



webMethods 8 Integration Workshop

Participant Guide

This publication is protected by international copyright law. All rights reserved. No part of this publication may be reproduced, translated, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior permission of Software AG.

Software AG and all Software AG products are either trademarks or registered trademarks of Software AG. Other product or company names mentioned herein may be the trademarks of their respective owners.



webMethods 8 Integration Workshop

TRA611-41E





Copyright

- This publication is protected by international copyright law. All rights reserved. No part of this publication may be reproduced, translated, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior permission of Software AG.
- Software AG and all Software AG products are either trademarks or registered trademarks of Software AG. Other product or company names mentioned herein may be the trademarks of their respective owners.

Software AG Training | Page 2



Welcome to Software AG Training!

- Housekeeping Items
 - Class hours
 - Refreshments
 - Smoking
 - Restrooms
 - Emergency exits
 - Sign-in sheets
- So Everyone Benefits, please:
 - Turn off/silence cell phones and pagers
 - Check e-mail only at breaks
 - Refrain from side discussions
We all want to hear what you have to say!
 - Feel free to ask questions during the lecture



Table of Contents

- | | | |
|-----------|--------------------------|--|
| Chapter 1 | webMethods Overview | |
| Chapter 2 | Designer | |
| Chapter 3 | Introduction to Services | |
| Chapter 4 | Document Types | |
| Chapter 5 | Flow Services | |
| Chapter 6 | Mapping | |
| Chapter 7 | Java Services | |
| Chapter 8 | Monitoring Services | |
| Chapter 9 | Invoking Services | |



Table of Contents

- | | | |
|------------|---------------------------|--|
| Chapter 10 | Flat File Handling | |
| Chapter 11 | Web Services | |
| Chapter 12 | Broker Messaging | |
| Chapter 13 | JMS Messaging | |
| Chapter 14 | JDBC Adapter | |
| Chapter 15 | Business Process Modeling | |
| Chapter 16 | Wrap Up | |



This page intentionally left blank.

Software AG Training | Page 6

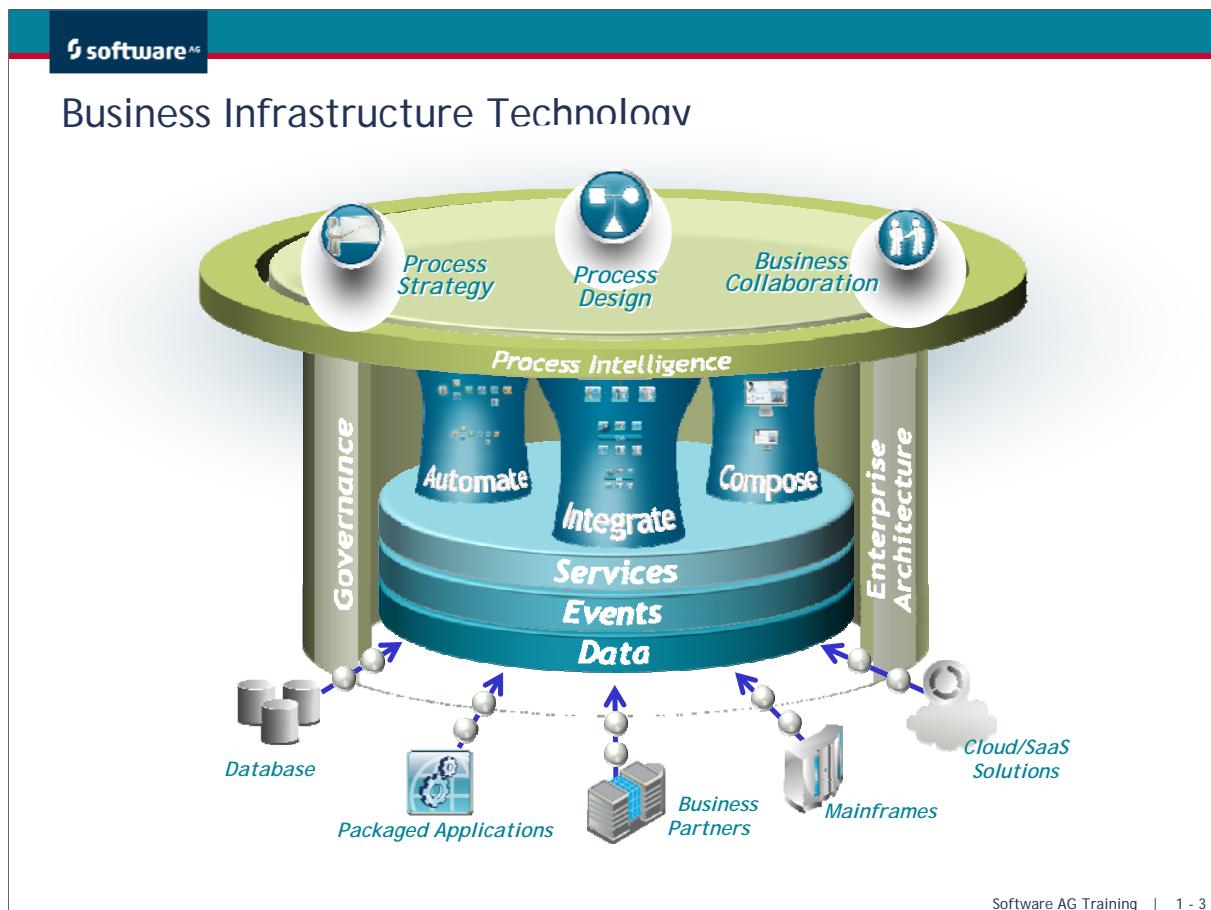


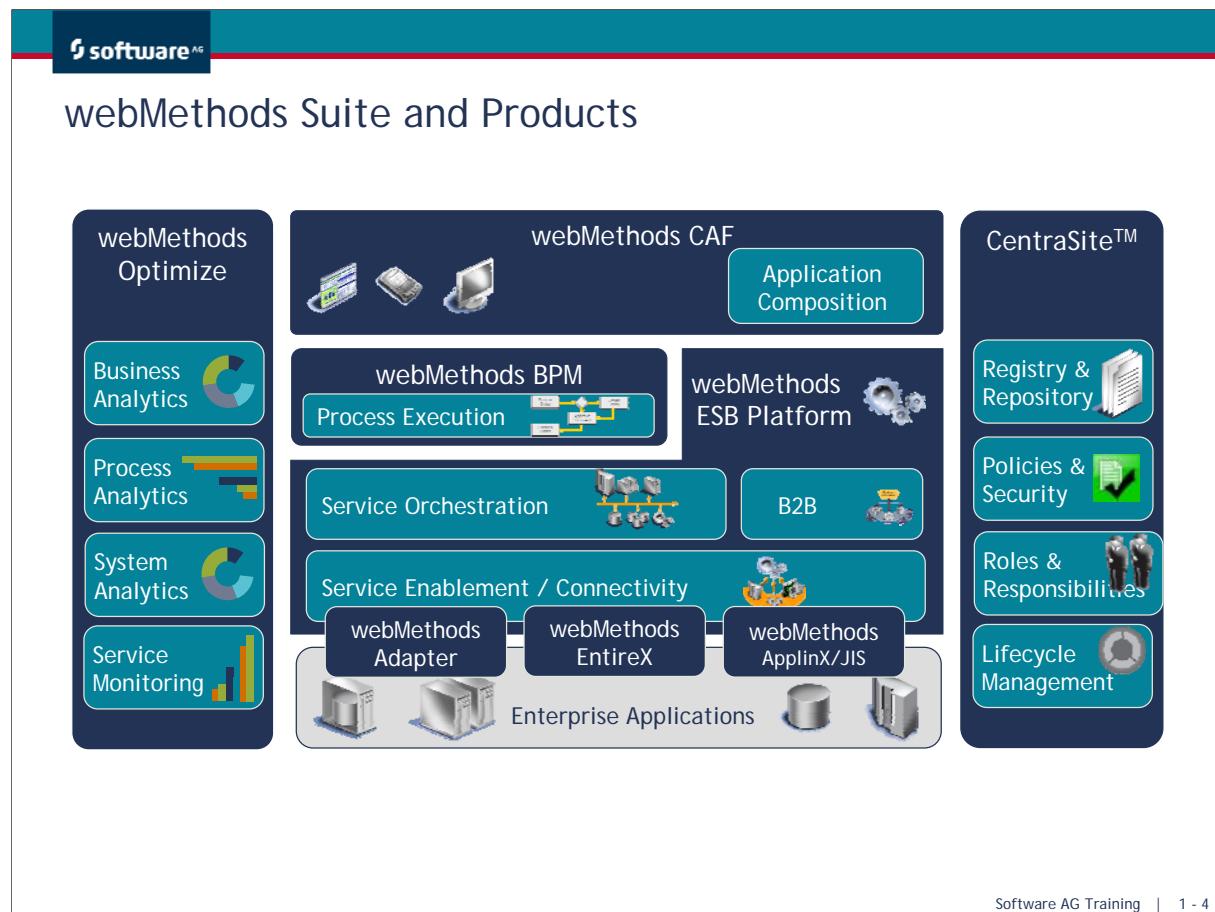
1

webMethods Overview

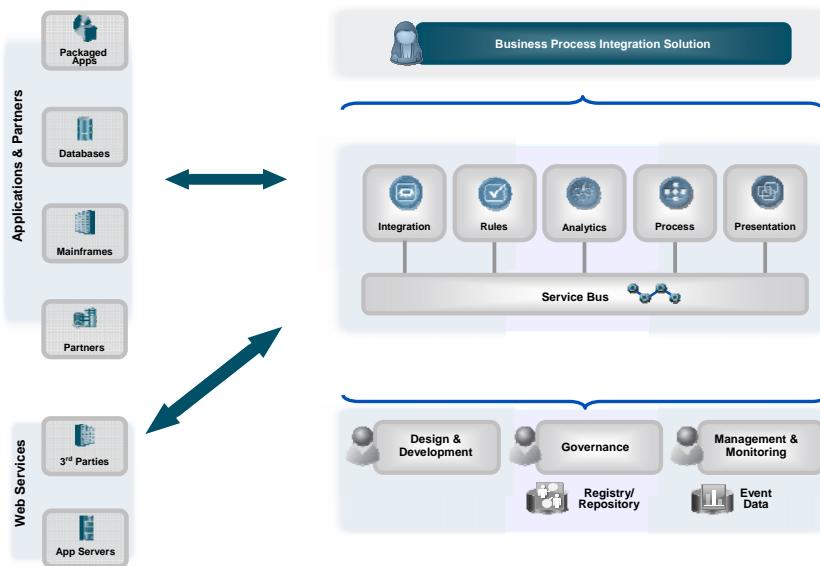
The Enterprise Environment

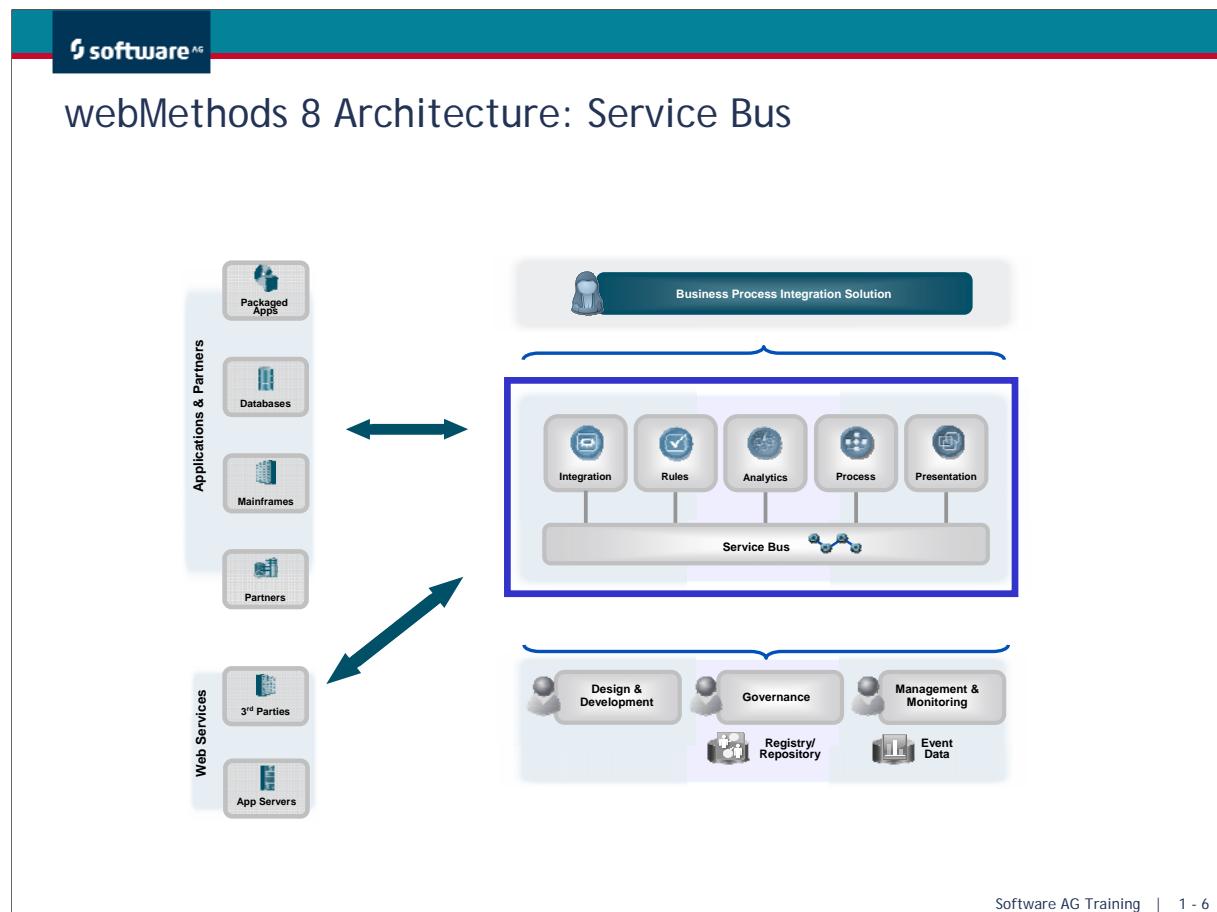
- Three key challenges faced by today's enterprises:
 - The need to integrate applications essential to the company's business
 - The need to analyze, streamline and improve business processes
 - The need to deliver agility and lower maintenance costs by exploiting Service-Oriented Architecture (SOA)
- To address these challenges, Software AG's Business Infrastructure Technology is designed to facilitate and deliver the convergence of integration (IS), B2B, Business Process Management (BPM), Business Activity Monitoring (BAM), and SOA.





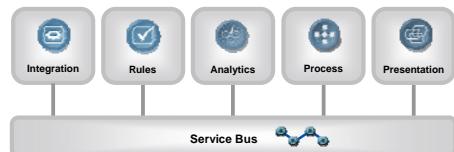
webMethods 8 Architecture

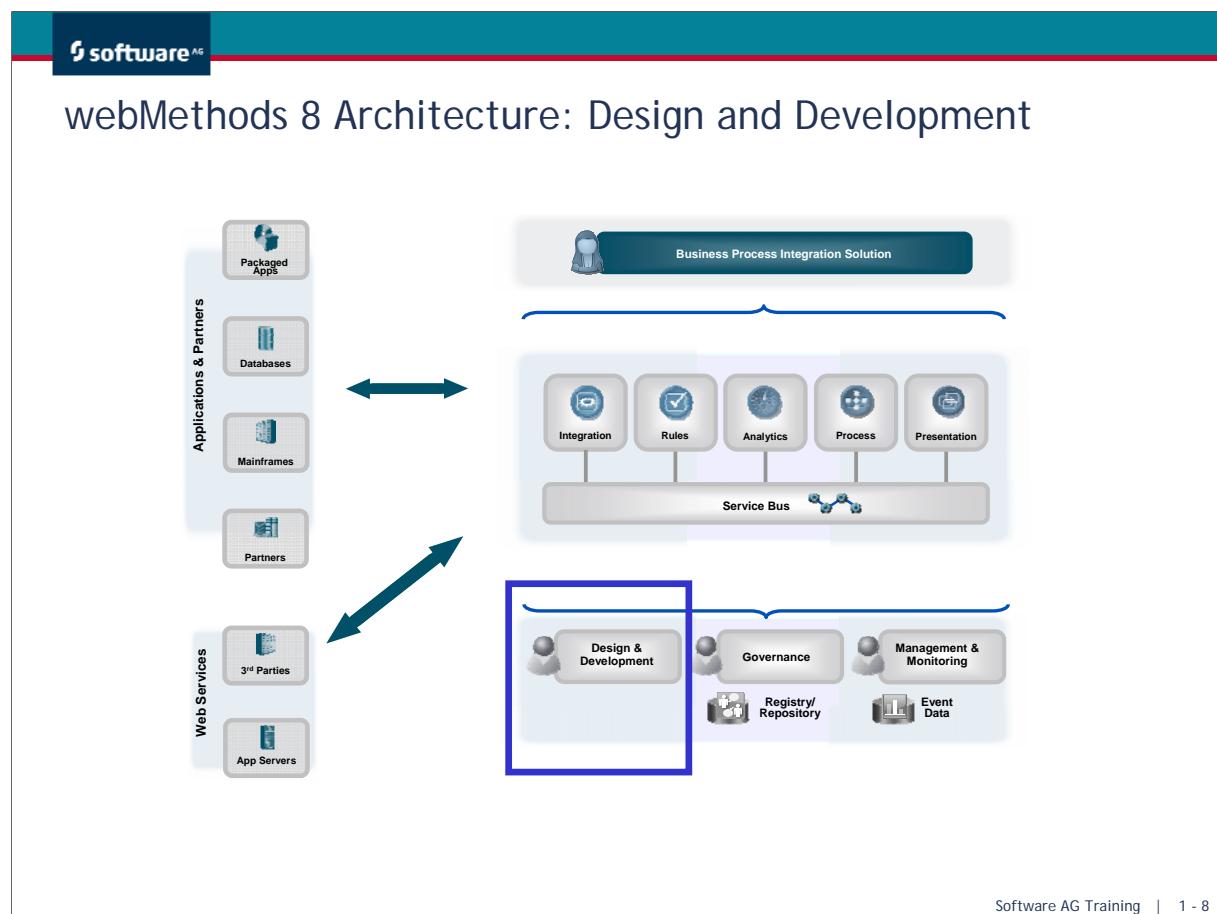




Service Bus

- The Enterprise Service Bus (ESB), provides:
 - Central middleware and messaging infrastructure
 - Supports multiple communications protocols
 - Data transformation (mapping)
- Consists of:
 - Integration Server
 - Trading Networks
 - Broker
 - Adapters, Industry Adapters & Solutions
 - Interfaces for process management, monitoring, rules & presentation

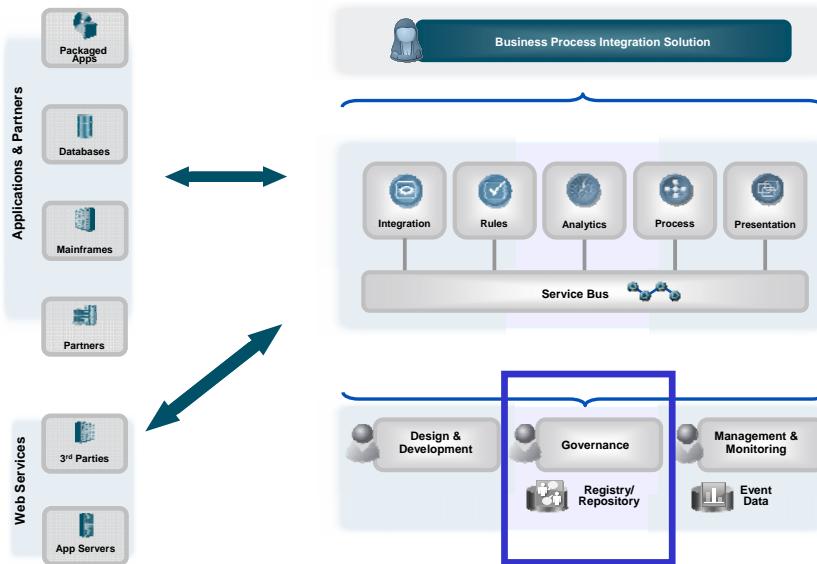


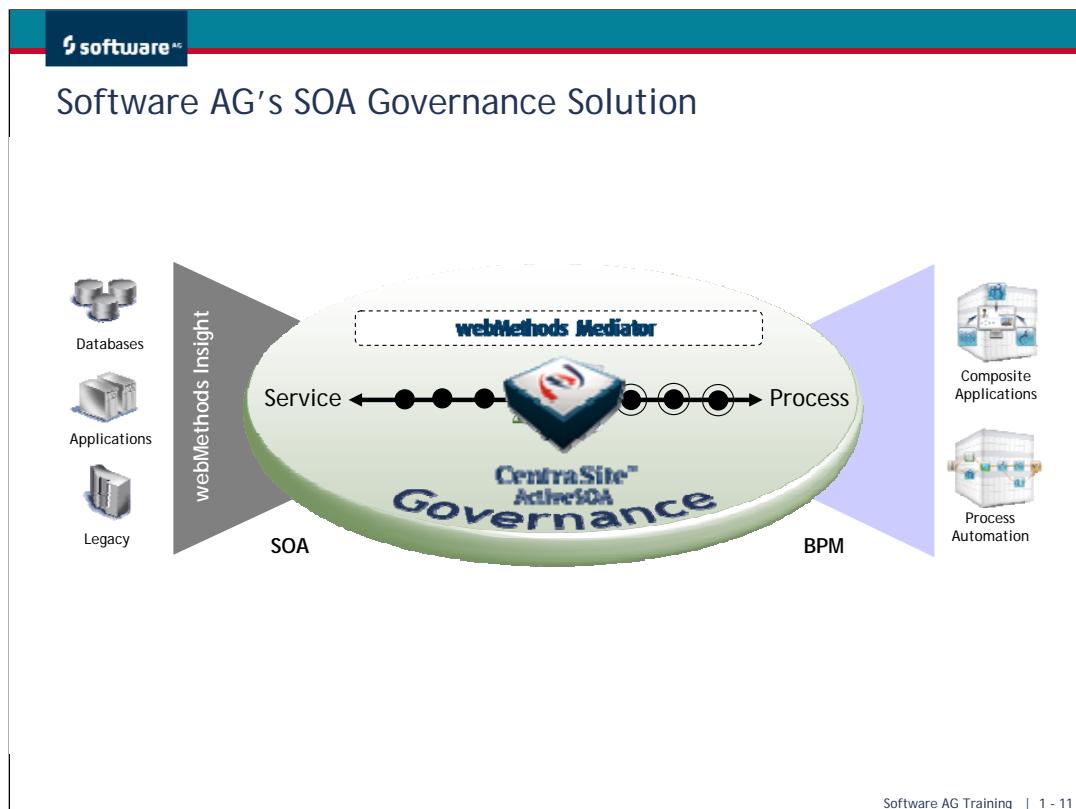


Design and Development

- The Design and Development tools consist of:
 - Designer
 - Services, Triggers, Process models, tasks, user interfaces
 - Developer
 - Legacy IS development
 - TN Console
 - Document types and processing rules
 - My webMethods Server
 - Interfaces for monitoring, configuration, and user interaction

webMethods 8 Architecture: Governance

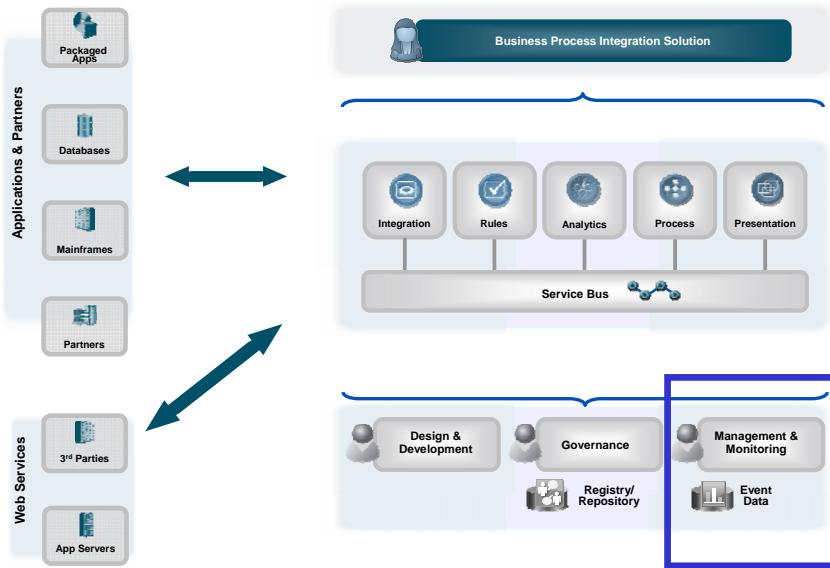


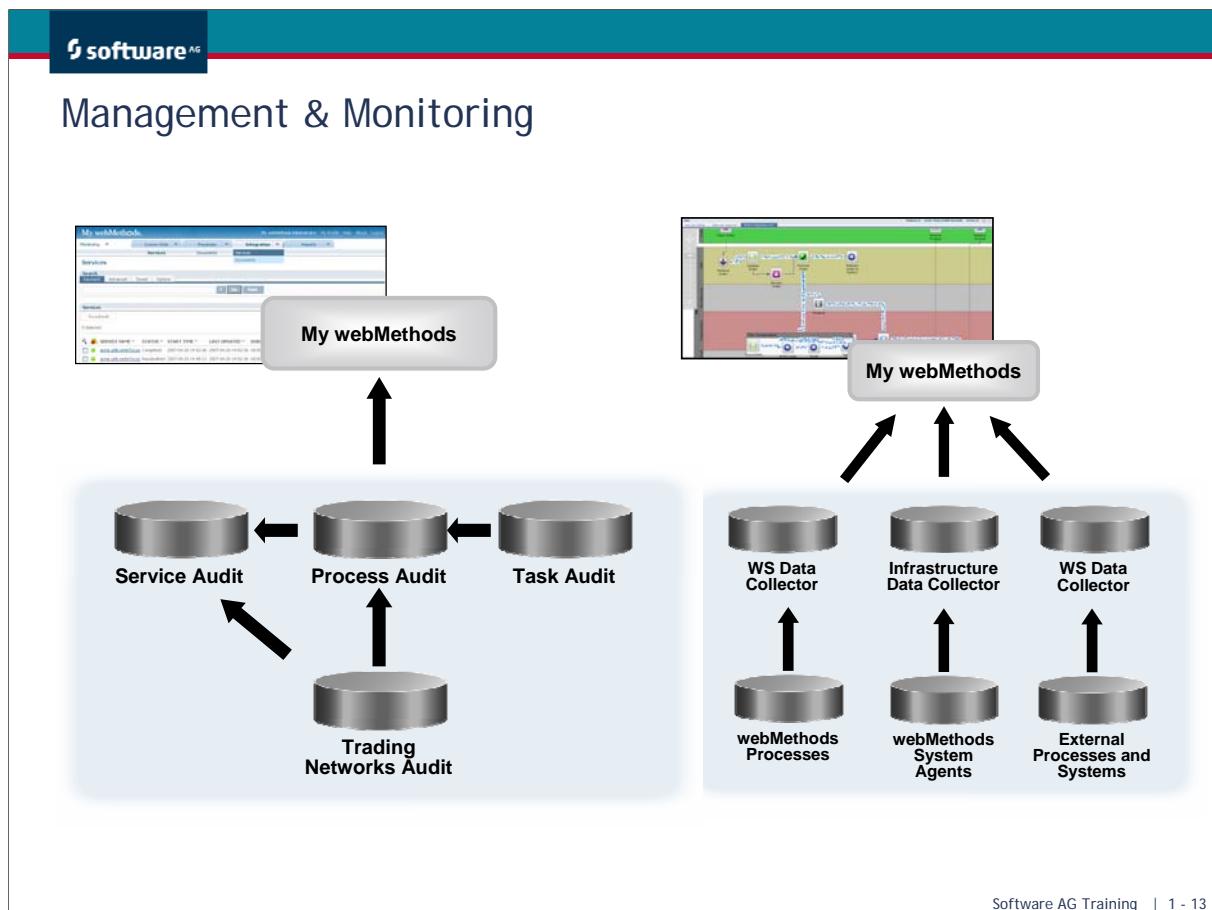


Software AG Training | 1 - 11

- For SOA: - Structure, scale and speed of adoption.
- Need for BSR: -
 - What is my portfolio of services that I need to support my BPM roadmap?
 - How do I define and associate my services, processes and data?
 - How do I create, compose and modify processes leveraging existing services?
 - How do I guide process lifecycles and manage change to processes that cut across organizational silos?
 - How do I manage service quality and reliability by creating a transparency and accountability structure?

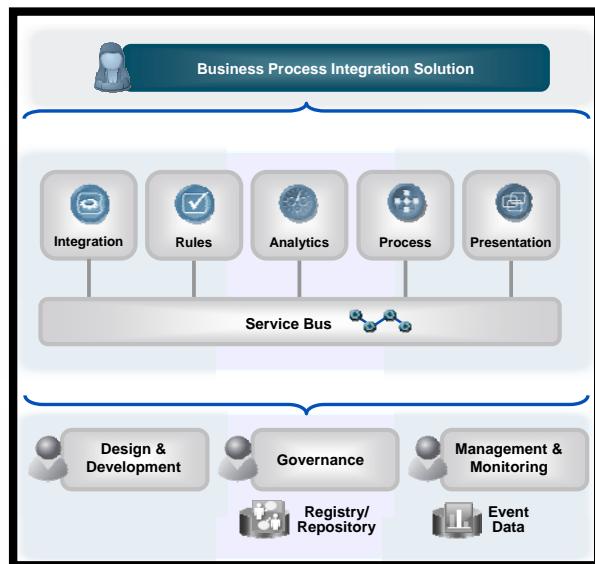
webMethods 8 Architecture: Management & Monitoring





Classroom Architecture

- Components used in this class:
 - Service Bus
 - Broker Server
 - Integration Server
 - Adapters
 - Design & Development
 - Designer (Developer)
 - Management & Monitoring
 - My webMethods Server
 - External Products
 - MS SQL Server (Express Ed.)
 - Other 3rd Party tools (SOAP UI)



Start/Stop the Broker Server

- Start and stop from the command line:
 - .../Broker/bin/broker_start -switch server[:port]
 - .../Broker/bin/broker_stop -switch server[:port]
- Start and stop as a Windows service:

 Software AG webMethods Broker Monitor 8.0 (6850)	Started	Automatic
 Software AG webMethods Broker Server 8.0 (6849)	Started	Manual
- Start / stop / restart from My webMethods



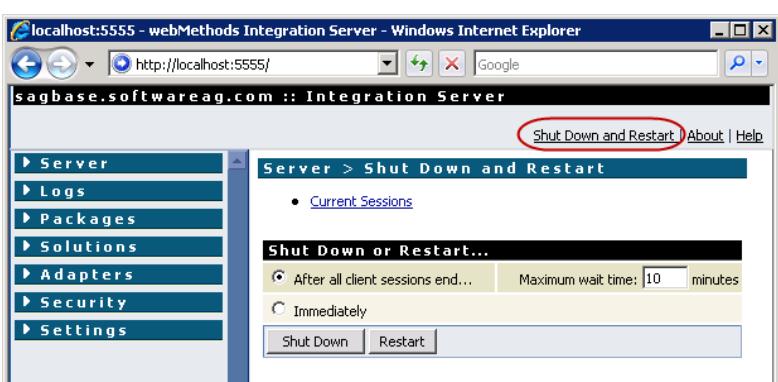
The screenshot shows a table titled "Broker Servers List" with the following data:

SERVER NAME	CONNECTED	CONNECTIONS	SSL
localhost	Error	-	-

At the bottom right of the interface, there is a navigation bar with links: « Previous | 1 | Next ».

Start/Stop the Integration Server

- Start from the command line:
 - .../IntegrationServer/bin/server.sh -switch -switch (Unix)
 - ...\\IntegrationServer\\bin\\server.bat -switch -switch (Windows)
- Start as a Windows service:

Software AG webMethods Infrastructure Data Collect...	Software A...	Manual	Local System
Software AG webMethods Integration Server 8.0	Software A...	Started	Manual
Software AG webMethods MWS 8.0 (default)		Started	Manual
- Stop/restart from the Administrator

The screenshot shows a Windows Internet Explorer window titled 'localhost:5555 - webMethods Integration Server - Windows Internet Explorer'. The URL is 'http://localhost:5555/'. The page displays the 'sagbase.softwareag.com :: Integration Server' interface. On the left, there is a navigation menu with items: Server, Logs, Packages, Solutions, Adapters, Security, and Settings. The 'Server' item is currently selected. On the right, a sub-menu titled 'Server > Shut Down and Restart...' is open. It contains a section 'Shut Down or Restart...' with two radio button options: 'After all client sessions end...' (selected) and 'Immediately'. Below this are 'Shut Down' and 'Restart' buttons. The 'Shut Down and Restart...' link in the sub-menu is circled in red.

Software AG Training | 1 - 16

Start/Stop MWS

- Start from the command line (in a new Window):

```
.../MWS/bin/mws.bat start
```

- Start from the command line (in the same Window):

```
.../MWS/bin/mws.bat run
```

- Register and start as a Windows service:

```
.../MWS/bin/mws.bat registerservice
```

```
.../MWS/bin/mws.bat startservice
```

This page intentionally left blank.

Software AG Training | 1 - 18

2

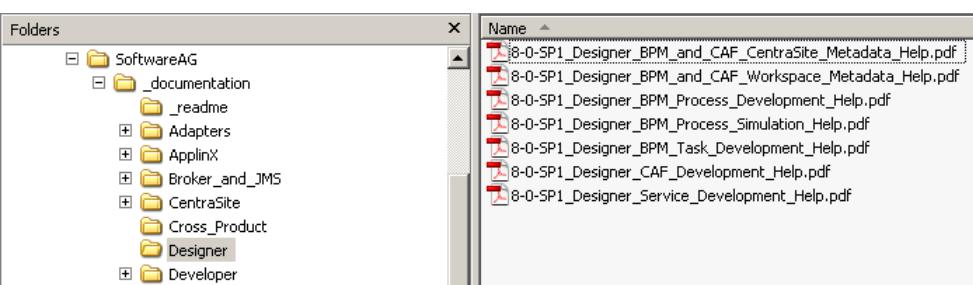
Designer

Objectives

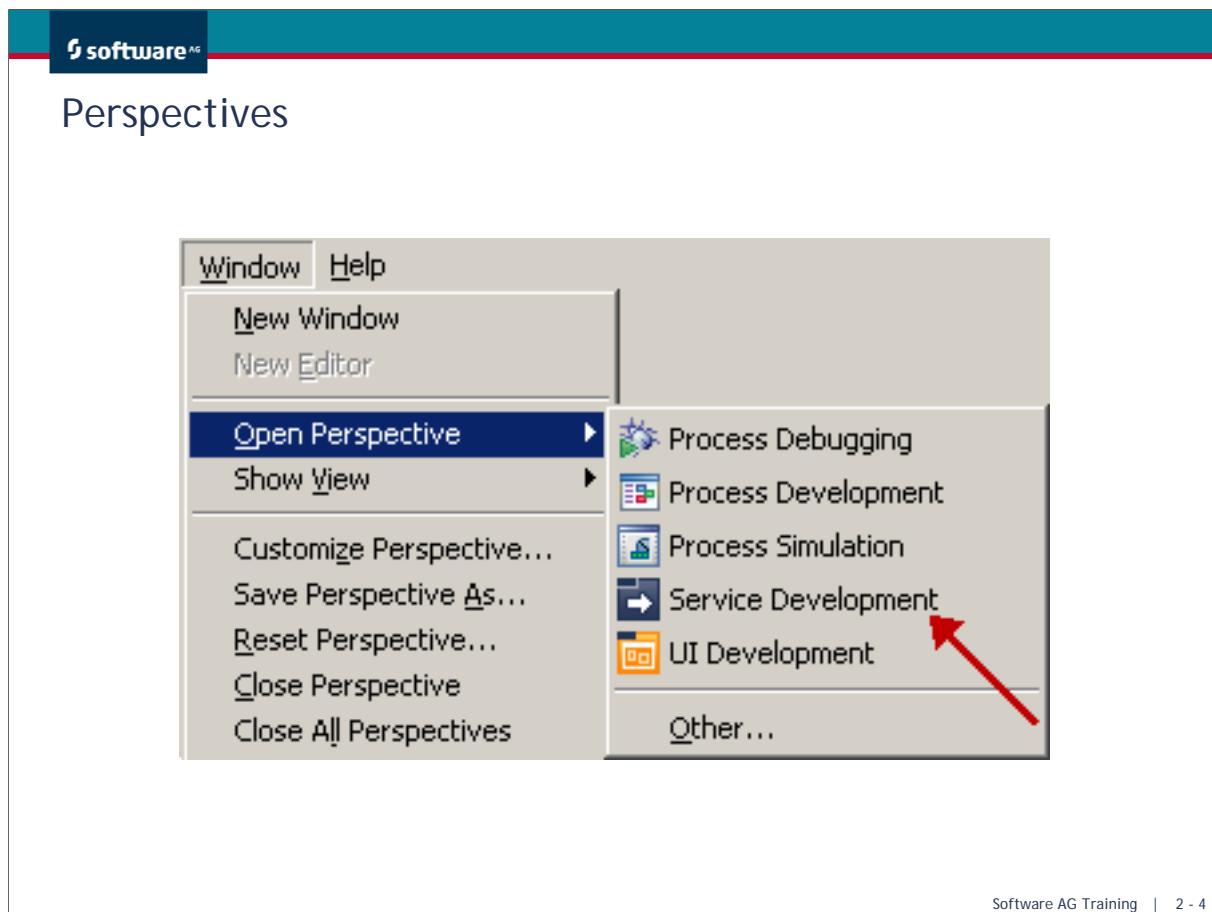
- At the end of this section, you will be able to:
 - find the Documentation
 - differentiate between Packages & Services
 - define the Integration Server Namespace
 - describe Best Practices for organizing project development

Documentation in the Filesystem

- All documentation for the webMethods 8 Product Suite is available in the _documentation folder, which is stored directly in the installation directory.
 - Usually: C:\SoftwareAG\documentation\Designer

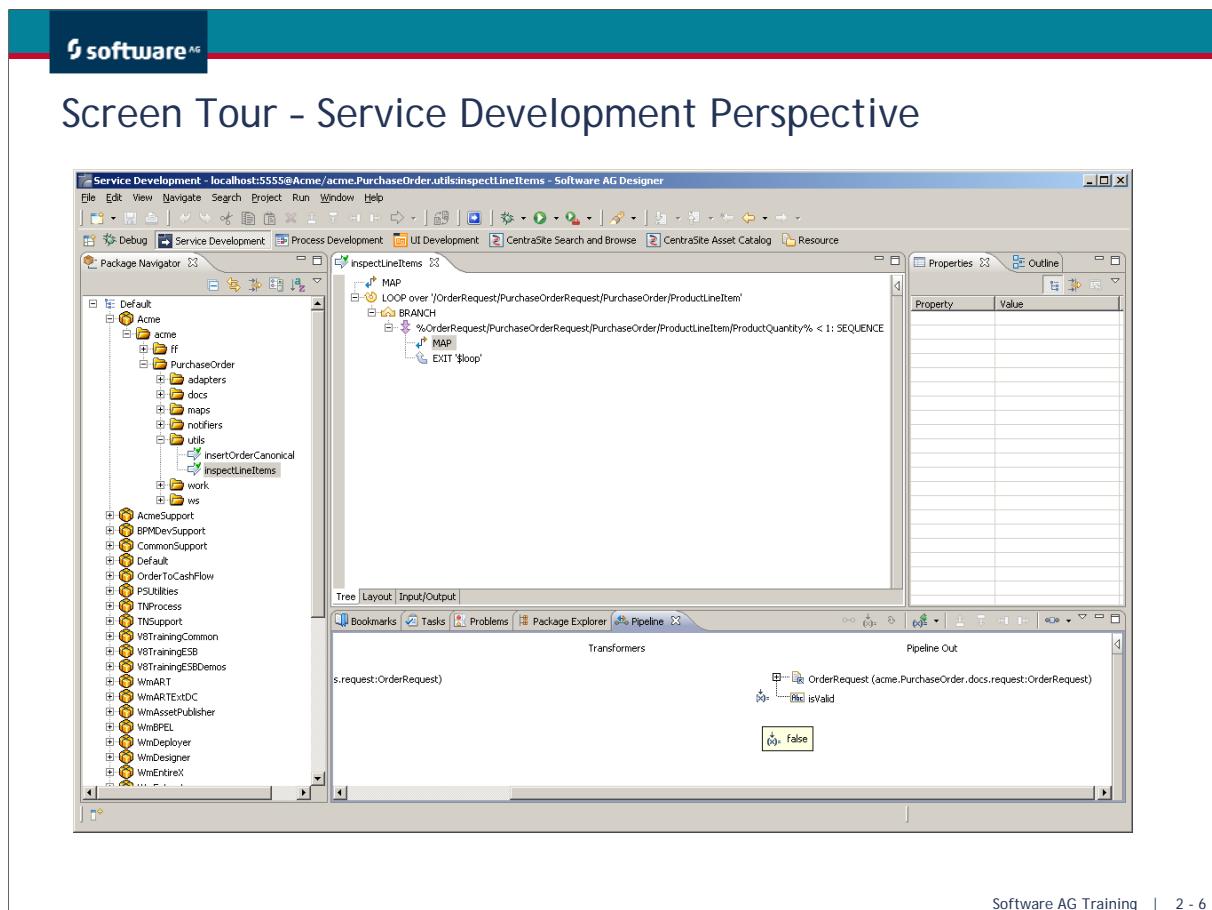


Software AG Training | 2 - 3



The screenshot shows the 'Help - Software AG Designer' window. The left pane contains a 'Contents' tree view with various developer guides listed. A red arrow points to the 'Software AG Designer Guide' node. The right pane displays the main help content for 'webMethods Service Development Help'. The title 'Software AG Designer Guide > webMethods Service Development Help' is at the top. Below it is a 'Contents' section with several links: Legal, webMethods Service Development, Working with webMethods Integration Server, Working with Elements, Assigning and Managing Permissions for Elements, and Locking and Unlocking Elements.

