapter to another



BizTalk Server 2010 Native Adapters

- SAP
- Siebel eBusiness Applications
- Oracle eBusiness Suite
- Oracle Database
- SQL Server
- PeopleSoft Enterprise
- JD Edwards OneWorld XE
- JD Edwards EnterpriseOne
- TIBCO Rendezvous
- TIBCO Enterprise Message Service
- IBM DB2
- WebSphere MQ
- SharePoint
- WCF
- FTPs
- HTTP
- MSMQ
- POP3
- SMTP
- SOAP

Demo: Configuring Adapters

In this demonstration, you see how to...

- Configure the FTP adapter
- Configure the POP3 adapter
- Configure the MSMQ adapter

Creates BizTalk Objects

- Supported by many adapters
- You provide connection details
- Wizard generates schemas and orchestrations from connection

Generating Adapter Metadata

The image illustrates the steps to generate adapter metadata in Visual Studio:

- Visual Studio Menu:** The **Add** menu is open, showing options like **Add Reference...**, **Add Service Reference...**, **View Class Diagram**, **Set as StartUp Project**, **Debug**, **Add Solution to Source Control...**, **Cut**, **Paste**, **Remove**, **Rename**, **Unload Project**, **Open Folder in Windows Explorer**, and **Properties**. The **Add** menu is highlighted, and the **Add Generated Items...** option is selected.
- Add Generated Items - Processes Dialog:** This dialog shows the **Installed Templates** section. The **Add Adapter Metadata** option is selected under the **Generated Schema Files** category. Other options include **Generate Schemas**, **WCF Service**, and **Adapter Service**.
- Add Adapter Wizard Dialog:** This wizard guides the user through selecting an adapter and configuring the database connection.
 - Select Adapter:** The user is prompted to select an adapter to query for metadata information. The list of registered adapters includes: J.D. Edwards EnterpriseOne(r), WCF-SQL, SQL, WCF-Siebel, J.D. Edwards OneWorld(r), WCF-SAP, and WCF-OracleDB.
 - Database Configuration:** The user is prompted to enter the **SQL Server** (LONDON), **Database** (BizTalkMgmtDb), and **Port**.
 - Navigation:** The wizard includes **Help**, **Back**, **Next**, and **Cancel** buttons.

Demo: Configuring a Data Adapter

In this demonstration, you will learn about...

- Generating SQL adapter metadata
- Examining the SQL metadata

Many BizTalk applications need human interaction

Options for integrating with humans include:

- Email
- File folders
- Custom web sites or applications
- Existing applications
- SharePoint

Demo: Integrating with SharePoint

In this demonstration, you will learn about...

- Configuring the SharePoint receive adapter
- Configuring the SharePoint send adapter
- Testing the SharePoint receive and send adapters

Integrating with WCF Applications

BizTalk WCF Adapters

- Allow BizTalk applications to communicate with WCF-based applications
- Includes seven adapters
 - Five represent predefined WCF bindings
 - Two allow custom configuration of bindings and behaviors for receive locations and send ports

List of WCF Adapters

- WCF-WSHttp
- WCF-BasicHttp
- WCF-NetTcp
- WCF-NetMsmq
- WCF-NetNamedPipe
- WCF-Custom
- WCF-CustomIsolated

Features of the WCF-BasicHttp Adapter

- Communicates with legacy Web services that adhere to the WS-I Basic Profile 1.1
 - Useful when using ASMX Web services
- Uses the HTTP(S) transport layer
- Supports message encoding in MTOM or Text format

Features of the WCF-WSHttp Adapter

- Provides the WS-* standards over HTTP transport
- Runs in an isolated IIS process
- Requires an IIS Virtual Directory application setup to host the WCF service
- Currently does not support WS-RM (Reliable Messaging) or WS-BA (Business Activities)

Lab: Integrating with Adapters

In this 60 minute lab, you will...

- Configure and Test the POP3 Adapter
- Configure and Test the FTP Adapter
- Configure and Test the Windows SharePoint Services Adapter
- Configure and Test the SMTP Adapter
- Configure and Test the WCF-SQL Adapter

Review Questions

- Why are adapters important to BizTalk's processing?
- How can you create highly-available receive and send processing?

Creating BizTalk Orchestrations

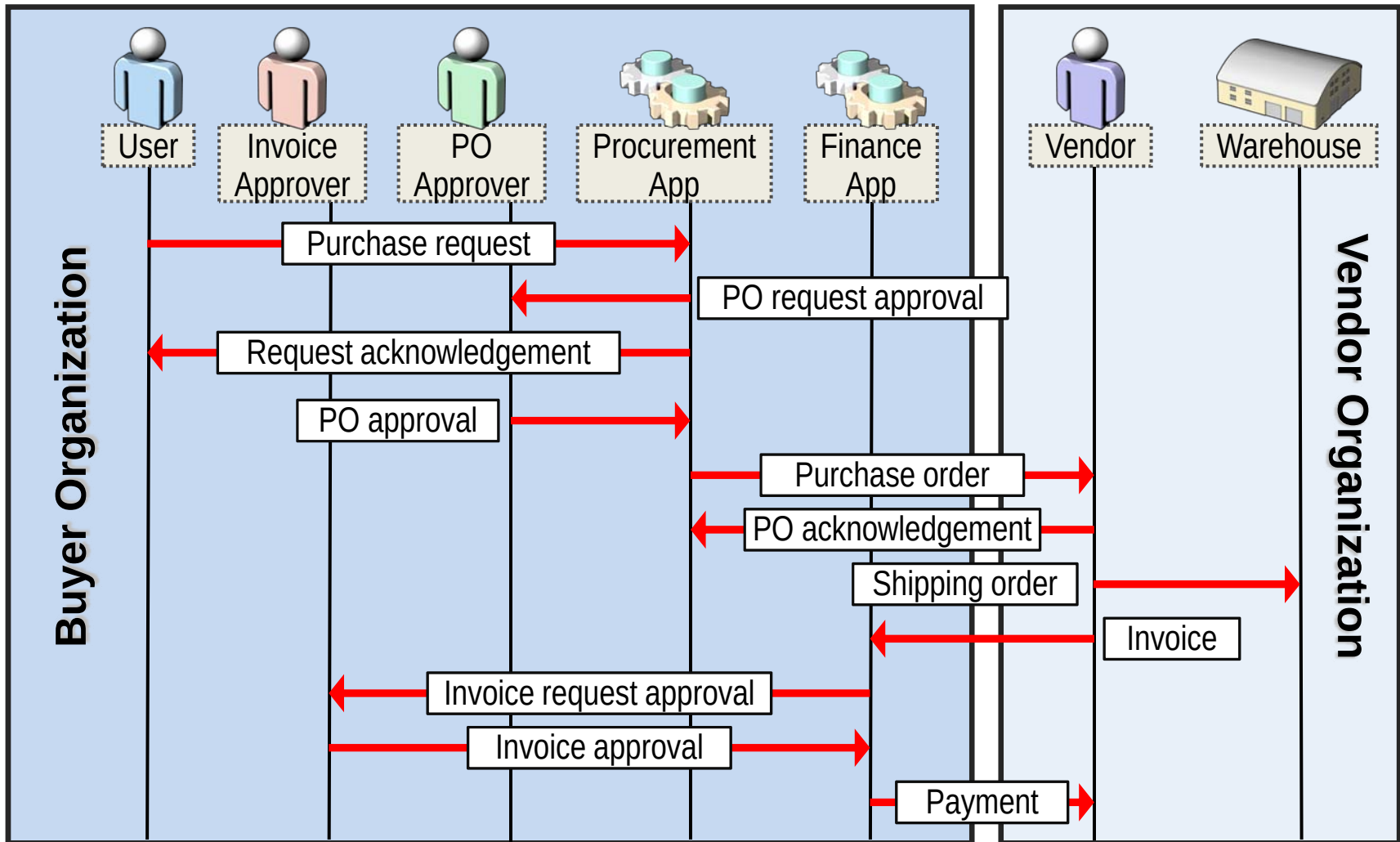
- Lesson 1: Introduction to BizTalk Orchestration
- Lesson 2: Building an Orchestration
- Lesson 3: Monitoring Orchestrations

Lesson 1: Intro to BizTalk Orchestrations

In this lesson, you will learn about...

- Modeling a Business Process
- What Is an Orchestration?
- How the Orchestration Engine Works
- Stages of Orchestration Development

Modeling a Business Process



What Is an Orchestration?

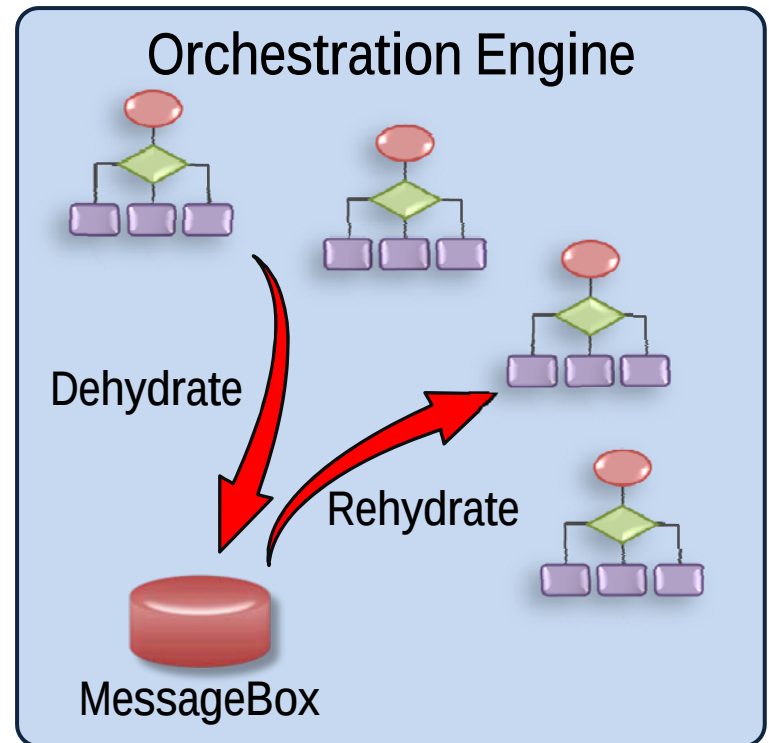
Orchestrations Provide

- Graphical representation of a business process
- Ability to work with multiple messages
- Coordination of multiple sends and receives
- Management of long-running processes

How the Orchestration Engine Works

Orchestration engine

- Creates orchestration instances
- Maintains state of orchestrations
- Performs resource optimizations
- Provides reliable shutdown and recovery



Stages of Orchestration Development

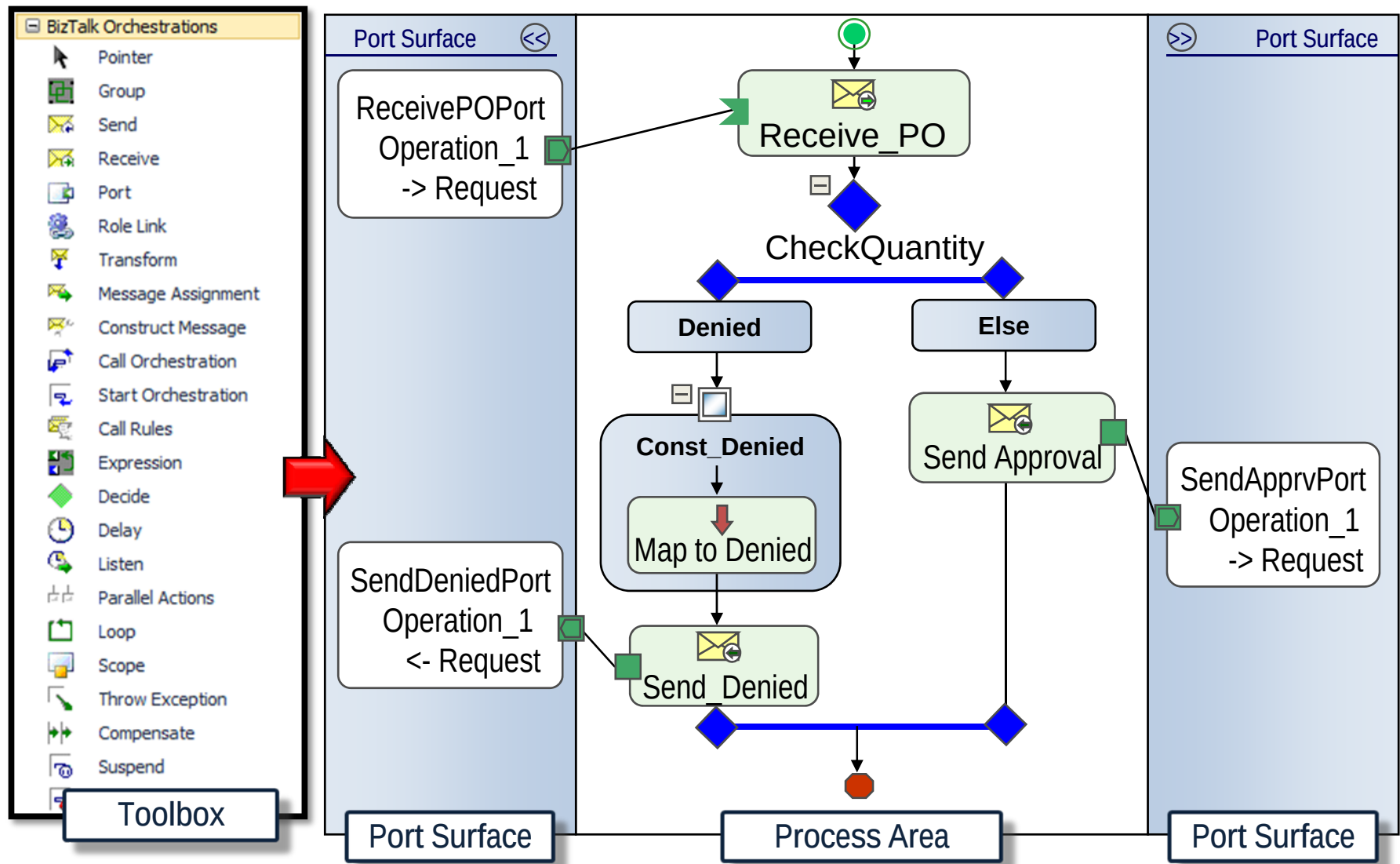
1. Define schemas
2. Add orchestration shapes
3. Assign or transform data
4. Define send and receive ports
5. Define orchestration variables and types
6. Bind send and receive shapes to ports
7. Build and deploy the orchestration
8. Test the orchestration

Lesson 2: Building an Orchestration

In this lesson, you will learn about...

- What Is the Orchestration Designer?
- Using Message and Data-Handling Shapes
- Working with Messages in an Orchestration
- Logical vs. Physical Ports
- Creating and Configuring Orchestration Ports
- Demonstration: Creating an Orchestration

What Is the Orchestration Designer?



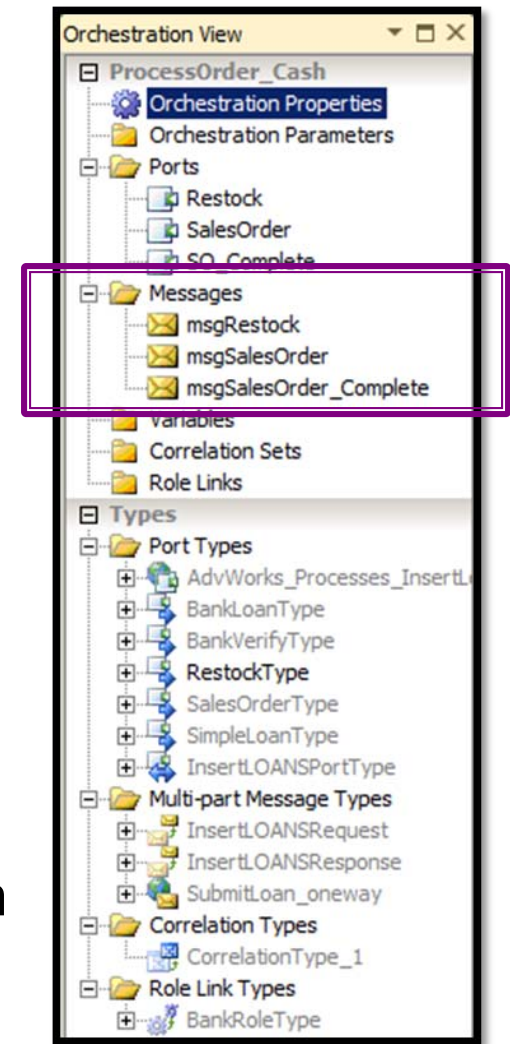
Working with Messages in an Orchestration

Use Orchestration View to define:

- Variables
- Ports and port types
- Correlation sets and correlation types
- Messages and multi-part message types

