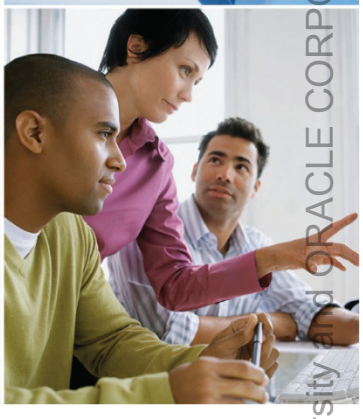
THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

ORACLE UNIVERSITY



Hardware and Software
Engineered to Work Together

Oracle University and ORACLE CORPORATION use only

Oracle Service Bus 12*c*: Design & Integrate Services

Student Guide
D88521GC10
Edition 1.0 | July 2015 | D92011

Learn more from Oracle University at **oracle.com/education/**

ORACLE

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Author

Iris Li

Technical Contributors and Reviewers

Dimitri Laloue
Greg Fichtenholtz
Chad Schoettger
James Mills
Takyiu Liu
Matthew Slingsby
Armando Hernandez
Robert Wunderlich
Mike Muller

Editors

Malavika Jinka
Aishwarya Menon
Nikita Abraham

Graphic Designer

Kavya Bellur

Publishers

Michael Sebastian
Jayanthy Keshavamurthy
Jobi Varghese

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Disclaimer

This document contains proprietary information and is protected by copyright and other intellectual property laws. You may copy and print this document solely for your own use in an Oracle training course. The document may not be modified or altered in any way. Except where your use constitutes "fair use" under copyright law, you may not use, share, download, upload, copy, print, display, perform, reproduce, publish, license, post, transmit, or distribute this document in whole or in part without the express authorization of Oracle.

The information contained in this document is subject to change without notice. If you find any problems in the document, please report them in writing to: Oracle University, 500 Oracle Parkway, Redwood Shores, California 94065 USA. This document is not warranted to be error-free.

Restricted Rights Notice

If this documentation is delivered to the United States Government or anyone using the documentation on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS
The U.S. Government's rights to use, modify, reproduce, release, perform, display, or disclose these training materials are restricted by the terms of the applicable Oracle license agreement and/or the applicable U.S. Government contract.

Trademark Notice

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

Contents

1 Introduction
Goals 1-2
Audience 1-3
Prerequisites 1-4
Class Introductions 1-5
Course Objectives 1-6
Course Map 1-7
Course Schedule 1-8
How Can I Learn More? 1-9

2 Introducing Oracle Service Bus
Objectives 2-2
Agenda 2-3
Service Integration Can End Up as Spaghetti 2-4
Enterprise Service Bus Reduces Complexity and Enables Flexibility 2-5
What Does an Enterprise Service Bus Do? 2-6
What an Enterprise Service Bus Should Not Do 2-7
Key Capabilities 2-8
Benefits 2-10
Agenda 2-11
About Oracle Service Bus 2-12
Example: Virtualizing Back-end Service 2-13
Functional Features of Oracle Service Bus 2-14
Features of Oracle Service Bus 12c 2-15
Agenda 2-16
SOA Reference Architecture 2-17
Oracle Service Bus in Oracle SOA Suite 2-18
Oracle SOA Suite and Oracle Service Bus Product Architecture 2-19
Key Components in a SOA Composite Application 2-20
Mediator Versus OSB 2-21
Oracle SOA Suite and Oracle Service Bus Deployment Topology 2-22

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda 2-23
Developer and Production Distributions 2-24
Domain Configuration 2-25
Summary 2-26
Practice 2: Overview 2-27

3 Getting Started with Oracle Service Bus Applications
Objectives 3-2
Agenda 3-3
Web Services Description Language (WSDL) 3-4
Abstract WSDL 3-6
XML Schema Definition (XSD) 3-7
XSD: Example 3-8
Agenda 3-9
OSB Components 3-10
Business Services 3-11
Service Types for Business Services 3-12
Business Service Configuration 3-14
Proxy Services 3-15
Proxy Service Configuration 3-16
Pipelines 3-17
Quiz 3-18
Agenda 3-19
Developing Service Bus in JDeveloper 3-20
Projects and Resources 3-21
A Basic Service Bus Application 3-22
High-Level Steps of Creating a Service Bus Application 3-23
Creating a Business Service 3-24
Creating a Proxy Service 3-26
Agenda 3-27
Features for Developers 3-28
Import and Export Resources 3-29
Share Artifacts with MDS 3-30
Populating Shared Data with File-Based MDS 3-31
Using Shared Data in MDS 3-32
Promote Reusability and Best Practices with Pipeline Template 3-33
Test and Debug Integrations 3-34
OSB Maven Support 3-35
Oracle Service Bus Console 3-36
Working with Sessions 3-37
Test Console 3-38

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Monitoring and Administering in EM Console 3-39
Development Tasks Support 3-40
Quiz 3-41
Summary 3-42
Practice 3: Overview 3-43

4 Basics of Message Flow
Objectives 4-2
Agenda 4-3
Message Exchange Pattern 4-4
Message Flow 4-5
Pipeline 4-6
Pipeline Components 4-7
Message Processing Actions 4-8
Communication Actions 4-9
Flow Control and Report 4-10
Message Flow Communication Actions 4-11
Pipeline: Example 4-13
Quiz 4-14
Agenda 4-15
Message Context 4-16
Predefined Context Variables 4-17
Message-related Variables 4-18
$inbound and $outbound Variables 4-19
$operation Variable 4-20
$fault Variable 4-21
Variables and Types 4-22
Context Variables in Message Processing 4-23
Pass Content by Reference 4-24
User-Defined Variables 4-25
Shared Variables 4-26
Shared Variable Rules 4-27
Creating a Shared Variable 4-28
Quiz 4-29
Agenda 4-30
Pipeline Construction 4-31
Creating a Pipeline 4-33
Pipeline Templates 4-34
Creating and Using Pipeline Template 4-35
Breaking a Template Link 4-36
Agenda 4-37

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Debugging Service Bus Applications 4-38
Enable/Disable Stages and Actions 4-39
Enable Flow Tracing in Pipeline 4-40
Debugging Service Bus Applications with Debugger 4-42
Features and Limitations 4-43
Configuring Debugger 4-44
Setting Breakpoints 4-45
Starting the Debugger 4-46
Stepping Through a Debugging Session 4-47
Viewing and Modifying Variable Values 4-48
Ending a Debugging Session 4-49
Summary 4-50
Practice 4: Overview 4-51

5 Validating Messages and Error Handling
Objectives 5-2
Agenda 5-3
Message Validation 5-4
Validate Action 5-5
Dynamic Validation 5-6
Quiz 5-7
Agenda 5-8
Faults: Overview 5-9
Fault-Handling Challenges 5-10
Handling Errors in OSB 5-11
Error Handlers 5-12
$fault Variable 5-13
Reply Action 5-14
Generating Error Messages Using Reply Action 5-15
Quiz 5-16
Agenda 5-17
Reporting Actions 5-18
Report Action 5-19
Log Action 5-20
Configuring OSB Log Level 5-21
Alert Action 5-22
Summary 5-23
Practice 5: Overview 5-24

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

6 Transforming Messages

Objectives 6-2
Agenda 6-3
Message Transformation in Service Bus 6-4
Terminology and Concepts 6-5
XPath 6-6
XPath: Examples 6-7
XPath Functions 6-8
XPath Expression Builder 6-9
Standard XPath Functions 6-10
Custom XPath Functions 6-11
Creating and Packaging Custom XPath Functions 6-12
Registering a Custom Function 6-13
Agenda 6-14
Data Standards 6-15
Transforming Data 6-16
XSL Transformations 6-17
XSLT Mapper 6-18
Using XPath Functions in XSLT Mapper 6-19
Agenda 6-20
XQuery: Introduction 6-21
Uses of XQuery 6-22
XQuery Support in Service Bus 6-23
XQuery Expression Builder 6-24
XQuery Mapper 6-25
Types of XQuery Maps 6-26
Steps for Creating XQuery Map Files 6-27
XSLT Versus XQuery 6-28
Test Expressions at Design-Time 6-29
Agenda 6-30
XML to non-XML with nXSD 6-31
Working with Native Format Data 6-32
Native Data Transformation 6-33
Specifying File Name and Native Data Format 6-35
Specifying a Sample File 6-36
Defining a Schema for a Native Format 6-37
nXSD Translate Action 6-38
Summary 6-39
Practice 6: Overview 6-40

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

7 Routing Messages
Objectives 7-2
Agenda 7-3
Message Routing: Overview 7-4
Routing Use Cases 7-5
Header-based Routing 7-6
Agenda 7-7
Content-based Routing 7-8
Conditional Branch Node 7-9
Configuring Conditional Branch Node 7-10
Routing Table Action 7-11
Conditional Branching Versus Routing Table 7-12
Quiz 7-13
Agenda 7-14
Dynamic Routing 7-15
Dynamic Routing with Dynamic Routing Action by Example 7-16
Mapping Table Used by Dynamic Routing Action: Example 7-17
XQuery Expression for Dynamic Routing Action 7-18
Configuring Dynamic Routing Action 7-19
Dynamic Routing with Routing Options Action by Example 7-20
Mapping Table Used by Routing Options Action: Example 7-21
Configuring Routing Options Action 7-22
Quiz 7-23
Summary 7-24
Practice 7: Overview 7-25

8 Enriching Messages
Objectives 8-2
Message Enrichment Options 8-3
Service Callout 8-4
Using Service Callout 8-5
Service Callout Action 8-6
Configuring Service Callout Action 8-7
Constructing Service Callout Messages 8-8
Java Callout 8-10
Use Cases 8-11
Java Callout Guidelines 8-12
Java Callout Action 8-13
Summary 8-14
Practice 8: Overview 8-15

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

9 Processing Messages with Concurrent Calls
Objectives 9-2
Split-Join 9-3
Split-Join Patterns 9-4
Split-Join Invocation 9-5
Split-Join Construction: Static Split-Join 9-6
Split-Join Construction: Dynamic Split-Join 9-7
Split-Join Operations 9-8
Global and Local Variables 9-9
Transaction Support 9-10
Summary 9-11
Practice 9: Overview 9-12

10 Adapters and Transports
Objectives 10-2
Agenda 10-3
JCA Transport 10-4
JCA Adapter Framework 10-5
JCA Adapters 10-6
Supported Adapters and Transports 10-7
File Adapter 10-8
Database Adapter 10-9
JMS Message 10-10
JMS Messaging 10-11
JMS Resources 10-12
Oracle JMS Transport 10-13
Agenda 10-15
SOA-DIRECT Transport 10-16
Features of SOA-DIRECT Transport 10-17
Transactional Boundaries 10-18
Synchronous Invocation of a SOA Composite 10-19
Endpoint URI Linking to a SOA Composite 10-20
Synchronous Invocation from a SOA Composite 10-21
Asynchronous Invocation of a SOA Composite 10-22
Service Roles in Asynchronous Invocation 10-23
Agenda 10-24
REST: Overview 10-25
REST Queries 10-26
Comparing REST and SOAP 10-27
Example: Use Cases for REST and SOAP 10-28
Supported Features in OSB 10-29

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Creating a REST Binding 10-30
Configuring a REST Binding 10-31
Summary 10-33
Practice 10: Overview 10-34

11 Reliable Messaging
Objectives 11-2
Reliable Communication 11-3
Quality of Service 11-4
Default Value of qualityOfService 11-6
Overriding the Default Value of QoS in Routing Options Action 11-8
Configurations for Reliable Communication 11-9
Configuring Retry Handling in JMS Queue 11-10
High-Level Steps 11-11
Configuring Delivery Failure Options 11-12
JMS Transport Configuration of the Proxy Service 11-13
Summary 11-14
Demonstration: Overview 11-15

12 Service Bus Security
Objectives 12-2
Agenda 12-3
Web Services Security: Overview 12-4
Transport-Level Security 12-5
Message Security 12-6
WS-Security 12-7
SAML 12-8
SAML Architecture 12-9
WS-Security and SAML 12-10
WS-Policy: Overview 12-11
Policy Assertion 12-12
Quiz 12-13
Agenda 12-14
SB Security Capabilities 12-15
Active and Passive Intermediary 12-17
Securing Services with OWSM Policies 12-19
Oracle Web Service Manager 12-20
Components of Oracle Web Services Manager Architecture 12-21
Inbound Security by Using OWSM 12-22
Outbound Security by Using OWSM 12-23
Policy Support (12c 12.1.3.0) 12-24

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

User Interface (UI) Support 12-25
Identity Propagation 12-26
Identity Propagation: Example 12-27
OWSM Predefined Policies 12-28
Applying OWSM Policies to Services in JDeveloper 12-29
Test Console Support: Business Service 12-30
Test Console Support: Proxy Service 12-31
Agenda 12-32
Service Access Control Policies 12-33
Role-Based Access: Application Security Roles 12-34
Summary 12-35
Practice 12: Propagating Identity from Service Bus to a Web Service 12-36

13 Advanced Topics
Objectives 13-2
Agenda 13-3
Principles and Best Practices of Using Service Bus 13-4
BPEL Versus Service Bus 13-5
Testing Approaches 13-6
Agenda 13-7
Service Result Caching 13-8
Use Cases of Result Caching 13-9
How Does Result Catching Work? 13-10
Configuring a Business Service for Result Caching 13-11
Using an Out-of-Process Coherence Cache Server 13-12
Coherence Configuration Files 13-13
Agenda 13-14
SLA Alert in OSB 13-15
SLA Alert Rules 13-16
Creating SLA Alert Rules 13-17
Agenda 13-18
Managed File Transfer (MFT) 13-19
OSB and MFT Integration 13-20
Agenda 13-21
Resequencing Message 13-22
Configuring the Resequencer 13-23
Work Manager in Service Bus 13-25

Oracle University and ORACLE CORPORATION use only

Agenda 13-26
About Maven 13-27
Concepts and Terminology 13-28
OSB Maven Support 13-30
Summary 13-31
Practice 13: Overview 13-32
Building Service Bus Projects with Maven 13-33

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

1

Introduction

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Goals

The main goals of this course are to:

- Get familiar with common Oracle Service Bus features and functionality
- Use Oracle Service Bus to develop service- and message-oriented applications

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

This course places strong emphasis on hands-on practices. It is written for a developer who has technical background, but does not have any experience with Oracle Service Bus.

The course provides an overview of Oracle Service Bus features and functionality.

The goal of the course is to provide you with an understanding of Oracle Service Bus, and help you to arrive at a level of comfort by using it to create service- and message-orientated (integration) solutions.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Audience

The target audience includes:

- Integration developers
- Service designers and developers
- Enterprise architects

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Prerequisites

To successfully complete this course, you should meet the following requirements:

- Oracle University course: *XML Fundamentals* or equivalent knowledge on XML. This is to help understand XML documents, XPath, XML schema, XSLT, and XQuery principles.
- Working knowledge of messaging, web service, and SOA concepts

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Class Introductions

Briefly introduce yourself:

- Name
- Title/position
- Company
- Knowledge of:
  - XML and web services
  - Messaging
- Reasons for attending

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Course Objectives

After completing this course, you should be able to:

- Describe Oracle Service Bus architecture and functional layers
- List Oracle Service Bus capabilities
- Enable service-level abstraction
- Perform message validation, intelligent routing, transformation, and enrichment
- Integrate services with legacy systems
- Provide reliable delivery of messages between applications and services
- Use Oracle Service Bus and OWSM to secure services
- Test, debug, and troubleshoot Oracle Service Bus applications

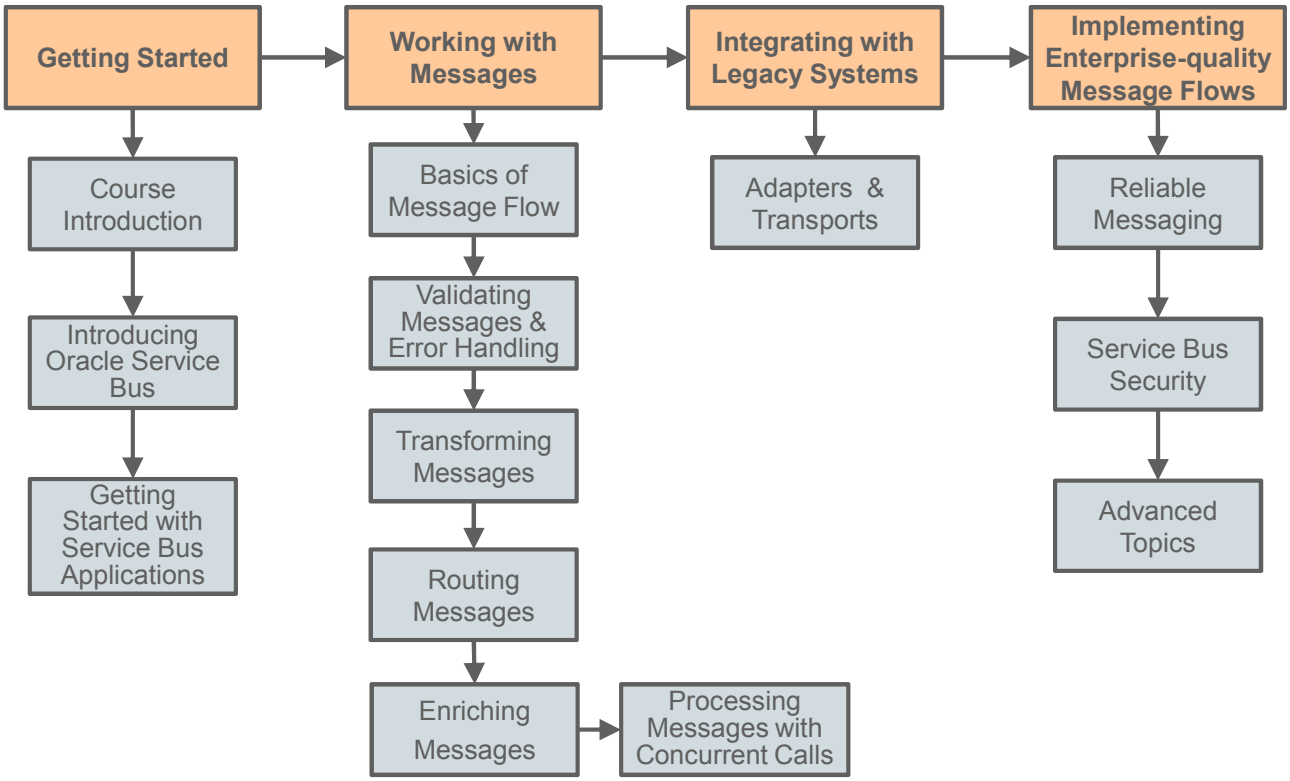
ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Course Map



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Course Schedule

Day	Module
Day 1	1. Course Introduction 2. Introducing Oracle Service Bus 3. Getting Started with Oracle Service Bus Applications
Day 2	4. Basics of Message Flow 5. Validating Messages and Error Handling 6. Transforming Messages
Day 3	7. Routing Messages 8. Enriching Messages 9. Processing Messages with Concurrent Calls 10. Adapters and Transports
Day 4	11. Reliable Messaging 12. Service Bus Security 13. Advanced Topics

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

How Can I Learn More?

Topic	Website
Education and Training	http://education.oracle.com
Product Documentation	http://www.oracle.com/technology/documentation
Product Downloads	http://www.oracle.com/technology/software
Product Articles	http://www.oracle.com/technology/pub/articles
Product Support	http://www.oracle.com/support
Product Forums	http://forums.oracle.com
Product Tutorials	http://www.otn.oracle.com/obe
Sample Code	http://www.samplecode.oracle.com/

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

2

Introducing Oracle Service Bus

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Objectives

After completing this lesson, you should be able to:

- Describe the challenges of enterprise integration
- Describe what an Enterprise Service Bus does
- List the functionality of Oracle Service Bus
- Describe Oracle Service Bus architecture and components
- Describe the relationship between Oracle Service Bus and Oracle SOA Suite
- List Oracle Service Bus installation and configuration options

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Enterprise Service Bus
- Introducing Oracle Service Bus
- Oracle Service Bus and Oracle SOA Suite
- Oracle Service Bus installation

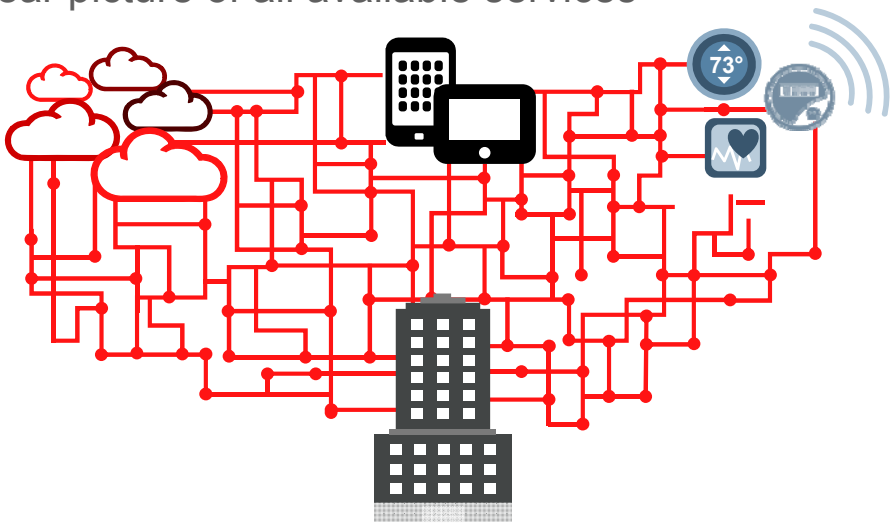
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Service Integration Can End Up as Spaghetti

- Too many point-to-point links
- Multiple transports and protocols
- No clear picture of all available services



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Integration can be done in many ways, each with its respective strengths and weaknesses. One approach is to use point-to-point integration, where two systems are integrated at a time. For any pair of systems, this technique is quite effective. However, as the number of integrations grows, this approach tends to be brittle and difficult to maintain over time. Functionality is often duplicated across applications. Changes to a single application interface have the potential to impact large numbers of systems, each of which may need modification—a time-consuming and expensive proposition.

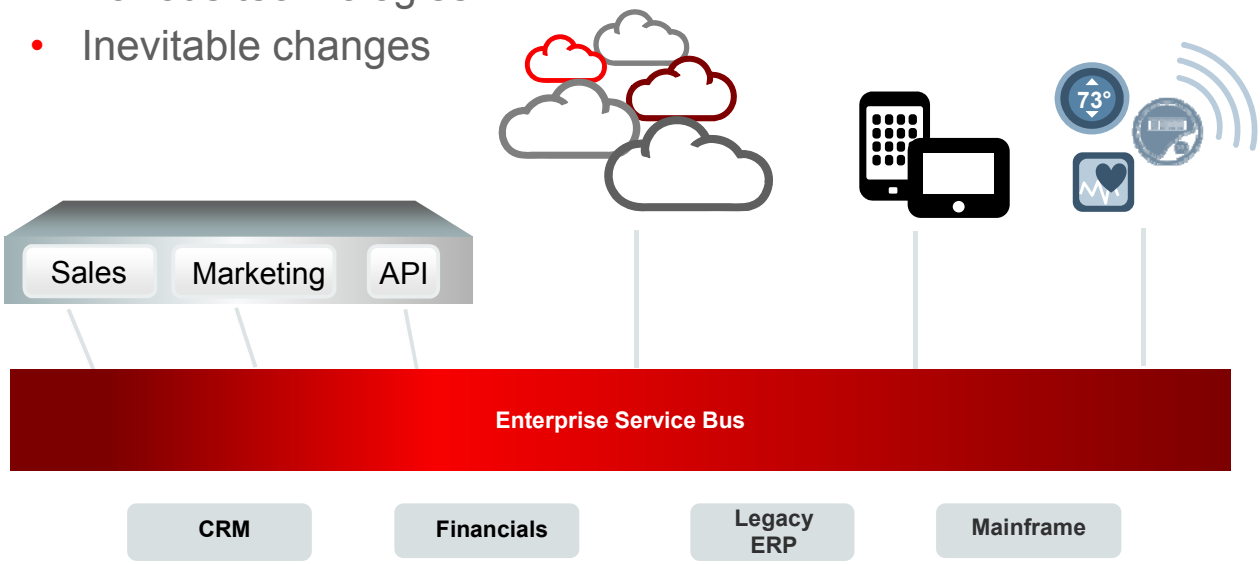
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Enterprise Service Bus Reduces Complexity and Enables Flexibility

Address integration challenges:

- Distributed system
- Various technologies
- Inevitable changes



ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

An Enterprise Service Bus (ESB) is one more layer of abstraction in your integration architecture that handles connecting, routing, transformation, protocol-bridging and security. It represents an environment designed to foster interconnectivity and data sharing between services. It establishes an intermediate layer of processing that addresses a few fundamental challenges of service integration:

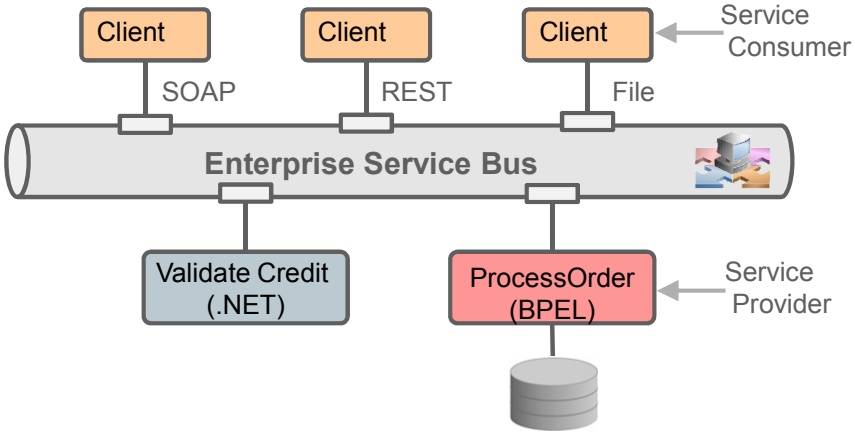
- Distributed system
- Various technologies: An integration solution must be able to interface with all these different technologies.
- Inevitable changes: An integration solution needs to minimize the dependencies from one system to another by using loose coupling between applications.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

What Does an Enterprise Service Bus Do?

- Mediate
- Abstract and decoupling
- Virtualize
- Provide a communication backbone to SOA



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

- The core functionality of an ESB is to mediate between service providers and service consumers.
- ESB operates as a middle abstraction layer, and as such reduces dependences by decoupling the applications.
- ESB can provide a virtual layer between the client application and the integration implementation.
- ESB is also being named as the communication backbone of a SOA landscape.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

What an Enterprise Service Bus Should Not Do

- Orchestration
- Workflow
- Business logic
- Complex transformations

✓ Nonfunctional requirements

⊘ Logic and intelligence

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

ESB is a solution for nonfunctional requirements. Developers are not encouraged to put more logic and intelligence into the infrastructure.

ESB should be a highly distributable infrastructure capable of connecting, mediating, and coordinating the interaction between applications and services across a variety of platforms, protocols, and interface technologies. ESB facilitates publishing of services at a higher level of abstraction than message types but does not contain service implementation. So an ESB does not include orchestration, workflow, rule engines, and portals.

ESB is much more suitable for atomic/entity services, and SOA/BPEL should be used for orchestration purposes.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Key Capabilities

The capabilities that an Enterprise Service Bus provide include:

- Protocol mediation
- Invocation
- Data mapping and transformation
- Routing
- Message enhancement
- Reliability
- Security
- Service management

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

An ESB adapts between the different realms that interacting partners may be living in, for example, physical location, business terminology and data structure, security, and communication protocol. ESB exposes services in ways that make them accessible to diverse potential consumers. ESB provides the following capabilities:

- **Protocol mediation:** Protocol mediation is a key capability provided by an ESB. Protocol mediation allows the service consumer to make a request by using one protocol, such as a web service, and the service provider to use a different protocol, such as RMI. ESB converts from one to the other and back again.
- **Invocation:** Support for synchronous and asynchronous transport protocols
- **Data mapping/transformation:** The ability to convert the structure and format of the incoming request message to the structure and format of the out going message that will be consumed by the service provider
- **Routing:** An important ESB task is intelligent routing. It is the ability to route a request to a specific service provider based on a static or variable routing criteria.
- **Message enhancement:** The ability to modify or add to the data contained in a message

Oracle University and ORACLE CORPORATION use only

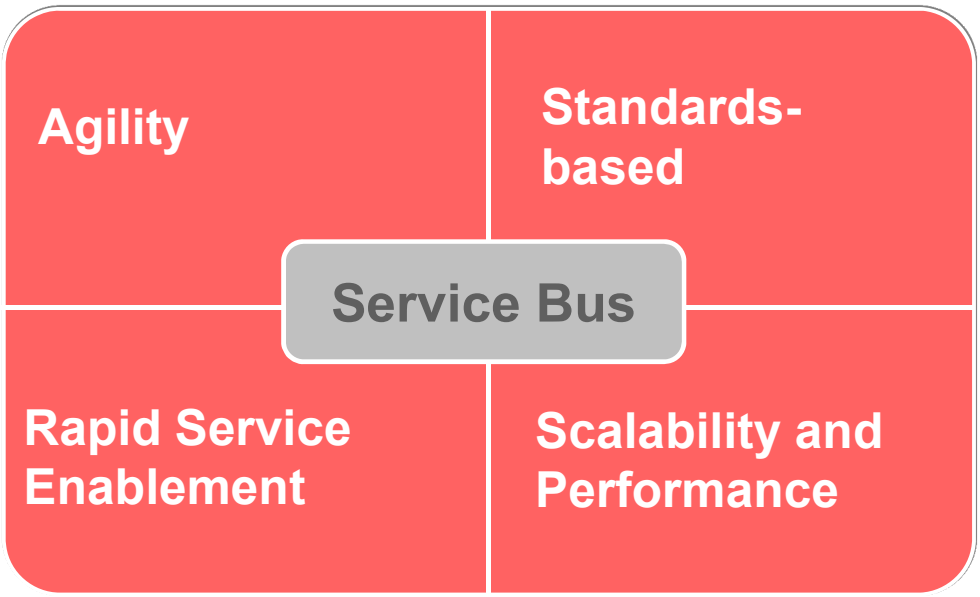
- **Reliability:** Different protocols provide different forms of reliability. For example, the HTTP protocol is not reliable, because it does not inherently guarantee delivery. Some other protocols may guarantee that a message will be delivered. For this reason, ESB defines how it will deal with reliability issues.
- **Security:** The ability to protect enterprise services from unauthorized access. ESB provides a way to handle the different aspects of security. The security in ESB systems builds on prevailing Web Services Security (WS-Security) standards.
- **Service management:** As a SOA grows, you may run into problems such as finding existing services and reusing them in new business processes, or there may be a technical requirement to be able to deploy and run a service infrastructure. ESB provides service management features to address these needs.

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Benefits



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The key advantages of ESB are:

Agility: ESB reduces the number of point-to-point connections required for applications communication. Increased flexibility; easier to adapt to change as requirements change.

Standards-based: Uses XML as the standard communication language and supports web services standards; allows developers to use generic forms of query, access and interaction.

Rapid Service Enablement: Enables you to find applications and interfaces for reuse. By leveraging proven and time-tested functionality, you lower the risk and reduce the time to market of your product.

Scalability and Performance: Being part of Oracle Fusion Middleware. Service Bus leverages Oracle Coherence, a unique technology in the domain of distributed caching. Service Bus uses it to cache the results of service invocations. For instance, when you call an external service you will first request that value but also store it in the cache for subsequent requests. What this gives you is reduced risk first because even if the remote service was to be unavailable, you could always access the cached value. It also greatly improves performance and resource utilization by eliminating the need to constantly request data faster than it is changing.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Enterprise Service Bus
- Introducing Oracle Service Bus
- Oracle Service Bus and Oracle SOA Suite
- Oracle Service Bus installation

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

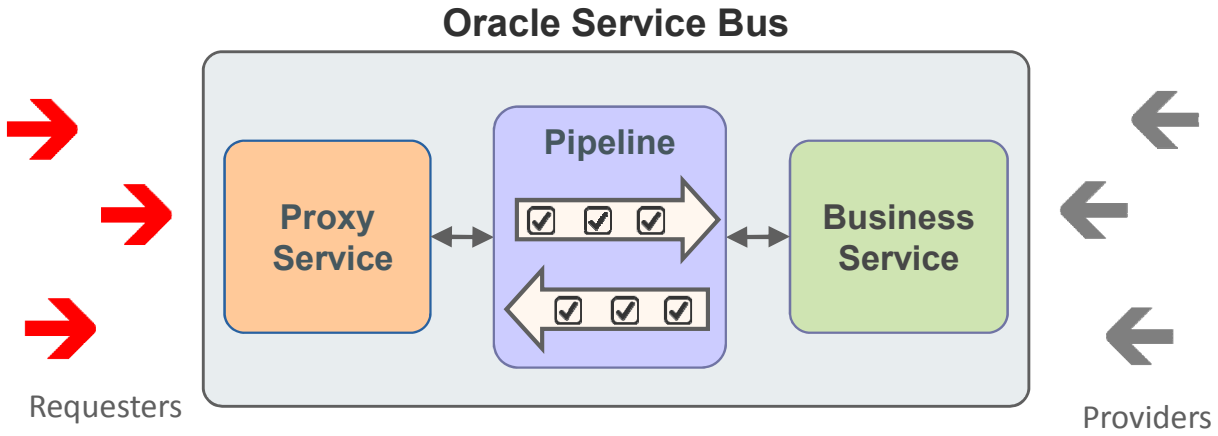
- Requirements
- What it does
- Architecture
- Capabilities
- Benefits
- Fit in SOA
- When to use

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

About Oracle Service Bus

- Supports different message formats
- Provides a layer of abstraction
  - Protect consumers from deployment location changes and implementation updates
  - Scale the service: Load-balance requests to back-end systems



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle Service Bus routes messages between service providers (such as enterprise services and databases) and service requesters/clients (such as presentation applications or other business services). Service Bus is composed of service and flow components:

- **Business Service:** Represents services that are external to Service Bus; take care of communication with service providers
- **Pipeline:** Defines the flow of request and response messages through Service Bus, including routing, transformations, validations, reporting, and exception management
- **Proxy Service:** The interfaces that service consumers use to connect with managed back-end services

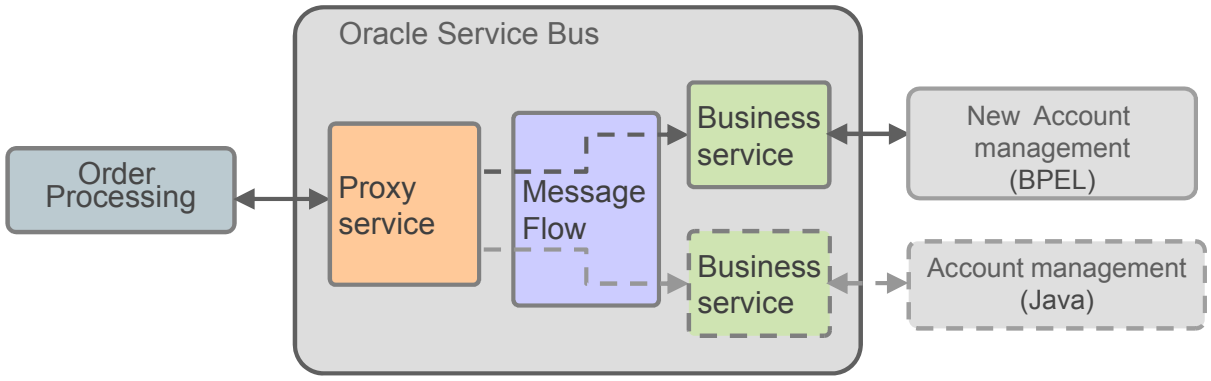
The Oracle Service Bus, illustrated in the diagram, provides the following benefits:

- **Abstraction layer:** Service Bus provides a layer of abstraction de-coupling service providers from service consumers. The changes can be made to providers without impact on clients or vice versa.
- **Location transparency:** The ability to proxy multiple endpoints for the same service. If you have the same service running on several different machines, OSB can load-balance the calls among those servers.
- **Support of different message formats:** Requesters and providers do not have to agree on the message format and transport.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Example: Virtualizing Back-end Service



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

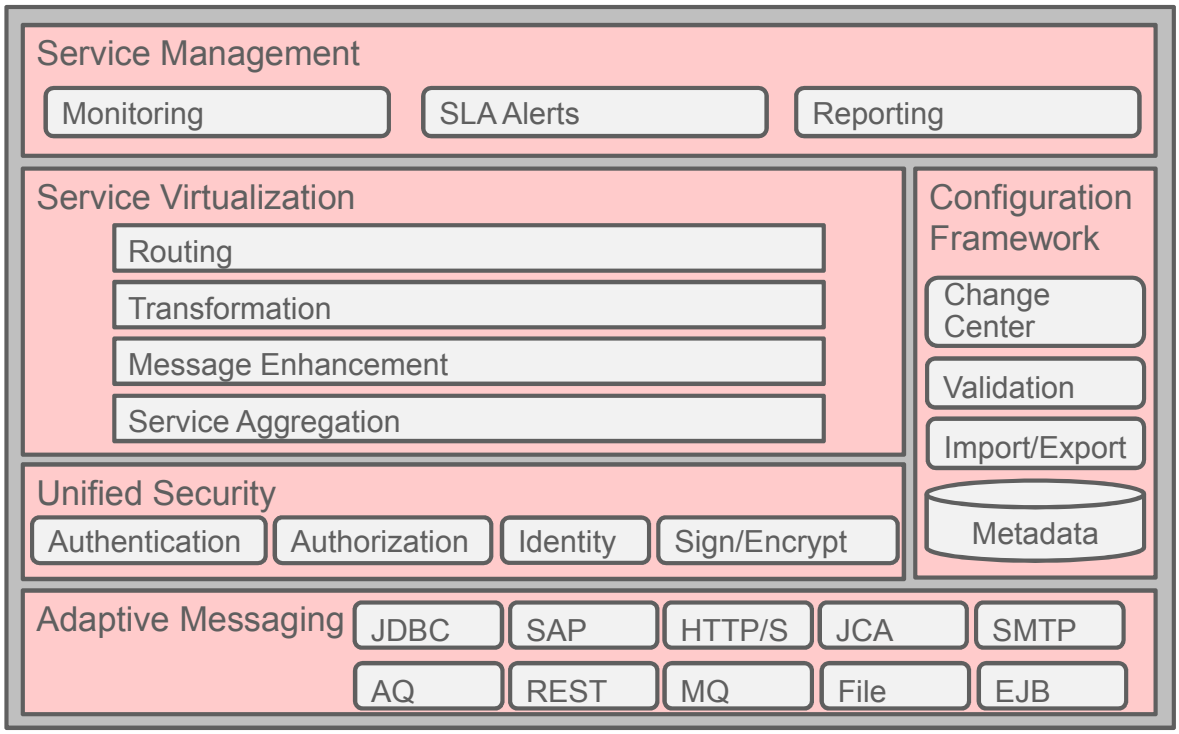
Service Bus enforces agility by replacing direct coupling with a flexible virtual endpoint for the requester/consumer.

For example, your company has an account management system, and you have created several service providers that provide services based on a set of XML document schemas that you have defined. At some point, your company decides to replace your old account management system with a newer system. This new system has web services built in. The problem is that the new system uses its own set of XML document schemas that are different from yours. For developer, it doesn't make sense to rewrite all your existing service clients. With Service Bus, you can simply create transformations that modify the new XML documents into the form expected by your current clients.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Functional Features of Oracle Service Bus



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle Service Bus allows management and integration of messages and services across a services network. Its core functional features are separated into the following categories:

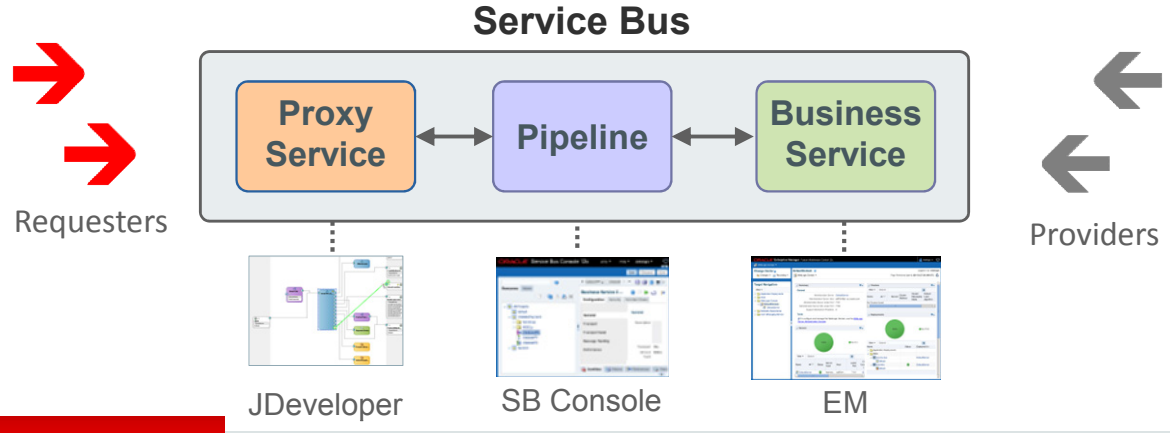
- **Service Integration:** Features used for integrating disparate service endpoints, adaptive messaging, and mediating and exposing services for reuse
- **Service Security:** Features used for service authentication and authorization, message security enforcement, and user identity validation
- **Service Composition:** Features used for configuring message routing logic, message transformation, service configuration, validation, and registry
- **Service Management:** Features used for monitoring and managing service activity and availability

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Features of Oracle Service Bus 12*c*

- Development in an IDE or web-based console (design time)
- Monitoring in Enterprise Manager Fusion Middleware Control (run time)
- Wide adapter portfolio to connect to many on-premise and cloud-based applications
- REST enablement for SOAP style services



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The list highlights some new features in the Service Bus 12*c* release.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

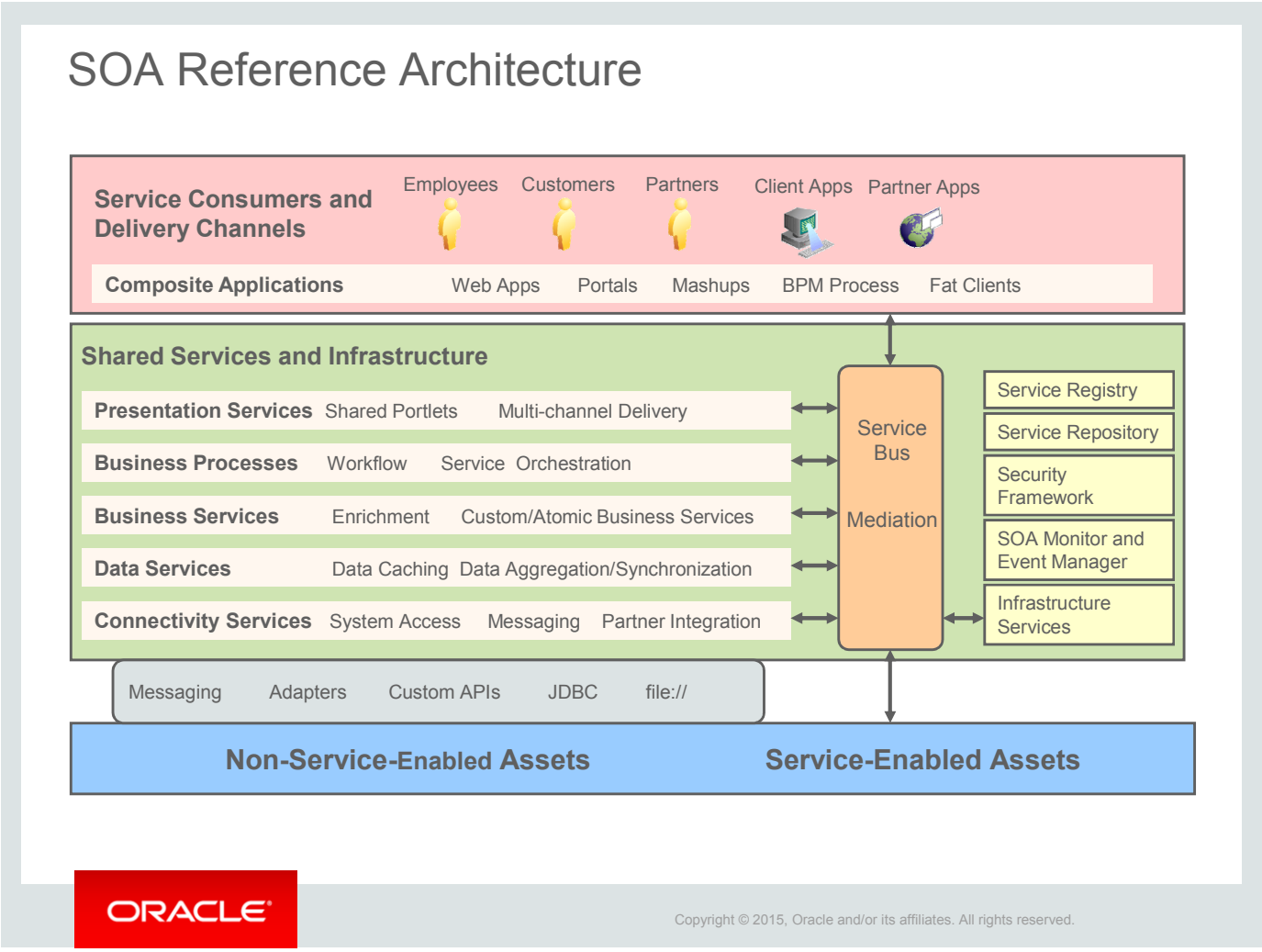
Agenda

- Enterprise Service Bus
- Introducing Oracle Service Bus
- Oracle Service Bus and Oracle SOA Suite
- Oracle Service Bus installation

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED



A SOA Reference Architecture is a blueprint or guide to creating a SOA infrastructure implementation for a business depending on the “business needs.”

The SOA Reference Architecture:

- Defines the target architecture and the principles to be used by an organization’s architects to make architecture and design decisions on their projects
- Is a key component of an effective strategy to deliver the benefits of SOA

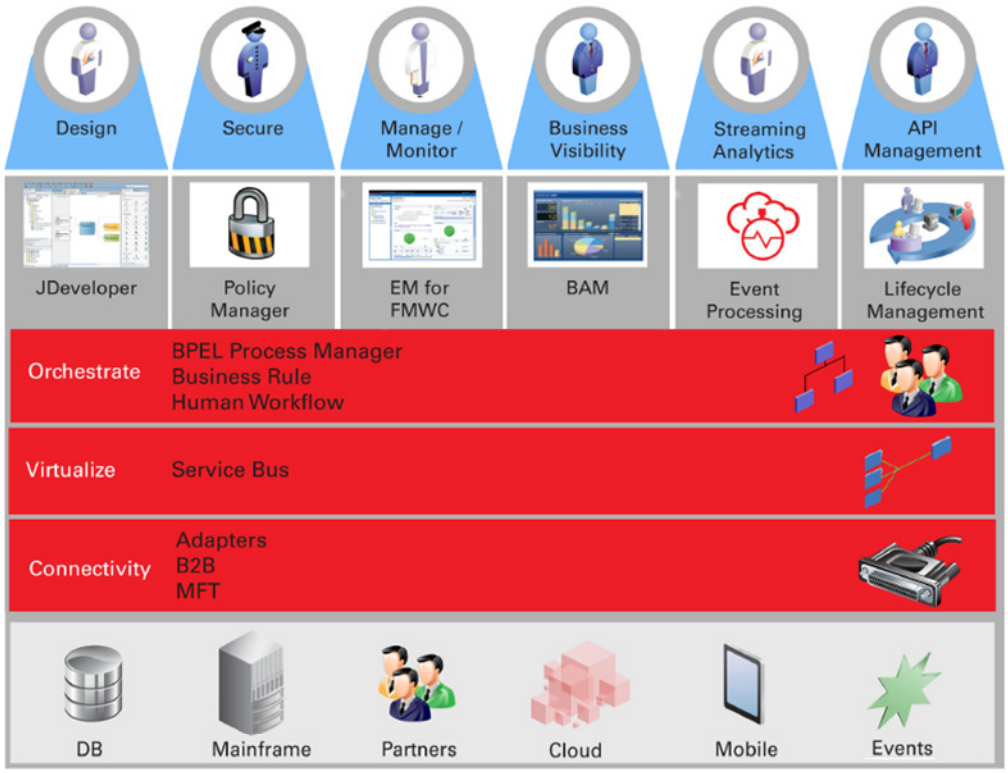
The architecture separates the users of enterprise functionality from the systems and applications that provide that functionality, placing the infrastructure for services and service delivery between them. The layers of services and their supporting infrastructure are referred to as the “innovation layer.” This analogy expresses their role in driving change in the way that IT is delivered. Existing applications, data, and middleware form the foundation from which services are drawn. Supporting and formalizing the existing enterprise activities is an important goal of a service-oriented infrastructure. Standards-based infrastructure services provide a common basis for the deployment of all other types of service.

Note: The SOA architectural framework for an organization is selected from this reference architecture based on business requirements. Not all components of the reference architecture are needed.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle Service Bus in Oracle SOA Suite



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

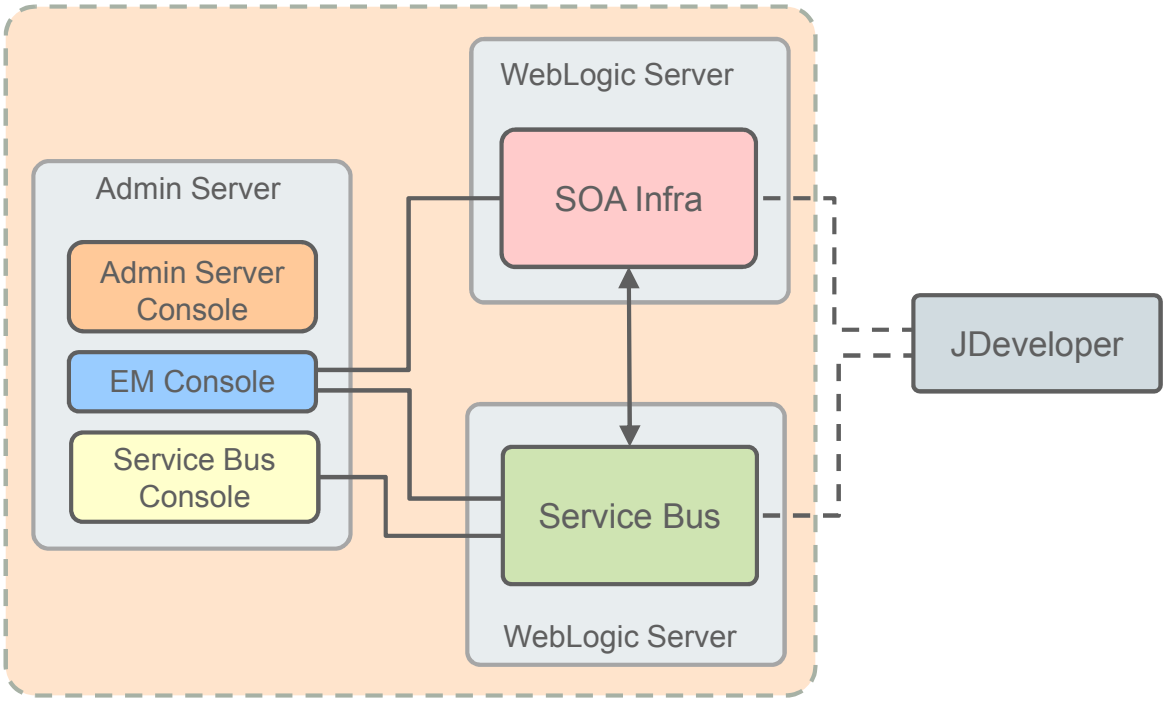
Service Bus is one important step to realize the full potential of SOA. It provides a robust shared services layer that virtualizes the endpoints from the composite application, thus providing loose-coupling between the composite and the applications. Its robust, scalable infrastructure provides the necessary high availability of endpoints from the composite perspective.

Service Bus is available stand-alone and included in Oracle SOA Suite.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle SOA Suite and Oracle Service Bus Product Architecture



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Service Bus 12*c* runs in the same version of WebLogic Server as SOA Infrastructure, where all SOA composite applications run. Invocations from OSB to SOA composite applications and vice versa are done by using native bindings (SOA-DIRECT), which is the SOA transport direct bindings, rather than as normal web services.