Software AG Training | 2 - 5

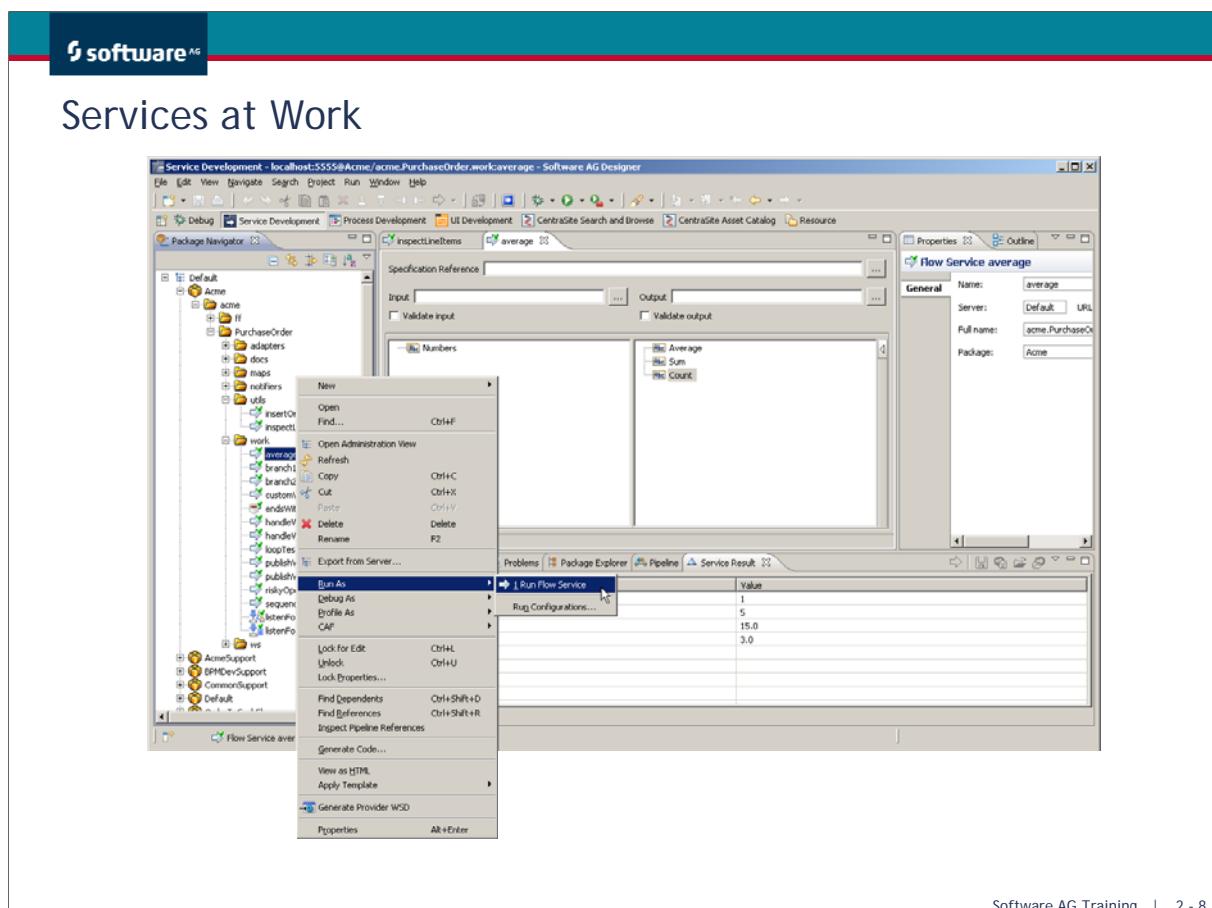


Software AG Training | 2 - 6

Services in Detail

- Integration is performed by Services
- Services:
 - are discrete functions or method calls
 - are the “nuts & bolts” of integration
 - can be written in Flow and Java
 - are accessible as web services
 - can perform any integration task
 - Transportation: e.g. HTTP Post
 - Transformation: e.g. XML -> IDOC
 - Auditing: e.g. custom event logging

webMethods 8 Integration Workshop



Package in Detail

- A package is a logical container for a set of services and related files
 - Smallest deployable unit of software
 - Can be archived/restored/disabled/enabled
- A package should contain all objects and files required by the services it hosts (e.g. jars, templates, documentation, configuration files, etc.).
- Packages exist as a single physical directory on the file system
 - Located in the [webMethodsHome]\IntegrationServer\packages*

software AG

Package Structure

Packages
Contain folders
which may contain:

- Flow Services
- Java Services
- Specifications
- Triggers
- Document Types
- C++ Services
- Schemas
- Web Service Descriptors
- Adapter Services
- Dictionaries
- XSLT Services

```
graph TD; TrainV6[TrainV6] --> trainV6[trainV6]; TrainV6 --> examples[examples]; trainV6 --> myFlowService[myFlowService]; trainV6 --> myJavaService[myJavaService]; trainV6 --> mySpecification[mySpecification]; trainV6 --> myBrokerTrigger[myBrokerTrigger]; trainV6 --> myJMSTrigger[myJMSTrigger]; trainV6 --> myDocumentType[myDocumentType]; trainV6 --> myCplusplus[myCplusplus]; trainV6 --> mySchema[mySchema]; trainV6 --> myProviderWSDescriptor[myProviderWSDescriptor]; trainV6 --> myAdapterService[myAdapterService]; trainV6 --> myFlatFileDictionary[myFlatFileDictionary]; trainV6 --> myXSLTservice[myXSLTservice]
```

Software AG Training | 2 - 10

NOTE: As of today (March 2010) not all of these Object types can be edited with designer. The most notable exceptions being Specifications, BrokerTriggers and FlatFile Dictionaries / Schemas.

Namespace in Detail

- Services are organized via the Namespace and are:
 - similar to a directory structure
 - uniquely identify Services
- A Service is uniquely identified by folder [.subfolder...]:service.
- Package names are NOT in the namespace.
- Folders merge; Services collide.



Software AG Training | 2 - 11

Namespace Conflicts

The diagram illustrates the namespace conflict between two packages, Pkg1 and Pkg2.

Pkg1 Structure:

- Pkg1 contains a folder named "folder1".
- "folder1" contains a service named "svc1".

Pkg2 Structure:

- Pkg2 contains a folder named "folder2".
- "folder2" contains a service named "svc2".
- Pkg2 also contains a folder named "folder1".
- "folder1" in Pkg2 contains a service named "svc2".
- "folder1" in Pkg2 contains a service named "svc1", which is highlighted with a red circle and a slash over it, indicating it cannot be created due to a conflict.

A horizontal arrow labeled "Services collide" points from the Pkg2 section to a box titled "Namespace:".

Namespace:

- folder1:svc1
- folder1:svc1** (highlighted in red)
- folder1:svc2
- folder2:svc2

Software AG Training | 2 - 12

The above diagram illustrates the Namespace:

- Pkg1 and Pkg2 do not appear in the Namespace at all
- folder1 in Pkg1 and folder1 in Pkg2 merge, i.e. svc1 and svc2 reside in the same namespace folder
- folder1:svc1 in Pkg2 cannot be created as it will conflict with folder1:svc1 in Pkg1

Namespaces and Designer

- File | New | Package
- File | New | Folder

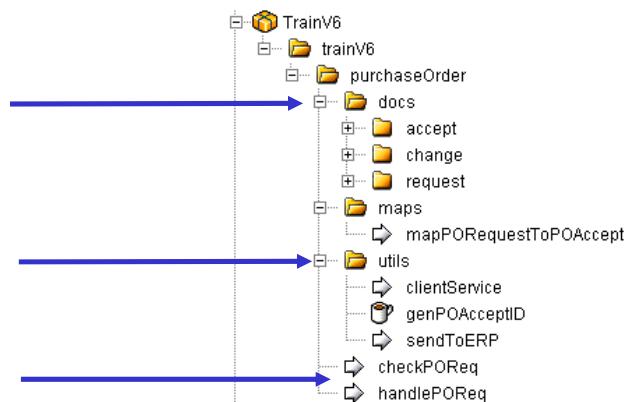
The screenshot shows the 'Package Navigator' window in the webMethods Integration Workshop. The tree view on the left shows a 'Default' package containing an 'Acme' folder, which contains several sub-folders like 'ff', 'Pu', and 'Pv'. A context menu is open over the 'acme' folder, with the 'New' option highlighted. The menu includes options like 'Expand', 'Collapse', 'Find...', 'Open Administration View', 'Refresh', 'Copy', 'Cut', and 'Paste'. To the right of the menu, a list of new item types is displayed: Document Type, Flow Service, Folder, JMS Trigger, Java Service, Package, and Web Service Descriptor.

Software AG Training | 2 - 13

Package Structure Recommendations

- Avoid namespace conflicts by nesting the primary folder.
- Make all of your packages consistent in structure!

Document types



Supporting services

Primary services



3

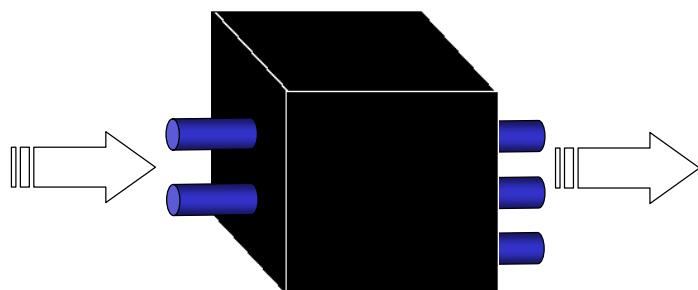
Introduction to Services

Objectives

- At the end of this section, you will be able to:
 - define the Service Pipeline
 - list the Pipeline data objects
 - explain how Services use the Pipeline
 - document your services

The Service Unit

- Each Service is like a black box:
 - Receives data**
 - Performs an operation**
 - Returns data**
- Any other Services only see declared inputs and outputs.



Steps for Building a New Service

- Identify Namespace location
- Create a new Service and give it a name
- Specify inputs and outputs
- Insert operation steps or code
- Map pipeline data (if flow service)

The screenshot shows the Software AG webMethods 8 Integration Workshop interface. On the left, there is a tree view of service components under the 'Acme' namespace, including 'acme', 'ff', 'PurchaseOrder' (containing 'adapters', 'docs', 'maps', 'notifiers', 'utils', and 'work'), and a local workspace with files 'a' through 'l'. A context menu is open over the 'work' folder, with 'Flow Service' highlighted. To the right of the tree view is a 'Document Type' dropdown menu listing various service types: Flow Service, Folder, JMS Trigger, Java Service, Package, and Web Service Descriptor.

software AG

Flow Pipeline Data

Value	Keyname	Datatype			
USD	currency	String			
0 1 2 3	USD CDN YEN EURO	currencies			
0 1 2 3	USD CDN YEN EURO	1 1.48 106 0.94	currencyTable	StringTable	
1999 12 08	Year Month Day	orderDate	Document		
0 1 2	1998 1992 1989	06 06 07	17 27 19	birthdays	DocumentList



Software AG Training | 3 - 5

currency is a String data type with a value of "USD"

currencies is an array of Strings, or a String List (0 based array due to Java using 0 based arrays)

currencyTable is a two-dimensional array of Strings, or a String Table

orderDate is a Document, or a structure containing an arbitrary set of other data structures or elements

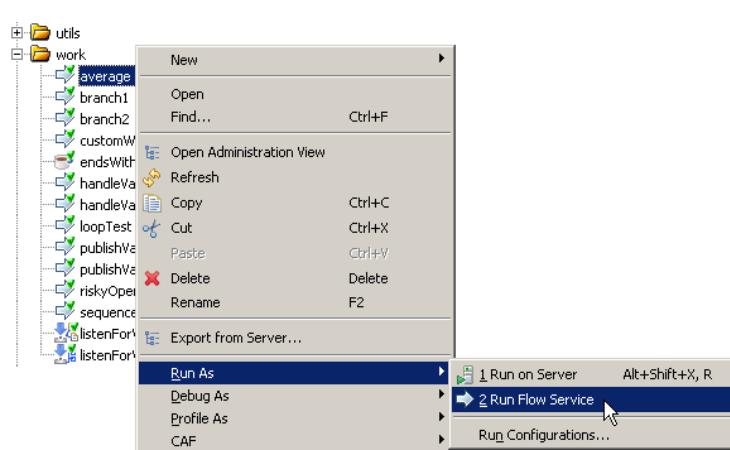
birthdays is an array of Documents, or a Document List

A two-dimensional array of Documents can be created by creating a Document List of Document Lists

Information about supported data types can be found in the the DeveloperUsersGuide.

Invoking a Service in Designer (run)

- Run
 - invokes the service in Designer
 - requests inputs via dialog



The screenshot shows the webMethods Designer interface. A context menu is open over a service named 'average' in the project tree. The menu path is: average > Run As > 2 Run Flow Service. Other options in the Run As submenu include 'Run on Server' and 'Run Configurations...'. The menu also includes standard options like New, Open, Find..., Copy, Cut, Paste, Delete, Rename, and Export from Server....

Invoking a Service in Designer (debug)

- Debug
 - Single step through a service
 - Step into / Step over

The screenshot shows the webMethods Designer interface. On the left is a tree view of workspace contents under 'work'. A right-click context menu is open over a service named 'average'. The menu path is: 'average' > 'Debug As' > '2 Debug Flow Service'. The menu items include:

- New
- Open
- Find... Ctrl+F
- Open Administration View
- Refresh
- Copy Ctrl+C
- Cut Ctrl+X
- Paste Ctrl+V
- Delete Delete F2
- Rename F2
- Export from Server...
- Run As
- Debug As
 - 1 Debug on Server Alt+Shift+D, R
 - 2 Debug Flow Service
 - CAF
- Lock for Edit Ctrl+L

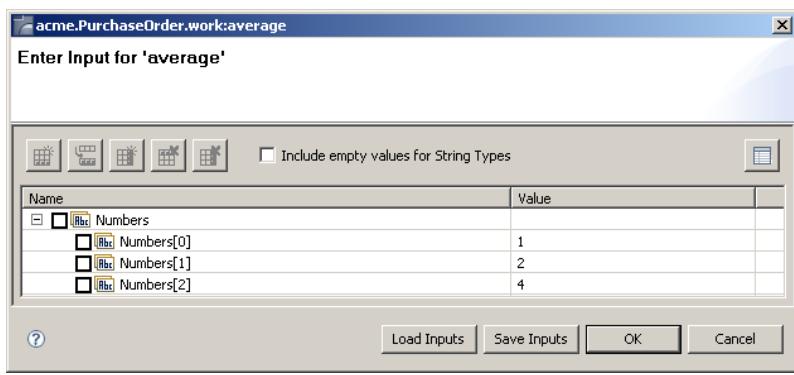
At the top of the screen, the Software AG logo is visible.

Invoking Services

- Client establishes a connection to the Integration Server via applicable protocol.
- Client is authenticated by the Integration Server.
- Client identifies service to be invoked via namespace and name.
- Client transmits service arguments to the IS.
- Integration Server:
 - instantiates a Pipeline
 - populates it with the inbound data
 - invokes service
 - returns the results to client
 - destroys the pipeline

Testing with Input Data

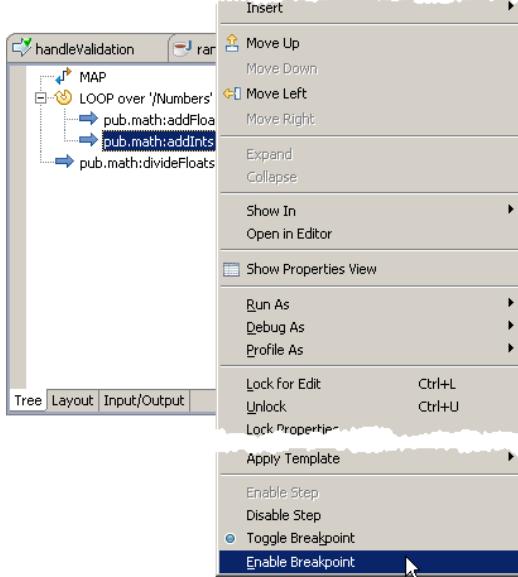
- How to test Services with data?
 - run the Service
 - type input data (data can be saved/restored using a text file)



Software AG Training | 3 - 9

Tools for Debugging

- Breakpoints
 - right-click an operation to set a breakpoint where execution will stop at runtime
- Disabling/Enabling steps
 - right-click on operations and Enable/Disable step
- Call Stack
 - to keep track of state of execution

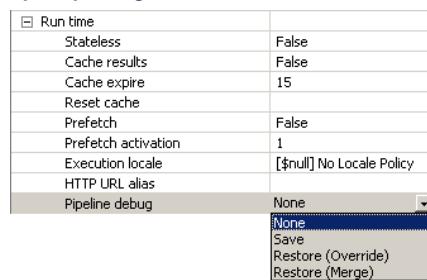


The screenshot shows the 'handleValidation' process tree. A context menu is open over the 'pub.math:addInts' step, with the 'Enable Breakpoint' option highlighted.

Software AG Training | 3 - 10

Debugging - Useful Built-In services

- Helpful services found in the WmPublic package:
 - pub.flow:debugLog
 - pub.flow:savePipelineToFile
 - pub.flow:restorePipelineFromFile
 - pub.flow:clearPipeline
- Instead of {save/restore}Pipeline{To/From}File you can use the “Pipeline debug” property of a service.



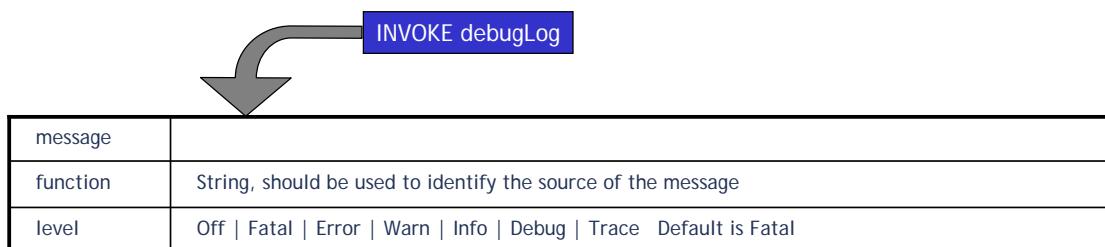
Built-in Services - Using Pipeline Services

- **pub.flow:savePipelineToFile** and **restorePipelineFromFile**
 - Saves the pipeline to a file, or restores pipeline from a file, respectively.
 - Both contain an input parameter **fileName**. By default, this file will be created in the Integration Server's pipeline directory.
- **pub.flow:clearPipeline**
 - Clears the current content of the pipeline.
 - It takes as an argument a String List of pipeline variables that should not be cleared.

Built-in Services - Using debugLog

- **pub.flow:debugLog**

- By default writes to ...\\IntegrationServer\\logs\\server.log



Before Changing a Service - Dependencies & References

- Changing or deleting a service or document type can affect any service that references it - check first!

The screenshot shows the webMethods IDE interface. In the top navigation bar, the Software AG logo is visible. Below the navigation bar, there is a tree view of project structure under the 'docs' folder, including 'request', 'Error', 'OrderCanonical', and 'Validation'. A context menu is open over the 'OrderCanonical' item, with 'Find Dependents' highlighted. An orange arrow points from this menu to a search results window titled 'Search'. The search results show '4 dependents found for 'acme.PurchaseOrder.docs:OrderCanonical''. The results list includes:

- 'acme.PurchaseOrder.docs:OrderCanonical' is used by 4 elements
 - + Acme/acme.PurchaseOrder.maps:orderRequestToCanonical
 - + Acme/acme.PurchaseOrder.utils:insertOrderCanonical
 - + AcmeOrder/AcmeOrder.OrdersInbound.OrdersInbound_1.Default:Map_to_Canonical
 - + AcmeOrder/AcmeOrder.OrdersInbound.OrdersInbound_1.Default:Send_to_ORMS

Software AG Training | 3 - 14

Cooperative Development Features

- Multiple developers working together on a single IS
- Built-in Features:
 - Object Locking and Unlocking
- On Your Own:
 - External file locking, source code control systems, etc.
 - Procedural systems
- NOTE: Cooperative Development means Coding Together.
These are NOT Security Features.

COOPERATIVE!



The Lock Commands

- Lock assigns editing control of the object to you. While you have the lock on the object, no one else can edit that object.

The Lock commands may be found on the File menu or by right-clicking an Editable object

Software AG Training | 3 - 16

Lock/Unlock States

- At any time, an object will be in 1 of 3 states.
- Locked by you: (green check on the right)
 - any developer can run the object
 - only you can edit the object
 - only you or an Administrator can unlock the object
- Locked by someone else (DeveloperX): (red check on the left)
 - any developer can run the object
 - you can't change the object
 - only DeveloperX or an Administrator can unlock the object
- Unlocked:
 - any developer can run the object
 - no one can edit the object
 - any developer can lock the object



Hierarchy Locking and Unlocking

- Select a container object like a Package or Folder, right-click and pick the Lock for Edit command, and all contained objects will be locked.

The screenshot shows the Software AG webMethods IDE interface. On the left is a tree view of project structure under 'utils'. A folder named 'work' is selected and highlighted with a blue border. A context menu is open over this folder, listing the following options:

- Lock for Edit (Ctrl+L)
- Unlock (Ctrl+U)
- Lock Properties...
- Find Dependents (Ctrl+Shift+D)

An orange arrow points from the 'Lock for Edit' option to the right. To the right of the menu, the tree view shows the contents of the 'work' folder, which now includes several items with green checkmarks next to them, indicating they are locked. The status bar at the bottom right of the interface displays the text "Software AG Training | 3 - 18".

Administrator Override

- Users with Administrative privileges can override the lock.
Non-admin users will fail.

Admin privileges

Non-admin

Error

The following elements were not unlocked:

Acme/acme.PurchaseOrder.work:average

Acme/acme.PurchaseOrder.work:average element is locked by owner:
Administrator, host: 127.0.0.1, on March 24, 2010 12:19:56 PM CET

OK

The screenshot shows the Software AG webMethods Integration Workshop interface. On the left, there is a file tree with the following structure:

- utils
- work
 - average
 - branch1
 - branch2

A red arrow points from the 'average' folder in the tree to the 'average' service in the main window. The main window title is 'average' and shows the following details:

- Server: localhost:5555
- Package: Acme
- Folder: acme.PurchaseOrder.work

The service overview shows:

- Service Input**: Numbers
- Service Output**: Average, Sum, Count
- Flow Overview**:

```
1 of MAP
2 LOOP OVER '/Numbers'
  2.1 → INVOKE addFloats
  2.2 → INVOKE addInts
3 → INVOKE divideFloats
4 → SEQUENCE
```

Software AG Training | 3 - 20

4

Document Types

Objectives

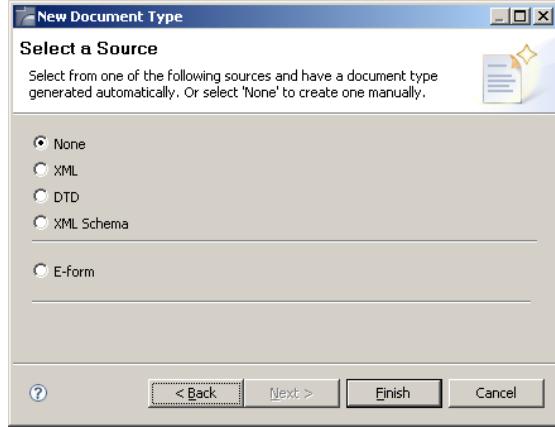
- At the end of this section, you will be able to:
 - create Document Types
 - reference Document Types in the Pipeline
 - discuss options for Document Type Validation

Documents and Document Types

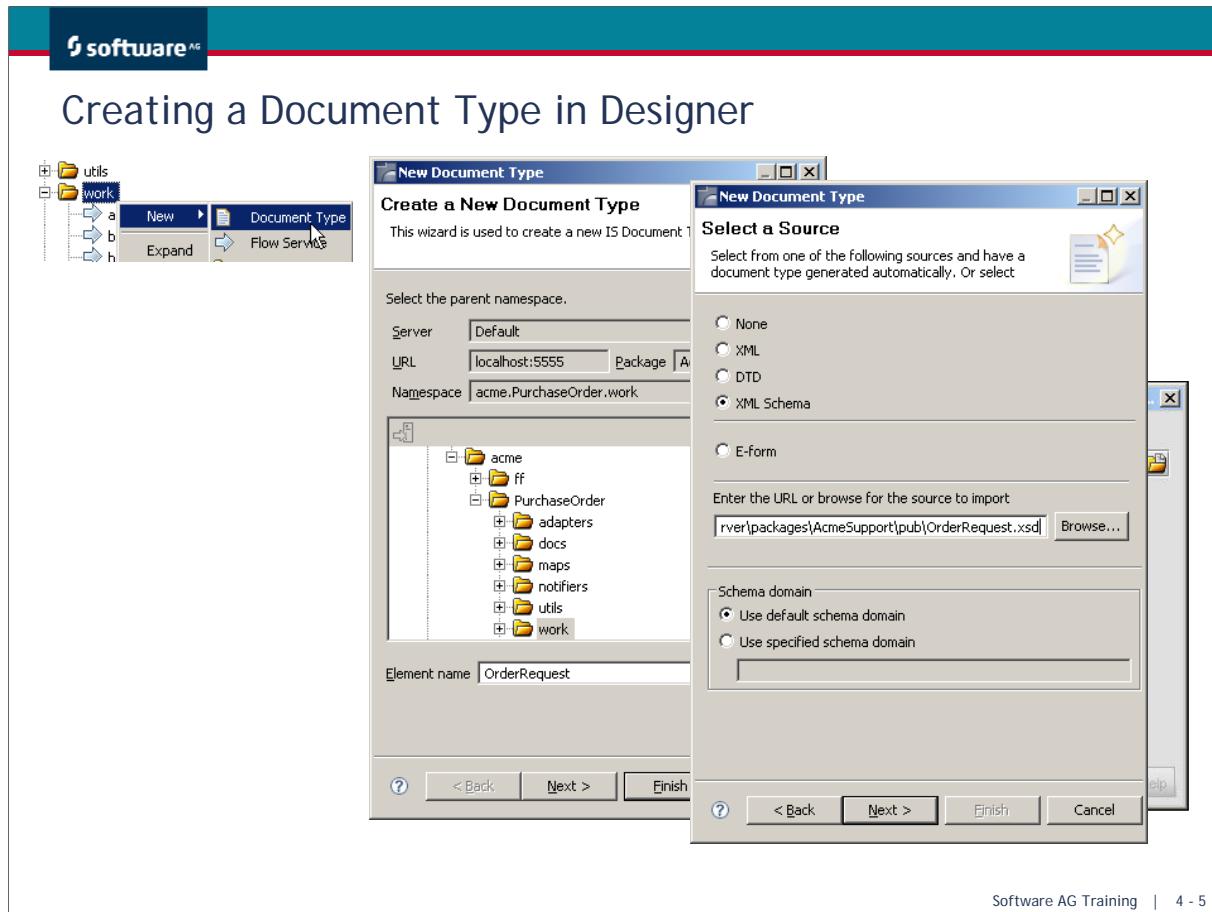
- A Document is a structured message (containing data) that flows through the Integration Server system.
- A Document Type is the definition of a document - it contains meta information about the fields that a document contains.

Document Types & XML

- The Integration Server is designed to manipulate XML easily.
 - Documents in Designer can be thought of as parsed XML.
- Document Types:
 - represent the document structure
 - are not XML, DTDs or XML Schemas
 - can be created:
 - manually
 - from an example XML document
 - from an XML DTD
 - from an XML Schema
 - from document types defined on the Broker



Software AG Training | 4 - 4

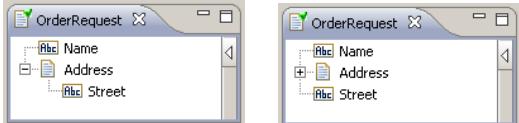
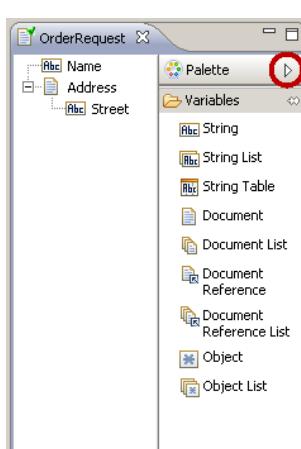


Software AG Training | 4 - 5

Manual Creation of Document Types

- Choose “None” as a source.
- Create new variables using a right click ➔ Insert
- Alternatively activate the palette and use drag and drop
- Use the arrows to move individual entries.    

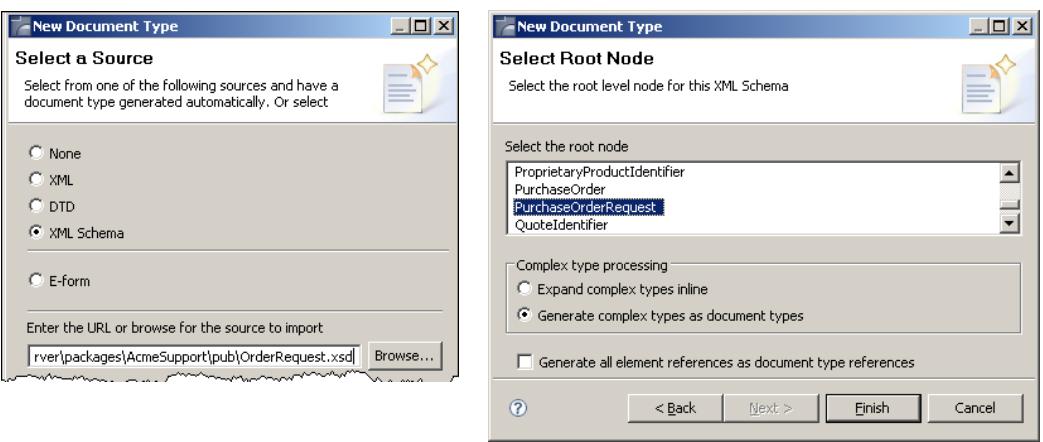
Be aware of indentation! Compare below:



Software AG Training | 4 - 6

Automatic Creation Using Schemas

- Designer allows the creation of complex types as document types.
- You must tell the wizard which element of the Schema is the Root node. You must understand the document structure before creating document types!



The image displays two windows from the 'New Document Type' wizard:

- Select a Source:** This window allows you to choose a source for generating a document type. The 'XML Schema' option is selected. A URL is entered in the 'Enter the URL or browse for the source to import' field: 'rver\packages\AcmeSupport\pub\OrderRequest.xsd'. There is also a 'Browse...' button.
- Select Root Node:** This window lets you select the root level node for the XML Schema. 'PurchaseOrderRequest' is selected as the root node. Under 'Complex type processing', the 'Generate complex types as document types' option is selected. There is also a checkbox for 'Generate all element references as document type references'.

Constraints (Treeview)

- Automatically created when using DTDs or Schemas as Document source.

The screenshot shows a software interface titled "Test" with a tree view of document structure. The structure includes nodes like OrderNumber, OrderDate, ShipDate, ReturnDate, and Address. Address further contains Name, Street1, Street2, City, ZIP, and State. Two specific nodes are highlighted with red circles: OrderNumber and Street1. A large blue arrow originates from the "Optional" constraint at the bottom left and points to the Street1 node. Another blue arrow originates from the "Constrained" constraint at the bottom right and points to the OrderNumber node.

Optional

Constrained

Software AG Training | 4 - 8

Constraints (Properties)

- All constraints are reflected in the properties of a variable

This field is required, but it can be <null> and must be formatted as a date

Property	Value
+ General	
Constraints	
Required	True
Allow null	True
Allow unspecified fields	True
Content type	date {http://www.w3.org/2001/XMLSchema}
Java wrapper type	UNKNOWN

This field is optional and it can be <null>. There are no formatting restrictions.

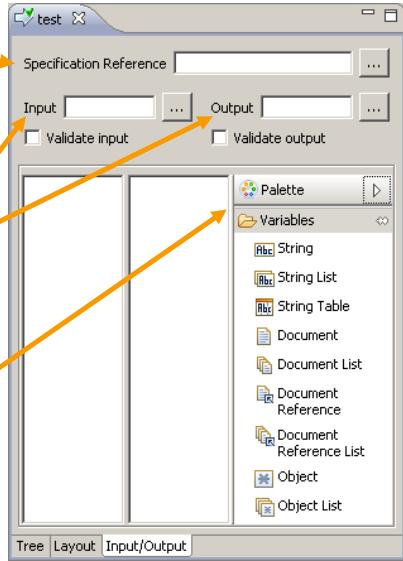
Property	Value
+ General	
Constraints	
Required	False
Allow null	True
Allow unspecified fields	True
Content type	
Java wrapper type	UNKNOWN

Document Validation

- Document Type constraints:
 - imported from DTD/Schema contents
 - added manually in Developer
- You can validate documents multiple ways:
 - “validate inputs/outputs” option on the Input/Output tab
(This is not recommended because it is harder to catch the error and process it.)
 - built-in validation services (pub.schema:validate)
 - Trading Networks built-in validation
 - custom validation services

Using Documents: Declaring Inputs & Outputs of Services...

- Declare Service Input / Output in one of three ways:
 - via Specification Reference text box
 - separate Specification object in the Namespace
 - allows multiple services to share a standard declaration
 - Input/Output values may not be edited
 - via Input or Output text boxes
 - via the input Palette



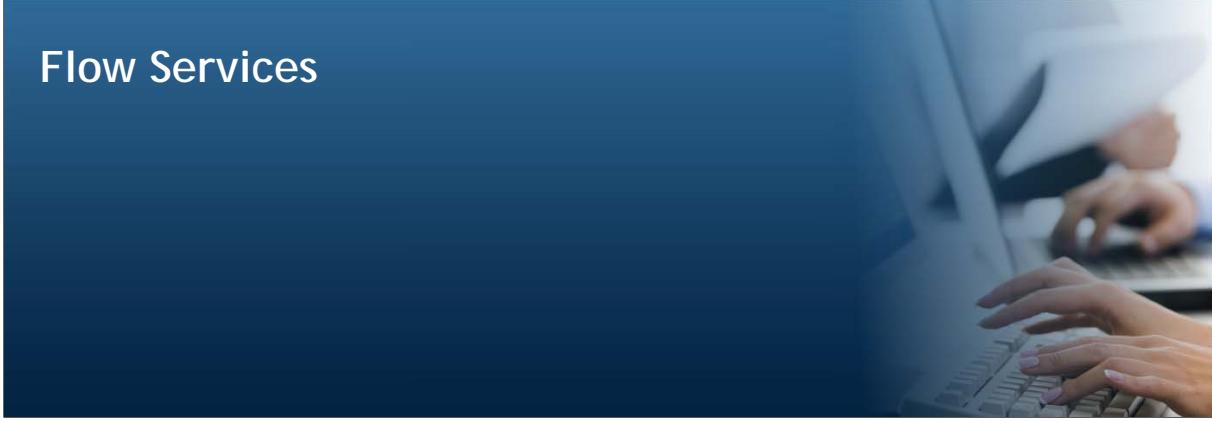
Software AG Training | 4 - 11

This page intentionally left blank.