Orchestration Messages

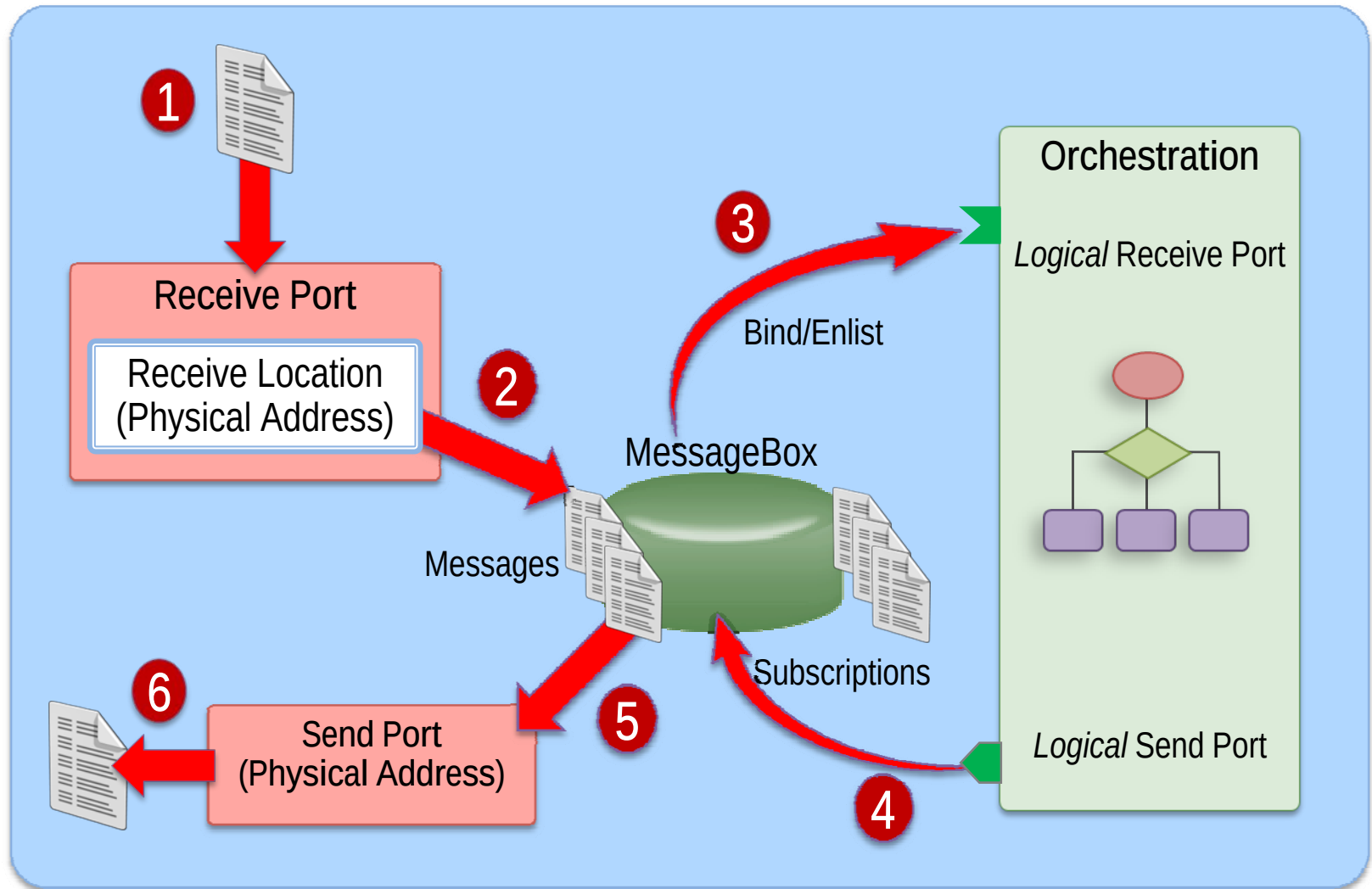
- Messages are variables
- Messages are immutable
- You must define a message for every message instance used in an orchestration



Using Message and Data-Handling Shapes

Shape	Use
Receive	• Use to receive a message from a port • Must specify message type and port type • Can apply a filter expression
Send	• Use to send a message to a port • Must specify message and port operation
Construct Message	• Use to construct a new message instance • Must always contain a Transform shape or Message Assignment shape
Message Assignment	• Use to assign one message to another • Uses Expression Editor to assign values
Transform	• Use to move data between two messages • Must specify input and output messages and a map

Logical vs. Physical Ports



Creating and Configuring Orchestration Ports

1. Create a Port Type
2. Create a Port
3. Choose Port Binding

Port Type

- Defines communication pattern (1-way or 2-way)
- Reusable type

Port

- Instance of a port type
- Defines direction

Port Binding

- Specify now ▪ Direct
- Specify later ▪ Dynamic

Demo: Creating an Orchestration

In this demonstration, you will learn about...

- Creating message variables
- Creating and configuring a receive shape
- Creating and configuring a transform shape
- Creating and configuring a send shape
- Creating a logical receive port
- Creating a logical send port

Lesson 3: Monitoring Orchestrations

In this lesson, you will learn about...

- Using the Group Hub to View Orchestrations
- Using the Orchestration Debugger
- Demo: Tracking an Orchestration Instance

Using the Group Hub to View Orchestrations

Service Instance States

- Ready to run
- Scheduled
- Retrying
- Dehydrated
- Suspended, resumable
- Suspended, non-resumable
- Active
- In breakpoint

Message Instance States

- Consumed
- Delivered (not consumed)
- Suspended, resumable
- Suspended, non-resumable
- Undelivered
- Undelivered (retrying)
- Undelivered (scheduled)

Using the Orchestration Debugger

ProcessCustomers [started] - Orchestration Debugger [replay] - {D7956445-66B3-4BFA-A2BC-8496...}

File Reporting Queries Tools Debug Help

ProcessCustomers [started] - Orchestration Debugger

Tracked Events

	Action Name	Action Type	Time
1	Initialization	Orchestrati...	10:32:39
2	Receive_1	Receive	10:32:39
3	Receive_1	Receive	10:32:39
4	ConstructMe...	Construct	10:32:39
5	ConstructMe...	Construct	10:32:40
6	Suspend Or...	Suspend	10:32:40

Orchestration

```
graph TD; Start(( )) --> Receive_1[Receive_1]; Receive_1 --> ConstructMessage_1[ConstructMessage_1]; ConstructMessage_1 --> Transform_1[Transform_1]; Transform_1 --> SuspendOrc[Suspend Orc...]; SuspendOrc --> Send_1[Send_1];
```

The diagram illustrates the orchestration flow. It begins with a green circle representing the start of the process. The first action is a green box labeled 'Receive_1' with a message icon. This is followed by a dashed box labeled 'ConstructMessage_1' which contains a blue box labeled 'Transform_1' with a message icon and a downward arrow. After the dashed box, there is a blue box labeled 'Suspend Orc...' with a pause icon. The final action is a blue box labeled 'Send_1' with a message icon. Arrows indicate the flow from one action to the next.

Orchestration Debugger

- View a rendered orchestration
- Set breakpoints
- View variables and message instance values
- Continue and resume debugging

Demo: Tracking an Orchestration Instance

In this demonstration, you will learn about...

- Debugging an orchestration instance

Lab: Creating a BizTalk Orchestration

In this 45 minute lab, you will...

- Import an existing schema and map
- Create a new project and orchestration
- Create orchestration ports
- Build, deploy, and test the solution

Review Questions

- In the lab, why did you create multiple message variables based on the same schema?
- What is the purpose orchestration instance dehydration, and what causes it?

Advanced Orchestration Design

- Lesson 1: Controlling the Flow of an Orchestration
- Lesson 2: Configuring Orchestrations

Lesson 1: Controlling Orchestration Flow

In this lesson, you will learn about...

- Scope and Group Shapes
- Orchestration Flow Control Shapes
- Expression and Exception Shapes
- Nesting Orchestrations

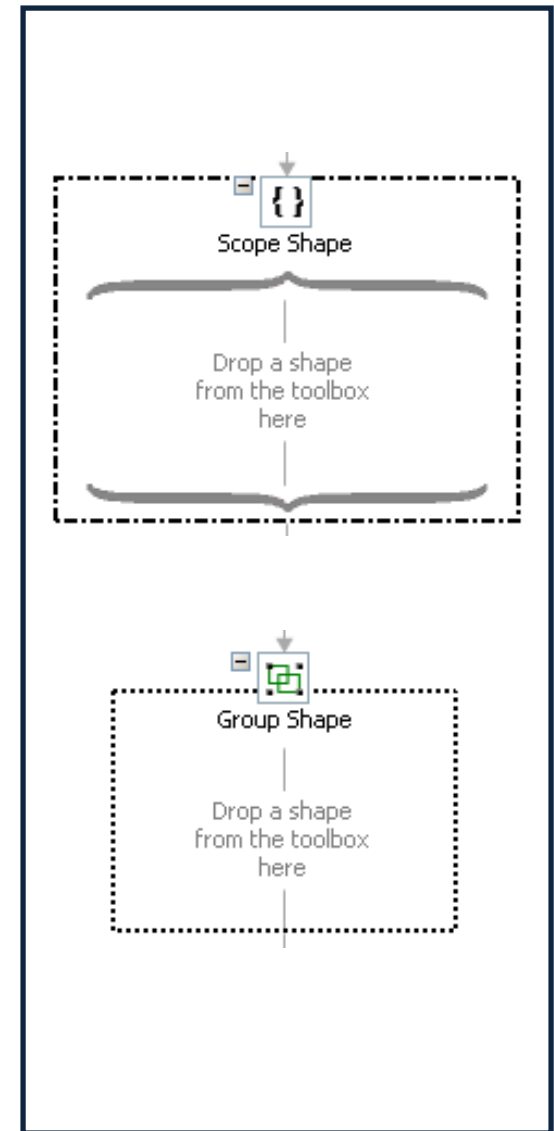
Scope and Group Shapes

Scope

- Framework for organizing actions
- Used for exception handling and transactional execution
- Contains one or more blocks
- Can have appended exception-handling blocks
- Can have appended compensation blocks if transactional

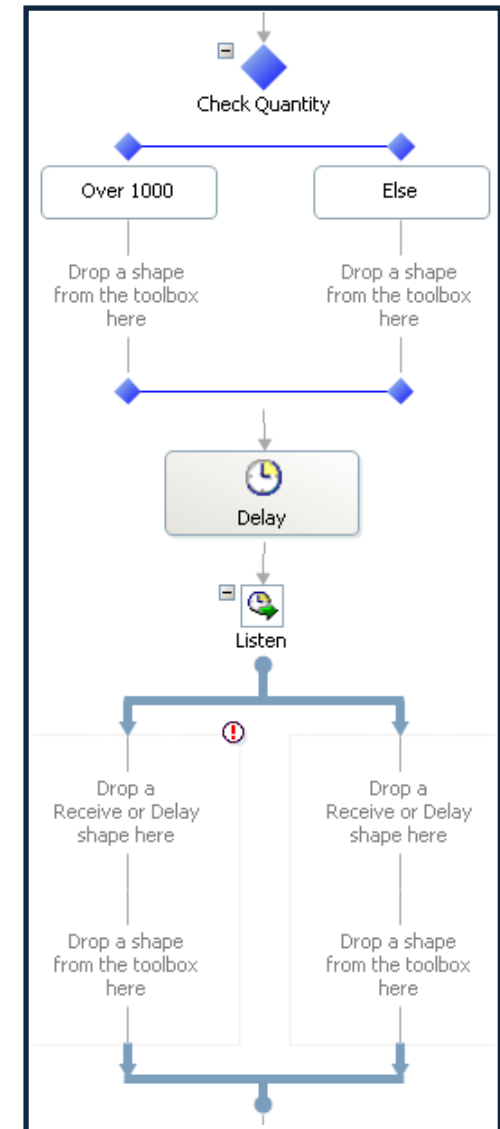
Group

- Visually group shapes
- Collapsible
- Not saved as code



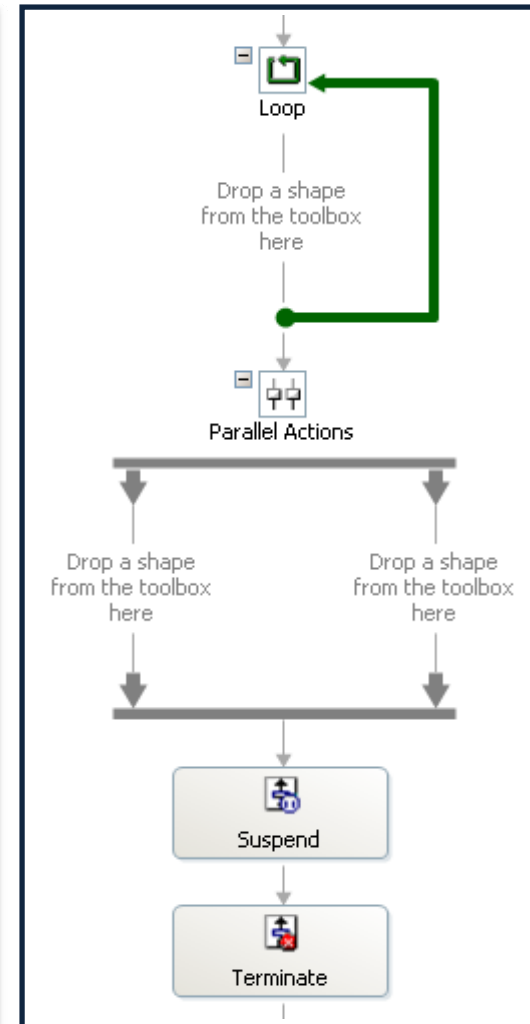
Orchestration Flow Control Shapes

Shape	Use
Decide	• To represent If/Else logic • Define rule for each branch
Delay	• To control the timing of the orchestration • Specify pause time or predefined date or time
Listen	• To wait for events to occur before proceeding • Must use Receive or Delay as the first shape • Follows the first condition met in the branch • Can use as many branches as needed



Orchestration Flow Control Shapes

Shape	Use
Loop	• To repeat actions while a condition is met • Use Expression Editor to set the condition
Parallel Actions	• To set branches for concurrent actions to run • Each action processes independently • Orchestration processing does not continue until all actions have completed
Suspend	• To suspend a running orchestration • Use Expression Editor to set the condition • Administrator intervention required to resume
Terminate	• To end a running orchestration immediately • Can specify an error message string to help diagnosis

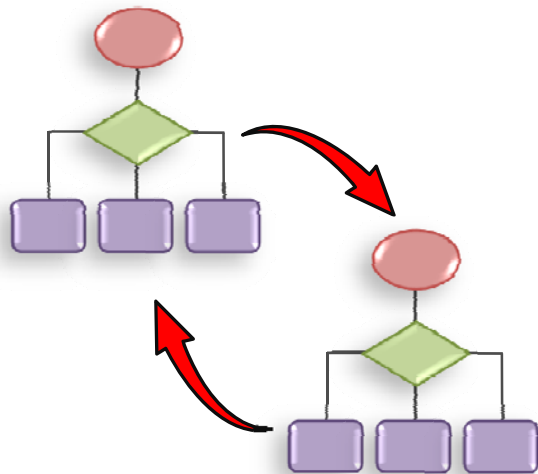


Orchestration Flow Control Shapes

Shape	Use	
Expression	• Enter any expression in an orchestration • Use Expression Editor to define expression	 Expression
Throw Exception	• Explicitly throw an exception • Exceptions are caught at present scope or at the containing scope	 Throw Exception

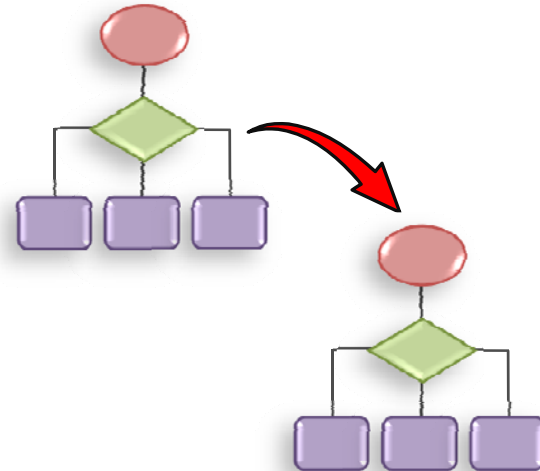
Nesting Orchestrations

Call Orchestration



- Invokes another orchestration synchronously
- Calling orchestration waits

Start Orchestration



- Invokes another orchestration asynchronously
- Calling orchestration continues processing

Lesson 2: Configuring Orchestrations

In this lesson, you will learn about...

- Distinguished Fields
- Demonstration: Configuring Orchestration Flow
- Creating Expressions
- Correlating Messages
- Demonstration: Correlating Messages
- Handling Exceptions
- Testing an Orchestration Using BizUnit

Demo: Configuring Orchestration Flow

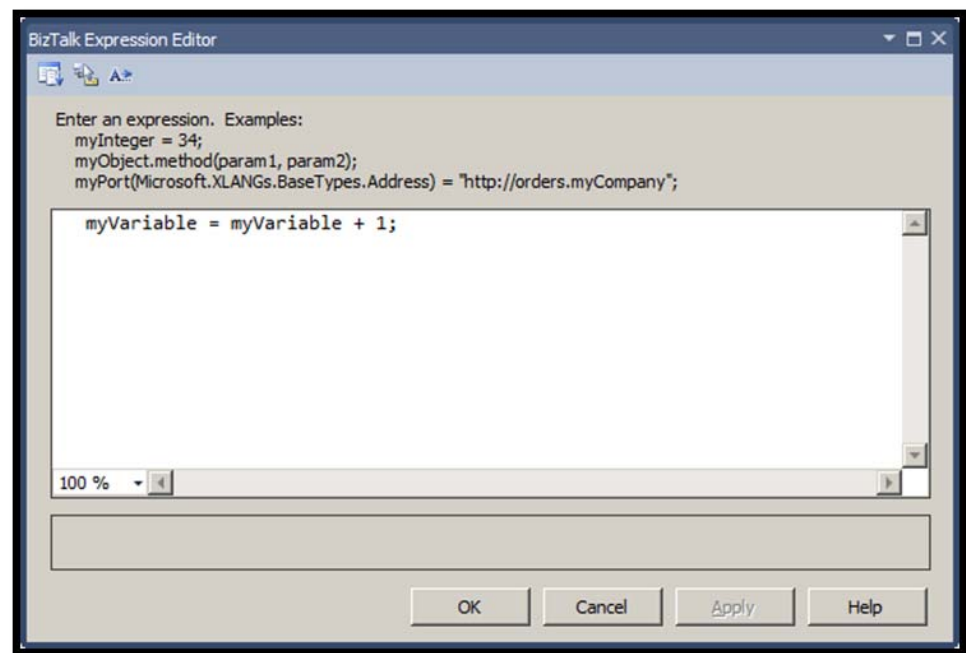
In this demonstration, you will learn about...