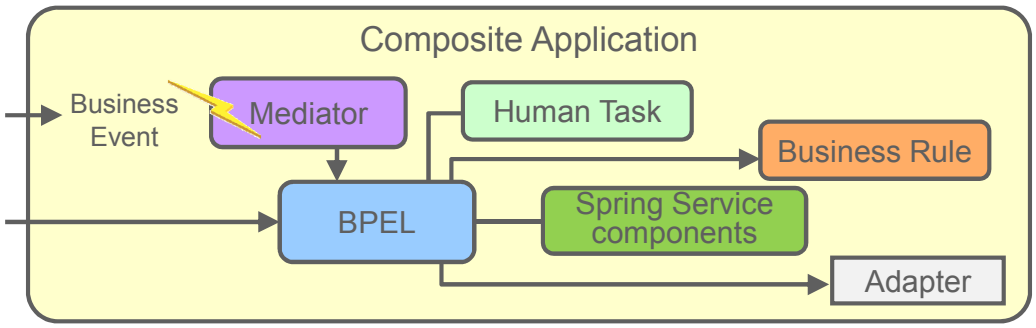
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Key Components in a SOA Composite Application

Oracle SOA Suite 12*c* supports the following components in a composite application:

- BPEL Process components
- Mediator components
- Business Rule components
- Human Task components
- Adapter services
- Business events
- BPMN Process components
- Spring Service components



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

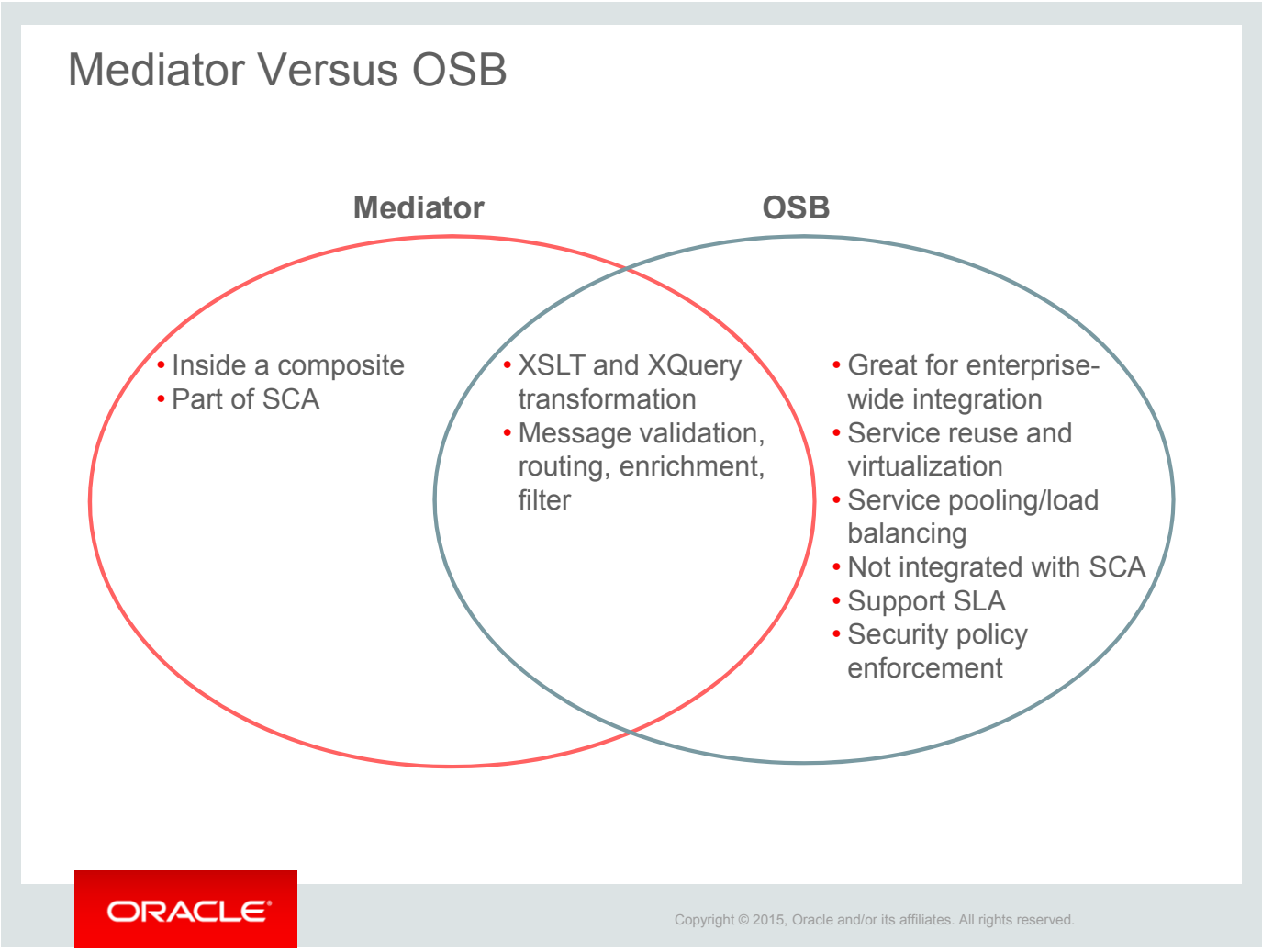
The key components you can use to build Oracle SOA composite applications include:

- BPEL Process components for service orchestration to manage more complex service interactions
- Mediator components for message routing within the composite, transformation, and filtering capabilities
- Business Rule components to enable specification of rules that are executed external to a process context and used to influence a process flow and its outcomes
- Human Task components to implement human workflow capabilities
- Adapter services for composite applications to communicate to external technology platforms, such as file system, database, and message queues
- BPMN Process components for modeling business process
- Spring Service components for integrating Java interfaces into SOA composite applications

The diagram represents a collection of cooperating components that can make up a composite application, which may expose multiple entry points. Subsequent pages in the lesson provide more information about each of these components.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED



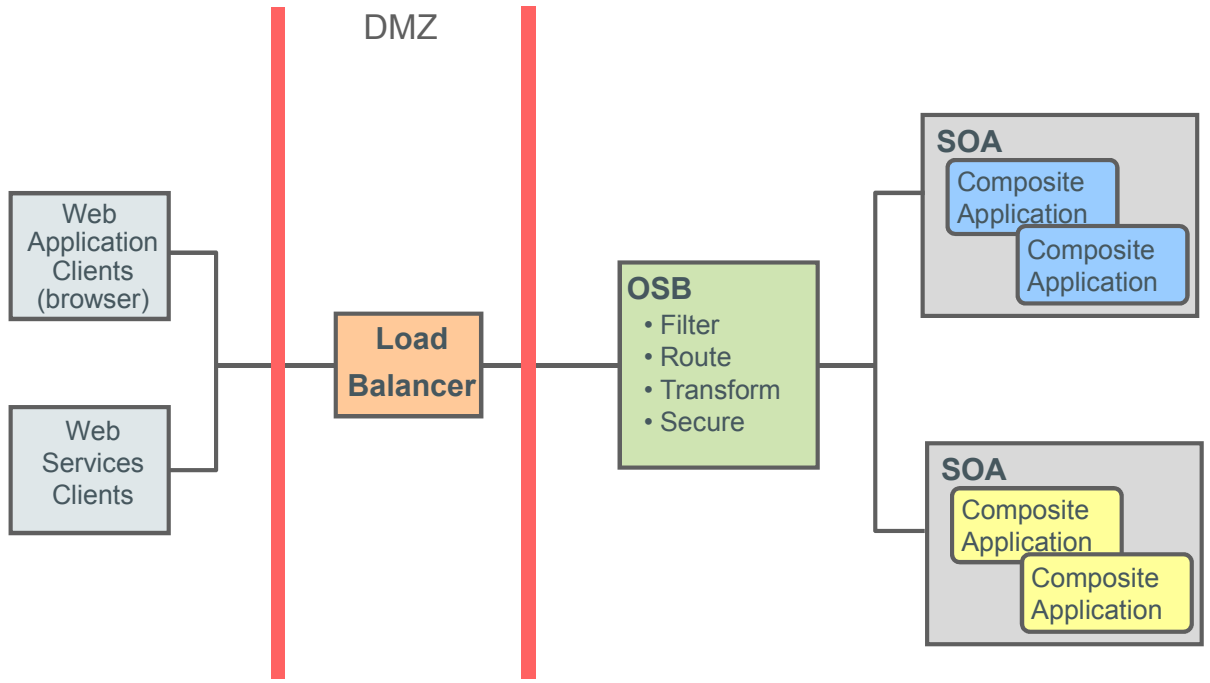
Oracle Service Bus and Mediator components have areas of functional overlap, but they target different users and have different life cycles. The key points to be aware of are:

- OSB can be useful to glue and integrate services, where a service can be a composite application implementation or another service. OSB's primary mission is to provide stand-alone service bus capabilities to shield application developers from changes in endpoint services or systems and to prevent those systems from being overloaded with requests from upstream applications. OSB focuses on solving the operational services management and scalability problems in a stand-alone service bus. The main users of Service Bus are integration developers and operations personnel in charge of shared SOA Infrastructure.
- Mediator's primary mission is to connect components within a SOA composite. Therefore, its integration scope is confined to the context of a composite application, which can be considered a narrower scope when compared with the OSB style of integration techniques. The main users of Mediator are application developers.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle SOA Suite and Oracle Service Bus Deployment Topology



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle Service Bus provides a robust shared services layer that hides the actual producer services from the consumers. OSB often performs the functions of protocol transformation, message modification, routing, logging, acting as a security gateway, and so on. For example, you might use an OSB to expose a service that was previously available only as a web service, as a JMS-based service as well.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Enterprise Service Bus
- Introducing Oracle Service Bus
- Oracle Service Bus and Oracle SOA Suite
- Oracle Service Bus installation

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Developer and Production Distributions

Developer/QuickStart	Production Distribution
One single package of an integrated server includes: • JDeveloper • WebLogic Server • SOA Suite (incl. *Service Bus*) • JavaDB • Enterprise Manager	Modularity: • Oracle SOA Suite Foundation • SOA Foundation with B2B • Oracle Managed File Transfer • Oracle Event Processing • Oracle Enterprise Repository * Does not contain database, WebLogic Server, and IDE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle SOA Suite 12*c* provides two types of distributions that allow customers to choose based on the need of their organizations.

The SOA Suite Quick Start distribution allows you to quickly install a development or evaluation environment on a single host computer. By default, the Oracle SOA Suite Quick Start installation contains Oracle JDeveloper and an Integrated WebLogic Server.

JDeveloper is preconfigured with some JDeveloper IDE extensions so that you can create various kinds of applications immediately. Oracle Service Bus is included in the preinstalled IDE extensions.

QuickStart provides quick installation on a single host computer. However, you will be limited to one Administration server. If you know you want a production environment and additional managed servers or clusters, you will need to install and configure Oracle SOA Suite production distribution.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Domain Configuration

Domain Type	Recommended use
Default Domain	Use for demonstrations or evaluation purposes, because it: • Is configured as an Integrated WebLogic Server that runs within Oracle JDeveloper • Shuts down when terminating Oracle JDeveloper • Uses JavaDB for persistent data
Stand-alone Domain	Best for demonstrations and development purposes: • Can use Oracle Service Bus Console instead of Oracle JDeveloper • Independently installed from JDeveloper • Uses JavaDB for persistence
Compact Domain	For development, proof of concept, and demonstration purposes: • Allows adding other components, such as Enterprise Scheduler, Managed File Transfer, B2B, and Business Activity Monitoring • Requires an Oracle (or supported) database

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The Oracle SOA Suite and Oracle Business Process Management Suite Quick Start options:

- Provide quick installation on a single host computer
- Support creating a development, demonstration, proof of concept, or evaluation environments
- Installed as a single administration server with no option to create managed servers (not scalable as required for production environments)
- Cannot be migrated to a production environment
- Provide three domain installation configurations described and compared in the table in the slide

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Summary

In this lesson, you should have learned how to:

- Describe what Enterprise Service Bus does
- List the capabilities of Enterprise Service Bus
- Describe Oracle Service Bus architecture and components
- List the features of Oracle Service Bus
- Describe the relationship between Oracle Service Bus and Oracle SOA Suite
- List Oracle Service Bus installation and configuration options

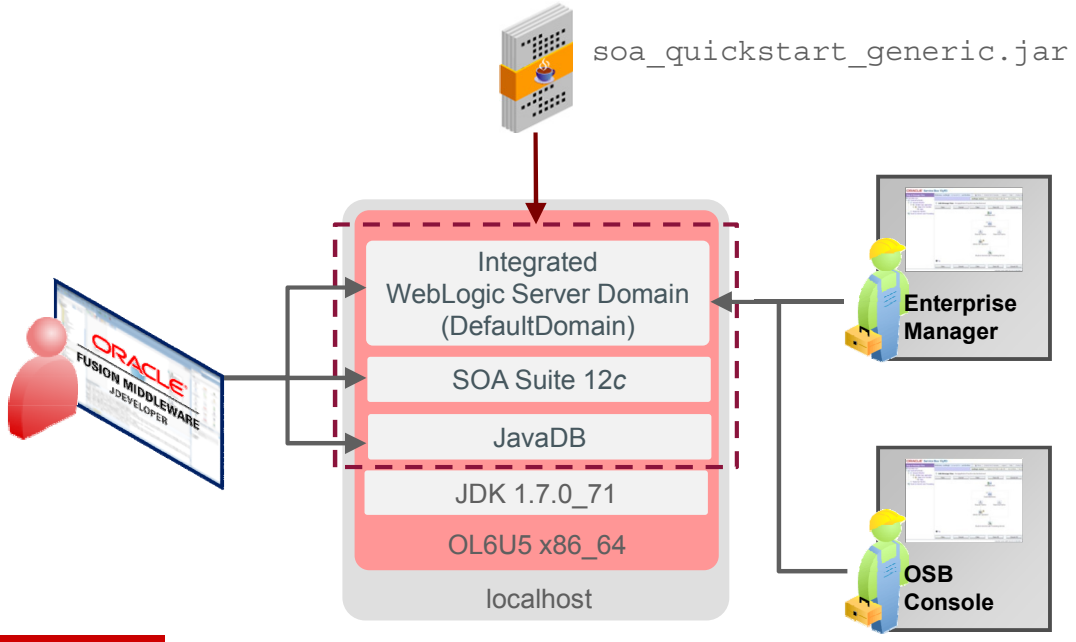
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Practice 2: Overview

2-1: Configuring the Integrated WebLogic Server's Default Domain in JDeveloper



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

3

Getting Started with Oracle Service Bus Applications

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Objectives

After completing this lesson, you should be able to:

- Describe the basics of WSDL and XSD
- Create a simple Service Bus application
- Describe the differences between JDeveloper and Service Bus Console in terms of development task support
- Populate MDS repository with source data

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

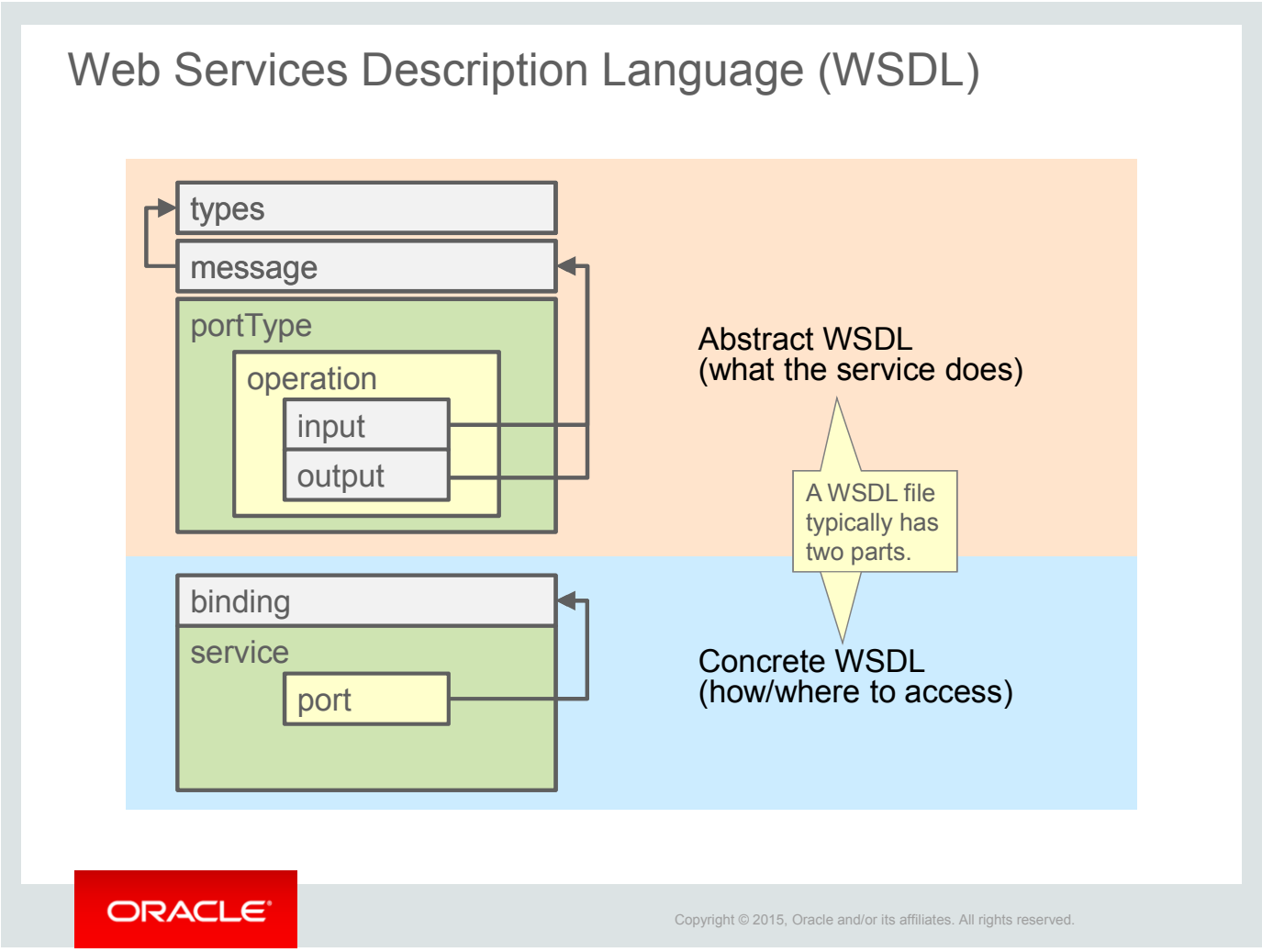
Agenda

- Basics of WSDL and XSD
- OSB components
- Creating a simple OSB application
- Features for developers

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED



Describing the structure of messages to be exchanged between a calling application and a service is important, but is not enough to enable the applications to converse. For this, a *Web Service Description Language* or WSDL document is used. This (XML) description includes the format of input, output, and fault messages, and the data types they use. It also includes the names of operations that the service provides, and how and where to find the service.

The information in a single WSDL document is organized into two parts: the *abstract* definition (what a web service does) and the *concrete* definition (how and where to access the service).

Abstract definitions include *types*, *message*, *operation*, and *portType*.

- **Types** define the data types used in messages. These types are often drawn from the XML Schema Language.
- **Messages** describe the parts of the input, output, and fault messages that are exchanged with the calling program.
- **Operations** provide a name for the action performed on messages.
- **portTypes** group messages with operations.

The WSDL documents generated by JDeveloper for local services (those within the WebLogic Server) may not include the concrete portion of the WSDL. However, the WSDL documents for services deployed outside the WebLogic Server will include the concrete elements of the

Oracle University and ORACLE CORPORATION use only

WSDL, which provide additional information about how and where to access the service:

- The ***binding*** describes how a given portType operation is transmitted, such as HTTP or SOAP (that is, the protocol), and provides information about where the service is located.
- The ***port*** specifies a combination of a network address and a binding, which constitute an endpoint.
- A ***service*** groups ports together. A service reveals to a calling program where to access the web service, and through which port. It also describes how communication messages are defined.

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Abstract WSDL

```
<?xml version= '1.0' encoding= 'UTF-8' ?>
<wsdl:definitions
    name="ReceiveData"
    targetNamespace="http://oracle.com/sca/soapservice/Basics/HelloWorld/ReceiveData"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:inp1="http://www.example.org/ns/porder"
    xmlns:tns="http://oracle.com/sca/soapservice/Basics/HelloWorld/ReceiveData"
   >
  <wsdl:types>
   <schema xmlns="http://www.w3.org/2001/XMLSchema" >
    <import namespace="http://www.example.org/ns/porder" schemaLocation="xsd/po.xsd" />
   </schema>
  </wsdl:types>
  <wsdl:message name="requestMessage">
    <wsdl:part name="part1" element="inp1:PurchaseOrder"/>
  </wsdl:message>
  <wsdl:portType name="execute_ptt">
    <wsdl:operation name="execute">
      <wsdl:input message="tns:requestMessage"/>
    </wsdl:operation>
  </wsdl:portType>
</wsdl:definitions>
```

1 2 3

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

In the sample abstract WSDL file shown in the slide:

1. Section 1 includes an `import` statement and a reference to the `po.xsd` file. Although the data types could have been included directly in this WSDL file, it is a better practice (because it promotes reuse) to separate the data type definitions into an XSD file, and then to import the file.
2. Section 2 describes the `requestMessage` message. The description includes the name of the (single, in this example) part, and the name of the element that describes the message structure of that part. The message element `PurchaseOrder` is fully described in the imported `po.xsd` document.
3. Finally, section 3 describes the port type `execute_ptt`, which groups a specific operation (`execute`) with the message `requestMessage`.

Note: The number and order of messages in the portType definition of a WSDL is important. Because you see only a single message listed in this example, you know that operation `execute` is a one-way operation. If you see a second message, you can assume that the operation returned a response upon completion.

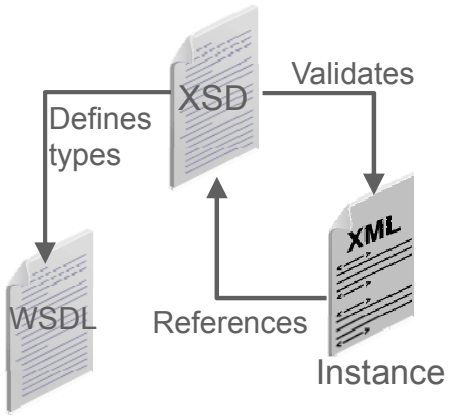
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XML Schema Definition (XSD)

XML schema is used to define the message structure of a service interface (in WSDL). XSD:

- Is an XML language that defines and validates the structure of XML documents
- Defines elements and types, such as:
  - Simple type definitions
  - Complex type definitions
  - Element declarations
  - Attribute declarations
- Supports XML namespaces and built-in simple and complex data types



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Message structures of a service interface are based on types expressed in an XML Schema Definition (XSD).

The W3C XSD language is an XML language (or vocabulary) used in an XSD document for describing and constraining the content of XML documents. The XSD is used to validate the structure of an XML document.

An XML document whose structure is based on the definitions in an XML schema is called an instance document of that XML schema. The slide shows an XML schema document stored separately from the XML instance document that it describes and validates.

The introduction of the XML schema allowed XML technology to represent data types in a standard format. The data types give a precise way of specifying the type of content that can be held in elements and attributes of an XML document.

In an enterprise application environment, XSDs are very important to ensure that messages exchanged during interactions between service consumers and providers are consistent, well-formed, and valid structures. In fact, WSDL message types are defined in terms of XSD elements and types to ensure a high degree of consistency and visibility between a service and the client invoking the service.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XSD: Example

```
<ns0:item>
    <ns0:productId>SKU301</ns0:productId>
    <ns0:name>Music Player 1Gb</ns0:productName>
    <ns0:price>45</ns0:price>
    <ns0:quantity>3</ns0:quantity>
</ns0:item>
```

XSD describes XML.

```
<xsd:complexType name="OrderItemType">
    <xsd:sequence>
      <xsd:element name="productId" type="xsd:string" minOccurs="1"/>
            <xsd:element name="name" type="xsd:string"minOccurs="1"/>
            <xsd:element name="price" type="xsd:decimal" minOccurs="1"/>
            <xsd:element name="quantity" type="xsd:int" minOccurs="1"/>
        </xsd:sequence>
    </xsd:complexType>
```

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The example in the slide shows a snippet of an XSD, which describes a complex type and its component elements, each of which is of a built-in data type. Note also the `minOccurs` attribute, which indicates that this element is *required*, and must occur at least one time in any valid message. The upper code box in the slide shows a piece of an XML document that is described by the XSD sample.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Basics of WSDL and XSD
- OSB components
- Creating a simple OSB application
- Features for developers

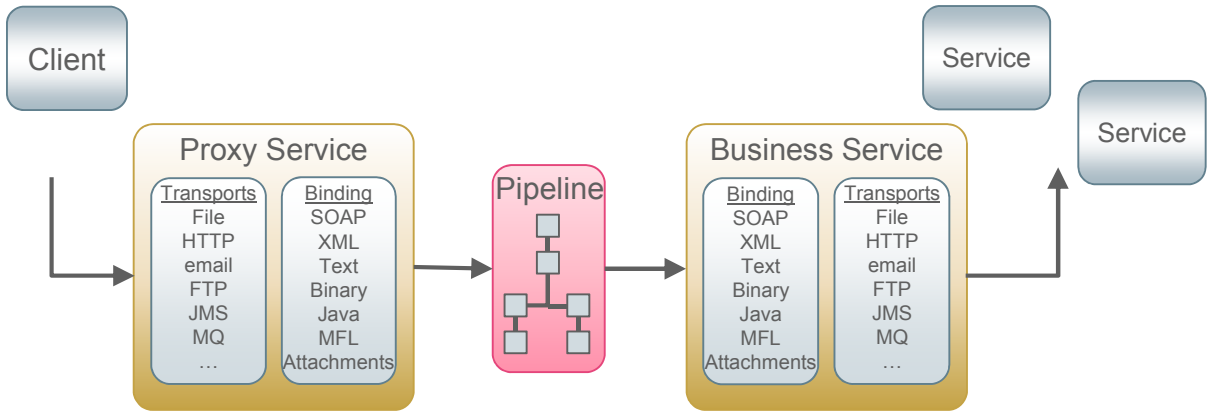
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

OSB Components

- Business services
- Proxy services
- Pipelines/split-join



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Proxy services and business services define the endpoints in a Service Bus system. They include the binding and transport layers, and are the points at which Service Bus communicates with external services, including producers and consumers. The message flow is typically defined in a pipeline, but can also be defined in a split-join for parallel processing.

The figure in the slide illustrates the flow of data through Service Bus, from inbound proxy service to outbound business service.

The processing of messages through Service Bus occurs in the following sequence of events:

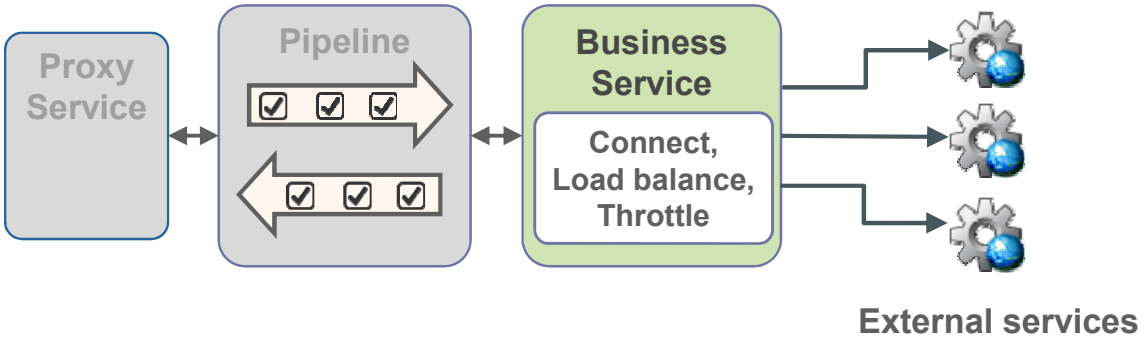
1. A client sends a message to Service Bus by using a specific transport protocol.
2. A transport provider processes the inbound message, handling communication with the service client endpoint and acting as the entry point for messages into Service Bus.
3. The binding layer packs and unpacks messages, handles message security, and hands messages off to the pipeline.
4. The pipeline performs any transformation, validation, logging, and reporting, and then routes the message to an endpoint (either a business service or another proxy service).
5. Service Bus processes the response message in a similar manner as the preceding steps.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Business Services

- Act as a wrapper of the external services
- Provide an abstraction layer that enables functionalities like load balancing, service throttling, and service pooling
- Defined by whether they are based on a WSDL web service or an OSB transport



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Business services are Service Bus definitions of the enterprise services that exchange messages during business processes. They define enterprise web services to which Service Bus is a client. Those external web services are implemented in and hosted by external systems, so Service Bus must know what to invoke, how to invoke it, and what to expect as a result. Business services model those interfaces so that Service Bus can invoke the external services.

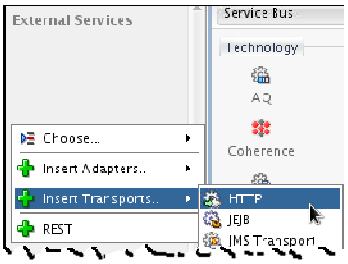
Business services allow the definition of properties for controlling how the external service is invoked. For example, business service can specify multiple endpoints for load balancing and high availability.

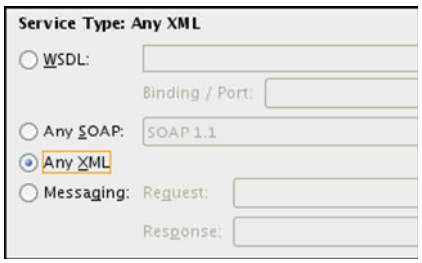
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Service Types for Business Services

- Transport-typed business service can have one of the following service types:
  - WSDL-based service
  - Any SOAP service
  - Any XML service
  - Messaging service
  - REST service
- The available types are restricted based on the transport used to communicate with the external service.
- Example of business service using HTTP transport:





Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Service Bus supports various service types ranging from conventional web services (using XML or SOAP bindings in WSDL files) to non-XML (generic) services. When you create a transport-typed business service, you also need to further define the service by specifying and configuring the service type. The service types you can select are restricted based on the transport used to communicate with the service endpoint.

A business service can have one of the following service types, identified by the types of messages it processes:

- **WSDL-based service:** This service type is generated from an existing WSDL document or one that you create at the same time you create the business service. When creating a WSDL-based service, you need to specify the port or binding to use.
- **Any SOAP service:** This service type exchanges SOAP messages. SOAP messages are constructed by wrapping the contents of the header and body variables inside a `<soap:Envelope>` element.
- **Any XML service (non-SOAP):** With this service type, messages to XML-based services are XML, but can be of any type the business service configuration allows.
- **Messaging service:** This service type can receive messages of one data type and respond with messages of a different data type. Supported data types include XML, Message Format Language (MFL), text, untyped, binary, Java, and attachments where the interface is not described by WSDL.

Oracle University and ORACLE CORPORATION use only

- **REST service:** This type of service is based on the REST binding, and can be generated from an existing WADL or one that you create at the same time you create the business service.

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

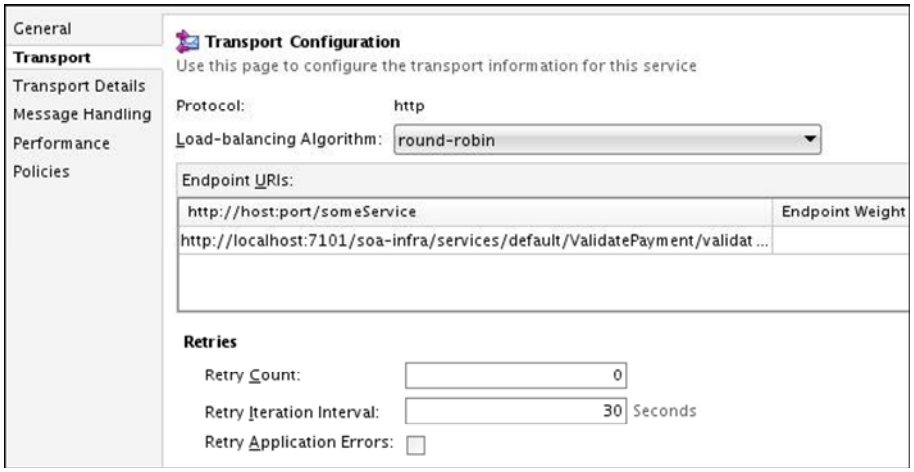
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Business Service Configuration

Business Service configuration, defining the interface it implements, includes:

- Communication interface
- Type of transport
- Message handling
- Performance tuning
- Security requirements
- SLA alert rules



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Business Service is configured by specifying their interface and the type of transport in WSDL or WADL.

A business service:

- Defines a protocol used with the service
- Defines a URL that is called when the service is invoked

The properties of a business service depend on which protocol is selected.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Proxy Services

- Act as a gateway for applications and services to access Service Bus
- Define the interface for consumers
- Defined by whether they are based on a WSDL web service or an OSB transport

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Proxy services provide the interface that service consumers use to connect with back-end services through Service Bus. They are definitions of intermediary web services that Service Bus hosts locally. Service Bus uses proxy services to route messages between business services (such as queue and databases) and service clients (such as presentation applications or other business services).

If all the service consumers no longer access the external services directly, but only through the proxy service, a lot of the features of OSB can be used transparently by a service consumer, such as SLA monitoring and alerting, service pooling and throttling, and so on.

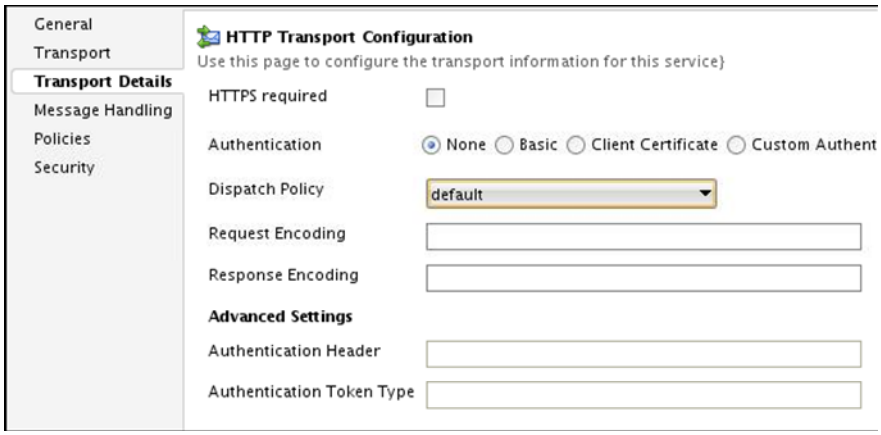
You can create a proxy service without routing to a target service.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Proxy Service Configuration

- Communication interface
- Type of transport
- Transport settings
- Message handling
- Security settings



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

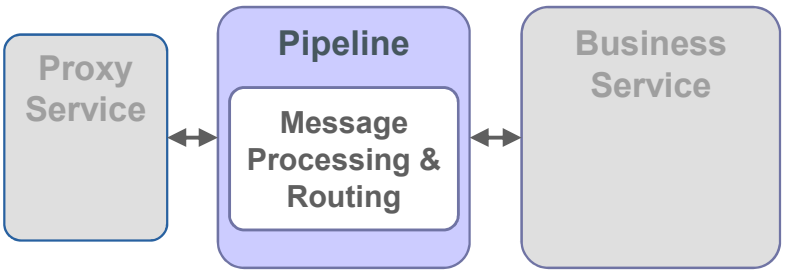
Proxy Service configuration includes adding security policies, modifying security settings, and setting up SLA alert rules. The information you can modify depends on how the service was originally configured.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Pipelines

- Define message routing and transformation logic, as well as message handling options
- Allow actions to be performed on a message
- Defined in XML



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Pipelines define message routing and transformation logic, as well as message handling options. This logic includes activities such as transformation, publishing, logging, reporting, alerts, and exception management. Each of these activities are configured as individual actions within the message flow. Both JDeveloper and the Oracle Service Bus Console provide graphical modeling tools to help you model your pipelines.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Quiz

Which service should be used for connecting to a target system?

a. Proxy service
b. Pipeline service
c. Business service
d. Split-join service

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Answer: c

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

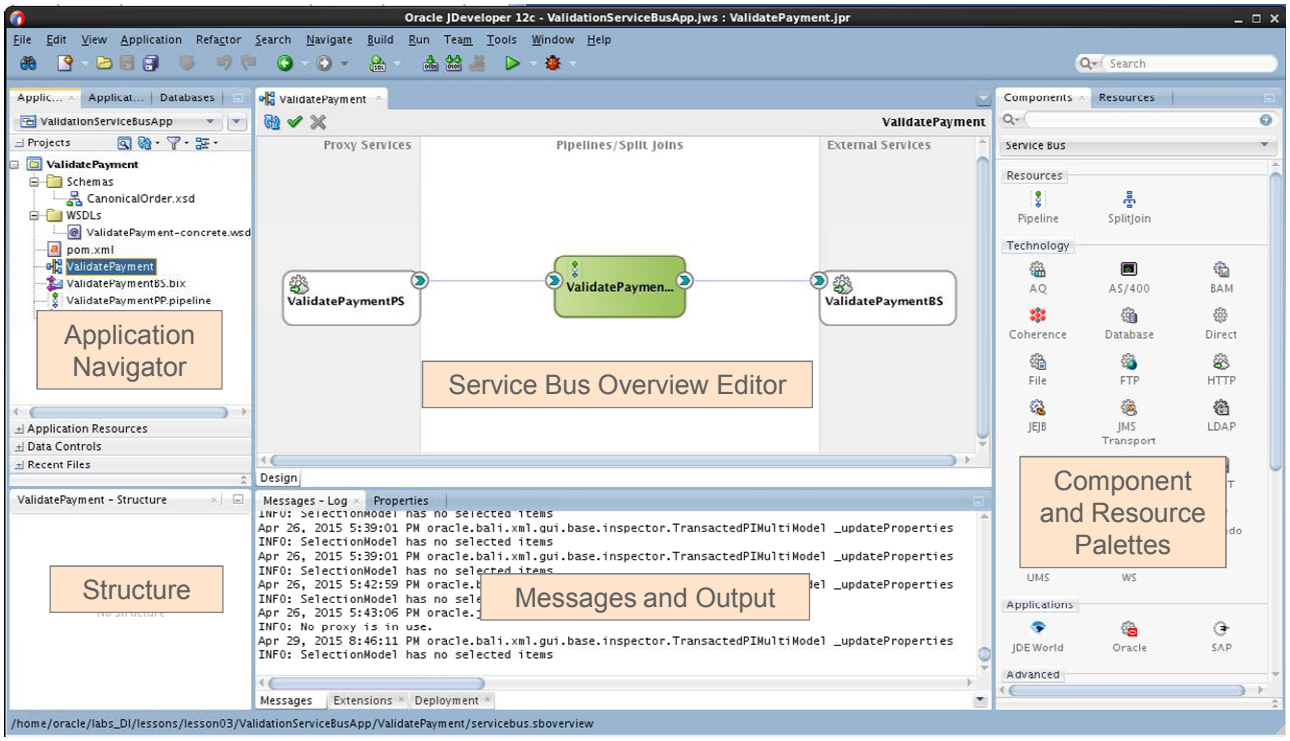
- Basics of WSDL and XSD
- OSB components
- Creating a simple OSB application
- Features for developers

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Developing Service Bus in JDeveloper



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Service Bus uses editors, wizards, and dialogs in JDeveloper to create and configure Service Bus applications. Some of these are specific to Service Bus components, and some are standard JDeveloper tools. For example, proxy service and business service editors are specific to Service Bus, but XSLT and XQuery mappers are shared by other products in JDeveloper. Service Bus shares many features with SOA Suite components in JDeveloper.

The figure in the slide shows Service Bus artifacts in JDeveloper, including project files in the application navigator, the Service Bus Overview Editor in the center, and the Components window on the right.

- The Application Navigator displays the key files for all the resources and services included in the Service Bus project.
- The Service Bus Overview Editor is where you define services and message flows.
- The Components window lists all the different Service Bus and SOA Suite components you can drag onto the Overview Editor design canvas.

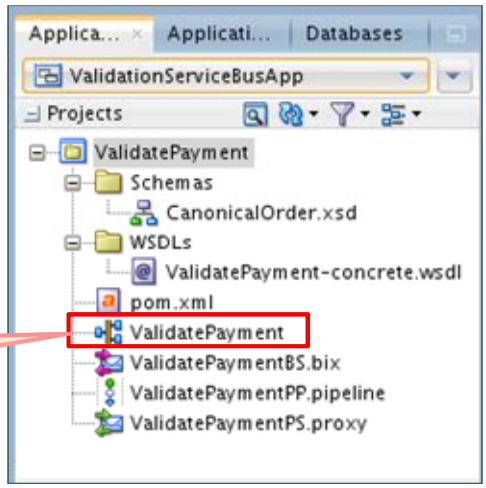
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Projects and Resources

- An OSB project is a container for resources.
- Various resources can be created within a project:
  - Project overview file
  - Proxy services
  - Business services
  - Pipelines
  - XML schemas
  - WSDLs
  - Transformations

Project overview file (`servicebus.sboverview`)



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle Service Bus resources (for example, services, WS-Policies, WSDL, XQuery transformations, and so on) reside in exactly one project. Projects do not overlap. Resources can be created directly under a project, or they can be further organized into folders. You can reference any resource regardless of the project in which it resides.

Project Explorer organizes configurations and resources into projects and folders.

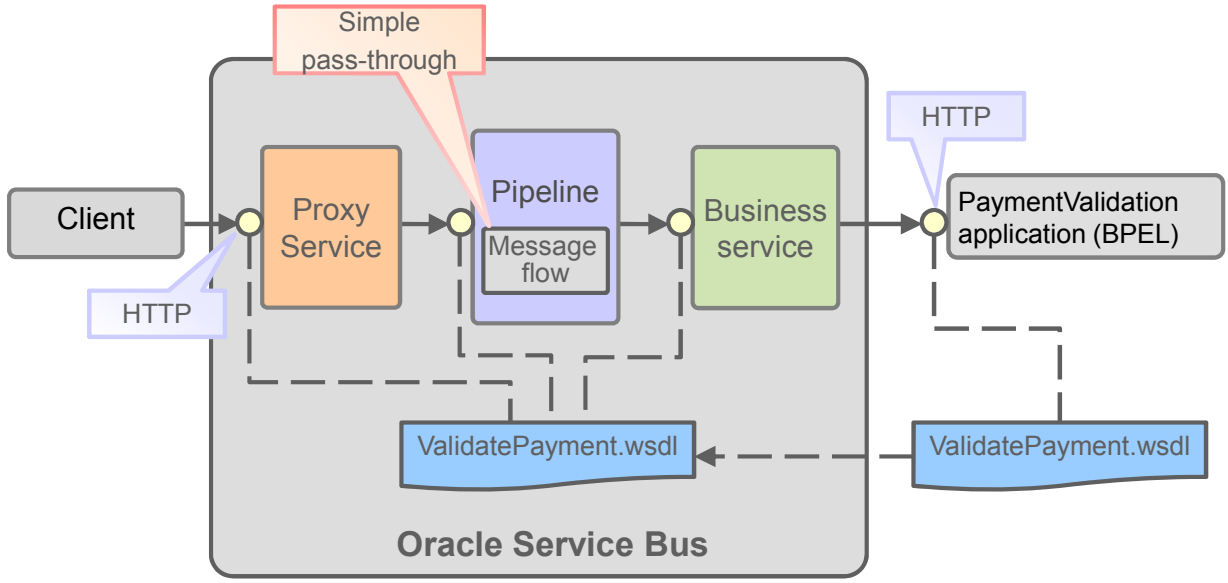
Note: When you create a domain in Oracle Service Bus, a default project is created.

An XML file is automatically created when you create a Service Bus project. This file describes the entire Service Bus project, including services, resources, references, and wires. In the Application Navigator, this file has the same name as the project. In the file system, it is named as `servicebus.sboverview`.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

A Basic Service Bus Application



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

This example shows that Service Bus needs to only support the same web service interface as the back-end service (PaymentValidation) does. Here, the proxy service offers the same SOAP-based web service interface as the business service/external service, so it is basically doing a pass-through of the request and response message.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

High-Level Steps of Creating a Service Bus Application

1. Create a project folder.
2. Define the folder structure.
3. Register the WSDL of the service that you want to virtualize.
4. Create (or register) the Business Service.
5. Create the Proxy Service.
6. Define the message flow (optional).

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

After creating the empty OSB project, you prepare a folder structure to be used to organize the project. OSB allows you to use folders to build a project structure that helps to better find the various artifacts inside the OSB project.

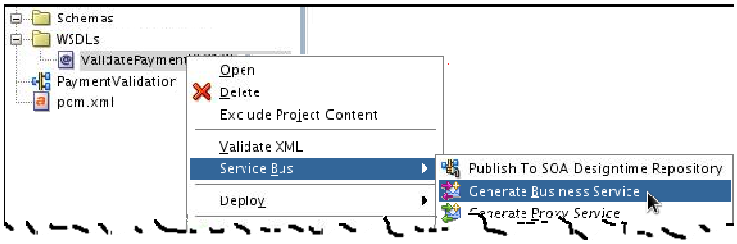
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

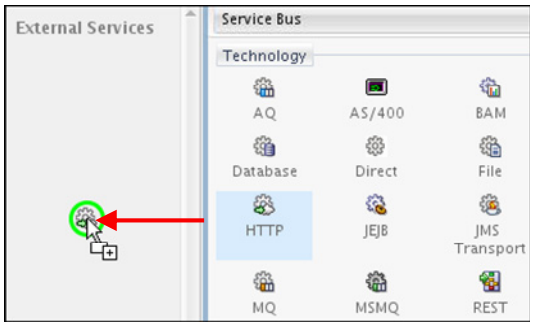
Creating a Business Service

A Business Service can be created:

- From a WSDL service definition document



- By dragging and dropping either an adapter or a transport from the component palette



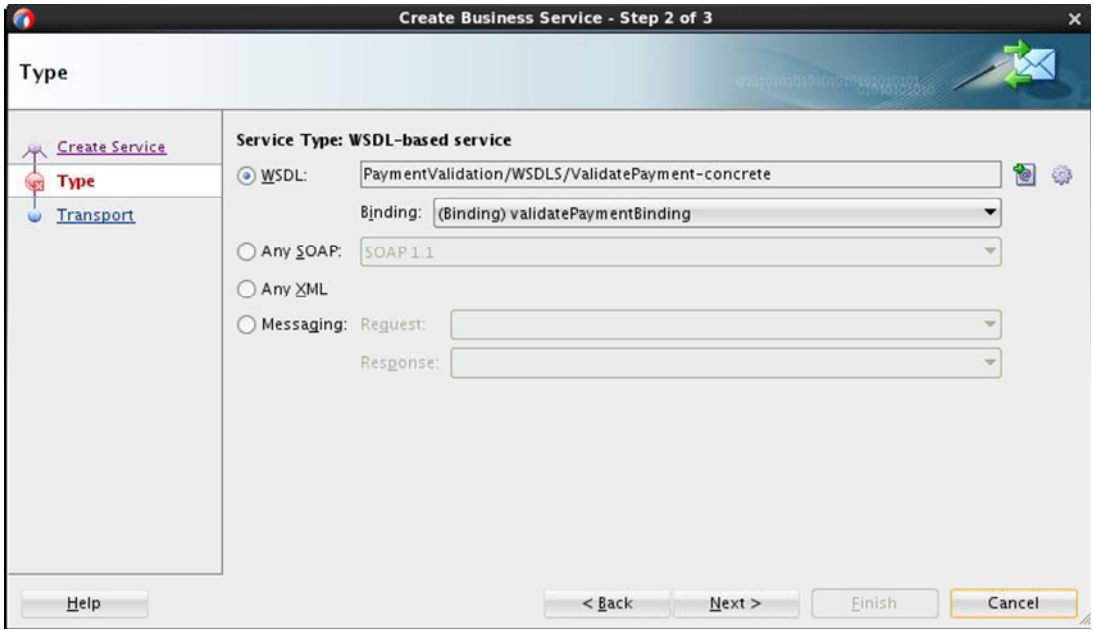
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Creating a Business Service

- Selecting a service type



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

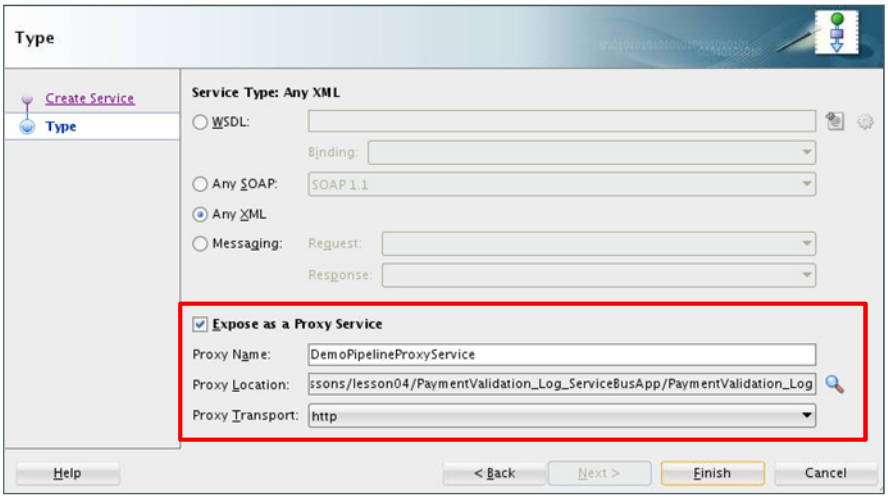
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Creating a Proxy Service

A Proxy Service can be created:

- From a WSDL service definition document
- By dragging and dropping either an adapter or a transport from the component palette
- By exposing a pipeline or split-join



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When creating a pipeline or split-join, you can expose it as a proxy service that creates a proxy service with the same service interface as the pipeline and a wire from the Proxy Service to the Pipeline.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Basics of WSDL and XSD
- OSB components
- Creating a simple OSB application
- Features for developers

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Features for Developers

- Quick Start to develop
- Reusability
  - Import and export projects and resources
  - MDS
  - Pipeline templates
- Debugging and testing
- Continuous integration with Maven

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Quick Start to develop:

- Single Package Installer
- Install and set up to “Hello World” in 30 minutes or less

Connectivity: Support wide range of adapters and transports

Reusability:

- Share and reuse from MDS
- Pipeline Templates and Placeholders to reuse code and to promote best practices

Debugging: Integration debugger with support for breakpoints and data watches

Continuous integration with Maven:

- Maven plug-in for Service Bus projects
- Project Object Model (POM) Editor
- Full-lifecycle support

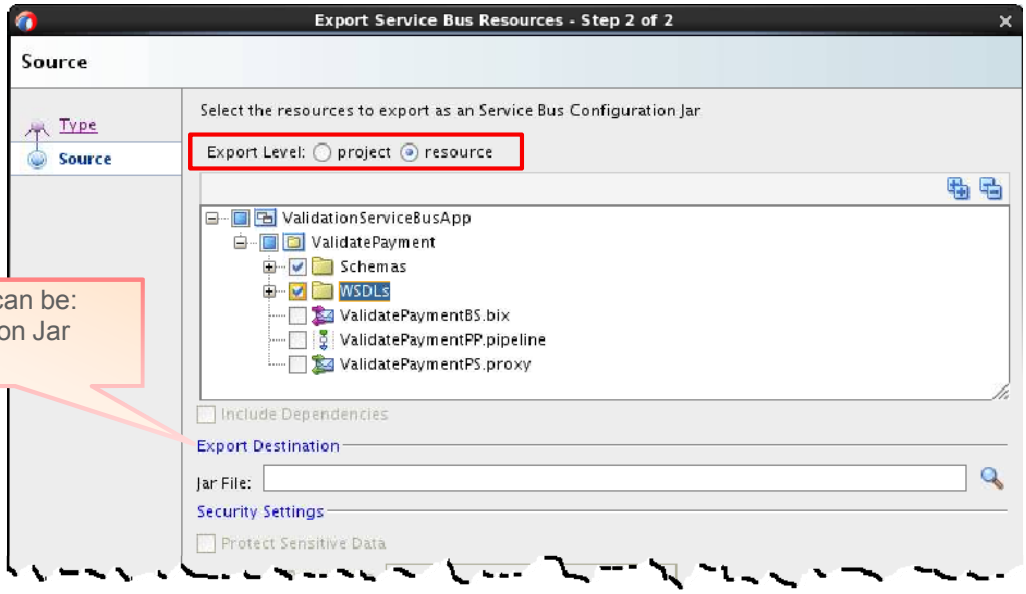
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Import and Export Resources

Propagation resources between:

- Development, testing, and production environments
- Development tools: JDeveloper, Service Bus Console



Destination can be:
- Configuration Jar
- Server

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Service Bus provides import and export features to help you move your Service Bus projects and resources between domains and between development tools. For example, when you move from development to testing, you can export projects from the development environment and import them into the testing environment.

You can also export and import resources to move them between JDeveloper and the Oracle Service Bus Console. For example, you create certain artifacts, such as JCA adapter files and WADL files, in JDeveloper. To use those artifacts in projects in the Oracle Service Bus Console, exporting them from JDeveloper and importing them to the console maintains any references between those artifacts.

Service Bus supports the ability to import and export artifacts and projects at fine-grain level. You may import artifacts individually, such as a WSDL or schema, or whole projects. You may control whether dependencies are included or not.

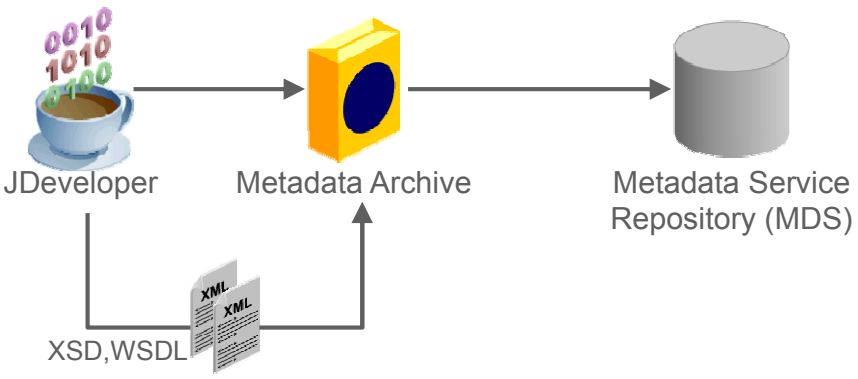
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Share Artifacts with MDS

Metadata services (MDS)

- A repository for sharing reusable artifacts such as WSDLs and schemas
- Support both design-time and runtime MDS
- Share artifacts between SOA and Service Bus projects



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

MDS repository can be used as the central location for storing and referencing shared service assets, such as XSDs, WSDLs, and other service documents, which can be deployed in a shareable archive format known as the Metadata archive (`.mar` files).