Software AG Training | 4 - 12

5

Flow Services

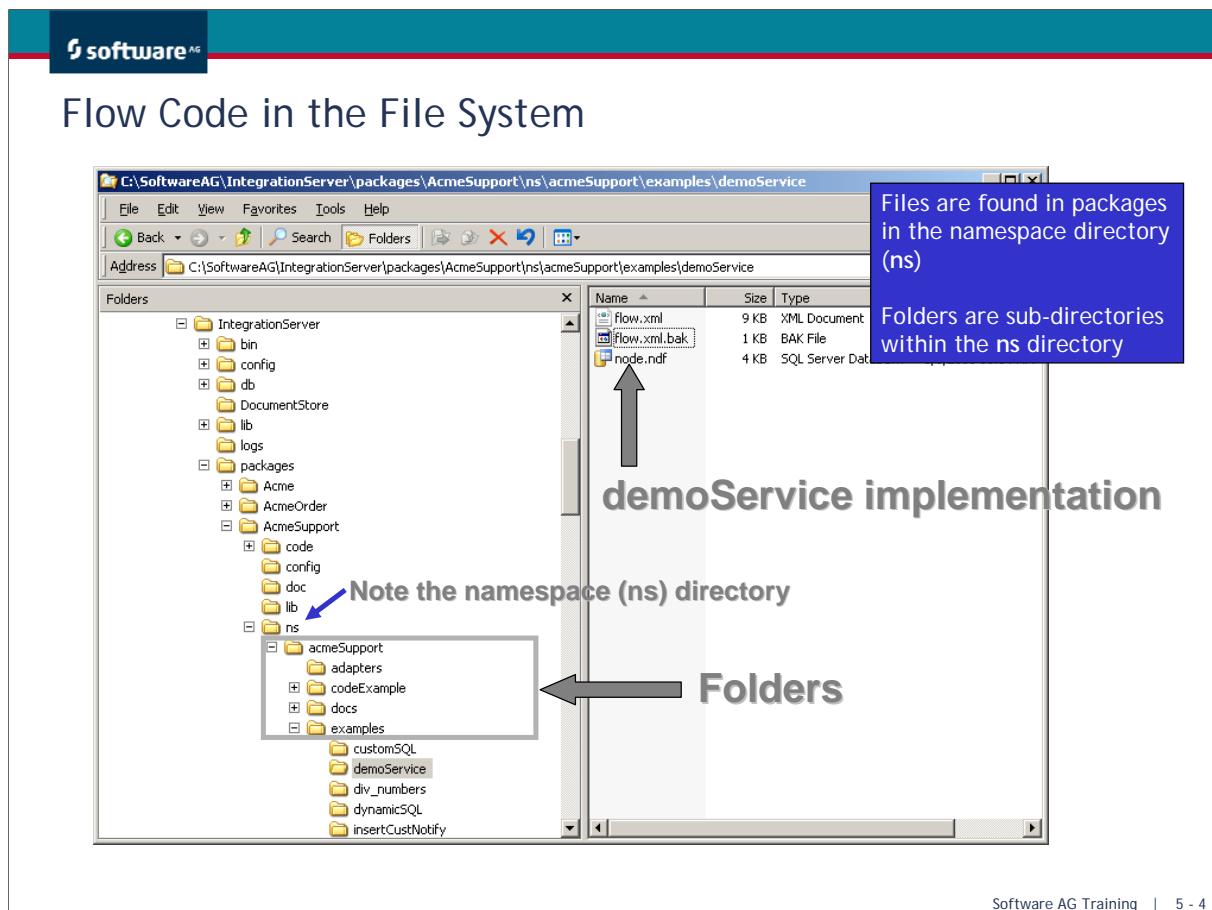


Objectives

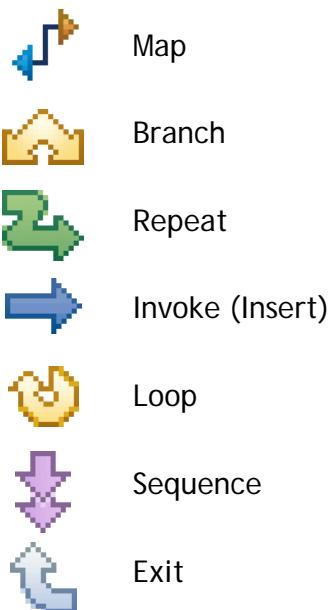
- At the end of this section, you will be able to:
 - describe the capabilities of webMethods Flow
 - list the seven Flow operation types
 - run your Flow services

Flow

- Flow is a webMethods graphical language for defining services
- Flow Services & Statements are:
 - interpreted and executed at run-time
 - stored in XML format



Flow Steps



Map Example (More Later)

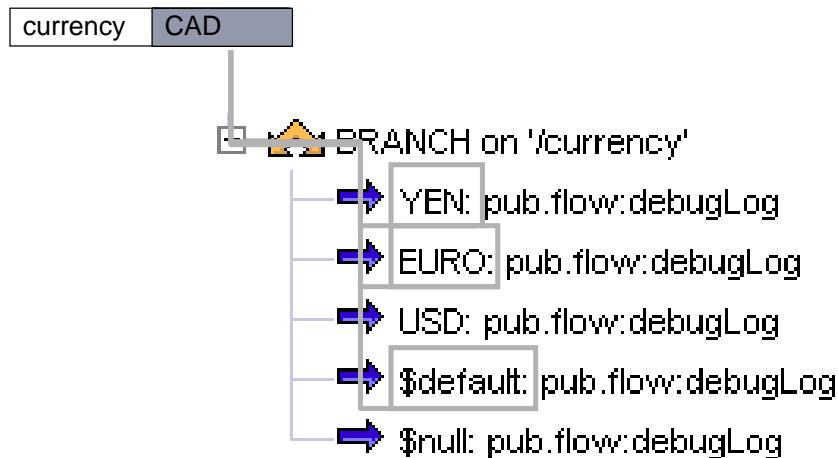
- Link fields in one of two ways:
 - select source field, hold down left button and drag it to the target field
 - select both fields and click the link button

The screenshot shows the 'Pipeline' window in the webMethods IDE. On the left, under 'Pipeline In', there are variables: \$abc:Numbers, \$abc:Count, \$abc:Sum, and \$abc:\$iteration. In the center, a service component named 'addInts' is connected between 'Service In' and 'Service Out'. The 'Service In' port has two variables: \$abc:num1 and \$abc:num2. The 'Service Out' port has one variable: \$abc:value. On the right, under 'Pipeline Out', there are variables: \$abc:num1, \$abc:num2, \$abc:Numbers, \$abc:Count, \$abc:Sum, and \$abc:value. A tooltip 'Create a Link between the Selected Variables' is visible above the connection line between \$abc:value in the Service Out and \$abc:value in the Pipeline Out. The connection line starts at the \$abc:value output of the addInts service and ends at the \$abc:value input in the Pipeline Out list.



Basic Branch - Using a Switch value

- Chooses a child step based on the value of a variable in the Pipeline.



Basic Branch Properties

Property	Value
General	
Comments	
Scope	
Timeout	
Label	
Switch	/currency
Evaluate labels	False

Remember to indent!

Comments	Free-text comment field PLEASE USE!	description of step
Scope	Which Document to restrict the operation to	e.g. Document/DocumentElement
Timeout	# of seconds before timeout	0, blank, or integer value
Label	Name of this instance of branch	string variable, \$null, \$default, blank
Switch	Decision variable. Do not set when evaluate-labels = true	string variable, e.g. Document/DocumentElement
Evaluate labels	If set, labels will be evaluated instead of the switch variable	True, False

software AG

Advanced Branch - Using Expressions

```
graph TD; currency[Currency: CAD] --> branch((BRANCH)); branch -- "%currency% = 'YEN' || %currency% = /[UC]/" --> log1[$pub.flow.debugLog]; branch -- "$default:" --> log2[$pub.flow.debugLog]
```

The diagram illustrates a webMethods integration flow. It starts with a variable assignment step where 'currency' is set to 'CAD'. This leads into a 'BRANCH' step. The 'BRANCH' step has two outgoing paths. The top path is triggered by the condition '%currency% = 'YEN' || %currency% = /[UC]/'. This path leads to an action step '\$pub.flow.debugLog'. The bottom path is triggered by the condition '\$default:'. This path also leads to an action step '\$pub.flow.debugLog'.

The screenshot shows the webMethods studio interface. On the left, a process tree window titled "validateService" is open, showing a "BRANCH on '/currency'" node with three branches: "%currency% = 'YEN': pub.flow:debugLog", "%currency% = /^[UF]/: pub.flow:debugLog", and "\$default: pub.flow:debugLog". A blue callout box labeled "Expression to test at runtime" points to the "%currency%" expression in the first branch. On the right, a "Properties" dialog box is open, showing the "General" tab with the "Evaluate labels" checkbox checked. A blue arrow points from the "Evaluate labels" row in the properties table to this checkbox.

Property	Value
General	
Comments	Free-text comment field PLEASE USE!
Scope	Which Document to restrict the operation to e.g. Document/DocumentElement
Timeout	# of seconds before timeout 0, blank, or integer value
Label	Name of this instance of branch string variable, \$null, \$default, blank
Switch	Decision variable. Do not set when evaluate-labels = true string variable, e.g. Document/DocumentElement
Evaluate labels	If set, labels will be evaluated instead of the switch variable True, False

Software AG Training | 5 - 10



Using Regular Expressions

- Refer to the Developer Users Guide, which is stored in folder _documentation/Developer and is called 7-1_Developer_Users_Guide.pdf
 - Appendix B: Regular Expressions
 - A regular expression is a pattern-matching technique. It is common to use such an expression to specify the switch value for a branch step.
 - Examples:
 - /`^webMethods/` evaluates to true if the value starts with "webMethods"
 - /`part +555-A/` evaluates to true if the value contains the word "part" followed by one or more spaces and then the characters "555-A"
 - /`part[0-9]*555-A/` evaluates to true if the value contains the word "part" followed by zero or more numbers and then the characters "555-A"
 - /`[abc]/` evaluates to true if the value contains "a" or "b" or "c"

software AG

Repeat

- Any Repeat can specify a max Count.
- Repeat is useful for iterating a number of times.
- Iterations = max Count + 1.

The diagram shows a 'REPEAT' node with a green arrow icon. A callout bubble above it says 'Count = 3'. Two blue arrows point from the node to external components: 'pub.client:http' and 'pub.flow:debugLog', each followed by three green checkmarks.

Count = 3

REPEAT

pub.client:http ✓ ✓ ✓

pub.flow:debugLog

Software AG Training | 5 - 12

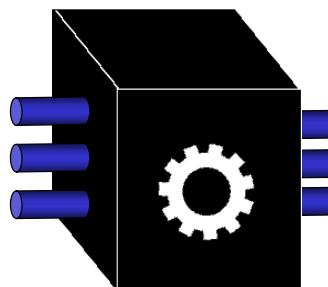
The current repeat count is stored in a field called `$retries` (*null until Repeat begins*)

Property	Value
General	
Comments	
Scope	
Timeout	
Label	
Count	
Repeat interval	
Repeat on	FAILURE

Invoke



- Performed for each service inserted into a flow.
- Invokes a previously built service.
- Allows service chaining for complex flows.
- Leverages modularity in design.



Software AG Training | 5 - 14

software AG

Loop

LOOP over '/Employee/Children'

- MAP
- pub.string:toUpperCase
- pub.flow:debugLog

```
graph TD; Start(( )) --> Decision{More elements in Children array?}; Decision -- No --> End(( )); Decision -- Yes --> Map[MAP]; Map --> ToUpper[toUpperCase]; ToUpper --> DebugLog[debugLog]; DebugLog --> Decision;
```

The flowchart illustrates a loop structure. It begins with a decision diamond labeled "More elements in Children array?". If the answer is "No", the process ends. If the answer is "Yes", it proceeds through a series of three rectangular boxes: "MAP", "toUpperCase", and "debugLog". After the "debugLog" box, the flow returns to the "More elements in Children array?" decision diamond.

Software AG Training | 5 - 15

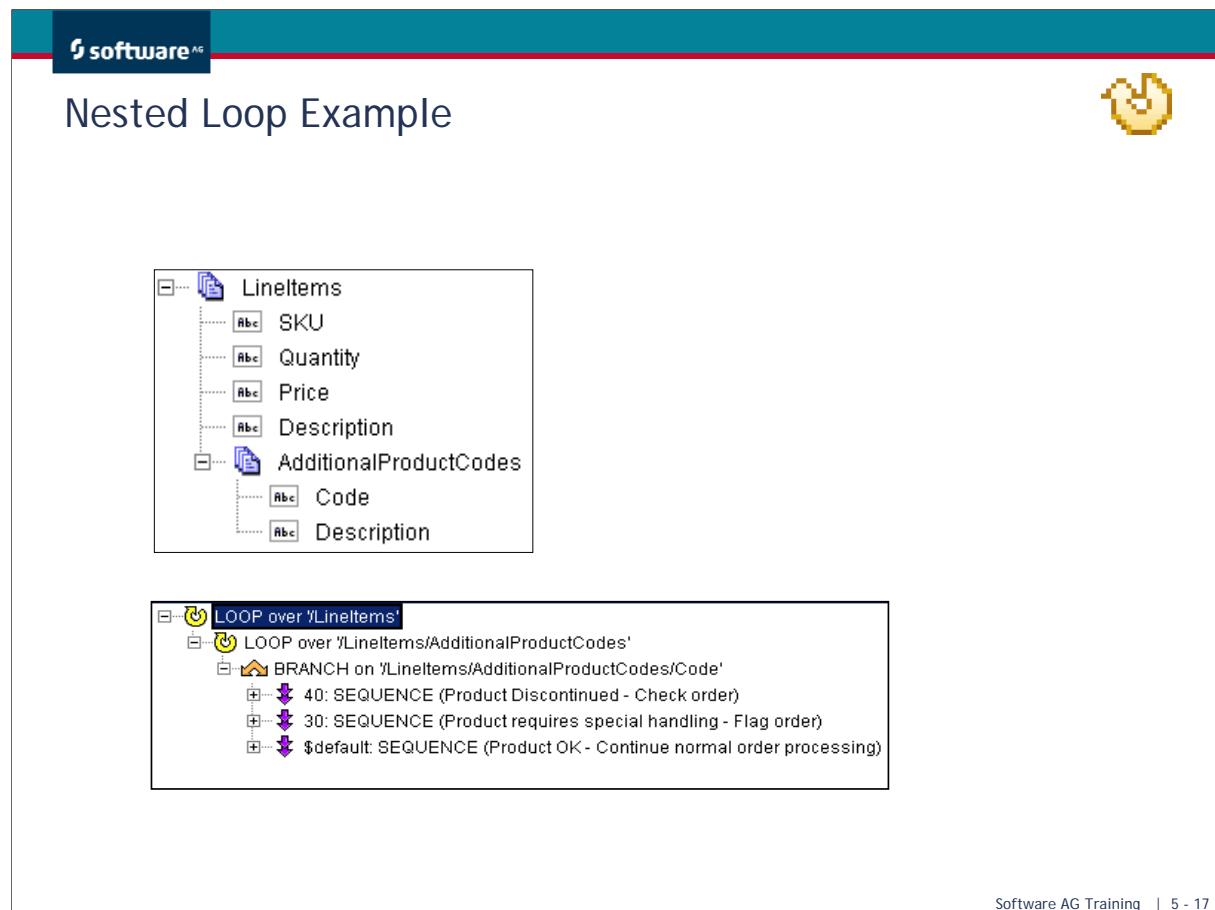
The screenshot shows the webMethods Studio interface. On the left is a pipeline editor window titled "numChildren" containing a "LOOP over '/Employee/Children'" step, which has a "MAP" step inside it. Below the pipeline editor are tabs for "Tree", "Layout", "Input/Output", and "Properties". On the right is a "Properties" dialog box with a tree view under "General" and a table with the following rows:

Property	Value
Comments	Free-text comment field PLEASE USE!
Scope	Which Document to restrict the operation to
Timeout	# of seconds before timeout
Label	Name of this instance of loop
Input array (required)	Array data
Output array	Name of output array variable

A blue arrow points from the "Input array" row in the properties table to a callout box containing the text: "The current Loop count is stored in a field called \$iteration". Another blue arrow points from the "Output array" row in the properties table to the same callout box.

Software AG Training | 5 - 16

Nested Loop Example



```
graph TD; Start(( )) --> L1[LOOP over 'Lineltems']; L1 --> L2[LOOP over 'Lineltems/AdditionalProductCodes']; L2 --> Branch[BRANCH on 'Lineltems/AdditionalProductCodes/Code']; Branch -- 40 --> S1[SEQUENCE (Product Discontinued - Check order)]; Branch -- 30 --> S2[SEQUENCE (Product requires special handling - Flag order)]; Branch -- $default --> S3[SEQUENCE (Product OK - Continue normal order processing)];
```

Software AG Training | 5 - 17

Loop Consideration

The screenshot shows the 'Loop Consideration' example in the webMethods Integration Studio. At the top, there is a configuration panel for a 'LOOP over /CSVDT/recordWithNoID' node. The 'Input array' is set to '/CSVDT/recordWithNoID' and the 'Output array' is set to '/Customers'. Below this, the process flow diagram illustrates the mapping of fields from the 'recordWithNoID' schema to the 'Customers' schema. The 'Company', 'Street', 'City', 'State', and 'Zip' fields from the input schema map to their respective fields in the output schema. The 'isValid' field also maps to the 'isValid' field in the output schema. The 'errors' field from the input schema maps to the 'errors' field in the output schema. A callout box with the following text is overlaid on the diagram:

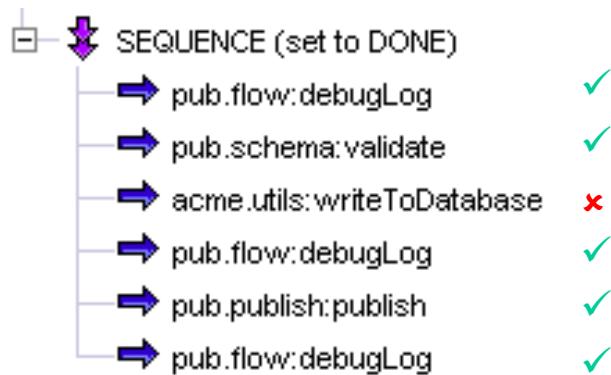
If you forget to set the **Output array**, this example will continuously overwrite the only instance of **Customers**.

Software AG Training | 5 - 18



Sequence - Exit on Done

- Ignores any failures (still written to error log).
- Mutually independent operations.



Sequence - Exit on Failure

- Regular behavior of services in a flow.
- No further Sequence operations are executed after a failure.

```
graph TD; Start(( )) --> Seq[SEQUENCE (set to FAILURE)]; Seq --> Log1[pub.flow:debugLog]; Log1 --> Log2[pub.schema:validate]; Log2 --> Fail(acme.utils:writeToDatabase); Fail --> Log3[pub.flow:debugLog]; Log3 --> Log4[pub.publish:publish]; Log4 --> Log5[pub.flow:debugLog]
```

SEQUENCE (set to FAILURE)

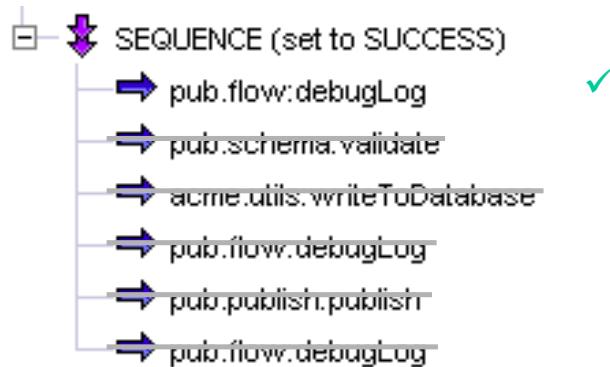
- pub.flow:debugLog ✓
- pub.schema:validate ✓
- acme.utils:writeToDatabase ✗
- pub.flow:debugLog
- pub.publish:publish
- pub.flow:debugLog

Software AG Training | 5 - 20



Sequence - Exit on Success

- Attempts operations until one is successful.
- No further Sequence operations will be completed.



Sequence Properties

The sequence diagram on the left shows a flow starting with `pub.art.transaction:startTransaction`, followed by a `SEQUENCE (success)` node which contains `trainV6.dbOps:insert`, `trainV6.dbOps:insert`, `trainV6.dbOps:update`, `pub.art.transaction:commitTransaction`, and another `SEQUENCE (done)` node containing `pub.flow:getLastError` and `pub.art.transaction:rollbackTransaction`.

The properties dialog on the right shows the following settings for the `SEQUENCE (success)` node:

Property	Value
General	
Comments	PLEASE USE!
Scope	Which Document to restrict the operation to
Timeout	# of seconds before timeout
Label	Name of this instance of sequence
Exit on (required)	SUCCESS

The table below lists the properties and their descriptions:

Comments	Free-text comment field PLEASE USE!	description of step
Scope	Which Document to restrict the operation to	Name of a document in the pipeline
Timeout	# of seconds before timeout	0, blank, or integer value
Label	Name of this instance of sequence	string variable, \$null, \$default, blank
Exit on (required)	condition	FAILURE, SUCCESS, DONE

Software AG Training | 5 - 22

Exit

```
graph TD; start[start: SEQUENCE] --> loop((LOOP over '/Partners')) --> branch((BRANCH on '/Partners/creditRating')) -- BAD --> exitBad((EXIT '$parent')) -- GOOD --> good1[GOOD: pub.client:smtp]; exitBad --> good1; exitBad --> log1[pub.flow:debugLog]; start --> log2[pub.flow:debugLog];
```

- Exit options:
 - Exit Loop (would exit entire loop if BAD credit found)
 - Exit Label (would exit to a specific label, such as start)
 - Exit Flow (would leave the entire flow if BAD credit found)
 - Exit Parent (would skip the BAD credit partner and continue)
 - Exit with Success
 - Exit with Failure and Reason

Software AG Training | 5 - 23

software AG

Exit Properties

Property	Value
General	
Comments	PLEASE USE!
Label	\$flow
Exit from	\$flow
Signal	FAILURE
Failure message	Line items less than 1 - invalid

Comments	Description
Free-text comment field PLEASE USE!	description of step
Label	Name of this instance of the exit operation string variable, \$null, \$default, blank
Exit from (required)	Which operation to exit from string variable
Signal (required)	Condition FAILURE, SUCCESS
Failure message	Text of exception message when signal is set to FAILURE Text, blank

Software AG Training | 5 - 24

6

Mapping

Objectives

- At the end of this section, you will be able to:
 - perform complex mapping
 - use transformers
 - understand Pipeline Variable Substitution

software AG

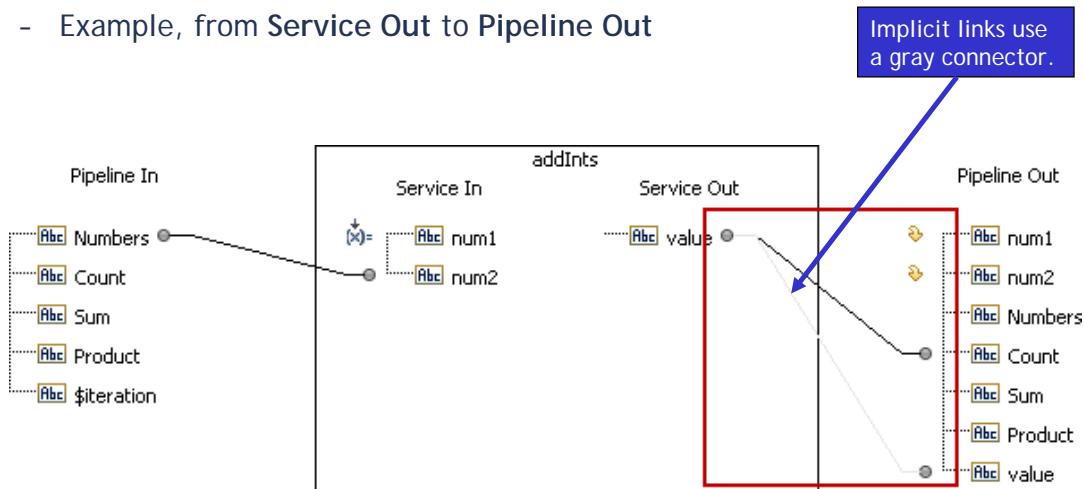
Basic MAP functionality

- Map statements are used:
 - to set values
 - to drop variables from the pipeline
 - to link data (copy) from one variable to another
 - to transform data during copying

```
graph LR; subgraph Pipeline_In [Pipeline In]; N1[Abc Numbers]; N2[Abc Count]; N3[Abc Sum]; N4[Abc $iteration]; end; subgraph Transformer [pub.math:addInts]; SIn[Service In] -- num1 --> SOut[Service Out]; SOut -- value --> C1((Abc Count)); end; subgraph Pipeline_Out [Pipeline Out]; N1; N2; N3; end;
```

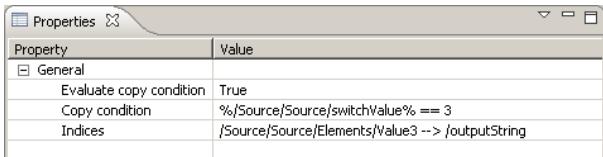
Implicit Mapping

- Within a flow, Developer implicitly connects, or maps, variables whose names are the same and whose data types are compatible
 - Example, from Service Out to Pipeline Out



Conditional Mapping

- You can place a condition on a link.
- Click on the link line and set the Copy condition property



Conditional links use a blue connector.

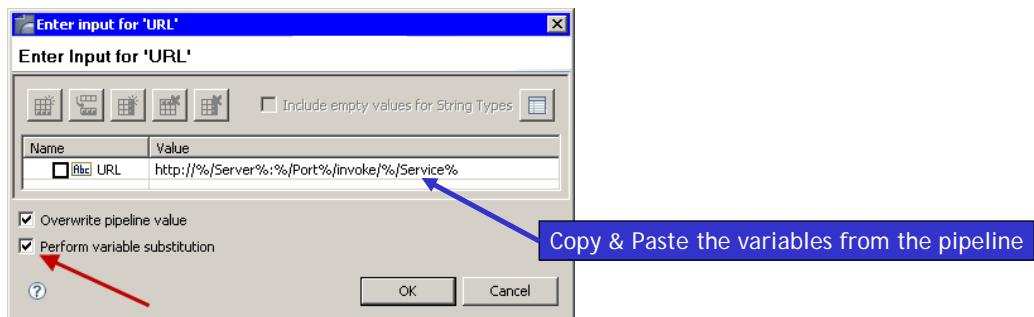
The diagram shows a pipeline mapping from a source node to a transformer node. The source node contains a 'Source' element with three 'Elements' under it: 'Value1', 'Value2', and 'Value3'. A blue connector links 'Value3' to an output port labeled 'outputString' on the transformer node. The pipeline is divided into 'Pipeline In', 'Transformers', and 'Pipeline Out' sections.

Software AG Training | 6 - 5



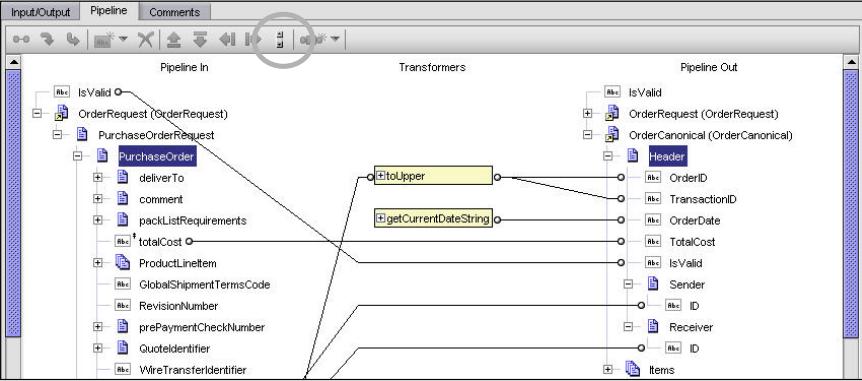
Mapping using Set Value

- The Integration Server can substitute variables with a Set Value:
 - check "Perform Variable Substitution"
 - identify a pipeline variable as %variable%
 - navigate Documents as /root/subnode/key (try copy/paste)
 - Can be used as 'poor mans printf'



Help Mapping Large Documents (DEVELOPER ONLY)

- Dual Scrolling enabled:
 - transformers scroll with the target tree scroll bar
 - if transformer is expanded, single scrolling mode is used until transformer is collapsed



Software AG Training | 6 - 7

The Find Variable Dialog Box (only in DEVELOPER)

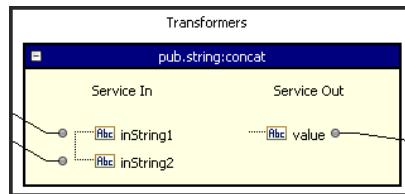
- Search is done on the pipeline tree with the focus.
- Search starts from selected node and moves downward.
- If node is not selected, search is done from top to bottom.

The screenshot shows the webMethods Integration Workshop interface. A 'Find Variable' dialog box is open in the center, titled 'Find Variable: Pipeline In'. The 'Find what:' field contains 'duns'. Below it are two checkboxes: 'Match case' and 'Match whole words only', neither of which is checked. At the bottom are 'Find Next' and 'Close' buttons. The background shows a pipeline tree with nodes like 'Pipeline In', 'Transformers', and 'Pipeline Out'. The 'PurchaseOrder' node under 'PurchaseOrderRequest' is selected. The 'Header' node under 'OrderCanonical' is also visible. The pipeline tree has nodes for 'IsValid', 'OrderRequest', 'PurchaseOrderRequest', 'PurchaseOrder', 'Header', 'OrderID', 'TransactionID', 'OrderDate', 'TotalCost', 'IsValid', and 'Sender'.



Transformers

- Transformers are any Service invoked within a Map operation.
- Transformers
 - Are executing independently
 - Do not cascade
 - Do not implicitly map pipeline variables
 - outputs are not automatically added to the pipeline
 - can be the most effective way to map - they do not copy and pass the whole pipeline



Mapping Indices

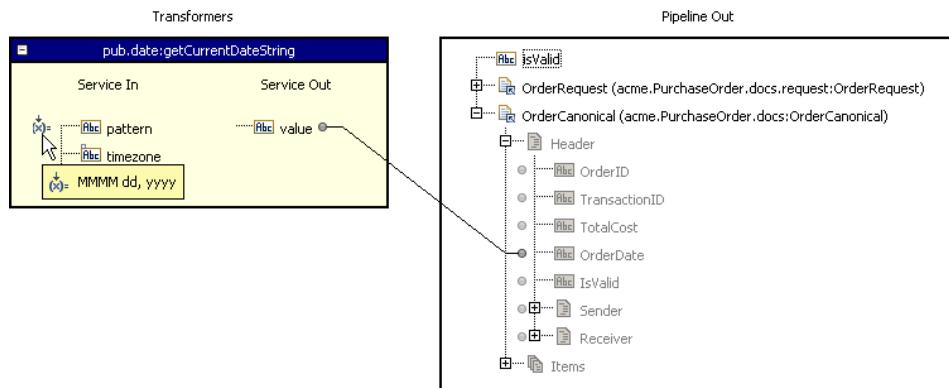
- You can map individual Elements of arrays. To do so, edit the Index Properties of the Input and Output variable:

The screenshot shows the 'Link Indices' dialog box and the pipeline editor. The 'Link Indices' dialog has two tables: 'Source' and 'Destination'. In the 'Source' table, the 'Field' column contains '/inputArray[%Index%]' and the 'Row Index' column contains '%Index%'. In the 'Destination' table, the 'Field' column contains '/outputArray[%Count% - %Index%]' and the 'Row Index' column contains '%Count% - %Index%'. Below the dialog is a portion of the pipeline editor showing 'Pipeline In', 'Transformers', and 'Pipeline Out' sections. A blue line connects the 'Out' port of the Pipeline In section to the 'In' port of the Pipeline Out section, indicating a mapped connection.

Software AG Training | 6 - 10

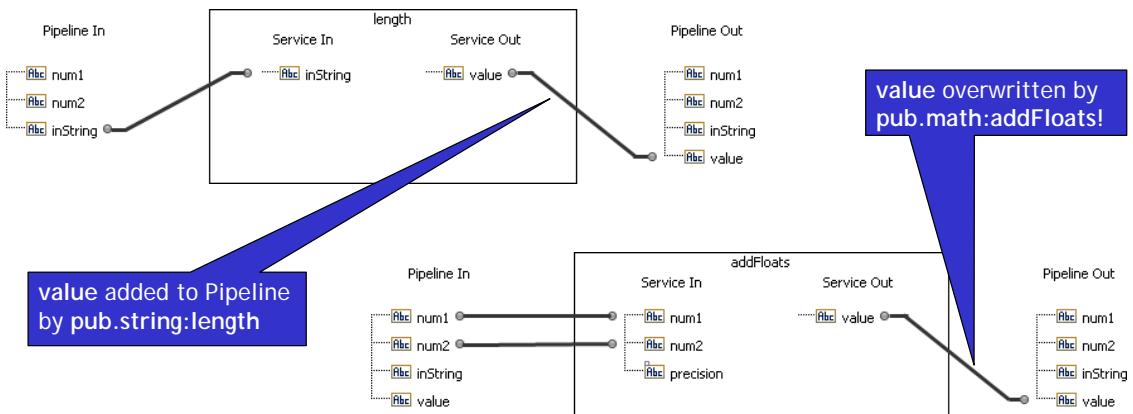
WmPublic Package Transformer Functions

- WmPublic contains many useful transformer services.
- Example from the pub.date folder: How to Insert the current date into a string field



The most common Mapping Error:

- Make sure you don't accidentally overwrite elements.



Inputs & Outputs

- Always map something to your declared outputs.
- Always drop unused fields.
- Declared inputs automatically appear in the Pipeline In of the first service step
- Declared outputs appear automatically in the Pipeline Out of the last step in the service. If you need them before, declare a Variable with the same name.

This page intentionally left blank.

Software AG Training | 6 - 14

7

Java Services

Objectives

- At the end of this section, you will be able to:
 - describe the capabilities of webMethods Java Services
 - write Java Services
 - understand the role IData plays in writing Java Services
 - understand how different Integration Server data structures are exposed to Java services
 - understand various ways of invoking Java services and generating Java code

Deciding What Kind of Service to Use

	Advantages	Disadvantages
Flow Services	<ul style="list-style-type: none">• Easy to create• No need to recompile after changes	<ul style="list-style-type: none">• Only permit use of existing services and Flow steps
Java Services	<ul style="list-style-type: none">• manipulate data directly• Better error handling• Make direct calls other APIs	<ul style="list-style-type: none">• Need to code and recompile after changes

JDK Options

- Integration Server ships with:
 - Sun's 1.5.0 Java Development Kit (JDK)
 - Sun's 1.4.2 Java 2 Software Development Kit (SDK)
- You will need to configure your environment to call the SDK.
 - PATH and CLASSPATH environment variables
 - Check Server...About in the Integration Server Administrator

The screenshot shows two windows side-by-side. On the left is a 'Software' configuration window with the following details:

Product	webMethods Integration Server
Version	8.0.1.0
Updates	IS_8-0_SP1 TNS_8-0_SP1
Build Number	209
SSL	Strong (128-bit)

Below it is a 'Server Environment' section with the following details:

Java Version	1.6.0_16 (50.0)
Java VM Name	Java HotSpot(TM) Client VM
Java Build Info	14.2-b01, mixed mode
Java Vendor	Sun Microsystems Inc.
Java Home	C:\SoftwareAG\jvm\win160\jre

On the right is a Windows File Explorer window titled 'C:\SoftwareAG\jvm'. It shows a folder structure with 'jvm' expanded to show 'win150' and 'win160' subfolders. The status bar indicates '2 objects (Disk free space: 38.3 GB)'.

Recap: Packages, Namespace, Services

■ Packages

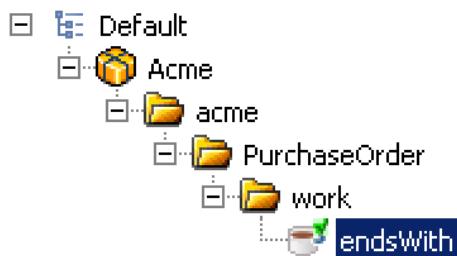
- logical containers for a set of services and related files
 - smallest deployable unit of software
 - exists as a single physical directory on the file system
 - not part of the Namespace
- should contain all objects and files required by the services
 - jars, templates, documentation, configuration files, etc.
- are not the same as Java packages

■ Services:

- are discrete functions or method calls
- can be written in Flow, Java
- organized and uniquely identified by the namespace

IS Packages vs. Java packages

- An Integration Server package (Acme) is similar to an IDE Project.
- The hierarchy of IS folders is similar to a Java package.
- The name of the Java class is the name service name concatenated with _SVC. **This was different in Version 7.**
- The service name is the name of the Java method.



```
package acme.PurchaseOrder.work;

import com.wm.data.*;□

public final class endsWith_SVC
{
    public static final void endsWith(IData pipeline)
    {
        IDataCursor cursor = pipeline.getCursor();
        String string = IDataUtil.getString(cursor, "string");
    }
}
```

Overview of Java Service Invocation

- Java Services are invoked just like any other Service.
- Like all services, a Java Service takes one, and only one, input variable - the Pipeline.
- Service receives the Pipeline as input and updates it.
 - The Service retrieves the input values it needs from the Pipeline.
 - A Service returns output by inserting it into the Pipeline.
- Multiple parallel invokes of a service manipulate different Pipeline objects.

The screenshot shows a Windows Internet Explorer window displaying the webMethods Java API Reference. The address bar shows the URL: C:\SoftwareAG\Documentation\Integration_Server\8-0-SP1_Integration_Server_Java_API_Reference\index.html. The left sidebar lists various Java files and packages. The main content area shows the **IData** interface definition:

```
public interface IData
```

The interface is described as defining the fundamental data abstraction used by the system.

Software AG Training | 7 - 8

Service Signature...

```
package acme.PurchaseOrder.work;

import com.wm.data.*;
import com.wm.util.Values;
import com.wm.app.b2b.server.Service;
import com.wm.app.b2b.server.ServiceException;

public final class endsWith_SVC

{
    public static final void endsWith(IData pipeline) throws ServiceException {
```

The code snippet shows a Java method definition. Four annotations are highlighted with blue callout boxes and arrows pointing to them:

- public**: can be invoked by other services
- static**: use a single (static) instance for all invokes - maximize throughput, minimize memory load
- final**: do not extend - improves performance
- void**: does not return an object

...Service Signature

```
package acme.PurchaseOrder.work;

import com.wm.data.*;
import com.wm.util.Values;
import com.wm.app.b2b.server.Service;
import com.wm.app.b2b.server.ServiceException;

public final class endsWith_SVC
```

```
{    public static final void endsWith(IData pipeline) throws ServiceException {
```

endsWith

The service name is the
method name

IData pipeline

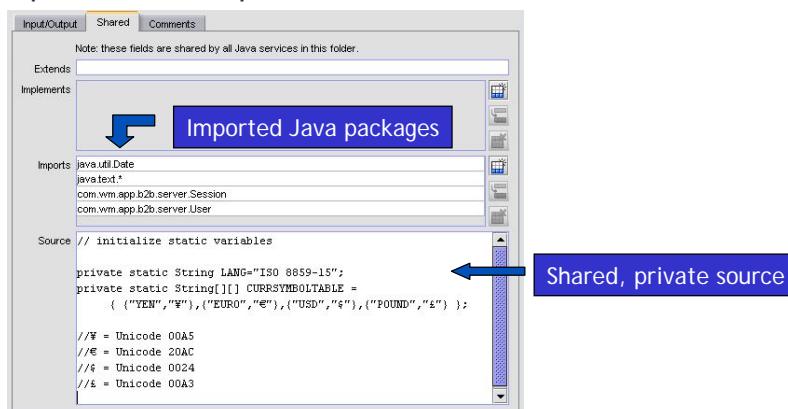
Takes IData object
named "pipeline" as
input

throws ServiceException

Failure on invoke throws
a ServiceException

The Shared tab (this feature is gone with Version 8)

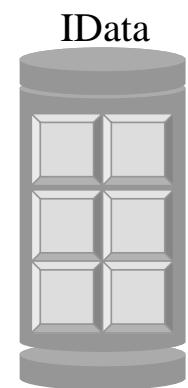
- Code common to all methods of the class:
 - i.e. common to all services in the folder
 - webMethods folder = Java class
- Define imports, shared private source, etc. here.



What is the Pipeline

- An **IData** Object:

- is an ordered collection of key / value pairs on which a service operates
- can contain any number of elements of any valid Java object, including other **IData** objects
- is instantiated by the server when the service is run
- is destroyed when service completes



software AG

What's in the IData Pipeline?

The diagram illustrates the mapping of various IData pipeline components to their corresponding Java and com.wm.data types. A central vertical bar separates the components from the mappings.

IData Component	Type Mappings
String	java.lang.String
String List	java.lang.String[] - a String array
String Table	java.lang.String[][] - array of arrays
Document	com.wm.data.IData
Document List	com.wm.data.IData[]
Document Reference	com.wm.IData
Document Reference List	com.wm.IData[]
Object	java.lang.Object (or any subclass)
Object List	Java.lang.Object[]

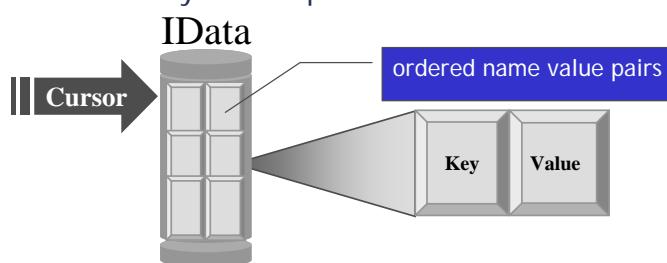
Software AG Training | 7 - 13

Useful Java Classes

- In order to manipulate the pipeline, the following Java Classes are available:
 - IData: The IData interface defines the fundamental data abstraction used by the webMethods Integration Server and its tools.
 - IDataFactory: This class provides methods for creating an IData object.
 - IDataUtil: This class provides various utility methods for populating, converting, copying, and merging the elements within an IData object.
 - IDataCursor: This interface defines the methods you use to access, traverse, and manipulate the contents of an IData object.