- Distinguishing fields
- Configuring a decision shape

Creating Expressions

Use Expressions to:

- Assign values to messages and message parts
- Initialize and manipulate the values of variables
- Assign values to dynamic ports
- Call a .NET program



Correlating Messages

Correlation

- Matches incoming messages to an existing orchestration instance

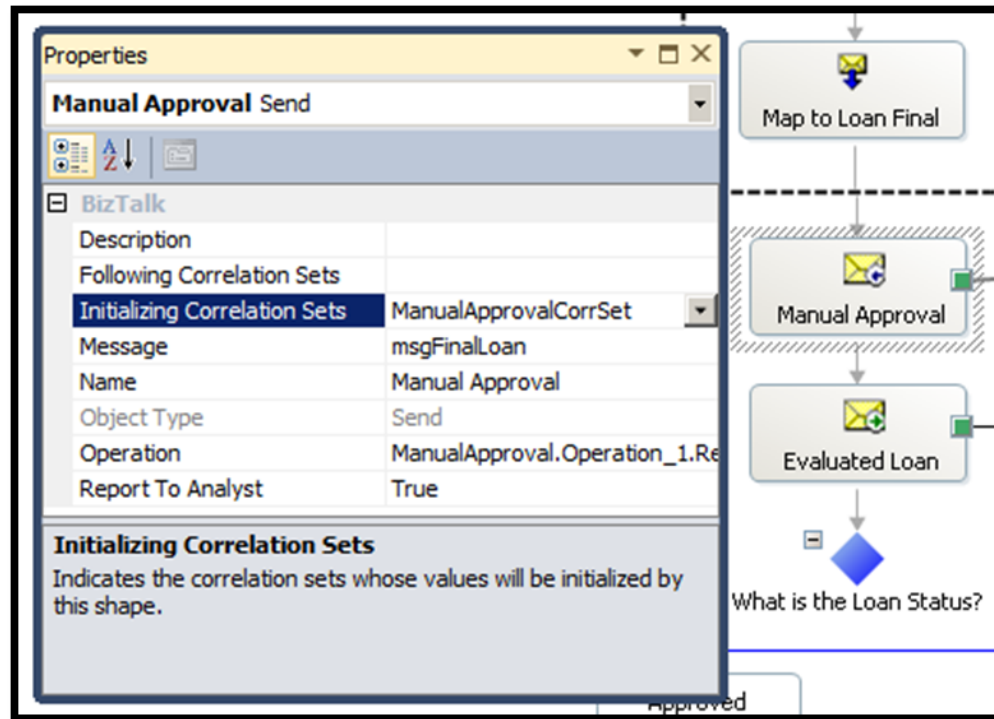
Correlation Type

- Collection of promoted properties used to route the message

Correlation Set

- Instance of Correlation Type

Orchestration shapes initialize or follow the correlation set



Demo: Correlating Messages

In this demonstration, you will learn about...

- Creating a correlation type
- Creating a correlation set
- Configuring correlation for an orchestration

Handling Exceptions

- Use to control orchestration errors
- BizTalk Server provides handling mechanisms

Exception causes

- Throw Exception shape
- Time-out occurs
- Transaction failure
- External user code failure
- System exception
- External fault message

Testing an Orchestration Using BizUnit

BizUnit

- Enables rapid development of automated tests
- Fully extensible framework designed to test distributed systems
- Uses XML file to define test cases

```
<TestCase testName="WcfRequestResponseTest">
  <TestExecution>
    <TestStep assemblyPath="Acme.BizUnit.Steps.dll" typeName="Acme.BizUnit.Steps.WcfRequestResponseStep">
      <InputMessageTypeNames>Acme.TestService.Contract.HelloWorldRequest, Acme.TestService.Contract</InputMessageTypeNames>
      <InputMessagePath>HelloWorldInput.xml</InputMessagePath>
      <InterfaceTypeNames>Acme.TestService.Contract.IHelloWorldService, Acme.TestService.Contract</InterfaceTypeNames>
      <ContractName></ContractName>
      <MethodName>HelloWorld</MethodName>

      <!-- Optional: Can specify validators here in the normal way for BizUnit -->

    </TestStep>
  </TestExecution>
</TestCase>
```

For more information or to download BizUnit, please visit: <http://www.codeplex.com/bizunit>

In this 60 minute lab, you will...

- Examine an existing project
- Distinguish fields
- Create a new orchestration
- Create orchestration ports
- Build, deploy, and test the BizTalk application

Review Questions

- How many threads are used to process the execution of a parallel actions shape?
- What is the difference between the scope and group shapes?

Creating Transactions

- Lesson 1: Introduction to Transactions
- Lesson 2: Configuring Transactions

Lesson 1: Introduction To Transactions

In this lesson, you will learn about...

- What Are Transactions?
- What Are Transactional Scopes?
- What Is Instance Persistence?
- Persistence Points in an Orchestration
- Steps for Setting Up a Transaction

What Are Transactions?

A single logical operation that may make multiple changes

- E.g. Funds transfer requires debit here and credit there

ACID properties guarantee reliability in processes

Atomicity

- All or nothing

Consistency

- Transaction will leave the system in a consistent state

Isolation

- Data is locked while transaction is in process

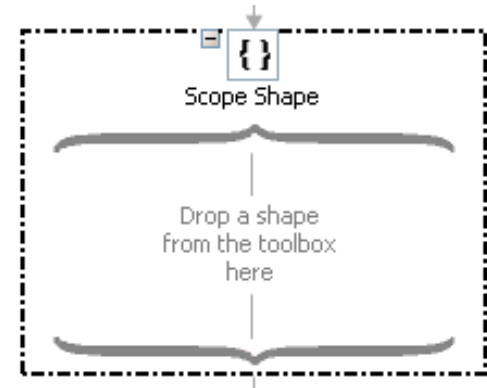
Durability

- Once transaction is committed the state is preserved

What Are Transactional Scopes?

Scope Shape

- Container for one or more actions
- Foundation for transactions
- Enables compensation of transactions
- Enables exception handling



Transaction Types

- Atomic: provide rollback capabilities
- Long-running: save state at persistence points

What Is Instance Persistence?

State Persistence

- Orchestration engine manages the persistence and restoration of orchestration service instances
- Critical when resources required by service instances are greater than resources available

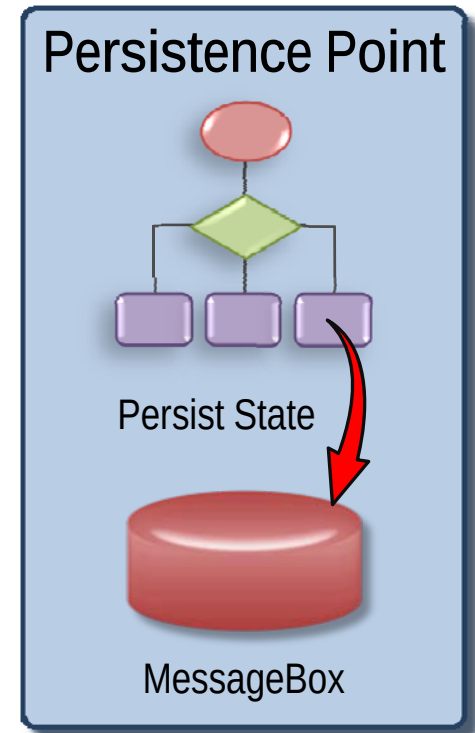
Persistence Functions

- Dehydration and rehydration
- Instance recovery
- Transaction compensation
- Controlled system shutdown

Persistence Points in an Orchestration

Persistence Points

- Message sent within a non-atomic scope
- End of transactional scope
- Starting a new orchestration instance
- Start Orchestration shape
- Debug break point
- Dehydration
- System shutdown



Steps for Setting Up a Transaction

1. Create a scope
2. Identify the type of transaction required
3. Identify potential errors
4. Add appropriate exception handlers
5. Determine if the transaction requires compensation
6. Define compensation code

Lesson 2: Configuring Transactions

In this lesson, you will learn about...

- Defining an Orchestration as Transactional
- Creating a Long-Running Transaction
- Creating an Atomic Transaction
- Creating Modular Business Processes
- Demonstration: Creating Transactions
- Adding Compensation Code
- Demonstration: Adding Compensation Code

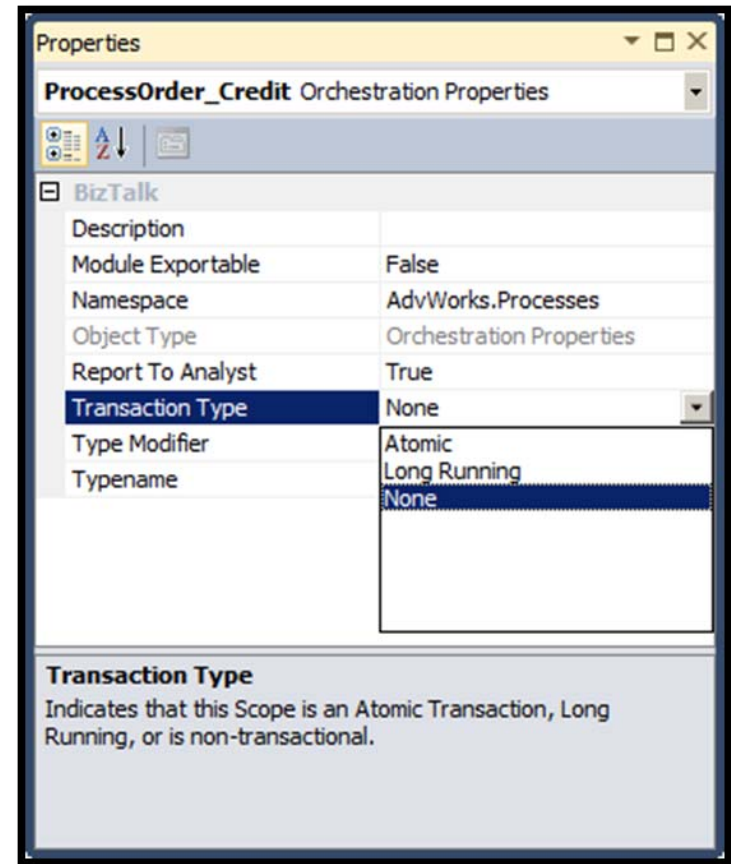
Defining an Orchestration as Transactional

Transactional Orchestration

- Use to define an entire orchestration as transactional

Transaction Nesting Rules

- Long-running transactions can contain other transactions
- Non-transactional processes cannot contain transactions
- Atomic transactions cannot contain transactions



Creating an Atomic Transaction

Atomic transactions:

- Mimic the ACID properties
- State changes are isolated until transaction is committed
- Prevents orchestration instance persistence
- Can set entire orchestration to atomic

Scenarios for use:

- Minimizing persistence points
- Using a class not marked as serializable
- Calling a COM+ component that's participating in the orchestration transaction

Creating a Long-Running Transaction

Long-running transactions:

- Do not mimic ACID properties
- Do not isolate state changes
- Do not prevent persistence

Scenarios for use:

- Container for other transaction
- Enable Compensation

Demo: Creating Transactions

In this demonstration, you will learn about...

- Setting an orchestration as transactional
- Adding a transactional scope to an orchestration

Compensation code

- Can be added to any transactional scope
- Used to “undo” a committed transaction
- Can only be called during exception handling
- Will only execute if the transaction committed
- Compensating a parent will automatically invoke compensation for children

Demo: Adding Compensation Code

In this demonstration, you will learn about...

- Adding an exception handling block to a scope
- Adding a compensation block to a transaction

Lab: Creating Transactional Business Processes

In this 45 minute lab, you will...

- Add exception handling to an orchestration
- Add compensation to an orchestration
- Call compensation
- Test the application

Review Questions

- For what reasons would you use a long-running transaction?
- How can transactions be nested?

Deploying & Managing Applications

- Lesson 1: Introduction to BizTalk Applications
- Lesson 2: Deploying an Application

Lesson 1: Introduction to BizTalk Applications

In this lesson, you will learn about...

- What Is a BizTalk Application?
- What is the Default Application?
- Configuring the Application for a BizTalk Project
- Demonstration: Deploying a Project to an Application
- Managing Project Versioning

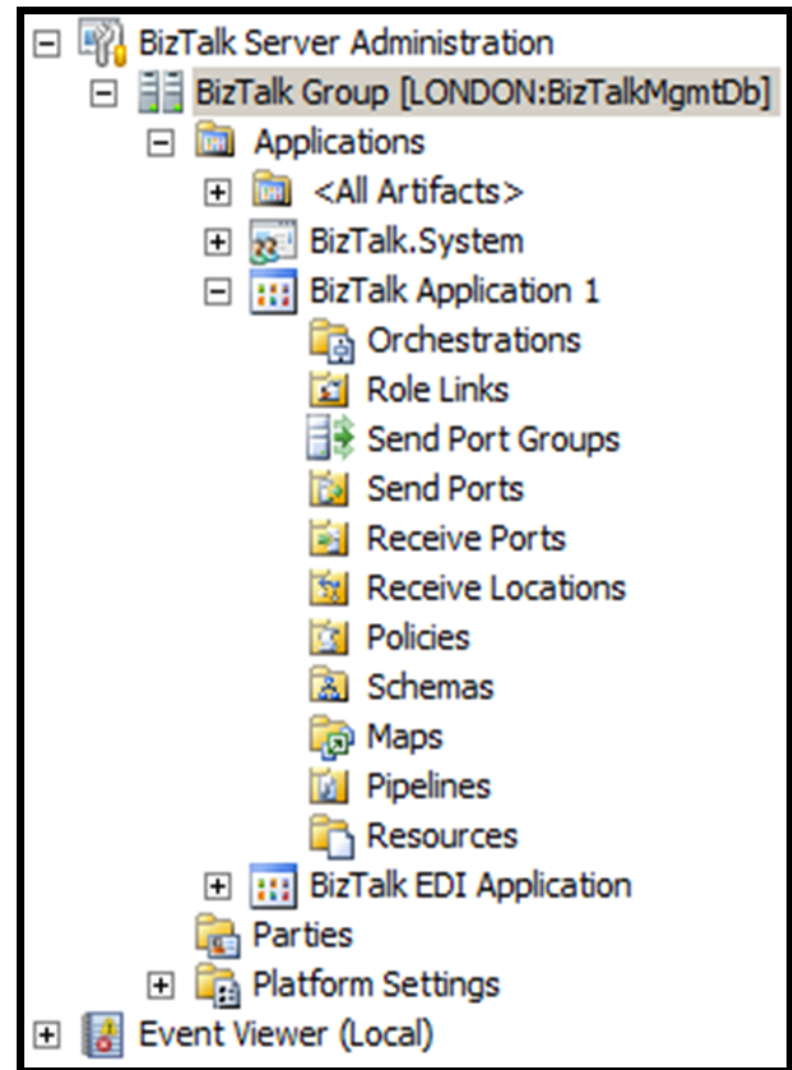
What Is a BizTalk Application?

BizTalk Artifacts

- Schemas
- Maps
- Pipelines
- Orchestrations

Messaging Components

- Receive ports
- Receive locations
- Send ports
- Adapters



What Is the Default Application?

Default Application

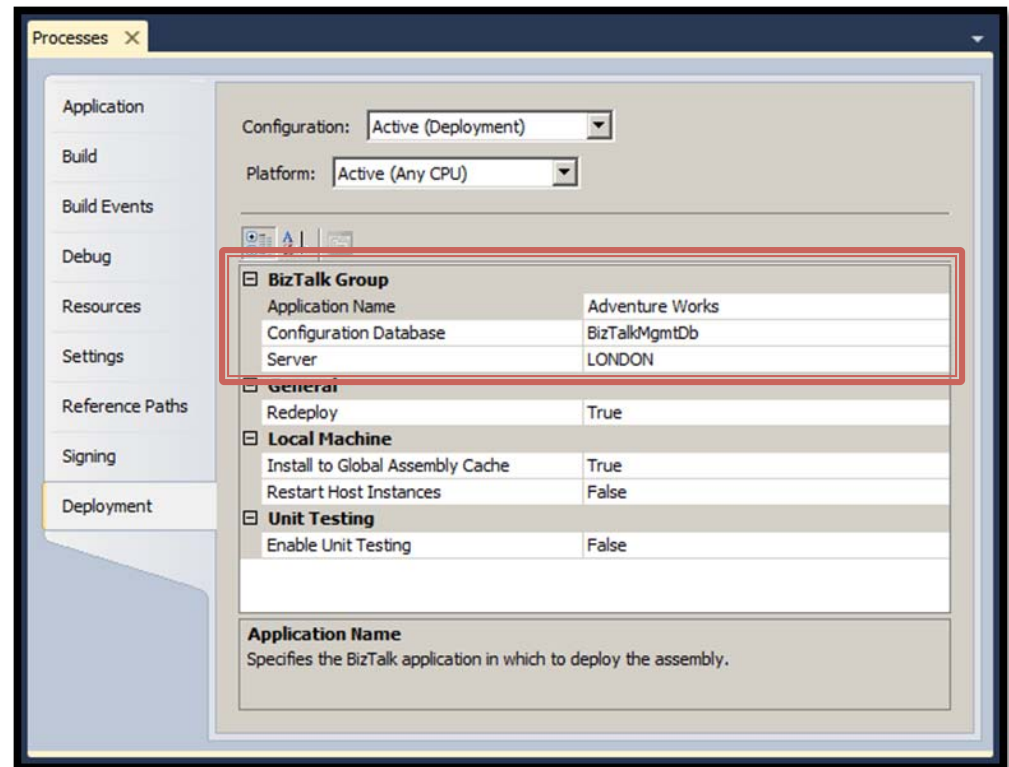
- Automatically created and used as default when BizTalk is installed
- Can be renamed
- Any application can be set as the default application

Adding Artifacts to the Default Application

- From Visual Studio when no application is specified
- Use the BTSTask command without specifying an application name

Configuring the Application for a Project

1. Configure strong name for assembly
2. Set application deployment properties
3. Build and deploy the application



Demo: Deploying a Project to an Application

In this demonstration, you will learn about...

- Configuring the application deployment property
- Viewing the application in the BizTalk Server Administration Console

Managing Project Versioning

Guidelines

- Group objects based on type
- Separate objects that are subject to change
- Group objects that must be versioned together
- For schemas, change target namespace when using multiple versions

Considerations

- Verify existing deployed version numbers
- Confirm the version used in a binding file
- Version the application rather than a single assembly?