Design-time and runtime MDS: Design-time MDS is file based where you keep all the artifacts in local file system whereas runtime MDS is DB based where you upload all the artifacts in DB schema.

Use of the MDS repository helps in preventing duplication of artifacts and the resultant synchronization challenges.

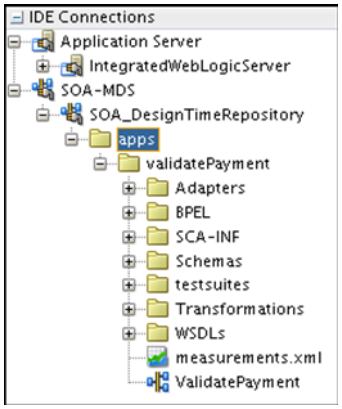
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Populating Shared Data with File-Based MDS

How does it work:

- A file-based, design-time MDS repository connection is automatically included when you create a SOA composite application.
- `/apps` is the folder recognized by SOA/OSB for shared data.
- Operations supported:
  - Export (JAR file)
  - Import (JAR file)
  - Transfer



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When you create a SOA composite application, the default SOA-MDS connection named `SOA_DesignTimeRepository` is automatically included. The `/apps` folder in the design-time MDS repository is automatically created. The connection name cannot be changed, but the default repository location can be modified to point to another folder or source control location.

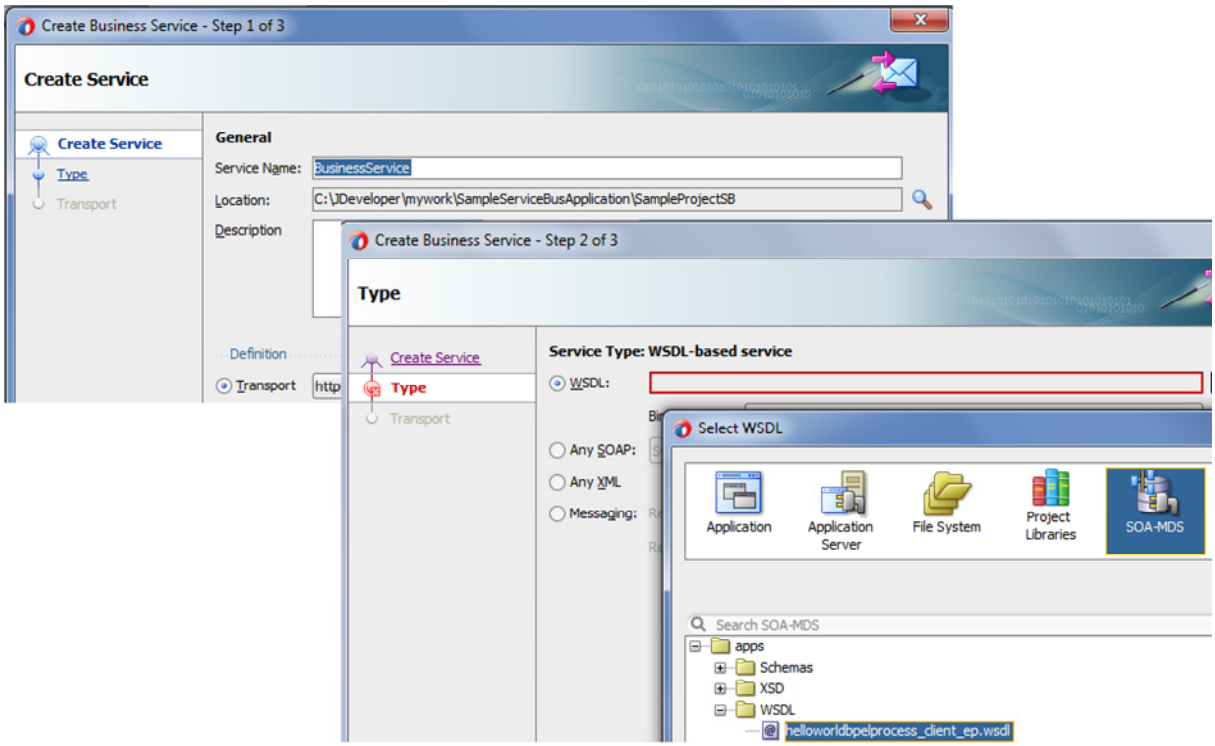
You can perform the following operations against the design-time MDS repository:

- Export selected contents of the `/apps` folder to a JAR file. The `/apps` folder itself is not included in the JAR file.
- Import the contents of a JAR file to the `/apps` folder. If the JAR file includes `/apps` as the root folder, it is created under the `/apps` folder of the design-time MDS repository, which gives you a top-level directory structure of `/apps/apps`.
- Transfer the contents of the `/apps` folder of one MDS repository to another MDS repository.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Using Shared Data in MDS



ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

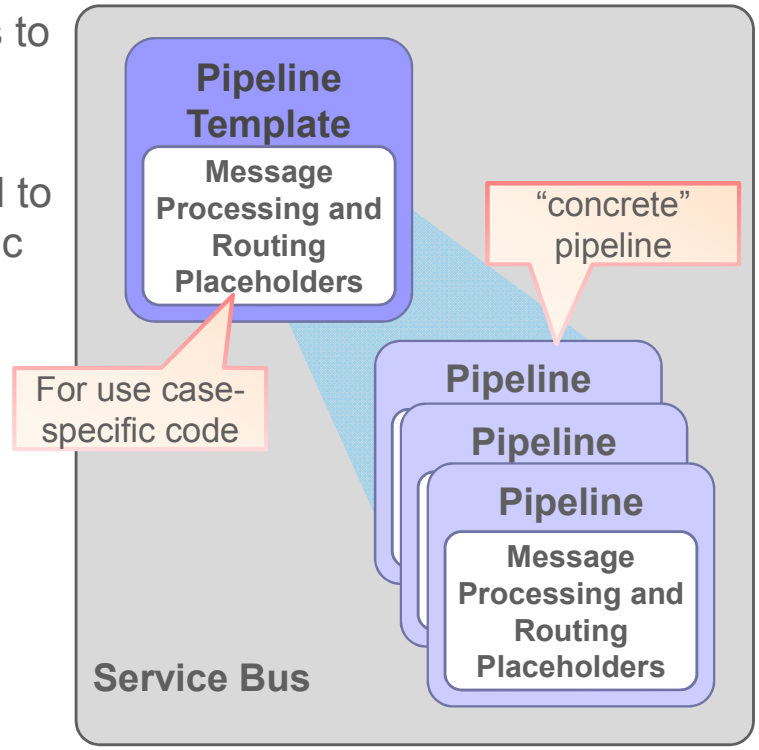
You can use the artifacts in MDS as the source of your WSDLs and schemas files. The image in the slide depicts using WSDL file in MDS to generate Service Bus business service.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Promote Reusability and Best Practices with Pipeline Template

- Use Pipeline Templates to create and provide best-practice patterns.
- Placeholder is designed to protect use case-specific code from template updates.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Similar to SOA composites, Service Bus also supports the use of templates. Service Bus templates are pipeline-based. There are two types of templates; Unlinked, which is a copy of the template, and Linked, Pipeline stays connected to template. In the template you can use Placeholders so permit changes to that part of the Pipeline. The benefits of using templates are:

- Reuse of message processing logic
- Shortened development time
- Creation of a consistent approach to implementing APIs

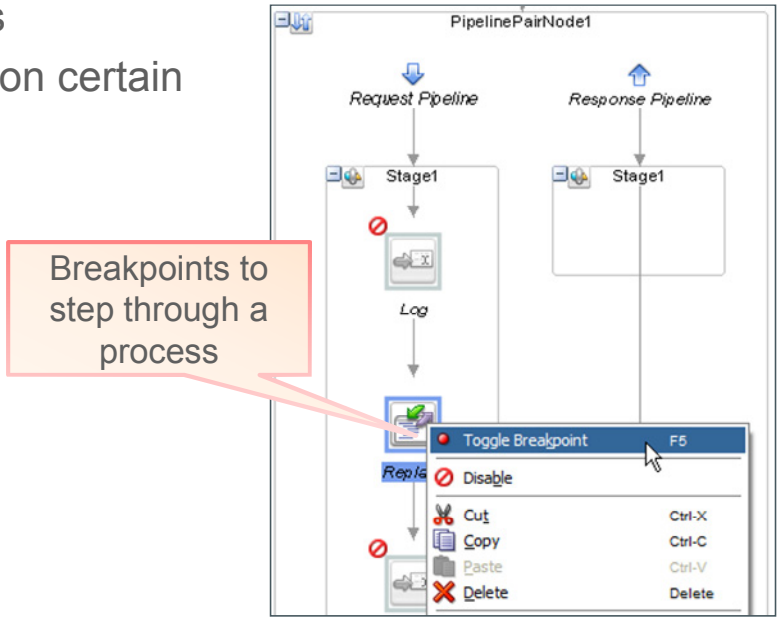
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Test and Debug Integrations

Use the integration debugger to step through a pipeline as it executes for:

- Identifying problems
- Validating behavior on certain conditions



Breakpoints to step through a process

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Use the debugger to step-debug an integration message processing logic to identify problems or to validate behavior on certain conditions.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

OSB Maven Support

- OSB provides:
  - A plug-in for pulling or downloading all required libraries and plug-ins into a local repository
  - A plug-in to package and deploy an OSB project
  - An archetype to create an OSB project
- Service Bus Maven plug-in goals:
  - `package`
  - `deploy`

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

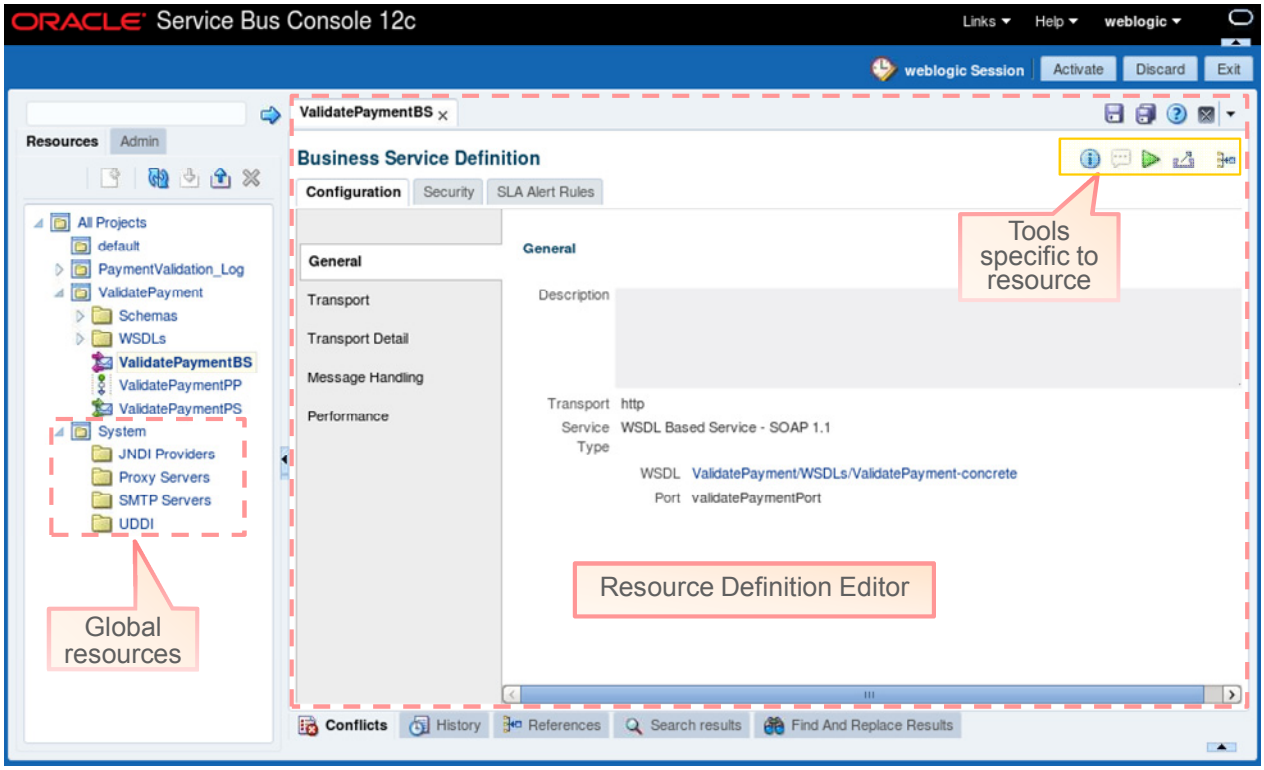
With Oracle SOA Suite 12*c*, you can build all SOA and Service Bus projects with Maven natively, without calling a utility like configjar or ANT from Maven.

The Oracle Service Bus development Maven plug-in provides goals that are specific to Service Bus projects and that can be used in conjunction with standard Maven goals to build and manage Service Bus projects and applications throughout the lifecycle phases.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle Service Bus Console



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle Service Bus Console is a web-based console where you can create and configure most Service Bus resources, test the resources, and activate your changes to the run time. You can also import and export Service Bus configuration JAR files. The console uses a change session mechanism similar to the WebLogic Server Administration Console, where you can complete your changes within a session, and when you are satisfied with those changes, activate them into the run time.

Like JDeveloper, all Service Bus resources in Service Bus Console, such as services, WSDLs, and XQuery transformations can be created directly under a project, or they can be further organized into folders.

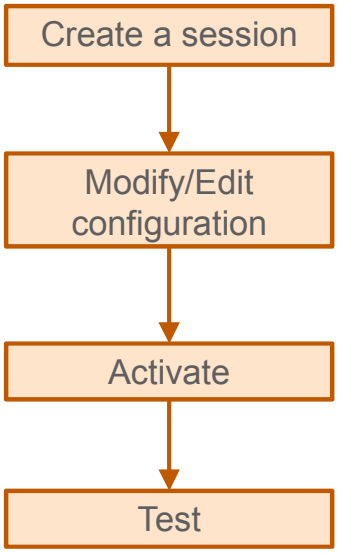
The Project Navigator includes a default project named System, which contains a flat file of global resources such as JNDI providers, SMTP servers, proxy servers, and UDDI registries. The global resources can be used by the resources in any of the Service Bus projects you create.

Each type of resource can be configured by using an editor in the console that is specific to that type of resource. Editors have a standard set of tools in the upper right toolbar (Save, Save All, Help, Close). In addition, each editor includes tools specific to the resource being configured. These tools allow you to perform additional tasks, like start the Test Console, start the Pipeline Editor, export a WSDL file, view references, and so on. The figure in the slide shows the Business Service Definition Editor.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Working with Sessions



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When you create and modify Service Bus resources in the console, you do so within the context of a session. When you are ready to test your resources, you activate the session to promote them to the run time. You must save all changes before activating or exiting a session.

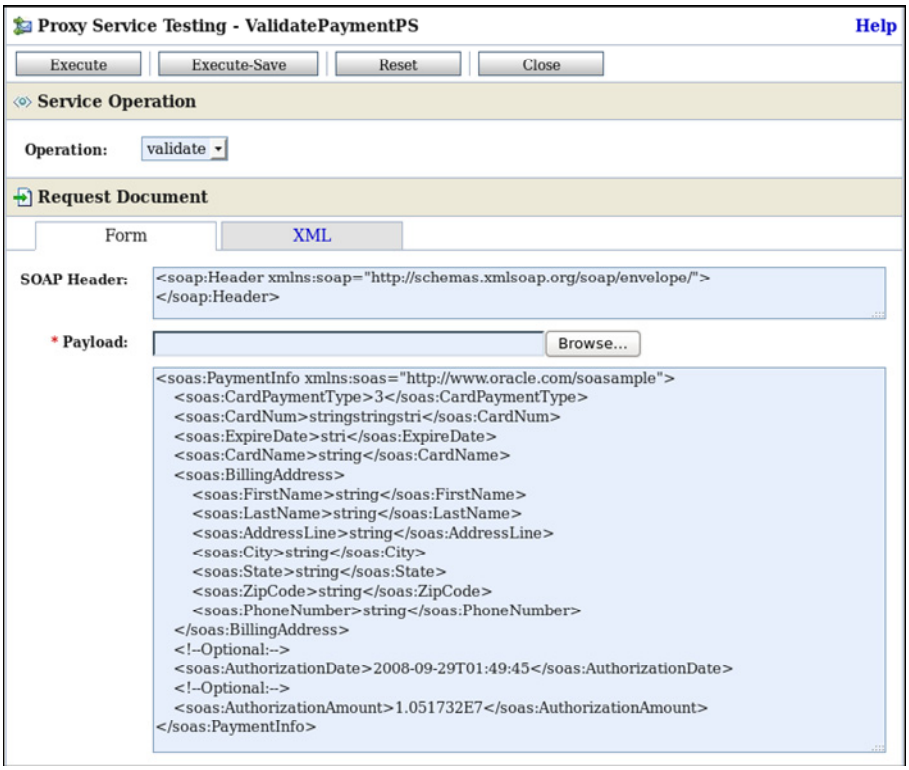
Service Bus can continue to receive and process requests for services while configuration changes are being made in a session. These changes do not affect the runtime configuration until you activate the current session. Configuration and resource changes you make are tracked, and you can undo or redo changes, resolve conflicts, maintain dependencies among resources, and test changes in the Test Console.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Test Console

- Test services
- Trace pipelines



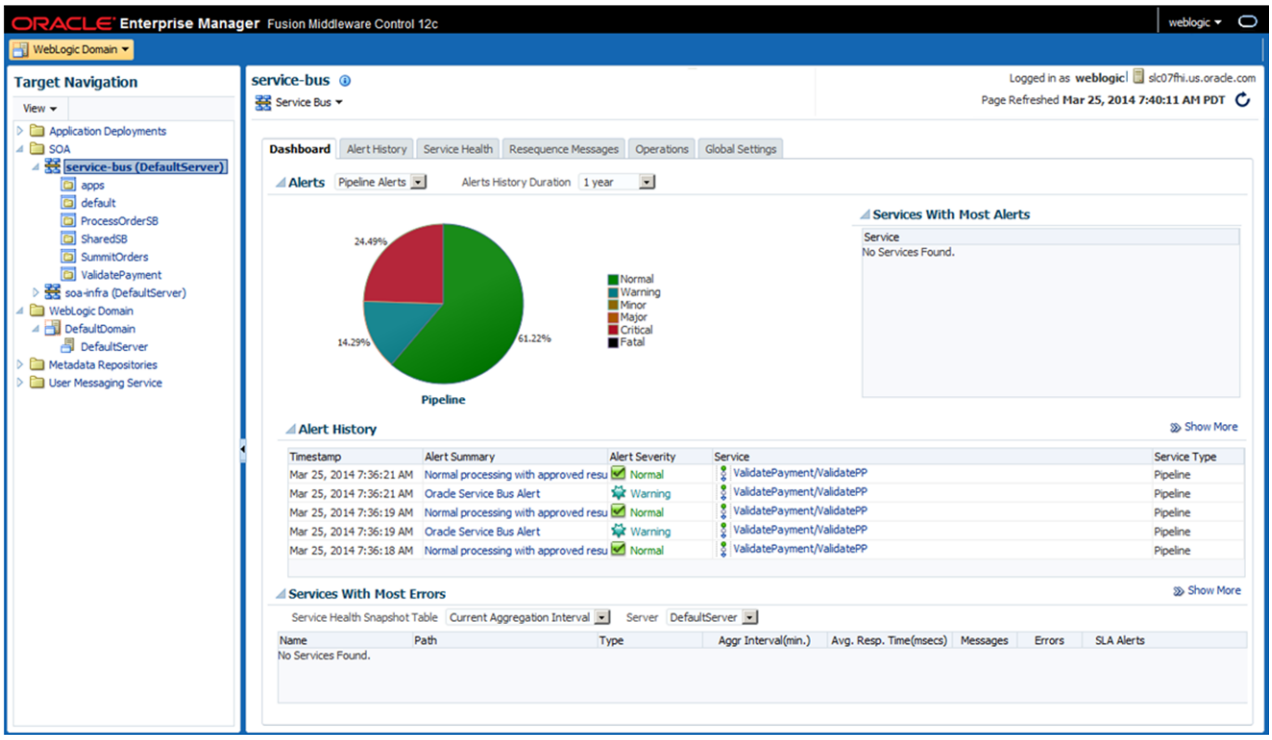
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The built-in Test Console is a browser-based test environment used to validate resources as well as inline expressions used in pipelines or split-joins. Use the Test Console to configure the test object (such as a pipeline, business service, or XQuery expression), execute the test, and view test results. It allows message flow tracing when testing a service, to examine the state of the message at specific trace points.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Monitoring and Administering in EM Console



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

In Enterprise Manager Fusion Middleware Control, you can monitor and manage Service Bus applications and their lifecycles. You deploy Service Bus applications designed in Oracle JDeveloper or the Oracle Service Bus Console to a Service Bus domain. Fusion Middleware Control accesses the information collected for those deployed applications.

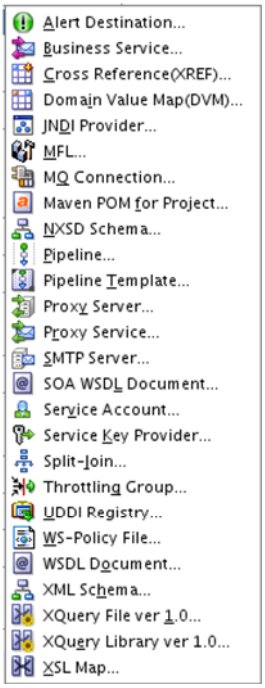
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

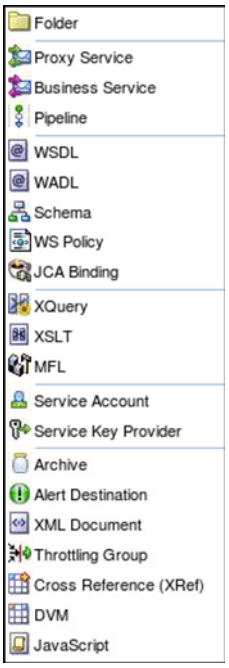
Development Tasks Support

Topic	JDeveloper	Console
Import & Export resources & projects	Y	Y
Modeling messages	Y	Y
Adapters	Y	N
Pipeline Template	Y	N
Split-Join	Y	N
Expression Builder	Y	Y
XQuery/XSLT Mapper	Y	N
Debugging	Y	N
REST enablement	Y	N

JDeveloper



Console



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Quiz

Which of the following web-based consoles allow you to create and modify Service Bus resources?

a. EM console
b. Test console
c. WebLogic Server Admin console
d. Service Bus console

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Answer: d

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Summary

In this lesson, you should have learned how to:

- Describe the basics of WSDL and XSD
- Create a simple Service Bus project in JDeveloper and Service Bus Console
- Describe the differences between JDeveloper and Service Bus Console in terms of development task support
- Populate MDS repository with source data

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Practice 3: Overview

- 3-1: Deploying and Testing a SOA Composite
- 3-2: Virtualizing Service with Service Bus
- 3-3: Populating MDS Repository with Source Data

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

4

Basics of Message Flow

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Objectives

After completing this lesson, you should be able to:

- Identify message exchange patterns
- Describe the message flow process
- Explain the functionality of Pipeline and Split-join
- Describe the usage patterns of different variables in a message flow
- Create a pipeline template and use it to create a pipeline
- Debug Service Bus pipelines

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Message flow
- Context variables
- Pipeline template
- Debugging

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

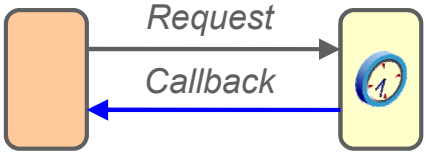
Message Exchange Pattern

Synchronous request/response

- Real-time response or error feedback
- Client in waiting mode

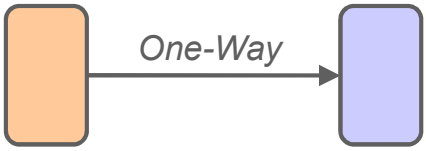
Asynchronous request/callback

- Client free after request submission
- Separate service invocation for response



Publish

- Also known as “fire and forget”
- Client free after request submission
- No response message (ACK only)



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Service Bus accommodates multiple messaging paradigms and supports the types of communication shown in the slide.

In a synchronous interaction, the client requests a service, and then waits for a response to the request. While the client waits, the communication channel between the parties is left open until the response occurs. This may be undesirable if large numbers of channels are left open for long periods of time. It may not be needed if the client does not need an immediate response. In these cases, an asynchronous response may be more appropriate.

In an asynchronous interaction, the client invokes the service but does not wait for a response before continuing. Asynchronous operations open a communication channel between the parties, make the request, and close the channel before the response occurs. The response may come at a later time via a callback operation, or not at all for one-way interactions.

Oracle University and ORACLE CORPORATION use only

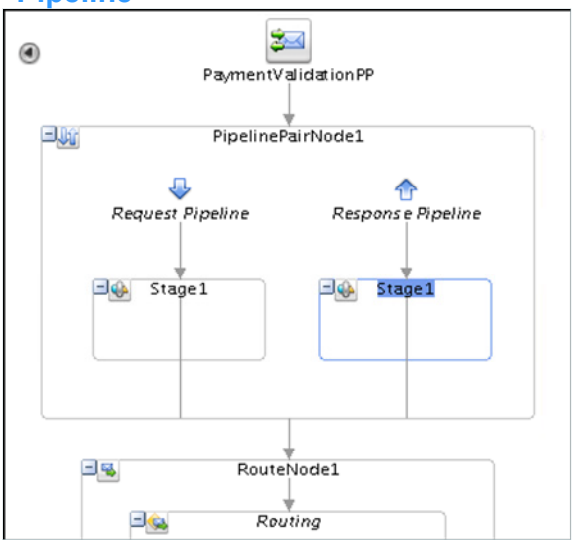
THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Message Flow

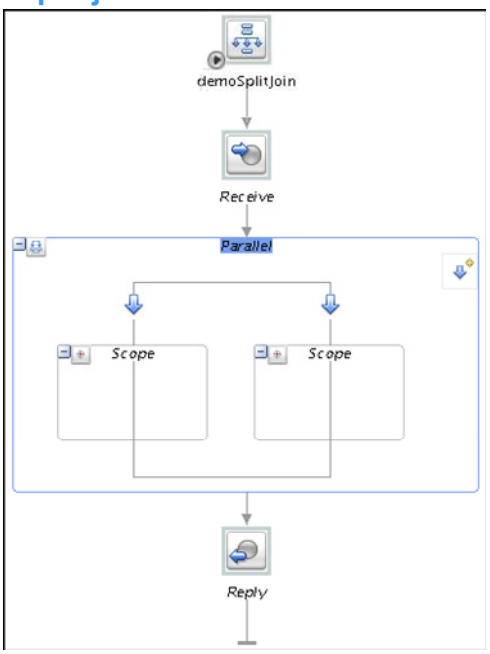
Message flow can be defined in a:

- Pipeline
- Split-join for parallel processing

Pipeline



Split-join



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

A message flow defines how messages are routed, validated, and transformed between services. The message flow is typically defined in a pipeline, but can also be defined in a split-join for parallel processing.

Split-join is a mediation pattern that can be used by a transport-typed business service in an Oracle Service Bus message flow.

Split-join allows you to send message requests to multiple services concurrently.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Pipeline

- Provides message processing capability
- Separated from Proxy Service, thereby allowing for better structured development
- A named sequence of stages containing *actions*
- Has its own unique message context and variables

Definitions:

- *Stages*: A group of actions
- *Actions*: Instructions, created graphically, with the occasional mix of XPath, XQuery, and/or XSLT

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

In 12*c*, message flow is no longer associated with proxy services. This allows for much better structured development with smaller chunks of functionality inside a single pipeline. A pipeline is a named sequence of stages containing actions, representing a nonbranching one-way processing path. Stage is a user-configured processing step. It is used to specify the message flow for service requests and responses. Pipelines are accompanied by a set of message context variables that contain the message contents. They can be accessed or modified by actions in the pipeline stages.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Pipeline Components

Primary elements for pipeline are:

- **Start node:** Mark the entry point of the message flow
- **Pipeline pair node:** Node for request and response pair processing
- **Stage node:** Container of actions to manipulate messages passing through the pipeline
- **Branch node:** Route the message based on condition either on type of operation or values
- **Route node:** Set the destination of message otherwise by default echo the request
- **Error Handler:** Handle errors either on stage or node

Note: Minimum start node and route node are required for a pipeline.

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Message flows may contain six main types of nodes: Start, Pipeline Pair, Stage, Branch, Route, and Error Handler. Each type of node is used for a specific purpose as described in the slide. There are rules concerning each node type, specifically about the number of nodes that might follow:

- A pipeline pair node can have only one direct descendant in the pipeline.
- Stage nodes are contained in request pipelines, response pipelines, and error handlers.
- A route node represents the boundary between request and response processing for the pipeline, so it cannot have any descendants in the pipeline.
- An error handler can be attached to any node or stage.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Message Processing Actions

	Assign	Assign the result of an XQuery expression to a context variable
	Delete	Delete a context variable or set of nodes specified by XPath expression
	Insert	Insert the result of an XQuery expression at an identified place relative to nodes selected by an XPath expression
	Rename	Rename an element selected by an XPath expression without modifying the contents of the element
	Replace	Replace a node or a node's contents specified by an XPath expression
	Validate	Validate elements selected by an XPath expression against an XML schema element or a WSDL resource
	Java Callout	Invoke a Java method from within the pipeline
	MFL Transform	Convert non-XML to XML or XML to non-XML within the pipeline
	nXSD Translate	Convert message content from XML to native format data, or vice versa

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Communication Actions

	Routing	Route a message to a statically specified service
	Dynamic Routing	Route to a service based on information in an XQuery resource
	Routing Table	Route to one of a set of services wrapped in a switch-style condition
	Publish	Publish a message to a statically specified service
	Dynamic Publish	Publish to a service based on information in an XQuery resource
	Publish Table	Publish to one of a set of services wrapped in a switch-style condition
	Service Callout	Configure a synchronous (blocking) callout to proxy or business service
	Routing Options	Override outbound request properties (such as URI or Retry Count)
	Transport Headers	Set the transport header values for outbound messages

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Service Bus provides Routing, Service Callout, and Publish activities to route incoming requests to an appropriate business service. Routing and Service Callout are used for synchronous, whereas Publish activity is used for one-way communication. You can select the required option as per requirement.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Flow Control and Report

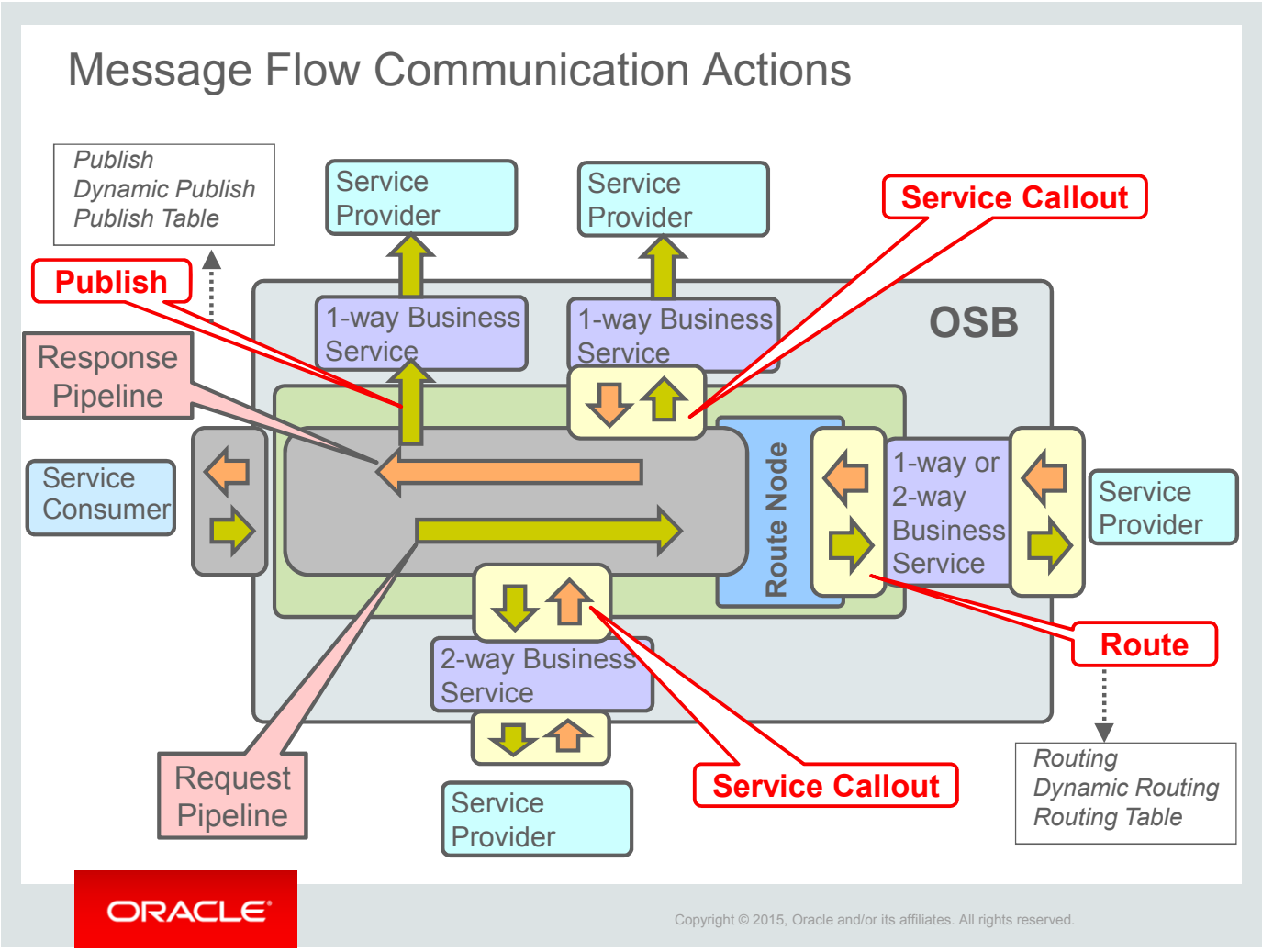
Flow Control Actions		
	For…Each	Iterate over a sequence of values and execute a block of actions
	If…Then	Perform an action or set of actions conditionally, based on the Boolean result of an XQuery expression
	Skip	Specify that at runtime, the execution of the current stage is skipped and the processing proceeds to the next stage in the message flow
	Reply	Force an immediate reply to be sent to the invoker of this proxy service
	Raise Error	Raise an exception with a specified error code (a string) and description
	Resume	Resume message flow after an error is handled by an error handler
Report Actions		
	Alert	Send an alert notification based on pipeline message context
	Log	Construct a message to be logged to the WebLogic server log file
	Report	Define data with an index and key value to be sent to the OSB Database

Only used in Error Handler stage

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED



The figure in the slide shows a collection of communication actions and possible ways of interacting with other services. Communication actions are focused on sending messages to other services. Those other services may be business services, proxy services, or pipeline services within the service bus.

A proxy is the inbound portion of Service Bus that can handle the incoming request and manipulate the payload before invoking co-located or remote services.

The execution logic is built using the pipeline actions. For executing the remote or local business service, Service Bus provides three forms of service invocations within a pipeline:

- **Route:** Invoke a single business service endpoint with (or without) a response. This happens entirely at end of a pipeline service execution and bridges the request and response pipeline. The route can be treated as the logical destination to reach for final service invocation. There can be only one Route action (There are multiple choices of Route actions, but only one actual execution) in a given pipeline execution.
- **Service Callout:** Invoke one or more business service(s) as part of message augmentation, enrichment, or validation but this is not the primary business service for a given proxy, unlike the Route action. The Service Callouts can be equivalent to credit card validation address verification, while Route is equivalent to final order placement. There can be multiple Service Callouts inside a pipeline.

Oracle University and ORACLE CORPORATION use only

- **Publish:** Invoke a business service without waiting for result or response. The caller does not care much about the response. It is just interested in sending out something (and ensuring it reaches the other side).

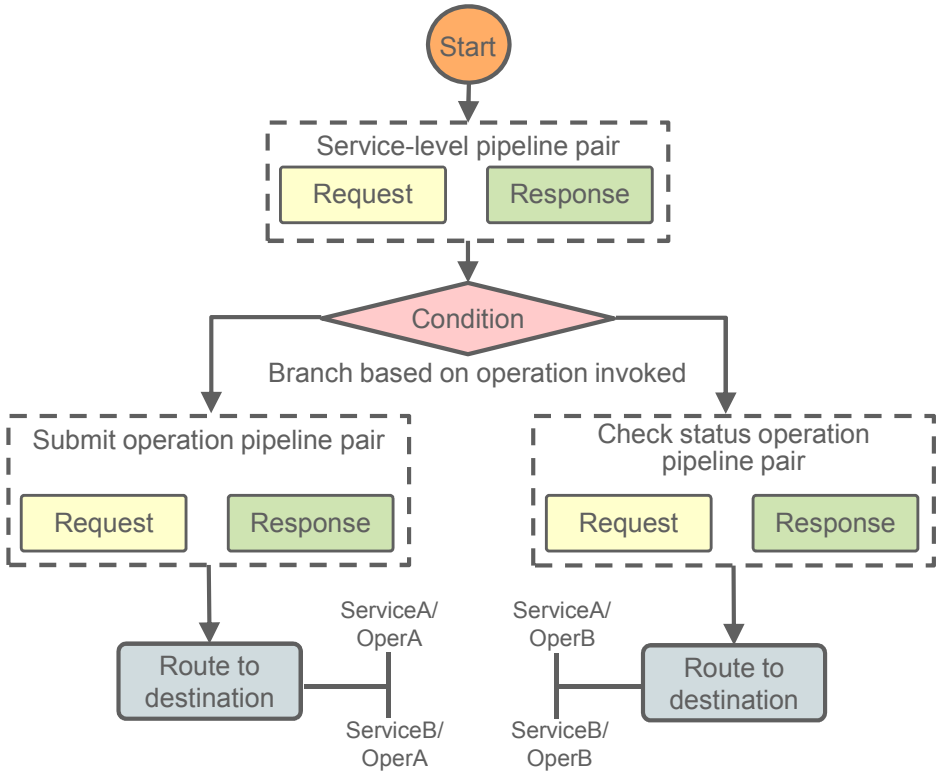
Note: There are always request and response pipelines regardless of whether the proxy/business service is using one-way or two-way communication.

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Pipeline: Example



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

In a pipeline, the request message starts at the start node and follows a path to a leaf node, executing actions in the request pipelines. If the leaf is a route node, a response is generated. If the leaf is an echo node, the request is also considered to be the response. The response follows the inverse path in the tree, skipping actions in the branch nodes but executing actions in response pipelines. A response is then sent from the top of the tree if the interface or operation was request/response; otherwise the response is discarded.

In general, a message flow is likely to be designed in one of the following ways:

- For nonoperational services (services that are not based on WSDL files with operations), the flow consists of a single pipeline pair at the root followed by a route node.
- For operational services, the flow consists of a single pipeline pair at the root, followed by a branch node based on an operation, with each branch consisting of a pipeline pair followed by a route node. The diagram in the slide shows an example of operational pipeline.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Quiz

You can use more than one route node in a message flow.

a. True
b. False

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Answer: b

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Message flow
- Context variables
- Pipeline template
- Debugging

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Message Context

A set of properties, known as *context variables*:

- Hold the message content as well as information about messages
- Shared across the request flow and response flow
  - Within one pipeline
  - Across multiple pipelines
- Used to:
  - Reassemble
  - Modify
  - Reroute
  - Make runtime decisions about message processing

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The Service Bus message context is a set of properties that hold the message content as well as information about messages as they are routed through Oracle Service Bus. These message properties are referred to as context variables.

Context variables are the access points to request and response messages. New variables can be dynamically added or deleted to the context, and these variables can be shared across multiple pipelines or used locally within one pipeline.

The message context is defined by an XML schema. You typically use XQuery or XSLT expressions to manipulate the context variables in the message flow.

A set of transformations that affects context variables can be defined before the message is sent to the selected endpoint or after the response is received. A web services callout can be an alternative to an XQuery or XSLT transformation to set the context variables.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Predefined Context Variables

The predefined context variables can be grouped into the following types:

- Message-related variables: `$header`, `$body`, `$attachment`
- Inbound and outbound variables: `$inbound`, `$outbound`
- Operation variable: `$operation`
- Fault variable: `$fault`

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Predefined context variables contain information about the message, transport headers, security principles, metadata for the current pipeline, and metadata for the primary routing and publishing services invoked by the pipeline.

For example, service endpoints are represented by predefined context variables. Oracle Service Bus also supports user-defined context variables.

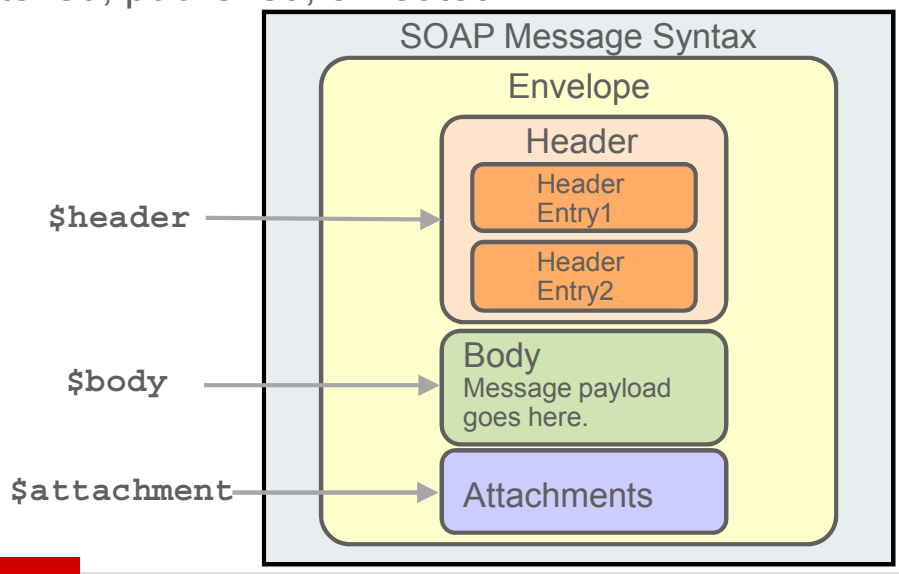
The message context schema specifies the element types for the message context variables.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Message-related Variables

- A message payload is contained in the `body` variable.
- The decision about which variable's content to include in an outgoing message is made at the point at which a message is dispatched, published, or routed.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The message-related `header`, `body`, and `attachments` variables represent the canonical format of a message as it flows through Oracle Service Bus. If you want to modify a message while processing it, you must modify these variables.

These variables are initialized by using the message content received by a pipeline and are used to construct the outgoing messages that are routed or published to other services.

The decision about which variable's content to include in an outgoing message is dependent on whether the target endpoint is expecting a SOAP or a non-SOAP message.

- If a SOAP message is required, the `header` and `body` variables are combined in a SOAP envelope to create the message.
- If a non-SOAP message is required, the content of the body element in the `body` variable constitutes the entire message.
- In either case, if the service expects attachments, a MIME package is created from the resulting message and the `attachments` variable.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

`$inbound` and `$outbound` Variables

- `$inbound` variable contains:
  - Information about the proxy service that received a message
  - Inbound transport headers
- `$outbound` variable contains:
  - Information about the target service to which a message is sent
  - Outbound transport headers

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The `inbound` and `outbound` context variables contain information about the inbound and outbound endpoints, respectively. The `inbound` variable contains information about the proxy service that received the request message; the `outbound` variable contains information about the target business service to which a message is sent.

The `outbound` variable is set in the route action in route nodes and publish actions. You can modify `$outbound` by configuring request and response actions in route nodes and by configuring request actions in publish actions.

The subelements of the `inbound` and `outbound` context variables include:

- `service`
- `transport`
- `security`

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

`$operation` Variable

- The `operation` variable is a read-only variable.
- It contains a string that identifies the operation to be invoked on a proxy service.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle Service Bus provides the `operation` variable as a stand-alone variable, rather than as a subelement of the `inbound` variable to optimize performance.

If no operations are defined for a proxy service, the `operation` variable is not set and returns the equivalent of null.

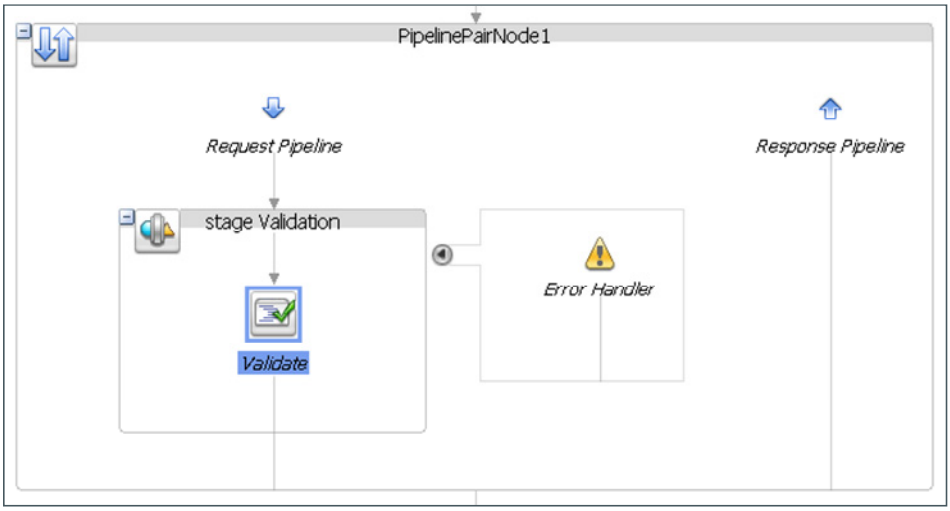
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

`$fault` Variable

`$fault` variable is:

- Populated with error information before the appropriate error handler is invoked
- Defined only in error handler pipelines



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The fault variable holds information about any error that has occurred during message processing. When an error occurs, this variable is populated with information before the appropriate error handler is invoked. This variable is defined only in error handler pipelines and is not set in request and response pipelines or in route or branch nodes.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Variables and Types

- At design time:
  - All variables (user and system defined) are scoped for the entire service
  - `$outbound`, `$inbound`, `$fault`, and `$operation` are typed. Others are not because their contents vary.
- At runtime:
  - Variables are automatically created when first assigned a value during execution
  - Variables are deleted when the service execution terminates

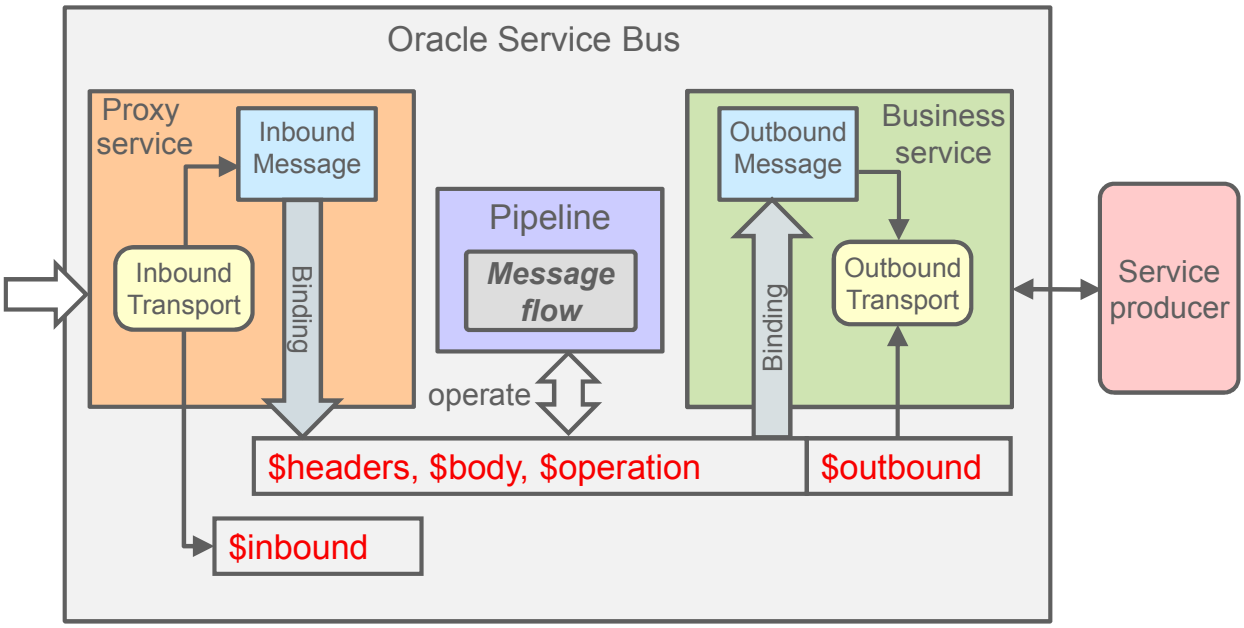
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

During design time, user-defined variables are not visible outside the pipeline in which they are created.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Context Variables in Message Processing



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle Service Bus message processing has several unique lifecycle phases:

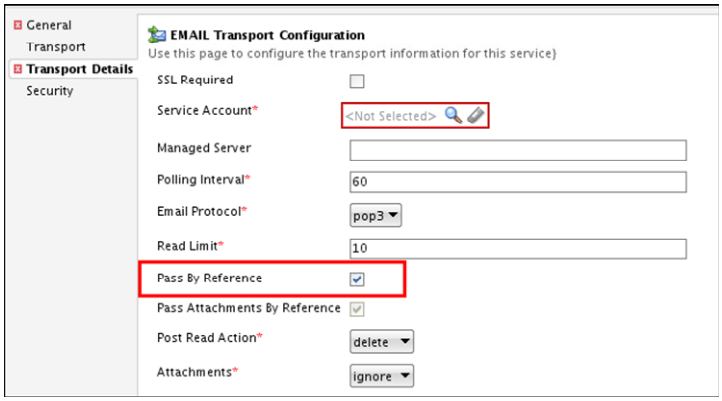
- **Inbound transport phase:** Receives the message
- **Binding phase:** Splits message content into predefined context variables: `$inbound` (read-only), `$operation` (read-only), `$body` – main message content; `$header` – SOAP headers for SOAP messages only; `$outbound` – set in anticipation of the Business service delivery transport
- **Message Flow:** Manipulate message data by modifying the context variables
- **Binding phase:** Message is reassembled from context variables and readied for delivery to the business service.
- **Outbound transport phase:** Message is shipped to the Business service endpoint, the behavior may depend on the `$outbound` context variable content.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Pass Content by Reference

- Binary content (non-XML, non-text) (*always*)
  - Stored in an in-memory hash table
  - Referenced in the `$body` and `$attachments` variables: `<binary-content ref=…>`
- File, email, and FTP (*optional*)
  - Passed as a reference in the message headers



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When binary content is received, the Oracle Service Bus run time stores it in an in-memory hash table and a reference to that content is inserted into the XML (body or attachments) element. Binary content is *never* left in `$body` (or `$attachments`).

- The benefit is processing efficiency.
- The tradeoff is that binary content cannot be modified.

You can configure proxy services based on file, email, and FTP to pass content by reference.

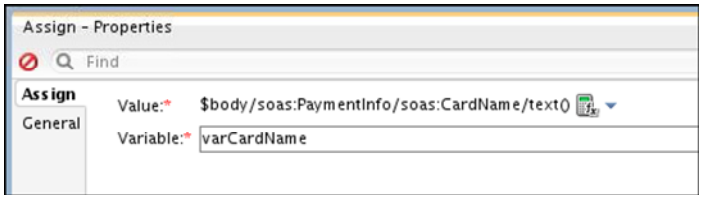
Pass By Reference: It tells copy the file to archive directory and pass a reference to the file in the message itself.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

User-Defined Variables

- Using the Assign action is one way of defining a variable in a pipeline.
- The scope of the variable is for the request pipeline, response pipeline, route node, and fault pipeline.