IData and Java Services

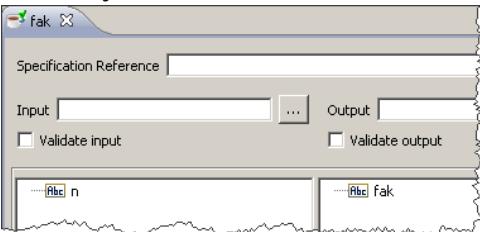
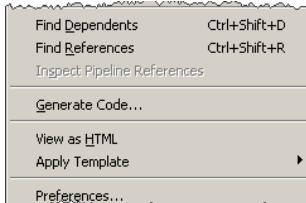
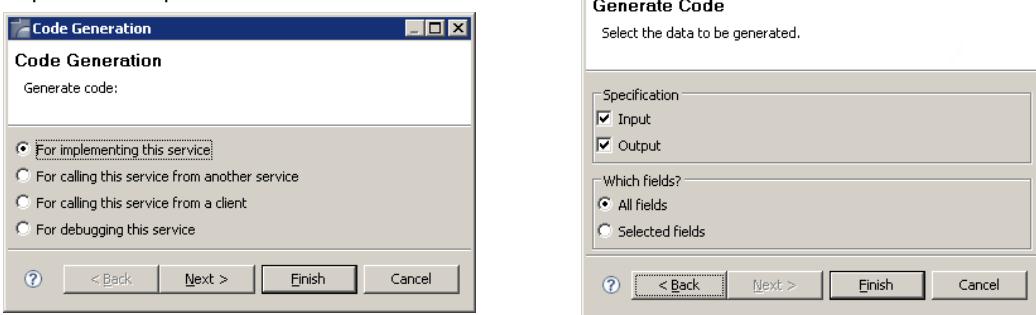
- **IData** objects are not accessed directly. A cursor is required to navigate and access **IData** objects.
 - **IDataCursor**, the cursor interface that the specific types implemented
- **IData** operations come in three parts:
 - get a cursor
 - position the cursor
 - get the value of the key at that position



Steps for Coding Using Designer

- 1) Create an empty Java service.
- 2) Specify the input/output.
 - Use the “input/output” tab
- 3) (Optional, but highly recommended) Generate code for implementing this service, and paste it into the source pane.
- 4) Specify packages to be imported.
 - Use the “Imports” section of the “Shared” tab or the Eclipse’s automatic Import functionality
- 5) Type your source code.
 - Use the Java Editor. Note that you must and can not change the grey areas

Writing Code to Manipulate Pipeline Data

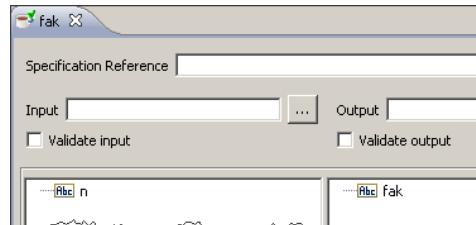
1. Write your service interface
2. Change to source view. Right click and choose Generate Code ➔ for implementing this service
3. Generate Code for implementing with inputs and outputs + all fields

Software AG Training | 7 - 17

Example: Generated String Input & Output Code...

- **Input code**

- 1) Get a cursor (required interface for navigating through the Pipeline).
- 2) Use the IDataUtil object to populate service variables.
- 3) Always destroy your cursor when you are done using it for most efficient performance).



```

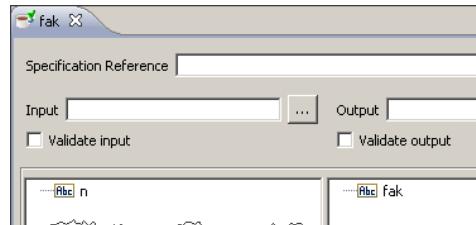
// pipeline
IDataCursor pipelineCursor = pipeline.getCursor();
String n = IDataUtil.getString( pipelineCursor, "n" );
pipelineCursor.destroy();

// pipeline
IDataCursor pipelineCursor_1 = pipeline.getCursor();
IDataUtil.put( pipelineCursor_1, "fak", "fak" );
pipelineCursor_1.destroy();

```

Software AG Training | 7 - 18

...Generated String Input & Output Code



```
// pipeline
IDataCursor pipelineCursor = pipeline.getCursor();
String n = IDataUtil.getString( pipelineCursor, "n" );
pipelineCursor.destroy();

// pipeline
IDataCursor pipelineCursor_1 = pipeline.getCursor();
IDataUtil.putString( pipelineCursor_1, "fak", fak );
pipelineCursor_1.destroy();
```

Output code

- 4) Get a cursor (or re-use an existing one).
- 5) Put data back in the Pipeline.
- 6) Destroy your cursor.

Software AG Training | 7 - 19

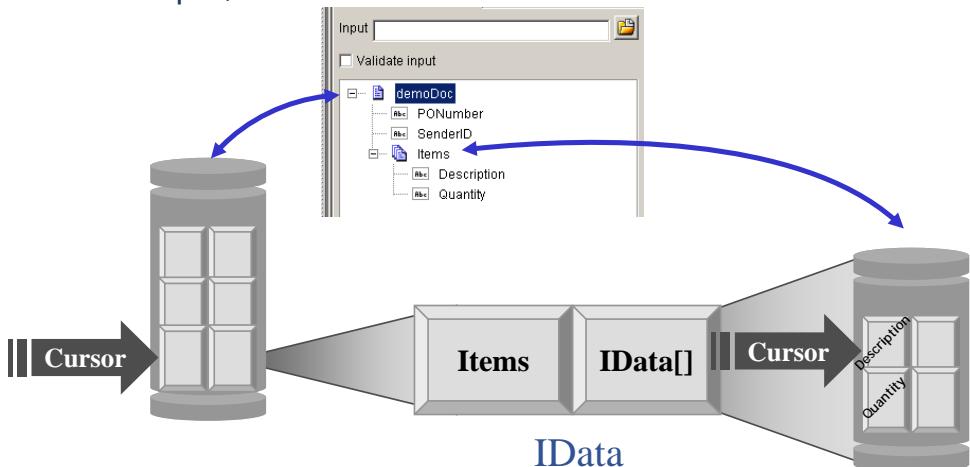
Example: Add this code to complete your service...

- Convert the input
- Multiply all numbers from 2 .. n This works for 0! and 1! as well.
- Convert the resulting BigInteger to a String

```
long l = Long.parseLong(n);
BigInteger f = BigInteger.ONE;
for (long i = 2; i <= l; i++) {
    f = f.multiply(BigInteger.valueOf(i));
}
String fak = f.toString();
```

Manipulating Documents In Java Services

- Documents are exposed to java services as nested IData objects or arrays of IData objects.
- In the example, we'll use the nested Document structure.

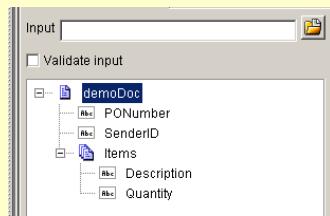


Example: Generated Document and Document List Input Code

```
// pipeline
IDataCursor pipelineCursor = pipeline.getCursor();
//demoDoc
IData demoDoc = IDataUtil.getData( pipelineCursor, "demoDoc" );
if ( demoDoc != null ) {
    IDataCursor demoDocCursor = demoDoc.getCursor();
    String PONumber = IDataUtil.getString( demoDocCursor, "PONumber" );
    String SenderID = IDataUtil.getString( demoDocCursor, "SenderID" );
    // i.Items
    IData[] Items = IDataUtil.getDataArray( demoDocCursor, "Items" );
    if ( Items != null ) {
        for ( int i = 0; i < Items.length; i++ ) {
            IDataCursor ItemsCursor = Items[i].getCursor();
            String Description = IDataUtil.getString( ItemsCursor, "Description" );
            String Quantity = IDataUtil.getString( ItemsCursor, "Quantity" );
            ItemsCursor.destroy();
        }
    }
    demoDocCursor.destroy();
}
pipelineCursor.destroy();
...
```

document

sub-documents

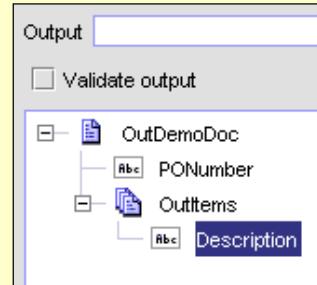


Example: Generated Document and Document List Output Code

- Output code uses IDataFactory.create() service
 - IDataFactory is a factory class which has static create () methods.
 - Creates document list:
 - IData[] DocumentList = new IData[arraysize];
 - DocumentList[i] = IDataFactory.create();

```
// OutDemoDoc
IData OutDemoDoc = IDataFactory.create();
IDataCursor OutDemoDocCursor = OutDemoDoc.getCursor();
IDataUtil.put(OutDemoDocCursor, "PONumber", "PONumber");

// OutDemoDoc.Items
IData[] OutItems = new IData[42];
foreach (IData OutItem : OutItems) {
    OutItem = IDataFactory.create();
    IDataCursor OutItemCursor = OutItem.getCursor();
    IDataUtil.put(OutItemCursor, "Description", "Description");
    OutItemCursor.destroy();
}
IDataUtil.put(OutDemoDocCursor, "OutItems", OutItems);
OutDemoDocCursor.destroy();
IDataUtil.put(pipelineCursor, "OutDemoDoc", OutDemoDoc);
```



More Object Types

- Create as a String and set Content type.

The screenshot shows the 'Specify Content type' dialog box and the 'Properties' panel. The 'Content type' dropdown in the dialog is set to 'integer_customized'. The 'Properties' panel on the left shows the following settings:

Property	Value
General	
Required	True
Allow null	True
Allow unspecific	True
Content type	integer_customized
Java wrapper ty	UNKNOWN

The 'Specify Content type' dialog also displays fields for primitiveType (decimal), minInclusive (0), maxInclusive (9), and pattern.

Software AG Training | 7 - 24

More Object Types

- Create as an **Object** and set **Java wrapper type**. Note that the Icon changes.

The screenshot shows the 'Properties' dialog for an object named 'long'. The Java wrapper type is set to 'java.lang.Long'. A callout box highlights the properties:

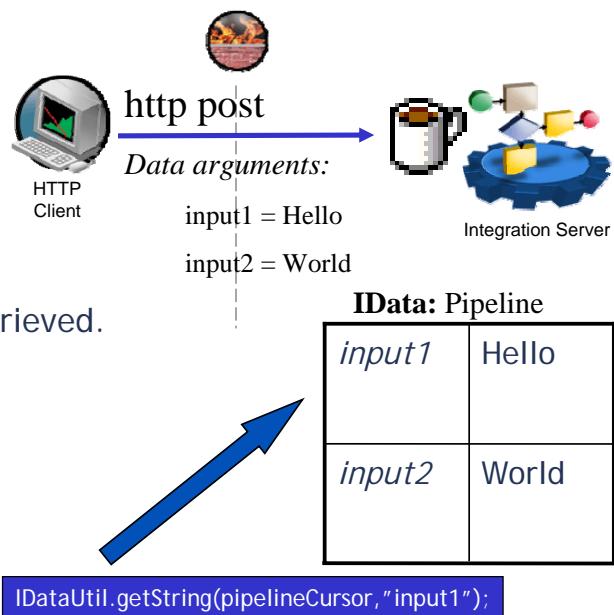
- Name=long
- Type=Object
- String display type=Text Field
- Required=True
- Allow null=True
- Allow unspecified fields=True
- Java wrapper type=java.lang.Long
- Path=long

Other Data Types

- Many other data types and combinations of data types can be used with Java services.
- **IDataUtil.get:** returns a generic `java.lang.Object`.
 - Therefore, it is the responsibility of the programmer to cast this to the appropriate type.
 - `Object object = IDataUtil.get (pipelineCursor, "object");`
- **IDataUtil.getObjectArray:** returns an array of `java.lang.Object`s.
- **IDataUtil.getInt:** returns an `int`.
- **IDataUtil.getBoolean:** returns a `boolean`.

Recap: IData in Action

- 1) IData (Pipeline) instantiated.
- 2) Pipeline populated with inputs.
- 3) Service executes.
- 4) Cursor created.
- 5) Cursor positioned and data retrieved.
- 6) Service logic code executes.
- 7) Cursor positioned and data inserted into pipeline.
- 8) Cursor destroyed.
- 9) Pipeline destroyed.



This page intentionally left blank.

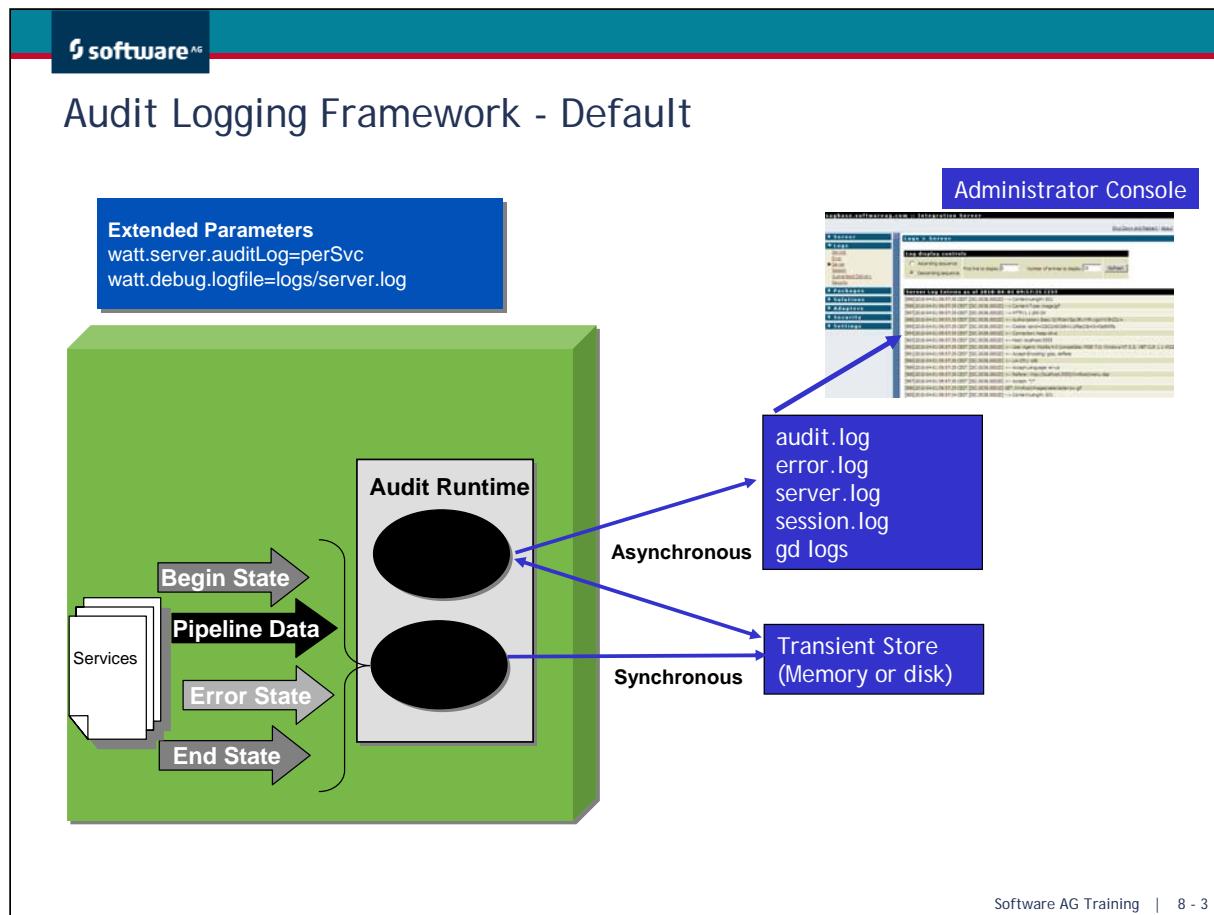


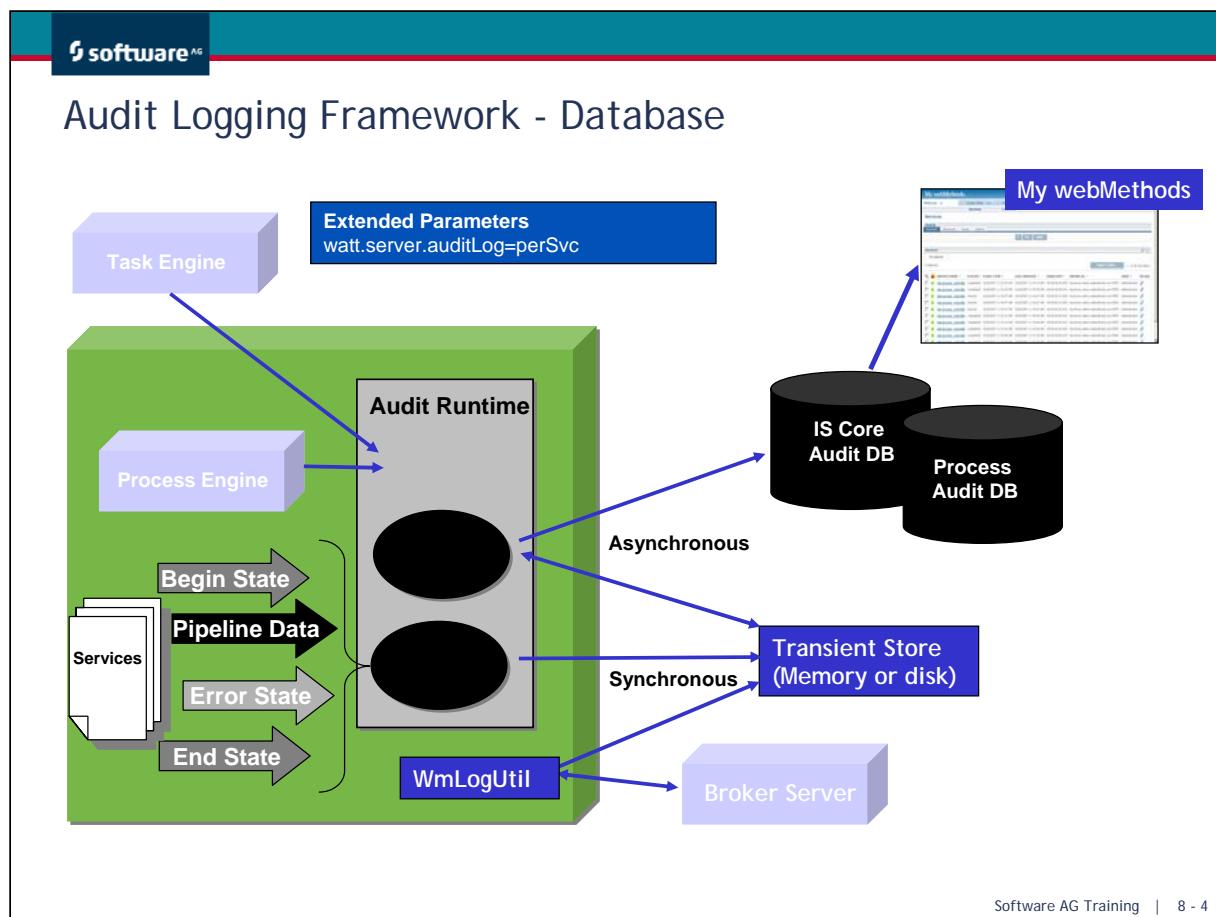
8

Monitoring Services

Objectives

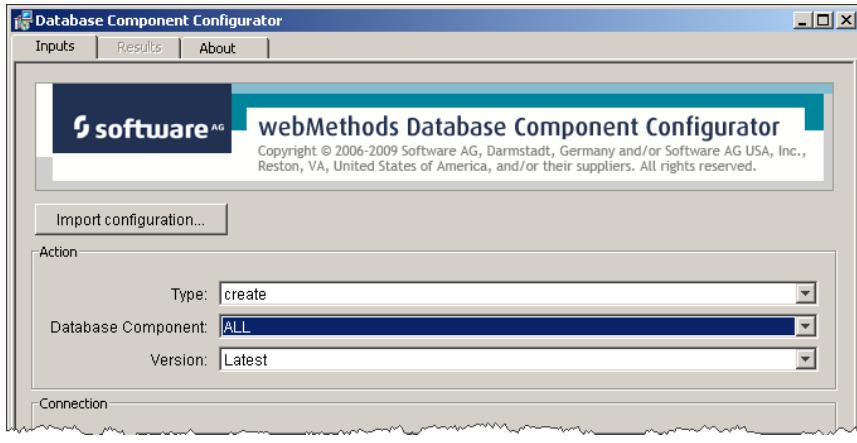
- At the end of this section, you will be able to:
 - set up logging for Services and Processes
 - describe the key features of My webMethods Monitor
 - monitor services





Administration Task - Prepare Database Tables

- Use the database configuration wizard



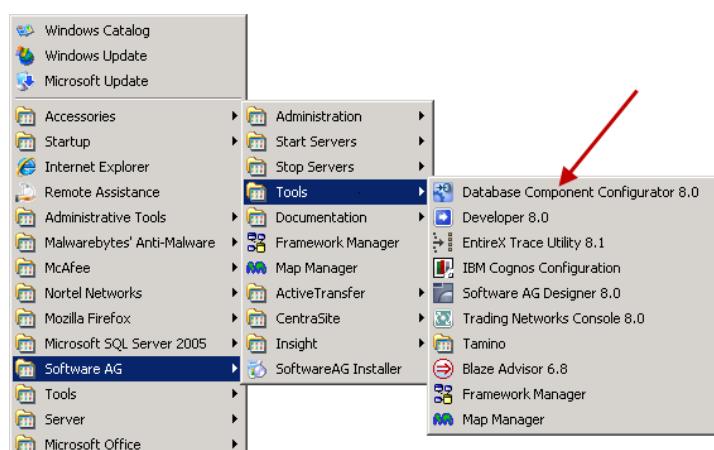
The screenshot shows the 'Database Component Configurator' application window. At the top, there's a toolbar with 'Inputs', 'Results', and 'About' buttons. Below the toolbar is a header bar with the 'software AG' logo and the text 'webMethods Database Component Configurator'. The header also includes a copyright notice: 'Copyright © 2006-2009 Software AG, Darmstadt, Germany and/or Software AG USA, Inc., Reston, VA, United States of America, and/or their suppliers. All rights reserved.' The main interface has a section titled 'Action-' with three dropdown menus: 'Type' set to 'create', 'Database Component' set to 'ALL', and 'Version' set to 'Latest'. There's also a 'Connection' section at the bottom.

Software AG Training | 8 - 5

Administration Task - Database Component Configurator

1. Start it from the START menu:

2. Select the appropriate Action Type and fill in the required Arguments



Action

Type:

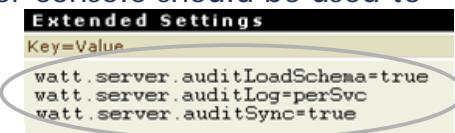
Database Component:

Version:

Software AG Training | 8 - 6

Administration Task - Configure Server Settings

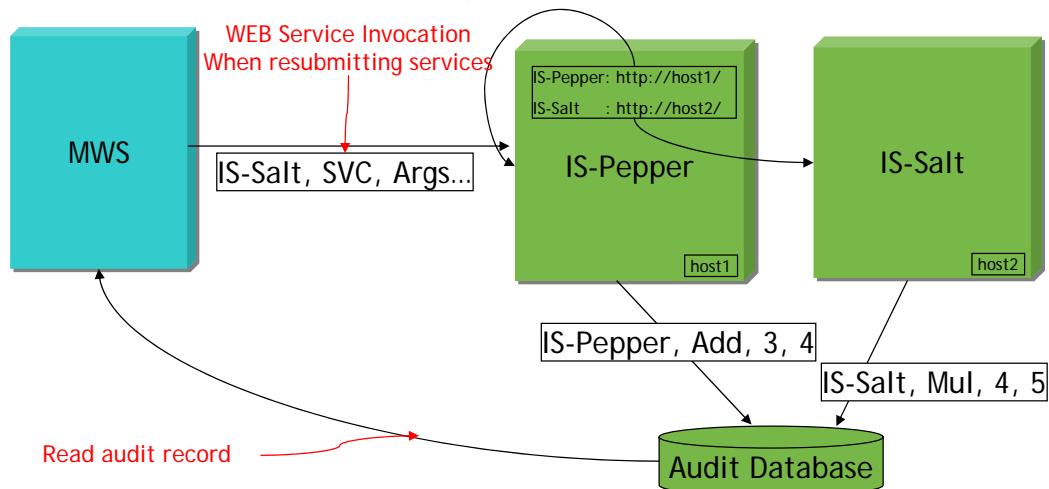
- The Integration Server can be configured to perform auditing and to what resource through extended Server parameters.
- Extended Settings in the Administrator console should be used to change this configuration.



<i>auditLog</i> Setting	Effect
off	Logging is turned off for the entire server
perSvc	Logging will only occur for services which have their audit settings enabled. This is the default setting.
brief	All service state changes (begin, end, and failure) will be logged. The pipeline will not be included unless the service setting for a particular service would normally include the pipeline.
verbose	All service state changes will be logged along with the input pipeline.

What is a remote Server alias?

- 2 Integration Servers sending audit data to the same DB.
- MWS only talks to one Integration Server
- This Integration server must know the Locations of the Integration Servers, given their logical name
- This Translation is done using the remote server alias



Administration Task - Prepare Remote Server Alias

- Used by the auditing subsystem to resubmit services.

The screenshot shows the 'sagbase.softwareag.com : Integration Server' administration interface. On the left, a sidebar menu under 'Settings' includes 'Remote Servers'. A blue arrow points from this menu to the main content area. The main content area shows the 'Remote Server Alias Properties' for 'IS1'. A blue callout box with the text 'Host Names must match exactly' points to the 'Host Name or IP Address' field, which contains 'sagbase.softwareag.com'. Other fields include 'Alias' (IS1), 'Port Number' (5555), 'User Name' (Administrator), 'Password' (redacted), 'Execute ACL' (Internal), 'Max Keep Alive Connections' (5), 'Keep Alive Timeout (minutes)' (1), 'Use SSL' (radio button selected for 'No'), and a 'Save Changes' button.

Host Names must match exactly

Remote Servers

sagbase.softwareag.com : Integration Server

Settings > Remote Servers > IS1

Return to Remote Servers

Remote Server Alias Properties

Alias	IS1
Host Name or IP Address	sagbase.softwareag.com
Port Number	5555
User Name	Administrator
Password	*****
Execute ACL	Internal
Max Keep Alive Connections	5 (Default 5)
Keep Alive Timeout (minutes)	1 (Default 1)
Use SSL	<input checked="" type="radio"/> Yes <input type="radio"/> No
Retry Server	

Save Changes

Software AG Training | 8 - 9

Create a new Pool Alias

seag.com :: Integration Server

Settings > JDBC Pools

- [Create a new Pool Alias Definition](#)
- [Create a new Driver Alias Definition](#)

Functional Alias Definitions

Function Name	Associated Pool Alias	Edit Association	Test	Restart Funct
Adapters		Edit		Restart
Archiving		Edit		Restart
BPELEngine	Local	Edit		Restart
CentralUsers	CentralUsersPool	Edit		Restart
DocumentHistory	Local	Edit		Restart
ISCoreAudit	Local	Edit		Restart
ISInternal	Local	Edit		Restart
ProcessAudit	Local	Edit		Restart
ProcessEngine	Local	Edit		Restart
Reporting	Local	Edit		Restart
Simulation	Local	Edit		Restart
Staging		Edit		Restart
TN	Local	Edit		Restart
Xref	Local	Edit		Restart

Shut Down and Restart | About | Help

Software AG Training | 8 - 10

software AG

Administration Task - Define a JDBC Pool Alias II

- Example - Defining a JDBC Pool using the “DataDirect Connect for JDBC” driver within IS Administration Console

JDBC Connection Pool Alias

Alias Name	IS Pool for Auditing
Alias Description	
Associated Driver Alias	DataDirect Connect JDBC Oracle Driver
Database URL	jdbc:wm:oracle://localhost:1521;SID=ORCL
Sample Database URL	jdbc:wm:oracle://<server>:<1521 port>;serviceName=<value>[;<option>=<value>...]
User Id	scott
Password	*****
Minimum Connections	0
Maximum Connections	5
Idle Timeout	60000 milliseconds
<input type="button" value="Test Connection"/> <input type="button" value="Save Settings"/>	

Software AG Training | 8 - 11

Administration Task - Associate the Pool Alias

- Refer to Advantage for performance considerations when associating JDBC pools with specific functions.

Associate the alias with the appropriate function

Description	Associated Pool Alias	Edit Association	Test	Restart Function
Schema for archiving data from the Process Audit and IS Core Audit schemas.		Edit		Restart
Central User Management Configuration		Edit		Restart
Document History for Exactly Once Processing		Edit		Restart
IS Core Audit Log Manager Function	SharedPool	Edit		Restart
For internal use by IS facilities	SharedPool	Edit		Restart
Process Audit Log Manager Function	SharedPool	Edit		Restart
Schema for Process Engine		Edit		Restart
Analytical schema for Process Reporting		Edit		Restart
Intermediate schema for Process Reporting		Edit		Restart
Trading Networks Function	SharedPool	Edit		Restart
Key Cross Referencing and Echo Suppression		Edit		Restart

Click Test to verify JDBC pool settings are correct

Administrator Tasks - Summary

- Modify server parameters using Extended Settings.
- Define a remote server alias if Resubmit is desired.
- Create JDBC pool aliases as needed.
- Associate the JDBC pool aliases with the appropriate functions.
- Test the JDBC pool once associated.
- Restart the Integration Server to put all changes into effect.

Note: these have already been applied to the VM

Don't restart until all changes have been made!

Development Task - Enable Auditing in Service

The screenshot shows the 'Properties' tab of the 'loopTest' service. The 'Audit' section is expanded, displaying the following configuration:

Property	Value
Enable auditing	Always
Log on	Error and success
Include pipeline	Always
Universal name	
Output template	

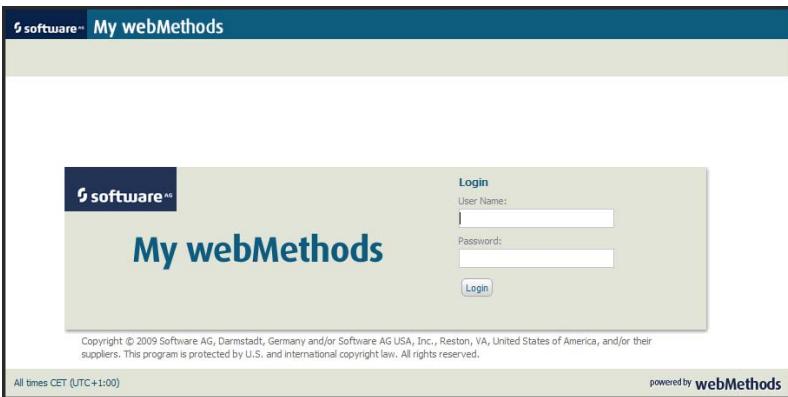
Annotations:

- 'When to log?' points to the 'Enable auditing' property.
- 'What to log?' points to the 'Log on' property.
- 'Include pipeline?' points to the 'Include pipeline' property.

Software AG Training | 8 - 14

Monitoring - My webMethods Server

- My webMethods Server (MWS) is an implementation of a Portal with specialized interfaces for webMethods products.
- The webMethods Suite uses MWS for monitoring and administration
- Access is done with a browser using `http://<hostname>:<port>`
 - `http://localhost:8585/` is used in our VM.



Software AG Training | 8 - 15

The screenshot shows the 'My webMethods' home page. At the top, there's a navigation bar with links for 'My webMethods Administrator', 'My Profile', 'Help', 'About', and 'Logout'. Below the navigation bar is a toolbar with 'Navigate', 'Tools', 'Applications', and 'Workspaces' buttons. A blue callout box points to the 'Applications' button with the text 'Select to view Monitoring or Administration'. The main content area is titled 'Welcome' and features a large central box with the heading 'Welcome to My webMethods 8' and the sub-instruction 'Click on each box below to view various screen samples. These samples will help you learn the new features of My webMethods.' To the right of this central box is a sidebar with text about new features and a link to the User's Guide. At the bottom of the page, it says 'All times CET (UTC+1:00)' and 'powered by webMethods'.

Software AG Training | 8 - 16

The screenshot shows the webMethods Monitoring Main Page. The left sidebar has a tree view under 'Applications' with 'Monitoring' expanded, showing 'System-Wide', 'Business', and 'Integration'. Under 'Integration', 'B2B' is selected, which is highlighted in orange. A blue callout box labeled 'Service monitoring' points to this selection. Another blue callout box labeled 'Business monitoring' points to the 'Business' node in the tree. The main content area has tabs for 'Welcome', 'Services', and 'New'. The 'Services' tab is selected. It contains a 'Search' section with a keyword input field and search buttons, and a 'Services' table listing one item. The table columns are: Service Name, Status, Start Time, Last Updated, Duration, Server ID, User, and Detail. The listed item is 'acme_Purchase..._Completed'. A blue callout box labeled 'Active alerts' points to the table. A blue callout box labeled 'View invocation details' points to the 'Detail' column header. The bottom right of the page says 'powered by webMethods'.

Service Name	Status	Start Time	Last Updated	Duration	Server ID	User	Detail
acme_Purchase..._Completed	Completed	3/26/2010 10:31:40 AM	3/26/2010 10:31:40 AM	0d 00:00:00.000	sagbase.softw...	bill	

The screenshot shows the 'Services' search interface. At the top, there's a search bar with tabs for 'Keyword', 'Advanced', 'Saved', and 'Options'. Below the search bar is a table titled 'Services' with two items listed. The columns are: Service, Name, Status, Start Date, Last Updated, Duration, Server ID, User, and Detail. The first item is 'acme.Purchase_Completed' and the second is 'acme.Purchase_Resubmitted'. Each row has a checkbox and a blue circular icon next to the service name. Arrows point from three callout boxes to specific elements: one arrow points to the 'Keywords:' input field with the text 'Add search terms if desired'; another arrow points to the 'Advanced' tab in the search bar with the text 'Choose Advanced for more search options'; and a third arrow points to the 'Detail' column header with the text 'Click to see details of a specific service'.

Services

Search

Keyword Advanced Saved Options

Keywords: ? Search Save

Services

Resubmit 0 selected Export Table... 1 - 2 of 2 Items

Service	Name	Status	Start Date	Last Updated	Duration	Server ID	User	Detail
	acme.Purchase_Completed	Completed	3/26/2010 11:07:51 AM	3/26/2010 11:07:53 AM	0d 00:00:00.000	sagbase.softw...	administrator	
	acme.Purchase_Resubmitted	Resubmitted	3/26/2010 10:31:40 AM	3/26/2010 11:07:52 AM	0d 00:00:00.000	sagbase.softw...	administrator	

Add search terms if desired

Choose Advanced for more search options

Click to see details of a specific service

The screenshot shows the 'Audited Services - Detail' page from the webMethods 8 Integration Workshop. The top navigation bar includes the Software AG logo and the title 'Audited Services - Detail'. Below the title is a 'Service Information' panel with the following details:

Service Name:	acme.PurchaseOrder.work:loopTest
Root Context ID:	6214b140-38ba-11df-a434-cfe14ec5bae6
Parent Context ID:	6214b140-38ba-11df-a434-cfe14ec5bae6
Context ID:	70f1d490-38bf-11df-a5e4-ea17f19dde1c
Server ID:	sagbase.softwareag.com:5555
User:	administrator
Timestamp:	3/26/2010 11:07:53 AM
Current Status:	Completed
Root Service:	acme.PurchaseOrder.work:loopTest
Parent Service:	acme.PurchaseOrder.work:loopTest

Below the service information is a 'History' table:

TIMESTAMP	STATUS
3/26/2010 11:07:53 AM	Completed

Annotations with arrows point to specific fields in the service information panel:

- An arrow points to the 'Edit Pipeline' button with the text 'Edit the service pipeline'.
- An arrow points to the 'Resubmit' button with the text 'Re-execute the service'.
- An arrow points to the 'Timestamp' field with the text 'Runtime data detail'.
- An arrow points to the 'Current Status' field with the text 'Runtime execution detail'.

Software AG Training | 8 - 19

software AG

Resubmitted Services

- Once resubmitted, the original instance of the service will show a resubmitted status in the summary list and in the details window.
- The new execution instance will appear as a new item.

The screenshot shows a table of services with columns: Service Name, Status, Start Time, Last Updated, Duration, Server ID, User, and Detail. Two rows are highlighted with a red oval: the first row has status 'Completed' and the second row has status 'Resubmitted'. Below the table are two detailed views side-by-side:

Service Name	Status	Start Time	Last Updated	Duration	Server ID	User	Detail
acme.Purchase...	Completed	3/26/2010 11:07:53 AM	3/26/2010 11:07:53 AM	0d 00:00:00.000	sagbase.softw...	administrator	
acme.Purchase...	Resubmitted	3/26/2010 10:31:40 AM	3/26/2010 11:07:52 AM	0d 00:00:00.000	sagbase.softw...	administrator	

« Previous | 1 | Next »

Original

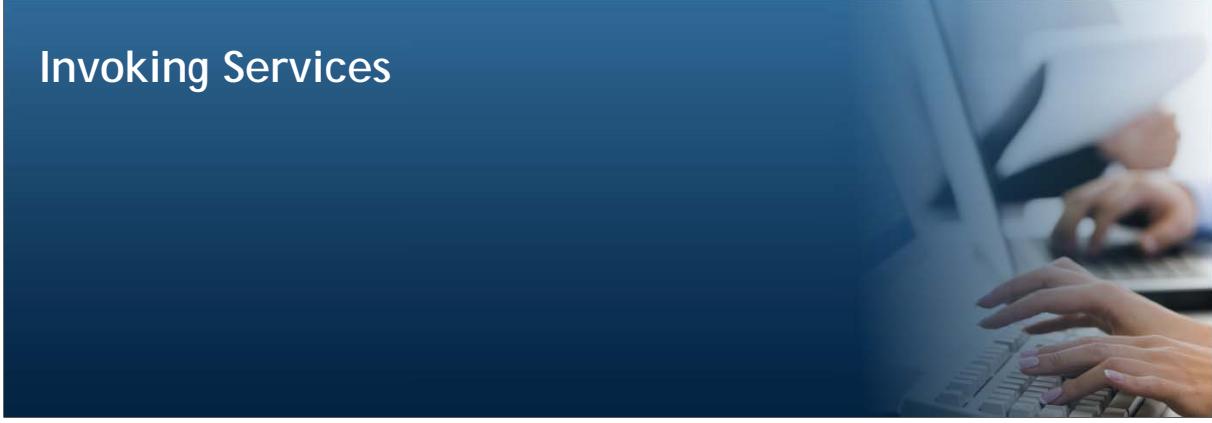
Service Name: acme.PurchaseOrder.work:loopTest
Root Context ID: 62065960-38ba-11df-a426-9e290c95b6ce
Parent Context ID: 62065960-38ba-11df-a426-9e290c95b6ce
Context ID: 6214b140-38ba-11df-a434-cfe14ec5bae6
Server ID: sagbase.softwareag.com:5555
User: administrator
Timestamp: 3/26/2010 11:07:52 AM
Current Status: Resubmitted
Child Services: acme.PurchaseOrder.work:loopTest

Service Name: acme.PurchaseOrder.work:loopTest
Root Context ID: 6214b140-38ba-11df-a434-cfe14ec5bae6
Parent Context ID: 6214b140-38ba-11df-a434-cfe14ec5bae6
Context ID: 70f1d490-38bf-11df-a5e4-ea17f19dde1c
Server ID: sagbase.softwareag.com:5555
User: administrator
Timestamp: 3/26/2010 11:07:53 AM
Current Status: Completed
Root Service: acme.PurchaseOrder.work:loopTest
Parent Service: acme.PurchaseOrder.work:loopTest

Software AG Training | 8 - 20

9

Invoking Services



Objectives

- At the end of this section, you will be able to:
 - send data to the Integration Server from any client
 - manipulate the data that was sent
 - list different ways to send and receive XML documents
 - describe the steps involved with processing incoming XML documents
 - list the most common built-in services for manipulating XML documents
 - query an XML document

Sending Data to the Integration Server

- Integration Server acts as multiple servers in one
- Common input methods:
 - HTTP/S
 - FTP/S
 - File Polling
 - SMTP
 - Web Service
- Ports must be defined for each incoming protocol
 - Define through the Administration console under Security ➔ Ports

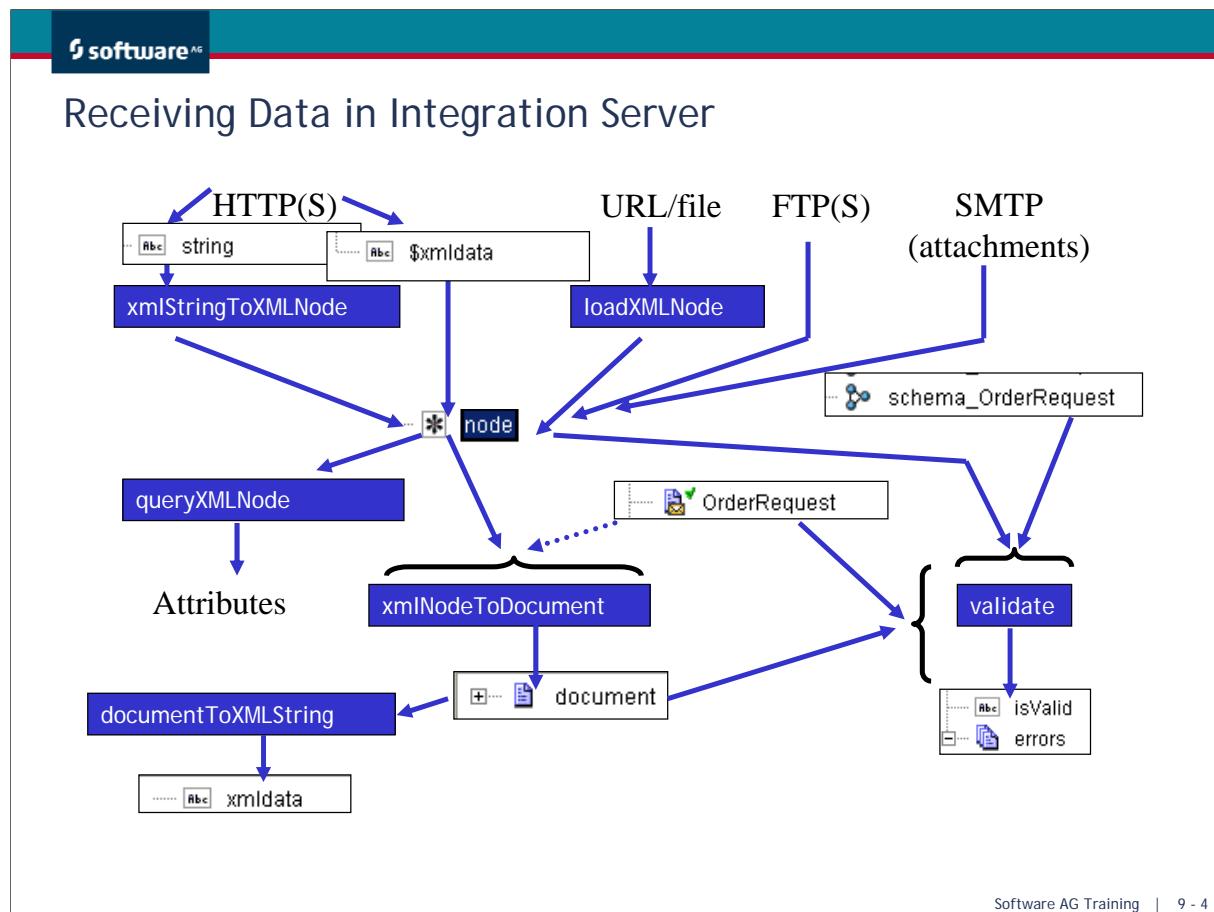
Built-in Web Server

Built-in FTP Server

via File System

via External E-mail Server

SOAP/RPC, SOAP/MSG



Testing with XML Documents

- How to test Services with data (XML documents)?
 - Invoke pub.xml:xmlNodeToDocument
 - Create a debug launch configuration
 - As Input select the Use XML radio button and browse for a file to send
 - If you want, make the run configuration available in the favorites

The image contains three side-by-side screenshots of the webMethods Integration Platform's configuration interface:

- Screenshot 1 (Left):** Shows the "Input" tab of a configuration dialog. It has fields for "Integration Server" (set to "Default") and "Location" (set to "localhost:5555"). Under "Service", it lists "AcmeSupport.xml:xmlAdd" and a "Browse..." button. A checkbox "Stop at first flow step" is checked.
- Screenshot 2 (Middle):** Shows the "Input" tab with the "Use XML" radio button selected. Below it, there is a text area containing XML code:


```
<?xml version="1.0"?>
<!--
webMethods Integration Platform
Integration Class
Sample WSDL Message describing input for an add request
-->
<root>
  <a>>10</a>
  <b>>24</b>
</root>
```
- Screenshot 3 (Right):** Shows the "Common" tab of the configuration dialog. It includes sections for "Save as" (radio buttons for "Local file" or "Shared file"), "Console Encoding" (radio buttons for "Default - Inherited (Cp1252)" or "Other" set to "ISO-8859-1"), and "Display in favorites menu" (checkboxes for "Debug" and "Run"). The "Standard Input and Output" section includes checkboxes for "Allocate Console (necessary for input)" and "File:" (with "Workspace...", "File System...", and "Variables..." buttons). A "Launch in background" checkbox is also present.