Lesson 2: Deploying an Application

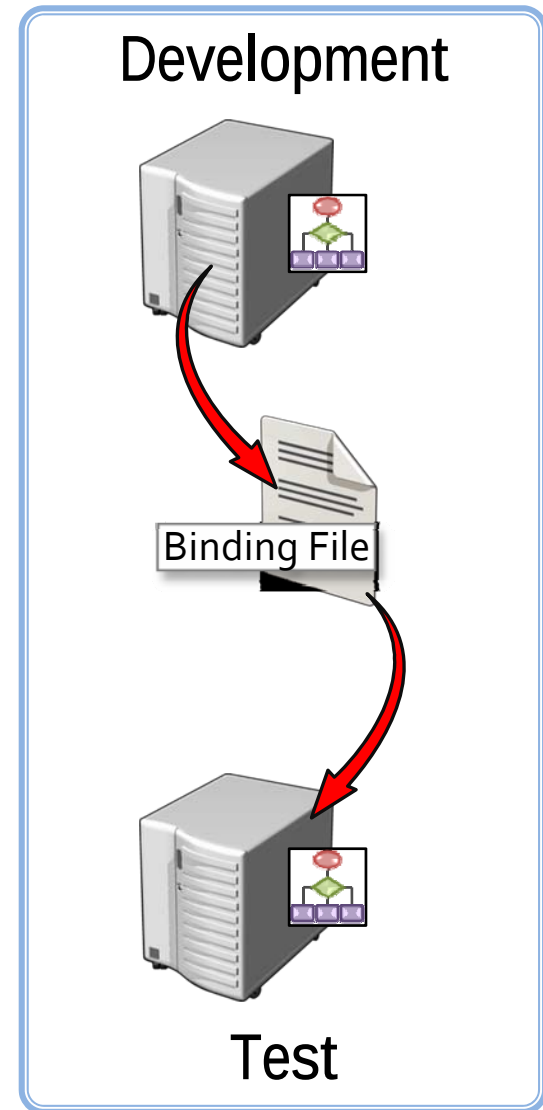
In this lesson, you will learn about...

- Deploying with MSI Packages
- Demonstration: Exporting an Application as an MSI Package
- Exporting and Importing a Binding File
- Deploying Applications from a Command Prompt
- Demonstration: Managing an Application

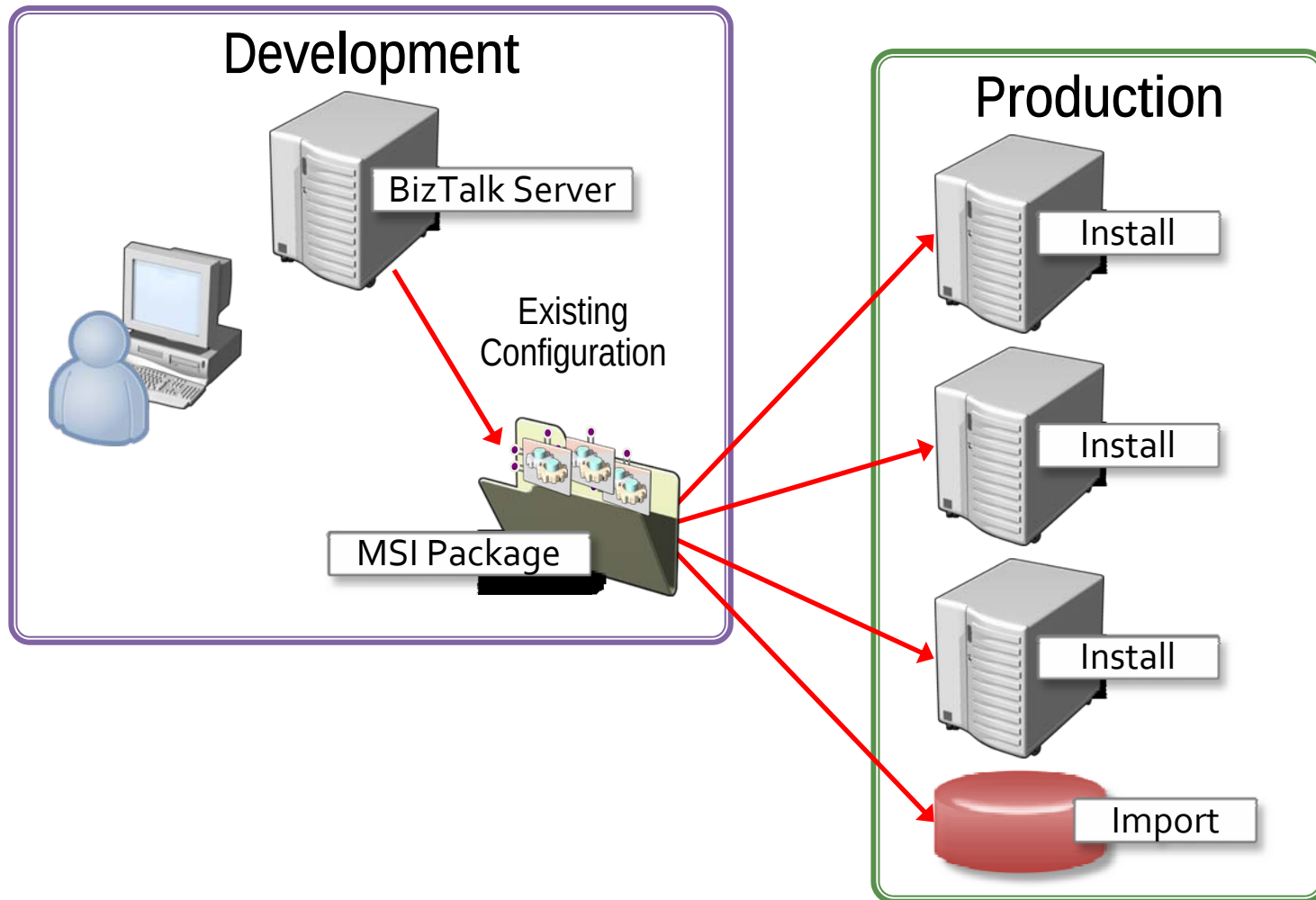
Exporting and Importing a Binding File

Binding Files

- Useful for moving configuration information from one physical environment to another
- Contains port configuration settings
- Contains orchestration configuration including physical/logical port relationships and host
- Can contain party and EDI settings



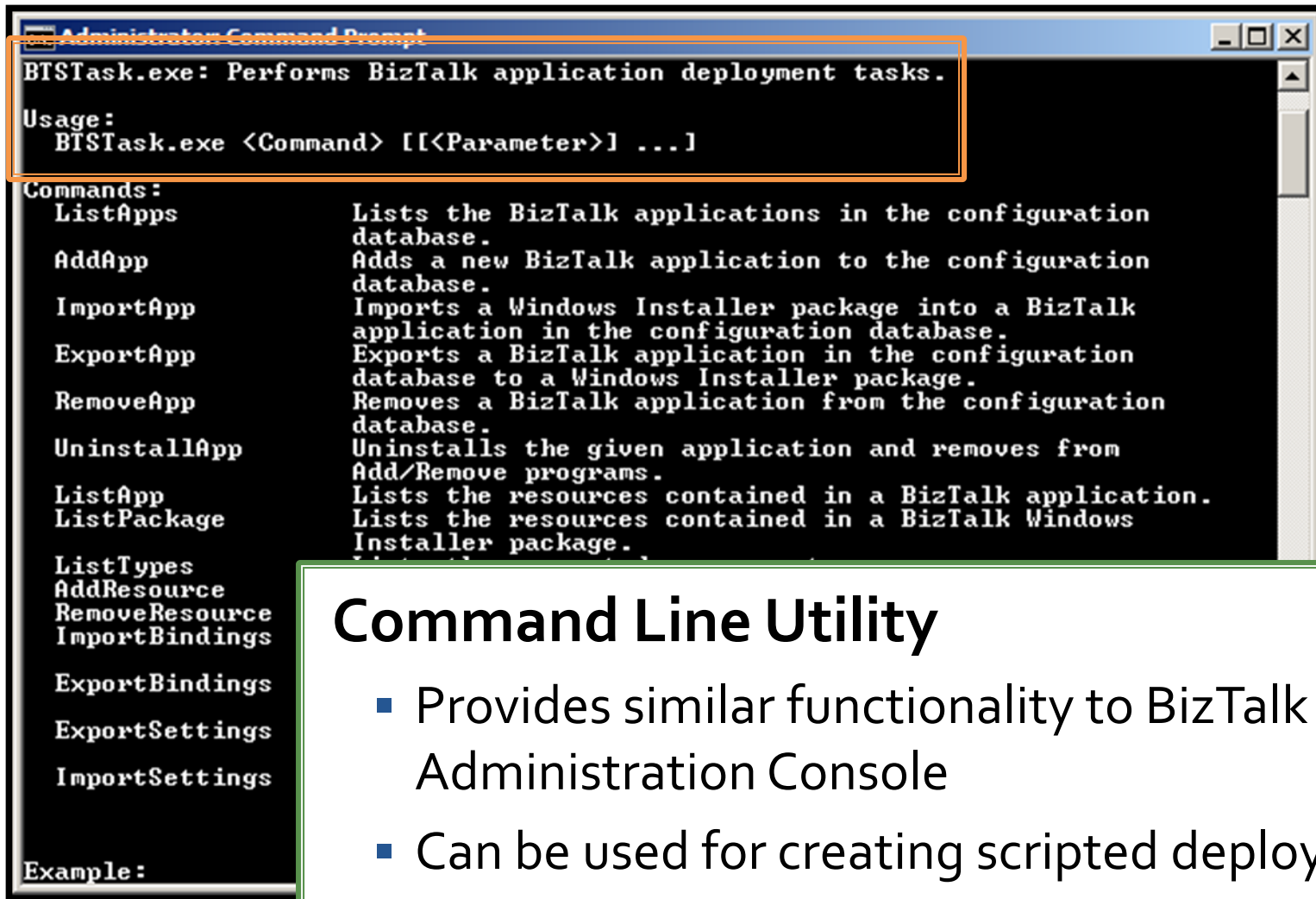
Deploying with MSI Packages



Additional Resources

- Adding them to BizTalk applications enables packaging in MSI
- Resources can include:
 - BizTalk assemblies
 - Binding files
 - .NET assemblies
 - BAM activities
 - Pre- and Post-processing scripts
 - Business Rules Policies
 - IIS Virtual directories
 - COM components

Scripting Deployments



```
Administrator Command Prompt
BTSTask.exe: Performs BizTalk application deployment tasks.
Usage:
  BTSTask.exe <Command> [[<Parameter>] ...]

Commands:
  ListApps          Lists the BizTalk applications in the configuration
                    database.
  AddApp            Adds a new BizTalk application to the configuration
                    database.
  ImportApp         Imports a Windows Installer package into a BizTalk
                    application in the configuration database.
  ExportApp         Exports a BizTalk application in the configuration
                    database to a Windows Installer package.
  RemoveApp         Removes a BizTalk application from the configuration
                    database.
  UninstallApp      Uninstalls the given application and removes from
                    Add/Remove programs.
  ListApp           Lists the resources contained in a BizTalk application.
  ListPackage       Lists the resources contained in a BizTalk Windows
                    Installer package.
  ListTypes
  AddResource
  RemoveResource
  ImportBindings
  ExportBindings
  ExportSettings
  ImportSettings

Example:
```

Command Line Utility

- Provides similar functionality to BizTalk Administration Console
- Can be used for creating scripted deployments

In this demonstration, you will see how to...

- Use the BizTalk Server Administration Console to export a BizTalk Application to an MSI package
- Import an MSI package using the Administration Console
- Import and export binding files
- Deploy using BTSTask

Lab: Deploying & Managing Applications

In this 30 minute lab, you will...

- Configure the Application Deployment property
- Manage ports using binding files
- Manage applications using MSI packages
- Move resources and ports between applications
- Manage applications using BTSTask

Review Questions

- What is the purpose of binding files?
- What tool can be used to import and export complete applications?

Integrating with Services

- Lesson 1: Consuming a Service
- Lesson 2: Creating Services Using BizTalk

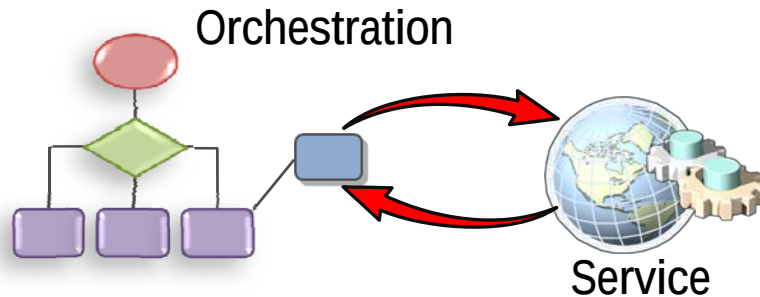
Lesson 1: Web Service Integration

In this lesson, you will learn about...

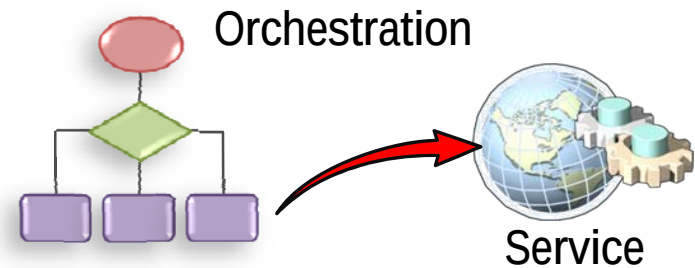
- Service Integration Scenarios for BizTalk
- The BizTalk WCF Service Wizard
- Demonstration: Consuming a Service

Service Integration Scenarios for BizTalk

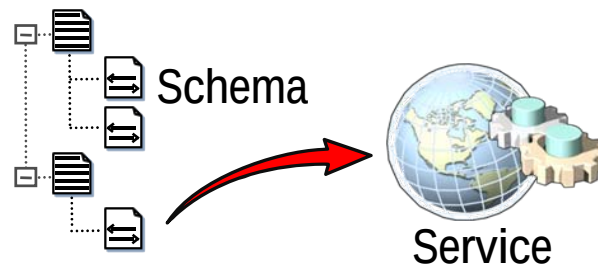
Consuming a Service



Publishing an Orchestration as a Service



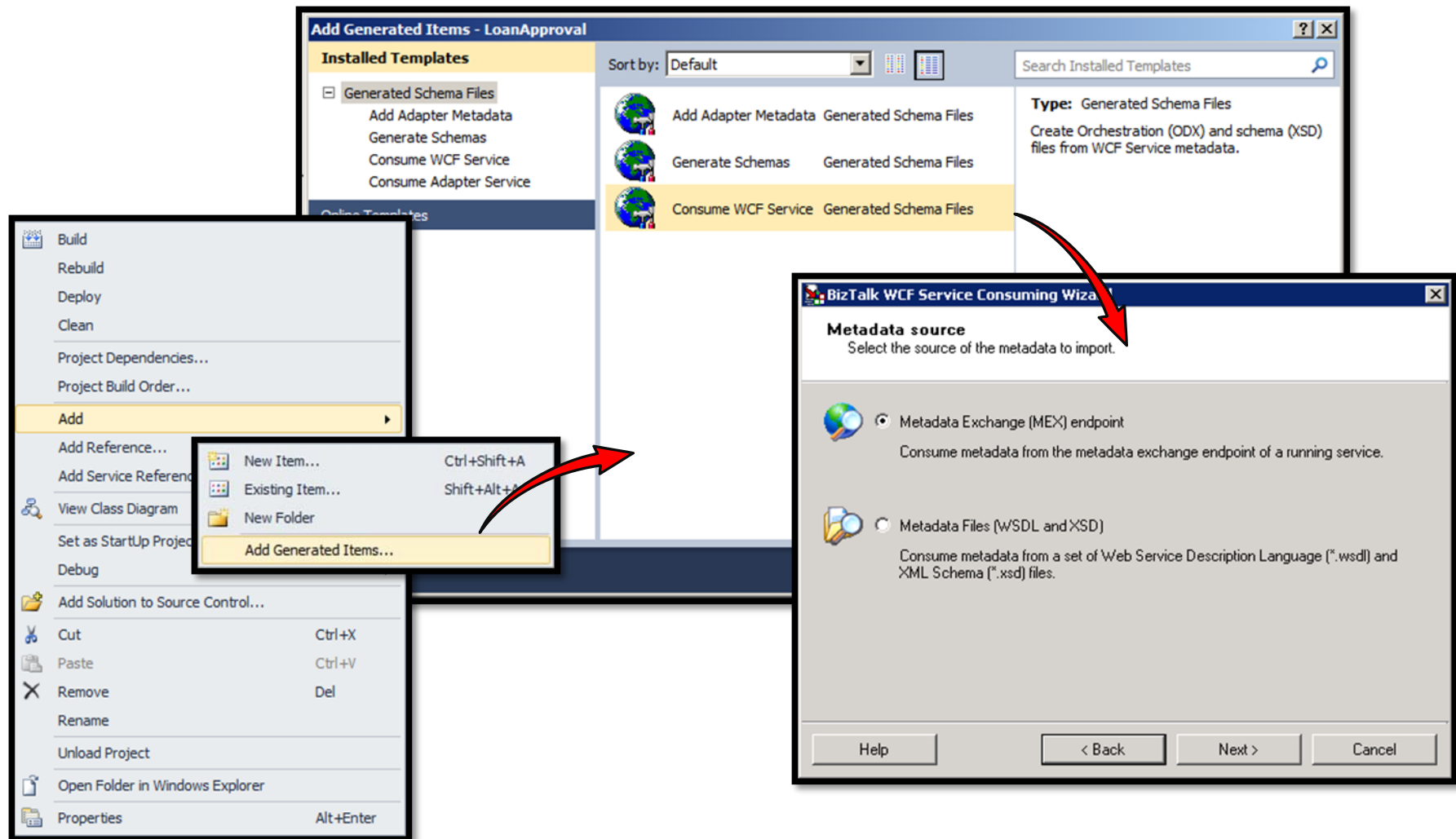
Publishing a Schema as a Service



Creates proxy classes based on service WSDL:

- Adds schemas to project
- Creates “empty” orchestration
- Creates binding file
- Use on any service with WSDL

The BizTalk WCF Service Wizard



Demo: Consuming a Service

In this demonstration, you will learn about...