Note: Service Bus is stateless and therefore variables do not persist.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

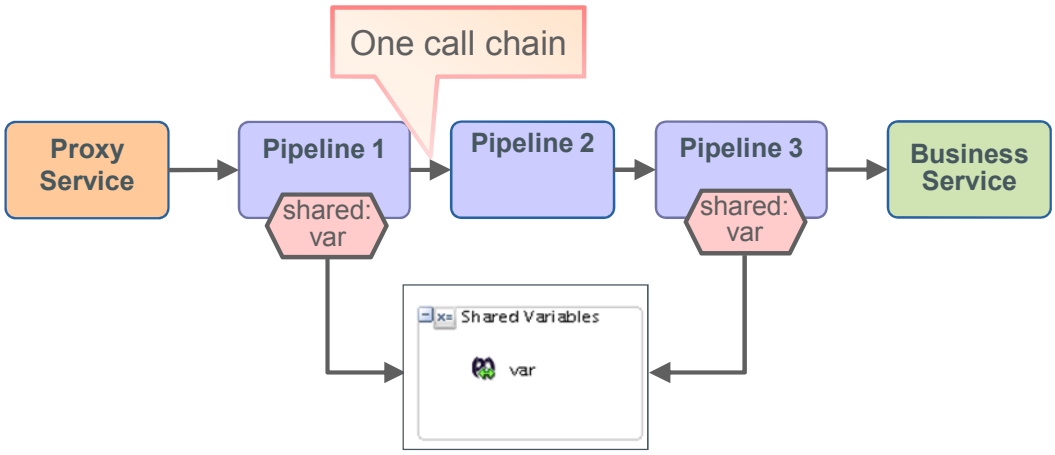
Service Bus supports user-defined variables. You use an Assign action in the pipeline to define a variable. In the properties of Assign action, give a name to the variable, and use XQuery or XSLT expression to define the value.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Shared Variables

When two pipelines in the same call chain declare a variable with the same name as shared, they will be reading and modifying the same variable.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

If two pipelines in a single call chain declare the same shared variable, then they read and modify the same variable in the scope of the invocation call chain. In other words, if pipeline P1 declares a shared variable `var`, and pipeline P1 invokes pipeline P2, which also declares a shared variable `var`, then any changes to var in P1 are visible in P2, and vice versa. A shared variable must be of the String, Boolean, or XML data type.

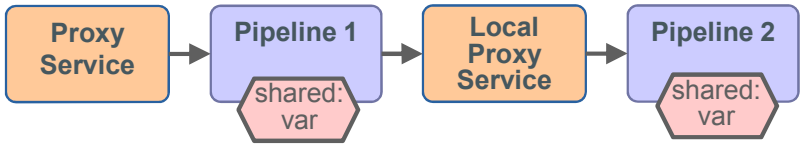
When a pipeline receives and processes a message, all invoked pipelines that use a shared variable, read and write the same value for the variable. A subsequent message received by the proxy creates a new instance of the shared variable in the invoked pipelines.

Oracle University and ORACLE CORPORATION use only

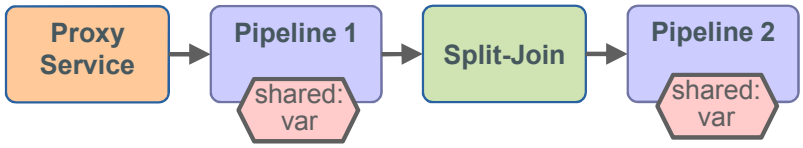
THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Shared Variable Rules

- The scope of sharing is confined to a single message being processed by pipelines within one call chain.
- Work across:
  - Local proxy invocations



  - Split-join component invocations



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Shared variables work across local proxy invocations and split-join component invocations. For example, say pipelines P1 and P2 declare a shared variable. Now, if P1 invokes a local proxy service or split-join component, which in turn invokes P2, then P1 and P2 continue to share the shared variable.

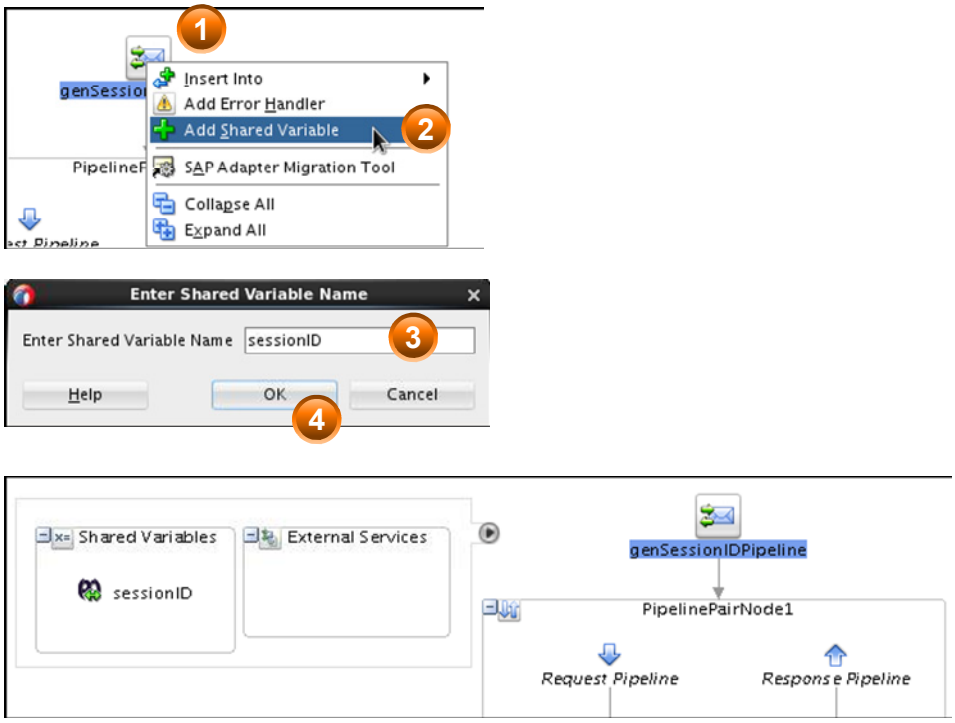
The following restrictions apply to using shared variables:

- System variables (such as `$body`, `$attachments`, `$operation`, `$inbound`, `$outbound`) cannot be shared.
- Variables cannot be shared across nonlocal proxy invocations. For example, say a pipeline invokes an HTTP proxy service, the shared variable is not propagated across this call.
- Variables cannot be shared between pipeline and split-join resources.
- Variables with Java and binary content types are not supported. For example, an XML-typed variable that has `<ctx:java-content/>` in its XML structure is not supported as a shared variable.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Creating a Shared Variable



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

To add a shared variable to a pipeline:

1. In the Pipeline Editor, right-click the start node.
2. Select Add Shared Variable from the context menu that appears.
3. Enter a name for the shared variable in the Enter Shared Variable Name field.
4. Click OK to add the shared variable to the pipeline.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Quiz

Which one of the following is not a predefined context variable?

a. `Header`
b. `body`
c. `outbound`
d. `fault`
e. `transport`

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Answer: e

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Message flow
- Context variables
- Pipeline template
- Debugging

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

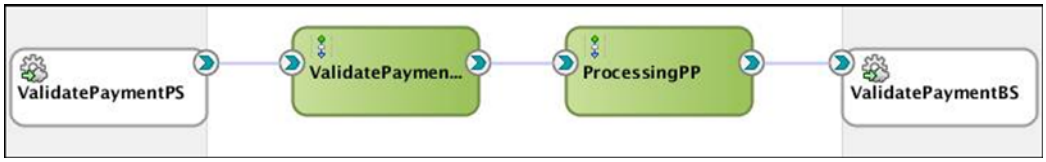
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

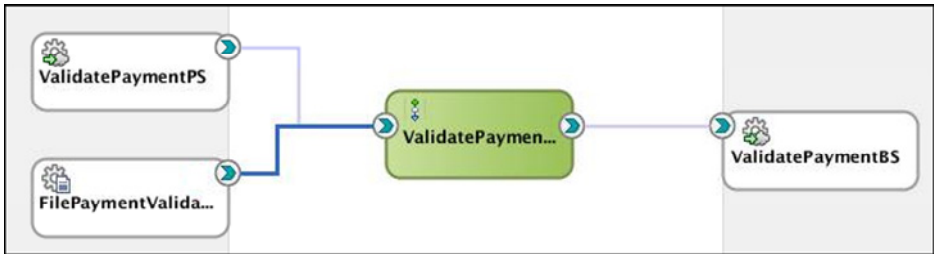
Pipeline Construction

Pipelines can be:

- Chained



- Reused



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

For all intents and purposes: Developing Service Bus projects by using the concept of pipelines is very similar to the development of SOA Composite applications.

Pipelines can be chained: A pipeline can invoke a business service but can also invoke another pipeline.

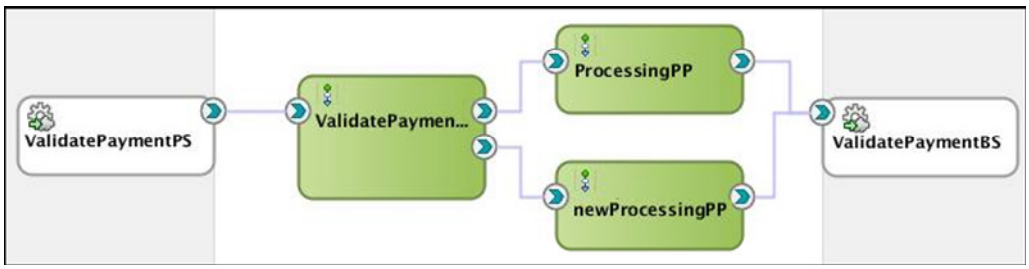
Pipelines can be reused: Multiple proxy services (and/or multiple pipelines) can invoke the same pipeline—directly or indirectly. For example, a SOAP-based web service proxy and a REST JSON service proxy can leverage the same pipeline.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Pipeline Construction

- One pipeline can invoke another pipeline by using a callout or publish action.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The example shows that a conditional branch can conditionally call out to one pipeline, selected from several pipelines (or business services), as can a split-join component.

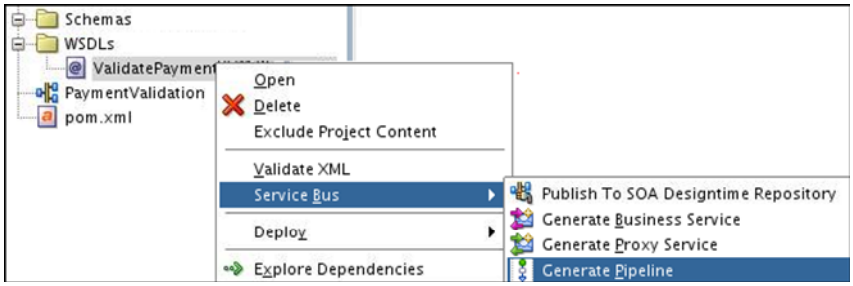
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Creating a Pipeline

A Pipeline can be created by using:

- WSDL service definition document



- Pipeline template
- Dragging the Pipeline component from the component palette
- Generating from the proxy service

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The slide lists the ways of creating a pipeline.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Pipeline Templates

- Pipeline templates can be used to design prototype flows.
- Typical components included in a template are:
  - Validation
  - Error Handler
  - Logging
  - Alerts
  - Reports

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When building a lot of services through different projects, you will always duplicate certain functionality such as validation, logging, error handling, and so on. They all conform your current development rules and guidelines of the project.

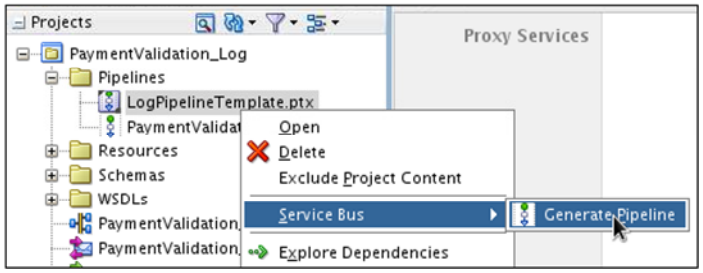
Pipeline template is a capability in Oracle Service Bus that allows developers to create a prototype of a pipeline with all the components that can be used across the enterprise services.

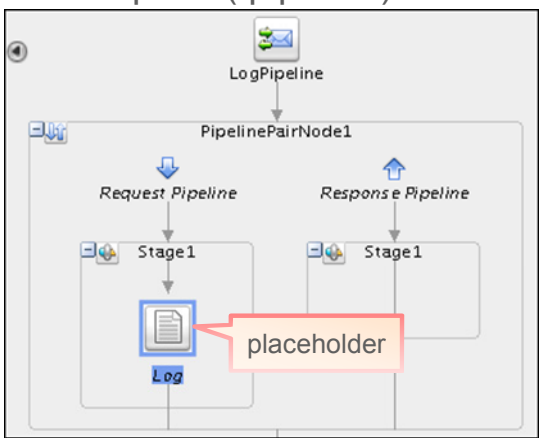
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Creating and Using Pipeline Template

- Create template:
  - Create a pipeline template (.ptx).
  - Export the template file as a resource (.jar).
- Use template:
  - Import the template file.
  - Create a concrete pipeline using the template (.pipeline).
  - Fill the placeholders.





Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

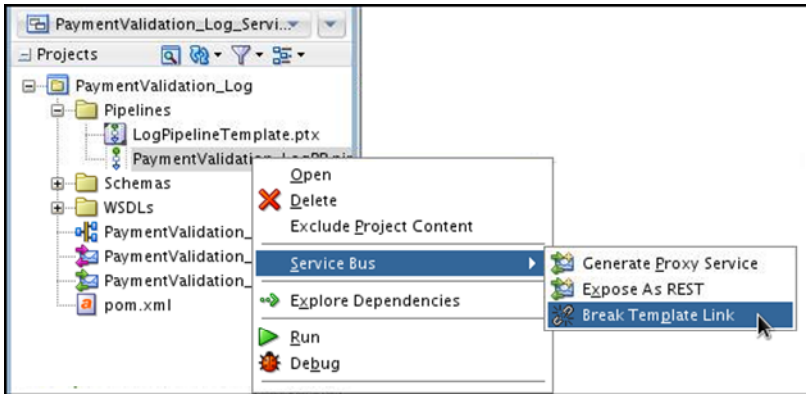
The concrete pipeline inherits the message flow from pipeline template. Placeholders is where modifications may be placed in the pipeline template.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Breaking a Template Link

- A concrete pipeline is automatically associated with the pipeline template.
- To break the link:
  - Use the Configuration tab of the Pipeline Editor.
  - Right-click the pipeline in Application Navigator, select Service Bus > Break Template Link.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

If you no longer need a concrete pipeline to be associated with a pipeline template, you can break the template link to convert the pipeline into a regular pipeline. The regular pipeline can be edited without the restrictions applicable to concrete pipelines.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Message flow
- Context variables
- Pipeline template
- Debugging

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Debugging Service Bus Applications

- Enable/disable stages and actions (JDeveloper)
- Flow tracing (EM)
- Debugger (JDeveloper)

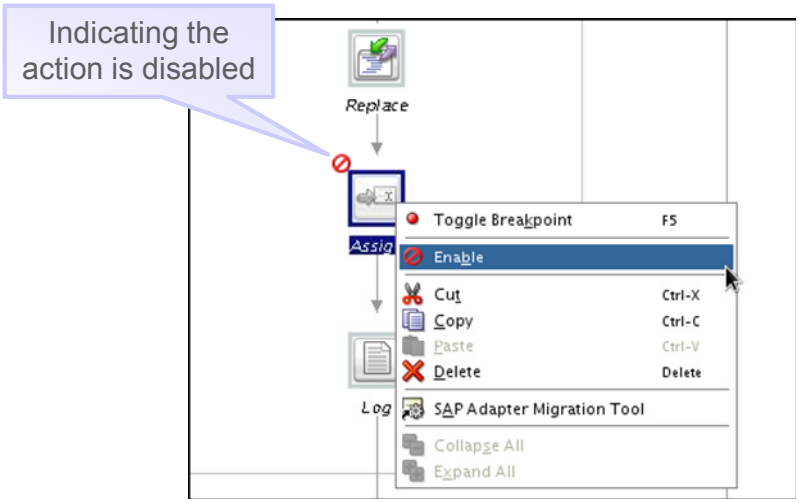
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Enable/Disable Stages and Actions

- Allow the user to “skip” a single activity or a stage at runtime without having to delete the action/stage from the flow.
- An action or stage that was previously disabled may be re-enabled.



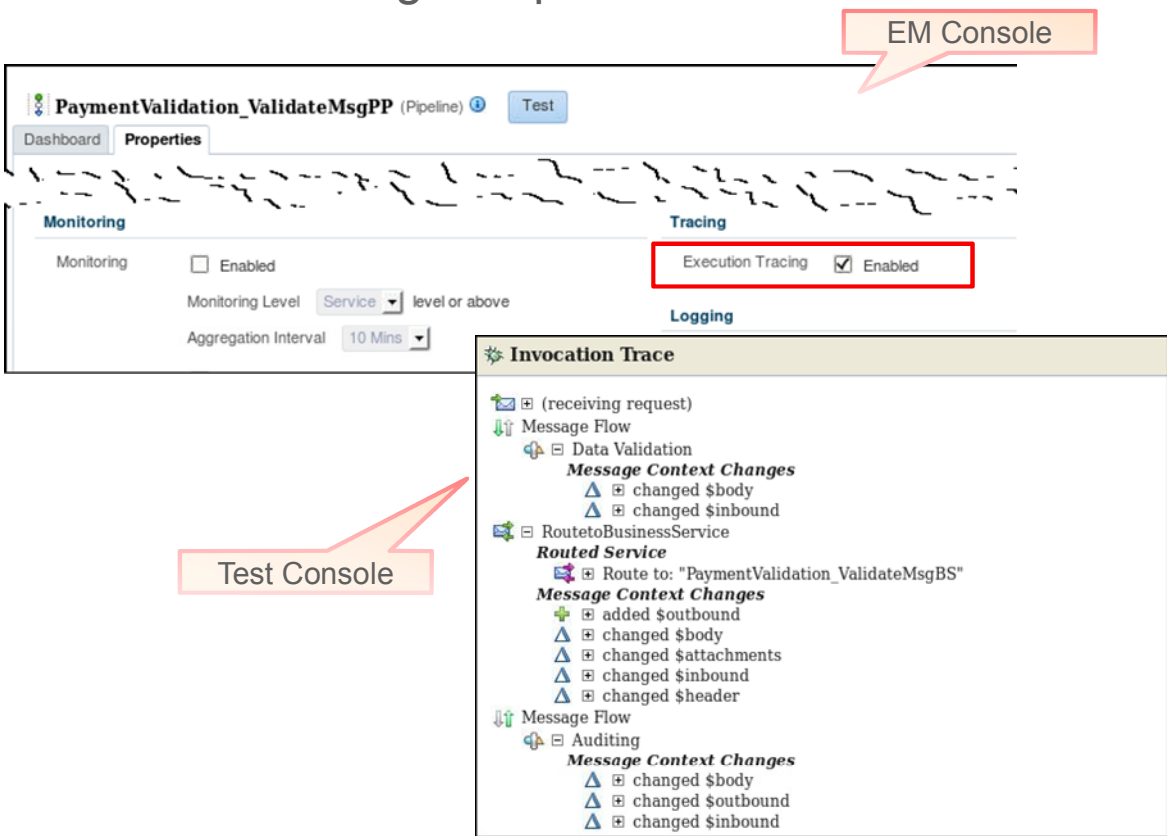
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Service Bus 12*c* has a new feature that allows you to enable and disable any activity in the process flow. That means when you disable the activity, you can actually bypass it in the real business flow. Enable/disable stages and actions is similar to commenting out code. This is useful when you do not want to remove any activity but just skip them for some particular set of testing. This feature is available both in design time as well as in run time using the service bus composer.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Enable Flow Tracing in Pipeline



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When testing pipelines, the input messages are sent directly to the pipeline. Turning on tracing allows you to diagnose and troubleshoot a message flow in the Test Console. Testing a pipeline tests the internal message flow logic. Use this test approach in conjunction with setting custom (inbound) transport headers in the Test Console Transport panel to accurately simulate the service call.

Tracing a message through a pipeline involves examining the message context and outbound communications at various points in the message flow. The points at which the messages are examined are predefined by Service Bus, which defines tracing for stages, error handlers, and route nodes. For each stage, the trace includes the changes that occur to the message context and all the services invoked during the stage execution.

The following stage information is provided by the trace:

- **Initial Message Context:** Shows the variables initialized by the proxy service when it is invoked. To see the value of any variable, click the + sign associated with the variable name.
- **New variables:** The names of all new variables and their values. Expand a variable to view its value.
- **Deleted variables:** The names of all deleted variables.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

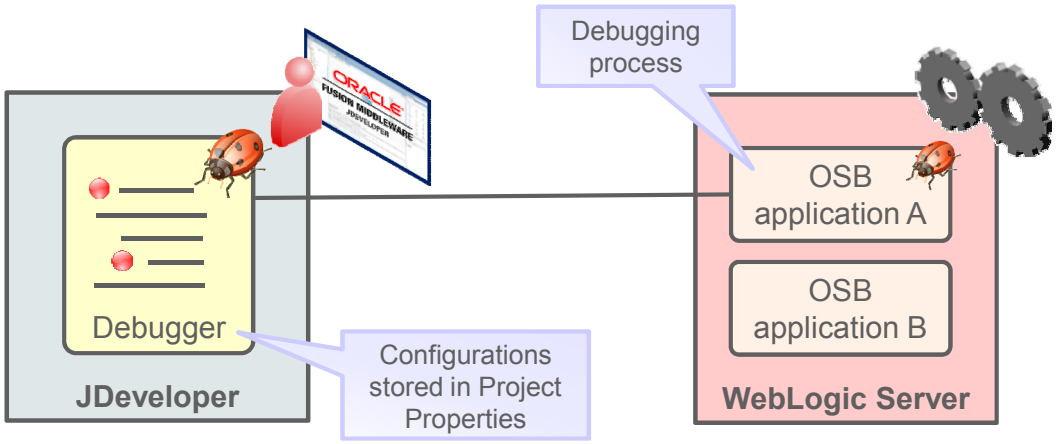
- **Changed variables:** The names of all variables for which the value changed, including `$header`, `$body`, and `$inbound`, changes as a result of the processing of the message through the stages. Expand a variable to view the new value.
- **Faults:** If an error occurs, the contents of the fault context variable (`$fault`) are shown as a result of the stage error handler handling the validation error.
- **Publish:** Every publish call is listed. For each publish call, the trace includes the name of the service invoked, and the value of the outbound, header, body, and attachment variables.
- **Service callout:** Every service callout is listed. For each service callout, the trace includes the name of the service that is invoked, the value of the outbound variable, the value of the header, body, and attachment variables for both the request and response messages.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Debugging Service Bus Applications with Debugger

Service Bus debugger provides a troubleshooting environment within JDeveloper.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

JDeveloper provides a comprehensive debugger to help you assess and repair your Service Bus project components. The debugger reduces the development cycle by providing a troubleshooting environment directly in the development environment. You can set breakpoints directly in JDeveloper for troubleshooting on pipelines and split-joins.

You are able to stop at breakpoints, look at the data, step into, step out, and so on.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Features and Limitations

- Supports:
  - Multi-threaded debugging on split-joins
  - Debugging on:
    - Either integrated or stand-alone WebLogic server
    - Either a local or remote WebLogic server
- Limitations
  - Limited to design view in JDeveloper
  - No debugging cross-language features, such as a Java callout action, XSLT and XQuery transformations
  - Only one client at a time can connect to the debugger.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The debugger can handle Java callouts and supports multi-threaded debugging on split-joins that use parallel processing. Note the following guidelines when using the debugger:

- Debugging is limited to design view in JDeveloper.
- You cannot debug cross-language features, such as a Java callout action, and XSLT and XQuery transformations, and so on.
- Only one client at a time can connect to the debugger.

Debugging can be performed on either a local or remote WebLogic server. A local debugging session is started by setting breakpoints in source files, and then starting the debugger.

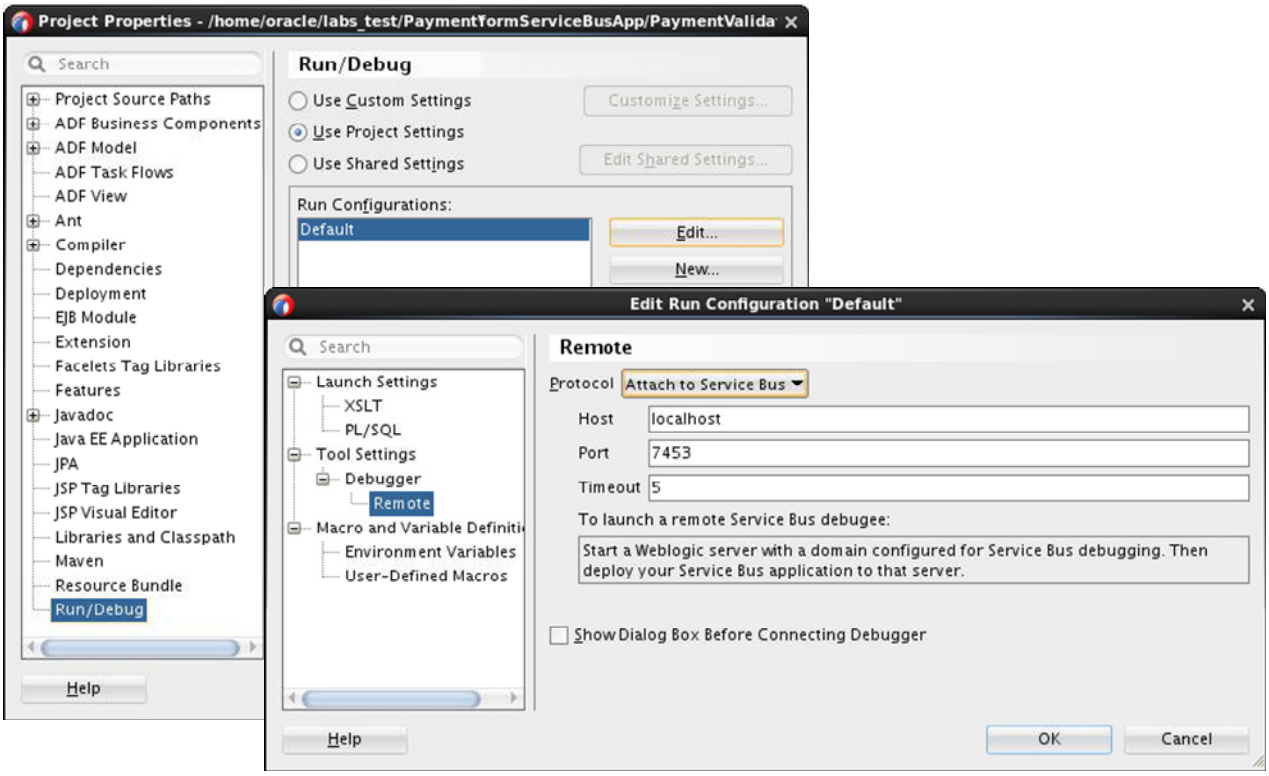
Remote debugging requires two JDeveloper processes: a debugger and a debugee. The debugee is a running server that may or may not be defined in JDeveloper and may reside on a different platform. A Service Bus application must be deployed on the debugee server. To perform remote debugging, you must configure a run configuration in the project properties.

When you use a stand-alone server for debugging, the configuration is the same whether it is local or remote.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Configuring Debugger



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

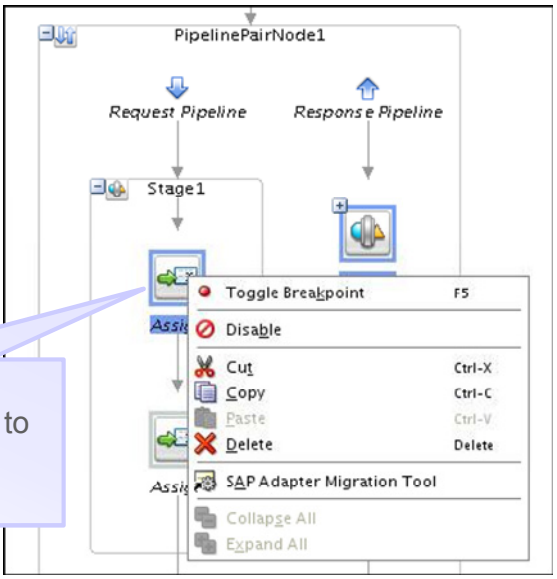
Both local and remote applications can be debugged. The debugger properties are stored per project. If you have imported a project from a different machine, the host will probably have to be changed. These values can be stored in JDeveloper properties, which enable you to skip the Connection Settings window if you always want to use the same options.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Setting Breakpoints

- Breakpoints can be added to the following nodes on a pipeline:
  - Route node
  - Route action
  - Branch node
  - Pipeline pair path (request or response)
  - Stage (pipeline pairs and error handlers)
  - Stage action
- Breakpoints can be added to all split-join actions.



Right-click the action/node/stage on which to set a breakpoint, and select Toggle Breakpoint.

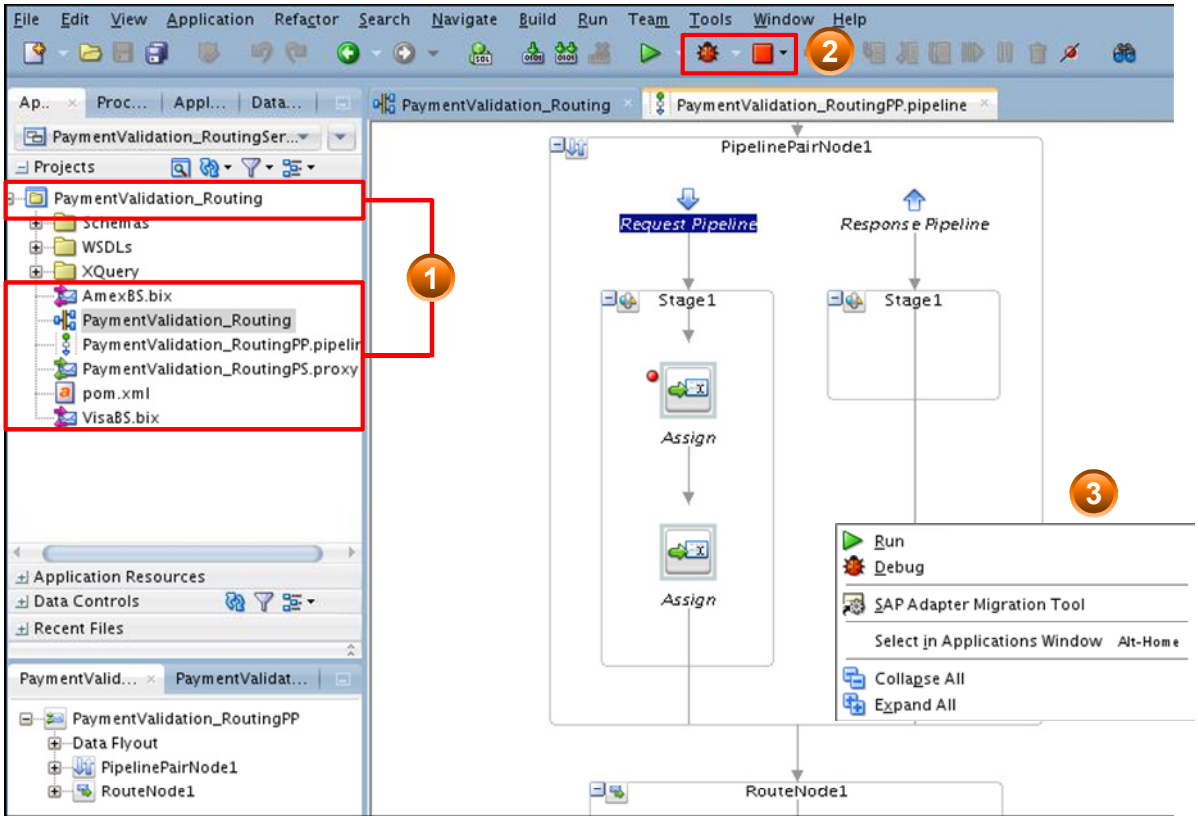
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

A breakpoint marks a point in a pipeline or split-join where message processing pauses. This lets you examine the values of some or all of the message variables. By setting breakpoints in potential problem areas of your message flow, you can run data through a message flow until it reaches a location you want to debug. When a breakpoint is encountered, message processing pauses and the debugger focuses on the action containing the breakpoint in the source editor. You can then use the debugger to view the state of your program. Breakpoints are flexible in that they can be set before you begin debugging or at any time while you are debugging.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Starting the Debugger



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

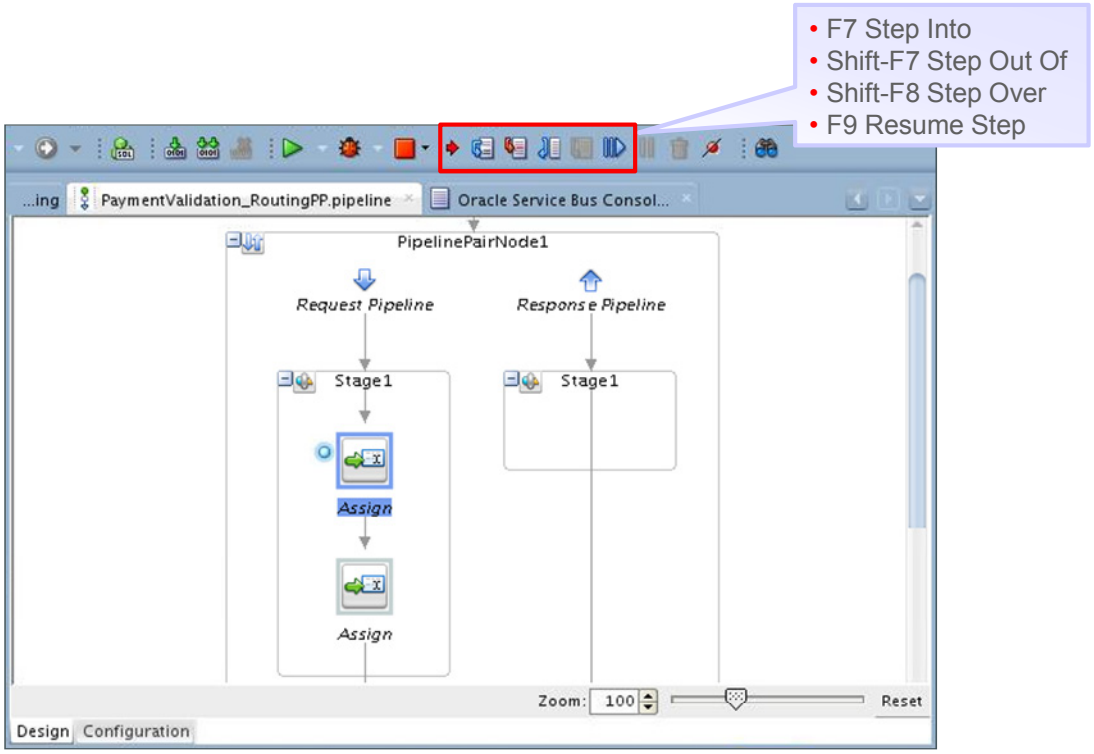
There are several ways to start the JDeveloper debugger. You can use any of the following methods to start the debugger when you are ready:

- Right-click a project or component in the application and select Debug (1).
- In the JDeveloper toolbar, click the Debug icon (2).
- Right-click a pipeline or split-join in the Service Bus Composite Overview Editor and select Debug.
- Right-click in the editor of an open pipeline or split-join and select Debug (3).

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Stepping Through a Debugging Session



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

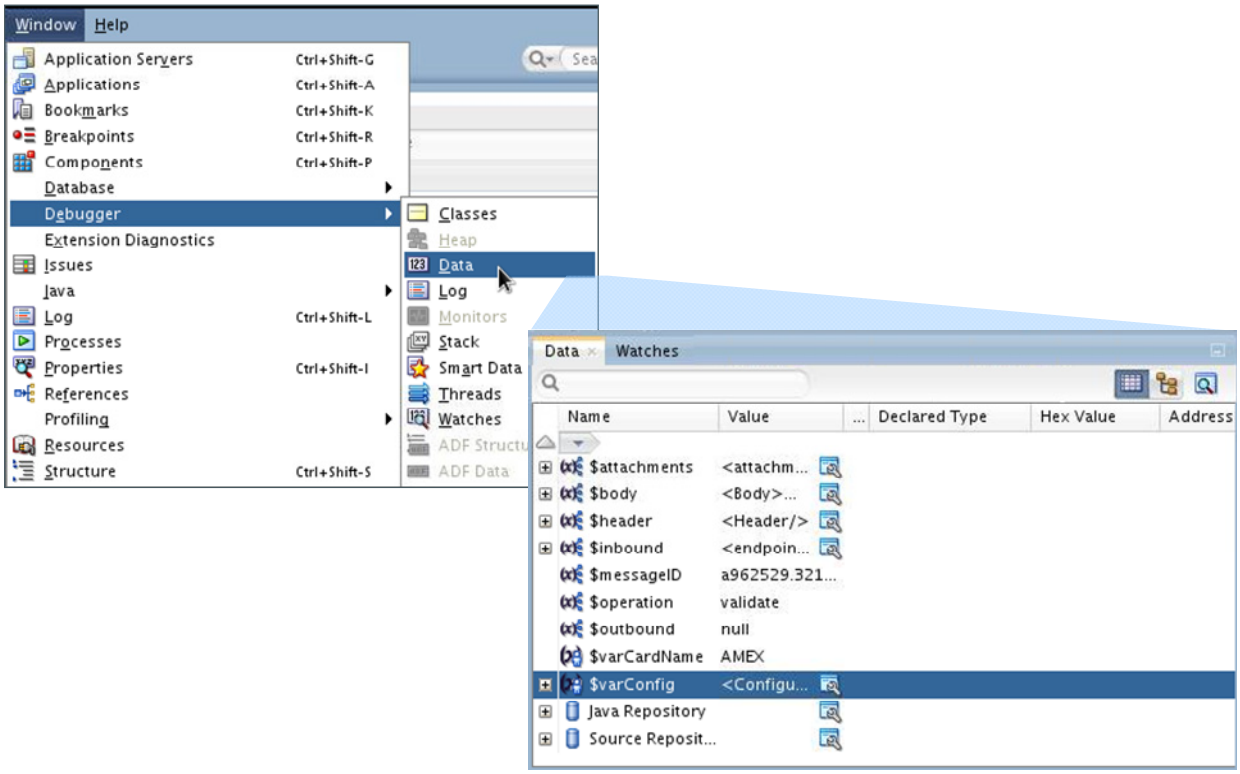
To step through an application, use the following function keys or the corresponding toolbar icons:

- To find the action on which the current breakpoint is stopped, click the Find Execution Point icon.
  The current breakpoint icon is blue instead of red.
- **F8 steps over a frame**: To step over a breakpoint and move to the next action.
- **F7 steps into the next valid location**: To step into the next valid location in the message flow.
- **Shift-F7 steps out of a frame**.
- **F9 resumes a step operation**: If the debugger is paused on a breakpoint, click the Resume icon to resume.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Viewing and Modifying Variable Values



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

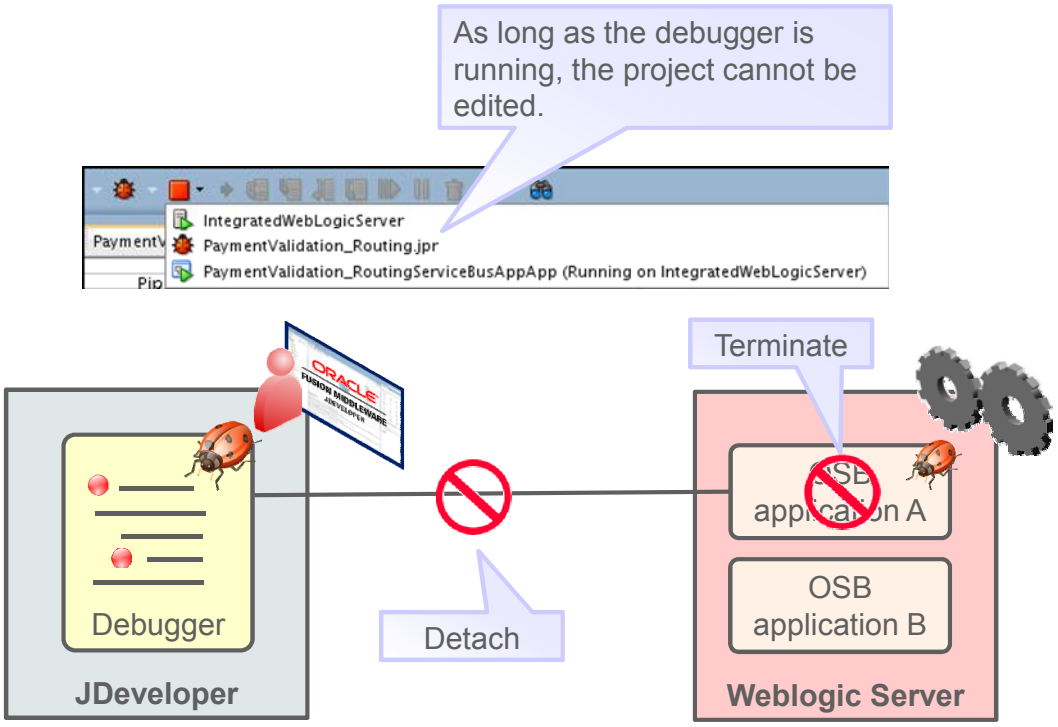
The Data window displays the context variables and their values for the current breakpoint. You can view request or response data throughout the debugging process, and modify their values for further debugging. Only values of simple-type variables can be modified.

A watch lets you monitor the changing values of variables or expressions as your program runs. After you enter a watch expression, the Watches window displays the current value of the expression. As your program runs, the value of the watch changes as your program updates the values of the variables in the watch expression.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Ending a Debugging Session



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

To end or detach from a debugging session, perform the following:

Click the Terminate icon in the tools menu to end a debugging session. The Terminate Debugger Process dialog box is displayed. Select an option:

- **Detach** removes the debugger without ending the debugging process.
- **Terminate** ends the debugging process.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Summary

In this lesson, you should have learned how to:

- Identify message exchange patterns
- Describe the message flow process
- Explain the functionality of Pipeline and Split-join
- Describe the usage patterns of different variables in a message flow
- Create a pipeline template and use it to create a pipeline
- Debug Service Bus pipelines

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Practice 4: Overview

- 4-1: Creating a pipeline template for logging
- 4-2: Creating a new pipeline using a pipeline template
- 4-3: Debugging a Service Bus Application in JDeveloper

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Option: You can include a brief summary of the practice here. Do not merely repeat the information in the slide.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

5

Validating Messages and Error Handling

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Objectives

After completing this lesson, you should be able to:

- Validate a message
- Describe basic fault handling and propagation
- Handle errors in Service Bus

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

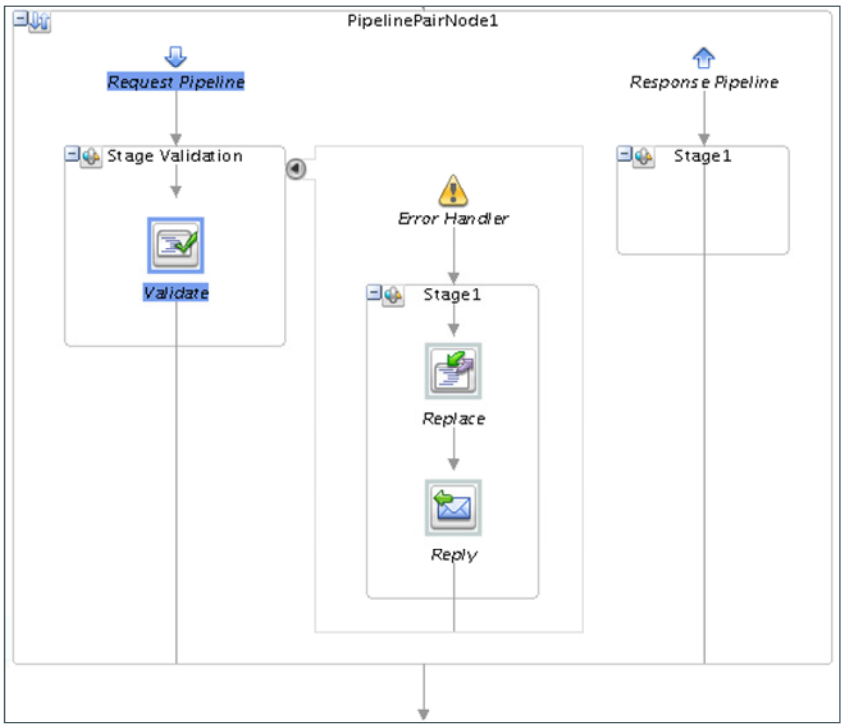
- Message validation
- Fault handling
- Reporting

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Message Validation



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The client application may or may not format the incoming message. To make sure that the message being passed in an enterprise environment is valid, Service Bus provides a validate action to let you validate elements.

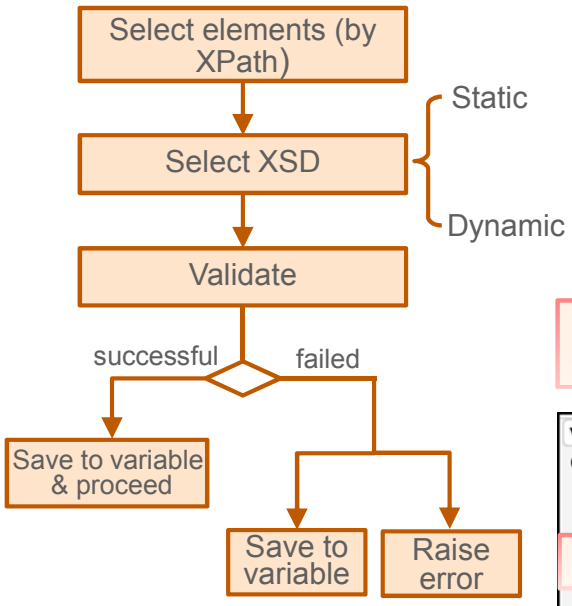
How to handle an error scenario:

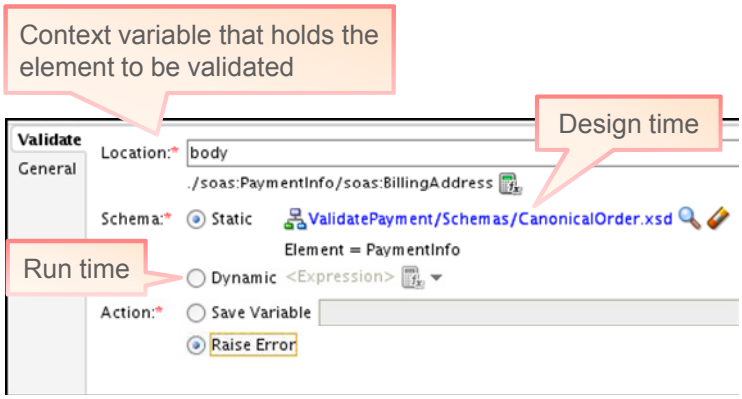
Configure a stage error handler to specify what should happen if an error occurs in a specific location in the message flow.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Validate Action





Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Use a validate action to validate elements selected by an XPath expression against an XML schema element or a WSDL resource. You can validate global elements only; Service Bus does not support validation against local elements. You can also choose to dynamically select the XML schema element or WSDL resource at run time, based on the result of an XQuery expression.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Dynamic Validation

Allows the user to select the resource (WSDL or schema) to validate against, based on the result of XQuery expression at run time.

- Example of dynamically specifying a schema resource:

```
<validate xmlns="http://www.bea.com/wli/sb/context">
  <schema>default/MySchema</schema>
  <schemaElement>
    <localname>MyElementType</localname>
  </schemaElement>
</validate>
```

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Dynamic validation allows the user to validate the input data or other data dynamically based on a certain condition. Thus, the process can be stopped at the beginning itself when the data validation goes wrong. You have the option to save the validation result or raise an error.

Here is another example of dynamically specifying a WSDL resource:

```
<validate xmlns="http://www.bea.com/wli/sb/context">
    <wsdl>default/MyWSDL</wsdl>
    <schemaType>
          <namespaceURI>http://openuri.org</namespaceURI>
          <localname>MyType</localname>
    </schemaType>
</validate>
```

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Quiz

The Validate action validates elements selected by an _____ expression.

a. XPath
b. XQuery
c. XSLT
d. nXSD

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Answer: a

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Message validation
- Fault handling
- Reporting

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Faults: Overview

- Faults:
  - Errors encountered during the execution of an application
  - Similar to exceptions in Java
- Types of faults:
  - Business faults:
    - Exceptional business behavior: Caused by a failure to meet a certain business rule
    - Faulty operation by users: Caused by human errors
  - Runtime/technical faults:
    - Software errors: Caused by programming bugs in applications
    - Technical errors: Caused by problems in the infrastructure on which applications run

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

A fault can be defined as something that is unusual and happens outside the normal and expected operational activity of a process. There are different types of faults, which can be categorized as follows:

- **Exceptional business behavior:** Faults caused by a failure to meet a certain business rule (for example, a customer with a bad credit rating, a new customer that wants to purchase something but is unknown in the CRM system, an invoice with an incorrect invoice amount)
- **Faulty operation by users:** Faults caused by human errors when using an application (for example, entering a wrong credit card number, accidentally switching the to and from date when booking flight tickets, and so on)
- **Software errors:** Faults caused by programming errors in applications, faults in third party software libraries that are used, software faults and bugs in packaged applications, and so on (for example, division by zero, infinite loops, memory leaks, null pointer exceptions, and so on)
- **Technical errors:** Faults caused by errors in the underlying infrastructure or middleware components on which applications run (for example, network errors, server failures, corrupt disks, full tablespaces, and so on)

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Fault-Handling Challenges

The handling and propagation (passing on) of faults is a multi-dimensional problem. Strategies for fault handling must consider the following:

- Fault type
  - Business
  - Runtime (technical)
- Interaction type
  - Synchronous
  - Asynchronous
- Transactionality of the services involved

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

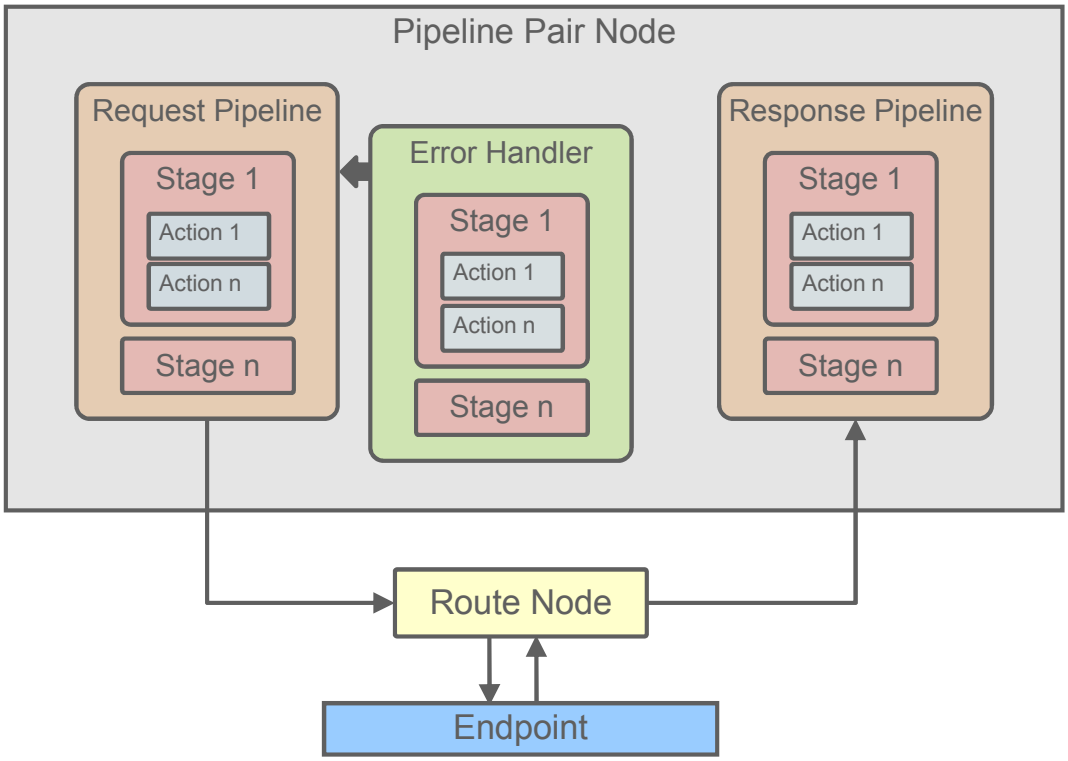
How the fault is passed to the consumer depends on the protocol being used.

- For a synchronous request/response message exchange pattern, if it is SOAP-based web service, you can declare and return a SOAP fault message.
- If you use asynchronous communications, a fault has to be returned as a normal callback message.
- In case of RESTful web services, a fault situation should be signaled using the HTTP status code, with the message holding more information about the error.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Handling Errors in OSB



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle Service Bus (OSB) provides a mechanism to handle errors by enabling you to define error handlers.

An error handler is a pipeline that allows you to perform various actions such as logging, transformation, and publishing to handle errors appropriately. If an error occurs in a stage, pipeline, or service, a sequence of steps is executed. This sequence of steps constitutes an error pipeline for that stage.

The graphic in the slide depicting message flow consists of a request pipeline and a response pipeline. Both pipelines are multistage. The pipeline level error handler is separate from the request pipeline and is also multistage. The A route node is shown passing messages from the proxy service to the business service.

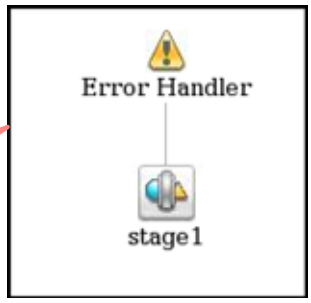
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Error Handlers

- An error handler is a special stage where a set of actions can occur if something goes wrong.
- The following error handler scopes are defined:
  - Stage
  - Route node
  - Pipeline pair (both Request Pipeline and Response Pipeline individually)
  - Pipeline service
- If no error handler is defined, there is a default system error handler.