From Thin (HTTP) Clients

- Invoke via HTTP “post” or “get”
- Specify service namespace location in URL

http://server:port/invite/folder.subfolder/service

<i>IP address and HTTP/S listening port</i>	<i>invoke key-word (case-sensitive)</i>	<i>folders (separate by “dot”)</i>	<i>service name</i>
---	---	------------------------------------	---------------------

- via Browser
 - HTML <FORM> tag
- via Application Server
 - Use any HTTP post client
 - e.g. java.net.HttpURLConnection

From FTP Clients

- Invoke via FTP “put”
- Specify service location via ftp directory

```
ftp>open server port  
ftp>cd ns/folder/subfolder/service  
ftp>put myfile.xml  
ftp>get myfile.xml.out
```

Open FTP connection

Point to service

Copy input file to service

Get results (if needed)

- ns - Namespace directory

The screenshot shows the Software AG Integration Server interface. At the top, there's a toolbar with icons for file operations. Below it is a header bar with the Software AG logo and the title "FTP Example".

FTP Session:

```
C:\SoftwareAG\IntegrationServer\packages\AcmeSupport\pub>ftp
ftp> open localhost 9021
Connected to sagbase.softwareag.com.
220 sagbase:9021 FTP server <webMethods Integration Server version 8.0.1.0> ready.
User sagbase.softwareag.com:<none>: Administrator
331 Password required for Administrator.
Password:
230 User Administrator logged in.
ftp> cd ns/acme/PurchaseOrder/utils/inspectLineItems
250 CWD command successful.
ftp> send PORequest.xml
200 PORT command successful.
150 ASCII mode data connection for PORequest.xml (127.0.0.1,1437).
226 ASCII transfer complete.
ftp: 7298 bytes sent in 0.02Seconds 456.13Kbytes/sec.
ftp> dir
200 PORT command successful.
150 ASCII mode data connection for /bin/ls (127.0.0.1,1438).
total 1
drwxr-xr-x 3 root      root          1 Mar 30 16:08 .
drwxr-xr-x 3 root      root          1 Mar 30 16:08 ..
r--r--r-- 1 tx        tx          110 Mar 30 16:08 PORequest.xml.out
226 ASCII transfer complete.
ftp: 233 bytes received in 0.00Seconds 233000.00Kbytes/sec.
ftp> get PORequest.xml.out
200 PORT command successful.
150 ASCII mode data connection for PORequest.xml.out (127.0.0.1,1439) (110 bytes).
226 ASCII transfer complete.
ftp: 110 bytes received in 0.00Seconds
ftp> !type PORequest.xml.out
<?xml version="1.0" encoding="UTF-8"?>
<Values version="2.0">
  <value name="isValid">true</value>
</Values>
ftp> quit
221 Goodbye.
```

XML Inspection:

A separate window titled "inspectLineItems" displays the XML structure of the "PORequest.xml" file. The tree view shows the following structure:

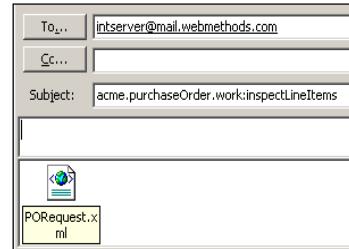
- pub.xml:xmlNodeToDocument
- MAP
- LOOP over '|OrderRequest/PurchaseOrderRequest/PurchaseOrder/ProductLineItem'
- BRANCH
- %OrderRequest/PurchaseOrderRequest/PurchaseOrder/ProductLineItem/ProductQuantity% < 1: SEQUENCE
- MAP
- EXIT '\$loop'

From SMTP Clients

- Invoke by sending an e-mail to a specific email account
- Specific service via subject line
- Send input as file attachments or body text

```
mailto:intserv@mail.webmethods.com  
subject:acme.purchaseOrder.work:InspectLineItems  
body:inString1=""  
attachment:POResult.xml
```

- Requires an external SMTP server
 - Can be POP3 or IMAP



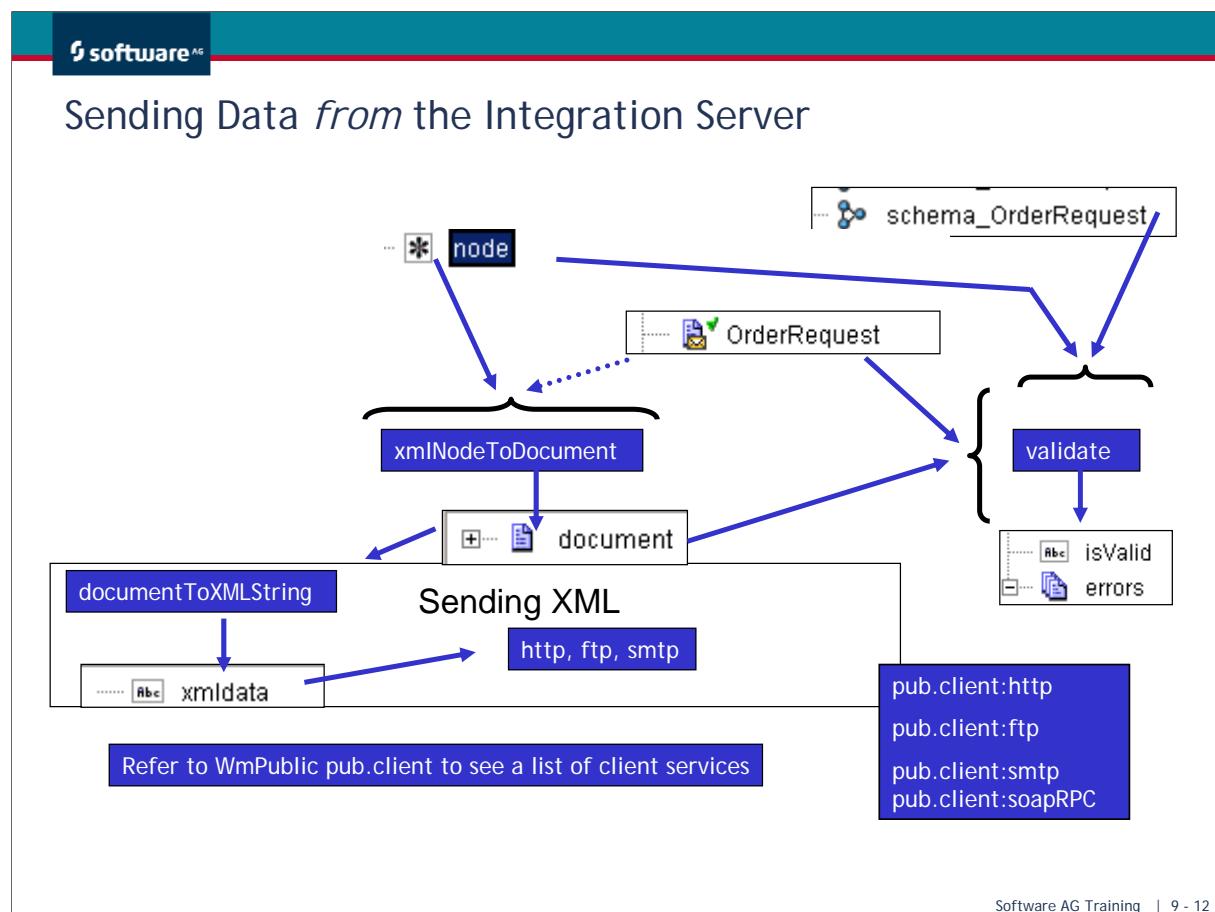
The screenshot shows the webMethods Designer interface. On the left, there's a tree view with nodes like 'xml' and 'xmlAdd'. A context menu is open over the 'xmlAdd' node, with options: 'Inspect Pipeline References', 'Generate Code...', and 'View as HTML'. In the center, there are two overlapping dialog boxes. The top dialog is titled 'Code Generation' and 'Generate client code.' It has a 'Language' dropdown set to 'Java' (which is highlighted in blue) and a list of other languages: Visual Basic 5/6, Excel 97/2000, C/C++ for Windows, C/C++ for Solaris, C/C++ for HP-UX, C/C++ for Linux, C/C++ for AIX, and C# for .NET. The bottom dialog is also titled 'Code Generation' and 'Generate code...' and has a section 'Generate code for:' with three radio button options: 'For implementing this service', 'For calling this service from another service', and 'For calling this service from a client'. The third option is selected. At the bottom of both dialogs are 'Finish' and 'Cancel' buttons.

From Coded Clients

- webMethods provides client APIs for Java, COM & C/C++.
 - Can be used to generate stand-alone clients or embedded in 3rd party applications.
 - Example: embed Java client in EJB server
- Client code can be auto-generated via Designer.

Which Service Can I Invoke?

- Any client can invoke any service, provided:
 - client is authenticated.
 - client possesses appropriate privileges
 - service invoke is allowed on that port
- Best Practice server administration
 - exposed a limited number of public "entry point" services
 - example: Trading Networks "Universal Inbox"
 - `wm.tn.receive`



10

Flat File Handling

Objectives

- At the end of this section, you will be able to:
 - describe the webMethods Flat File Architecture
 - create a Flat File Dictionaries and Schemas using Developer
 - parse delimited, fixed-length, variable-length, and complex Flat Files
 - create custom Format services

What is a Flat File?

- Flat files contain hierarchical data in a record based format.
- Flat files metadata is separate from its data, unlike XML.
- A Flat file contains:
 - Records - contain fields and composites
 - Composites - multiple related fields (address)
 - Fields - atomic data (zip code)
- Flat File types:
 - record Identifier present
 - no record Identifier present

Flat File Examples...

- Flat file with Record ID (ID in Bold)

- Simple

```
ADDRESS,Acme Hammer Company,123 Wilson St.,Sacramento+CA+95833  
ADDRESS,Johnson Supply Co.,456 Nadia Ave.,Seattle+WA+98188
```

- Complex (Parent-Child relationships)

```
START*PURCHORD*00025387  
BEGINDOC*00*BE*M1Q512345*120030221*12345*57823  
-12  
CURRENCY*BY*USD
```

...Flat File Examples

■ Flat files without Record ID

- Delimited (CSV)

```
Acme Hammer Company,123 Wilson St.,Sacramento,CA,95833  
Johnson Supply Co.,456 Nadia Ave.,Seattle,WA,98188
```

- Fixed length

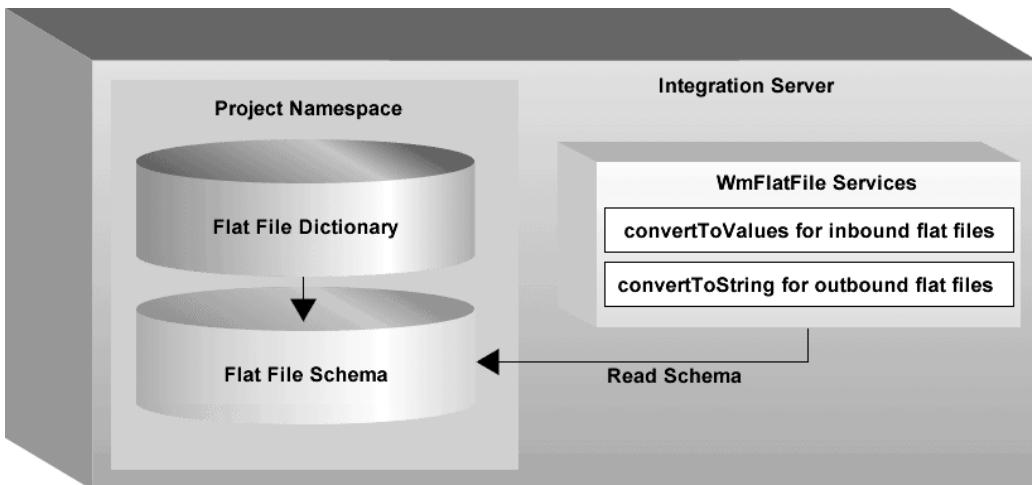
Acme Hammer Company	123 Wilson St.	Sacramento	CA95833
Johnson Supply Co.	4256 Nadia Ave.	Seattle	WA98188

- Variable length

- First columns in each record specifies the length. Can be different

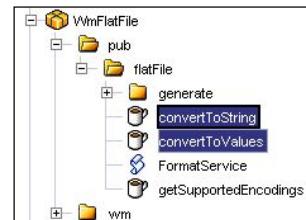
Flat File Architecture

- The built-in services in the WmFlatFile package uses Flat File Schemas and those Schemas can reference Flat File Dictionaries.



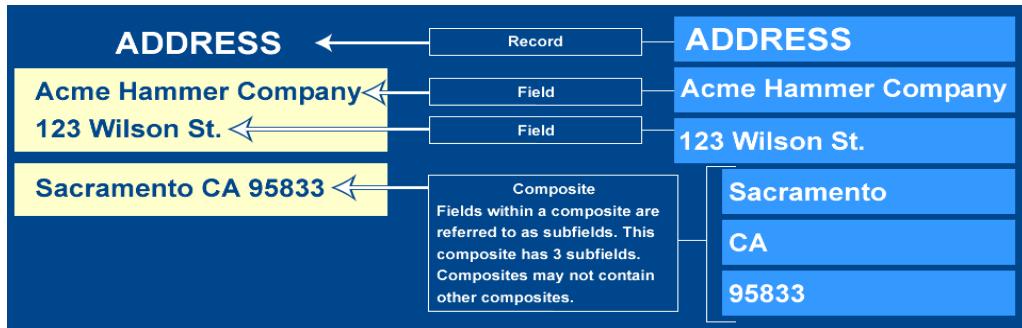
WmFlatFile Package

- The built-in services that process flat files are located in the WmFlatFile package.
- The convertToString service is used to translate a record based flat file to an IS document structure.
- The convertToValues service does the opposite. It translates an IS document structure back to a record based flat file.



Records, Fields, and Composites: Example

ADDRESS,Acme Hammer Company,123 Wilson St.,Sacramento+CA+95833



Delimited Record: Example

- Record Delimited by NewLine Character

```
ADDRESS,Acme Hammer Company,123 Wilson St.,Sacramento+CA+95833
ADDRESS,Johnson Supply Co.,456 Nadia Ave.,Seattle+WA+98188
```

Record delimiter	= newline
Field delimiter	= ,
Subfield delimiter	= +

- Record Delimited by ! character

```
ADDRESS*Acme Hammer Company*123 Wilson St.*Sacramento+CA+95833!
ADDRESS*Johnson Supply Co.*456 Nadia Ave.*Seattle+WA+98188!
```

Record delimiter	= !
Field delimiter	= *
Subfield delimiter	= +

Flat File Schemas

- A Flat File Schema is a webMethods object that contains the structural information of a flat file.
- Acts as a blueprint for parsing and creating flat files.
- Includes record formats, delimiters and the model which inbound flat files are optionally validated against.

Creating a Flat File Schema

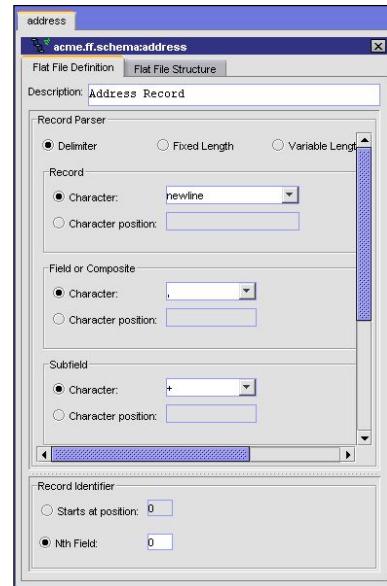
- Steps to create a flat file schema.
 - Create and configure the flat file schema.
 - Specify settings.
 - Create the structure of the flat file schema.
 - Test the flat file schema.
- A flat file schema for the following file will be shown:

```
ADDRESS,Acme Hammer Company,123 Wilson St.,Sacramento+CA+95833
ADDRESS,Johnson Supply Co.,4256 Nadia Ave.,Seattle+WA+98188
ADDRESS,Garcia Hammer Emporium,4950 St. Moritz Blvd.,Dallas+TX+75202
ADDRESS,Mary Annes Hardware,601 Jefferson Ave.,Detroit+MI+48230
ADDRESS,Hard Times Development Co.,91235 Anderson St.,Boston+MA+02125
```

Record delimiter	= newline
Field delimiter	= ,
Subfield delimiter	= +

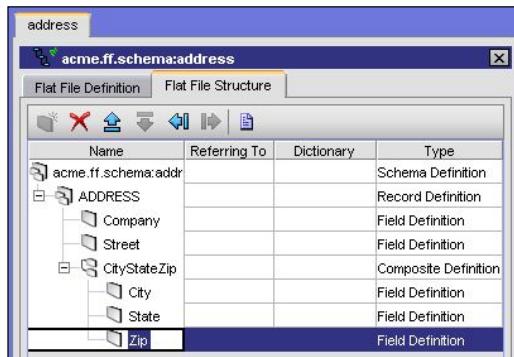
Schema Definition

- In the image to the right, a new flat file schema has been created in Developer.
- A 'Description' has been entered.
- The 'Record Parse' is 'Delimiter'.
- The Record Character is 'newline'.
- The Field or Composite Character is ','.
- The Subfield Character is '+'.
- The Record Identifier (ADDRESS) is at the beginning of each record, so 'Nth Field' for Record Identifier is '0'.



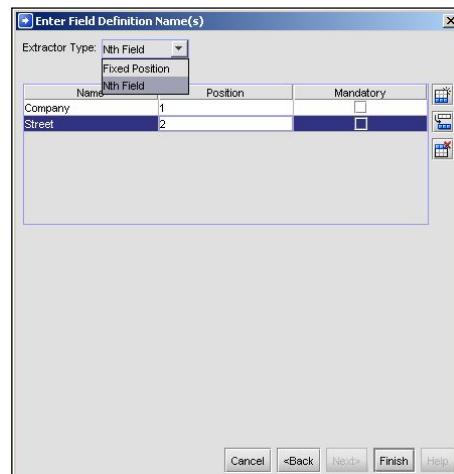
Flat File Structure

- The 'Flat File Structure' tab looks like the image below.
- A Record has been created called ADDRESS.
- ADDRESS contains two fields (Company and Street).
- ADDRESS contains one composite called CityStateZip with three subfields:
 - City
 - State
 - Zip



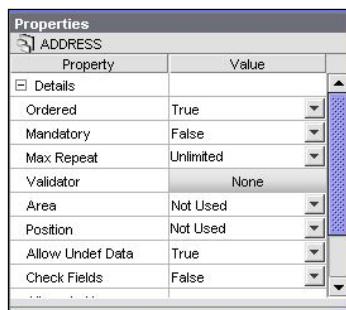
Extractors

- Extractors parse fields out of records.
- Types of Extractors
 - fixed position: used for fixed width records
 - nth fields: used for Delimited records
 - The ADDRESS record contains delimited field and composites, so an Nth field extractor is used.



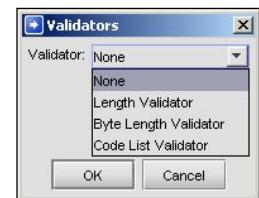
Record Properties

- Record Properties can be set in the Properties panel in Developer.
- ‘Max Repeat’ for the ADDRESS record is set to ‘Unlimited’ so any number of ADDRESS records can be included.
- A Validator can be specified to validate the record.



Validators

- A Validator can be specified in the properties of a record, composite, and a field.
- There are four types of Validators:
 - Conditional Validators - Used mostly with EDI documents. Only for Records and Composites.
 - Length Validators - Set the minimum and maximum character length.
 - Byte Length Validators - Set the minimum and maximum byte length. Used with multi-byte character sets.
 - Code List Validators - Set the valid values allowed.

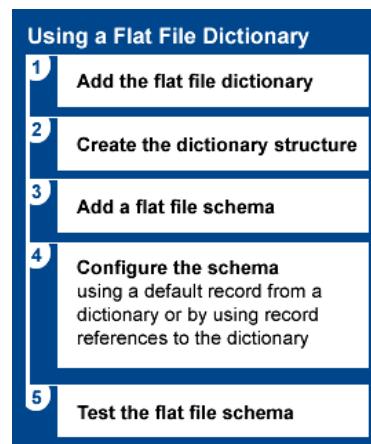


Flat File Dictionary

- A Flat File Dictionary is a reusable repository of record definitions.
- Record definitions can be referenced by multiple Flat File Schemas.
- If a record format is modified in a Flat File Dictionary, all of the Flat File Schemas that reference this record are automatically updated too.
- If a flat file has no record identifier, the record format MUST be defined in a Flat File Dictionary and set as the default record in the Flat File Schema definition.

Using a Flat File Dictionary

- The steps outlined in the figure to the right will be followed to parse a fixed length flat file with no record identifier.
- Complete Step 1 by clicking on File -> New and selecting Flat File Dictionary then give it a name.



Example Fixed Width Flat File

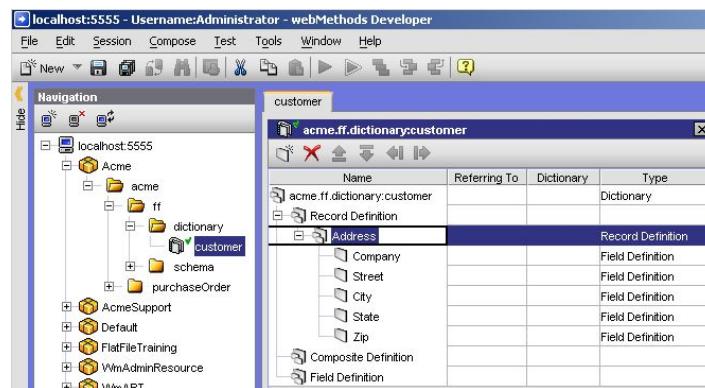
- The following records are contained in a fixed width flat file with no record identifier:

Acme Hammer Company	123 Wilson St.	Sacramento	CA95833
Johnson Supply Co.	456 Nadia Ave.	Seattle	WA98188
Garcia Hammer Emporium	4950 St. Moritz Blvd.	Dallas	TX75202
Mary Annes Hardware	601 Jefferson Ave.	Detroit	MI48230
Hard Times Development Co.	91235 Anderson St.	Boston	MA02125

- In a fixed width flat file, fields are identified by the number of bytes from the start of the record and the length of the field.

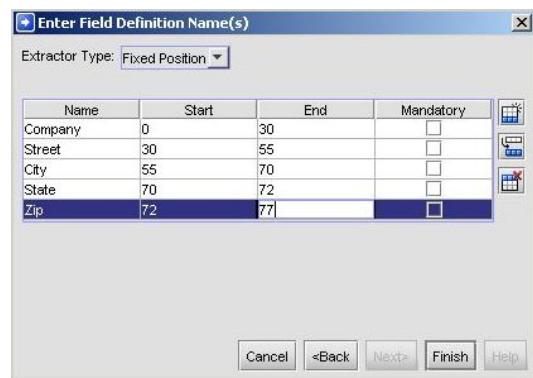
Step 1: Create the Dictionary Structure

- Creating the records and fields in a Flat File Dictionary is very similar to creating them in a Flat File Schema.
- Below, there is a record called Address that contains the 5 fields in the fixed width flat file example.



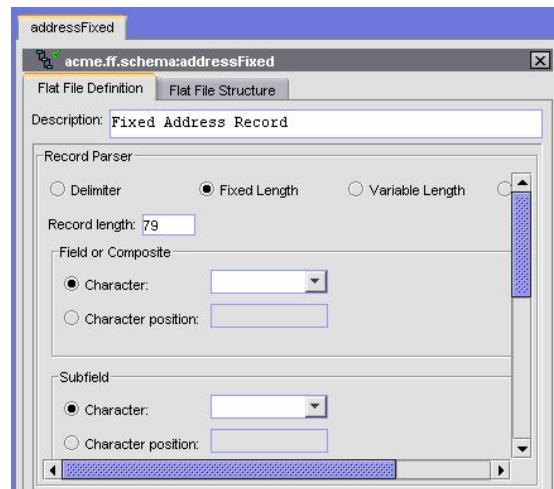
Step 2: Configure the Extractor

- Fixed Length files require a Fixed Position extractor.
- Fields and composites are extracted from the record based on their Start and End positions.
- Note that the End position of the Company field is the same as the Start position of the Street field.
- The first character in a record is position 0.
- The End position is the character immediately to the right of a field.



Step 3: Add a Flat File Schema

- A Flat File Schema has been created called addressFixed as shown to the right.
- The Record Parse type is Fixed Length.
- The Record length is 79.
There are 78 characters per record plus one character for the new line record delimiter.



Software AG Training | 10 - 22

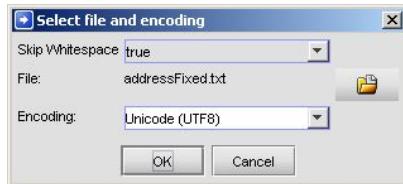
Step 4: Configure the Schema

- Since the example flat file has no record identifier, a Default Record must be Set.
- The Default Record must reference a Record in a Flat File Dictionary.
- Set the Default Record to the Address record that was created in the Flat File Dictionary.
- By default, 'Undefined Data' is set to False. Errors will be generated if there is undefined data in the record.

Properties	
acme.ff.schema:addressFixed	
Property	Value
Default Record	
Set	Set...
Delete	Delete
Dictionary	acme.ff.dictionary:customer
Name	Address
Settings	
Undefined Data	False
Areas	Choices...
Floating Record	
Permissions	
List ACL	<Default> (inherited)
Write ACL	<Default> (inherited)

Step 5: Test the Flat File Schema

- With the Flat File Schema shown in Developer, click the Run icon. 
- Use the Select File to Test icon and locate the addressFixed.txt file. 
- You can change the text encoding by selecting the Encoding drop down.
- Click on OK to run the test.
- View the results in the results panel.
- If errors occur, they will appear in the errors document list.



Variable Length Files

- Each record is preceded by two bytes that indicate the length of the record. Records in the flat file can have different lengths.
- The first two bytes of the record should contain a binary number representing the length of the record. Max record length is 0xFFFF (Hexadecimal) which is 65535.
- Records can have record identifiers.
- Fields can be identified by the number of bytes from the start of the record and the length of the field (fixed width).
- Fields can be separated by delimiters.

Example Variable Length File

- companyVariable.txt
- First two bytes contain the record length in binary format.
- Has Record IDs and field delimiter ',',.'

Hex Address	Hex Value	ASCII Value
00	00 2F 43 4F 4D 50 2C 54 68 65 20 44 6F 65 27 73	/ C O M P , The Doe's
10	2C 77 77 77 2E 64 65 65 72 2E 63 6F 6D 2C 37 30	, www . deer . com , 70
20	33 35 35 35 31 32 31 32 2C 52 65 74 61 69 6C 0D	3 5 5 5 1 2 1 2 , R e t a i l 0
30	0A 00 30 45 4D 50 2C 44 6F 65 2C 4A 6F 68 6E 2C	0 0 0 E M P , Doe , John ,
40	49 54 2C 31 32 2F 32 2F 31 39 39 30 2C 53 79 73	I T , 1 2 / 2 / 1 9 9 0 , S y s
50	74 65 6D 20 41 64 6D 69 6E 69 73 74 72 61 74 6F	t e m A d m i n i s t r a t o
60	72 0D 0A 00 2D 45 4D 50 2C 44 6F 65 2C 42 61 6E	r 0 0 0 - E M P , Doe , Jan
70	65 2C 45 78 65 63 75 74 69 76 65 20 4F 66 66 69	e , E x e c u t i v e O f f i
80	63 65 2C 32 2F 32 30 2F 31 39 38 30 2C 43 45 4F	c e , 2 / 2 0 / 1 9 8 0 , C E 0
90	0D 0A 00 2B 45 4D 50 2C 44 6F 65 2C 4A 65 72 72	0 0 0 + E M P , Doe , J e r r
A0	79 2C 53 61 6C 65 73 2C 38 2F 31 35 2F 32 30 30	y , S a l e s , 8 / 1 5 / 2 0 0
B0	34 2C 53 61 6C 65 73 20 41 67 65 6E 74 0D 0A	4 , S a l e s A g e n t 0 0

11

Web Services

Objectives

- At the end of this section, you will be able to:
 - create Web Service provider nodes
 - use Web Service consumer nodes
 - work with custom fault documents

Web Services

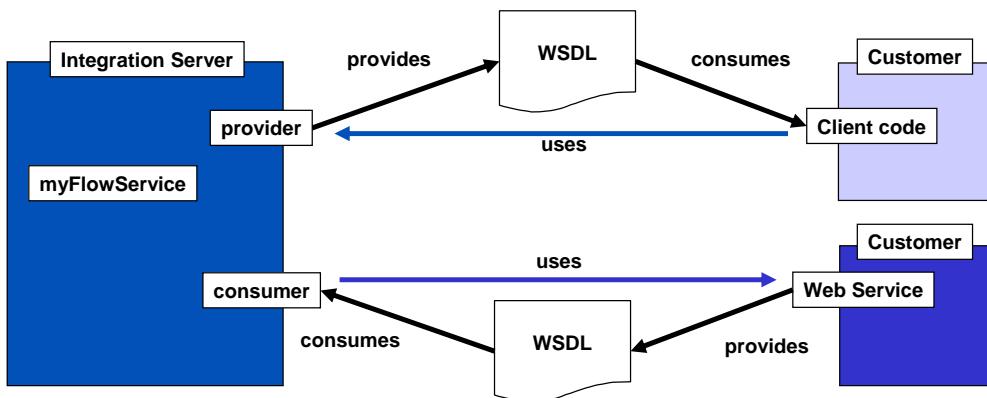
- Software components used as building blocks for distributed systems:
 - collection of functionality bundled as a addressable unit
 - used to expose higher-order functionality
 - designed for interoperability and reuse
- Web services are:
 - self-describing
 - accessible by open protocol
 - discoverable
 - one possible implementation of a Service Oriented Architecture
- The webMethods Integration Platform has always been a Service-Oriented Architecture (SOA)!

The Web Service Standards

- The “Web Service” uses vendor-independent, multiplatform standard formats and protocols.
- The core Web Service specifications standards used in the Integration Server are:
 - Web Service Description Language (WSDL)
 - Simple Object Access Protocol (SOAP) 1.1 and 1.2
 - Universal Description, Discovery and Integration (UDDI) v3
- IS also supports:
 - WS-Security 1.0
 - WS-I Basic Profile 1.1
 - WS-I Simple SOAP Binding Profile 1.0

Web Services Descriptor Overview

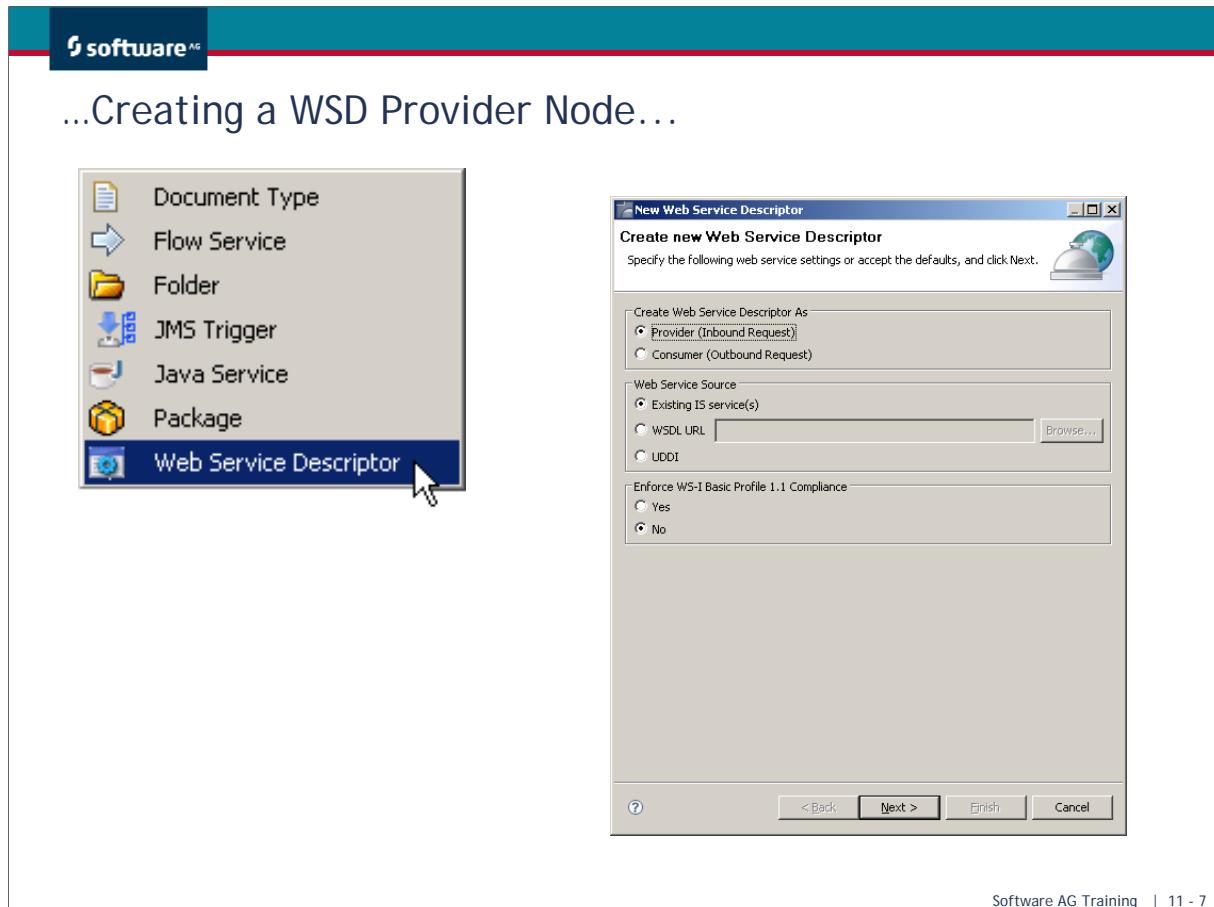
- WSD Provider node
 - Used to expose an IS service or set of IS services as web services
- WSD Consumer node
 - Used to consume a web service



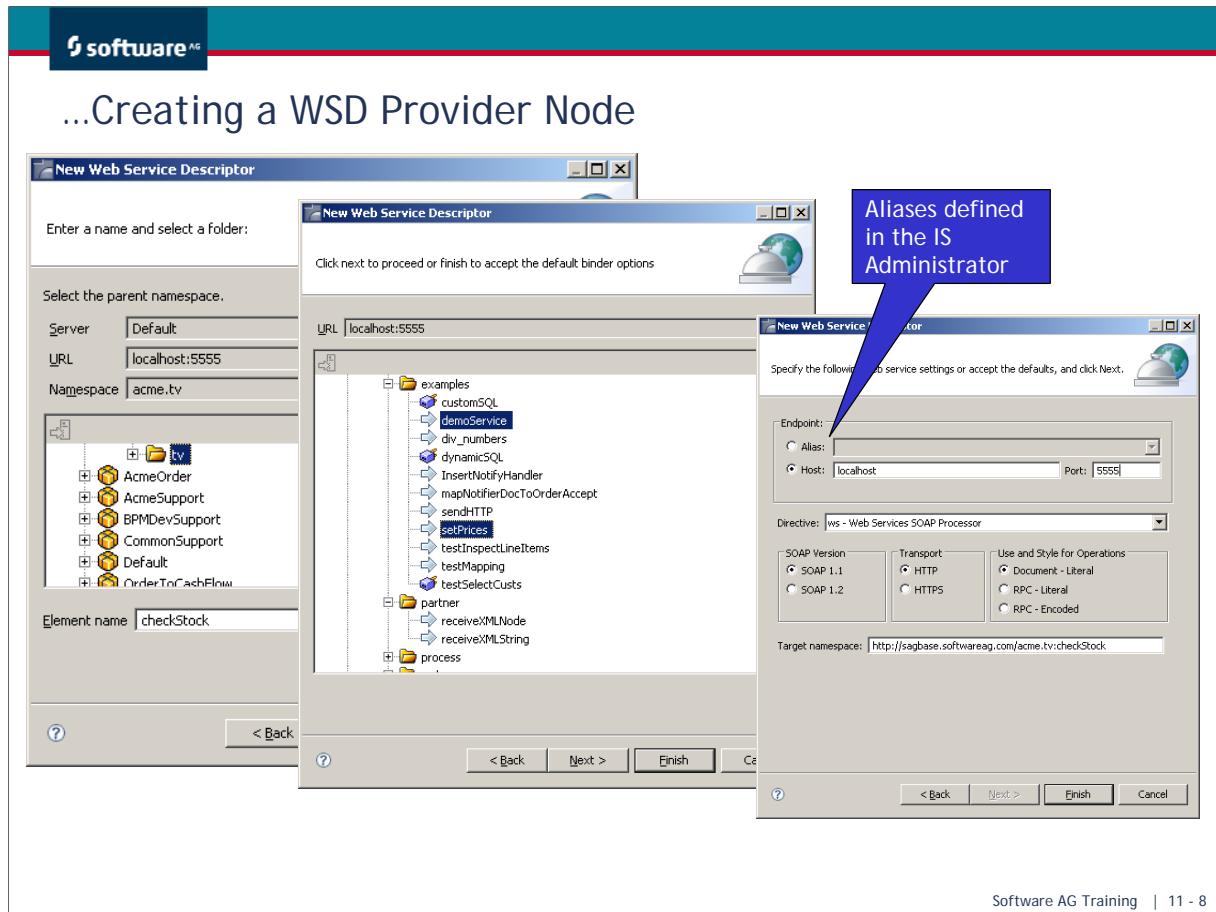
Creating a WSD Provider Node...