- Adding a service reference
- Creating an orchestration to call a service

Lesson 2: Creating Services Using BizTalk

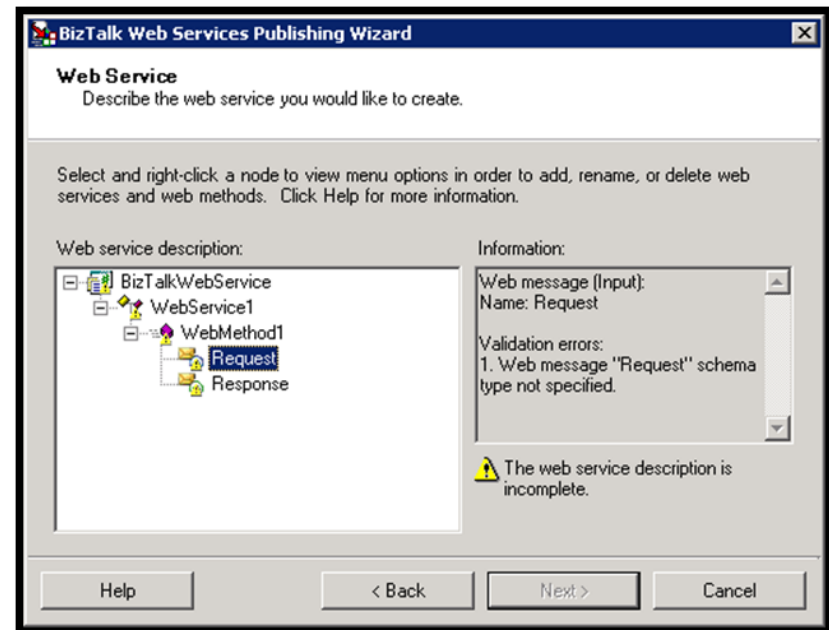
In this lesson, you will learn about...

- The BizTalk Web Services Publishing Wizard
- The BizTalk WCF Services Publishing Wizard
- Configuring an Orchestration for Publishing
- The Publishing Wizard MEX Only Option

The BizTalk Web Services Publishing Wizard

Publish Schemas or Orchestrations:

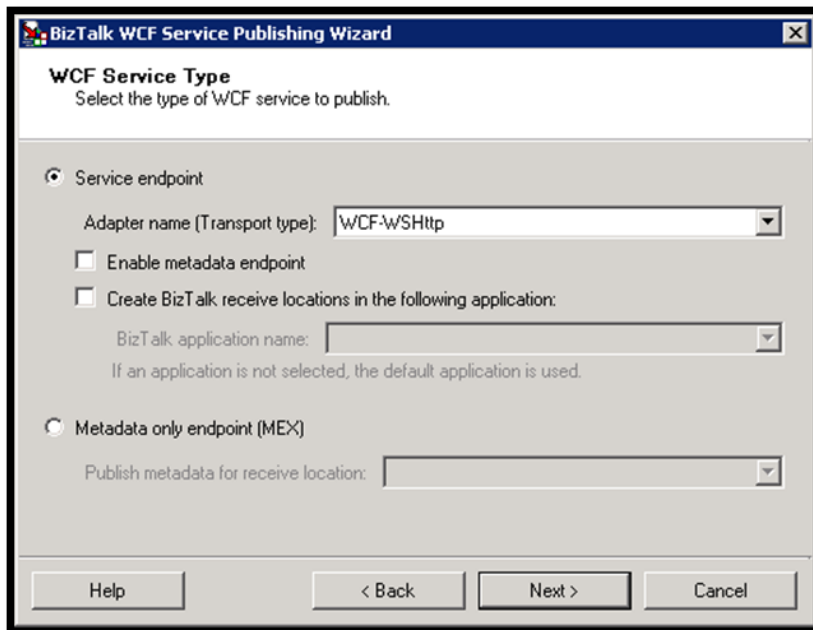
- Creates ASMX service
- Enables web-based receipt of messages
- Best practice is to publish schemas



The BizTalk WCF Service Publishing Wizard

Creates WCF-based Services:

- Creates SVC instead of ASMX
- Publish either schemas or orchestrations
- Choose between *Service endpoint* or *MEX endpoint*



Orchestration Port Types

- By default port types have Internal Type Modifier
- A published orchestration must have at least one logical port type configured with a Public Type Modifier

What is the MEX Only Option Used For?

- In-process WCF adapters do not expose metadata endpoints or WSDL export
 - WCF-Custom adapter is an exception
 - Only loosely-typed
- The WCF Publishing Wizard generates MEX only endpoints based on orchestrations and schemas
- SvcUtil.exe can also be used to generate WSDL and proxy class for communication
- Useful with message-based routing solutions

In this demonstration, you will learn about...

- Publishing an orchestration as a service

Lab: Integrating with Services

In this 60 minute lab, you will:

- Consume a WCF Service
- Publish an Orchestration as a WCF Service

Review Questions

- How is publishing a schema as a service different from publishing an orchestration as a service?
- How is it better?

Integrating with Business Rule Policies

- Lesson 1: Introduction to Business Rules
- Lesson 2: Integrating Business Rules

Lesson 1: Introduction to Business Rules

In this lesson, you will learn about...

- Three Types of Knowledge
- What Are Business Rules?
- Features of Business Rules
- Rule Engine Scenarios
- Why use the Business Rule Engine?
- Rule Engine Scenarios
- BizTalk Business Rules Terminology
- How Rules and Facts Work
- Business Rules Orchestration Scenarios
- Identifying Business Rule Personas

Three Types of Knowledge

Factual Knowledge

- Facts or data easily stored in a database
 - Customer info, order details, etc.

Procedural Knowledge

- Sequence of steps executed in order (If/Then)
 - Typical .NET applications are procedural

Logical Knowledge

- Relationships between complex entities
 - Price and market conditions
 - Symptoms leading to a diagnosis
- Difficult to encode using typical database and procedural processes
- Rules-based decision tools are needed

What Are Business Rules?

Rule Usage

- Much more than another If/Then construct
- Enable analysis and classification of complex sets of data
- Provide a mechanism for easy redefinition

Rule Example

Truths
• Frogs are green
• Frogs croak
• Toads croak
• Toads are orange
• Orioles are orange
• Orioles sing

Rules
• If an animal croaks and is green it is a frog
• If an animal croaks and is orange it is a toad
• If an animal sings and is orange it is a oriole

```
<animal>  
  <traits sound="croak"  
color="orange" type="toad"/>  
</animal>
```

Features of Business Rules

Feature	Description
Forward Chaining Inference	• Begins with the available data (facts) and uses inference rules to extract more data until an optimal goal is reached • Searches rules until one is found where the <i>If</i> clause is true • Infers a consequence which <i>may add</i> new data • Lather, rinse, repeat
Backward Chaining Inference	• Must know the value of a piece of data and search for rules whose conclusions mention that data • AKA – Goal-directed or hypothesis driven
Rule Processing	• Loads facts into memory and uses only those facts • Facts may be changed in memory for future rule evaluation (Update) or for all evaluations (Assert)
RETE Classes	• In-memory network cache of objects that holds the execution path of expressed rule • Commonly accepted algorithm • Sacrifices memory for speed (several orders of magnitude faster)

Why use the Business Rules Engine?

Business Rule Engine:

- Enables policy changes in real time
- Enables policy changes without recoding or rebuilding the orchestration

Use business rules to:

- Trigger notifications
- Automate approvals
- Reroute documents

Business Event	Business Decisions	Rules
Receive Purchase Order	Is customer known?	If customer ID exists, then customer is known
	Is product known?	If product ID exists, then product is known
	Is product available and in inventory?	If quantity on hand $\geq$ amount ordered

Scenarios for Using the Rules Engine

- Applications involving significant decision making
- Complex or frequently changing rules
- Rules that need to be shared across multiple applications and/or processes
- Volatile industry sector with frequent changes or extensive regulations

Scenarios for Maintaining Rules in Code

- Static, computational rules
- Simple but numerous rules
- No chance of rules being modified in the future

BizTalk Business Rules Terminology

Term	Description
Business Rule	• If/Then comparisons of Facts that, if evaluated to be true, cause action(s) to be performed
Fact	• A value that is evaluated against other value(s) • Instance Facts are used for one rule call • Long Term Facts are cached for reuse (speed) • Anything can be the source of facts
Action	• Steps to be performed when a rule is true • Facts may be changed, added or removed as part of an action
Policy	• Versioned set of one or more business rules • Called by an orchestration or by code
Business Rule Store	• Database or file-based store where policies and vocabularies are kept
Vocabulary	• Friendly set of custom operations that can be derived from actions or facts • Used as an alias for more technical terms

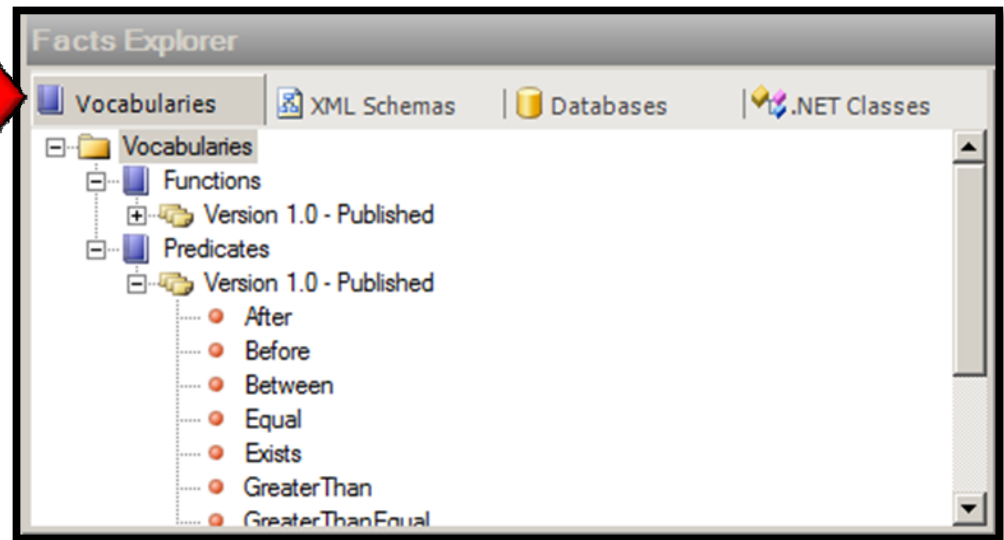
How Rules and Facts Work

Policy or Rule Set Rules

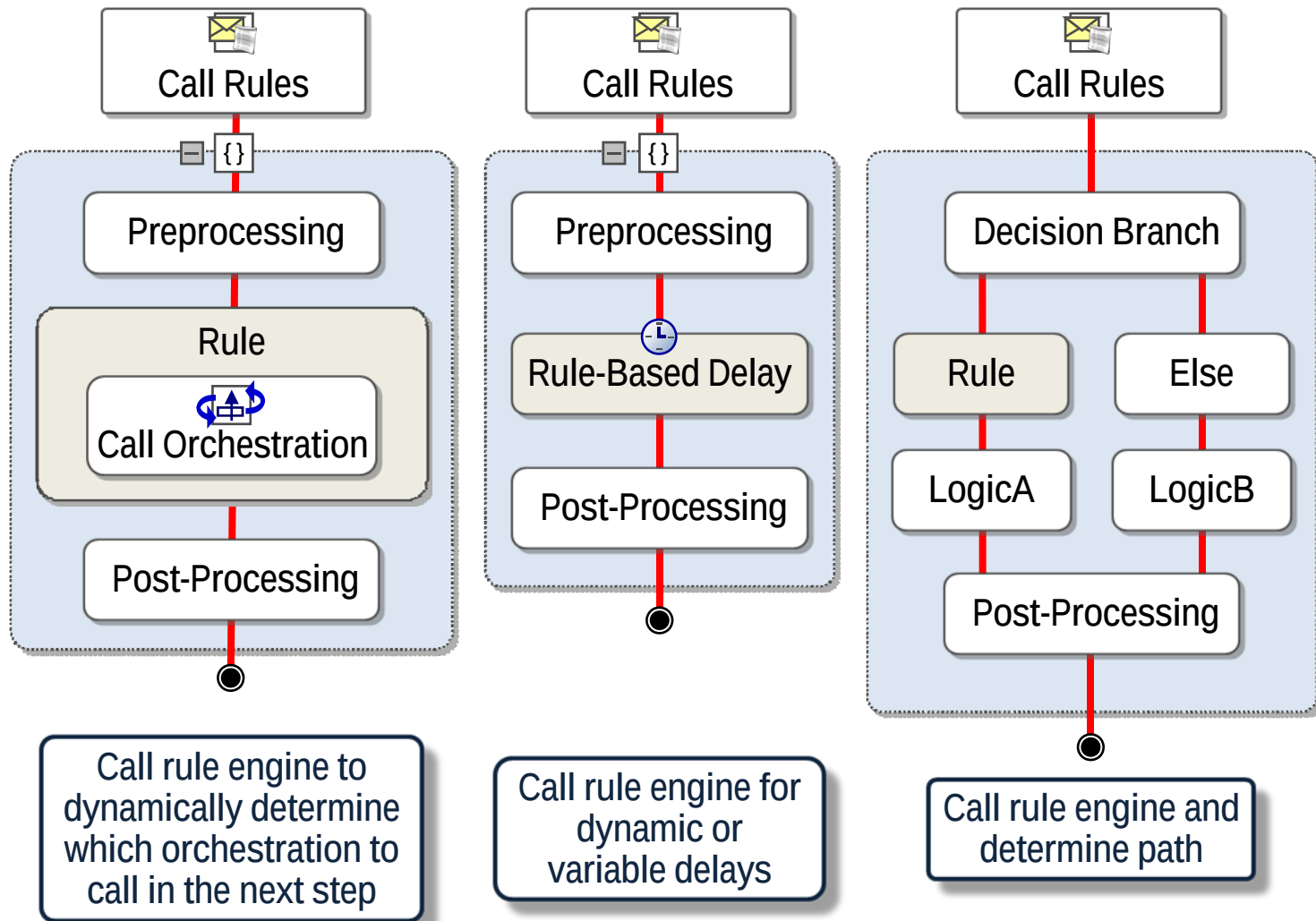
- Conditions: True/false expression with predicates
- Actions: Initiated when rule condition is met
- Facts: Data upon which rules operate

Facts derived from:

- XML schemas
- SQL databases
- .NET classes
- Predefined & custom vocabularies



Business Rules Orchestration Scenarios



Identifying Business Rule Personas

Developers

- Integrate policies within an orchestration
- Define policies and vocabularies

Information Workers

- Use business vocabularies to design and test policies

Administrators

- Secure, deploy, and migrate policies
- Track policy execution

Lesson 2: Integrating Business Rules

In this lesson, you will learn about...

- Steps for Integrating Business Rules
- Composing Business Rules
- Demo: Using the Business Rule Composer
- Methods for Managing Policies
- Managing Business Rules
- Integrating Business Rules into an Orchestration
- Demo: Integrating Business Rules into an Orchestration
- Tracking Business Rule Policy Execution

Steps for Integrating Business Rules

1. Identify the business logic to be represented
2. Identify data sources for rule elements
3. Create rules from vocabulary definitions
4. Test and debug with sample facts
5. Publish and deploy the policy
6. Call the policy from within the orchestration

Composing Business Rules

The screenshot shows the Microsoft Business Rule Composer interface. The **Policy Explorer** on the left lists policies, including **AdventureWorksOrderLimit** (Version 1.0 not saved) and **CheckOrderLimit**. The **Facts Explorer** below it shows vocabularies like **ESB.Transport Types** and **OrderValidity**, with a specific entry for **Version 1.0 - Published RequestStatus** highlighted. The main workspace displays a rule named **AdventureWorksOrderLimit - Version 1.0 - CheckOrderLimit**. The rule's **IF** condition is **Maximum Order Amount is greater than OrderAllowed**. The **THEN** section is currently empty. Red lines connect the labels on the left to their corresponding components in the interface.

Policies

Vocabularies

Definitions

- Create versions of vocabularies
- Create vocabulary definitions
- Compose policies
- Publish vocabularies and policies

Rule Composer

Demo: Using the Business Rule Composer

In this demonstration, you will learn about...