All configuration for an error handler is in one or more stages contained by the error handler.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When a stage node or route node fails, the error gets propagated to the stage or route error handler. If you haven't defined a stage or route error handler, the error gets propagated to the Request/Response pipeline error handler. If you haven't defined a Request/Response pipeline error handler, the error gets propagated to the pipeline service error handler. If you haven't defined a pipeline service error handler, the error gets propagated to the top level, the system error handler.

There are exceptions to the scope of error handlers. For example, an exception thrown by a non-XML transformation at the stage level is caught only by the service-level error handler. Suppose a transformation that transforms XML to MFL for an outgoing proxy service response message occurs; it always occurs in the binding layer. Therefore, if a non-XML output is missing a mandatory field at the stage level, only a service-level error handler can catch this error.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

`$fault` Variable

- A predefined context variable used to hold information about any error that occurs during message processing
- Populated before the appropriate error handler is invoked
- Includes the following subelements:
  - `errorCode`
  - `reason`
  - `details`
  - `location`
  - `java-exception`
  - `stack-trace`

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

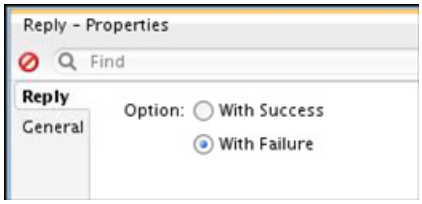
- **`errorCode`**: Specifies the error code as a string value
- **`reason`**: Contains a text description of the error
- **`details`**: Contains the user-defined XML content related to the error
- **`location`**: Identifies the node, pipeline, and stage in which the error occurred; also identifies whether the error occurred in an error handler. The subelements include:
  - **`node`**: The name of the pipeline, branch, or route node where an error occurred; a string
  - **`pipeline`**: The name of the pipeline where an error occurred (if applicable); a string
  - **`stage`**: The name of the stage where an error occurred (if applicable); a string
  - **`error-handler`**: Indicates whether an error occurred in an error handler; a Boolean

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Reply Action

- Specify that an immediate reply be sent to the invoker
- Can be used in the request, response, or error pipeline
- Can be configured to result in a reply with success or failure



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

You can configure Reply action to result in a reply with success or failure. There may be a scenario in which you receive a message from an HTTP request and the back-end service is constructed as a one-way JMS business service. In this case, you may want to respond with a Success after the message is sent (placed on the outbound transport) to the one-way JMS business service.

You can use the Reply action in an error handler to indicate that there is an error. Any received SOAP fault is stored in `$body`, so if a “Reply with Failure” is executed without modifying `$body`, the original SOAP fault is returned to the client that invoked the service. If a Reply action is not configured, the system error handler generates a new SOAP fault message.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Generating Error Messages Using Reply Action

The type of error returned depends on the transport used.

Transport	Error Returned
HTTP	HTTP 500
JMS	Sets the `JMS_BEA_Error` property to `true`
SOAP	SOAP fault

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

In the event of errors for request/response type inbound messages, it is often necessary to send a message back to the originator outlining the reason why an error occurred. You can accomplish this by using a Reply with Failure action after configuring the message context variables with the response you want to send. For example, when a JMS message fails, Reply with Failure sets the `JMS_BEA_Error` property to `true`.

A Reply action will always return the content of the `$body` variable as it is. The difference between Reply with Success and Reply with Failure lies in the content of transport headers being returned. In case of HTTP Service, if you Reply with Success, the calling service will receive HTTP status code 200 or 0. In case of Reply with Failure, the calling service will receive HTTP status code 500, which will trigger the error handling in calling service. The best practice for SOAP-based services is to always replace the content of `$body` with a SOAP Fault when doing a Reply with Failure, so that the calling application receives details of the fault. Otherwise, it just gets transport status code for failure but the content of reply will still be the original content of `$body`, which might not indicate what the error was.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Quiz

The smallest error handler scope you can define is at action level.

a. True

b. False

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Answer: b

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Message validation
- Fault handling
- Reporting

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Reporting Actions

- The Reporting Actions category has actions to log or report errors and generate alerts if required in a message flow in a stage.
- There are three types of actions:

	Report	Enable message reporting for a proxy service.
	Log	Construct a message to be logged.
	Alert	Send an alert notification based on pipeline message context.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Some use cases require error reporting. You can use the report action in these situations. For example, consider a scenario in which the request pipeline reports a message for tracking purposes, but the service invoked by the route node fails after the reporting action. In this case, the reporting system logged the message, but there is no guarantee that the message was processed successfully, only that the message was successfully received.

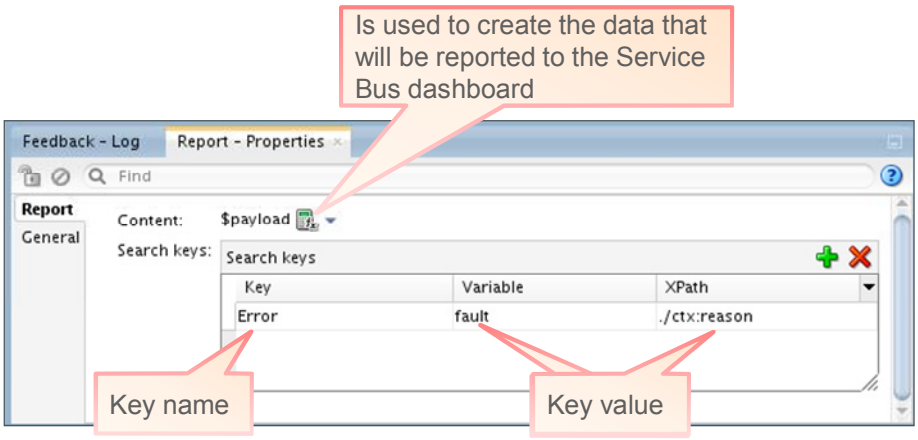
You can use the Oracle Service Bus Console to track the message to obtain an accurate picture of the message flow. This allows you to view the original reported message indicating the message was submitted for processing, and also the subsequent reported error indicating that the message was not processed correctly

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Report Action

The Report action allows you to extract information from each message and write it to the OSB Reporting Data Stream.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When you configure a Report action, use key values to extract key identifiers from the message. You can configure multiple keys. You can capture information not only from the body of the message but also from any other variable associated with the message, such as header or inbound variables.

Under Search Keys, the Key field is used to specify a key name. The Variable field and XPath field together specify the key value. The Variable field specifies the context variable and the XPath field specifies an XPath expression.

You use key value pairs to extract key identifiers from any message context variable or message payload, and ignore the rest of the message. The keys are a convenient way to identify a message. They are displayed as report indexes in the Reporting module.

For example, consider a report action configured on an error handler in a stage. The action reports the contents of the fault context variable in the event of an error. The report action is configured as follows:

- - Key name = Error
- - Key value = ./ctx:reason in variable $fault

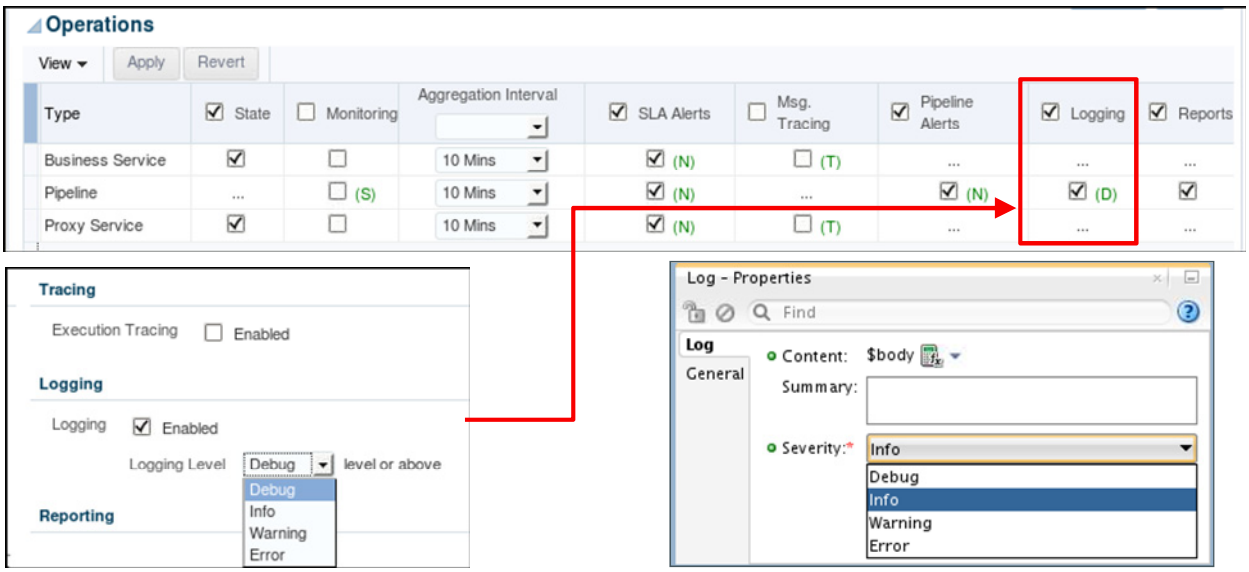
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Log Action

- Ensure that global and pipeline logging are enabled in EM.
- Configure log level at OSB and pipeline level.

Configuring logger for pipeline



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

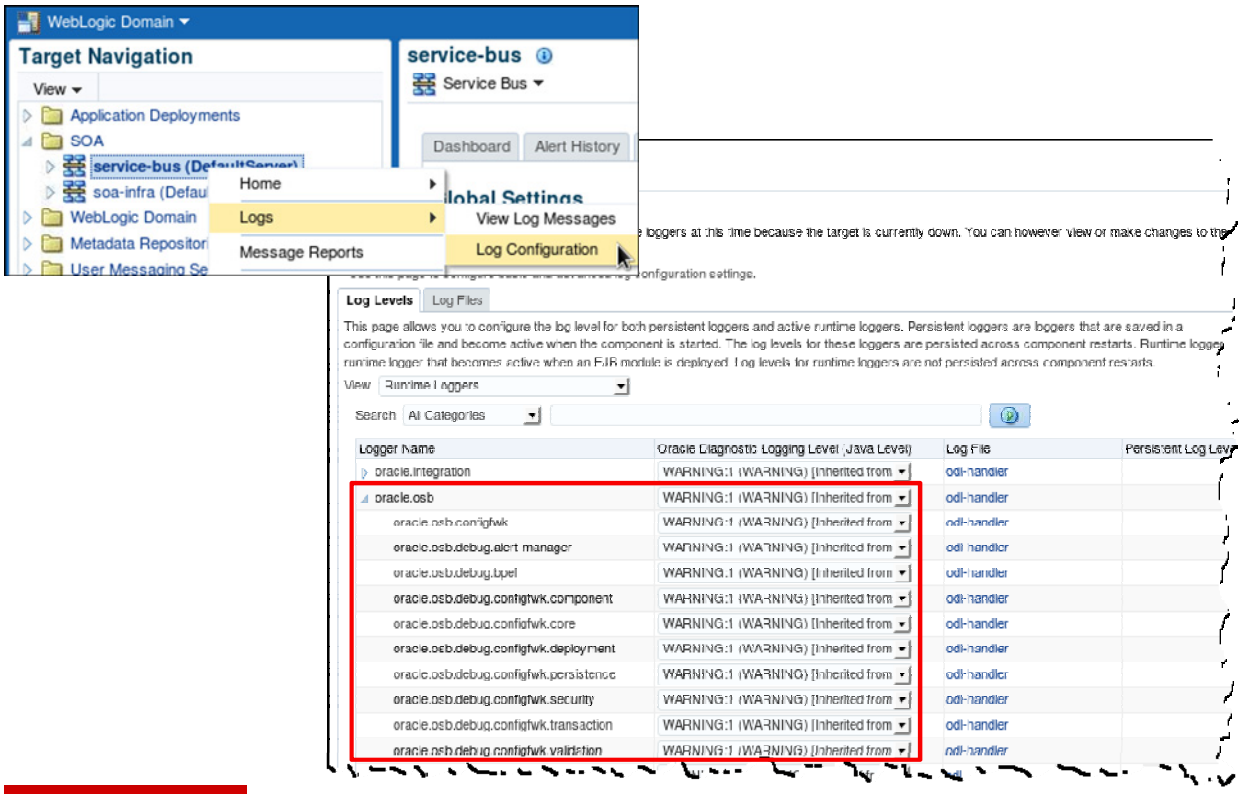
Log Action has four log severity levels:

- **Debug:** Though your application is under development, you may find it useful to create and use messages that provide verbose descriptions of low-level activity in the application.
- **Info:** This is used for reporting normal operations. It is a low-level informational message.
- **Warning:** A suspicious operation or configuration has occurred but it may not affect normal operation.
- **Error:** A user error has occurred. The system or application can handle the error with no interruption and limited degradation of service.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Configuring OSB Log Level



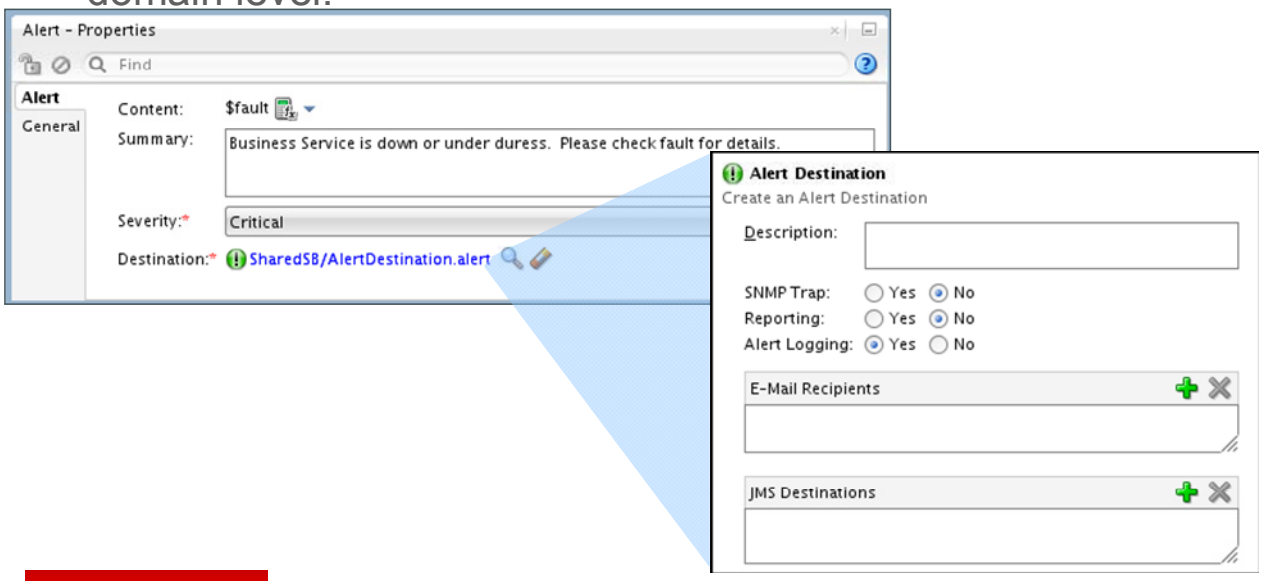
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Alert Action

- Alert action is for business use or reporting errors, and not for monitoring system health.
- Pipeline alerting should be enabled for the service or at the domain level.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Use the alert action to generate alerts based on message context in a pipeline, to send to an alert destination. Unlike SLA alerts, notifications generated by the alert action are primarily intended for business purposes, or to report errors, and not for monitoring system health. Alert destinations should be configured and chosen with this in mind.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Summary

In this lesson, you should have learned how to:

- Validate a message
- Describe basic fault handling and propagation
- Handle errors in Service Bus

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Practice 5: Overview

- 5-1: Importing template resources and creating a business service
- 5-2: Creating the pipeline that validates messages using the Pipeline template

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

6

Transforming Messages

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Objectives

After completing this lesson, you should be able to:

- Describe the role of XPath, XQuery, and XSLT in the way Service Bus handles data
- Describe standard and custom XPath functions
- Describe how the XSLT Mapper can be used to create XSL transformations
- Use the XQuery Mapper to create XQuery transformations
- Transform non-XML data to XML data with nXSD

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

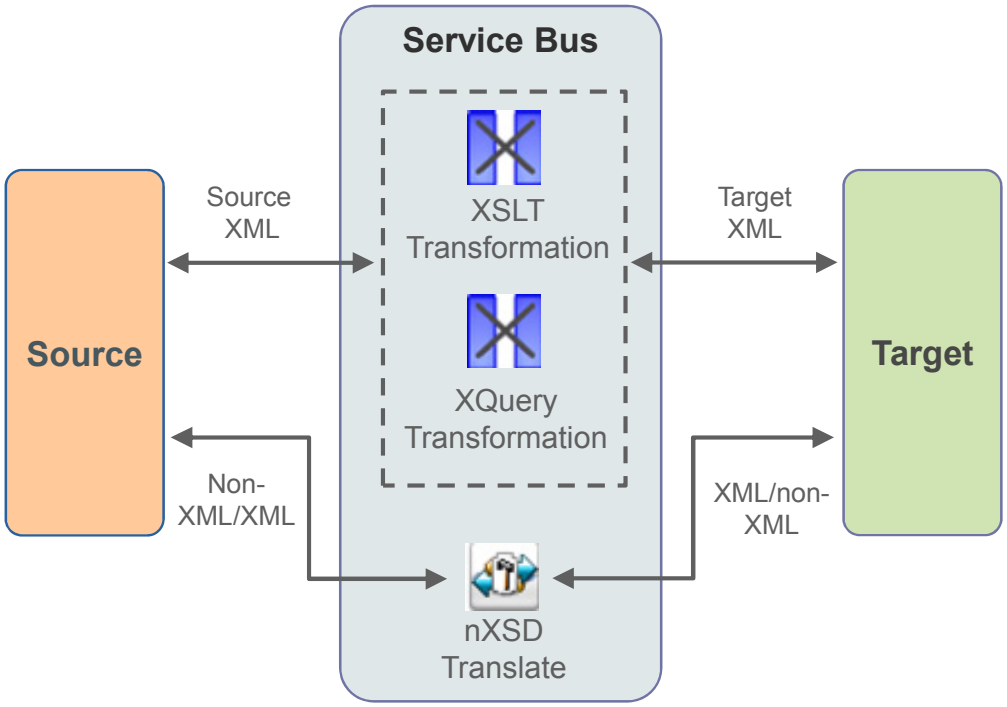
- XPath
- XSLT transformation
- XQuery transformation
- nXSD

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Message Transformation in Service Bus



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Transforming messages from one format to another is an important feature in Oracle Service Bus. XML-to-XML transformation can be performed by using a stylesheet language called eXtensible Stylesheet Language Transformation (XSLT) or an XML Query language called XQuery. Both of them can extract and restructure XML data from one format to another, as well as transform or enhance the content. This lesson examines each of these technologies, along with the XPath addressing language that both are dependant on.
Oracle Service Bus also supports translation from native data format to XML, or vice versa. Translation is performed by sampling native data and creating the native XSD (nXSD) grammar for translation of the native data.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Terminology and Concepts

- **XPath:** A language used for addressing (finding, retrieving, or manipulating) values in XML sources.
- **XML Stylesheet Language for Transformations (XSLT):** A language used for transforming data from one XML structure to another.
- **XQuery:** A language designed for querying (finding and extracting elements and attributes), transforming, and accessing XML and relational data.

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

- **XPath:** XPath is a language designed to address specific parts of an XML document. XPath models an XML document as a tree of nodes. The types of nodes include element nodes, attribute nodes, and text nodes. By using XPath, you construct an expression (location path) that accesses a node or set of nodes. XPath is a major element in W3C's XSLT standard. See http://www.w3.org/TR/xpath for the specification.
- **XSLT:** XML documents represent data, organized in some predefined, structured way. Sometimes, data in XML documents must be transformed before it can be sent or processed. Source and target documents are generally based on different XSD documents. XSLT is a language that defines the XML elements and attributes to produce and the values for these to write, usually queried from the source document. Data is queried from the source document by using XPath expressions.
- **XQuery:** XQuery is the W3C standard language designed for querying, transforming, and accessing XML and relational data. XQuery is similar to SQL in many ways. As SQL is designed for querying structured, relational data, XQuery is designed especially for querying unstructured, XML data from a variety of data sources.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XPath

- Treats XML documents as trees of nodes
- Uses path expressions and conditions to select nodes or node-sets in an XML document
- Is a fundamental component of XSLT and XQuery

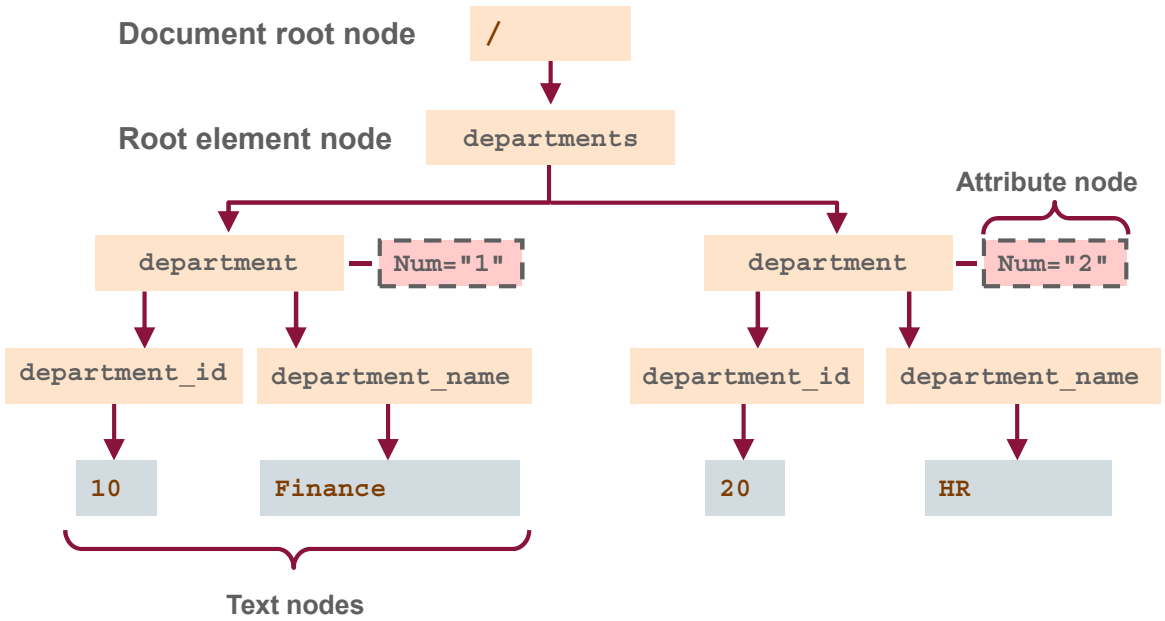
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

XPath is a W3C standard language used for finding information in an XML document. XPath models an XML document as a tree of nodes, and expresses a navigation path through the hierarchical structure of an XML document. XPath can be used by itself, but is often used in conjunction with XSLT and other XML topics, such as XQuery and XPointer. The result of the XPath can be a collection of nodes. It does not need to be a unique node.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XPath: Examples



Example:

- `/departments/department/department_id`
- `/departments/department[@Num="1"]`

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

XPath models an XML document as a tree of nodes. The graphic in the slide depicts the different types of nodes as seen by XPath in an XML document.

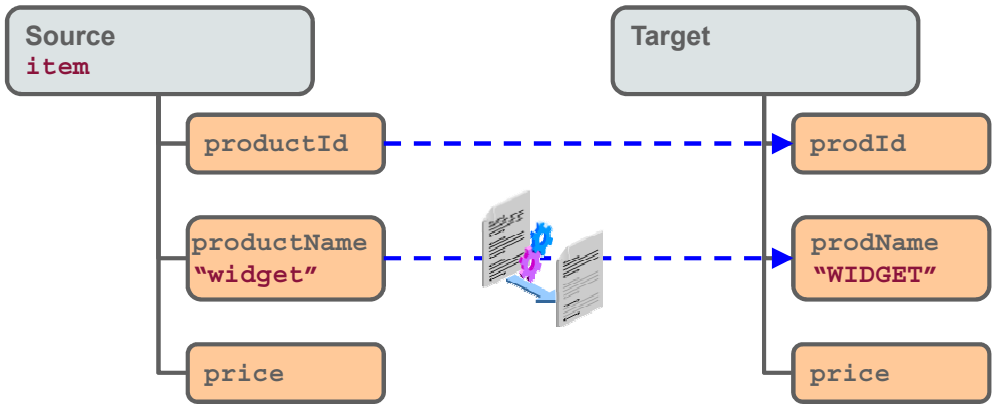
The examples mean:

- `/departments/department/department_id`: Select the `department_ids` of all department elements that are the children of the departments element.
- `/departments/department[@Num="1"]`: Select the department that has a `Num` attribute whose value is "1."

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XPath Functions




```
<ns1:item>
    <ns1:prodId>
        <xsl:value-of select="impl:productID"/>
    </ns1:prodId>
    <ns1:prodName>
        <xsl:value-of select ="xp20:upper-case(impl:productName)"/>
    </ns1:prodName>
</ns1:item>
```

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

As you have seen, XPath is used to reference and navigate through the hierarchical structure of an XML document. XPath also includes a library of functions that enable evaluation and manipulation of data found at the referenced location in the document.

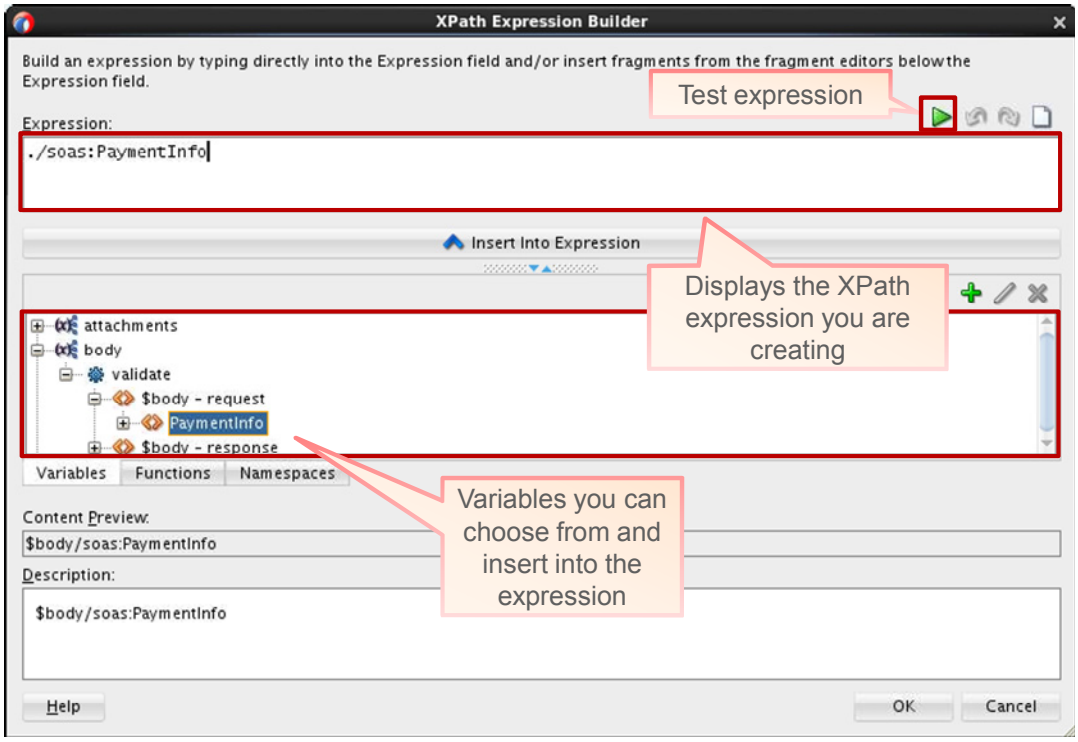
In the example in the slide, the value found in the `productName` node is converted to uppercase before being assigned to the `prodName` node in the destination document.

XPath functions can be nested as well. If you want to copy only the first three characters of the uppercase value to the destination, you could pass the result of the `upper-case` function to the `substring` function in an expression such as `<xsl:value-of select="substring(xp20:upper-case(impl:prodName),3)"/>`.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XPath Expression Builder



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

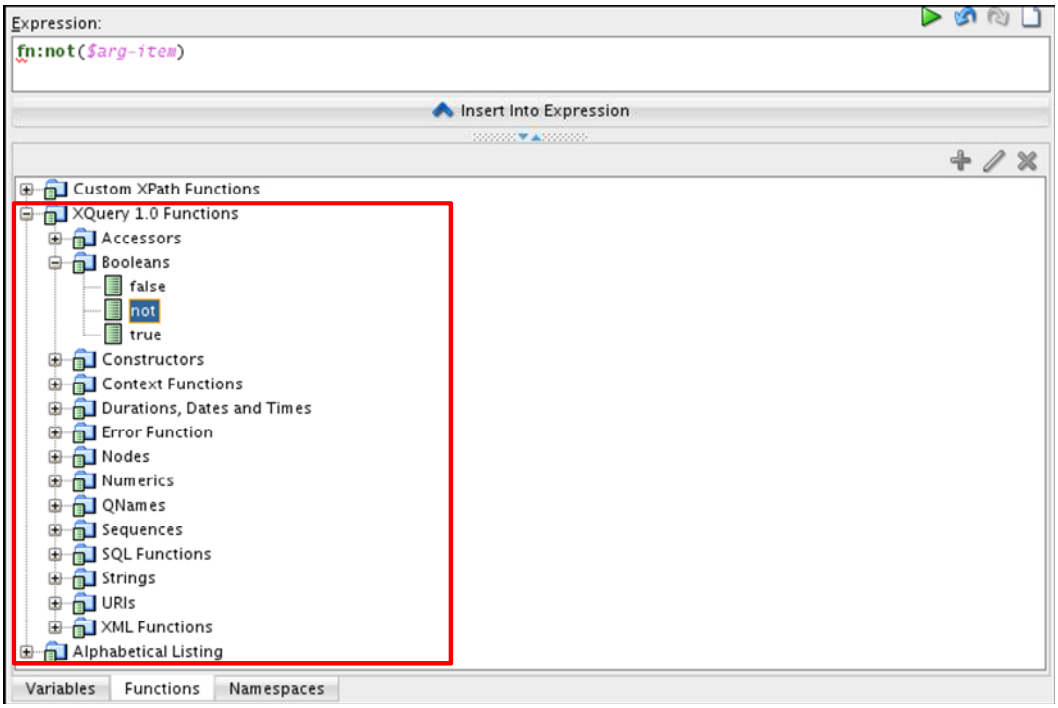
XPath Expression Builder/Editor is a tool that allows you to define an XPath expression through navigating available elements or just write it into the editor manually.

Click Insert Into Expression, or double-click the item in Variables, Functions, or Namespaces section, to insert it into the XPath expression field.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Standard XPath Functions



Note: XPath 2.0, XQuery 1.0, and XSLT 2.0 share the same functions library.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The Functions section contains a set of standard XPath/XQuery functions, organized alphabetically and by type, including any custom functions you have developed.

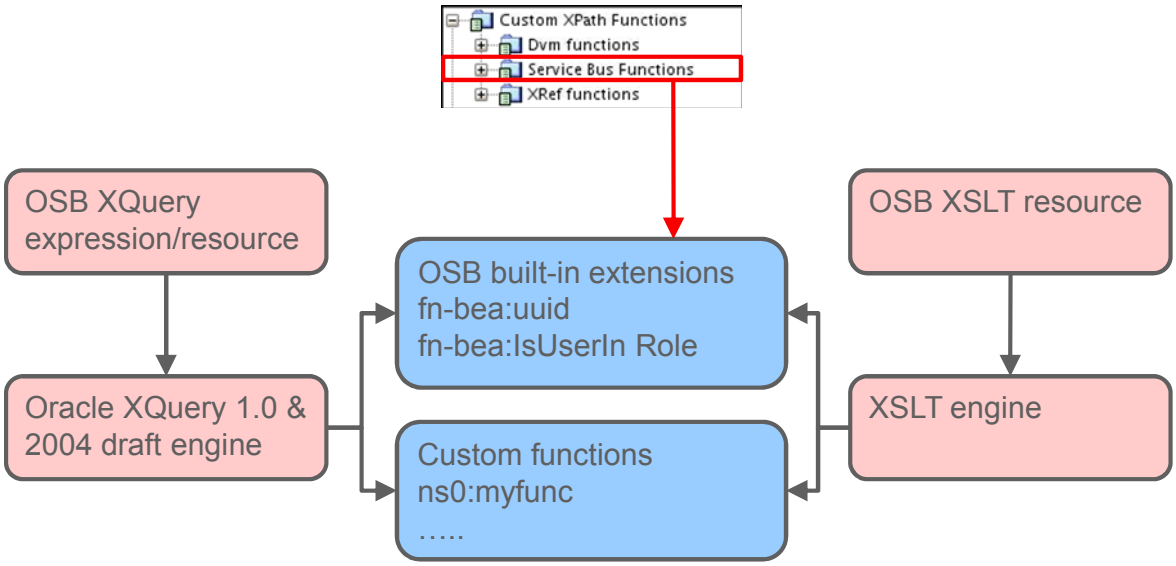
Insert a function into the Expression field by double-clicking it or clicking Insert Into Expression. For example, double-clicking the `not` Boolean function displays it in the Expression field.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Custom XPath Functions

Service Bus provides an extensible framework for creating custom XPath functions that you can use in the XQuery Expression Editors.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Custom XPath functions are used in XQuery Expression Editors in the development or runtime tooling, such as in message flows, Split-Joins, and XQuery Mapper transformations. Custom functions are available to all Oracle Service Bus projects and services in an installation; they are not scoped to specific domains.

You can include custom functions in both inline XQuery expressions and resources or XSLT resources, just as you would use functions provided by Oracle Service Bus. However, the syntax for invoking a custom function from within an XQuery resource or XSLT resource varies by the XQuery or XSLT engine you use with Oracle Service Bus.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Creating and Packaging Custom XPath Functions

1. Create Java class and method for a custom function.
2. Package the custom function class in a JAR file.
3. Place the JAR file in the OSB XPath-functions directory: `$ORACLE_HOME/osb/config/xpath-functions/`
4. Register the custom function.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

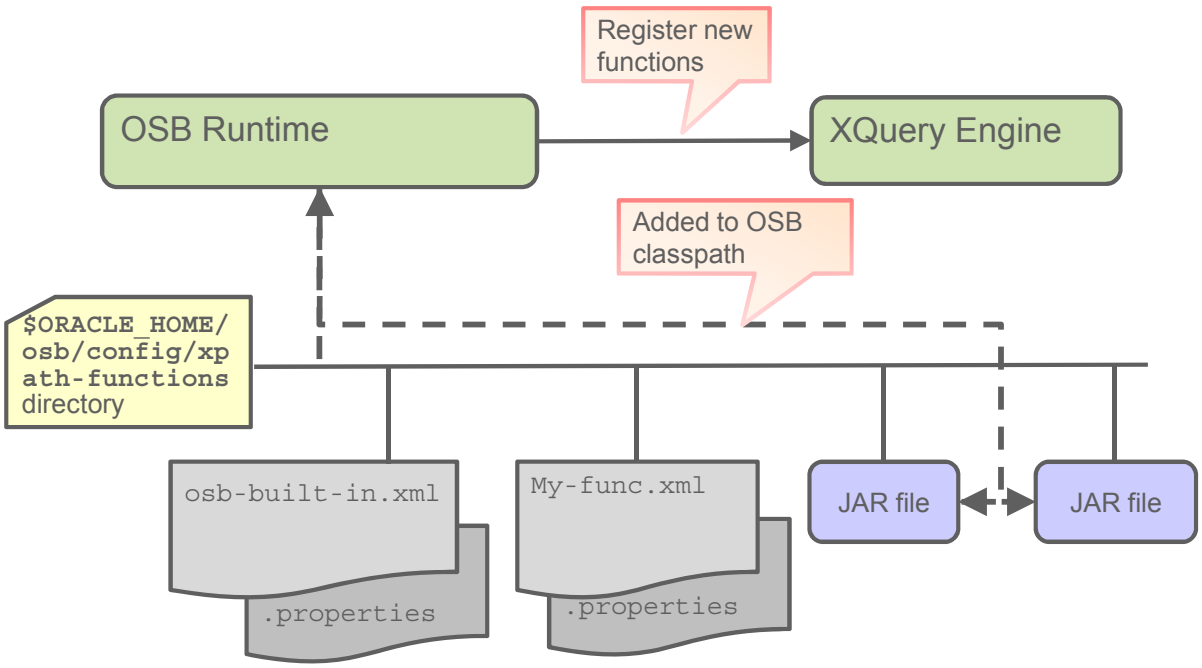
Your custom functions do not appear in the XQuery expression editor until Oracle Service Bus can find your custom class. The slide lists the steps for making Oracle Service Bus know about your custom function class to include them in the XQuery editors and let you use those functions.

When you finished the mentioned steps, at IDE and server startup, Oracle Service Bus looks for custom function classes in the `$ORACLE_HOME/osb/config/xpath-functions/` directory.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Registering a Custom Function



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Registering a custom function involves creating an XML file with an optional properties file for localization. Custom functions use the same extensible framework as the Oracle Service Bus built-in functions, so you can use those existing registration resources as guidance for registering custom functions. You can find these files at the following location:

`$ORACLE_HOME/osb/config/xpath-functions/`

OSB reads the content of the directory and then does the following:

- For each `.xml` file it finds, it goes through the list of functions in it and “registers” them with XQuery engine.
- Each function gets an XQuery signature based on its Java signature.
- For each `.jar` file it finds, it adds it to classpath when evaluating results of the XQuery expressions.

The content of the configuration XML file (for each new function) includes:

- Category ID (enter `.properties`)
- Group ID (enter `.properties`) [This is optional.]
- Name
- Namespace
- Java signature
- Deterministic or not (always produces same result for same arguments)

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

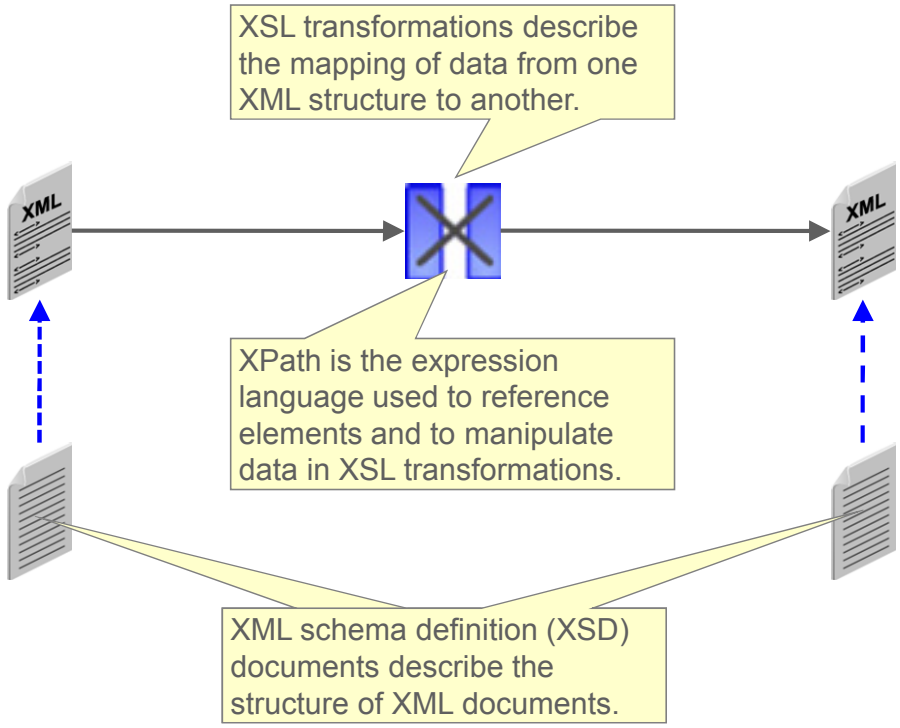
- XPath
- XSLT transformation
- XQuery transformation
- nXSD

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Data Standards



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

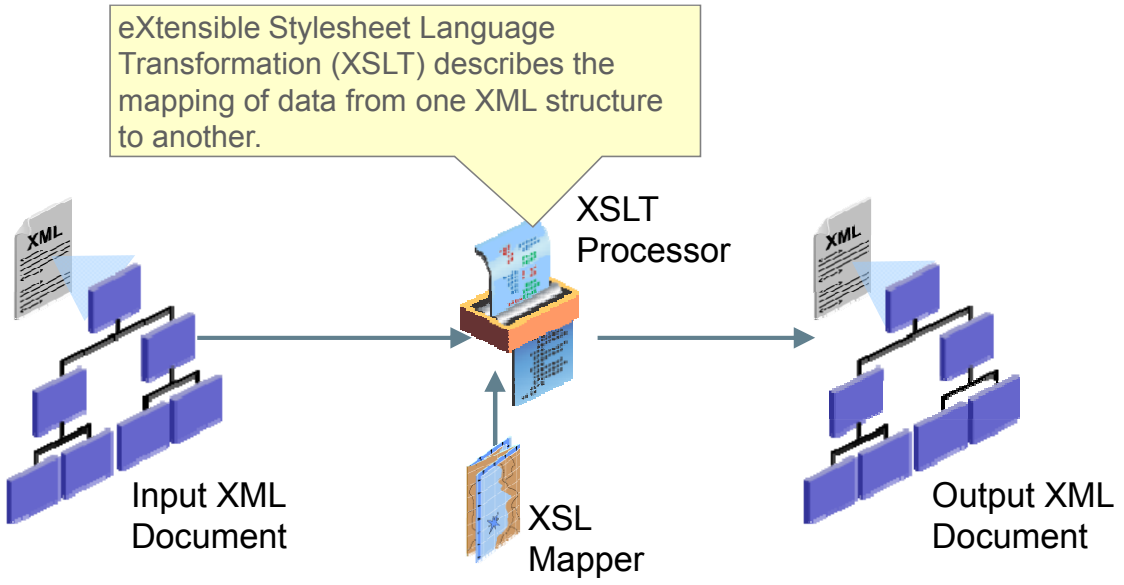
An XML schema definition, or an XSD document, is used to describe the structure of these XML messages. As its name suggests, an XSD is used to describe (or define) the structure (or schema) of an XML message. In fact, an XSD uses the XML format to provide that description.

eXtensible Stylesheet Language Transformation (XSLT) is used to describe how to map data from one XML structure to another. XSLT makes use of the expression language defined by XPath for selecting elements for processing, for conditional processing, and for generating text.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Transforming Data



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

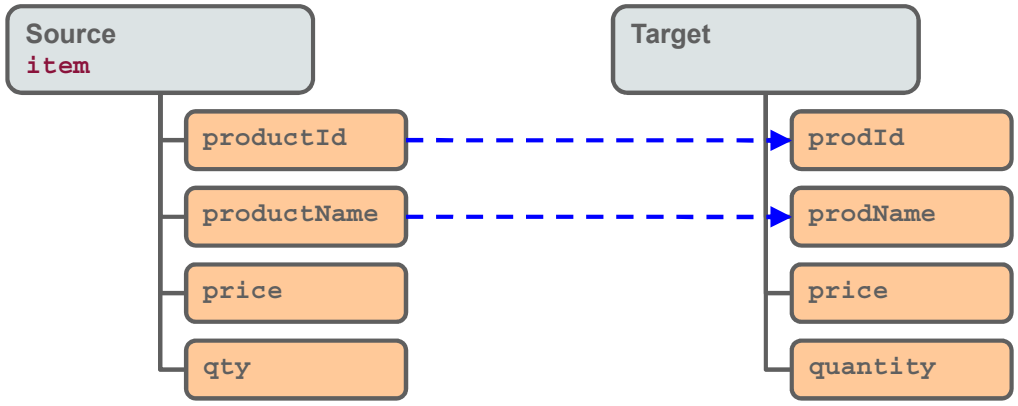
Service Bus supports data transformation from one XML schema to another. This enables data interchange and enrichment between applications by using different schemas. For example, the source XML schema may have last name and first name as separate values. The target XML schema may have the full name (both first and last) stored as one value. Through transformations, you can concatenate the first and last names from the source data before it reaches the target data.

Transformations are defined in XSL files that can be visually designed by using the XSLT Mapper (a visual editing tool), which is provided by Oracle JDeveloper 12*c*. The XSL file is deployed with the service so that Service Bus can process the mapping rules in an XSL processor that is available in the runtime environment.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XSL Transformations




```
<ns1:item>
    <ns1:prodId>
        <xsl:value-of select="imp1:productID"/>
    </ns1:prodId>
    <ns1:prodName>
        <xsl:value-of select="imp1:productName"/>
    </ns1:prodName>
</ns1:item>
```

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

In Oracle Service Bus 12*c*, an XSLT transformation references two documents—one describing the inbound message and the other describing the outbound message. It then includes a set of instructions that result in the creation of the outbound document.

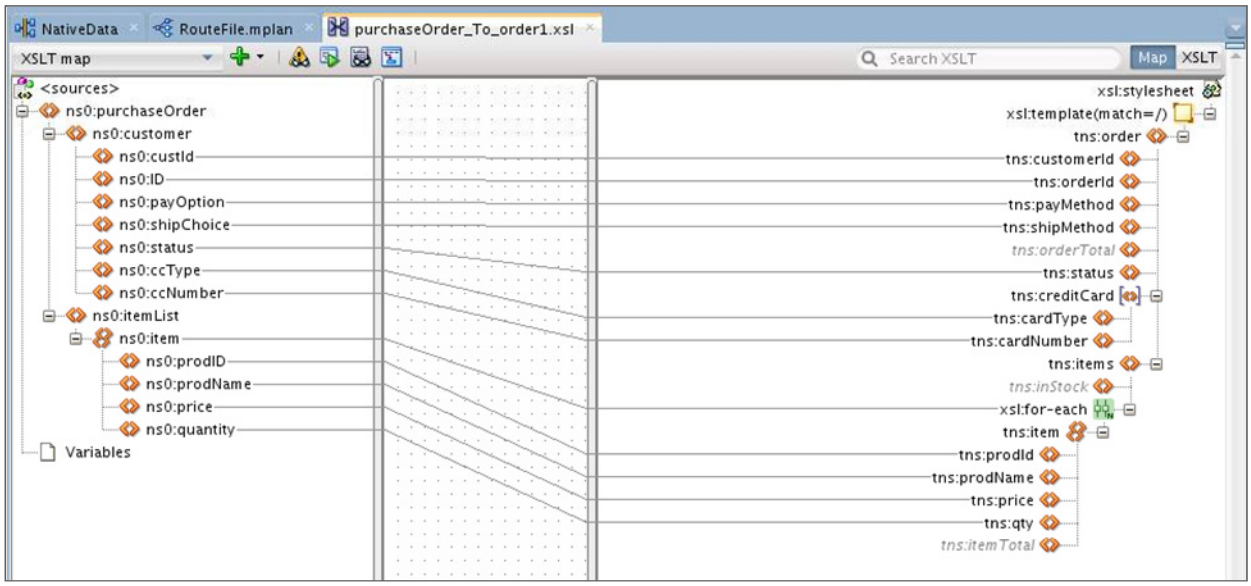
In the example in the slide, the `<xsl:value-of>` element is used to extract the value of an XML element and add it to the output stream of the transformation. The `select` attribute contains an XPath expression that references the path to the desired source node.

This expression instructs the XSLT processor to take the contents found at the `productId` location in the source message, and to place those contents in the `prodId` node in the outbound message. Similar instructions map the `productName` node to the `prodName` node.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XSLT Mapper



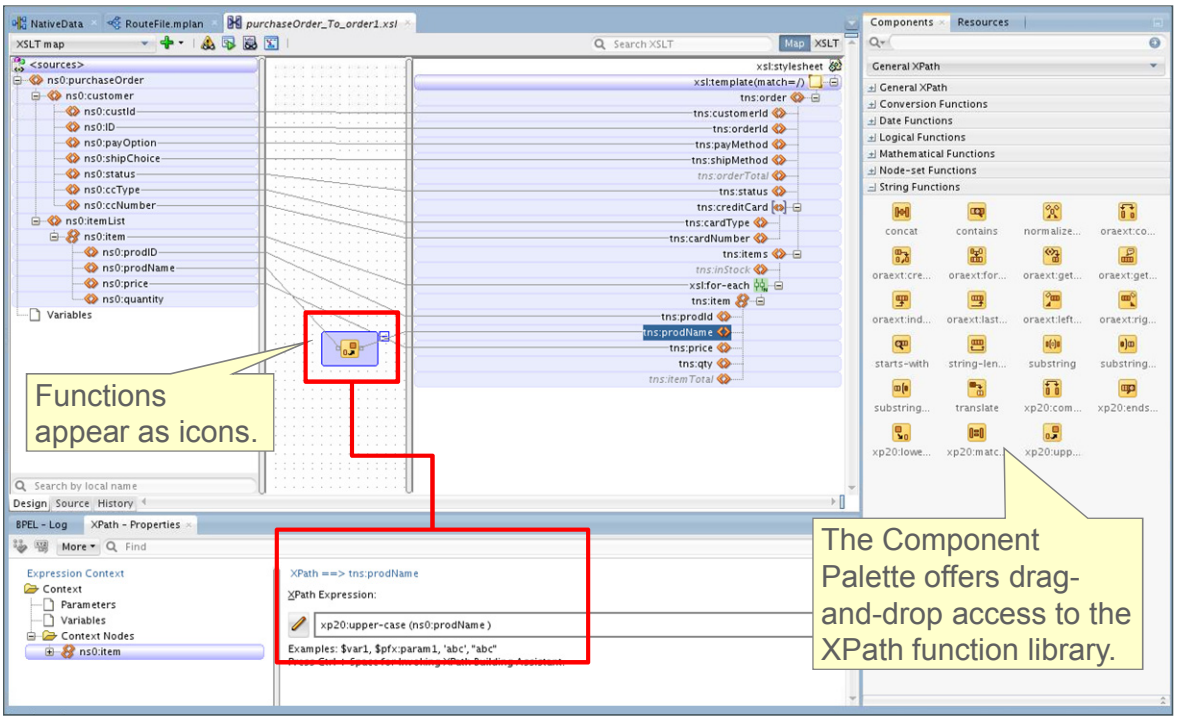
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The XSLT Mapper window enables you to visually create transformation rules that can be applied to message structures that are processed in Mediator routing rules. Transformation rules are created by dragging source elements to their respective target elements. A large number of built-in functions exist that you can use to enrich and alter source values before they are copied to their target elements. The XSL source editor can be used for XSL constructs and mappings that cannot be visually represented by the visual designer.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Using XPath Functions in XSLT Mapper



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

XPath functions provide the means to evaluate and manipulate data in an XSL transformation. Categories of functions include type conversion, date, logical, mathematical, string, and others. Text literals can also be added in a similar manner. The functions are accessed from the Component Palette, and added to mappings by dragging and dropping the appropriate function onto the center of the canvas, or directly onto an existing mapping.

Functions are displayed as icons. Nested functions are displayed with the innermost function to the left, and the subsequent outer functions to the right. Execution of the functions is also performed left to right.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- XPath
- XSLT transformation
- XQuery transformation
- nXSD

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XQuery: Introduction

XQuery:

- Allows you to select the XML data elements of interest, reorganize and transform them, and return the results in a structure of your choosing
- Uses and extends XPath to help navigate and extract elements and attributes from an XML document

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Data can be used for a variety of purposes. For example, sales data may be useful for reporting results to the tax authorities, or creating internal reports for planning. For each of these uses, we are interested in different elements of the data and expect it to be formatted and transformed according to our needs.

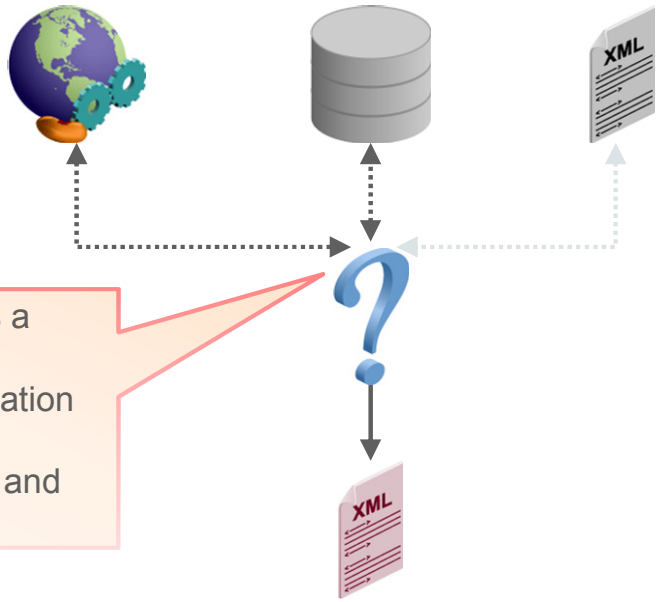
XQuery is a query language designed by the W3C to address these needs. It allows you to select the XML data elements of interest, reorganize and transform them, and return the results in a structure of your choice.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Uses of XQuery

- XQuery is built on XPath expressions.
- Applications of XQuery include:
  - XML querying
  - XML generation
  - XML transformation



XQuery provides a mechanism for extracting information from relational databases, files, and web services.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

XQuery is built on XPath expressions. It is used to query, generate, and transform XML content.