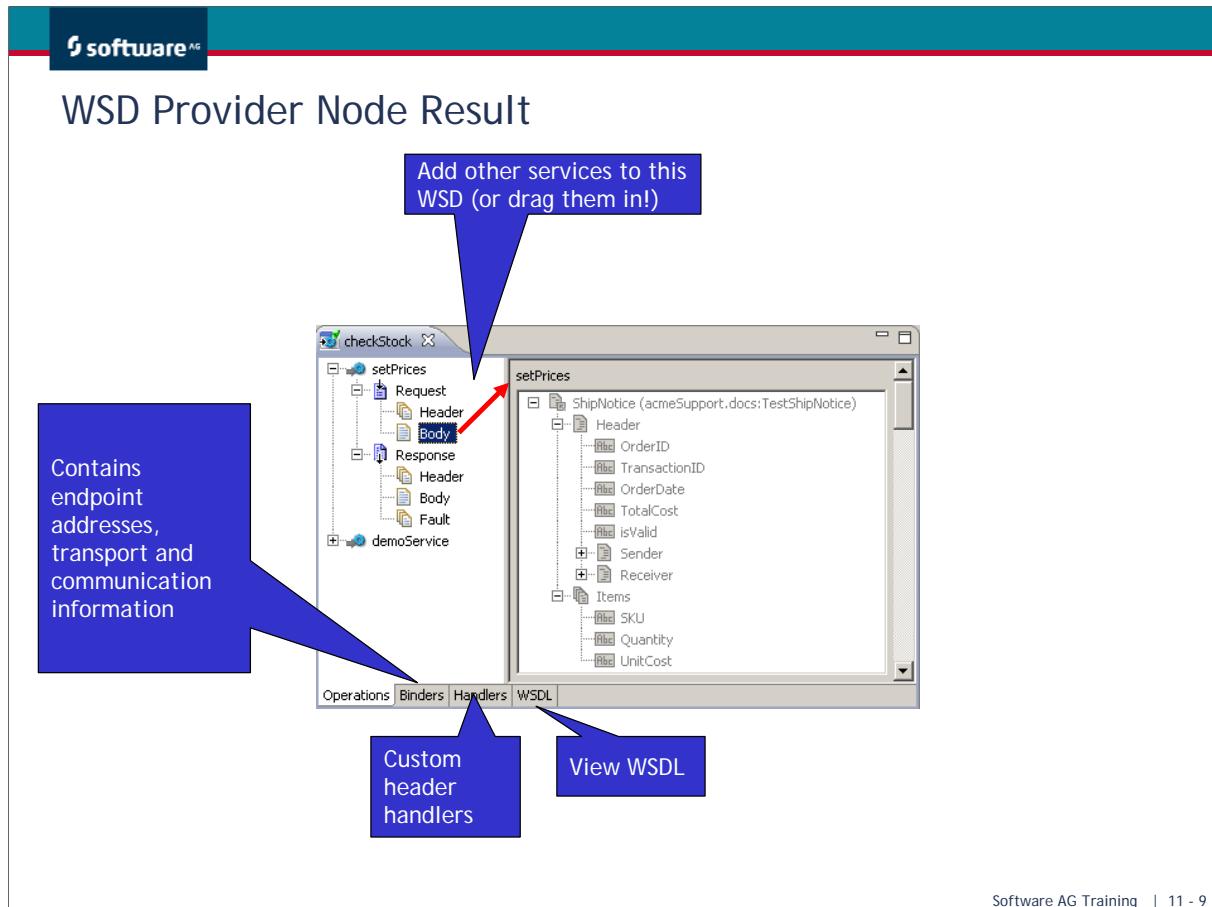
- Provider nodes can be created from:
 - Existing IS services
 - WSDL file
 - WSDL from UDDI registry
- At time of creation, choose:
 - Type
 - Source
 - Interoperability compliance
 - Alias endpoint
 - Processor to use
 - SOAP version
 - Transport and operation type

WSDL-based development assumes WSDL have been created outside of webMethods. The IS will generate flow service and document type stubs to match the WSDL contents.



Software AG Training | 11 - 7





WSDL Provider Node Properties

The screenshot shows the 'Properties' panel in the webMethods studio. The 'General' tab is selected, displaying the following properties:

Property	Value
Name	acme.tv:checkStock
Direction	Provider
WS-I compliance	False
Permissions	
Comments	
Target namespace	http://sagbase.softwareag.com/acme.tv:checkStock
WSDL URL	http://sagbase.softwareag.com:5555/ws/acme.tv:checkStock?WSDL
Namespaces	Click here to view Namespaces details
Attachment enabled	False
Pipeline headers enabled	False

Two callouts point to specific properties:

- A blue arrow points to the 'Name' property with the text: "Set at creation - can be changed at any time."
- A blue arrow points to the 'WSDL URL' property with the text: "URL for access to the WDSL (created at request time)"

Software AG Training | 11 - 10

The screenshot shows the Software AG Integration Server Administrator console. The left side features a navigation tree with categories like Server, Logs, Packages, Solutions, Adapters, Security, and Settings. Under Security, 'Web Services' is listed. The right side displays two tables under 'Settings > Web Service Endpoints'. The first table, 'Web Service Provider Endpoints List', shows three entries: 'defaultProvider' (alias HTTP, description default provider), 'HTTPS' (alias HTTPS), and 'JMS' (alias JMS). The second table, 'Web Service Consumer Endpoints List', shows two entries: 'defaultConsumer' (alias HTTPS, description default Consumer) and 'JMS' (alias JMS).

Web Service Provider Endpoints List	
Alias	Description
defaultProvider	default provider
HTTPS	
JMS	

Web Service Consumer Endpoints List	
Alias	Description
defaultConsumer	default Consumer
HTTPS	
JMS	

Software AG Training | 11 - 11

Assigning a Web Service Alias...

- At time of provider node creation

New Web Service Descriptor

Specify the following web service settings or accept the defaults, and click Next.

Endpoint:

Alias:

Host:

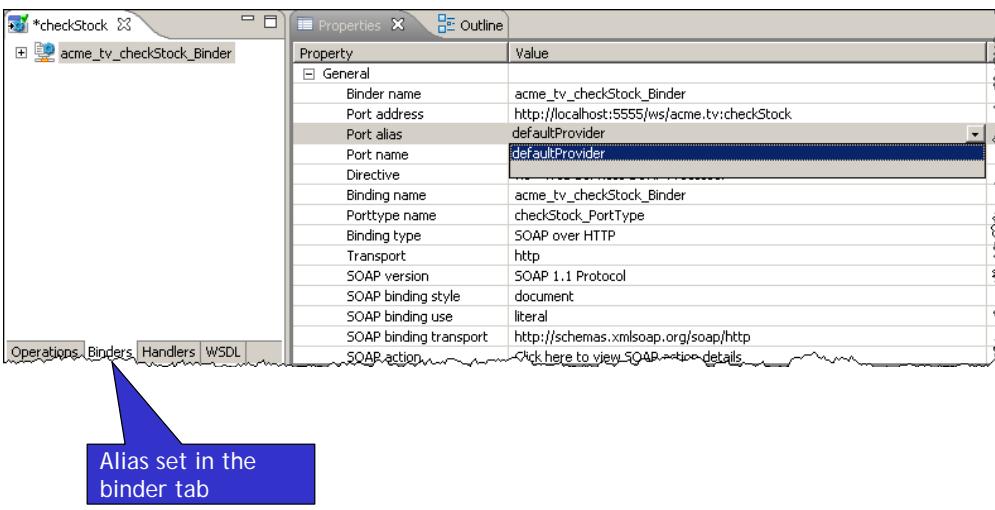
Actions: Back | Forward | Stop | Help | New Web Service | Web Services SSDL Generator

Set at creation - can be changed at any time.

Software AG Training | 11 - 12

...Assigning a Web Service Alias

- After the node has been created
 - Consistent for consumer and provider nodes



The screenshot shows the 'Properties' tab of the webMethods Studio interface. A blue callout box highlights the 'Port alias' field in the 'General' section of the properties grid, which is currently set to 'defaultProvider'. The 'Binders' tab is highlighted at the bottom of the interface.

Property	Value
General	
Binder name	acme_tv_checkStock_Binder
Port address	http://localhost:5555/ws/acme.tv:checkStock
Port alias	defaultProvider
Port name	defaultProvider
Directive	
Binding name	acme_tv_checkStock_Binder
Porttype name	checkStock_PortType
Binding type	SOAP over HTTP
Transport	http
SOAP version	SOAP 1.1 Protocol
SOAP binding style	document
SOAP binding use	literal
SOAP binding transport	http://schemas.xmlsoap.org/soap/http
SOAP action	Click here to view SOAP action details

Alias set in the binder tab

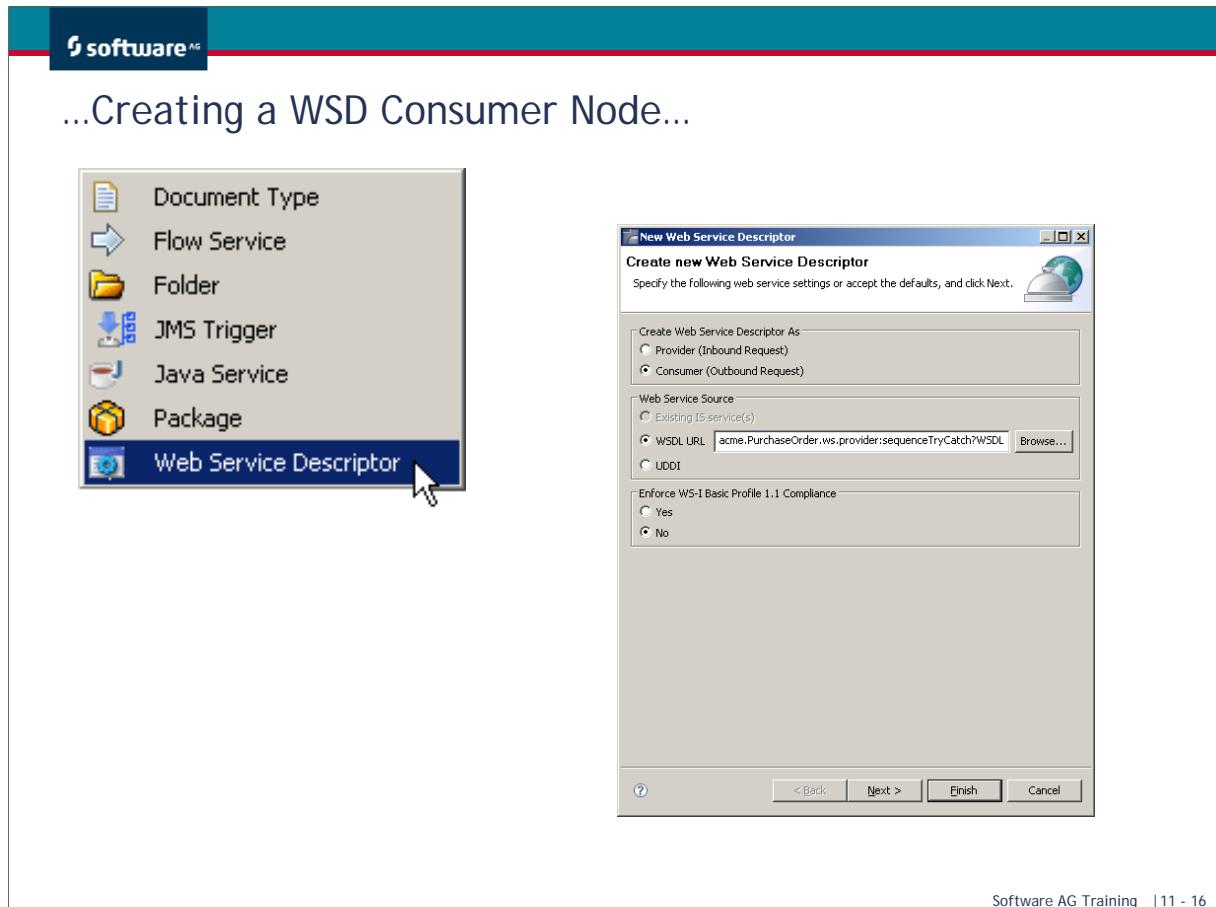
Software AG Training | 11 - 13

Testing the WSD Provider Node

- Using the WSDL URL to access the URL, generate client access code in your desired development tool.
 - Most vendors provide auto-generation of web service client code from WSDL files.
- No test run functionality for provider nodes from within Developer
 - Provider node is a mechanism to your flow or java service
 - Why not just test the service directly?
 - If needed to test from within Developer, create a WSD consumer node to test the provider node!

Creating a WSD Consumer Node...

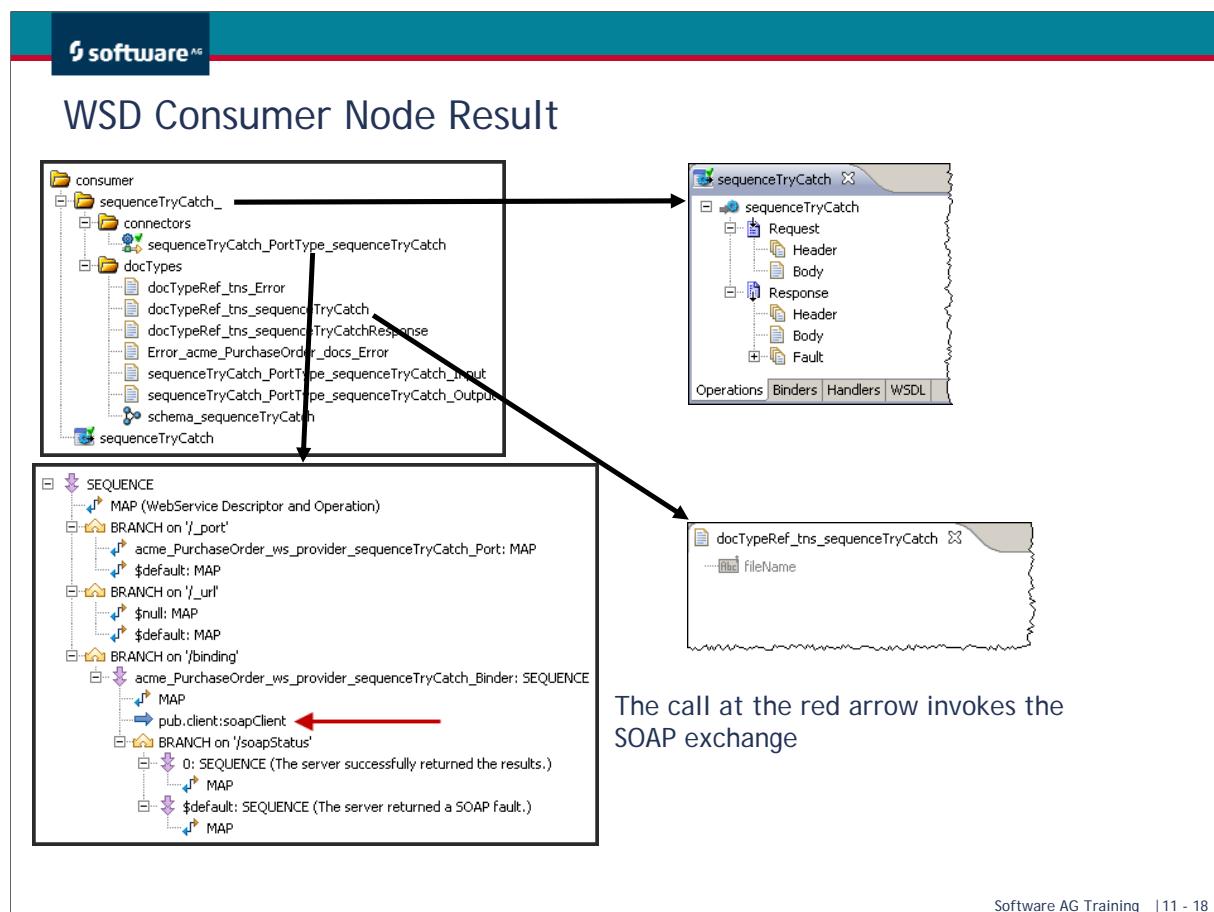
- Consumer nodes can be created from:
 - WSDL file (local or URL)
 - WSDL from UDDI registry
- At time of creation, choose:
 - Type
 - Source
 - Interoperability compliance
 - WSDL from which to create the consumer node



Software AG Training | 11 - 16

...Creating a WSD Consumer Node

The screenshot shows the 'New Web Service Descriptor' dialog box. At the top, it says 'Enter a name and select a folder:' with a globe icon. Below that, it says 'Select the parent namespace.' with fields for 'Server' (Default), 'URL' (localhost:5555), 'Package' (Acme), and 'Namespace' (acme.PurchaseOrder.ws.consumer). A tree view shows various namespaces like AcmeOrder, AcmeSupport, BPMDevSupport, CommonSupport, Default, OrderToCashFlow, PSUtilities, TMProcess, TNSupport, and V8TrainingCommon. An 'Element name' field contains 'sequenceTryCatch'. At the bottom are buttons for '?', '< Back / Next >', 'Finish', and 'Cancel'. The status bar at the bottom right says 'Software AG Training | 11 - 17'.



Testing the WSD Consumer Node

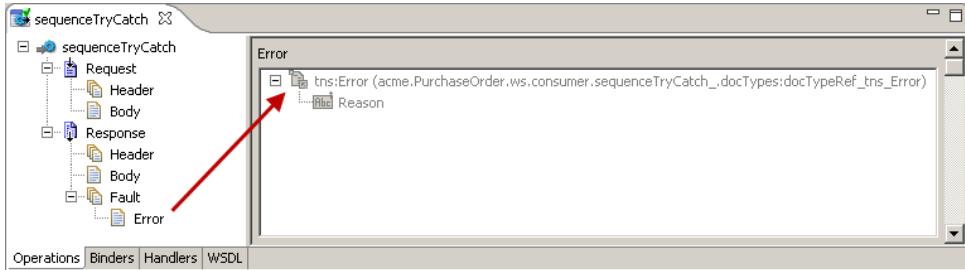
- Run the Web Service Connector in Developer.
- Provide necessary inputs (if required).
 - If executing against a webMethods web service, credentials must be provided (unless ACL is set to Anonymous)!
- Check the output in the Results window.

Custom Headers in WSD Nodes

- Used when:
 - A provider node needs to accept custom headers from a client
 - A provider node needs to return custom headers to a client
 - A consumer node needs to send custom headers to a web service
 - A consumer node needs to accept custom headers in a response
- Custom headers can not be processed without a header handler
 - Integration Server ships with a WS-Security header handler
 - Custom handlers are built in Java services

Custom Fault Documents in WSD Node

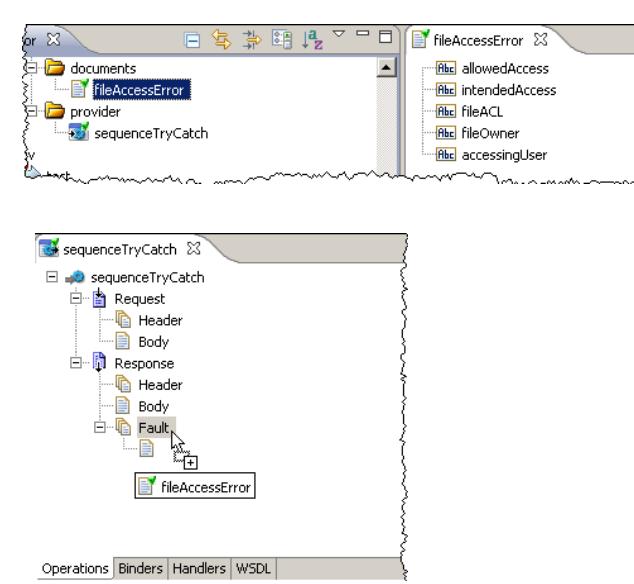
- Custom fault element returned as part of a web service response
 - soap:fault element
- Auto-generated if defined in a WSDL used to generate a WSD node
- Can be added after the WSD consumer was created



The screenshot shows the 'sequenceTryCatch' editor window. On the left, there's a tree view with nodes: sequenceTryCatch > Request > Header, Body; sequenceTryCatch > Response > Header, Body > Fault > Error. On the right, a panel titled 'Error' displays a single item: 'tns:Error (acme.PurchaseOrder.ws.consumer.sequenceTryCatch_.docTypes:docTypeRef_tns_Error)'. A red arrow points from the text 'Custom Fault Documents in WSD Node' in the slide content to this 'Error' item in the editor. At the bottom of the editor window, there are tabs: Operations, Binders, Handlers, and WSDL.

Adding Custom Fault Documents to a WSD

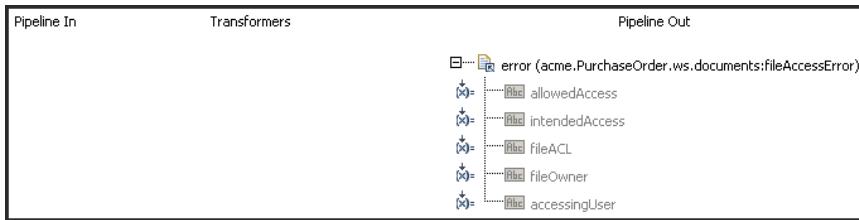
- Build IS document with desired fields
- Add the document(s) using drag and drop to the Response / Fault documentList



Software AG Training | 11 - 22

Custom Fault Documents

- Make sure your fault document is populated in the underlying service!
- When the SOAP processor recognizes the identified fault document type in the response, it will return it as a fault element.



Software AG Training | 11 - 23

This page intentionally left blank.

12

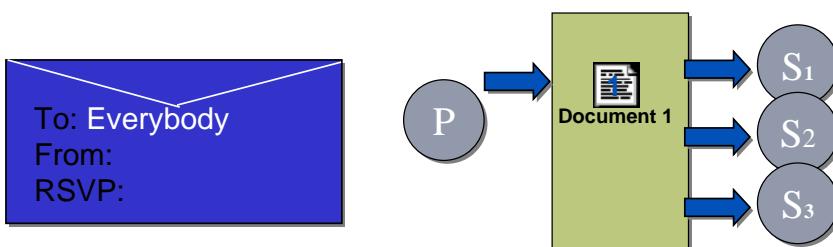
Broker Messaging

Objectives

- At the end of this section, you will be able to:
 - publish documents via the Broker
 - subscribe to documents via the Broker

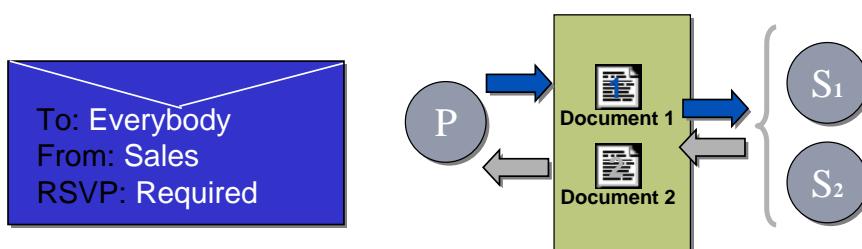
Publish

- Publishing documents to Broker for potential subscription
 - Clients who publish documents do not know who receives them
- Any subscriber can receive and process the document
 - Clients must subscribe to receive published documents
- Published documents are routed to **all** clients subscribing to that document type.



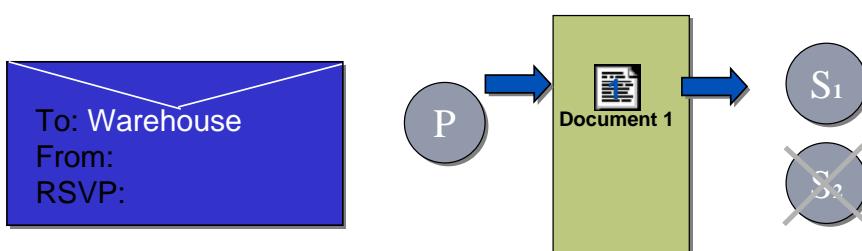
Publish & Wait

- Documents are published as usual but replies are expected from subscribing clients.
- Replies are delivered (not published) to the publisher
- The publisher does not need to subscribe to the response document.



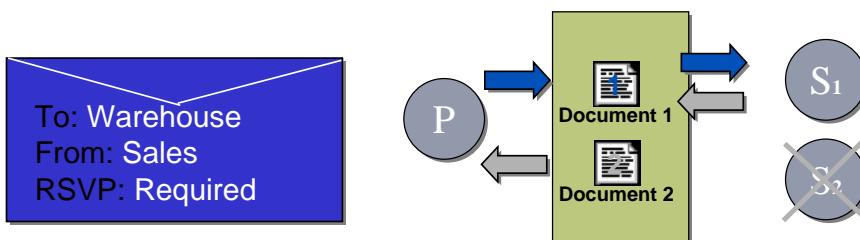
Deliver

- Documents are published to one specific client only.
 - All other subscriptions are bypassed.
- Publisher must know who is to receive the document.



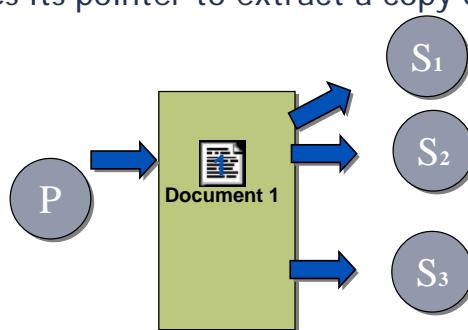
Deliver & Wait

- Documents are published to one specific client, and a reply is expected from that client.
 - All other subscriptions are bypassed.
- The Reply is delivered to the publisher.
 - The publisher does not need to subscribe to the reply document.



Subscribing

- Receiving published documents.
- Client subscribe to receive specific documents.
- When a publisher publishes a document, a subscription pointer to the document will be placed in the inbound queues of all subscribers.
- Each subscriber uses its pointer to extract a copy of the document from the queue.



Publisher/Subscriber Services

- Publish
- Publish & Wait
- Deliver
- Deliver & Wait
- Wait For Reply
- Reply

pub.publish:publish

pub.publish:publishAndWait

pub.publish:deliver

pub.publish:deliverAndWait

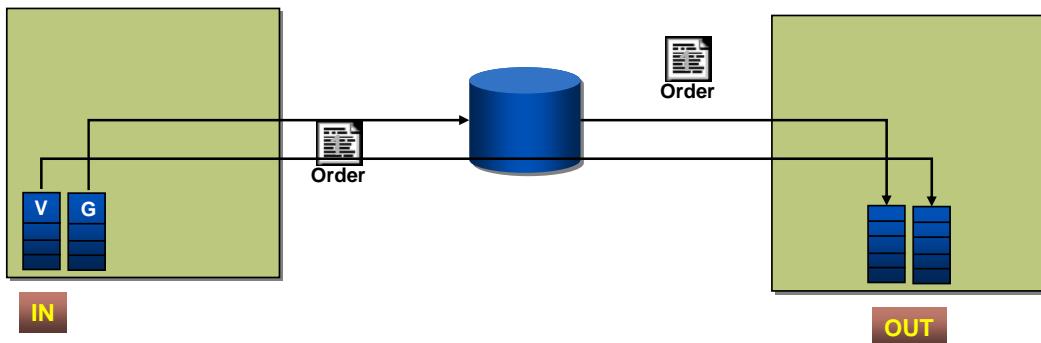
pub.publish:waitForReply

pub.publish:reply

- Subscribers in Broker Pub/Sub are Broker Triggers in the IS

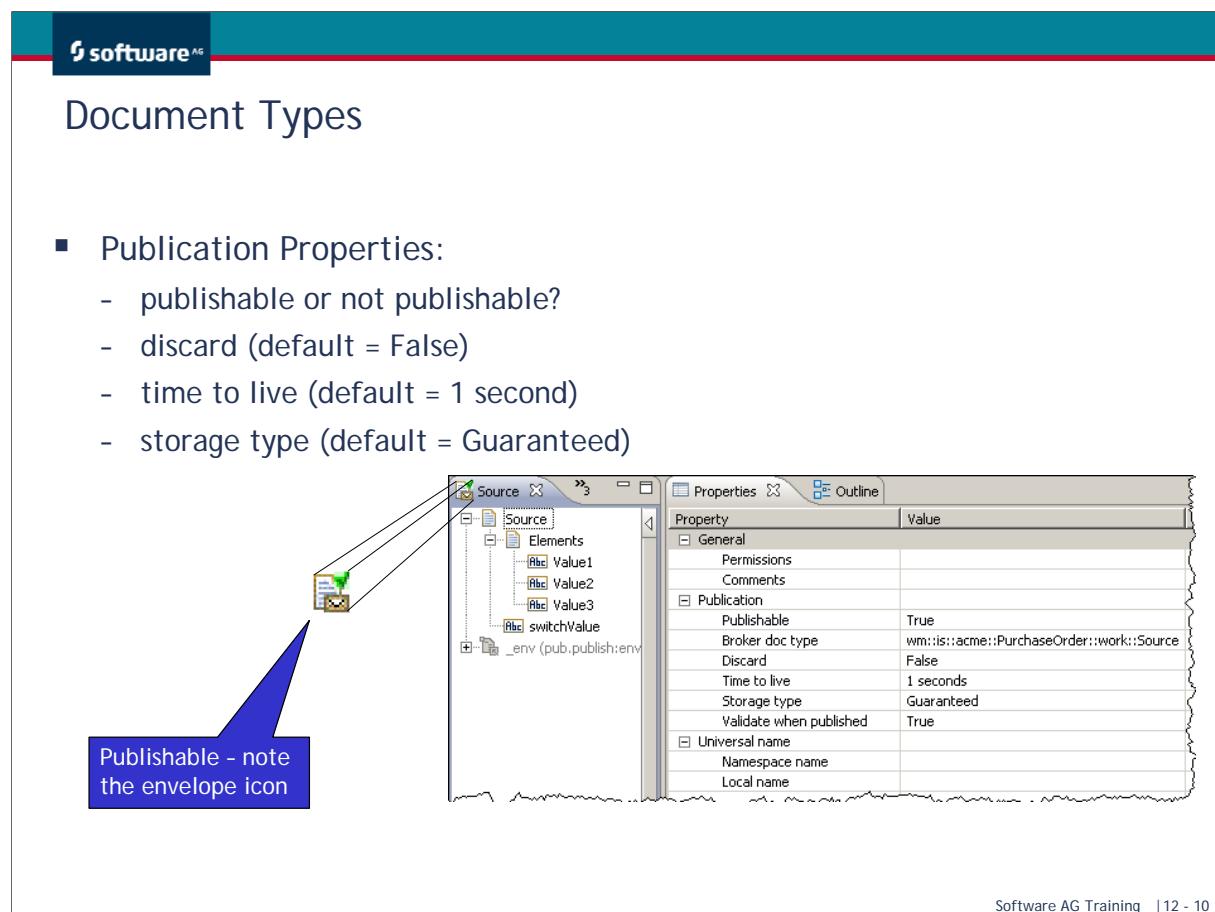
Document Storage Types

- Volatile - fast
 - data is stored in memory.
 - data is lost if the system shuts down.
- Guaranteed - slower
 - data is stored to disk.
 - no data should be lost, even if the system shuts down.



Document Types

- Publication Properties:
 - publishable or not publishable?
 - discard (default = False)
 - time to live (default = 1 second)
 - storage type (default = Guaranteed)



Publishable - note the envelope icon

Property	Value
General	
Permissions	
Comments	
Publication	
Publishable	True
Broker doc type	wm::is::acme::PurchaseOrder::work::Source
Discard	False
Time to live	1 seconds
Storage type	Guaranteed
Validate when published	True
Universal name	
Namespace name	
Local name	

Software AG Training | 12 - 10

Subscription & Trigger



- Clients can receive publishable documents in two ways:
 - subscription to the document type
 - delivery of the reply document (in response to your publishAndWait)
- Subscriptions are created via Trigger.
- Triggers define:
 - what Documents to listen for
 - what service to invoke when the Document arrives

Example: Create a handling Service

- Trigger will place the document in the input pipeline:
 - document will have a fully-qualified name.
 - to receive it correctly, you must define it with the same fully-qualified name!

Create a referenced document as input

Use fully-qualified Namespace name for variable!

Software AG Training | 12 - 12

Triggers in Detail

- File | New | Trigger
- Properties:
 - which Document Type(s) to subscribe to?
 - what Filters to use when subscribing?
 - which Service to invoke on receipt?

Invoked automatically by IS when matching documents arrive

The screenshot shows the Software AG webMethods IDE interface. On the left is a 'Navigation' tree with various project components like 'modeler', 'recordLibrary', 'scheduleStartupShutDown', 'testPubSub', 'chainReaction', 'pubAndNet', 'simpleJoin', and 'simplePubSub'. Under 'simplePubSub', there are nodes for 'handleCustomer', 'pubCustomer', and 'trigCustomer'. The 'trigCustomer' node is selected. To its right is a detailed configuration window titled 'trigCustomer' with the sub-title 'demoV6.testPubSub.simplePubSub.trigCustomer'. The window has tabs for 'Condition1' (selected), 'Service', 'Document types and filters', and 'Script'. The 'Condition1' tab shows a table with one row: Name: Condition1, Service: demoV6.testPubSub.simplePubSub.handleCustomer, Document Types: Customer, Join Type: N/A. The 'Service' tab shows the service details. The 'Document types and filters' tab shows a table with one row: Document Type: demoV6.testPubSub.simplePubSub.Customer and a 'Filter' field.

software AG

Filters

- Specify conditions based on the contents of a document.
- All elements of a document are available for filtering.
- Use lexical operators to set conditions.
- Example: Consumer orders where Sender is "ACME".

The screenshot shows the webMethods interface. On the left, there is a document tree for a purchase order. The tree includes nodes for Header (with fields like OrderID, TransactionID, OrderDate, TotalCost, IsValid), Sender (with ID), Receiver (with ID), and Items (with SKU and Quantity). On the right, a dialog box titled 'Condition1' is open, showing a service selection of 'TestPubSub:TestFilterSub'. Below it is a table for defining filters:

Document Type	Filter
acme.purchaseOrder.docs:OrderCanonical	%Header/Sender/ID% L_EQUALS "ACME"

Software AG Training | 12 - 14

Joins

- A single Trigger may subscribe to multiple document types.
- The trigger will only invoke the service when the Join condition is met:
 - AND
 - all of the subscribed document types have been received.
 - OR
 - any of the subscribed document types have been received.
 - XOR
 - only one of the subscribed document types has been received.

Joins - Watch Out for Activation IDs

- Joins work only if the activation IDs are the same.
- The publishing service must set the activation IDs

The screenshot shows the webMethods Developer interface. The left pane is the 'Navigation' tree, which includes nodes for 'localhost:5555', 'Acme', 'purchaseOrder', 'adapters', 'docs', 'request', 'Validation', 'maps', 'notifiers', 'utils', and 'work'. The right pane is the 'Validation' editor for a file named 'acme.purchaseOrder.docs:Validation'. This editor displays an XML structure with nodes like '_env (envelope)', 'activation', 'appLastSeqn', 'appPassword', 'appSeqn', 'appUserName', 'businessContext', 'controlLabel', 'errorsTo', 'errorRequestsTo', 'locale', and 'maxResults'. A blue callout box with the text 'Set the activation in the _env document' points to the 'activation' node in the validation tree. The top bar of the interface has the 'software AG' logo.

The screenshot shows the webMethods Test Tool: Developer interface. The main window has a toolbar at the top with various icons. Below the toolbar is a navigation pane titled "Navigation" which lists a hierarchy of integration components under "localhost:5555". A blue callout box points to the "Validation" document type under the "purchaseOrder" folder. Another blue callout box points to the "Run test for 'Course_BookOrder'" dialog box, which is overlaid on the main window. This dialog box contains several radio button options for publication actions, with the first option ("Publish locally to this Integration Server") selected. The status bar at the bottom right of the interface displays "Software AG Training | 12 - 17".

Test Tool: Developer

Validation

acme.purchaseOrder.docs:Validation

Valid

Run test for 'Course_BookOrder'

Choose the type of Publication action:

Publish locally to this Integration Server

Publish locally to this Integration Server and wait for a Reply

Publish to the Broker

Publish to the Broker and wait for a Reply

Deliver to a specific Client

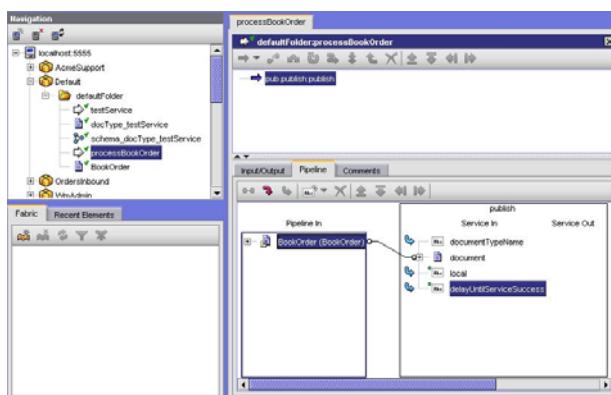
Deliver to a specific Client and wait for a Reply

Cancel <Back Next> Finish Help

Double-click the document type, then run

Publishing Example

- Set the parameter `documentTypeName` of the full namespace name. Example `defaultFolder:BookOrder`
- Set Value on **local** to true or false
- Set `delayUntilServiceSuccess` to true or false



Local publish can be expensive in terms of resources.

local = false is preferred

The screenshot shows the webMethods Integration Workshop interface with two pipeline examples:

- 1. PublishAndWait Example:** This pipeline example is titled "demoV6.testPubSub.pubAndWaitpubWaitCustRequest". It includes a "pub.publish:publishAndWait" step with the condition "(Ask For CustName SYNC)". A callout box states: "Setting the receive document is optional". A blue arrow points from this box to the "Input for 'receiveDocumentTypeName'" dialog window.
- 2. reply Pipeline:** This pipeline example is titled "demoV6.testPubSub.pubAndWaitreplyToCustRequest". It includes steps like "demoV6.adapters.getCustName", "MAP", "pub.flow:debugLog", and "pub.publish:reply". A callout box states: "If set, define the reply document as required by the Publish service". A blue arrow points from this box to the "Input for 'receiveDocumentTypeName'" dialog window.

Input for 'receiveDocumentTypeName'

receiveDocumentTypeName demoV6.documentsToPlayWith:CustomerReply
 Overwrite pipeline value
 Perform variable substitution

Software AG Training | 12 - 19

This page intentionally left blank.