- Examining an existing business rule policy
- Testing a business rule policy

Business Rule Composer

- Tool used to create, publish, and deploy policies and vocabularies
- Cannot export policies or vocabularies

BizTalk Administration Console

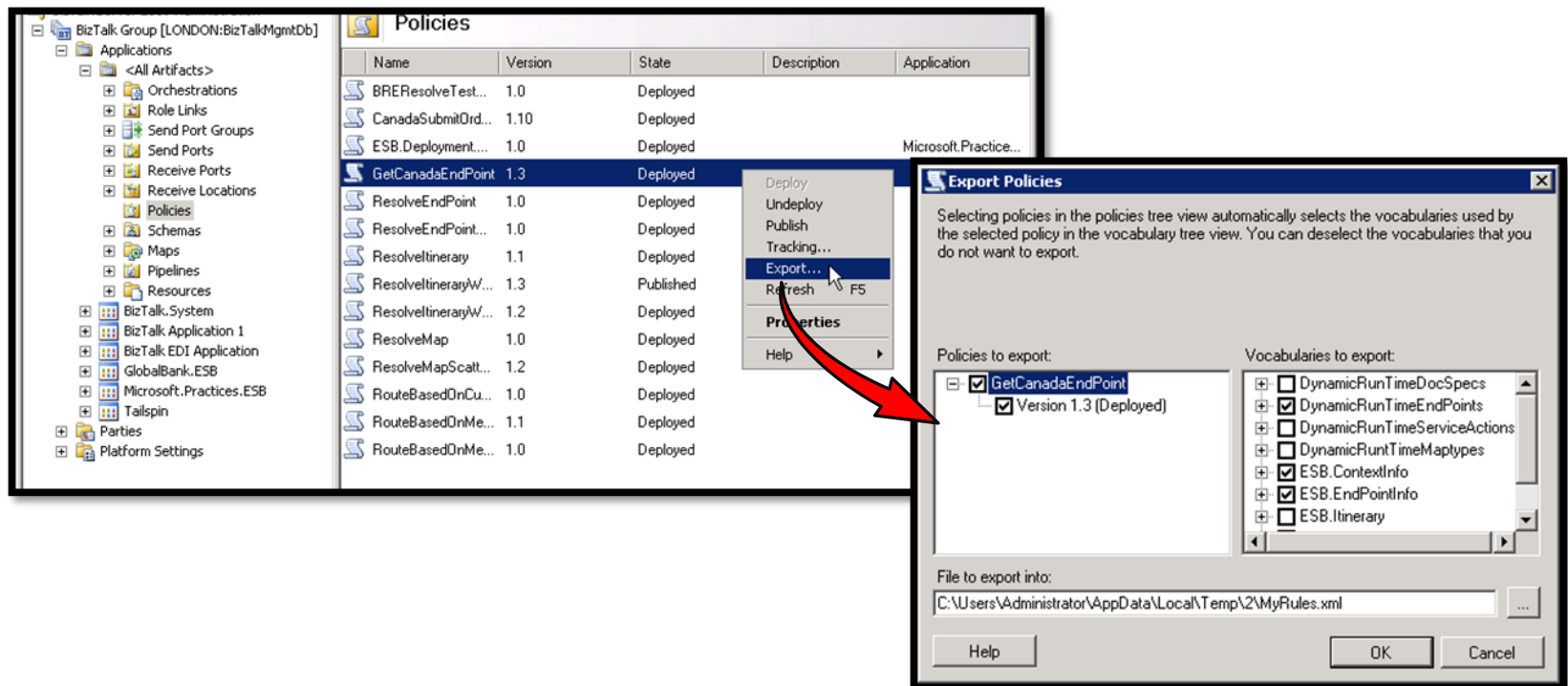
- Import and export policies and vocabularies
- Publish, deploy, and un-deploy policies
- Delete policies and vocabularies ☹

Business Rule Deployment Wizard

- Import and export policies and vocabularies
- Deploy and un-deploy policies

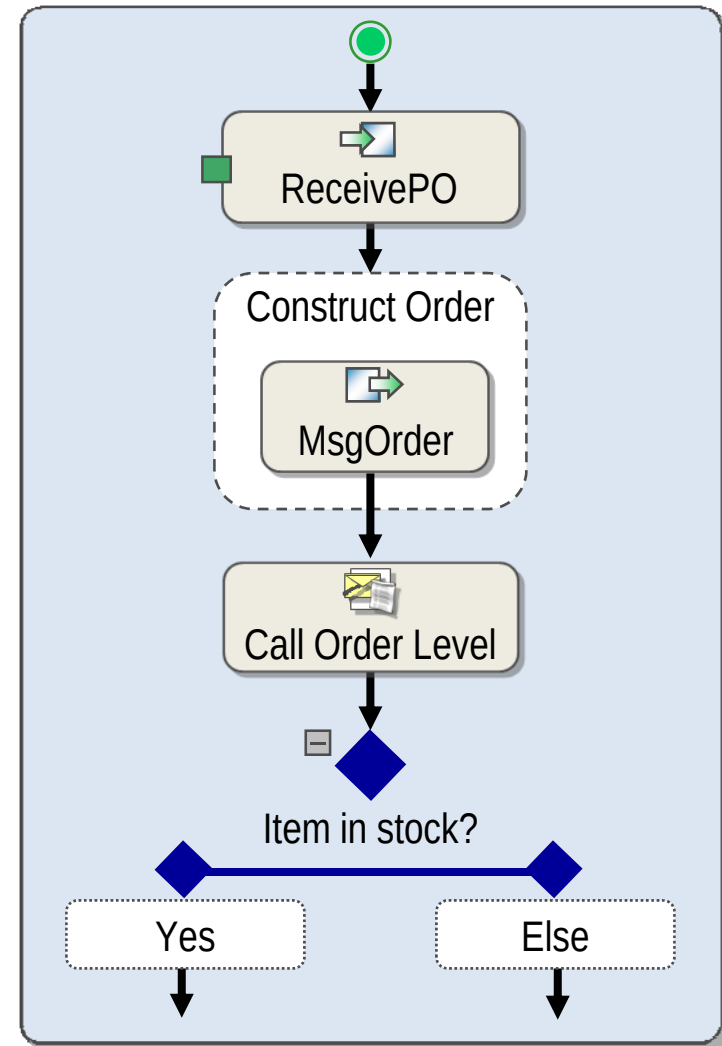
Managing Business Rules

- Export policies or vocabularies
- Import policies and vocabularies
- Deploy policies and publish vocabularies
- Undeploy policies and remove vocabularies



Integrating Policies into an Orchestration

1. Add a Call Rule shape to the orchestration
2. Create Messages and variables to pass to policy
3. React to the changes made in the policy call



Demo: Calling a Policy from an Orchestration

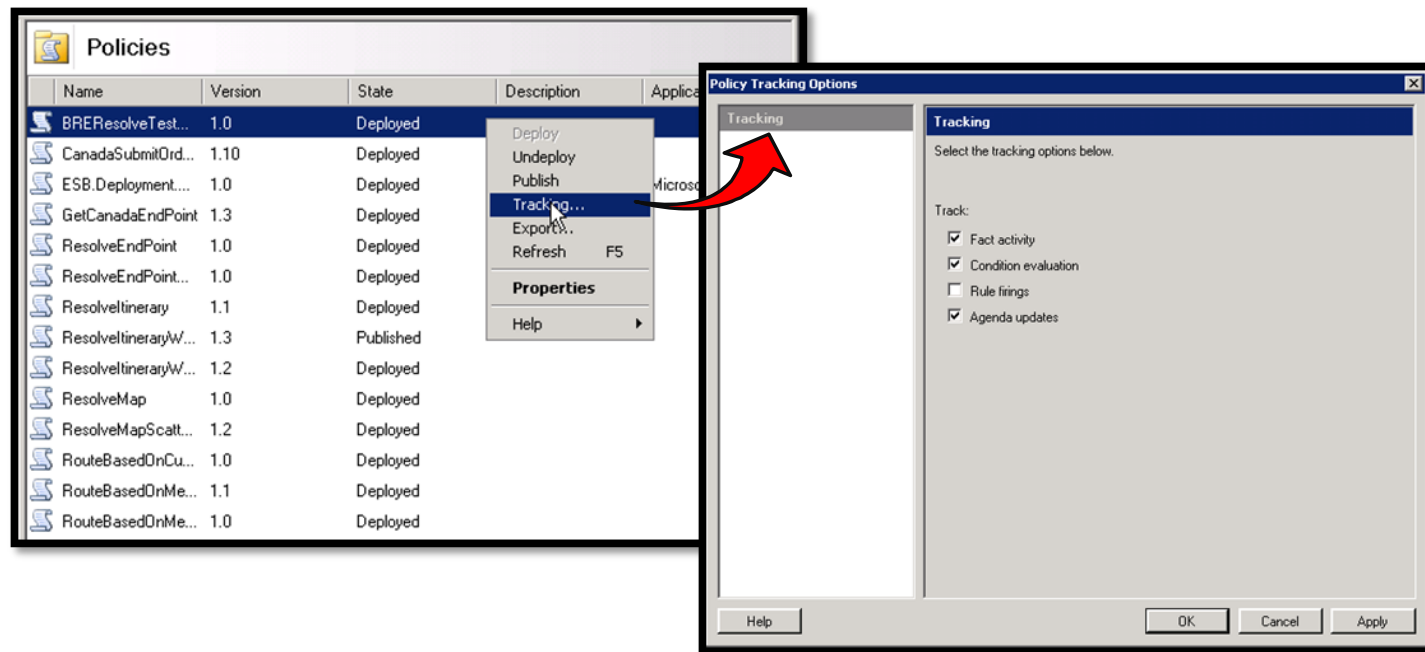
In this demonstration, you see how to...

- Integrate a business rule policy evaluation into a BizTalk orchestration

Tracking Business Rules Policy Execution

Tracking Business Rules

- Use the BizTalk Server Administration Console
- Monitor rule activities
- Track the progress of an orchestration that calls rules



Lab: Integrating Business Rules

In this 60 minute lab, you will....

- Create a Business Rule Engine Vocabulary
- Compose a Business Rule Policy
- Integrate a Business Rule Policy into an Orchestration

Review Questions

- What advantages does the business rules engine provide?

Enabling Business Activity Monitoring

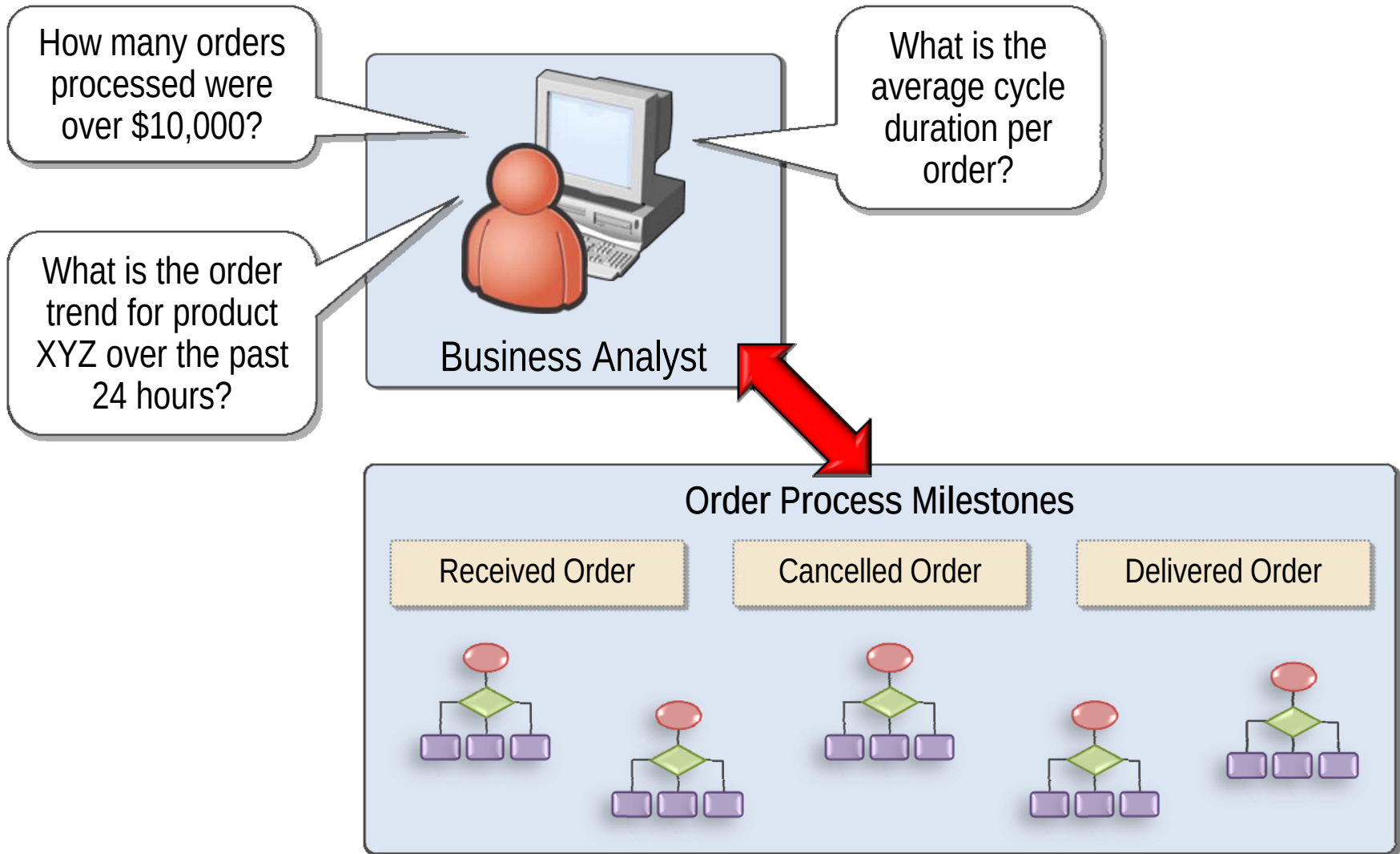
- Lesson 1: Intro to Business Activity Monitoring
- Lesson 2: Enabling Business Activity Monitoring

Lesson 1: Intro to Business Activity Monitoring

In this lesson, you will learn about...

- Gaining Visibility into Business Processes
- What Is Business Activity Monitoring (BAM)?
- Business Activity Monitoring Components

Gaining Visibility into Business Processes



What Is Business Activity Monitoring (BAM)?

BAM allows users to:

- View aggregate data related to thousands of instances of a business process
- View a single instance of a business activity
- Browse/search for related instances of business activities
- View aggregate data using KPIs
- Filter activities at a given stage of completion
- Define charts, reports, KPIs, and alerts
 - PerformancePoint dashboards and SQL Server Reporting Services

Business Activity Monitoring Components

- Primary Import database
- BAM activity aggregations and OLAP cubes
- Real-Time Aggregations
- BizTalk Orchestration and Pipeline Interceptors
- BAM add-ins for Excel
- BAM Manager command line utility
- Tracking Profile Editor

Lesson 2: Enabling BAM

In this lesson, you will learn about....

- Enabling Business Activity Monitoring (Analyst)
- Enabling Business Activity Monitoring (Administrator)
- Enabling Business Activity Monitoring (Developer)
- What Is the BAM Portal?
- Demo: Monitoring Business Activity

Enabling BAM (Analyst)

Analyst defines:

- Events and data to be tracked
- Logical groupings and aggregations of tracked data views

Use BAM Activity Wizard to define:

- Activities
 - Milestones
 - Data points
- Views
 - Durations
 - Dimensions
 - Measures

Select the base data item and add or remove the available ranges.

Dimension Name: AmountSize

Dimension Type: Numeric Range Dimension

Base data item: PurchaseOrder.Amount

Available Ranges:

Name	From(Incl.)	To(Excl.)
High	500	100000
Low	0	50
Medium	50	500

New Range...
Edit...
Delete...

Help OK Cancel

Enabling BAM (Administrator)

Administrator:

- Deploy the workbook to create the BAM database structure

```
C:\> BM deploy-all -DefinitionFile:SalesManagerView.xls
```

```
C:\> BM deploy-all -DefinitionFile:SalesManagerView.xml
```

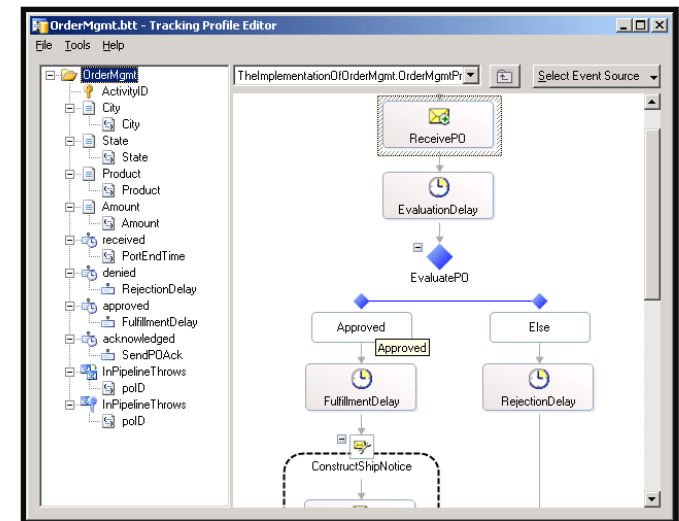

Enabling BAM (Developer)

Tracking Profile:

- Has characteristics that define a business-related process
- Contains a mapping between an orchestration and activity definition

Use Tracking Profile Editor to:

- Create and modify tracking profiles
- Select orchestration shapes
- Select schema elements
- Link multiple activities together
- Modify existing tracking profiles as long as the activity has not changed



What Is the BAM Portal?

Activity Instance Search - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites

Address http://london/BAM/Pages/Search.aspx?viewName=SalesManager&ActivityName=OrderMgmt Go Links

Business Activity Monitoring

Business Activity Monitoring

Activity Search: OrderMgmt Help

My Views

- SalesManager
 - Activity Search
 - OrderMgmt
 - Aggregations
 - OrderProgress

Query

Column Chooser

Results

Rows per page: 20

City	OrderProgress	Amount
Bellevue	Fulfillment	5778.52
Los Angeles	Fulfillment	11905.68
Los Angeles	Fulfillment	9146.07
Los Angeles	Fulfillment	1022.74
Portland	Fulfillment	5684.64
Portland	Fulfillment	4915.47
Portland	Fulfillment	8975.55
Portland	Fulfillment	10849

Activity Status

Milestones

received: 8/26/2005 9:35:06 AM
approved: 8/26/2005 10:05:06 AM
acknowledged:
denied:

Data

ActivityID: 5c426710-16bd-40b4-b554-47e5738409a7
Amount: 11905.68
City: Los Angeles
CycleDuration (Second):
OrderProgress: Fulfillment
Product: Infrared Decoder
State: CA

Demo: Monitoring Business Activity

In this demonstration, you will see how to...

- Design a BAM activity and view
- Deploy a BAM definition file
- Create a BAM tracking profile
- View tracked data using the BAM Portal

Lab: Creating a Simple BAM Solution

In this lab, you will...

- Create a simple BAM activity and view
- Deploy an activity using the BAM Manager utility
- View collected data using the BAM portal

Review Questions

- What performance impact does a BAM activity have on the execution of the BizTalk processes?
- What “non-business” data could you track using BAM?