- **XML querying:** XML can map a large number of different data models. XQuery thus provides a mechanism for extracting information from relational databases, files, and web services simultaneously.
- **XML generation:** XQuery enables you to construct XML data by using the result of a query.
- **XML transformation:** XQuery can construct XML as a result of evaluating a query expression, in many cases more expressively and efficiently than XSL Transformations (XSLT). XQuery enables you to scan a hierarchy of elements. You can use it to apply transformations at each level of the hierarchy by defining a recursive function.

XPath and XQuery overlap to a very large degree. They have the same data model and the same set of built-in functions and operators. XPath is essentially a subset of XQuery. But XQuery has several features that are not included in XPath, such as XML constructors. Because these features are not relevant to selecting, but instead have to do with structuring or manipulating query results.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XQuery Support in Service Bus

Service Bus:

- Makes use of XQuery resources for various activities, like transformations, data selection, condition evaluation, and data manipulation
- Fully supports:
  - XQuery 1.0
  - XQuery 2004

Tool Type	Recommended Use	JDeveloper	Service Bus Console
XQuery Expression Builder	Script transformations using XQuery	X	X
XQuery Mapper	Create complex mappings	X	

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

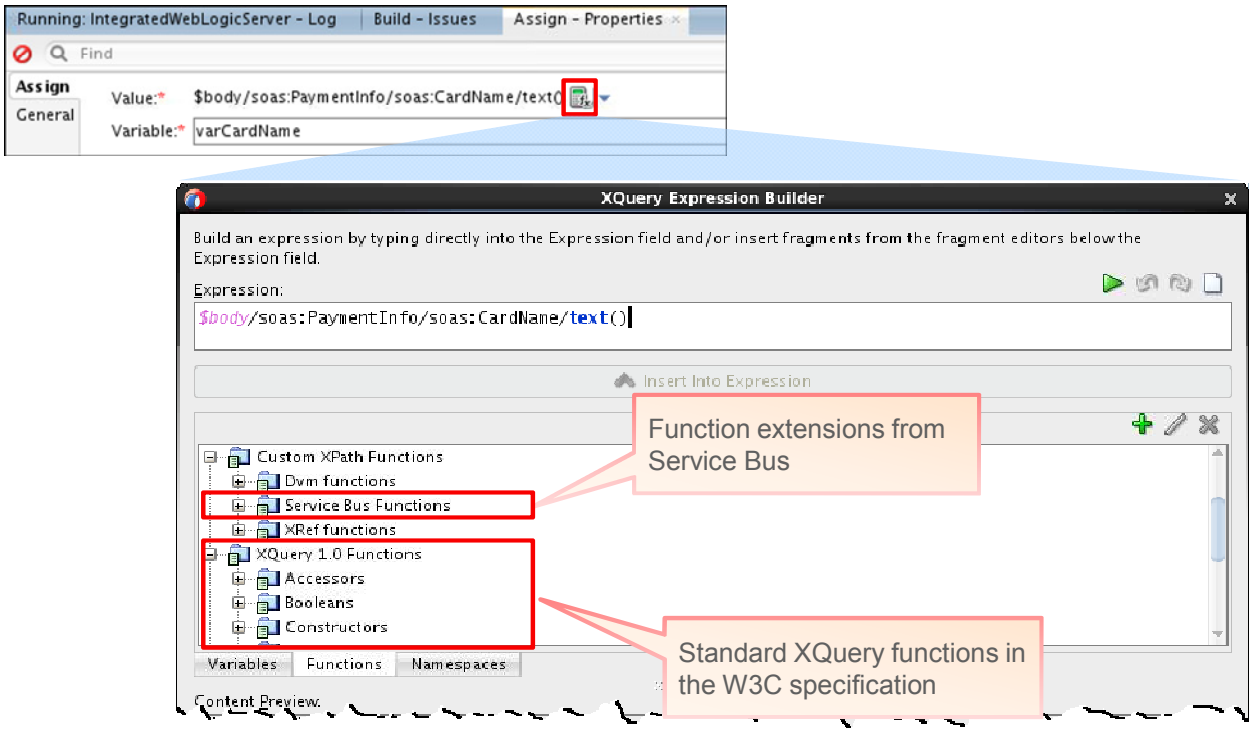
Service Bus fully supports XQuery 1.0. This includes optional features such as modules. The older XQuery 2004 is also supported.

JDeveloper provides both an Expression Builder, where you can script transformations by using XQuery, and an XQuery Mapper, where you can create complex mappings. The Oracle Service Bus Console provides an editor for scripting transformations by using XQuery. The editor provides options to define an XQuery expression or to define an expression that evaluates at run time to the name of an existing XQuery resource.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XQuery Expression Builder



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

For both JDeveloper and the console, you access the XQuery editor from an action in either a pipeline or split-join.

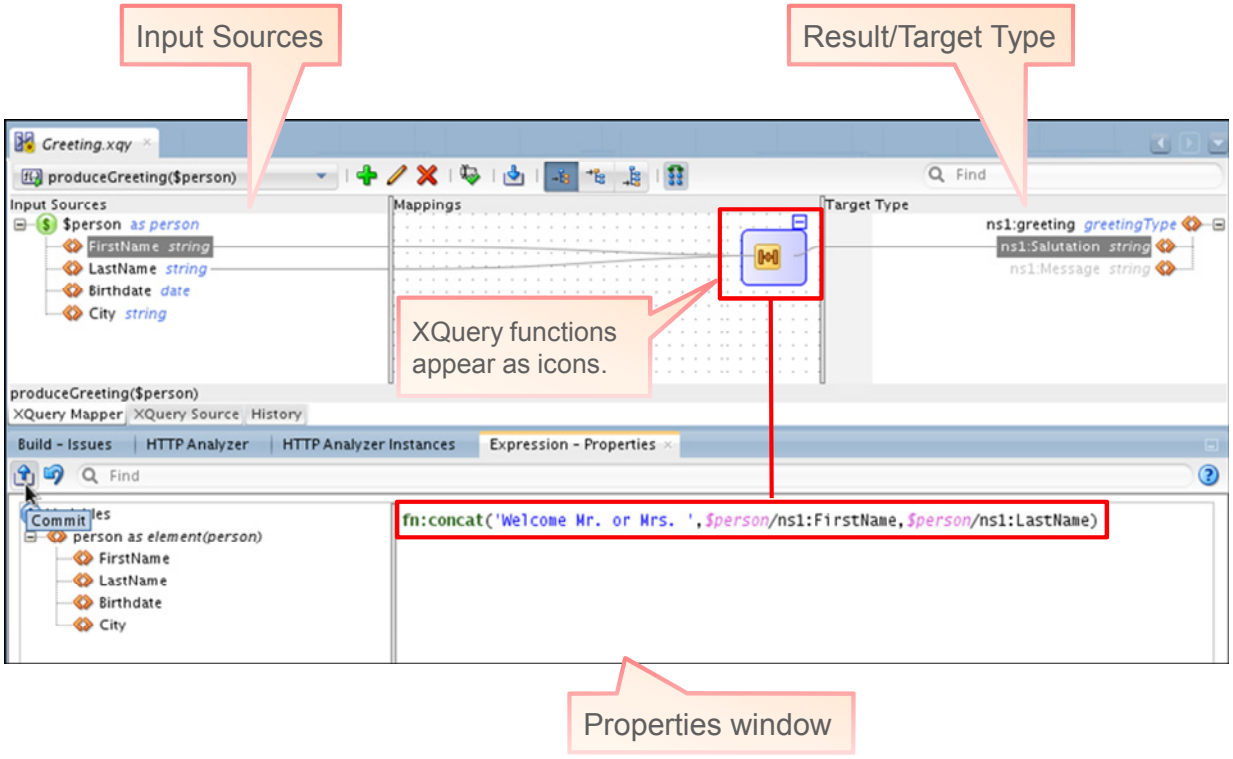
Service Bus supports the following XQuery functions:

- The standard XQuery functions described in the W3C specification:
  `http://www.w3.org/TR/xpath-functions/Opens a new window`
- Oracle function extensions and language keywords provided as part of the Oracle XQuery engine—with a small number of exceptions, as described in Section 19.5.1, "Supported Function Extensions from Oracle."
- Service Bus-specific function extensions

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XQuery Mapper



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The XQuery Mapper in JDeveloper is a graphical tool that lets you define mappings that transform data between XML, non-XML, and Java data types so you can rapidly integrate heterogeneous applications. For example, XML data that is valid against one schema can be converted to XML that is valid against a different schema. The data can be based on XML schemas, WSDL files, and Message Format Language (MFL) files. You can create an XQuery mapping in JDeveloper, and then upload the `.xqy` file generated by the mapper to an XQuery resource in the Oracle Service Bus Console. XQuery mappings are stored in XQuery resources in Service Bus, which can be referenced from the expressions you create by using the expression editors in a message flow action.

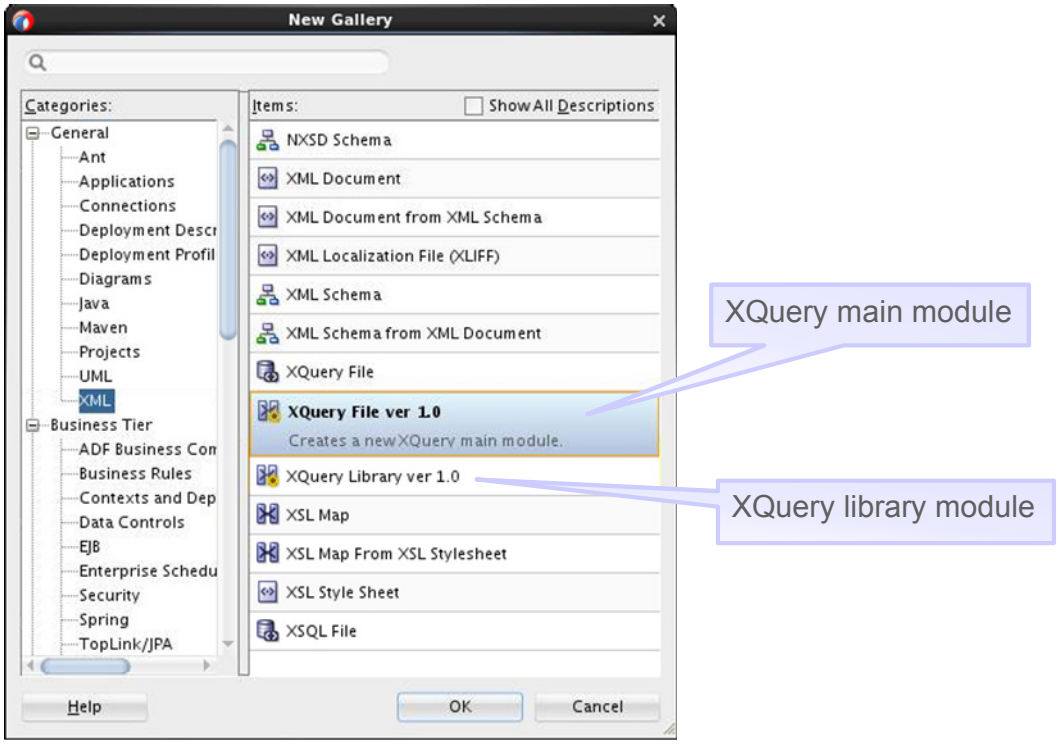
In the image in the slide, the left pane of the XQuery Mapper includes the input sources or parameters for the XQuery function. If your XQuery file has multiple functions, you can select the function to display by using the toolbar over the mapper panes. The right pane includes the target schema tree, which corresponds to the XQuery function's result type. The center pane helps you map the source and the target schema elements using XQuery functions.

The Properties window displays the XQuery expression for the node selected in the target tree. XQuery expressions created using drag and drop can be edited in the Properties window. The Properties window can also be used to create more complex XQuery expressions.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Types of XQuery Maps



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

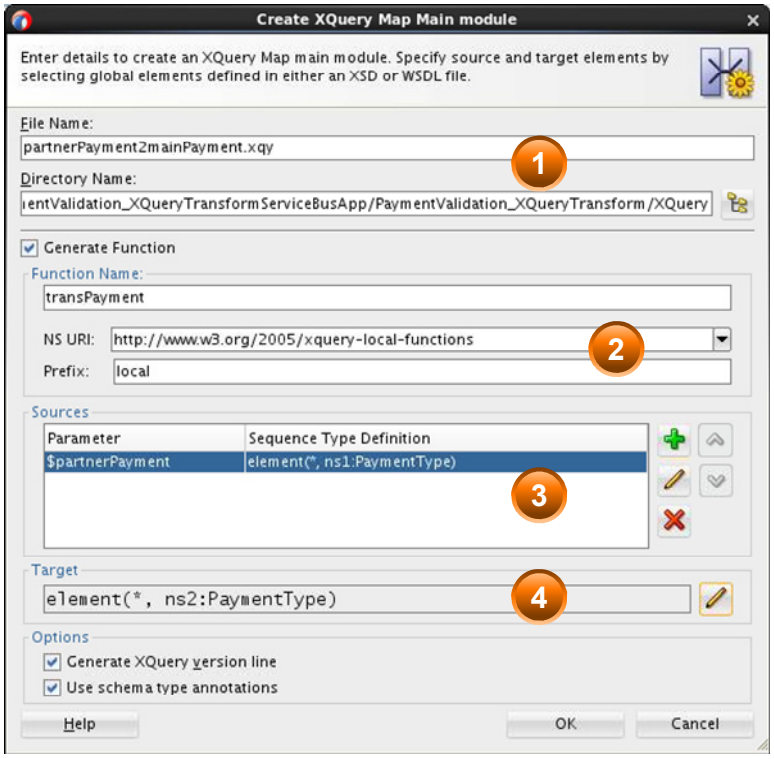
XQuery maps can be created as main modules and library modules. A main module is an executable XQuery file. A library module is used to group and store XQuery functions. When you import a library module into a main module, all functions in the library module become available in the main module.

You can create a library module with XQuery functions that can be reused across XQuery files.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Steps for Creating XQuery Map Files



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

1. Specify file name and directory.
2. Specify the namespace for the function.
3. Add a parameter for the function's input sources.
   - Specify the data type for the parameter.
4. Define the function's result data type.
5. Map source and target by using XQuery Mapper.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XSLT Versus XQuery

- Similarities:
  - XPath
  - Data model
  - Functions and operators
- Differences:
  - Syntax
    - XQuery is similar to SQL.
    - XSLT stylesheets use XML syntax.
  - Performance
    - XSLT loads the entire input document in the memory.
    - XQuery loads only the objects that need to be used by the current statement.

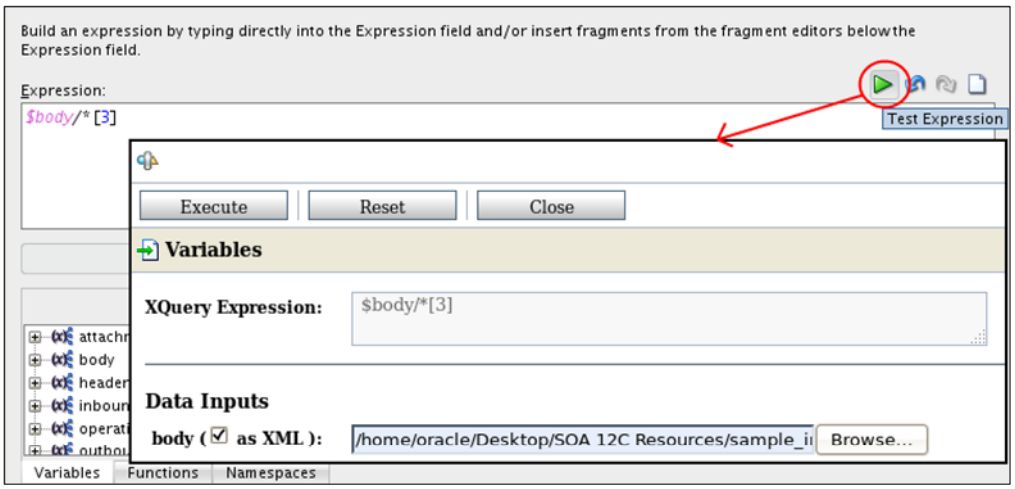
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Both XSLT and XQuery are languages designed to query and manipulate XML documents. Their features and capabilities have lots of overlap. The slide lists the similarities and differences between them.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Test Expressions at Design-Time



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

An other Developer feature in 12*c* is the possibility to test expression on Design-Time. When working in the Expression Builder you can click the Test Expression icon to test the expression. JDeveloper will deploy a small artifact to the Design-Time environment of SB and start a small browser window where you can test the expression.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- XPath
- XSLT transformation
- XQuery transformation
- nXSD

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XML to non-XML with nXSD

Requirement:

- Non-XML-based data should be processed by the Service Bus, and should be treated as structured.

Solution options:

- nXSD
- MFL

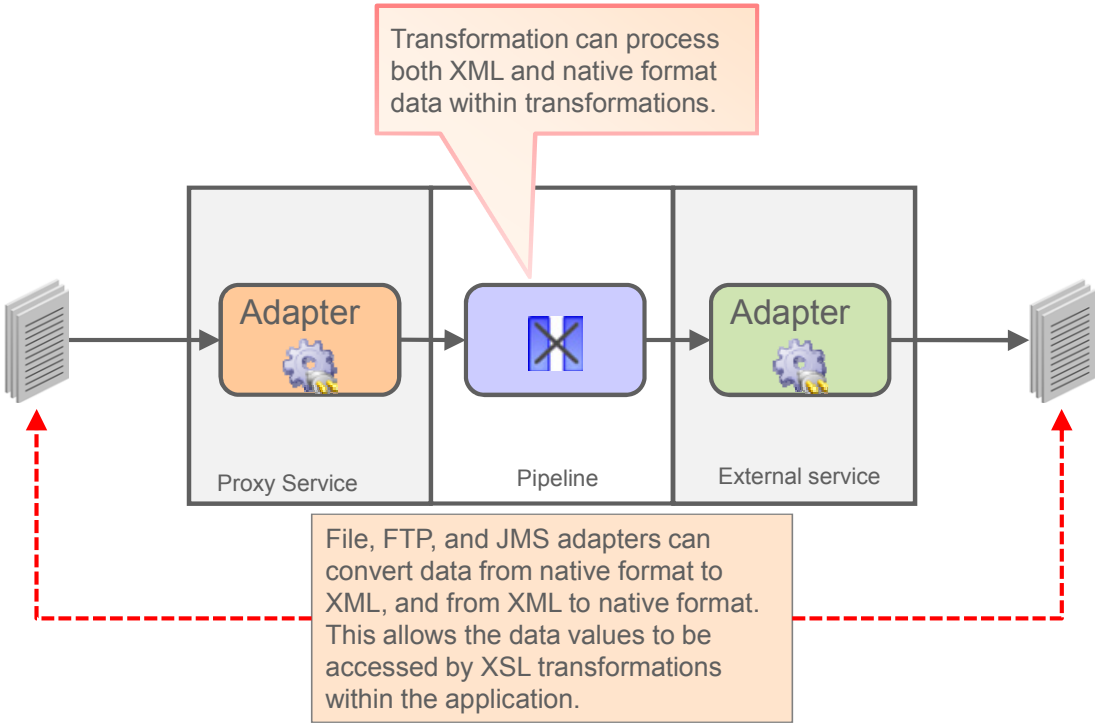
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The Service Bus deals with XML-based data. Data that is not XML based is not structured in the eyes of the Service Bus. Sometimes non-XML-based data should be processed by the Service Bus, and should therefore be treated as structured. Service Bus in SOA Suite 12*c* has a valuable addition that provides the way to address this challenge: the NXSD schema editor and the associated Native Translation activity in Service Bus pipelines.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Working with Native Format Data



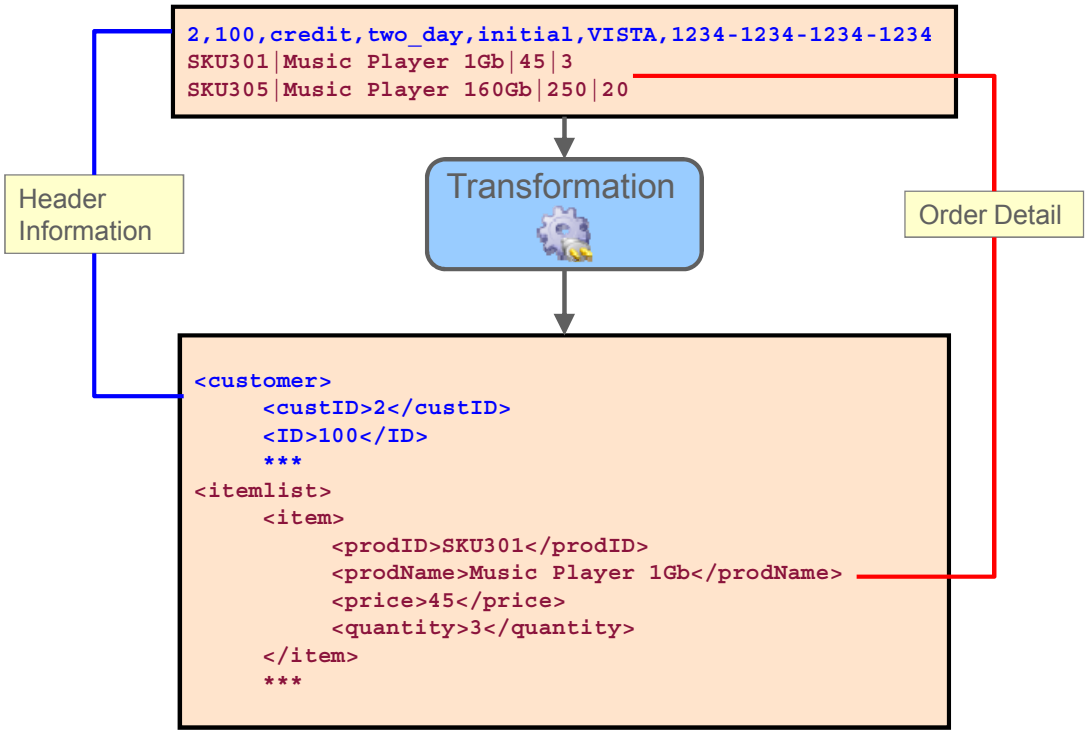
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

File, FTP, and JMS adapters accept native messages in XML or non-XML format and publish them as XML messages. They can also accept XML messages and convert them back to native EIS format. This translation from native data format to XML and back is performed by using a definition file (non-XML schema definition), which itself is defined in the XML schema format. The Native Format Builder wizard enables you to sample native data and create the native XSD (NXSD) grammar for translation of native data.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Native Data Transformation



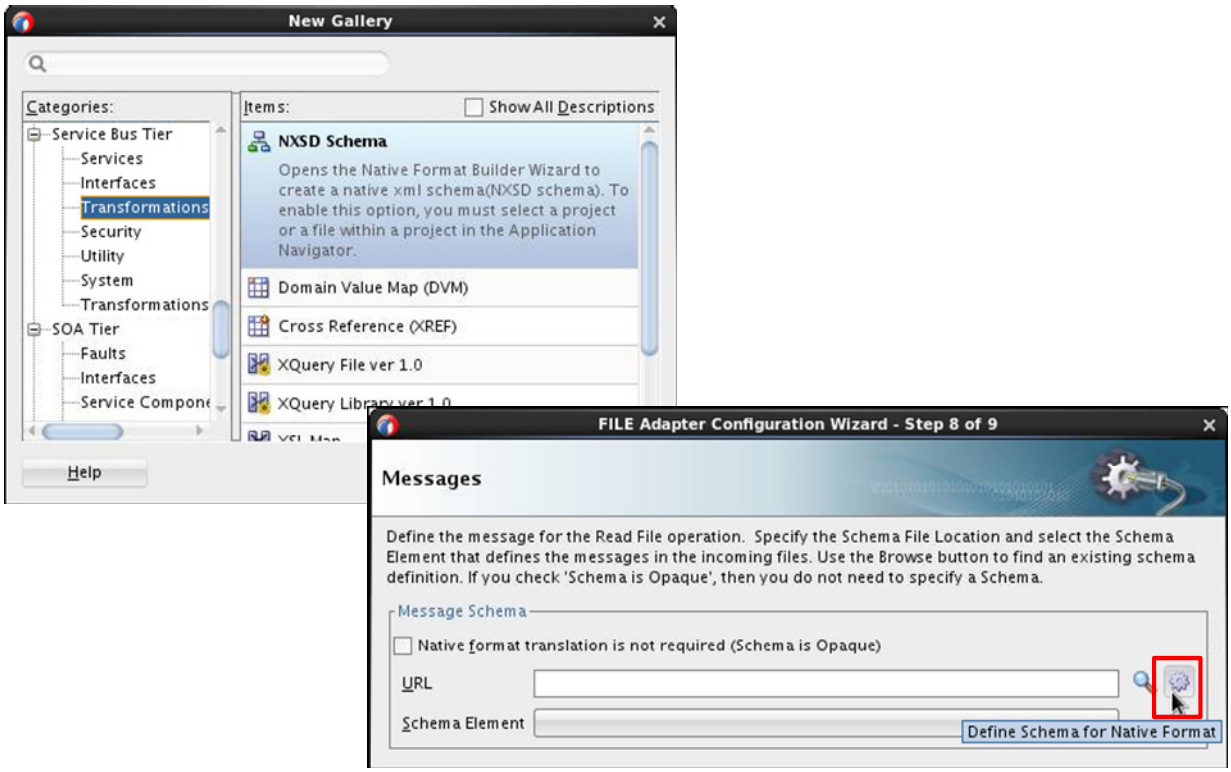
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The example in the slide shows incoming delimited data passed into the adapter, and that data passed out of the adapter in XML format, which is suitable for processing in Oracle Service Bus. An outbound adapter can be configured to perform the opposite transformation, passing XML data out in a delimited (or other) format.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Starting Native Format Builder Wizard



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The Native Format Builder wizard enables you to sample native data and create the nXSD grammar for translation of native data.

You can start the Native Format Builder wizard by:

- Selecting Service Bus Tier > Transformations > NXSD Schema in the New Gallery window

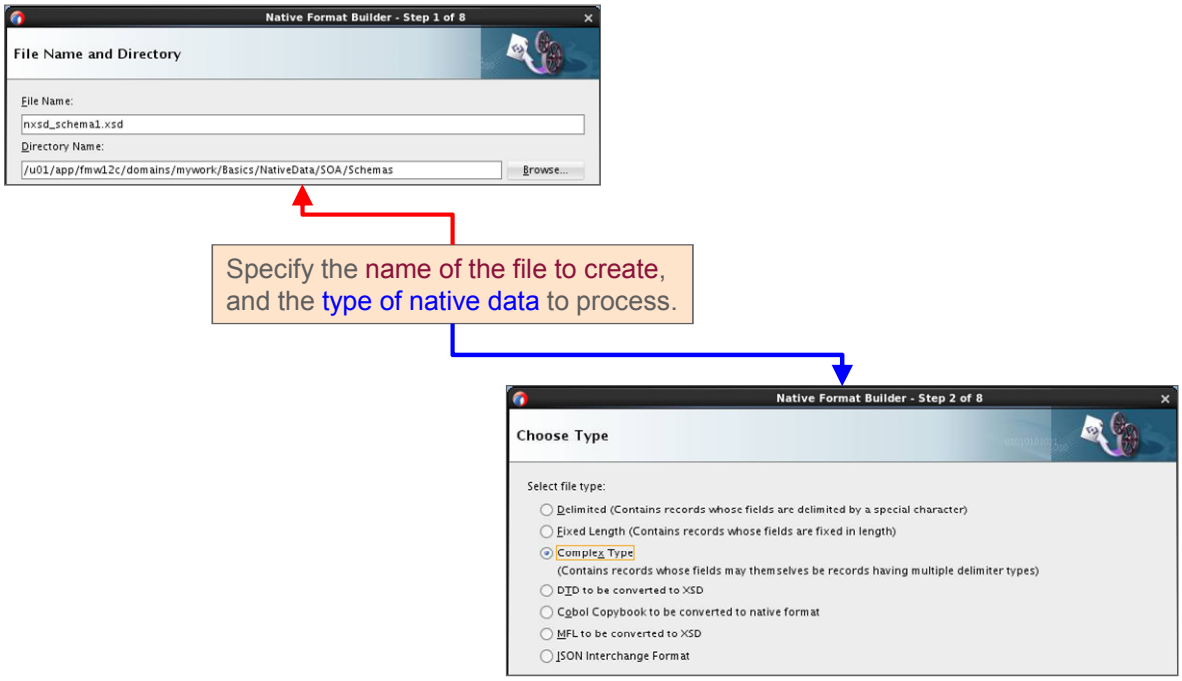
Or

- Clicking the Define Schema for Native Format button in the Messages page of the Adapter Configuration Wizard. The example in the slide shows a File adapter.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Specifying File Name and Native Data Format



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Begin by specifying the name of the file (`.xsd`) that will describe the native format, and by selecting the type of data it will represent.

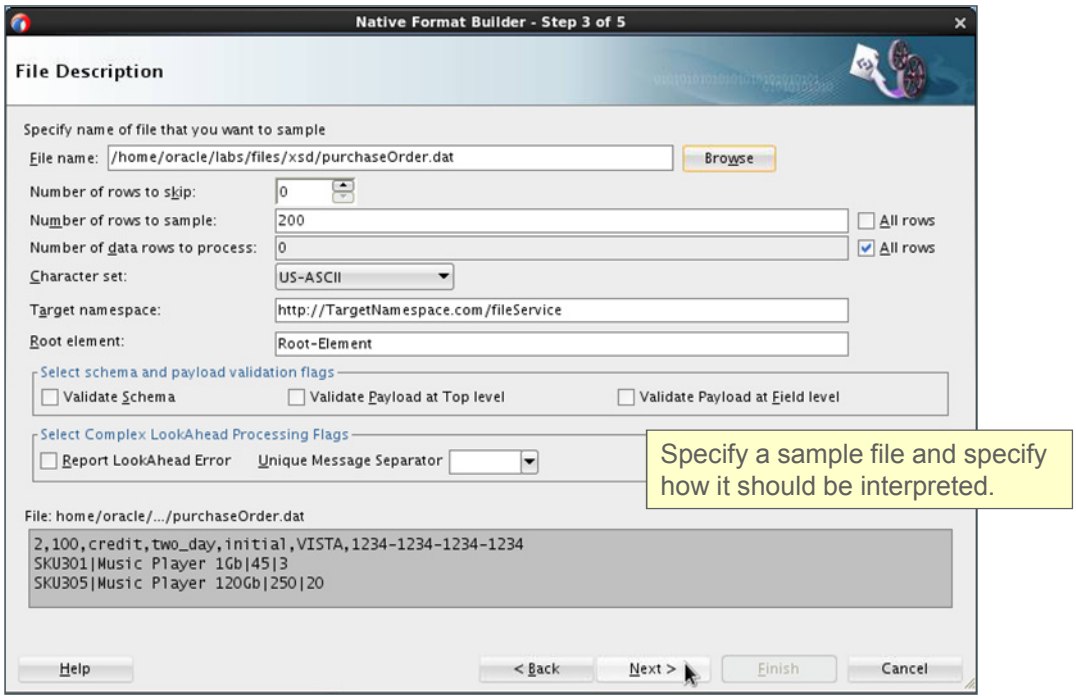
The Native Format Builder wizard supports file formats such as delimited, fixed-length, complex type, data type description (DTD), and COBOL Copybook.

- The *Delimited* option enables you to create native schemas for records, where fields are separated by a value such as a comma or number sign (#).
- The *Fixed Length* (Positional) option enables you to create native schemas for records where all fields are of fixed length.
- The *Complex Type* option enables you to create native schema for records where the fields may themselves be records having multiple delimiter types.
- The *DTD* option enables you to generate native schema from the user-supplied DTD, which contains information about the structure of an XML document.
- The *COBOL Copybook* option enables you to generate native schema from the user-supplied COBOL Copybook definition.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Specifying a Sample File



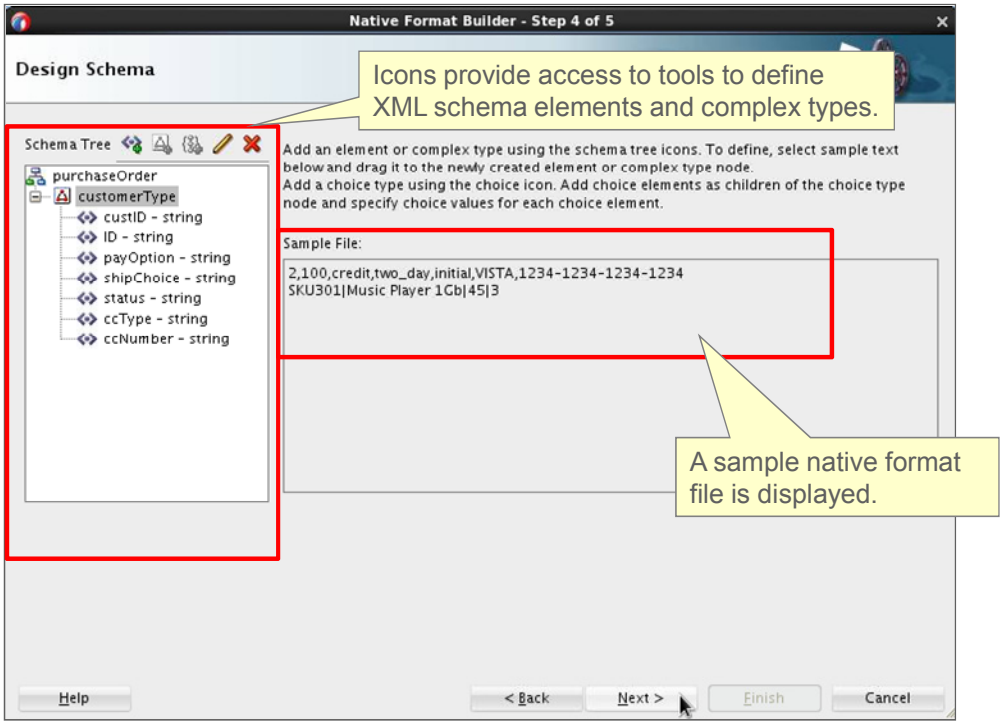
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The Native Format Builder reads a sample native file to initiate the definition process. On the screen shown in the slide, specify the sample file and provide additional instructions on how to interpret the contents of that file.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Defining a Schema for a Native Format



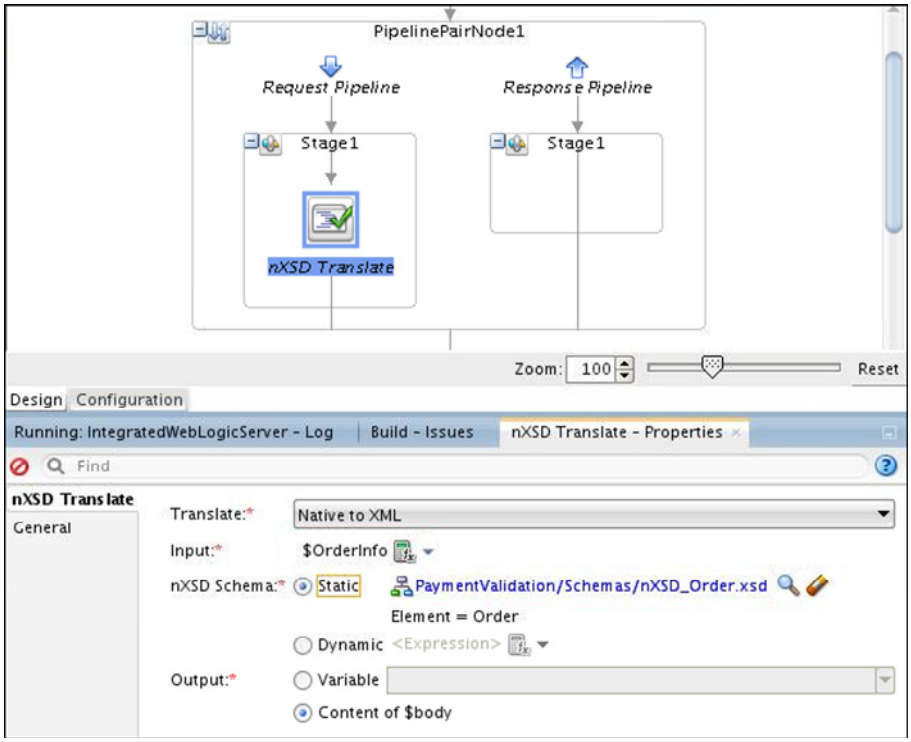
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

After reading the sample file, the wizard provides features to allow you to fully define the resultant XML schema file.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

nXSD Translate Action



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

As shown in the example, an NXSD Translation activity is added to the Request pipeline. This activity uses a Native XSD to map data in native format into the corresponding XML representation.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Summary

In this lesson, you should have learned how to:

- Describe the role of XPath, XQuery, and XSLT in the way Service Bus handles data
- Describe standard and custom XPath functions
- Describe how the XSLT Mapper can be used to create XSL transformations
- Use the XQuery Mapper to create XQuery transformations
- Transform non-XML data to XML data with nXSD

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Practice 6: Overview

- 6-1: Transforming Data Using XQuery Transformation
- 6-2: Configuring a File Adapter Proxy with nXSD Transformation
- Appendix: Building the nXSD Transformation from the Beginning

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

7

Routing Messages

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Objectives

After completing this lesson, you should be able to:

- Describe routing concepts
- Explain how routing works
- Use conditional branching or routing table for content-based routing
- Use XQuery-based policies for dynamic routing

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Routing overview
- Content-based routing
- Dynamic routing

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Message Routing: Overview

- The ability to route a request to a specific service provider based on a static or variable routing criteria
- Types of routing:
  - Static routing
    - Routing action
  - Content-based routing
    - Conditional Branch node
    - Routing Table action
  - Dynamic routing
    - Dynamic Routing action
    - Routing Options action

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

In Service Bus pipelines, you can route a message to different service providers.

- **Static routing:** A routing action identifies a target service for the message and configures how the message is routed to that service.
- **Content-based routing:** Route each message to the correct recipient based on message content. Conditional branching and routing table are the two mechanisms that the content-based routing can be realized.
- **Dynamic routing:** Service Bus also supports two dynamic routing activities: Dynamic Routing action and Routing Options action. The business service to route the message to is determined by using XQuery expression. With dynamic routing, you can add additional endpoints by a simple edit of the XQuery resource.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Routing Use Cases

- Header-based routing
- Routing a message based on:
  - The type of the payload
  - Operation
- Runtime protocol selection

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

With the intelligent routing that OSB provides, you are able to examine the content of a service request and route it to one or more potential recipients based on the value of the data that is being carried, some header information that was placed in the request, or even contextual information that is associated with the process context (if there is a process flow being executed).

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Header-based Routing

Header-based routing is more efficient as payload does not have to be processed.

```
<Soap:Envelope>
<Soap:Header>
  <m:path xmlns:m="http://oracle.example.com/rp/"
   Soap:actor="http://schemas.example.com/soap/actor"
   S:mustUnderstand="1">
    <m:action>http://example1.com/</m:action>
    <m:to>http://example2.com/router</m:to>
    <m:fwd>
       <m:via>http://example3.com/router</m:via>
   </m:fwd>
  </m:path>
</Soap:Header>
<Soap:Body> ... </Soap:Body>
</Soap:Envelope>
```

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Header-based routing is more efficient if payload does not have to be processed. As best practice, pass routing criteria through headers if complex decisions are involved. This way you do not program the business logic into Service Bus. The slide shows an example SOAP message with routing information in header.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

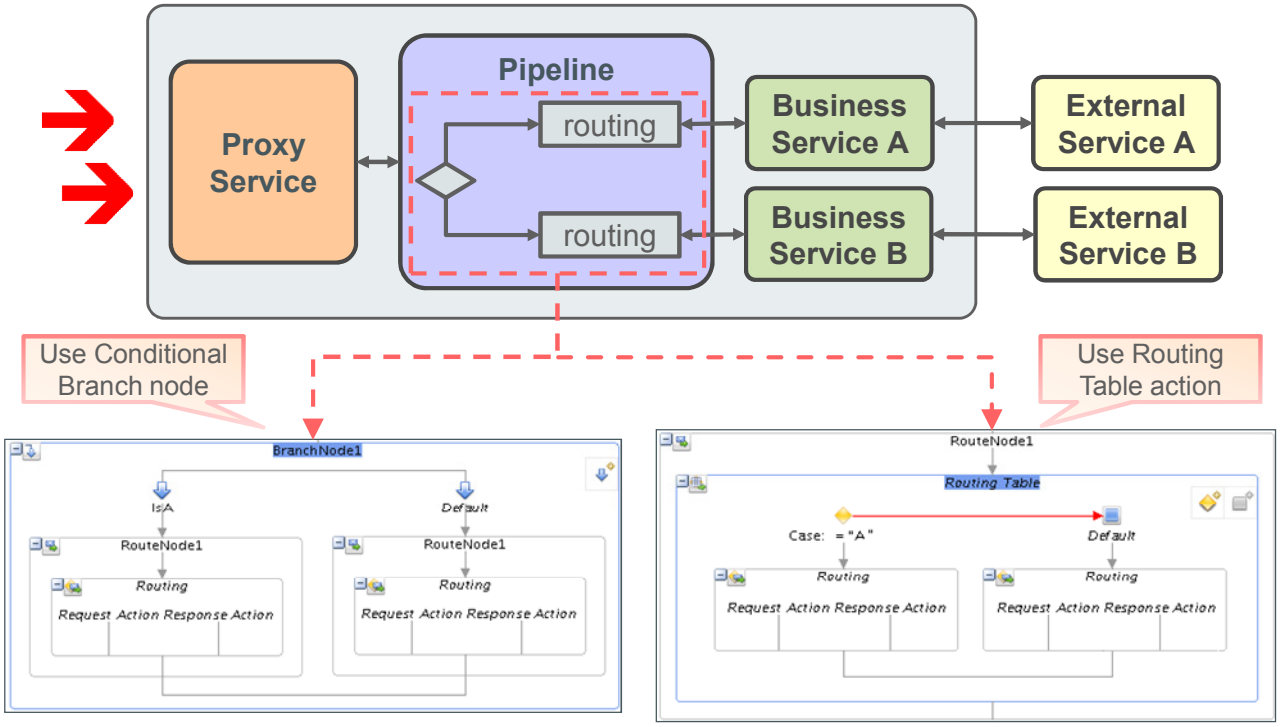
- Routing overview
- Content-based routing
- Dynamic routing

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Content-based Routing



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The content-based routing examines the message content and routes the message onto a different service provider based on data contained in the message. The routing can be based on several criteria such as specific field values. You can implement a routing filter by using conditional branch and routing table.

Conditional branch can be configured with multiple branches, one of which is executed according to an XPath expression. Whichever condition is satisfied first determines which branch is executed. If none of the branch conditions is satisfied, then the default branch is performed.

Routing Table is a similar activity where multiple routing options are configured, one of which is executed depending on the evaluation of an XQuery expression.

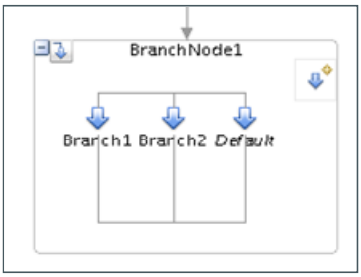
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Conditional Branch Node

The Conditional Branch node allows you to route the incoming requests to different target services based on different conditions including:

- equal to (=)
- not equal to (!=)
- less than (<)
- less than or equal to (<=)
- greater than (>)
- greater than or equal to (>=)



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Conditional branch can have multiple branches, one of which is executed according to an XPath expression. Whichever condition is satisfied first determines which branch is executed. If none of the branch conditions is satisfied, then the default branch is performed.

Conditional branch, configured in a conditional branch node, is driven by a lookup table, with each branch tagged with simple, unique string values (for example, `QuantityEqualToOrLessThan150` and `QuantityMoreThan150`).

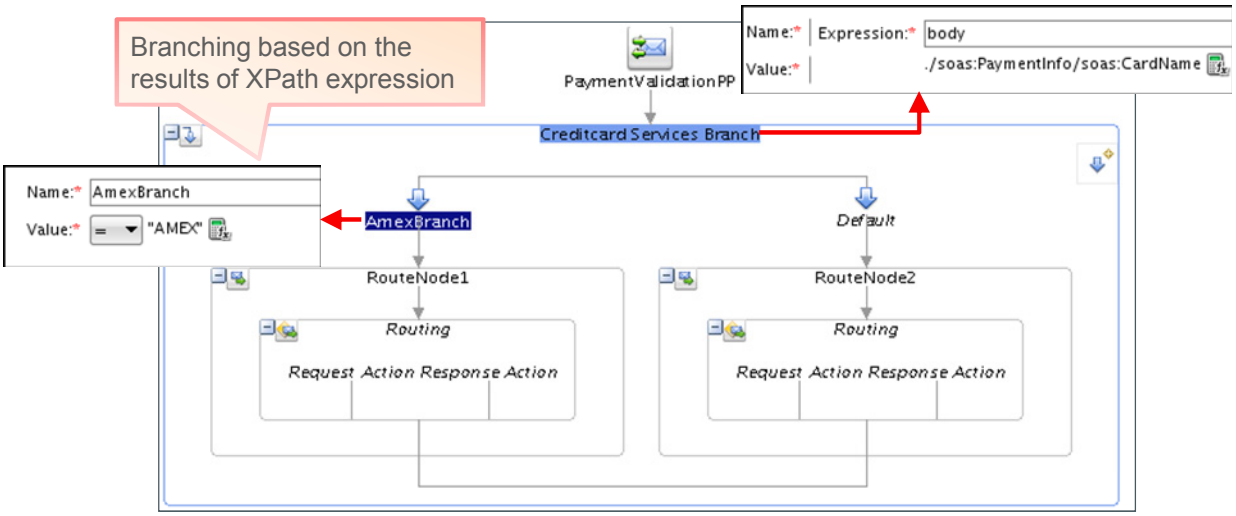
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Configuring Conditional Branch Node

Conditional branching can be based on:

- Variable value in the message context
- Results of an XPath expression



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

You can configure a conditional branch to branch based on the value of a variable in the message context (declared, for example, in a stage earlier in the message flow), or you can configure the condition to branch based on the results of an XPath expression defined in the branch itself, as shown in the example in the slide. At run time, the variable or the expression is evaluated, and the resulting value is used to determine which branch to follow.

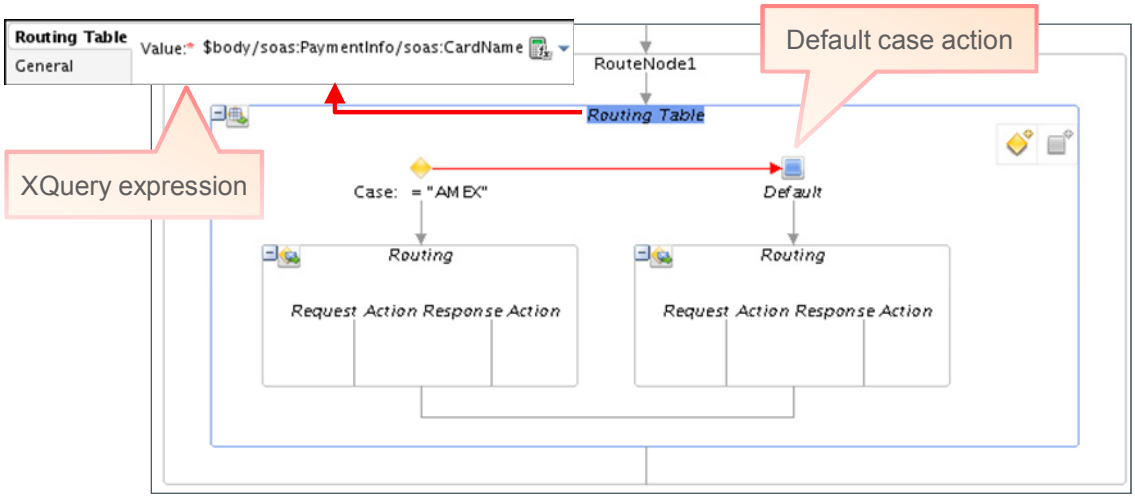
A branch node may have several descendants in the message flow: one for each branch, including the default branch. You should always define a default branch. If no branch matches the value, the default branch is followed. You should design the pipeline service in such a way that the value of a lookup variable is set before reaching the branch node.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Routing Table Action

- A routing table is used to select different routes based on the results of a single XQuery expression in a message flow.
- A routing table action contains a set of routes wrapped in a switch-style condition table.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The routing table action contains a case action with a path to a routing action, in addition to a default case action with a path to a default routing action. When you click the case action, you can set the comparison operator and value options against which the value returned from the XQuery expression defined for the routing table action is evaluated.

In the routing table, use the Expression option to enter a valid XQuery expression, which at run time returns the value on which the routing decision is made.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Conditional Branching Versus Routing Table

- Use conditional branching if the condition that you are branching on is known early in the message flow.
- If your logic is common regardless of the message destination, use routing table for routing decision-making until the business service invocation.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Use conditional branching to expose the routing alternatives for the message flow at the top-level flow view. Consider a situation where you want to invoke service A or service B based on a condition known early in the message flow (for example, the message type). You could configure conditional branching in a routing table in the route node. However, this makes the branching somewhat more difficult to follow if you are just looking at the top level of the flow. Instead, you could use a conditional branch node to expose this branching in the message flow itself and use simple route nodes as the subflows for each of the branches.

The idea of routing table is to abstract the pipeline logic from the routing logic. If your logic is common regardless of the message destination, delay the decision up until the business service invocation. If there are no common logic, you can branch your pipeline right away by using conditional branch.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Quiz

What do you use to select different routes based on the results of a single XQuery expression?

a. Dynamic routing
b. Routing table
c. Route node
d. None of the above

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Answer: b

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Routing overview
- Content-based routing
- Dynamic routing

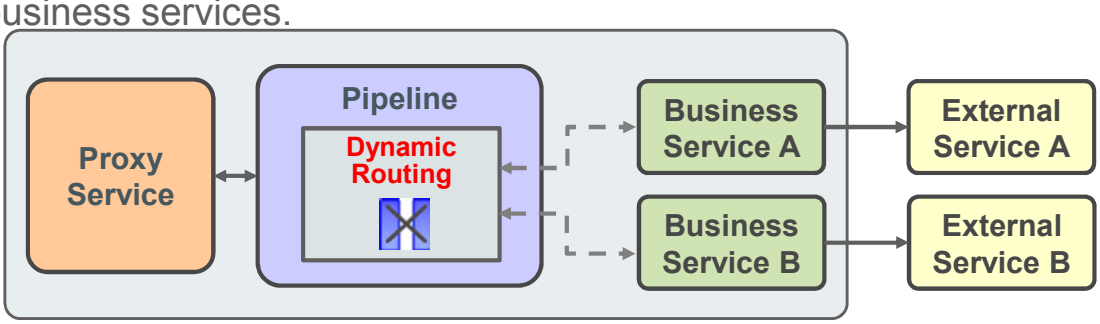
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

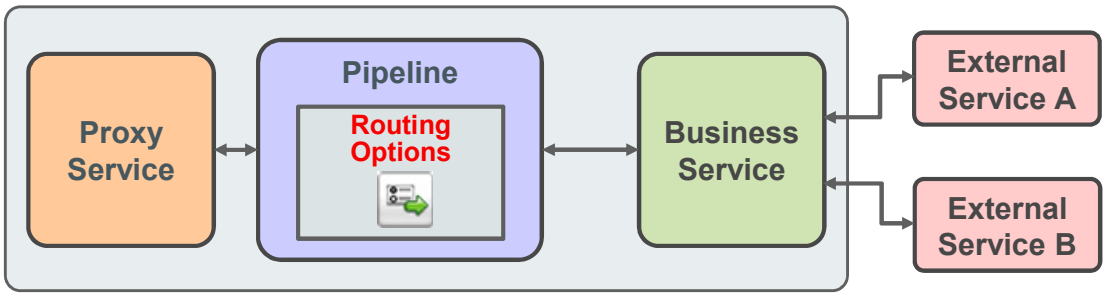
THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Dynamic Routing

- Use a Dynamic Routing action to dynamically invoke different business services.



- Use a Routing Options action to dynamically set the endpoint URI used on the business service.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Dynamic routing provides another layer of abstraction from the client to the service bus by separating the control logic, which determines the route taken by the process, from the execution of the process. Dynamic routing enables the request to be routed to the appropriate endpoint, not defined during design time.