13

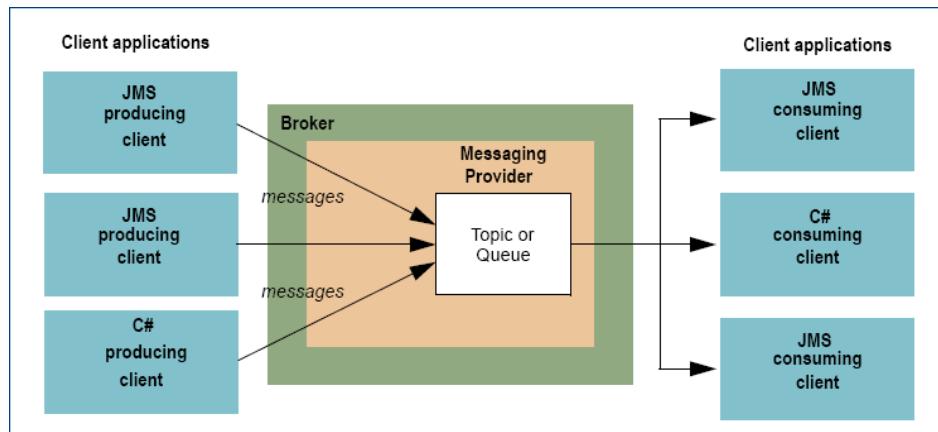
JMS Messaging

Objectives

- At the end of this section, you will be able to:
 - understand JMS messaging options
 - publish documents via a JMS provider
 - subscribe to documents via a JMS provider

Broker and JMS

- Broker JMS provider implements JMS 1.1
- Supports IS JMS clients, standalone coded clients, application servers



Java Message Service (JMS) Definition

- Java Message Service (JMS) API is a messaging standard:
 - Allows application components to create, send, receive, and read messages.
 - Based on the Java Platform, Enterprise Edition (Java EE)
 - Enables distributed communication that is loosely coupled, reliable, and asynchronous.

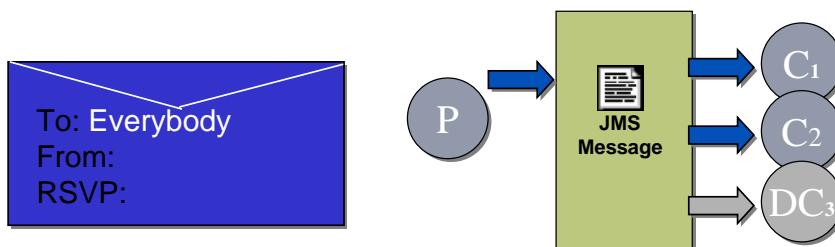
Software AG Training | 13 - 4

Documentation can be found in the
C:\webMethods6\Broker\doc\BrokerJMSProviderProgrammersGuide.pdf file.

The library of JMS services can be found in the C:\webMethods6\Broker\doc\jms_api\index.html file.

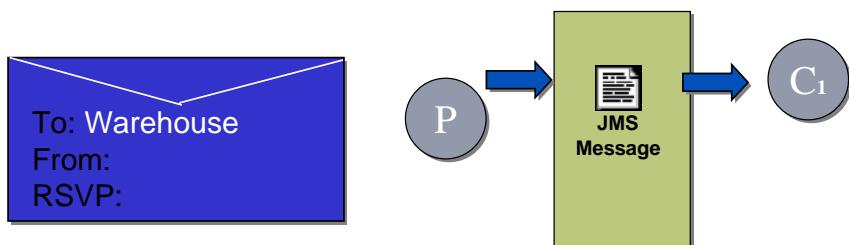
JMS Pub-Sub Messaging

- Message producer sends to a topic
- Multiple message consumers can subscribe to the same topic
- All subscribing consumers receive a copy of the message
- Supports durable subscribers
- Implemented as:
 - JMS Topic = Broker document type
 - Durable subscribers = guaranteed Broker client



JMS Point to Point Messaging

- Message producer sends to a message queue
- Message consumer retrieves message from queue
- Implemented as:
 - JMS message queue = guaranteed Broker client of same name



Administrator Tasks To Enable JMS Publish

- Integration Server Administration console
 - If using Broker for JMS, specify:
 - JMS Alias (using Native webMethods API) for use in JMS services
 - If using external JMS provider, specify:
 - JNDI Provider Alias
 - JMS Alias (using JNDI lookup) for use in JMS services
- My webMethods Server
 - Set up topics
 - Define Can Publish and Can Subscribe to IS-JMS client group
 - Set up queues

These admin tasks are already completed in the training environment.

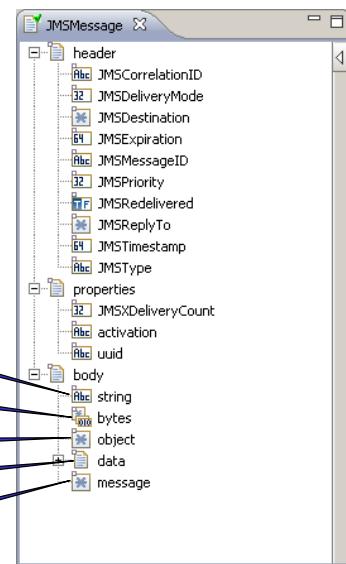


JMS Messaging Services and Support

- JMS Send
 - JMS Send & Wait
 - Create consumer
 - Receive
 - Acknowledge
 - Reply
 - Wait For Reply
 - Message document type
 - JMS trigger Specification
 - Subscribers in JMS Pub/Sub are JMS Triggers or consumers created in a Flow service
- | |
|------------------------|
| pub.jms:send |
| pub.jms:sendAndWait |
| pub.jms:createConsumer |
| pub.jms:receive |
| pub.jms:acknowledge |
| pub.jms:reply |
| pub.jms:waitForReply |
| pub.jms:JMSMessage |
| pub.jms:triggerSpec |

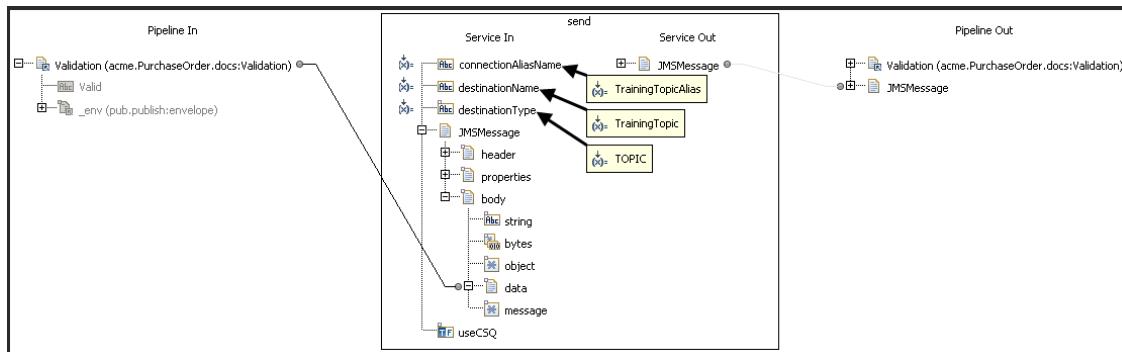
JMS Message Types

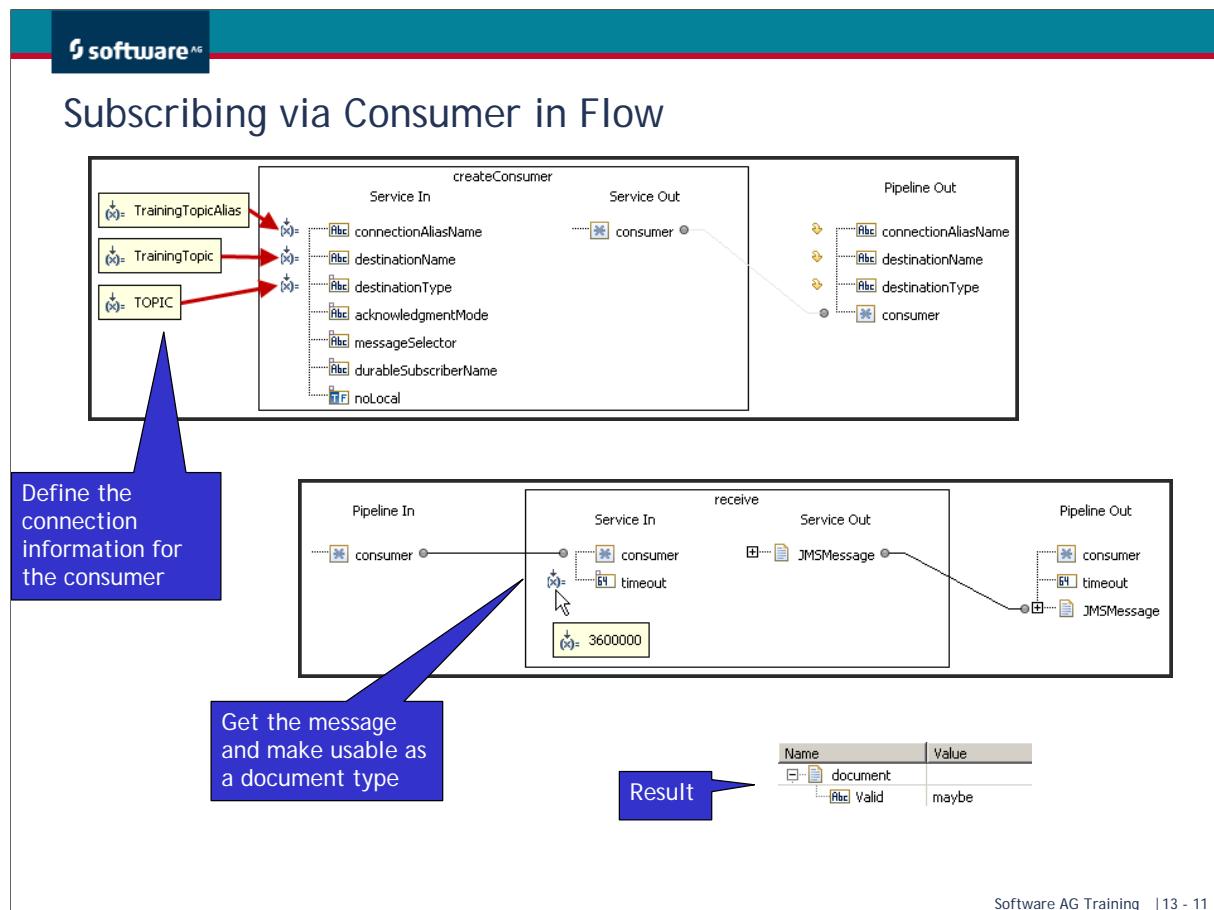
- A JMS Message is a Broker document.
 - Based on document type pub.jms:JMSMessage
- JMS Messages have:
 - JMS Headers
 - JMS Properties
 - Body contents, representing JMS message types
- JMS supports two message storage types:
 - Persistent, equivalent to Broker guaranteed
 - Non-persistent, equivalent to Broker volatile



Publishing a JMS Message

- Set connectionAliasName to alias from IS Administrator
- Set destinationName to name of topic or queue
- Set destination type to Topic or Queue (queue is default)
- Map the string, bytes, object, document, or stream to the body





Subscribing via JMS Trigger

- Subscribing trigger requires
 - Alias name (from IS Administrator)
 - Destination name (topic or queue name)
 - Destination type
 - Message routing rule (handling service and filter)

The screenshot shows the 'listenForValidationJMS' configuration dialog. At the top, it displays the JMS connection alias name as 'TrainingTopicAlias' and the JMS trigger type as 'Standard'. Below this, under 'JMS destinations and message selectors', there is a table with columns: Destination Name, Destination Type, JMS Message Selector, Durable Subscriber Name, and Ignore Locally Published. A single row is present with 'TrainingTopic' in the Destination Name column and 'Topic' in the Destination Type column. The 'JMS Message Selector' column is highlighted with a blue arrow and labeled 'Filter on JMSMessage properties'. Under 'Join type', the 'Any(OR)' option is selected. Below this, under 'Message routing', there is another table with columns: Name, Service, and Local Filter. A single row is present with 'Rule1' in the Name column and 'acme.PurchaseOrder.work:handleValidation...' in the Service column. The 'Local Filter' field is highlighted with a blue arrow and labeled 'Filter on message content'.

JMS Message Handling Service

- JMS trigger will place a pub.jms:JMSMessage document type in the input pipeline
- To receive it correctly, specify the pub.jms:triggerSpec as the input/output specification for your handling service

The screenshot shows the Pipeline Designer interface. At the top, there's a toolbar with tabs like Pipeline, Bookmarks, Tasks, Problems, and Package Explorer. Below the toolbar is a tree view labeled 'Pipeline In' which shows a 'JMSMessage (pub.jms:JMSMessage)' node with sub-nodes: header, properties, body, string, bytes, object, and message. A line connects the 'message' node to a service component below. The service component is titled 'documentToXMLString' and has two sections: 'Service In' and 'Service Out'. The 'Service In' section contains nodes: attrPrefix, document, nsDecls, addHeader, encode, documentTypeName, generateRequiredTags, generateNilTags, enforceLegalXML, dtdHeaderInfo, and bufferSize. The 'Service Out' section contains a single node: xmldata.

Software AG Training | 13 - 13

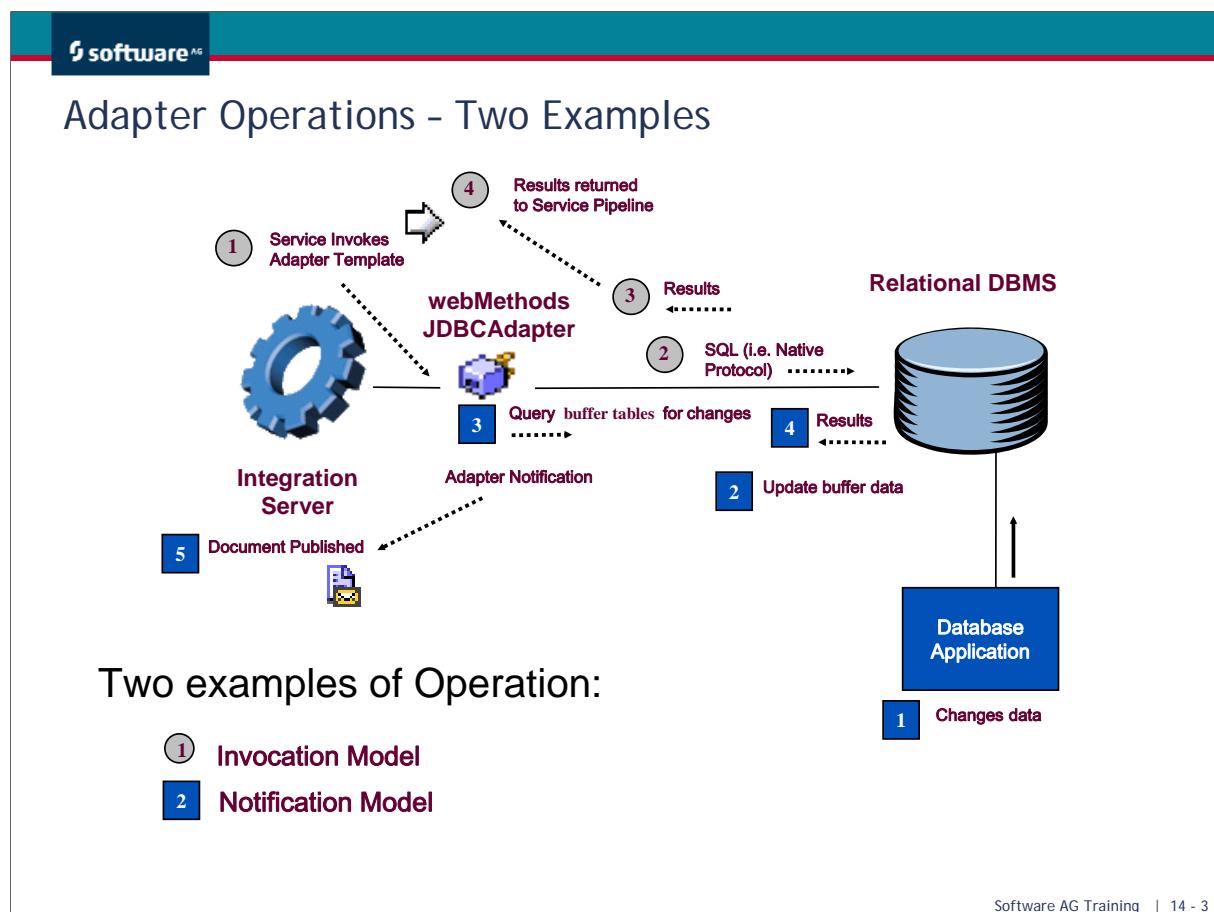
This page intentionally left blank.

14

JDBC Adapter

Objectives

- At the end of this section, you will be able to:
 - configure an Adapter Connection
 - create Adapter Services
 - create Notification Services



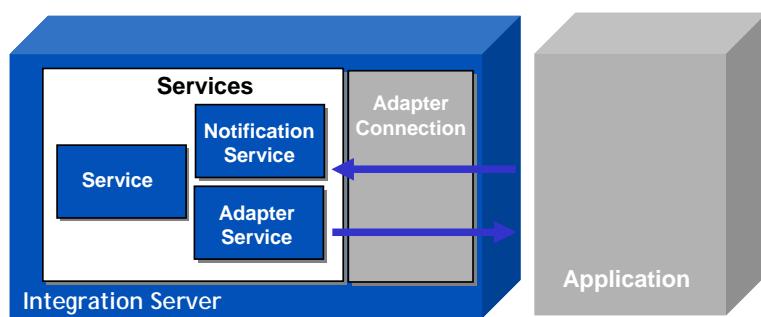
This diagram shows the two adapter communication models.

The **Invocation model** begins at the **Integration Server**. A service invokes an **Adapter service**. The Adapter service calls the resource, retrieves the results and returns the results into the Pipeline of the initial service invoke. For the DB Adapter above, you might use this model to update a customer address record in your database with data received from the customer via a Web Service.

The **Notification model** begins at the resource. Data is changed in the application. The Adapter polls the resource for changes (typically polling a buffer table). When the Adapter discovers changed data, an Adapter notification occurs. This notification publishes a notification document. For the DB Adapter example above, you might use this to publish changes to other interested systems when a customer service rep updates account information.

Adapter Runtime Architecture

- All webMethods components of an adapter are installed on the Integration Server:
 - Adapter connection
 - Adapter services
 - Adapter notification services
 - Parent flow services



Working with Adapters

The screenshot shows the Software AG webMethods IS Administrator interface. On the left, there's a navigation sidebar with links like Server, Logs, Packages, Solutions, Adapters, Security, and Settings. The 'Adapters' link is expanded, and 'JDBC Adapter...' is highlighted with a red oval. A grey arrow points from this oval to a detailed view window on the right.

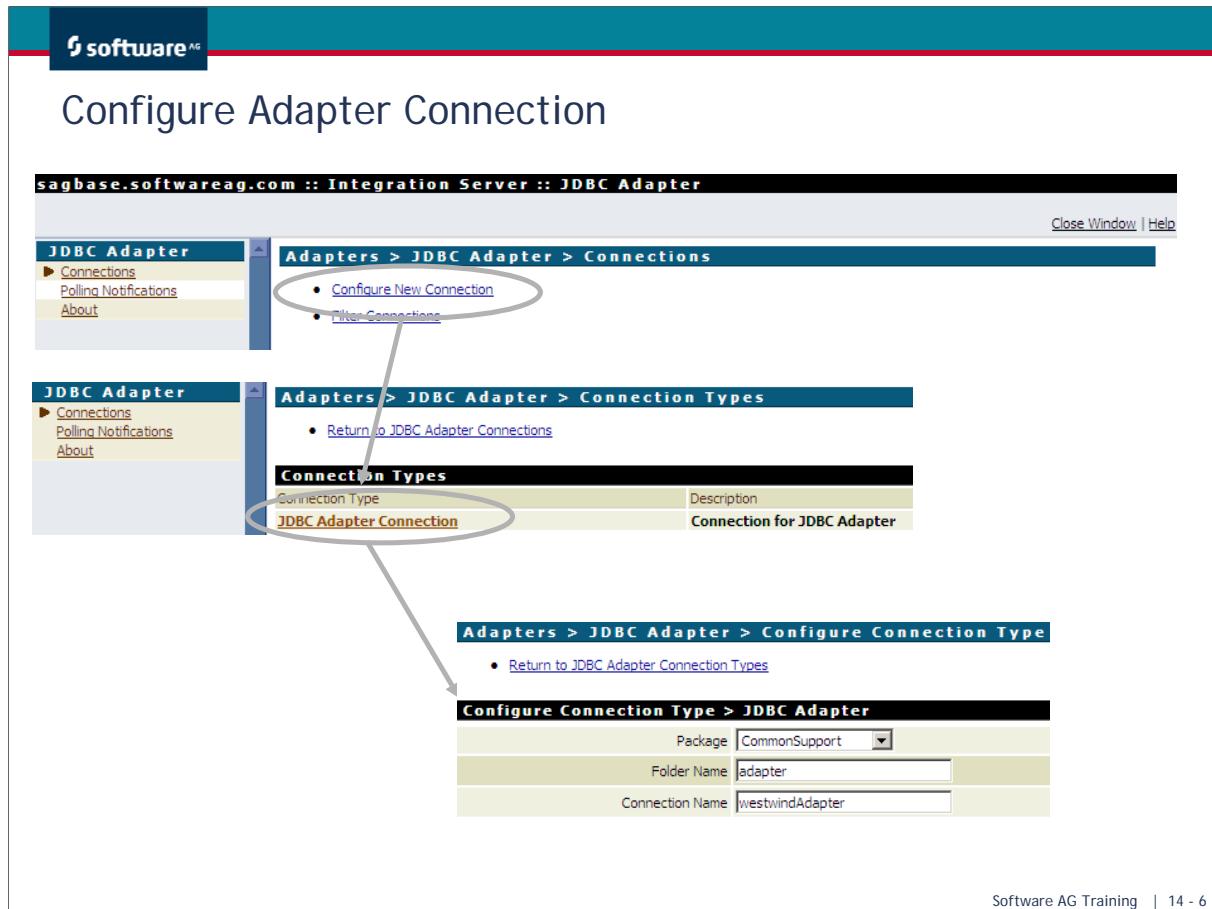
The detailed view window has a title bar 'Adapters > JDBC Adapter > Connections'. It contains two main sections: a list of actions ('Configure New Connection', 'Filter Connections') and a table titled 'Connections'.

Connections Table Data:

Connection Name	Package Name	Connection Type	Enabled	Edit	View	Copy	Delete
commonSupport.adapters:CAFShopAdapter	CommonSupport	JDBC Adapter Connection	No				
commonSupport.adapters:acmeAdapter	CommonSupport	JDBC Adapter Connection	Yes				
commonSupport.adapters:northwindAdapter	CommonSupport	JDBC Adapter Connection	No				

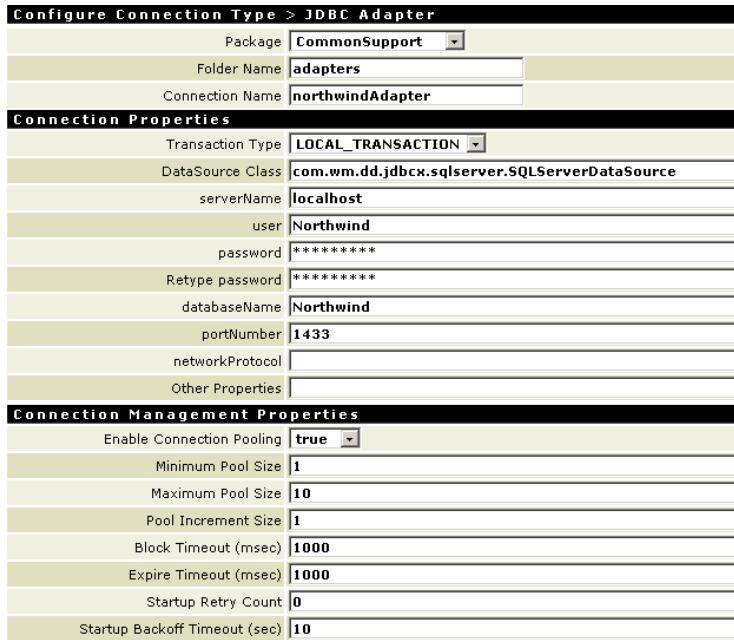
At the bottom right of the window, it says '1-3 of 3' and 'Software AG Training | 14 - 5'.

- webMethods Adapters are deployed as packages.
- Configured using the IS Administrator:
 - connection data
 - adapter specific parameters
- Services are then created using templates in Developer.



Software AG Training | 14 - 6

Example Connection



The screenshot shows the 'Configure Connection Type > JDBC Adapter' dialog box. It has three main sections: 'Connection Properties' (top), 'Connection Management Properties' (middle), and 'Other Properties' (bottom). The 'Connection Properties' section includes fields for Package (CommonSupport), Folder Name (adapters), Connection Name (northwindAdapter), Transaction Type (LOCAL_TRANSACTION), DataSource Class (com.wm.dd.jdbcx.sqlserver.SQLServerDataSource), serverName (localhost), user (Northwind), password (*****), Retype password (*****), databaseName (Northwind), portNumber (1433), networkProtocol (TCP/IP), and Other Properties (empty). The 'Connection Management Properties' section includes fields for Enable Connection Pooling (true), Minimum Pool Size (1), Maximum Pool Size (10), Pool Increment Size (1), Block Timeout (msec) (1000), Expire Timeout (msec) (1000), Startup Retry Count (0), and Startup Backoff Timeout (sec) (10).

- Specify package and namespace
- Define connection properties and transaction support
 - No
 - Local
 - XA
- Configure appropriate connection pooling

Service Templates



Adapter Service

- Once a connection is configured and enabled, service templates are available in Developer.
- Adapter Services use service templates to perform operations:
 - included with adapters
 - perform very specific operations with the target resource of the adapter
 - communicate with the resource
 - are aware of the expected format that the resource expects for input and output
 - example: JDBC Adapter templates
 - SelectSQL
 - InsertSQL
 - UpdateSQL
 - BatchInsertSQL
 - BatchUpdateSQL
 - DeleteSQL
 - CustomSQL
 - DynamicSQL
 - StoredProcedure
 - StoredProcedureWithSignature
 - ExecuteService

Creating an Adapter Service in Developer

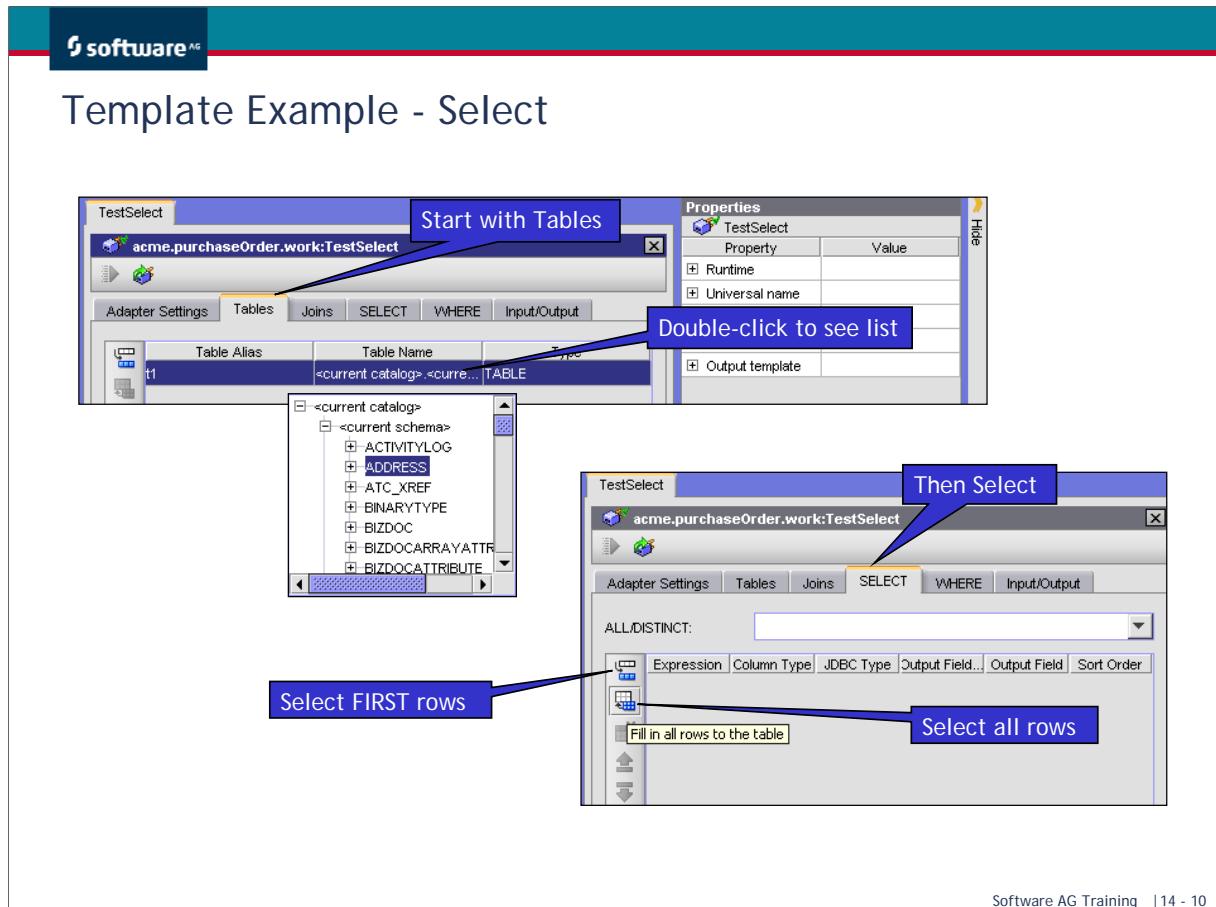
The screenshot shows the 'New Adapter Service' wizard in three steps:

- Step 1: Select an Adapter Type. Adapter Type: JDBC Adapter.
- Step 2: Select an Adapter Connection Alias. Adapter Connection Name: acmeSupport.adapters:acmeAdapter.
- Step 3: Select a Template Name. Adapter Service Template: SelectSQL.

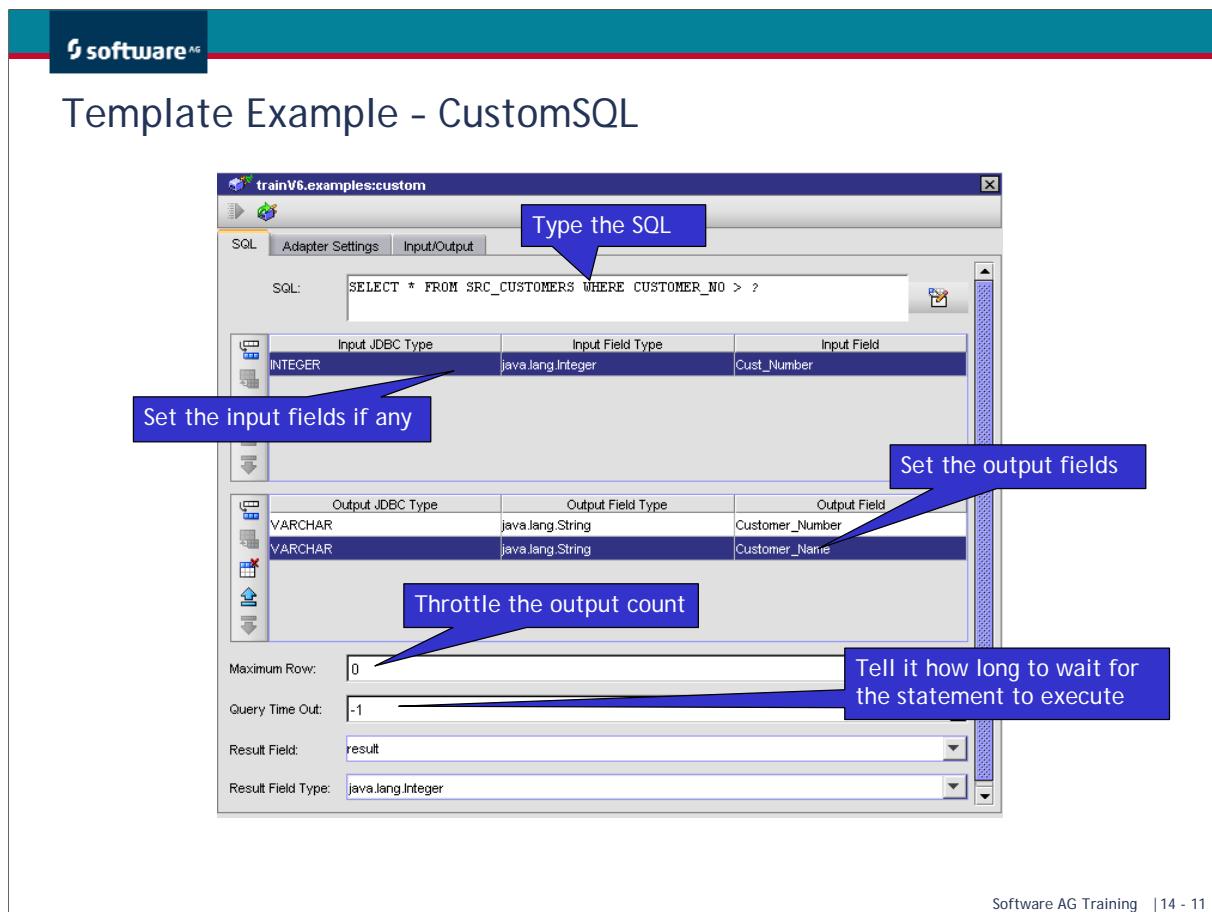
The main workspace shows the 'TestSelect' adapter service created under the 'AcmeSupport' category. The properties panel on the right lists the following properties:

Property	Value
Runtime	
Universal name	
Audit	
Permissions	
Output template	

Software AG Training | 14 - 9



Software AG Training | 14 - 10



Template Example - DynamicSQL

The screenshot shows the 'Dynamic SQL' configuration window. The 'SQL' field contains the query: `SELECT * FROM ${InSQL}`. A blue callout arrow points from the placeholder `${variable}` in the SQL field to a blue box containing the text: **In this case no inputs are needed**.

Output JDBC Type	Output Field Type	Output Field
VARCHAR	java.lang.String	Customer_Number
VARCHAR	java.lang.String	Customer_Name

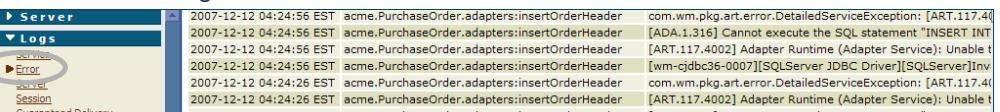
Configuration settings at the bottom:

- Maximum Row: 0
- Query Time Out: -1
- Result Field: result
- Result Field Type: java.lang.Integer

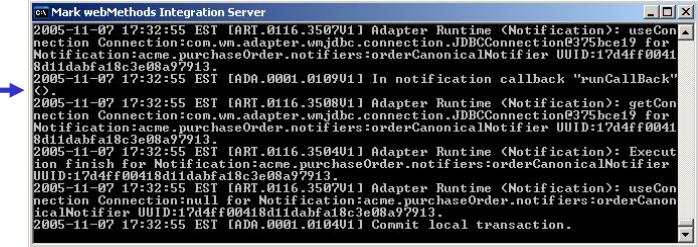
Software AG Training | 14 - 12

Troubleshooting

- Check the Error log and Server log in the Administrator for Adapter messages.
 - This example uses the JDBC Adapter, other adapters may write to different logs.



Server Log in Console



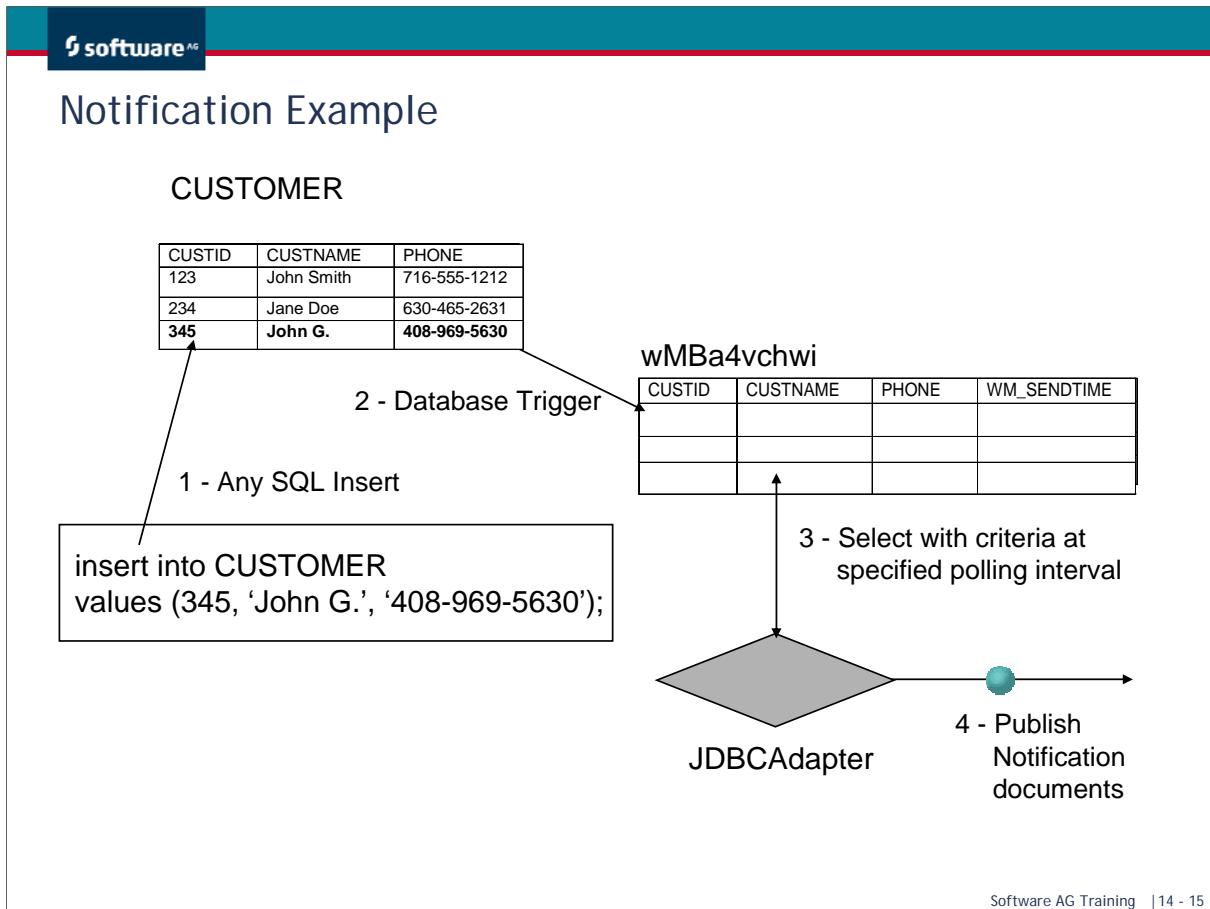
Software AG Training | 14 - 13

Notification



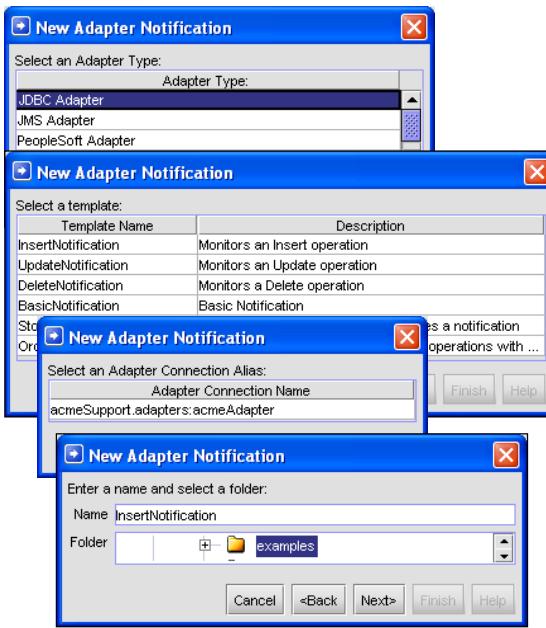
Adapter Notification

- Adapter publishes a selected document in response to a specific database activity.
 - record insertion, update, or deletion
- Best accomplished by attaching a database trigger to the table.
 - DB Triggers store data in a buffer table
 - adapter performs notification by periodically checking the buffer table for entries



The entries (rows) are deleted from the buffer table once they are placed on the Broker queues.