Integrating using Parties and EDI

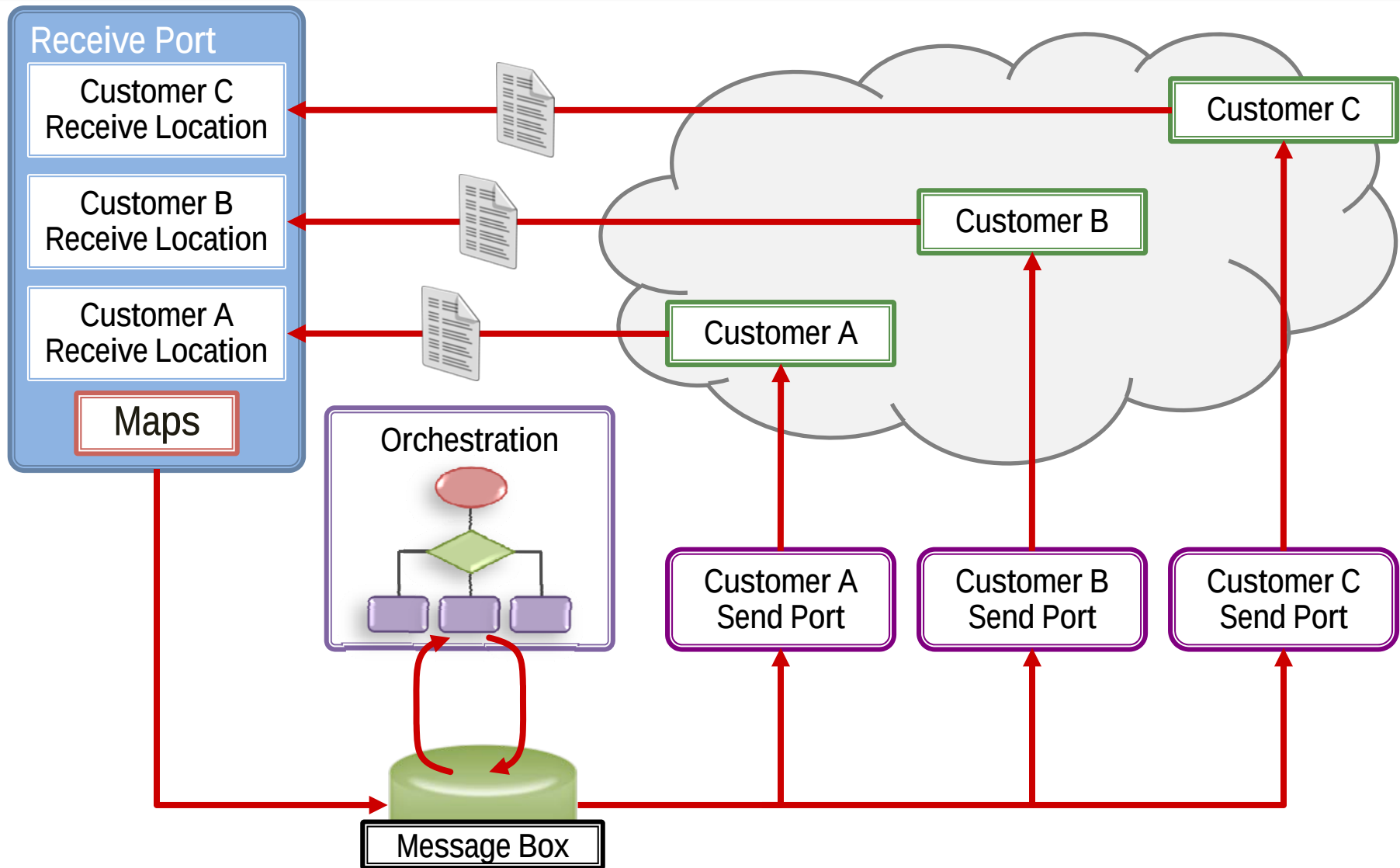
- Lesson 1: Introduction to Trading Partner Integration
- Lesson 2: Introduction to EDI
- Lesson 3: Creating EDI and AS2 Solutions
- Lesson 4: The Pieces of a TPM Solution

Lesson 1: Intro to Trading Partner Integration

In this lesson, you will learn about...

- The Need for Trading Partner Integration
- How Roles and Parties Work
- Steps to Role-Enable a Business Process
- Demo: Creating Role-Enabled Business Processes

The Need for Trading Partner Integration



How Roles and Parties Work

Role

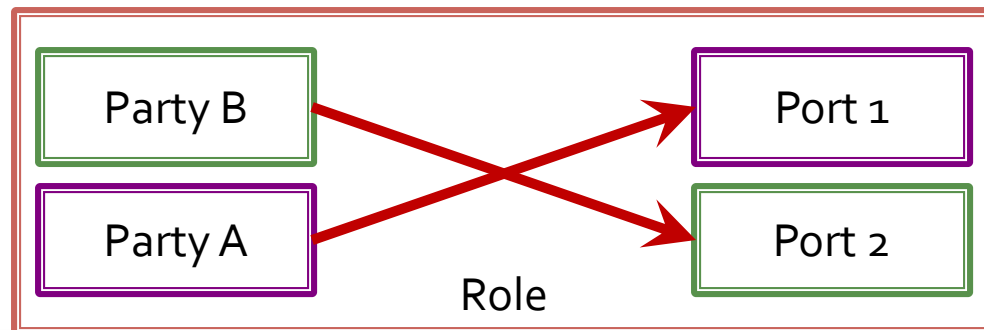
- Acts as an abstraction layer between logical and physical ports
- Orchestration is designed with a role instead of a port
- “The role a party plays in the process”

Party

- Represent external entity BizTalk communicates with

Relationship

- Parties are enlisted to a role, each party is bound to a port



Steps to Role-Enable a Business Process

1. Identify the various roles in the business process
2. Create the parties with which the business process will interact
3. Determine how parties will be resolved
4. Add role links to the orchestration
5. Assign party information to the role links
6. Deploy the orchestration
7. Enlist parties for the role links
8. Bind ports to enlist parties

In this demonstration, you will learn about...

- Role-enable an orchestration
- Create parties
- Enlist and bind parties
- Test the business process

Lesson 2: Introduction to EDI

In this lesson, you will learn about...

- What Is EDI?
- Example of an EDI Message
- Steps for Enabling EDI Messaging
- What is AS2?

What Is EDI?

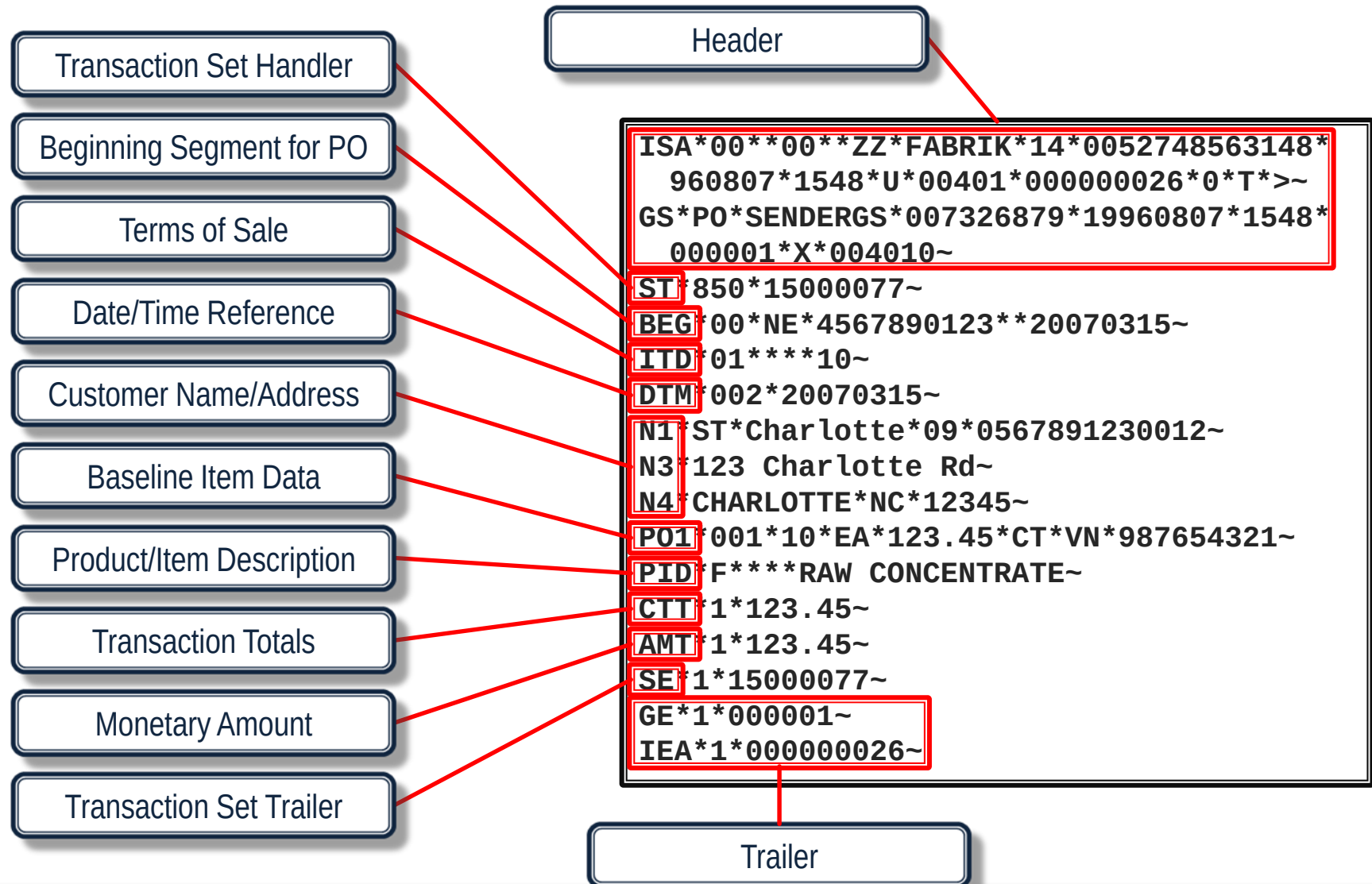
Electronic Data Interchange (EDI)

- Standard format of business documents such as: invoices, bills, and purchase orders

Traditional Characteristics of EDI Messaging

- Transported over VAN which ensures security and validity of documents
- Multiple messages aggregated into a single interchange
- Customizable standard where each organization implements a subset

Example of an EDI Message (X12)



Steps for Enabling EDI Messaging

1. Extract the EDI schemas
 - ❖ %BizTalkDir%\XSD_Schema\EDI\MicrosoftEdiXSDTemplates.exe
2. Add the desired EDI schemas to a BTS project
3. Create maps to transform messages
4. Choose the pipelines required for processing

Applicability Statement 2 (AS2)

- Is a specification that helps transport data securely and reliably over the Internet
- Eliminates or reduces Value-Added Network (VAN) costs
- Provides fast and reliable connectivity
- Supports encryption to ensure that only the sender and receiver can view the data
- Supports digital signatures to ensure authentication
- Supports hash algorithm to ensure integrity by detecting whether the document was altered during transmission

Lesson 3: Creating TPM Solutions

In this lesson, you will learn about...

- EDI and AS2 Pipelines and Components
- The Role of Batching in EDI
- How BizTalk Creates EDI Batches
- How the EDI Batching Orchestrations Work
- How BizTalk Resolves Parties for EDI Processing
- Configuring EDI Properties for Parties
- Configuring AS2 Properties for Parties
- Demonstration: Processing End-to End EDI and AS2 Messages
- Lab: Integrating with Trading Partners or Configuring EDI Processing and Batching

EDI and AS2 Pipeline Components

Component	Description
AS2 Decoder (decode)	• Provides S/MIME decoding for AS2 and MDN messages • Verifies signatures for signed messages • Decrypts and decompresses messages where enabled
EDI Disassembler (disassemble)	• Parses the EDI documents into intermediate XML messages • Splits interchanges into individual messages when enabled • Generates technical and functional acknowledgments when enabled • Promoted EDI specific properties
AS2 Disassembler (disassemble)	• Determines and defines MDN information
Batch Marker (resolve party)	• Marks the parties that need batching • Used in the party resolution section of a receive pipeline • Used when received messages are to be batched when sent as EDI documents • Sets the DestinationPartyId and ToBeBatched context properties
EDI Assembler	• Serializes the EDI interchange
AS2 Encoder	• Applies AS2/HTTP header information • Signs, encrypts and compresses outgoing messages, when enabled

EDI and AS2 Default Pipelines

Pipeline	Components
AS2EdiReceive	• AS2 Decoder • EDI Disassembler • Batch Marker
AS2Receive	• AS2 Decoder • AS2 Disassembler
BatchControlMessageRecv	• XML Disassembler
EdiReceive	• EDI Disassembler • Batch Marker

Receive Pipelines

Send Pipelines

Pipeline	Components
AS2EdiSend	• EDI Assembler • AS2 Encoder
AS2Send	• AS2 Encoder
EdiSend	• EDI Assembler

When Receiving

1. Match the **sender** and **receiver qualifiers** and **identifiers** in the header to a party
2. If step **1** does not succeed: Match only the **sender qualifier** and **identifier** in the header to a party
3. If step **2** does not succeed: Use global party properties

When Sending

1. Match the **DestinationPartyName** property to a party
 - ❖ Set this property in a custom component
2. If step **1** does not succeed: Match the **sender** and **receiver qualifiers** and **identifiers** in the message context to a party
3. If step **2** does not succeed: Match the send port to a party based on association
4. If step **3** does not succeed: Use global party properties

The Role of Batching in EDI

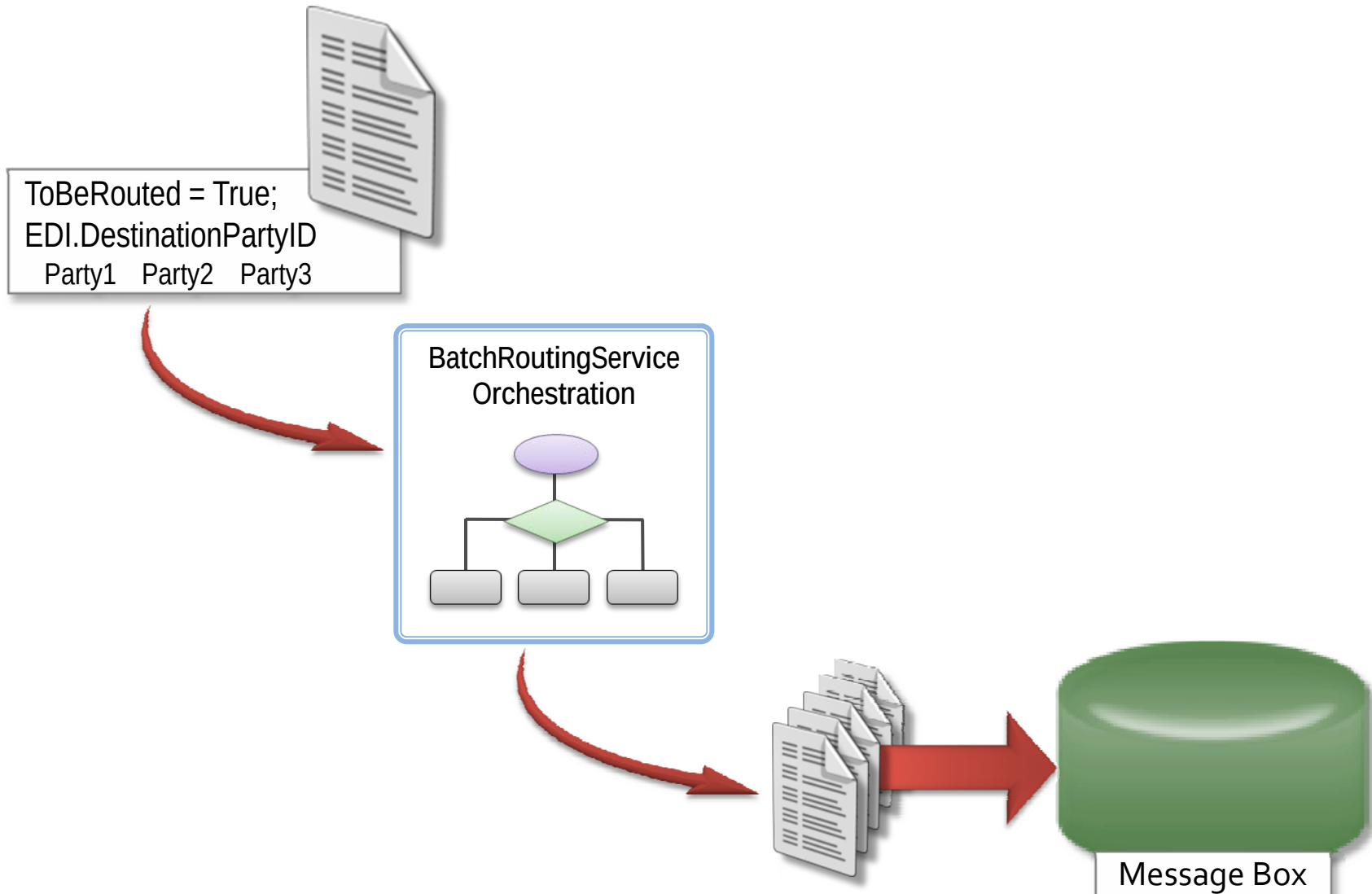
Rules for EDI Batching

- A Transaction Set includes multiple messages of a single type
- A Message Group can contain multiple Transaction Sets, as long as they all have the same message type
- An Interchange (batch) can contain multiple message groups