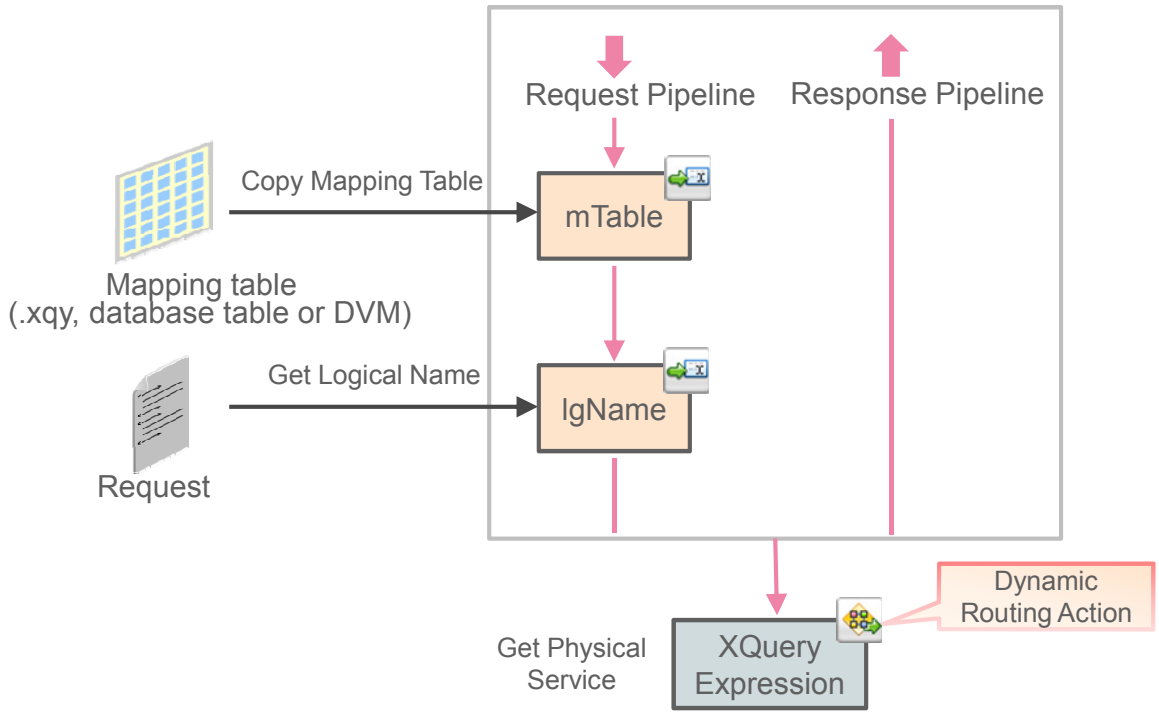
There are two techniques that can be used with dynamic routing:

- Use XQuery of a Dynamic Routing action to dynamically set the fully qualified business service name.
- Set the endpoint URI used on the business service dynamically through a Routing Options action. In this case, all external services should have the same service interface.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Dynamic Routing with Dynamic Routing Action by Example



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

In a message flow pipeline, design an XQuery expression to dynamically set the fully qualified service name in OSB and use the Dynamic Route or Dynamic Publish actions.

At run time, you provide the entry in the mapping table that will determine the routing or publishing destination of the proxy service.

The XML file or the XQuery resource contains a mapping table that maps a logical identifier (such as the name of a company) to the physical identifier (the fully qualified name of the service in OSB). An XML file or the XQuery resource can be maintained easily. You can also use a database table or Domain Value Maps (DVM) for mapping.

The logical identifier, which is extracted from the message, maps to the physical identifier, which is the name of the service that you want to invoke.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Mapping Table Used by Dynamic Routing Action: Example

mapping.xqy

```
<map>
   <row>
      <logical>Oracle</logical>
      <physical>corpRentalcar/goldserviceBS</physical>
   </row>
   <row>
      <logical>ABC Corp</logical>
      <physical>corpRentalcar/silverserviceBS</physical>
   </row>
</map>
```

Project name

Business service

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

This slide shows a sample mapping file. Map each logical name that is used, but map a client service to the actual service that should be called.

When implementing a dynamic routing action, you assign the mapping XML table in the XQuery resource to a variable.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XQuery Expression for Dynamic Routing Action

Invoke a business service.

```
<ctx:route>
    <ctx:service
      isProxy='false'>{$Projectname/$BizService}
    </ctx:service>
    <ctx:operation>{$operation}</ctx:operation>
</ctx:route>
```

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

You implement a query against the variable in the routing table to extract the physical identifier based on the corresponding logical identifier. Using this variable, you can invoke the required service.

If a proxy service is being invoked, the `isProxy` attribute should be set to `true`. The service name is the fully qualified service name. The `operation` element is optional.

For example:

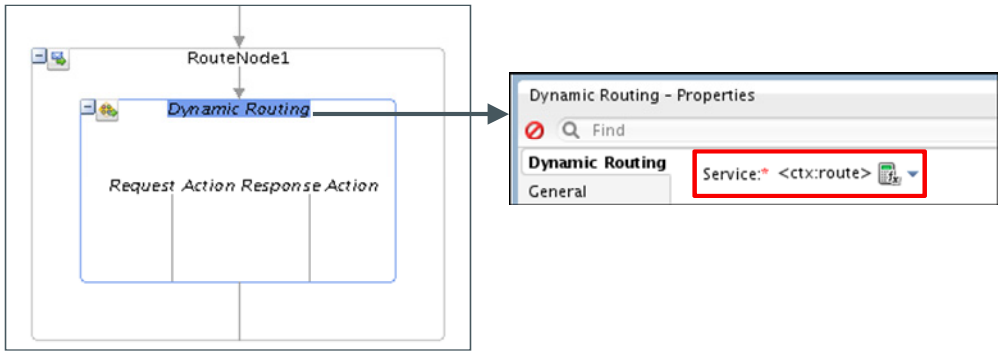
```
<ctx:route>
    <ctx:service
isProxy='false'>{$rTable/row[logical/text()=$lgName]/physical/text()
}</ctx:service>
    <ctx:operation>{$operation}</ctx:operation>
</ctx:route>
```

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Configuring Dynamic Routing Action

In a message flow, use a dynamic routing action to assign a route for a message based on the routing information available in an XQuery resource.



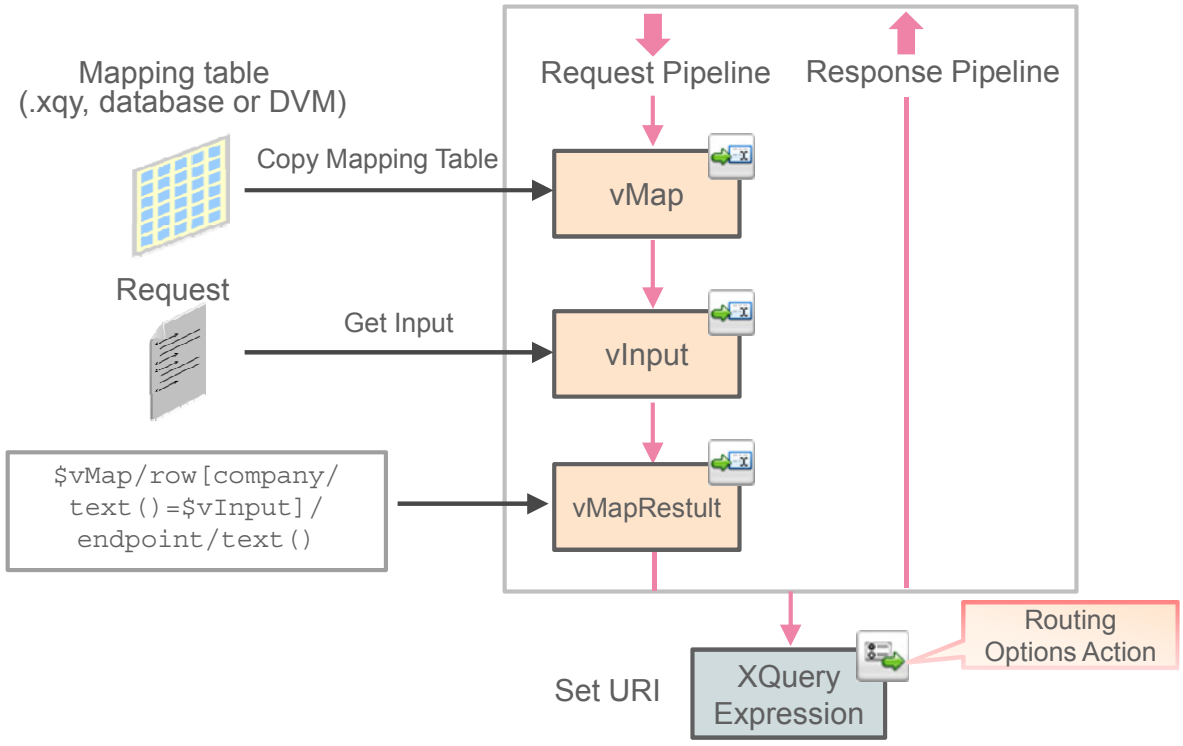
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

An XQuery expression defines the route to be taken by a message. The graphic in the slide shows the creation of a dynamic routing action in JDev.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Dynamic Routing with Routing Options Action by Example



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Routing Options Action is used when the overview of the interface is fixed. The overview of the interface includes message types, port types, and binding; it excludes the concrete interface. The concrete interface is the transport URL at which the service is located.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Mapping Table Used by Routing Options Action: Example

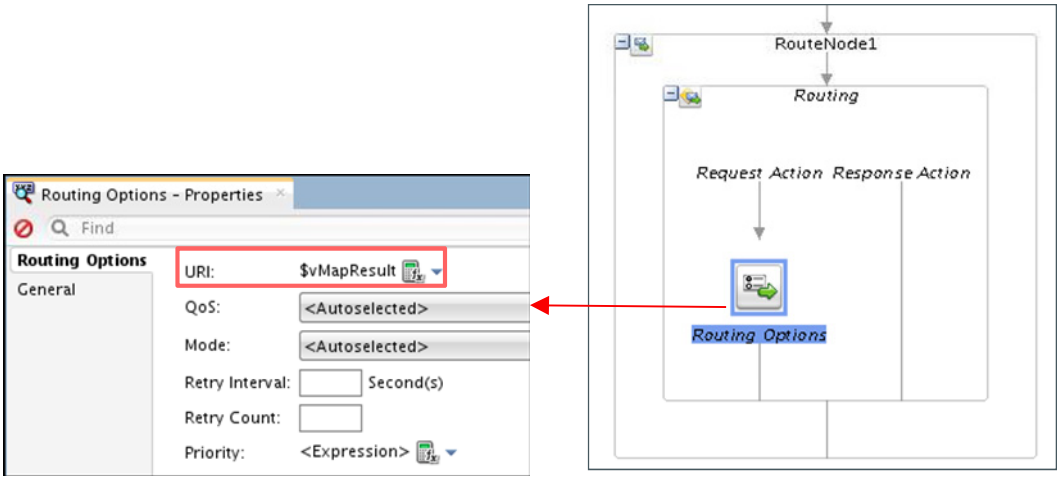
```
<map>
   <row>
      <company>Oracle</company>
      <endpoint>http://corpRentalcar.com:8888/goldservice</endpoint>
   </row>
   <row>
      <company>ABC Corp</company>
       <endpoint>http://corpRentalcar.com:9999/silverservice</endpoint>
   </row>
</map>
```

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Configuring Routing Options Action



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Configure a pipeline service to route or publish messages to a business service. In the request actions section for the route action or publish action, add a routing options action that dynamically specifies the URI of a service.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Quiz

Which action can be used to change the endpoint URI of the external service when you invoke using business service?

a. Routing
b. Routing Table
c. Dynamic Routing
d. Routing Options

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Answer: d

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Summary

In this lesson, you should have learned how to:

- Describe routing concepts
- Explain how routing works
- Use conditional branching or routing table for content-based routing
- Use XQuery-based policies for dynamic routing

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Practice 7: Overview

- 7-1: Static Routing Messages Using Conditional Branching
- 7-2: Dynamic Routing Messages Using XQuery-based Policy

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

8

Enriching Messages

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Objectives

After completing this lesson, you should be able to:

- Describe the message enrichment options
- Use service callout to enrich a request message before routing it to the target service
- Describe the usage of Java callout in OSB

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

This lesson talks about how to add additional data or convert the existing data to the outgoing messages in OSB.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Message Enrichment Options

- Service Callout
- Java Callout

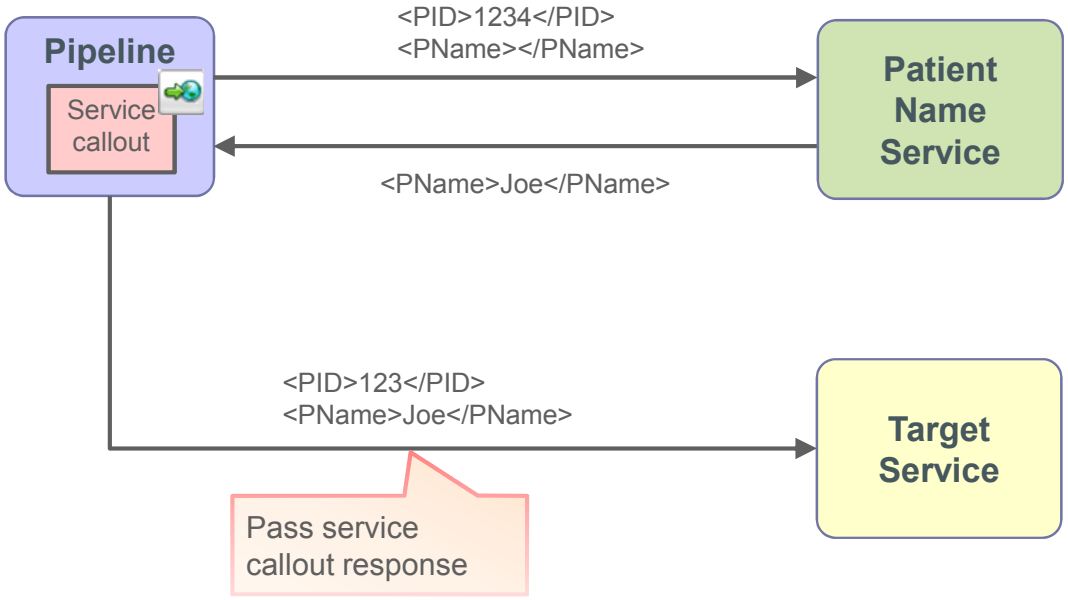
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Service Callout

Enables you to query another service for a value that can then be used in the message flow



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

A service callout action configures a synchronous (blocking) callout to a proxy, pipeline, or business service that is registered in Oracle Service Bus. The service callout action is used in response to decisions made in complex dynamic routing processing; it is used to perform message enrichment.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Using Service Callout

- Used for real-time request/response calls (Synchronous calls)
- Multiple service callouts can be executed in any order
- A response is needed from the service execution.
  - When responses are not needed, use Publish action.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Service callout is a service invocation to validate, modify, augment or enrich the incoming request or outgoing response within a pipeline execution.

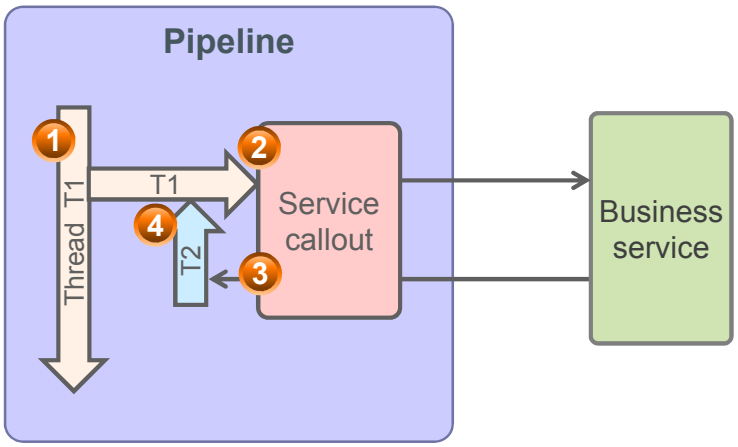
Service callout is used in real-time request-response scenarios for synchronous response. Being a synchronous call, the thread gets blocked until a response is returned from target service.

Multiple service callouts can be executed in any order, which enables more complex message flows. A response is expected from the service execution of a service callout, so caller of the service callout will block till a response is available. If responses are not needed or its strictly one-way communication, Publish Action can be used.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Service Callout Action



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The diagram depicts how service callout action works in a message flow. The service callout action requires two threads: one (T1) for waiting and one (T2) for notification.

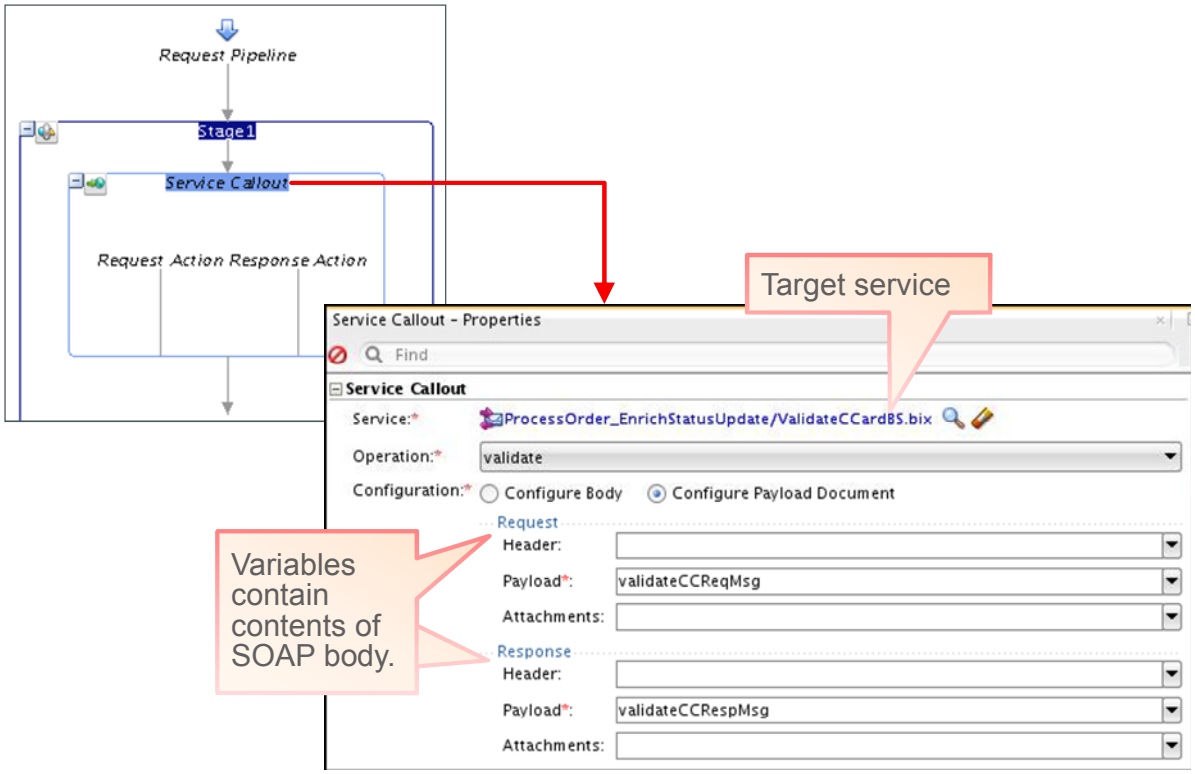
1. T1 executes pipeline logic.
2. T1 makes outbound service callout and waits for a notification of response for that invocation.
3. T2 receives response from the target service via service callout.
4. T2 notifies the waiting thread, T1.
5. T1 then continues logic execution with the response data.

Because Service Bus will hold a thread and not continue until the service callout completes, this can tie up resources and degrade performance under heavy loads.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Configuring Service Callout Action



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Service callout pipeline actions allow you to make a callout from the pipeline to another service.

The service callout action is used in a message flow routing stage to call on the destination service to perform some action on the message. The destination service returns a response to the message flow, which is assigned to a local variable. The variable may be used within the current message flow for conditional branching. Service callouts allow custom Java code to be invoked from within pipeline.

Service callout action options are:

- **Service:** Target service for the service callout action
- Configure Soap Body or Configure Payload Parameters

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Constructing Service Callout Messages

- The service options that can be selected for a service callout action are:
  - SOAP Document Style Service Callout
  - SOAP RPC Style Service Callout
  - XML Service Callouts
  - Messaging Service Callouts
- You can use attachments with service callout messages.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When Service Bus makes a call to a service using a service callout action, the content of the message is constructed using the values of variables in the message context. The message content for outbound messages is handled differently depending upon the type of the target service.

SOAP Document Style Service Callout: Messages for SOAP Document Style services (including EJB document and document-wrapped services) can be constructed as follows:

- The variable assigned for the request document contains the SOAP body.
- The variable assigned for the SOAP request header contains the SOAP header.
- The response must be a single XML document; it is the content of the SOAP body plus the SOAP header (if specified).

SOAP RPC Style Service Callout: Messages for SOAP RPC Style services (including EJB RPC services) can be constructed as follows:

- Request messages are assembled from message context variables.
- Response messages are assembled as follows:
  - The output content is the content of the SOAP header (if a SOAP header is specified).
  - Part as element: The output content is the child element of the parameter; there is at most one child element.
  - Part as simple/complex type: The output content is the parameter itself.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

XML Service Callouts: Messages for XML services can be constructed as follows: The request message is the content of the variable assigned for the request document. The content of the request variable must be a single XML document. The output document is the response message.

Messaging Service Callouts (for messaging services):

- The request message is the content of the request variable. The content can be simple text, XML, or binary data represented by an instance of `<binary-content ref=.../>` reference XML.
- The response messages are treated as binary, so the response variable will contain an instance of `<binary-content ref= ... />` reference XML, regardless of the actual content received.

You can specify an optional variable to hold attachments for the outbound request and another variable to hold attachments from the outbound response. The Request Attachments and Response Attachments variables are used to specify the request and response attachments, respectively.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Java Callout

- Java Callout activity is used to invoke Java custom code when the OSB predefined functions are not sufficient to complete the requirement.
- The Java code must be packaged as jar.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The Java callout action lets you access the methods in a Java archive (JAR) file to add processing logic to your pipelines and split-joins. When you configure the callout, you can specify arguments for the method and optionally specify a service account for security. The parameters can be mapped to message context variables. Static methods can be accessed from any POJO. You can also use Java callouts to create Java objects to store in the pipeline and to pass Java objects as parameters to other Java callouts.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Use Cases

- Performing custom validations, such as validating against a DTD, or doing cross-field semantic validation in Java
- Performing custom transformations, such as converting a binary document to base64Binary (or vice versa) or using a custom Java transformation class
- Performing custom authentication and authorizations
- Performing lookups for message enrichment. For example, a file or Java table can be used to look up any piece of data that can enrich a message.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The use cases list some of the scenarios in which you can use Java callouts in Service Bus.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Java Callout Guidelines

- The method written for Java callout must be static method.
- You can specify arguments for the method.
- Java callout is not recommended to call big methods with lots of business functionality behind it.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

- Invoking methods from Java callout in Service Bus projects should be **`public`** and **`static`**.
- When you configure the callout, you can specify arguments for the method.

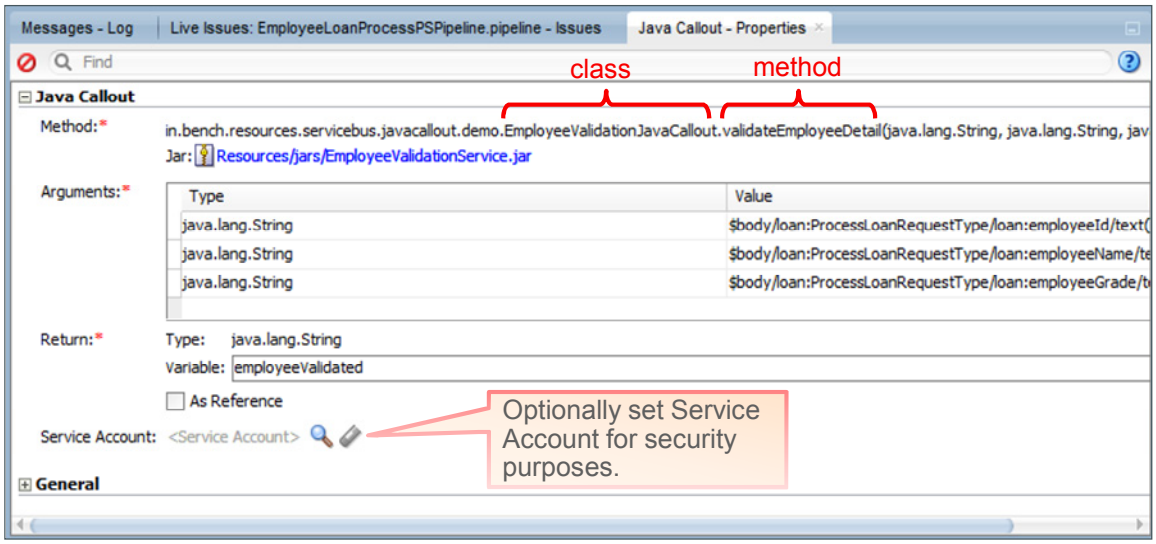
  The following data types are supported for input parameters:
  - Boolean, byte, char, double, float, int, long, short, and arrays of these types
  - Java.lang.[Boolean | Byte | Character | Double | Float | Integer | Long | Short | String] and arrays of these types
  - Java.math.[BigInteger | BigDecimal] and arrays of these types
  - Org.apache.xmlbeans.XmlObject and arrays of this type

  The following data types are supported for method return:
  - All types supported for input parameters except their array equivalent
  - Void
- In general, it is recommended that the JARs are small and simple. Big and complicated methods will make tracing, error handling, and managing transaction much more difficult and has no transparency.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Java Callout Action



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The example shows validating an employee details by invoking a Java class `EmployeeValidationJavaCallout` with one method `validateEmployeeDetail` that is packaged and stored as JAR in the Service Bus project.

You can optionally specify a service account for security purposes.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Summary

In this lesson, you should have learned how to:

- Describe the message enrichment options
- Use service callout to enrich a request message before routing it to the target service
- Describe the usage of Java callout in OSB

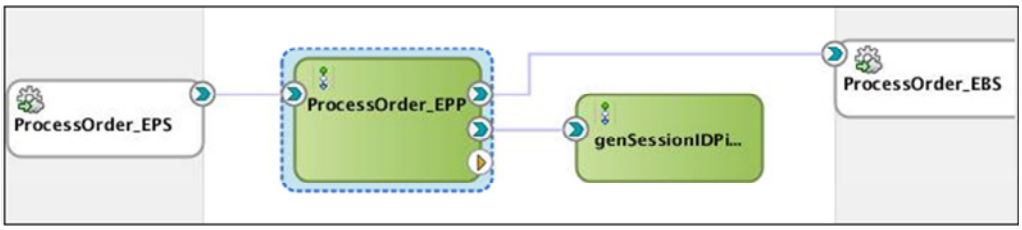
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

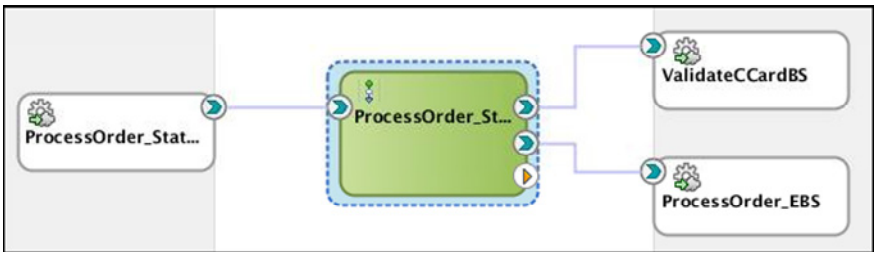
THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Practice 8: Overview

- 8-1: Enriching the Message by Calling a Pipeline Service (Session ID Generation)



- 8-2: Enriching the Message by Calling a Business Service (ValidateCreditCard)



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

9

Processing Messages with Concurrent Calls

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Objectives

After completing this lesson, you should be able to:

- Describe how Split-Join is used to split and aggregate messages
- Explain the differences between static Split-Join and dynamic Split-Join
- Process messages by using Split-Join

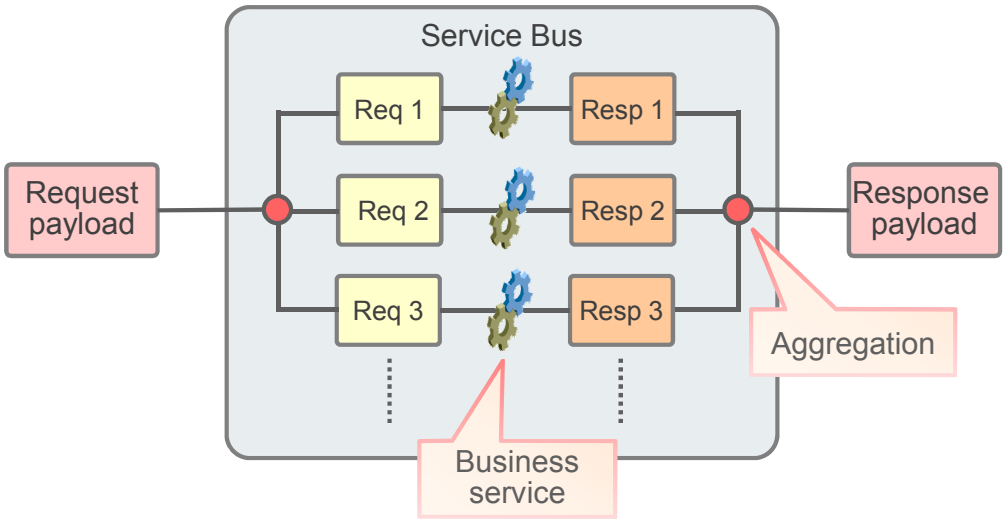
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Split-Join

- Incoming message is split into multiple service invocation and handled in parallel.
- Multiple service responses are aggregated into single payload.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Split-Join allows you to split request payload so that you can call multiple services concurrently as opposed to sequential processing. This way, you will achieve improved service performance. Split-Join can only be created based on WSDL.

Split-Joins are particularly useful for optimizing overall response times in scenarios where payloads delivered by faster systems are directed to responding services on slower systems. Without Split-Join, individual messages in a payload are normally resolved in sequential order by the recipient, which can take a long time if the responding system is slow. The overall response time is the sum of the individual response times for each message.

With Split-Join, multiple messages are processed simultaneously, which reduces the burden on the responding system and greatly enhances response times. The overall response time is roughly that of the longest individual message's response time plus some minor system overhead.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Split-Join Patterns

- Static Split-Join: Fixed number of parallel branches that are determined at design level
  - Example: Cable package includes Internet service, TV service, and telephone service.
- Dynamic Split-Join: Handles a variable number of requests
  - Example: The retailer places a batch order containing a variable number of individual purchase orders.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

A static split-join branches from the main execution thread of a Service Bus message flow by splitting a payload into a fixed number of new branches according to the configuration of the split-join. At design time you determine the number and variety of services to be invoked. For instance, a customer places an order for a cable package that includes three separate services: Internet service, TV service, and telephone service. In the Static pattern, you execute all three requests in separate parallel branches to improve the performance time over that of a standard sequential execution.

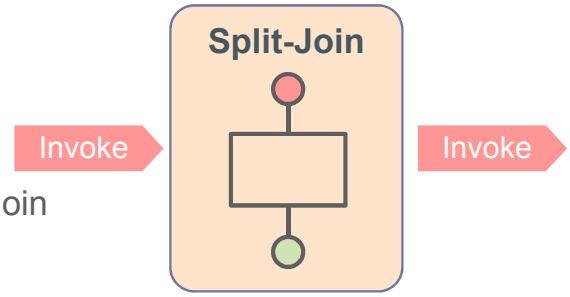
A dynamic split-join branches from the main execution thread of a Service Bus pipeline by dynamically creating new branches according to the contents of the incoming payload. The number of message requests created is variable. For example, a retailer places a batch order containing a variable number of individual purchase orders. In the Dynamic pattern, you parse the batch order and create a separate message request for each purchase. As in the Static pattern, these messages can then be executed in parallel for improved performance.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Split-Join Invocation

- Proxy service
- Pipeline
- Another split-join



- Proxy service
- Business service
- Pipeline
- Another split-join

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

A split-join can be invoked from a proxy service, a pipeline, or another split-join. A split-join can invoke a proxy or business service, a pipeline, or another split-join.

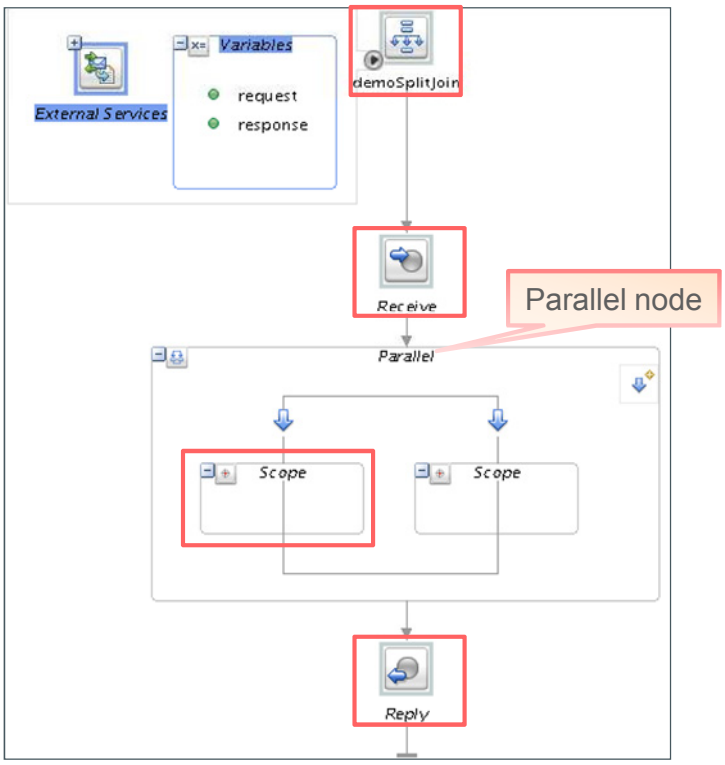
A split-join can invoke another split-join in the same Service Bus configuration, providing more flexibility in service design by letting you split up complex split-join functionality into multiple split-joins. This allows for componentization and reuse of split-join functionality. Performance is maintained, because there is no marshalling and unmarshalling of data between the split-joins.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Split-Join Construction: Static Split-Join

- WSDL based
- JDeveloper only



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The following primary elements are used to construct a message flow in a static split-join:

- A start node, which contains the request and response variables introspected from the WSDL operation
- A receive node, to receive incoming request messages
- A reply node, to send response messages
- A parallel node, which is a placeholder for a fixed number of processing branches, each with its own scope. Two branches are generated by default.
- A scope, which is a container that creates a context that influences the behavior of its enclosed elements

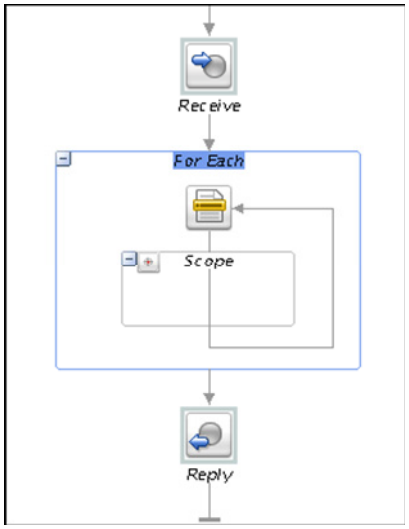
Note that you can only create and configure split-joins in JDeveloper. JDeveloper provides a graphical modeling editor so you can easily model your split-joins by dragging operations to the canvas, and configuring their properties in the Properties window.

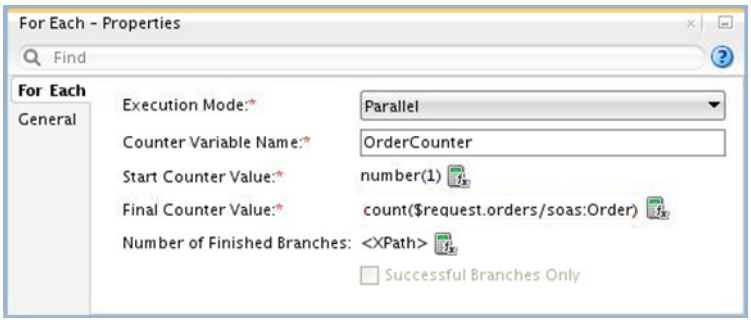
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Split-Join Construction: Dynamic Split-Join

A dynamic split-join uses conditional logic to determine the number of branches to create.





Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

A dynamic split-join uses conditional logic to determine the number of branches to create. All requests are handled simultaneously, and the responses are aggregated into a single reply.

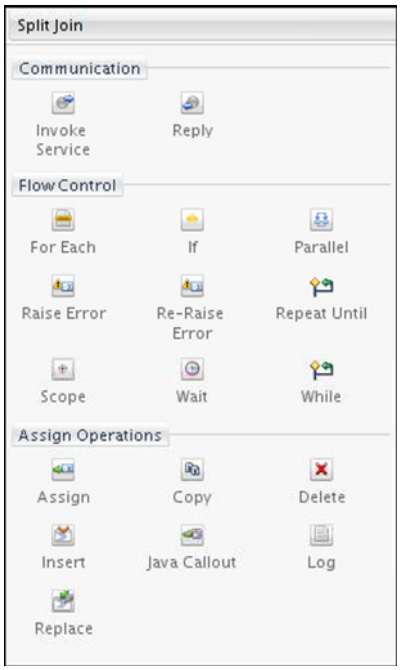
You can use the for-each flow control to iterate over a sequence of values and execute a block of actions. As shown in the example, for-each loops through all the orders and places them in parallel. In the properties of for-each, you need to define the counter variable name, which starts in this case with 1 and ends with the count of the order elements.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Split-Join Operations

- Communication operations define how the split-join interacts with external services.
- Flow control operations define how incoming messages flow through the split-join.
- Assign operations let you manipulate the data in the message you process, including initializing and updating a variable.



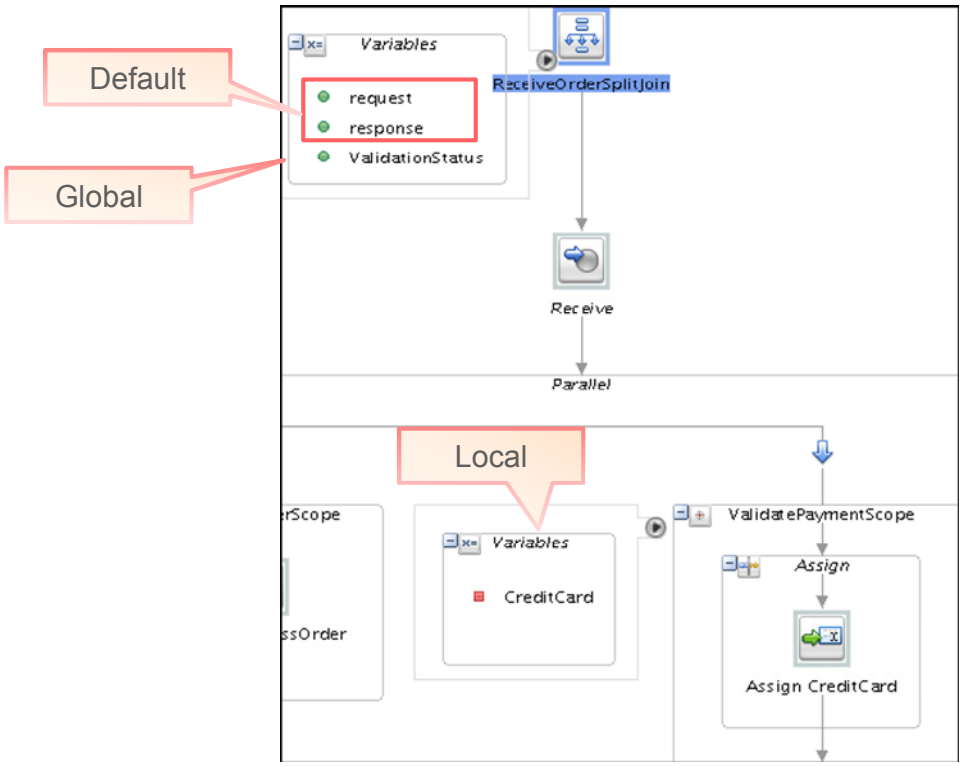
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The Split-Join Components window lists all the operations you can use to construct a split-join. The operations are divided into the following categories: Communication, Flow Control, and Assign Operations.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Global and Local Variables



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Variables in the Start Node store data that can be referenced globally, that is by any node in the split-join. By default, every Start Node is assigned both a request and a response variable when the split-join is initially created. From the Start Node, you can either create a new global variable or edit an existing global variable. Variables in a Scope store data that can be referenced locally, that is only by the node to which it is attached. From a Scope, you can create new local variables or edit an existing local variable.

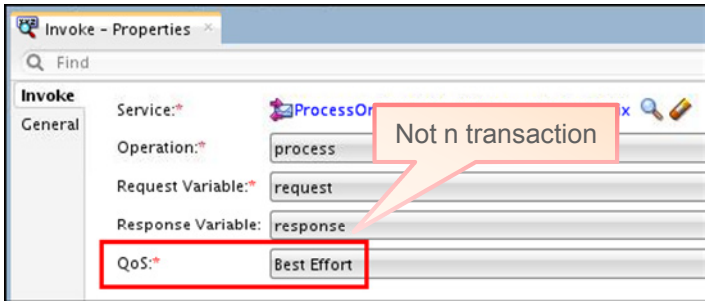
The Request and Response variables in Start Node were determined by the WSDL operation initially selected.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Transaction Support

- Split-join operations provide an option for setting specific quality of service (QoS) values that control transaction support:
  - Invoke Service
  - Assign
  - Delete
  - Insert
  - Java Callout
  - Replace



- Operations set with a QoS of:
  - *Exactly Once* are executed in the transaction.
  - *Best Effort* do not execute in the context of a transaction.
- Split-joins do not handle transaction rollback for exceptions.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Split-joins provide support for propagating transactions. Many split-join operations provide an option for setting specific quality of service (QoS) values, which control transaction support. The QoS value of Exactly Once on a split-join operation ensures the operation executes in the context of a transaction if one exists.

Setting QoS values on individual operations gives you the flexibility to execute multiple operations in the context of a transaction and execute other operations outside of a transaction in a single split-join. Operations set with a QoS of Exactly Once are executed in the transaction. Operations set with a QoS of Best Effort do not execute in the context of a transaction.

Split-joins do not handle transaction rollback for exceptions. It is the responsibility of the service component that called the split-join to handle transaction exceptions and rollback.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Summary

In this lesson, you should have learned how to:

- Describe how Split-Join is used to split and aggregate messages
- Explain the differences between static Split-Join and dynamic Split-Join
- Process messages by using Split-Join

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Practice 9: Overview

9-1: Processing Messages with Static Split-Join

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

10

Adapters and Transports

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Objectives

After completing this lesson, you should be able to:

- Describe JCA transport and adapters
- Describe SOA-DIRECT transport
- Call SOA services using SOA-DIRECT protocol
- Expose existing SOAP services as REST services using the REST binding

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- JCA transport and adapters
- SOA-DIRECT transport
- REST binding

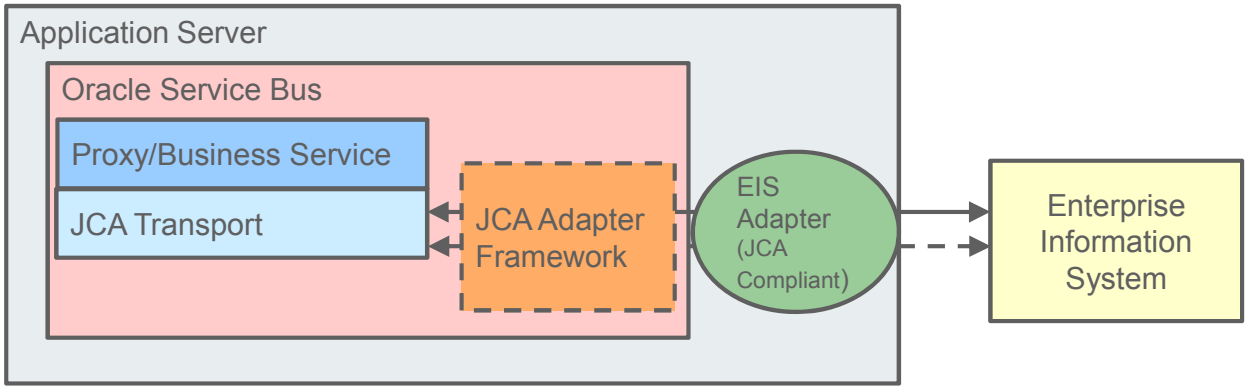
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

JCA Transport

- Provides native connectivity between Service Bus and external systems
- Uses JCA adapter framework to interact with JCA-compliant adapters that, in turn, provide connectivity to external EIS services



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The JCA transport provides native connectivity between Oracle Service Bus and EIS systems, letting those systems interact in the service bus layer and leverage the capabilities and features of Oracle Service Bus.

In JCA proxy or business services, the JCA transport works in conjunction with a built-in JCA adapter framework and JCA-compliant adapters to interact with EIS systems.

JCA proxy services listen for inbound requests from supported JCA adapters, and JCA business services invoke EIS endpoints through supported adapters.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

JCA Adapter Framework

- The JCA transport uses the Oracle Service Bus JCA adapter framework to interact with JCA-compliant adapters that, in turn, provide connectivity to external EIS services.
- The JCA adapter framework abstracts the complexity of interacting with those adapters, enabling you to focus on proxy and business service development in Oracle Service Bus.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

For inbound interactions, the JCA proxy service registers a listener with the associated JCA adapter endpoint. When an event occurs in the EIS system, where a message is sent to the JCA proxy, the JCA adapter framework invokes the proxy service with a request-only or request-response pattern.

On outbound interactions, when a client invokes an EIS service through Oracle Service Bus, the JCA business service invokes the JCA adapter endpoint through the JCA adapter framework.

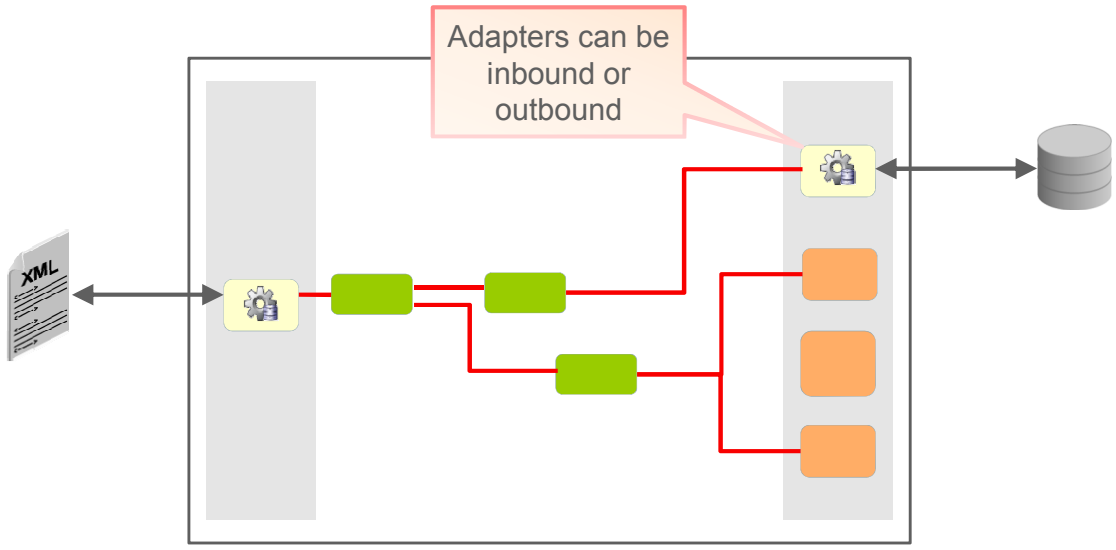
No configuration of the JCA adapter framework is necessary. It is deployed and functions automatically as you create JCA proxy and business services in Oracle Service Bus and deploy your adapters as required.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

JCA Adapters

Adapters integrate with existing back-end applications through Java EE Connector Architecture standards.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Adapters provide a service interface that exposes external application functionality in a form that can be used by the services. An adapter converts requests and responses into a form that is suitable for other (external) systems.

Oracle Service Bus 12*c* is shipped with a variety of adapters that are commonly used to access functionality that is not normally available in a service-oriented context, such as:

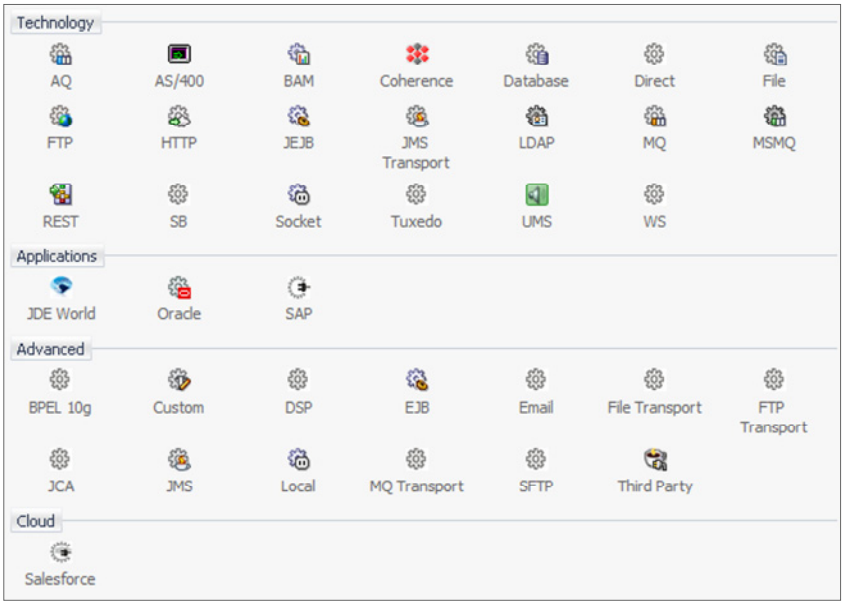
- File and FTP adapters for reading, writing, and transferring files
- Database Adapter for interacting with relational database data
- JMS Adapter for communicating asynchronously with other applications through message-oriented middleware (MOM) services, including Oracle Advanced Queuing and IBM MQ

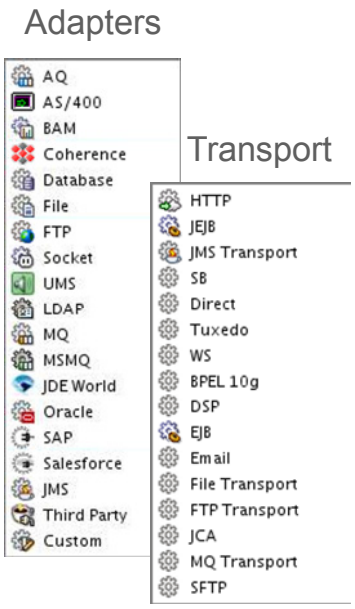
Because JCA adapters implement interfaces by using the Java Connector Architecture (JCA) API standards, custom adapters can be developed to expose existing functionality that is not usually available in a SOA application context. Adapters are an integration technology that extends the reach of the application or service.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Supported Adapters and Transports





Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

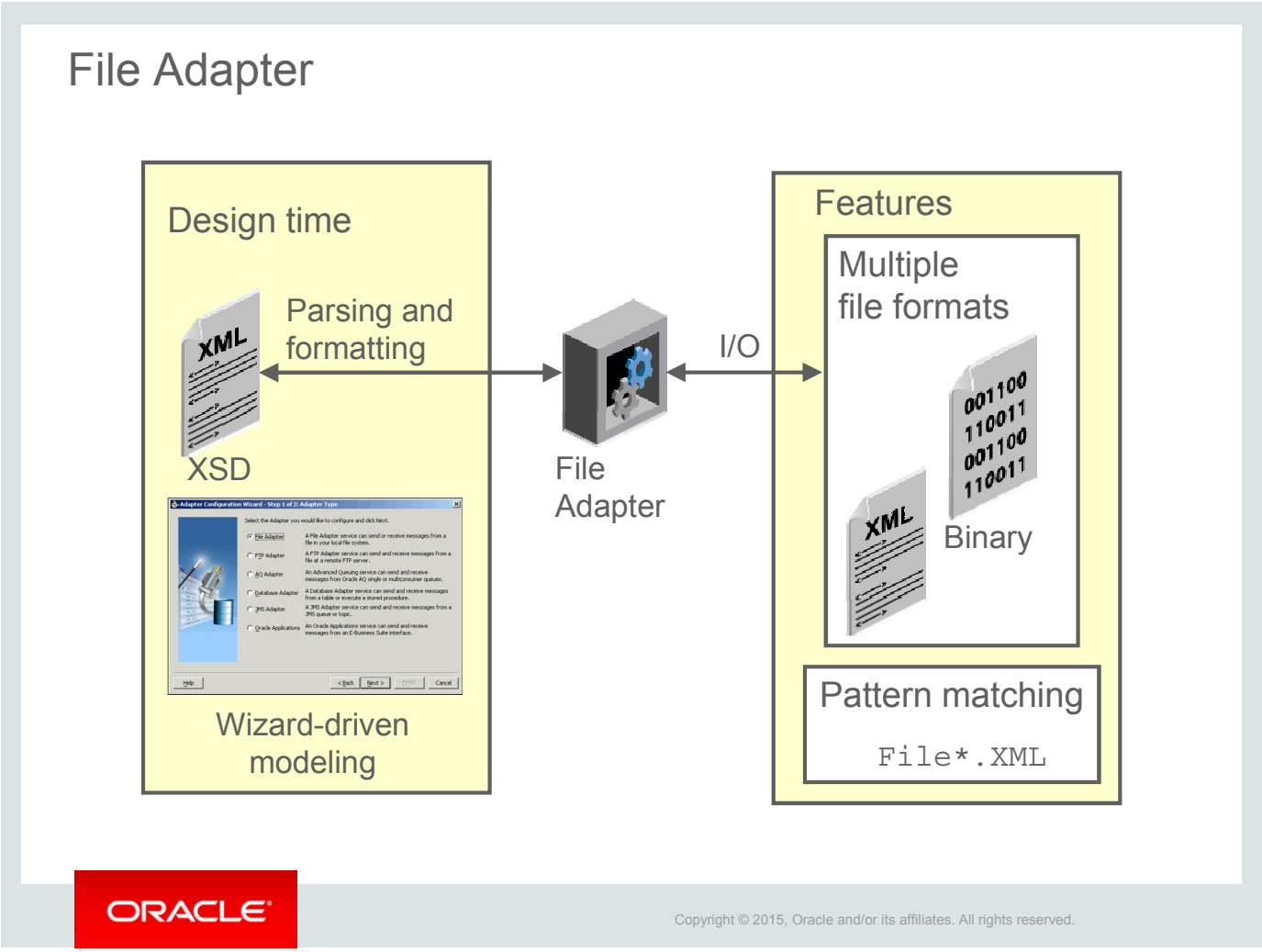
Service Bus provides connectivity to external systems through a variety of transports, each of which is specific to a type of external system. Service Bus supports optimized database queries, and interoperability with web service integration technologies such as .NET, IBM MQ Series, IBM WebSphere, Apache Axis, and iWay adapters. The JCA transport expands the list of supported technologies by letting you connect to external systems using Oracle JCA technology and applications adapters. Additionally, Service Bus supports the REST binding, allowing you to connect to RESTful services using the HTTP transport.