Creating an Adapter Notification Service...



1. Developer New...Adapter Notification, click Next.
2. Select Adapter Type, click Next.
3. Select Template Name, click Next.
4. Select Adapter Connection Name, click Next.
5. Enter name/folder info, click Next.
6. View new publishable Document Name, click Finish.
7. Configure Notification properties (table, etc.).
8. Create handling Flow Service.
9. Setup Trigger.

Software AG Training | 14 - 16

The screenshot shows the webMethods Developer interface. In the navigation tree on the left, under 'InsertNotificationPublishDocument', there is a folder named 'partner' which contains several items: Default, WmAdmin, WmAdminResource, WmART, WmBrokerAdmin, WmDB, WmDotNet, WmFlatFile, and WmFlatFileSet. A new item, 'InsertNotification', has been added to this folder. A blue callout box points to this newly created item with the text: 'Note the newly created publishable document type.'

In the main workspace, a notification configuration window is open for 'acmeSupport.examples:InsertNotification'. The 'Tables' tab is selected. A blue callout box points to the 'Base Name:' field with the text: 'Each table needs a unique name. Use this field to set a base name to force uniqueness.' Below this, a table lists resource types and their names:

Resource Type	Resource Name
Buffer Table	WMBbac0ftt
Trigger	WMTbac0ftt
Sequence	WMSbac0ftt

A blue callout box points to the bottom of the table with the text: 'To ensure uniqueness, the resource name combines the following elements:

- Resource prefix (WMB (buffer table), WMT (trigger) or WMS (sequence))
- The name you typed in the Base Name field
- A suffix, based on a system timestamp

Note: You cannot edit this name!'

Software AG Training | 14 - 17

...Creating an Adapter Notification Service

The screenshot illustrates the process of creating an Adapter Notification Service in the webMethods Integration Workshop. It consists of two main windows:

- Top Window:** Shows the 'InsertNotification' configuration screen. A blue arrow points from the 'Tables' tab to a callout box labeled 'Select a table.' The table 't1' is selected, with columns CUSTOMER_NO and CUSTOMER_NAME defined.
- Bottom Window:** Shows the 'InsertNotificationPublishDocument' configuration screen. A blue arrow points from the 'Publish Document' tab to a callout box labeled 'Select which fields to publish in the document.' The fields CUSTOMER_NO and CUSTOMER_NAME are selected for publication.

Software AG Training | 14 - 18

Scheduling a Notification

- Create a Notification in Developer.
- Go to Polling Notifications Screen (Adapter UI).
- Disable Notification (if enabled).
- Schedule the Notification.
 - Set Polling Interval
 - Set other options
- Enable Notification.

Scheduling a Notification in Administrator

Suspended - Stop polling, do not delete DB Trigger & Buffer Table
 Enabled - Create DB Triggers & Buffer Table
 Disabled - Stop polling, delete DB Triggers & Buffer Table

JDBC Adapter

- Connections
- Polling Notifications**
- About

Adapters > JDBC Adapter > Polling Notifications

- Suspend all enabled
- Enable all suspended
- Filter Polling Notifications

JDBC Adapter Polling Notifications

Notification Name	Package Name	State	Edit Schedule	View Schedule
acme.PurchaseOrder.notifiers:oderCanonicalNotifier	Acme	Enabled		
acmeSupport.examples:insertCustNotify	AcmeSupport	Disabled		

1-2 of 2

JDBC Adapter

- Connections
- Polling Notifications**
- About

Adapters > JDBC Adapter > Edit Notification Schedule

- Return to JDBC Adapter Notifications

Details for acme.PurchaseOrder.notifiers:oderCanonicalNotifier

Interval: (seconds)	10
Overlap:	<input type="checkbox"/>
Immediate:	<input type="checkbox"/>

Cluster settings

Coordination mode:	Standby
Max process time: (seconds)	180
Max setup time: (seconds)	180

Save Settings

Software AG Training | 14 - 20

15

Business Process Modeling



Objectives

- At the end of this section, you will be able to:
 - Understand Business Process Management in webMethods 8 Product Suite
 - Use the Designer to modify business processes
 - Generate business processes
 - Update generated services for execution

Business Process Management

- BPM: the ability to define, manage, analyze, and optimize interaction between systems, partners, and personnel.
 - Doing so gets the right information, approvals, responses at the right time to the right people.
- Business processes can represent any set of inter-related tasks in a corporation:
 - handling a new sales order
 - processing for a new employee
 - product lifecycle activities
 - etc.

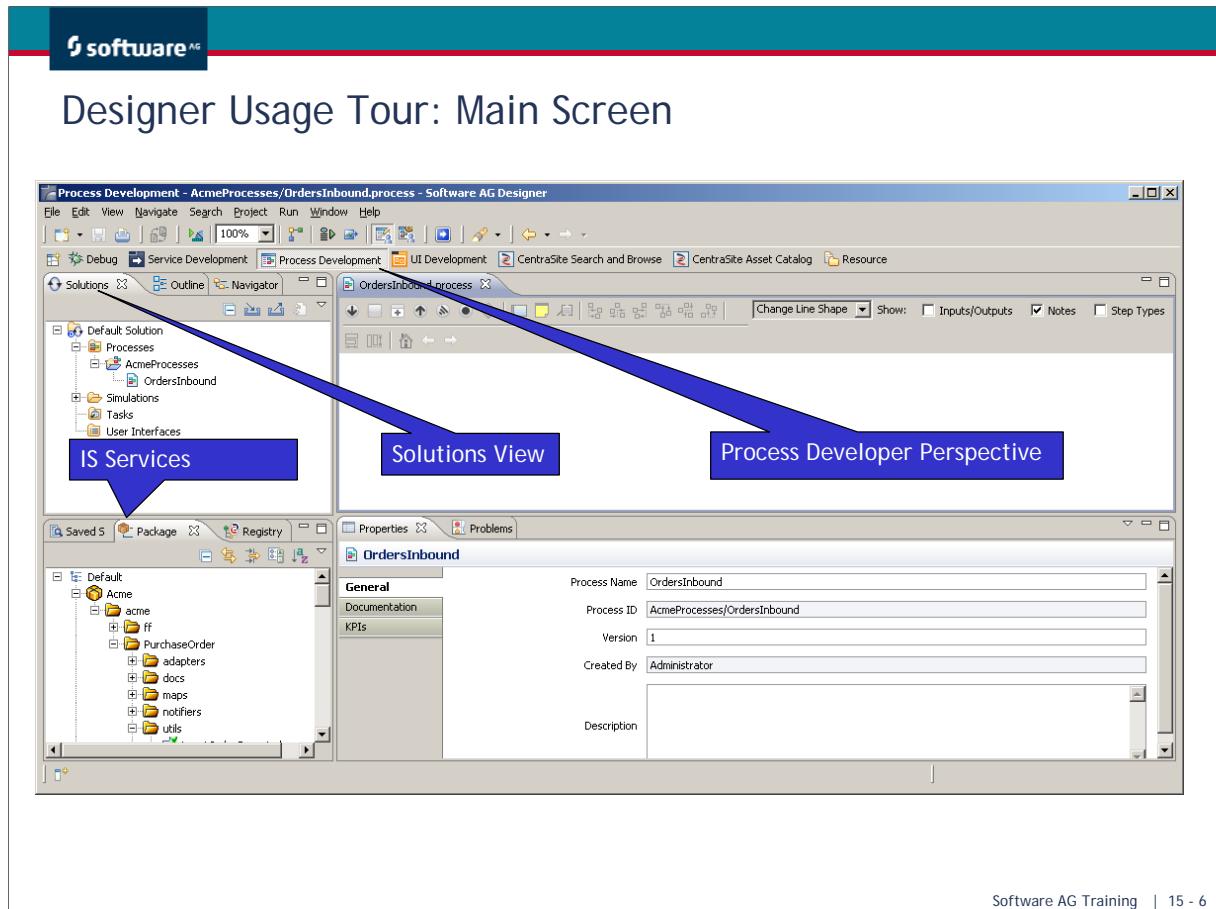
Business Process Management Suite

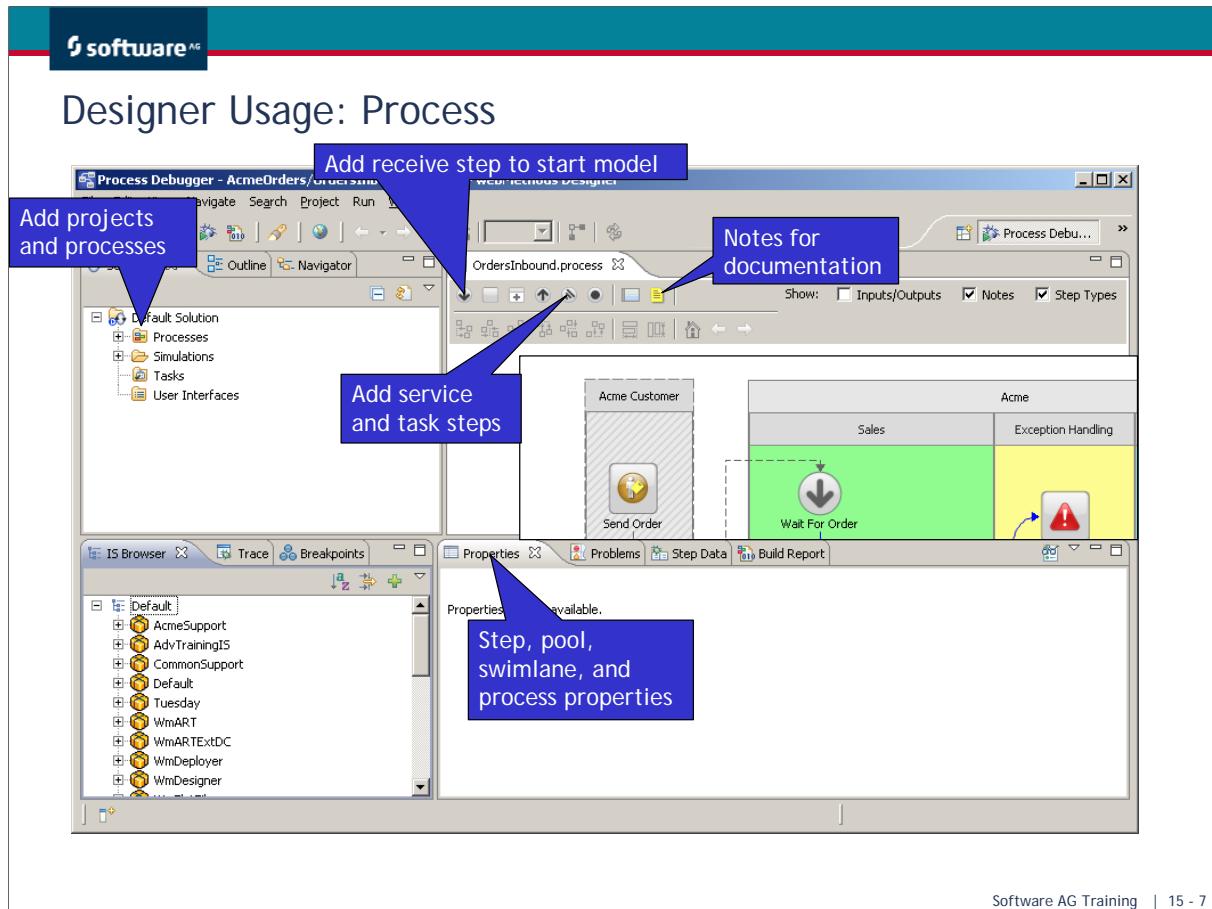
- As part of webMethods 8 Product Suite, webMethods provides Business Process Management via four primary components:
 - Process Engine (execution and coordination)
 - Task Engine (handling of human interaction)
 - My webMethods Server (administration, monitoring and optimization)
 - Broker (messaging in distributed process environments)
- At design-time, processes are created with the Eclipse-based Designer.

Steps to Creating a Basic Business Process

- Add a Designer project.
- Add a business process.
- Add pools (internal and external as needed).
- Add swimlanes to define organizations or actors.
- Add a receive step.
- Add invoke and task steps.
- Connect the steps with transitions.
- Define input and outputs.
- Define process level properties.
- Save and build!

Details on all of these steps are covered in our "BPM for Developers" course.





Software AG Training | 15 - 7

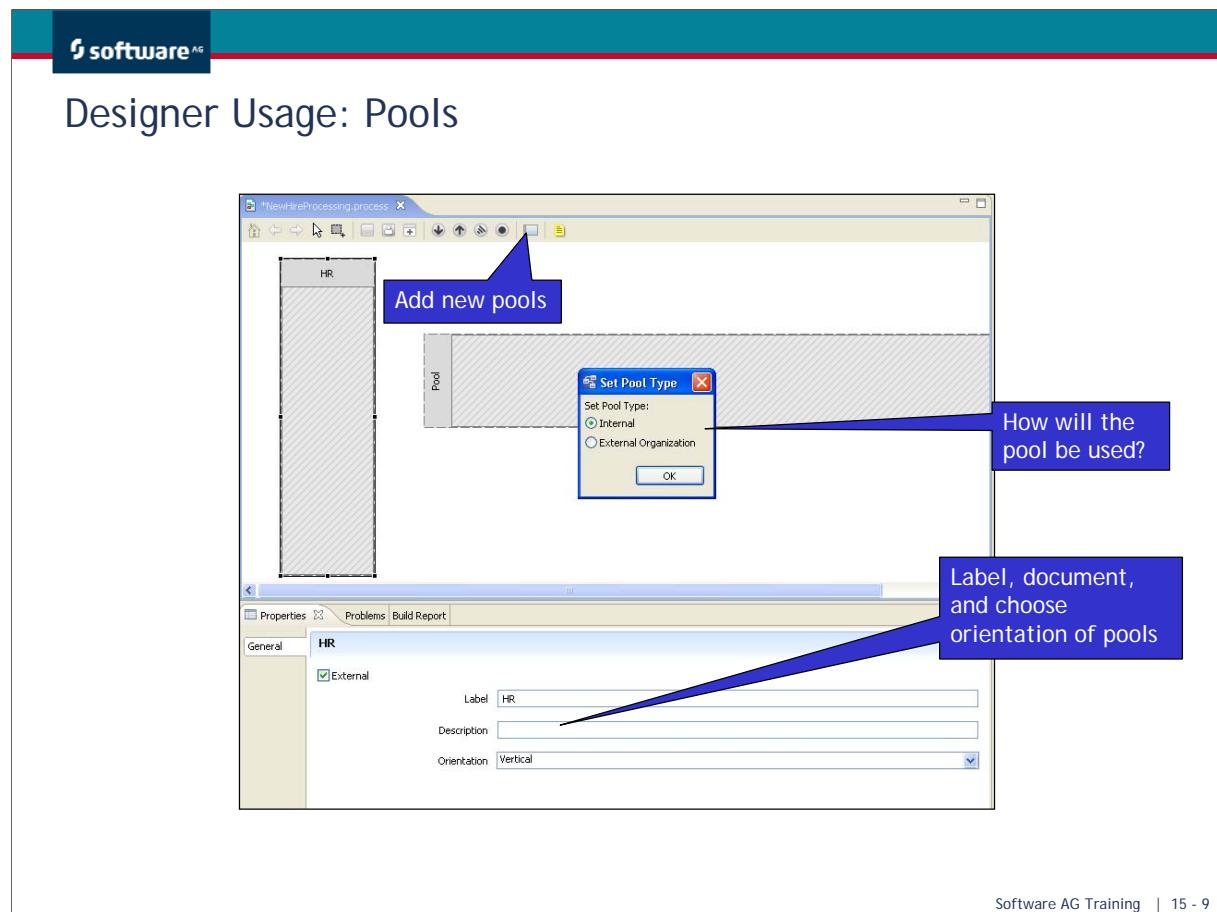
Designer Usage: Process Display

- Once a process is opened in the Designer, use the icons in the main icon bar to change the display.

The screenshot shows the top portion of the webMethods Designer application window. A toolbar contains several icons and labels pointing to specific functions:

- Zoom level**: Points to a zoom control icon.
- Auto layout**: Points to an auto layout icon.
- View inputs/outputs**: Points to an icon for viewing process inputs and outputs.
- View notes**: Points to an icon for viewing process notes.
- View step type**: Points to an icon for viewing process step types.
- Refresh**: Points to a refresh icon.

Below the toolbar is a menu bar with options: File, Edit, View, Navigate, Search, Project, Run, Window, Help. The main workspace area is visible below the toolbar.

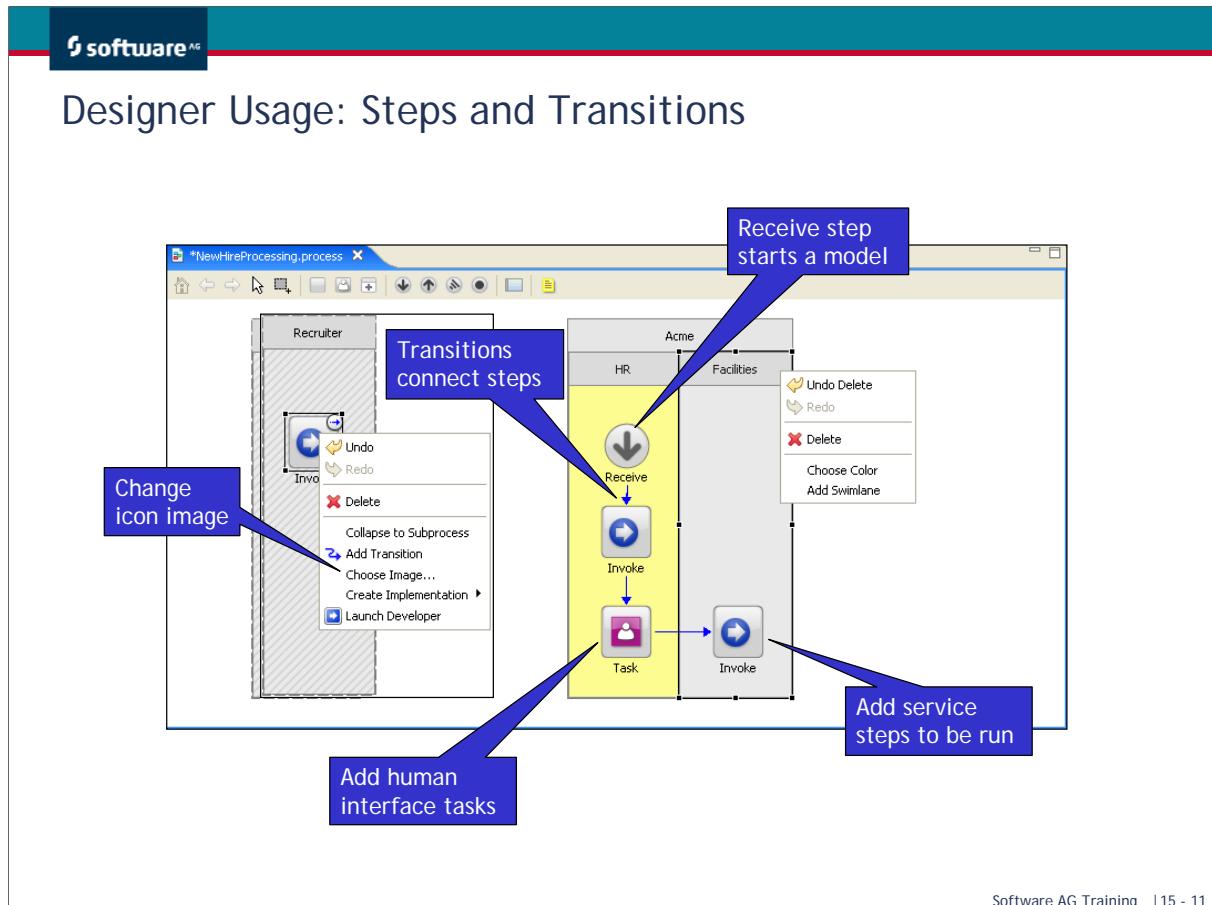


Software AG Training | 15 - 9

Designer Usage: Swimlanes

- Right-click on pools to add swimlanes.

The screenshot shows the webMethods Designer interface with a process titled "NewHireProcessing.process". On the left, there's a subprocess labeled "Recruiter" with a "Task" node. On the right, the main process has two pools: "HR" and "Facilities". The "HR" pool contains a "Receive" node, an "Invoke" node, and a "Task" node. An "Invoke" transition connects the "Task" node to another "Invoke" node in the "Facilities" pool. A context menu is open over the "HR" pool, listing options: Undo, Redo, Delete, Collapse to Subprocess, Add Transition, Choose Image..., Create Implementation, and Launch Developer. Two blue callout boxes point to specific items in this menu: one pointing to "Change pool colors" and another pointing to "Add Swimlane".



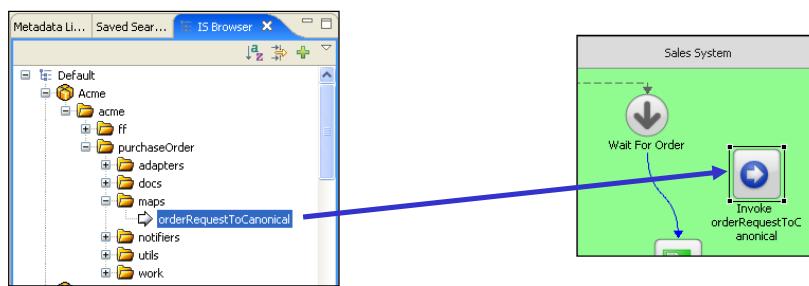
Step Details

- Steps can be added immediately to represent future services.
An empty service will be created when the process is generated.
- Or modify under Implementation to point to an existing service.
A wrapper service containing a call to the service will be created.

The screenshot shows the webMethods Studio interface. In the top window, titled 'Validate', the 'Implementation' tab is selected. The 'Service' field contains the value 'acme.purchaseOrder.utils:inspectLineItems'. In the bottom window, the 'Package Navigator' shows a folder structure under 'Default': 'AcceptOrder', 'Errors', 'ReceiveOrder', 'RejectOrder', 'Validate', 'ValidateOrder', and 'transitionTrigger'. The 'Validate' node is highlighted with a green checkmark icon. A blue arrow points from the 'Service' field in the Properties view to the 'Validate' node in the Package Navigator, indicating the connection between the two.

Drag and Drop to Add Steps

- Steps can be added by clicking on the Activity Step tool bar button.
- Service invoke steps can also be dragged from the IS browser:
 - select the desired server from the browser on the left
 - drag the item onto the appropriate area of the process canvas
 - change properties (step name, signature) as appropriate

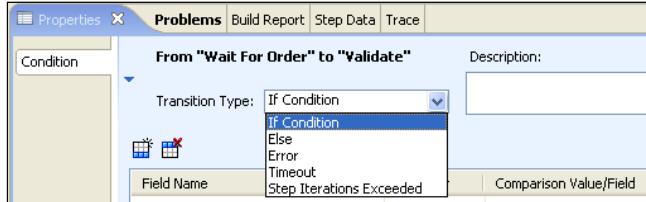


Software AG Training | 15 - 13

software AG

Transition Details

- A process may take different routes based on transition logic.
Logic can be based on process data.
- For example, a single step may have two transitions leaving it. The transitions each have expressions that will be tested prior to execution.
 - If (isValid = true)
 - Else (execution if other expression conditions are not met)
 - Error
 - Timeout
 - Step iterations exceeded



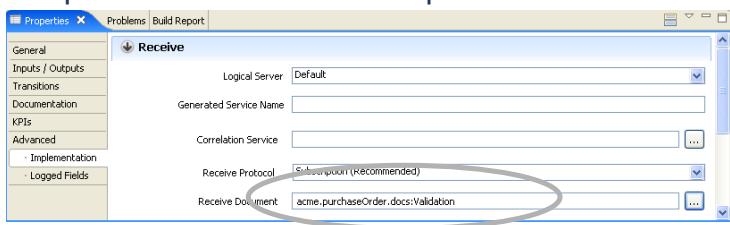
The screenshot shows the 'Properties' dialog box with the 'Problems' tab selected. The main area displays a transition from 'Wait For Order' to 'Validate'. The 'Transition Type' dropdown is open, showing the following options:

- If Condition
- If Condition
- Else
- Error
- Timeout
- Step Iterations Exceeded

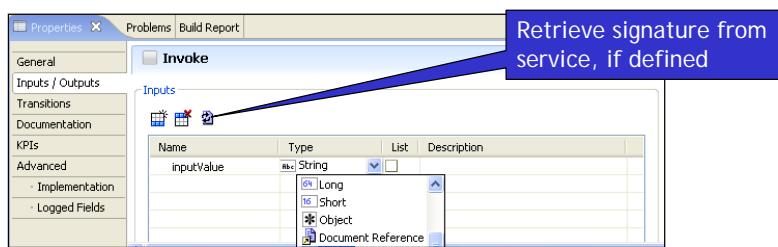
The 'If Condition' option is highlighted. On the right side of the dialog, there is a 'Description:' field and a 'Comparison Value/Field' field.

Designer Usage: Inputs and Outputs

- Receive steps should have a subscription.



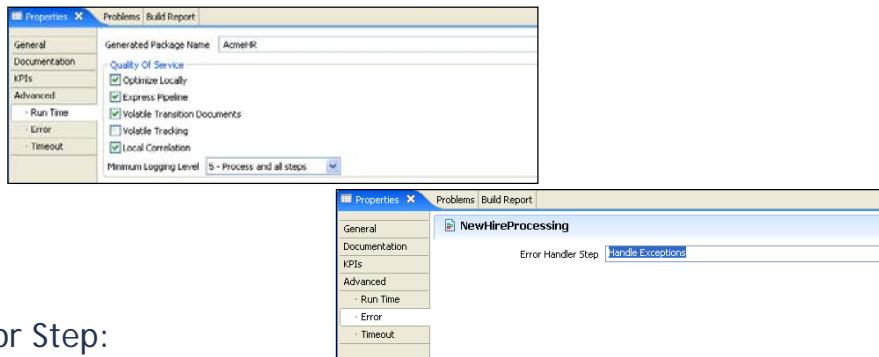
- Invoke steps can have manually added signatures, or they can be retrieved from the service defined in Implementation.



Software AG Training | 15 - 15

Designer Usage: Process Level Properties

- Click on the white space to see Process Level Properties.



- Error Step:
 - add a standalone step to the process (this will later contain error handling logic)
 - choose the step from the Error Handler Step drop down under Error

Saving and Building

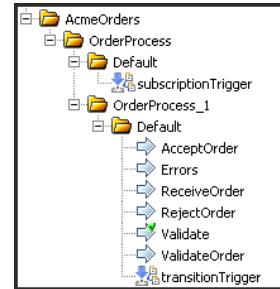
- Once complete, the process must be saved and built.
- Building the process:
 - generates flow services and triggers
 - places process information in the database set by the Designer Process Audit Database preference

The screenshot shows the webMethods Designer interface. At the top, there's a toolbar with icons for File, Edit, View, and Navigate. A blue callout box points to the 'Build Report' icon in the toolbar, with the text 'Click to build process'. Below the toolbar is a menu bar with Properties, Problems, Build Report, Step Data, and Trace. The main area is titled 'Properties | Problems | Build Report | Step Data | Trace'. It displays a log of build steps for a process named 'HR'. The log includes tasks like generating databases, retrieving previous generation information, testing the integration server connection, and generating flow services for various steps. A blue callout box on the right side of the log area contains the text: 'Check the Build Report for a successful build. All errors will be listed in the Problems view'.

Software AG Training | 15 - 17

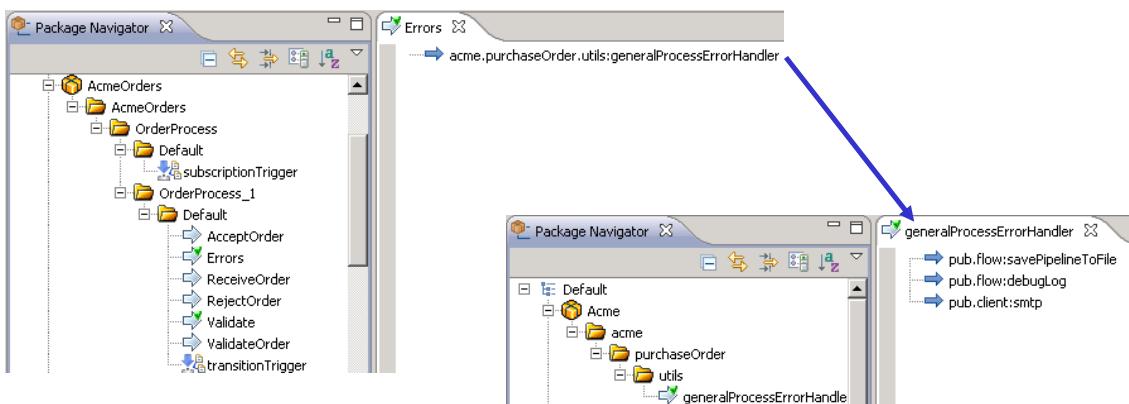
Results of Successful Build

- Building a process generates a set of IS services and triggers. These can be viewed in the Service Development Perspective - remember to refresh!
- Triggers subscribe to documents, such as the starting document in the receive step.
- Services:
 - one for each step in the model
 - empty if no service was selected in Designer
 - wrapper invoking the service if selected



Add Logic to Generated Services

- Steps without selected Implementation services will be generated as empty services.
- Logic can be added directly to these services or a call made to another service containing the logic. This protective layer of separation is a good practice!



Software AG Training | 15 - 19

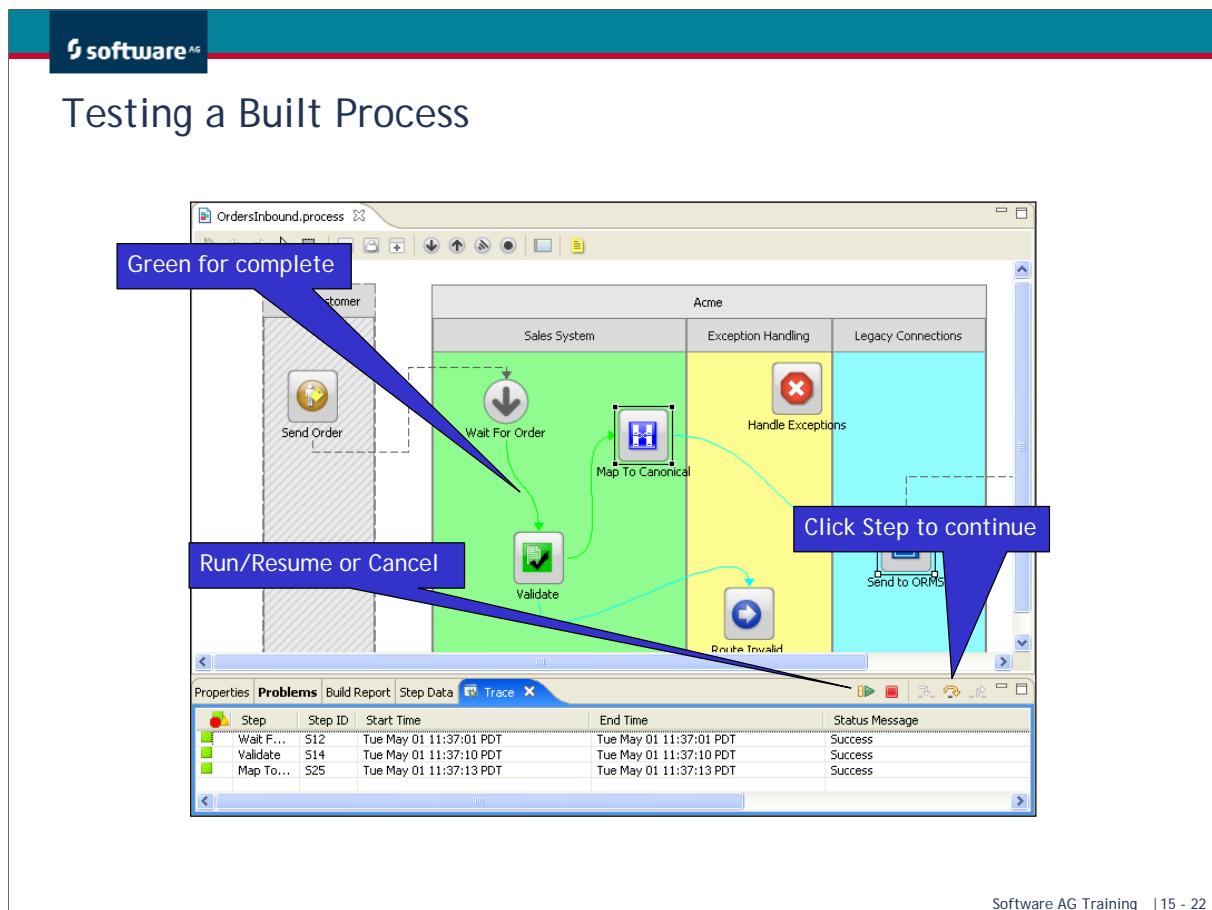
Mapping in Generated Services

- Steps with Implementation services will be generated as flow services that call the chosen service.
- Always verify the mapping the generated services have between the generated service and the implementation service they call.

Ready to be Tested

- A completed process can be tested using the Designer debugging function:
 - processes must be saved and built before debugging
 - run -> debug Selected Process
 - provide data as needed for the starting Receive step
 - step through the process
 - run/resume to complete or cancel debugging

The screenshot shows the webMethods Designer application window. On the left, there's a toolbar with various icons. On the right, the main workspace displays a 'Debug Selected Process' dialog box. This dialog has four input fields: 'cityName' (FreeFormText Fairfax VA), 'addressLine1' (FreeFormText webMethods), 'addressLine2' (FreeFormText 3877 Fairfax Ridge Rd), and 'addressLine3' (FreeFormText Fairfax VA). At the bottom of the dialog are buttons for 'OK', 'Cancel', 'Load', and 'Save'. Above the workspace, the menu bar includes 'File', 'Edit', 'View', 'Navigate', 'Search', 'Project', 'Run', 'Window', and 'Help'. The 'Run' menu is open, showing options like 'Launch Blaze Advisor' and 'Launch webMethods Developer', with 'Debug Selected Process' highlighted.



Software AG Training | 15 - 22

Ready to be Enabled

- Upon building a process, it is uploaded to the process database and ready to be enabled for execution.
- Changes to the logic of the underlying services do not require a rebuild/upload of the process, unless service signatures change and the process will utilize the change!
- If the model changes, the save and build step should be repeated.

 Build of process "OrdersInbound" completed successfully at Mon Apr 30 09:22:04 PDT
 Upload for execution of process "OrdersInbound" completed successfully.
 Restoring Quality of Service settings from previous values for process: "OrdersInbound".

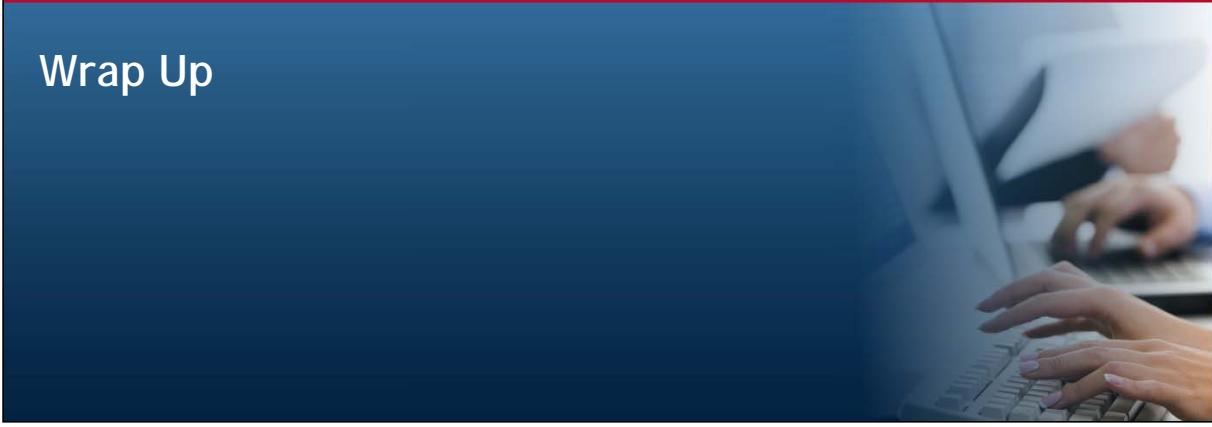
If the process has been debugged (tested),
then it will already be enabled!

This page intentionally left blank.



16

Wrap Up

A blurred background image showing a person's hands with pink nail polish typing on a white computer keyboard. The background is a dark blue gradient.

What Should I Take Next?

- Now that you have completed the "webMethods 8 Integration Workshop", there are a number of other classes you can take:
 - Interested in webMethods administration?
 - webMethods 8 Administration Workshop
 - Interested in Business Process Development?
 - webMethods 8 BPM for Developers
 - webMethods 8 Optimize for Process

<http://www.softwareag.com> ➔ Services ➔ Global Education Services



Certification

- Our certification programs establish standards for knowledge and skills necessary to successfully implement mission-critical IT systems using Software AG technology.
 - <http://www.softwareag.com> ➔ Services ➔ Global Education Services

Further Information

- Find our Developer Communities at

<http://communities.softwareag.com>

- Contact our Support web site at

<http://empower.softwareag.com>

- Discuss product and feature ideas at

<http://brainstorm.softwareag.com>

Thank you

Questions and Comments

Thank you!

Please don't forget to complete a course evaluation.



The End

Questions

Check Your Understanding



Check Your Understanding - webMethods Overview

- What components make up the webMethods 8 Product Suite Enterprise Service Bus?
- What tools are used for Design and Development?
- What three servers should be running in our environment?
- What are the two common switches to use when starting IS?

Check Your Understanding - Developer

- What are packages used for in webMethods?

- What is the name of the package that contains built-in Services?

- What is the Namespace?

- Is this Namespace correct?
MyPackage.myFolder.mySubFolder/myService

Check Your Understanding - Introduction to Services

- What is the Pipeline?

- List 7 Pipeline data types

- When should you create the Pipeline?

Check Your Understanding - Document Types

- What is the difference between a document and document type?
- What are the ways a document type can be created?
- How does one place a constraint on a field?
- When are constraints used?

Check Your Understanding - Flow Services

- Services can be written in what languages?

- List the 7 flow commands.

- What are required parameters of Branch?

- What are regular expressions?

Check Your Understanding - Flow Services

- What are the key parameters for the Loop operation?
- What operation do I use to exit when a series of services when one of those services is successful?
- How can I use Flow to throw a java exception?
- How many time will a service in a Repeat count = 1 execute?

Check Your Understanding - Mapping

- What are three different colors of map link lines? What do they each mean?
- How can you “map” without drawing lines?
- What are transformers?
- Is implicit mapping a feature of transformers?

Check Your Understanding - Java Services

- When should Java Services be used instead of Flow?

- What is a cursor? How is it used?

- Which Java API class would be used to generate an IData object?

Check Your Understanding - Monitoring Services

- Which file contains the setting: auditLog= ?

- Why do you need a remote server alias?

- How do you connect the IS to the audit database?

Check Your Understanding - Invoking Services

- What are the different protocols the Integration Server can use?
- When would `queryXMLNode` be used?
- Which built-in Service would you use to convert a document type to a XML string?
- Where can you find the services to send data from the IS?

Check Your Understanding - Flat File Handling

- What is the difference between a flat file schema and a flat file dictionary?

- What are the four types of Validators?

- What are the two services used for converting to and from flat files?

Check Your Understanding - Web Services

- What are the three principle standards for web services?

- When do you need a Web Service Consumer Node?

- When do you create a WSDL file?

Check Your Understanding - Broker Messaging

- Can you list the six different Publish/Subscribe Services?
- What three things are required to subscribe?
- What two things are required to publish?
- What's the difference between XOR, OR, AND and joins?

Check Your Understanding - JMS Messaging

- What are the two primary JMS publish services?

- What are the standard JMS Message types?

- What are the two ways to subscribe to a JMS message?

This page intentionally left blank.

Last page