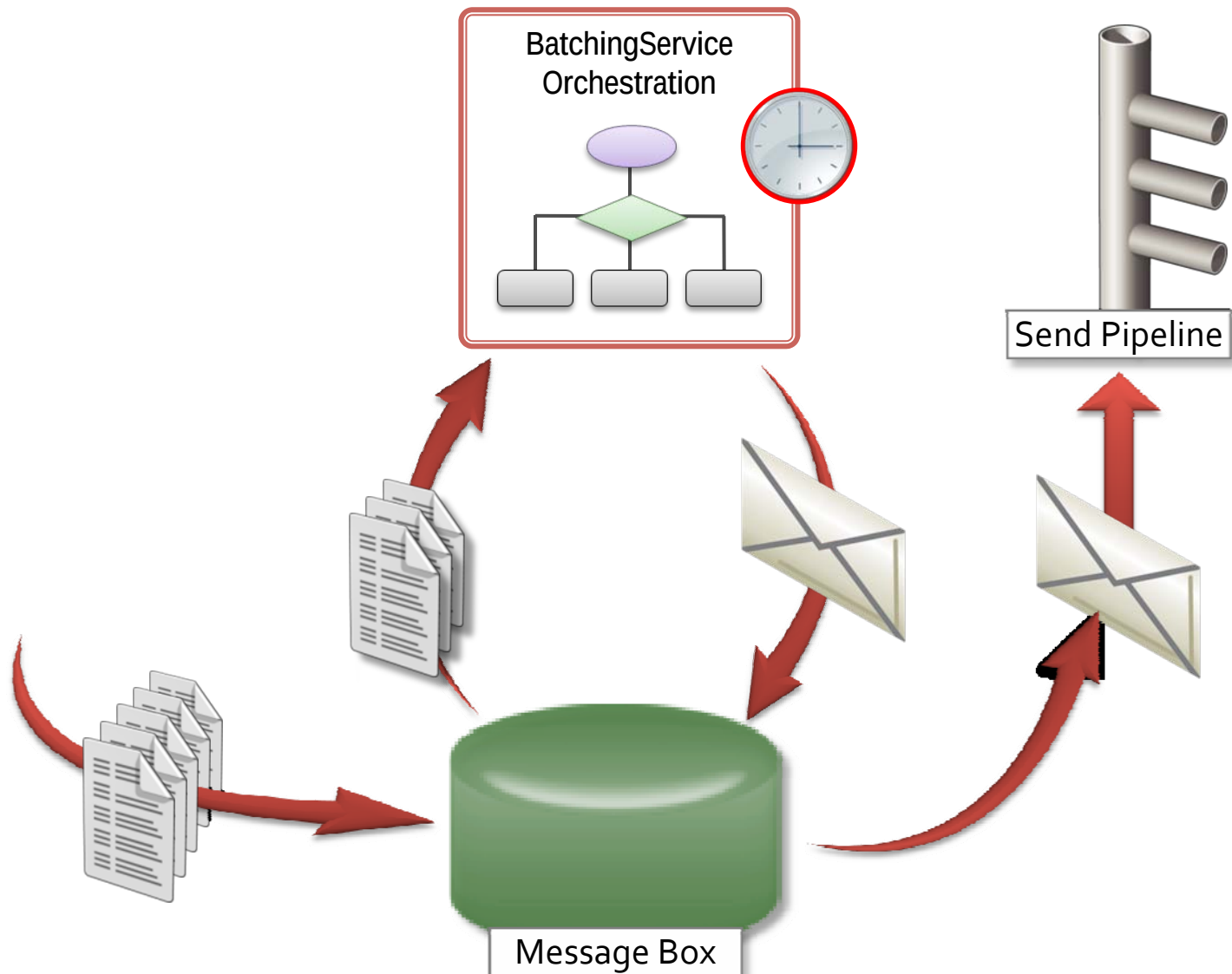
How BizTalk Creates EDI Batches

1. The **BatchMarker** pipeline component evaluates filters of batch properties in parties and marks the message for batching
2. The routing orchestration subscribes to marked messages and creates copies for each batching process and publishes those messages
3. Batching orchestration subscribes to its messages and aggregates multiple messages
4. The batching orchestration creates batch properties on message and publishes
5. A send port subscribes to batched interchanges

How the EDI Batching Orchestrations Work



How the EDI Batching Orchestrations Work



Lesson 4: The Pieces of a TPM Solution

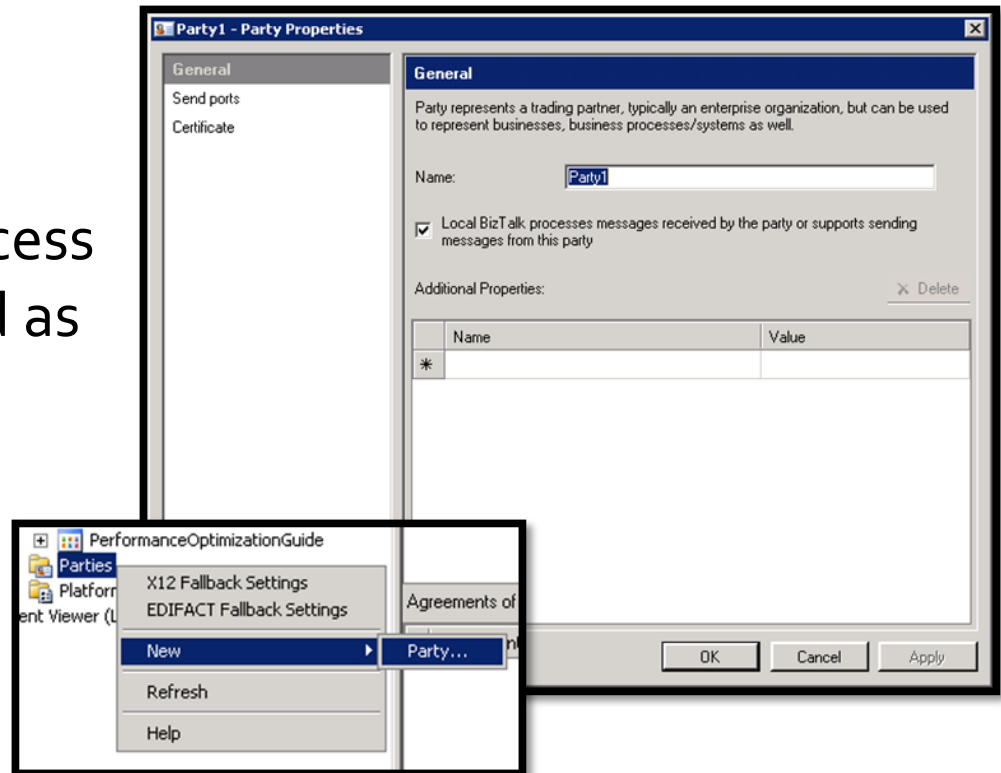
In this lesson, you will learn about...

- What are Trading Partners?
- What are Business Profiles?
- Profile Specific Protocol Settings
- What Are Trading Partner Agreements?
- Creating a TMP Solution
- Demo: Implementing a Party Based B2B Solution

What are Trading Partners?

Trading partners:

- Also known as *Parties*
- Root level of the TPM architecture
- Each entity in the process should be represented as a party
 - Including home party



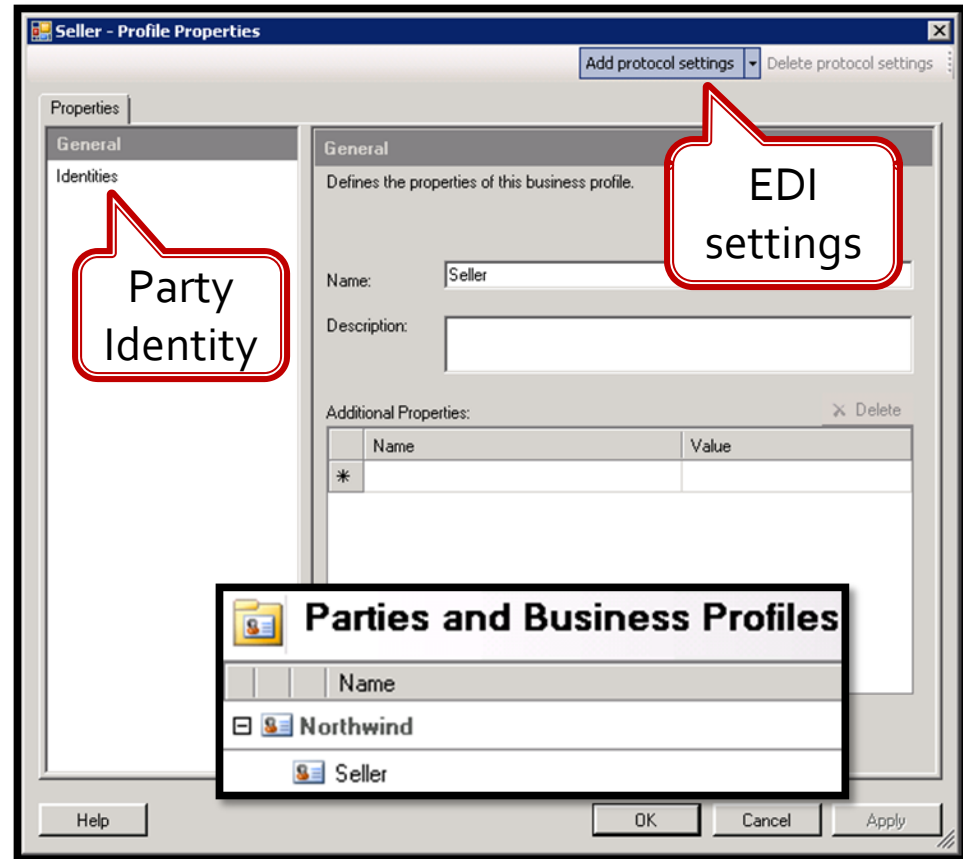
What are Business Profiles?

Profiles:

- Represent facets of a party
- A single party may have multiple profiles
 - Shipping
 - Invoicing
- Each profile has protocol and identity settings

For Example:

- One business has US and European divisions



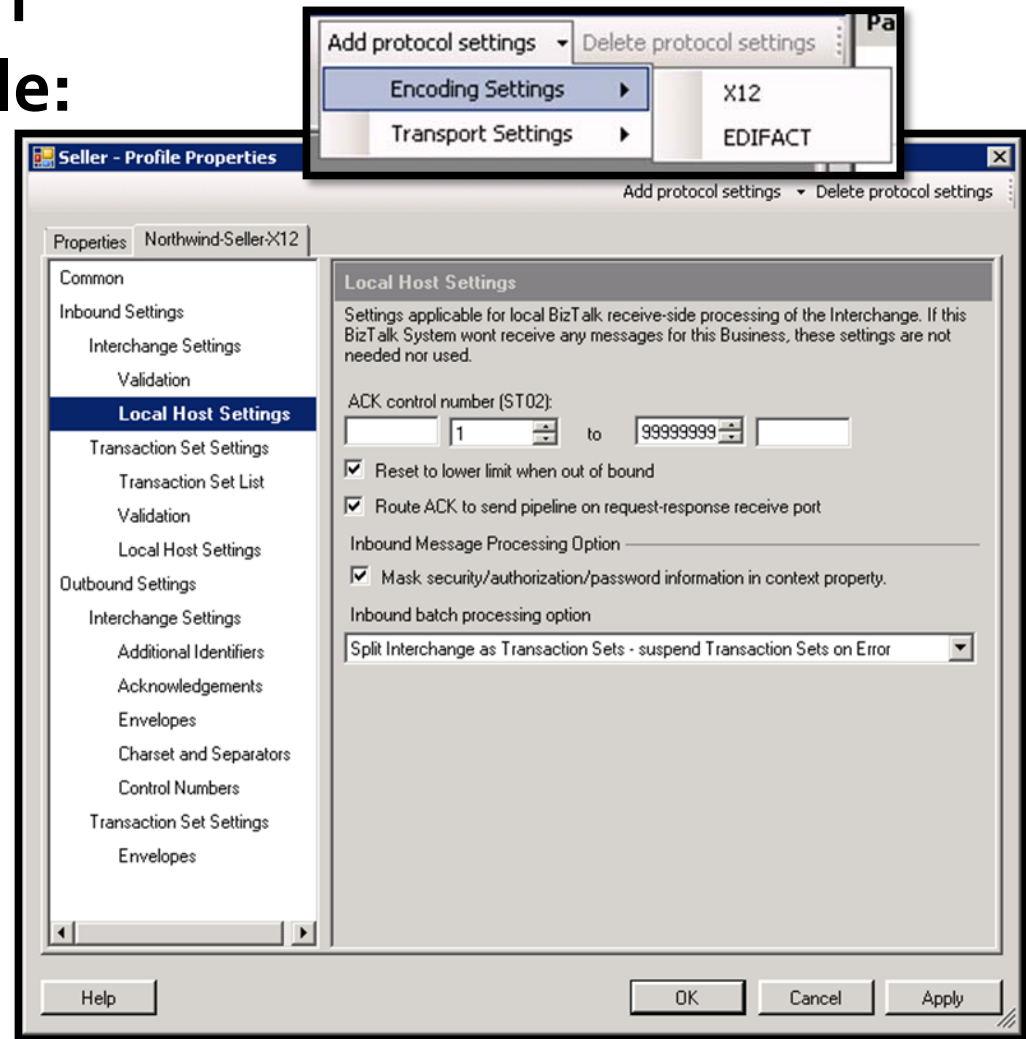
Profile Specific Protocol Settings

Define communication parameters for a profile:

- Encoding Protocol Settings
 - X12, EDIFACT
- Transport Protocol Settings
 - AS2

For Example:

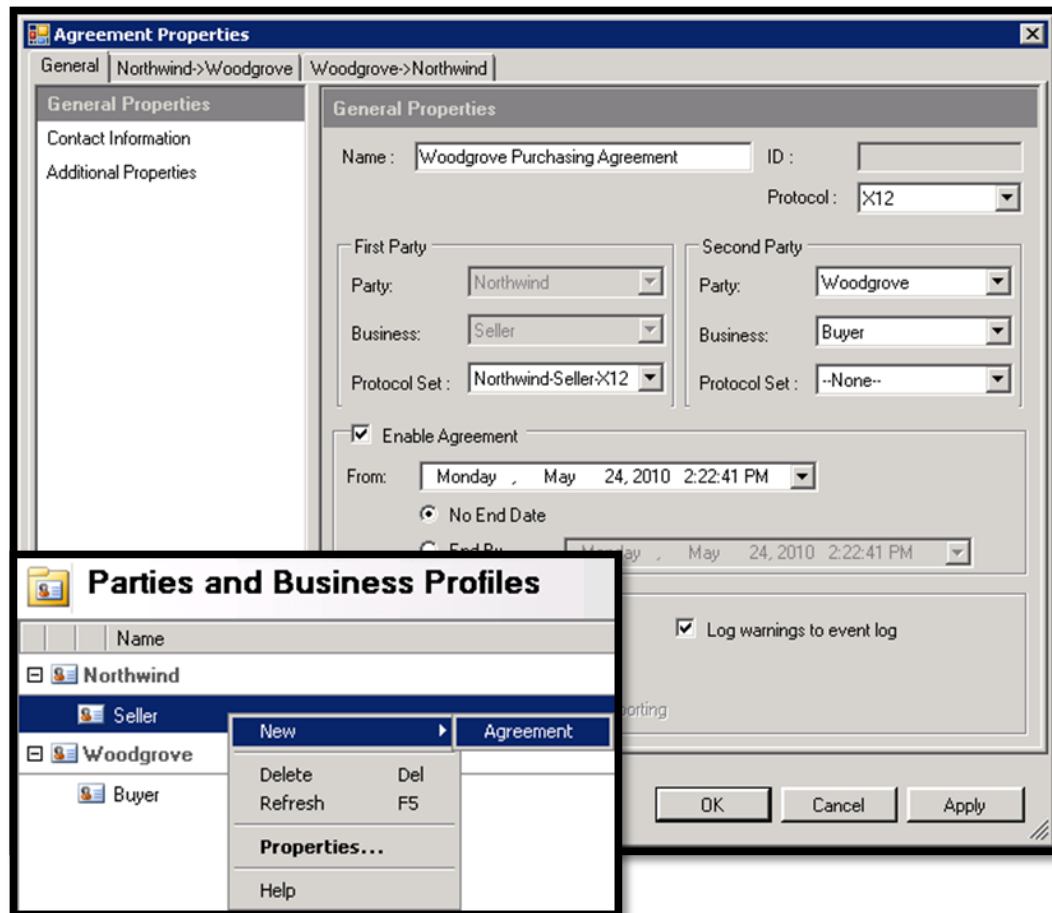
- US division uses X12
- European division uses EDIFACT



What Are Trading Partner Agreements?

Agreements:

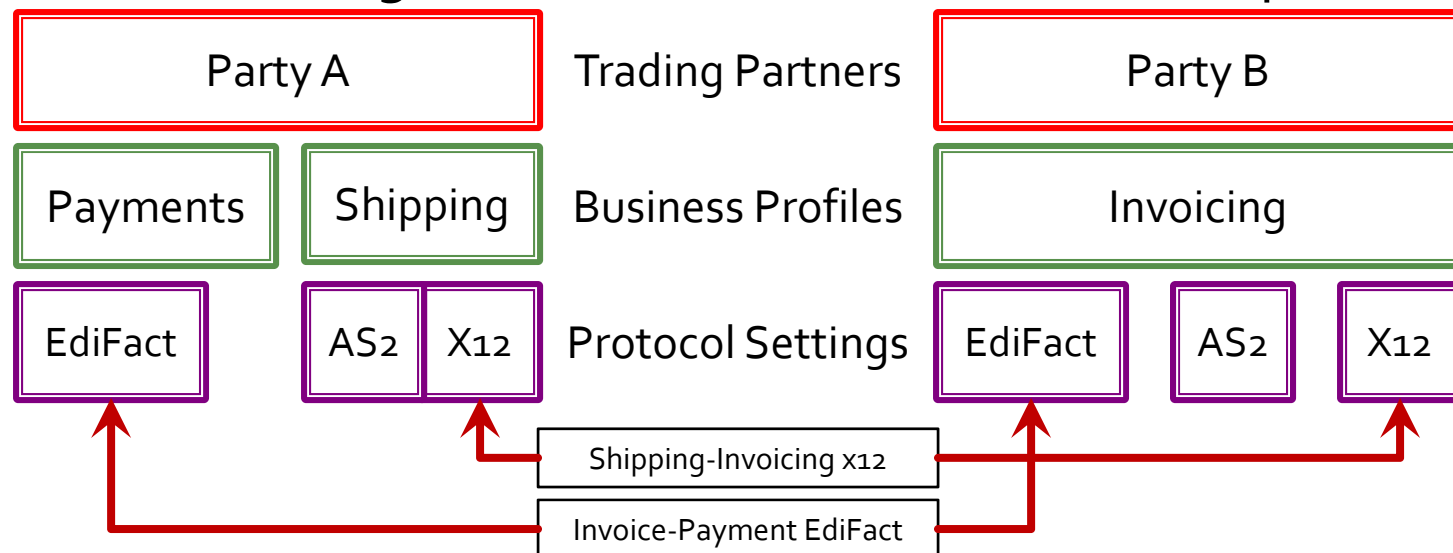
- Define how two profiles interact
- Includes the protocol settings
- Provides better conceptualization of relationships



Creating a B2B Integration Solution

The typical process for implementing party-based integration:

1. Create a party for each organization in the process
2. Create a profile for each division or role of the party
3. Define the B2B protocol settings for each profile
4. Create the agreements between the business profiles



Demo: Implementing a Party Based B2B Solution

In this demonstration, you will...

- Examine a role-based orchestration
- Examine Parties
- Examine Party Profiles
- Examine Business Agreements
- Examine EDI configuration and batching

Lab: Implementing an EDI Messaging Solution

In this lab, you will...

- Create and configure trading partners.
- Route messages based on trading partner preferences.
- Design a role-enabled orchestration.
- Configure EDI properties to enable creation of EDI messages.
- Configure the batching mechanism for EDI messages.

Review Questions

- Why are parties and role-enabled processes important for having a loosely-coupled design?
- Explain how parties are used for EDI processing.