A lot of transports and adapters are available in 12*c* for connecting to a large scale of back-ends and technologies. For backwards compatibility, Oracle supports both the old OSB 11*g* Business Services transports and better integrates the SOA Suite JCA transports in the designer. This allows you to use two default options for the File, JMS, and FTP transport. That is why you can see File, JMS, and FTP in both Technology and Advanced. The “default” or “best-practice” transport is located under Technology. For instance:

- The Technology panel File and FTP transport open the JCA Adapter wizard, and the HTTP and JMS transport open the Business Service wizard.
- The Advanced panel File and FTP transport open the Business Service wizard, and the JMS transport opens the JCA Adapter wizard.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED



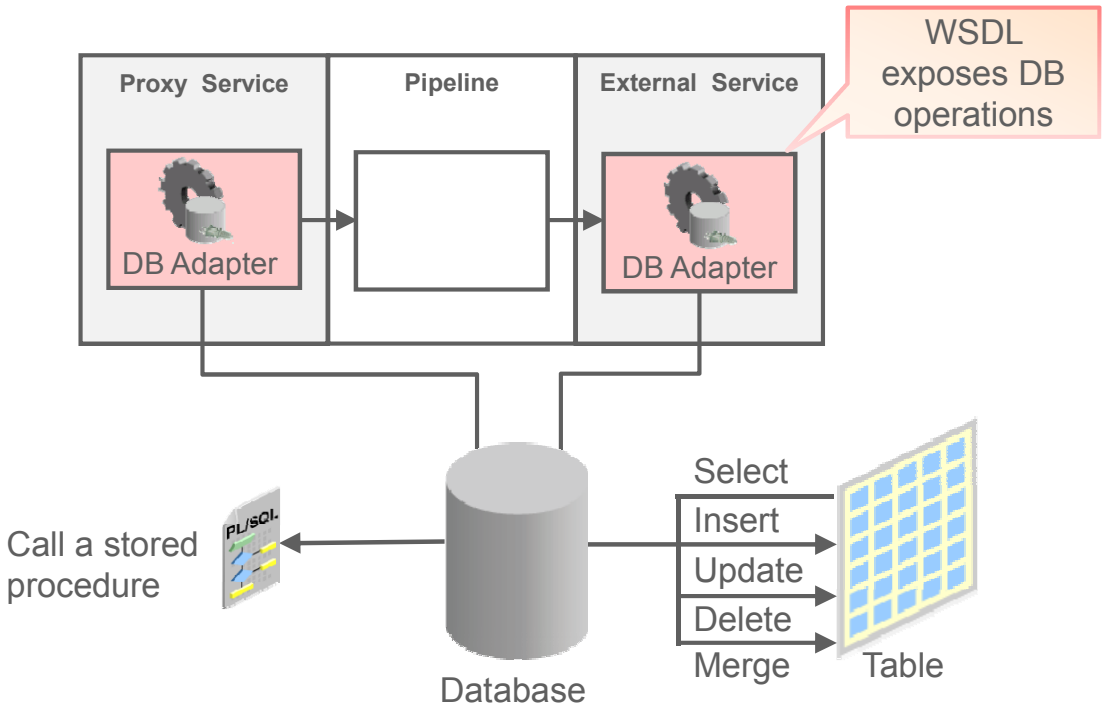
File adapters use XML schema documents to support the parsing and formatting of different file formats such as delimited, positional, XML, binary, and COBOL copy books. Rich file-pattern matching capabilities enable filter processing for specific sets of files. Error recovery and checkpoint capabilities facilitate management of exception conditions.

Scalability is obtained through the multithreaded execution environment, which supplies connection management, and batching and debatching capabilities.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Database Adapter



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The database adapter in Service Bus helps to expose data and operations from relational databases in a service-compliant fashion. The database adapter connects to any relational database using JDBC.

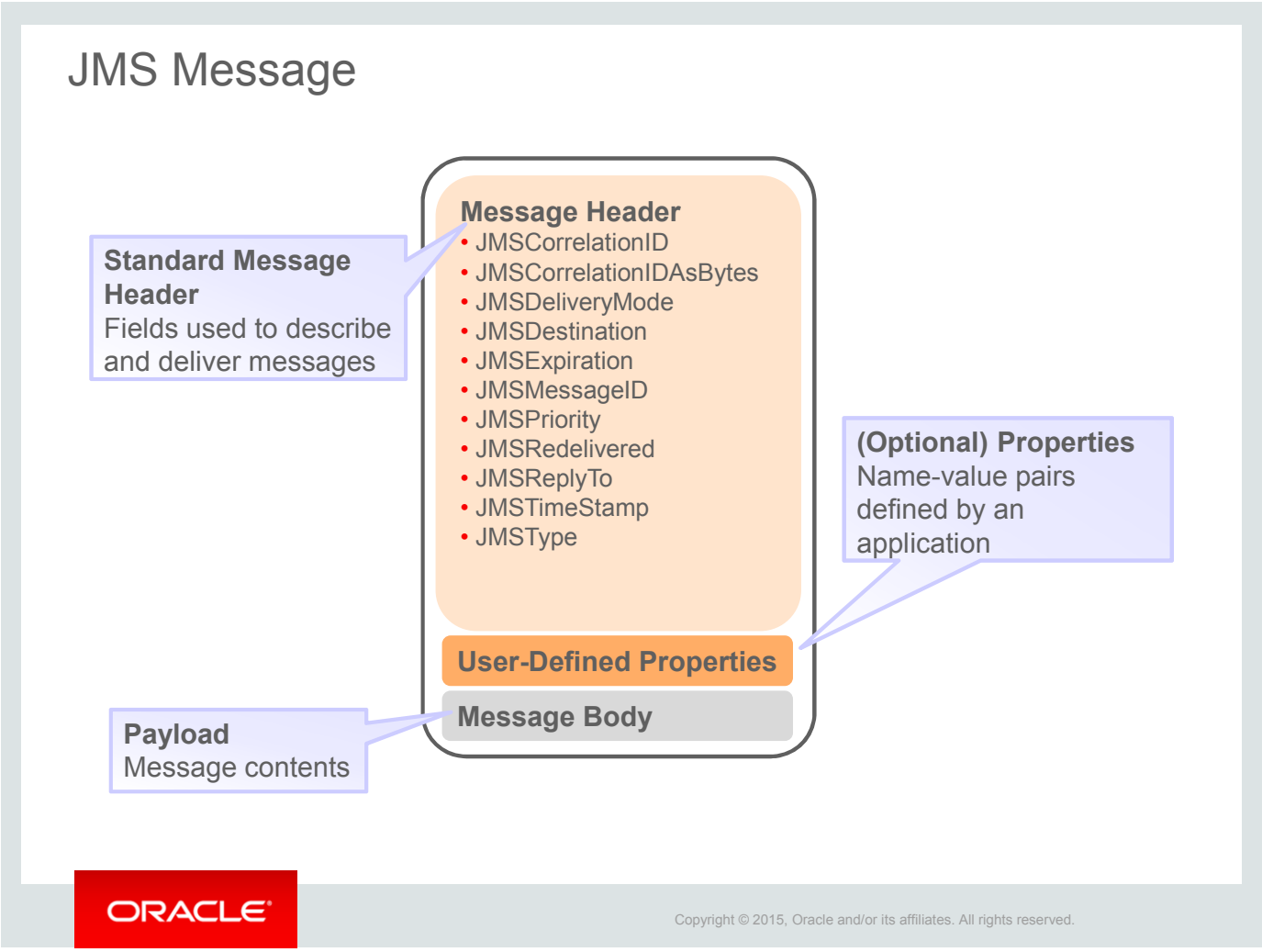
The graphic illustrates that a Database Adapter exposes database operations that can be implemented and used by components in a composite application. The Database Adapter provides a WSDL interface to defined the operations and message structure in an XML format enabling it to be used as a service by composite application components.

The Database Adapter configuration is defined in the Adapter Configuration Wizard, which enables you to specify one of the following types of operations:

- Data manipulation operations (insert, update, merge, delete)
- Query operations (select)
- Custom SQL statements
- Calls to stored procedures, in which the adapter honors any logic in the stored procedure and acts as a call relay interface

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED



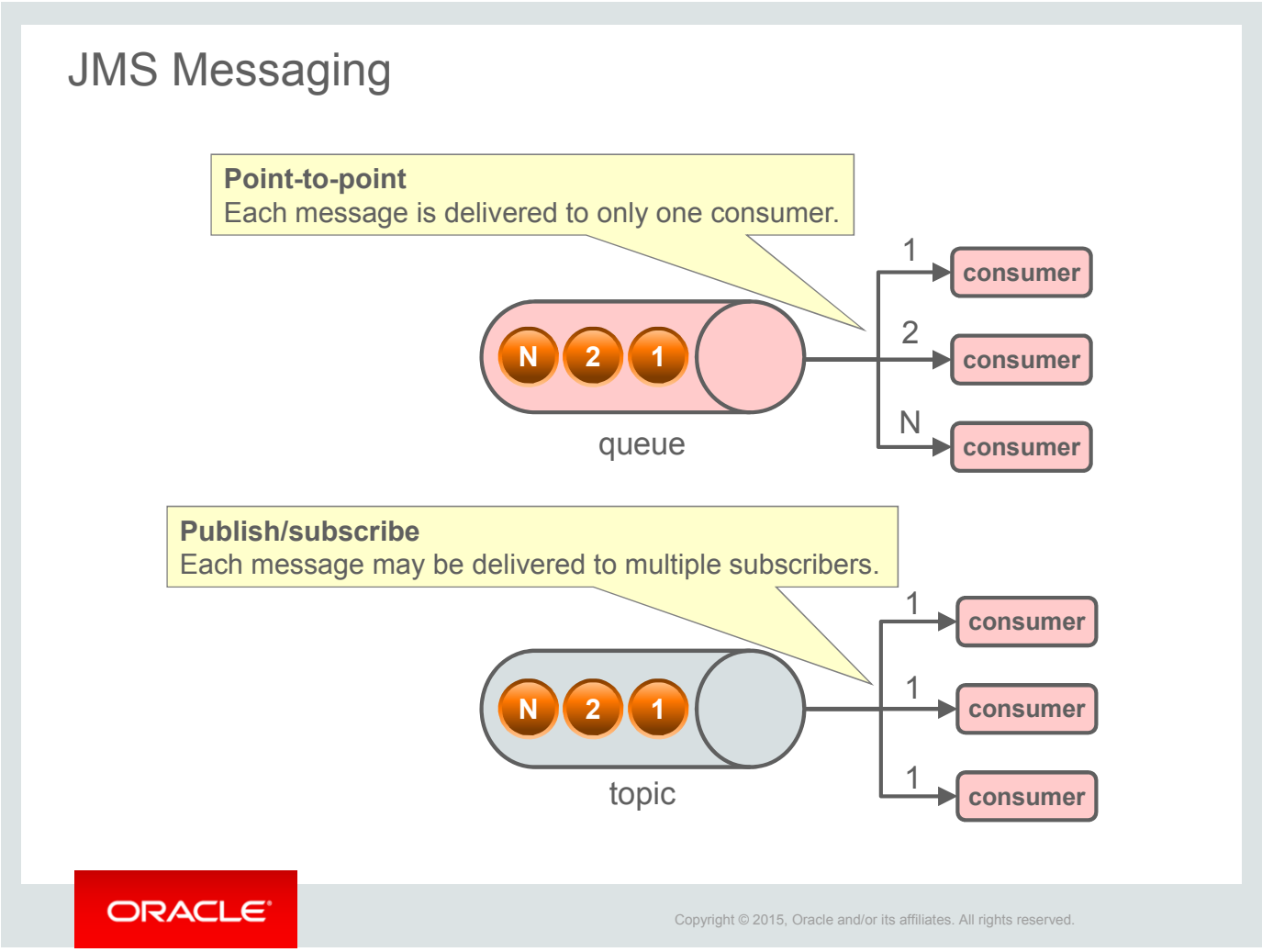
The JMS specification mandates that messages consist of three sections:

- A standard message *header*, which contains fields that are used to describe and deliver messages
- An optional *properties* element, which contains application-defined properties (as name-value pairs)
- The message *body*, which holds the contents (payload) of the message

The payload can be a text message, byte message, map message, stream message, or object message. The properties element is JMS provider specific and varies from one JMS provider to another.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED



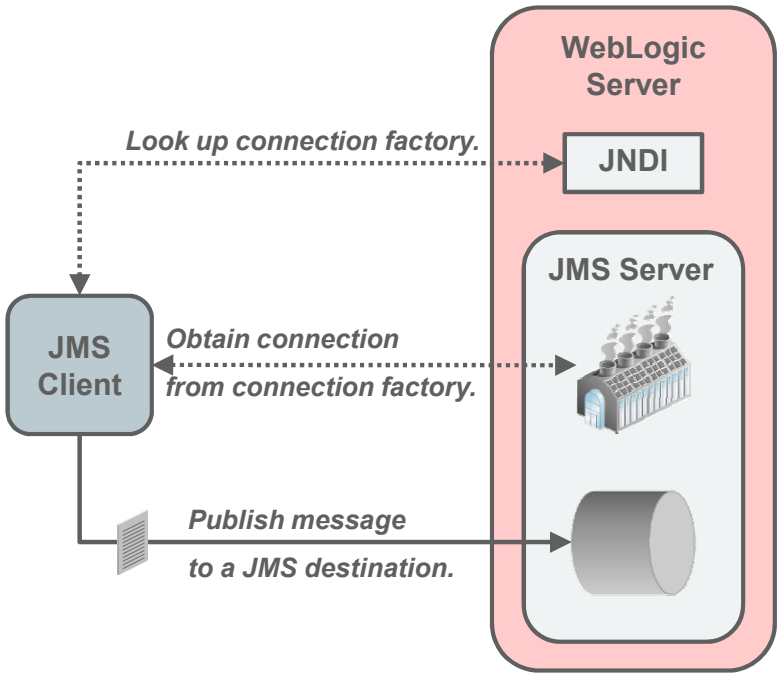
JMS supports two types of messaging: *point-to-point* and *publish/subscribe.*

- In point-to-point messaging, messages are stored in a *queue* until they are consumed. One or more producers write to the queue and one or more consumers extract messages from the queue. A queue may have many receivers, but only one receiver receives each message. The JMS consumer sends an acknowledgment after consumption of a message; this results in purging of the message from the queue.
- In publish/subscribe messaging, producers publish messages to a *topic*, and the consumer subscribes to a particular topic. Many publishers can publish to the same topic, and many consumers can subscribe to the same topic. All the messages published to the topic by the producers are received by all the consumers that are subscribed to the topic. By default, subscribers receive messages only when the subscribers are active. However, the JMS API supports *durable* subscriptions that ensure that consumers receive messages that were published even when the subscribers are not up and running. These messages are persisted by the JMS provider and are either sent to the consumer when it becomes active again or purged from storage if the message expires. The JMS producer can be set either to *persistent* or *non-persistent* mode. The messages are not persisted in the latter case and can be used only for *nondurable* subscriptions.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

JMS Resources



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

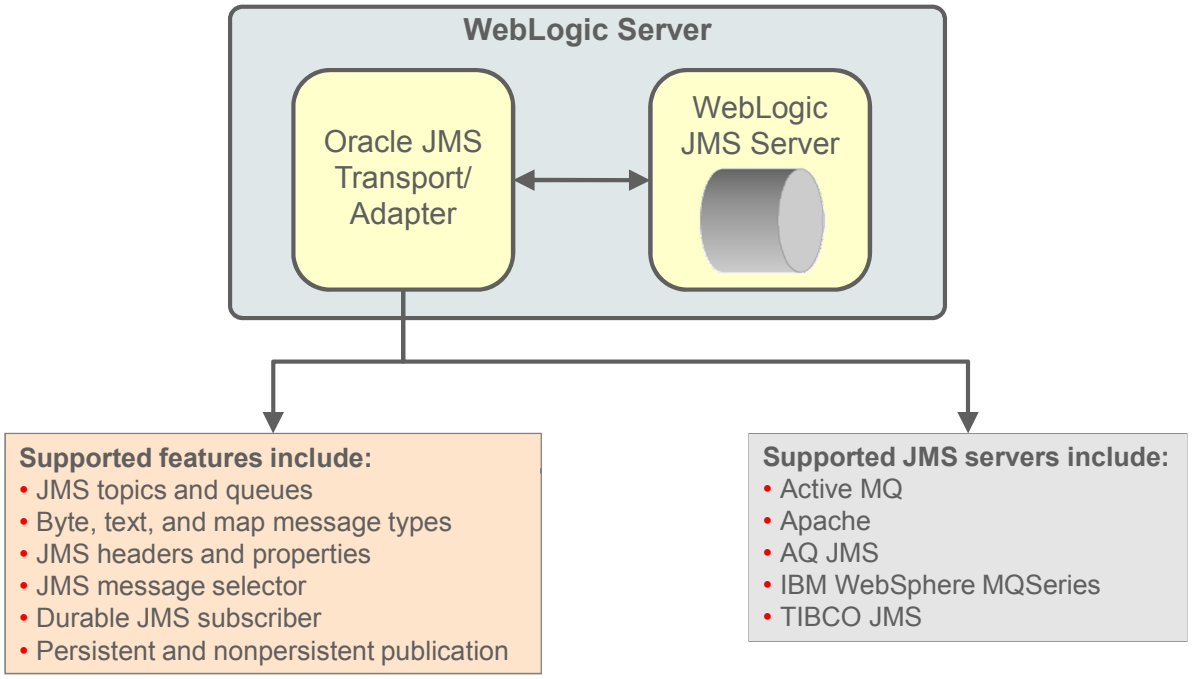
Before an application can publish a JMS message, it must have a *connection* to a *message destination*. That connection is provided by a *connection factory*, which is an administered object that is configured with predefined parameters, such as a default priority to use for messages when a priority is not explicitly defined on the message. The JMS client looks up a connection factory by using the Java Naming and Directory Interface (JNDI), which is a Java API for a directory service that allows Java software clients to discover and look up data and objects via a name.

After the JMS client is provided a connection, the message can be created and published to the appropriate message destination (a topic or queue).

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle JMS Transport



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The JMS transport uses a messaging model that separates service consumers from service providers. This messaging model is defined by using Web Services Description Language (WSDL), which describes the operations exposed by the adapter.

In an outbound message flow, the JMS transport acts as a service provider. The adapter receives a normalized message from the SOA infrastructure, converts that message to a JMS message, and then sends the message to a JMS destination. In an inbound message flow, the JMS transport acts as a message consumer. The adapter converts the JMS message that it receives from a topic or queue to a normalized message, and then sends the normalized message as part of the message exchange to another component as a service request.

The JMS transport supports JMS topics and queues. It supports byte, text, and map JMS message types. The Adapter Configuration wizard provides the Native Format Builder wizard for consuming native data payloads at run time. The adapter supports the JMS message selector for performing filtering while subscribing to JMS topics and queues. (This parameter is based on the SQL 92 language for filtering messages based on the fields present in the JMS header and properties section.)

Oracle University and ORACLE CORPORATION use only

The WebLogic Application Server includes a JMS-compliant message server that provides interim storage for messages in message destinations, which may take the form of topics and/or queues. The message server acknowledges the JMS client that produces the message and ensures successful delivery to the JMS client that consumes the message.

Messages remain live until picked up and committed by all eligible consumers, or until their expiration time is met. The Time-to-Live configuration parameter of the JMS adapter defines this window of opportunity for consumers to receive a message. The timeout may be turned off completely (default), or set to a specific number of milliseconds.

Messages are deleted from storage when they have been received by all subscribers, or after their Time-to-Live has elapsed. After being picked up or expiring, messages are automatically cleaned up (deleted) by the Message Server and are no longer accessible.

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- JCA transport and adapters
- SOA-DIRECT transport
- REST binding

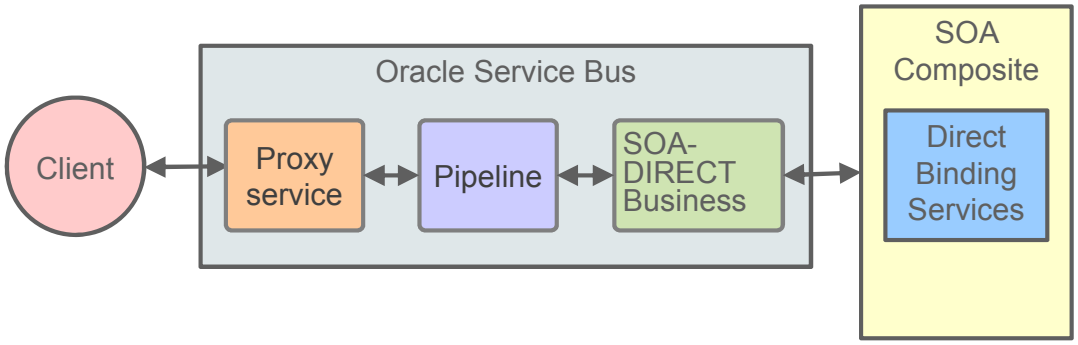
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

SOA-DIRECT Transport

The SOA-DIRECT transport provides native connectivity between Oracle Service Bus and Oracle SOA Suite service components.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle SOA Suite provides a direct binding framework that lets you expose Oracle SOA Suite service components in a composite application.

The Oracle Service Bus SOA-DIRECT transport interacts with exposed services through the SOA direct binding framework. The SOA service components interact in the service bus layer and leverage the capabilities and features of Oracle Service Bus.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Features of SOA-DIRECT Transport

The SOA-DIRECT transport supports the following features:

- Invocation of any SOA binding component services through Java remote method invocation (RMI)
- Optimized RMI transport for invoking SOA services
- Transaction propagation
- WS-Addressing
- Identity propagation
- Attachments
- High-availability and clustering support
- Failover and load balancing
- Connection and application retries on errors

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The SOA-DIRECT transport uses only WS-Addressing for message correlation in synchronous and asynchronous communications. The transport automatically generates ReplyTo properties for asynchronous callbacks.

The SOA-DIRECT transport supports one-way SSL. You can provide identity propagation with the SOA-DIRECT transport by passing the caller's subject through the service or with a service account bound to the service. Because the SOA-DIRECT transport deals with only normalized, abstract messages, the transport does not support WS-Security.

The SOA-DIRECT transport recognizes connection and application errors, letting you configure the appropriate retry settings in the transport configuration. The transport throws generic errors for errors that are not related to either connections or applications.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Transactional Boundaries

- When synchronous BPEL components use the direct binding to interact with proxy services, Service Bus and BPEL components share the same transactional context.
- Transaction must be rolled back if something fails.
- Service Bus direct binding failures are thrown back to the BPEL component as system faults.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When synchronous BPEL components use the direct binding to interact with proxy services, the Service Bus and BPEL components share the same transactional context by participating in the same global transaction. The pipeline can perform any back-end activity within the same transactional context initiated by the BPEL component. To guarantee data consistency, everything that was done in that transaction must be rolled back if something fails to maintain state during processing. Note that the direct binding is typically used because transaction or security propagation is needed.

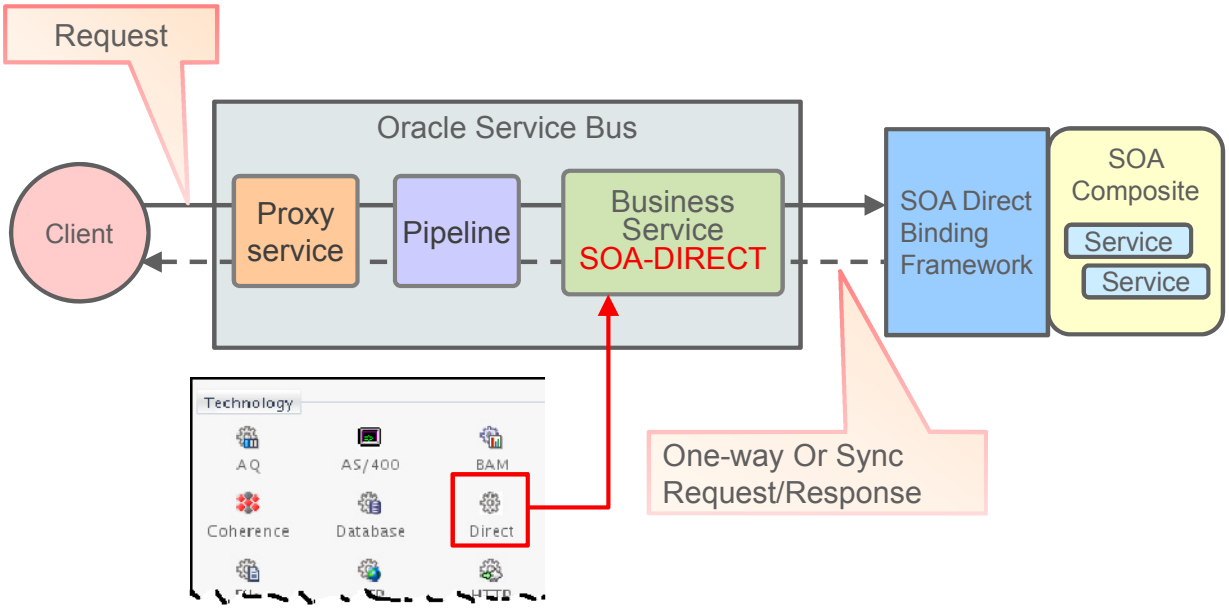
Service Bus direct binding failures are thrown back to the BPEL component as system faults, because Service Bus always marks the transaction for rollback in case of processing failure within the request pipeline. Therefore, any fault thrown from the Service Bus direct binding is a rollback fault and is interpreted as system fault on the SOA Suite side.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Synchronous Invocation of a SOA Composite

The SOA-DIRECT transport can invoke any component in a SOA composite that is exposed as a direct binding service.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The diagram illustrates a synchronous communication pattern between a client and an Oracle SOA service component through Service Bus using a SOA-DIRECT business service and direct binding service. Guidelines for invoking an Oracle SOA direct binding service from a client through Oracle Service Bus are as follows:

- Create a SOA-DIRECT business service in Oracle Service Bus that represents the SOA service component that you want to invoke:
  1. In Oracle Service Bus, create a WSDL resource based on the corresponding Oracle SOA direct binding service WSDL.
  2. Create a new business service with a SOA-direct transport type and a WSDL service type.
  3. Select the WSDL resource you created, and then select the appropriate port or binding.
  4. Set the endpoint URI.
  5. Configure the remainder of the business service.
- Create a proxy service in Oracle Service Bus that invokes the business service. Select the transport type that is used by the client.

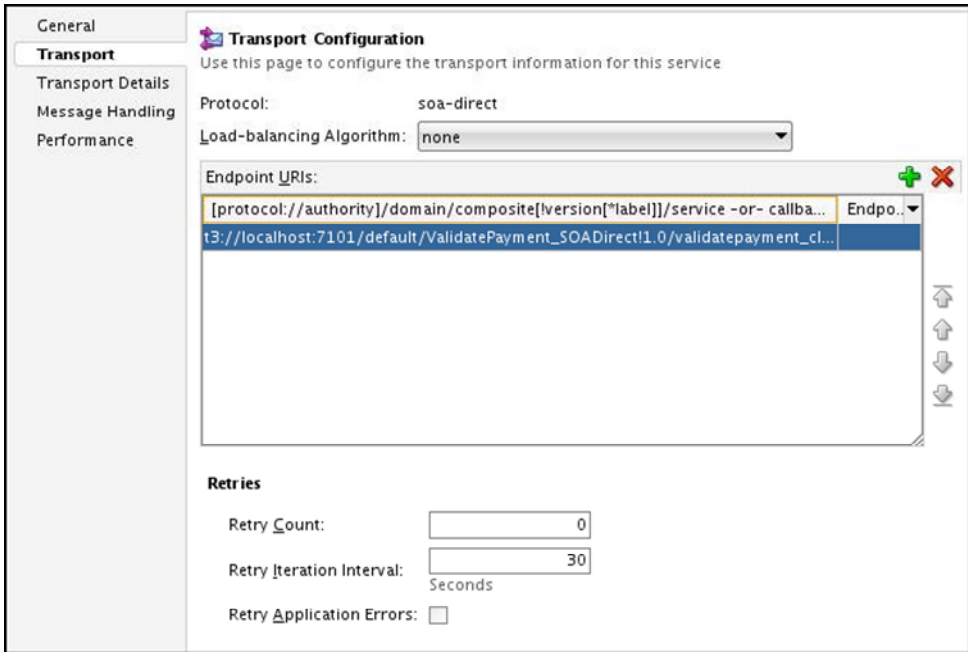
If you are using stateful services, ensure that messages are associated with the correct conversation.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Endpoint URI Linking to a SOA Composite

Get the URI from `http://hostname:port/soa-infra`



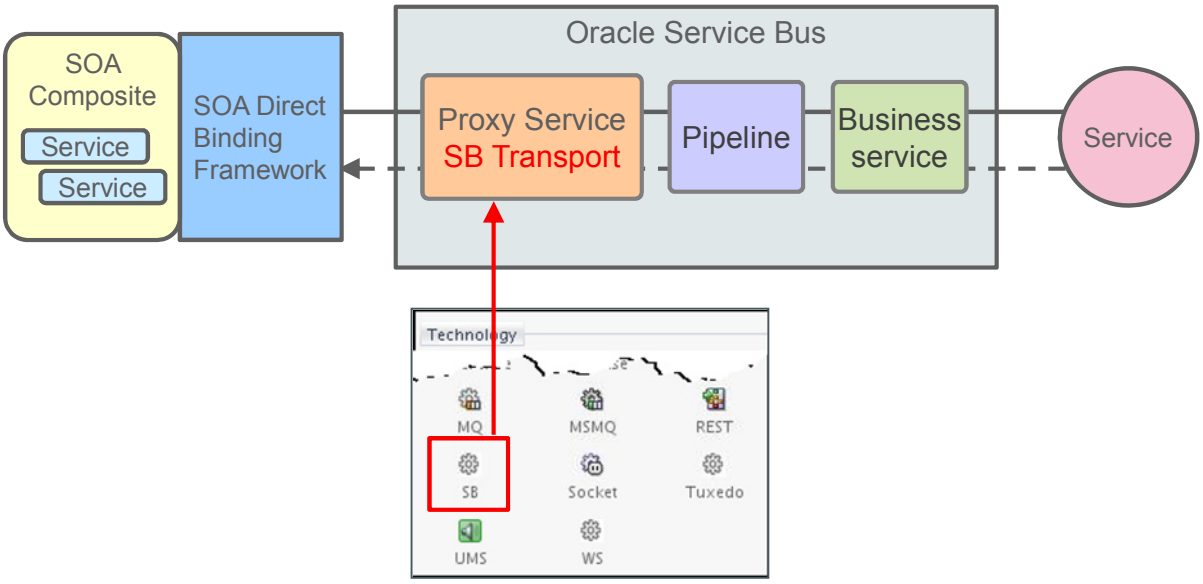
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Synchronous Invocation from a SOA Composite

A SOA composite can invoke any Oracle Service Bus SB WSDL-based proxy service.



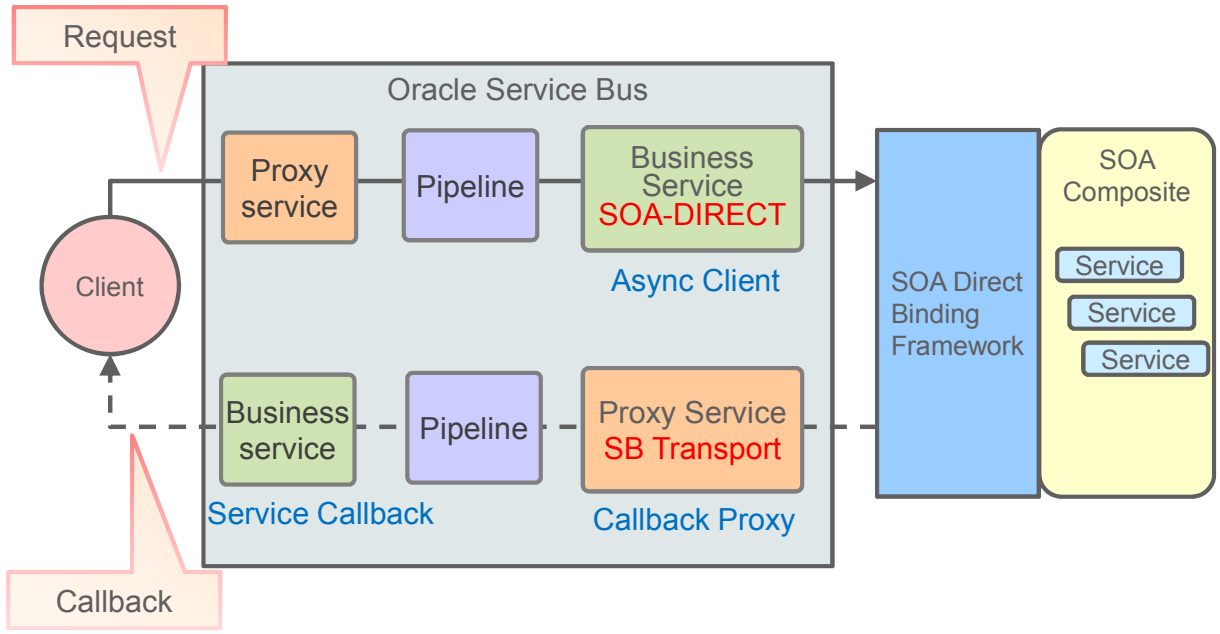
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

To invoke an SB proxy service, the SOA service component must use a direct binding reference of target type “Oracle Service Bus.”

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Asynchronous Invocation of a SOA Composite



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The SOA-DIRECT transport can invoke asynchronous SOA service components that are exposed as direct binding services. The guidelines for invoking an Oracle SOA direct binding service asynchronously from a client through Oracle Service Bus are:

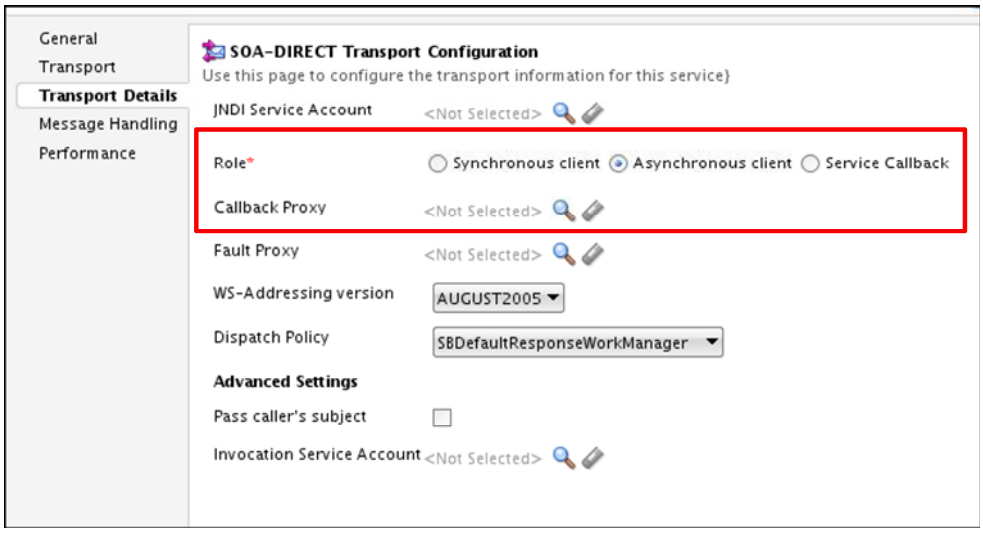
- On the inbound client side, create the OSB artifacts to interact with the client:
  - A request proxy service that invokes the outbound SOA-DIRECT business service
  - A callback business service that handles the callback to the client. Use the transport type used by the client.
- On the OSB outbound side, create the artifacts to interact with the SOA composite:
  - A request SOA-DIRECT business service that makes the request to the Oracle SOA direct binding service, exposing the asynchronous service component that you want to invoke
  - A callback SB proxy service that handles the callback from the direct binding service and invokes the inbound callback business service

The graphic in the slide illustrates an asynchronous communication pattern between a client and an Oracle SOA composite through Oracle Service Bus using a direct binding service, the SOA-DIRECT transport, and the SB transport.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Service Roles in Asynchronous Invocation



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

- Request Proxy Service: Receives the client request. This proxy service invokes the outbound request SOA-DIRECT business service. Because the callback is sent to a different connection, Service Bus must be able to remember the original callback location when calling back the client. When using WS-Addressing, the callback address is sent to the request proxy service in the ReplyTo address header. Before invoking the SOA-DIRECT business service, the request proxy service can pass this address as a referenceParameter property inside the ReplyTo header. Following the WS-Addressing specification, the referenceParameter property is inserted in the SOAP header block of the callback. The callback SB proxy can then extract this callback address and set the callback business service URI.
- **Asynchronous Client:** SOA-DIRECT business service that makes the request to the Oracle SOA direct binding service exposing the asynchronous service component you want to invoke
- **Callback Proxy:** Handles the callback from the direct binding service and invokes the inbound callback business service
- **Callback Business Service:** Handles the callback

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- JCA transport and adapters
- SOA-DIRECT transport
- REST binding

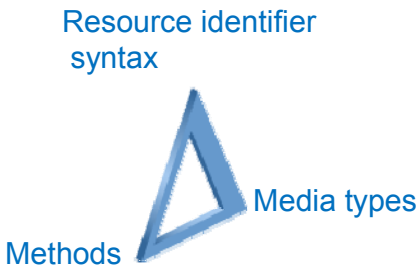
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

REST: Overview

- REST is a model based on resources.
- RESTful services rely on the HTTP protocol.
- REST is based on three fundamental elements:
  - Resource identifier syntax – Represent the actual resources that a service exposes: URI
  - Methods – Protocol mechanisms used to transfer the data: GET, PUT, POST, DELETE
  - Media types – Type of data being transferred: XML, JSON



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

REST-based services are based on a very different model than SOAP-based services. The model is based on resources. The idea is that you send a request for a resource, and the resource that is returned shows you the options for the next step. This model is very flexible and does not require that the client know in advance what services and service calls are available.

REST services rely on a stateless, client-server, cacheable communications protocol—and in virtually all cases, the HTTP protocol. HTTP methods such as GET, PUT, POST, and DELETE are used to represent CRUD type operations (Create, Read, Update, Delete).

RESTful services do not impose constraints on the data format exchanged. Data can be formatted as XML, HTML, JavaScript Object Notation (JSON), and so on.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

REST Queries

Action	Method	URI
Get all the items	GET	/items
Get a single item	GET	/items/id
Create a new item	POST	/items
Edit an item	PUT	/items/id
Delete an item	DELETE	/items/id

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Recall that REST describes resources (reference to data) in terms of URIs and that REST services nearly always use the HTTP protocol. HTTP messages include a header and a body (although the body can be empty). HTTP request messages include methods in the header to specify an action. With this combination, REST queries use HTTP verbs and URIs to specify actions on resources (or data).

The table in the slide lists some typical actions, and the HTTP methods and URIs that might be used to accomplish them.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Comparing REST and SOAP

Action (REST)	Method (SOAP)
Is an architectural style that leverages web standards	Is a formal standard for message exchange
Uses HTTP	Is protocol independent
Permits many data formats	Uses XML
Uses URI and HTTP verbs to access *resources* (data)	Uses a WSDL document to access *operations* (business logic)
Is stateless	WS* standards provide support for stateful transactions.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

REST is an architectural style, whereas SOAP is a message exchange standard. Comparing them is much like comparing apples and oranges, but in practical terms, they describe ways to access web services. As such, the table in the slide lists some features of each. These differences in features mean that in some situations, REST offers advantages, and in others, SOAP is the better choice.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Example: Use Cases for REST and SOAP

REST

- Operations with limited bandwidth and resources
  - Mobile
  - Series of short, chatty conversations
- Stateless operations

SOAP

- Operations that require contextual information and conversational state management
- Asynchronous operations
- Operations that require high levels of security and reliability

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

For web services that require robust security, transactions, and reliability, SOAP can leverage the WS* standards, and would typically be a better choice. However, because of its use of URIs and lighter-weight message formats such as JSON, REST can offer better performance and scalability. REST is well-suited for situations such as mobile computing, which are constrained by limited bandwidth and resources.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Supported Features in OSB

REST support in proxy and business services include:

- Integration with external REST APIs
- XML, JavaScript Object Notation (JSON) with translation to and from XML, and URL-encoded data
- Automatic creation of the required WADL file
- OWSM policy support for REST security
- Setting the HTTP link header in a REST proxy service response
- Reading the value of the HTTP link header in a REST business service response
- Overriding the endpoint URI for a REST business service request
- Ability to browse and consume Oracle REST endpoints from within JDeveloper

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

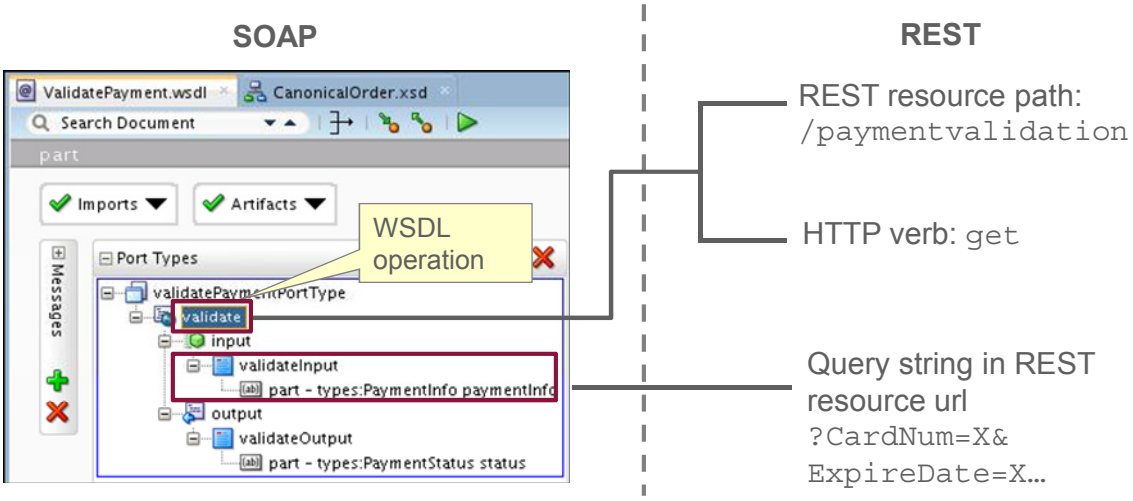
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Creating a REST Binding

To create REST proxy services, you need to:

- Specify a RESTful resource path
- Map the HTTP verb of the resource path to a WSDL operation



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

You can only create REST proxy and business services from JDeveloper.

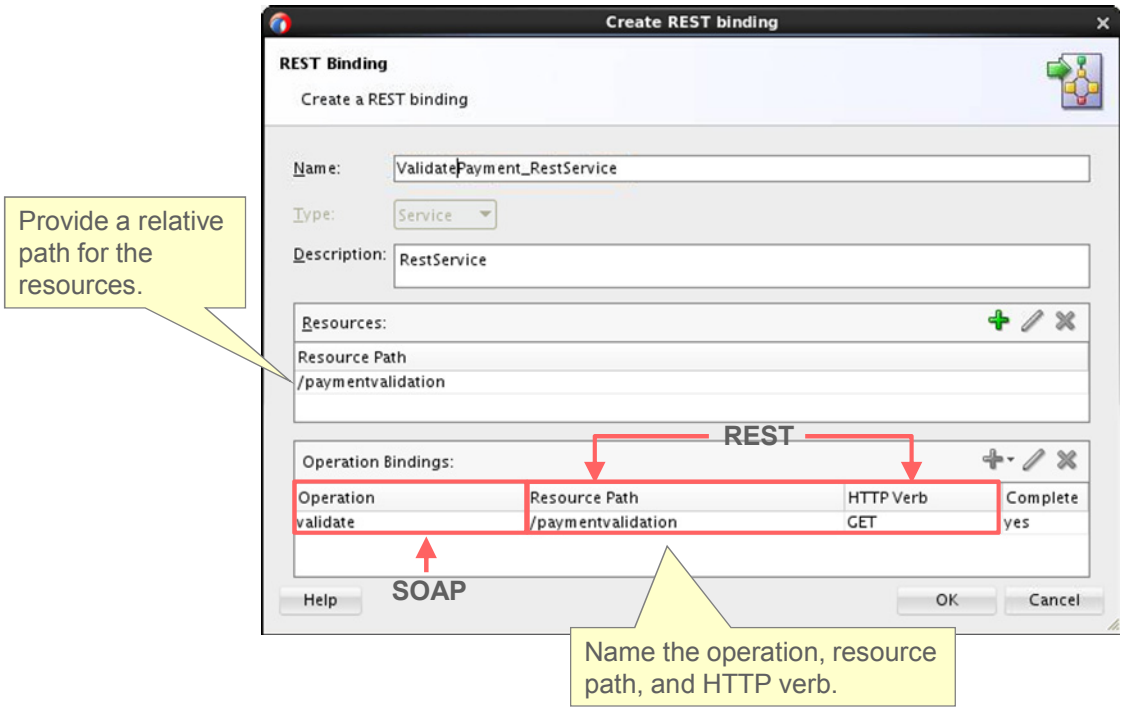
The main task of creating a REST proxy service is to map its WSDL operation to the REST resource URI and the HTTP verb of the resource path.

The diagram shows the example of exposing ValidatePayment as a RESTful service. The WSDL operation `validate` is mapped to resource path `/paymentvalidation` and HTTP verb `get`.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Configuring a REST Binding



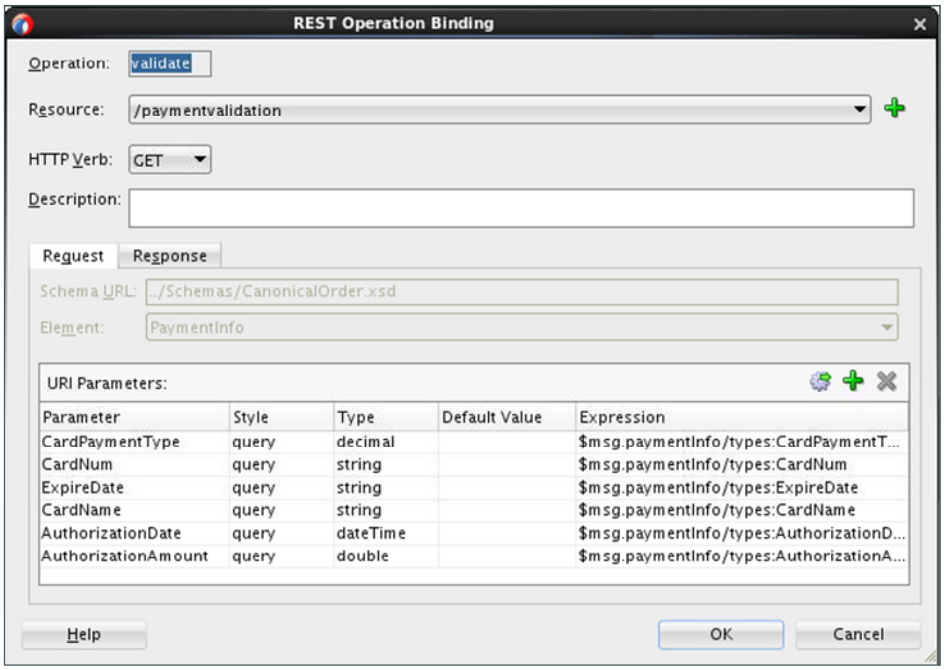
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

To create a REST binding, you provide a name and an optional description, and identify it as either a service or a reference. You then provide a relative path (URI) for the resources. Finally, you bind the operation, the resource path, and the HTTP verb.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Configuring a REST Binding



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

You can also:

- Specify the format of the request payload: XML (default selection), JSON, URL-encoded, or no payload
- Convert HTTP headers and URI to variables as required

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Summary

In this lesson, you should have learned how to:

- Describe JCA transport and adapters
- Describe SOA-DIRECT transport
- Call SOA services using SOA-DIRECT protocol
- Expose existing SOAP services as REST services using the REST binding

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Practice 10: Overview

- 10-1: Calling a SOA Service Using SOA-Direct Protocol
- 10-2: Exposing a SOAP Service as a REST Service

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

11

Reliable Messaging

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Objectives

After completing this lesson, you should be able to:

- Describe reliable messaging
- Describe OSB Quality of Service configurable settings and default behaviors
- Describe the configuration options of using JMS transport for reliable communication

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

OSB supports queuing, holding messages when applications are temporarily unavailable. This lesson discusses how to implement a reliable message flow in OSB.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Reliable Communication

Three aspects of reliable communication:

- Distributed Transactions (XA)
- Quality of Service (QoS)
- Persistence

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

XA is a transaction that can be shared across multiple resources such as a JMS queue, coherence, direct binding, and so on. HTTP transport does not support XA and cannot take part in the so-called global transaction. When an OSB transport or a JCA adapter starts a transaction, this transaction will be handled or controlled by the Java Transaction API (JTA) of the WebLogic server.

QoS allows you to control if the destination will take part in the global transaction. Select, for example, `exactly Once` for joining the global transaction or `best effort` for not.

Persistence is important when the WebLogic server crashes unexpectedly. When the messages are stored in memory, they are lost in cases of a server crash or a reboot. To make data available, the persistent store must be configured on a shared disk.

Oracle University and ORACLE CORPORATION use only

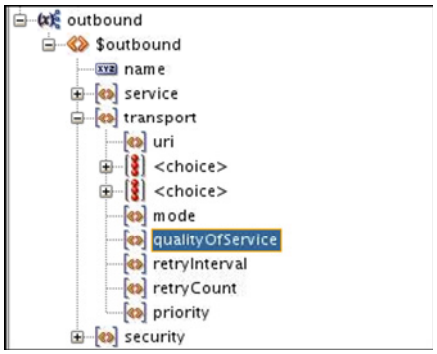
THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Quality of Service

- Quality of service (QoS) is set in the `qualityOfService` element in the `$inbound` and `$outbound` context variable.

```
$inbound/ctx:transport/ctx:qualityOfService
$outbound/ctx:transport/ctx:qualityOfService
```

- Service Bus provides:
  - Two QoS settings:
    - `Exactly-once`
    - `Best-effort`
  - Three delivery behaviors:
    - `Exactly-once`
    - `At-least-once`
    - `Best-effort`



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Service Bus supports reliable messaging. When messages are routed to another service from a route node, the default quality of service (QoS) is exactly-once if the proxy service is configured to be transactional; otherwise best-effort QoS is supported.

Quality of service is set in the `qualityOfService` element in the `$outbound` context variable.

The following delivery guarantee behaviors are provided in Service Bus:

- **Exactly-once:** Exactly-once reliability means that messages are delivered from inbound to outbound exactly-once, assuming a terminating error does not occur before the outbound message send is initiated. Exactly-once means reliability is optimized.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

- **At-least-once:** At-least-once semantics means the message is delivered to the outbound from the inbound at least once, assuming a terminating error does not occur before the outbound message send is initiated. Delivery is considered satisfied even if the target service responds with a transport-level error. However, it is not satisfied in the case of a timeout, a failure to connect, or a broken communication link. If failover URLs are specified, at-least-once semantics is provided with respect to at least one of the URLs.
  At-least-once delivery semantics is attempted if exactly-once is not possible but the `qualityOfService` element is exactly-once.
- **Best-effort:** Best-effort means that there is no reliable messaging and there is no elimination of duplicate messages—however, performance is optimized. It is performed if the `qualityOfService` element is best-effort. Best-effort delivery is also performed if exactly-once and at-least-once delivery semantics are not possible but the `qualityOfService` element is exactly-once.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Default Value of `qualityOfService`

`exactly-once`	`Best-effort`
Proxy configured to be transactional	• Service callout action • Publish action
Route node with following inbound transport: • email • FTP • SFTP • File • JMS (transactional) • Tuxedo (transactional) • MQ (with Backout Threshold set to zero) • WS	Route node with following inbound transport: • HTTP • JMS (non-transactional) • Tuxedo (non-transactional) • MQ

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

OSB provides transactional support while configuring the proxy service. The quality of service is always `exactly-once` by default if a proxy service is transactional.

The default value of the `qualityOfService` element is also `exactly-once` for a route node action for the following inbound transports:

- email
- FTP
- SFTP
- File
- JMS (transactional)
- Tuxedo (transactional)
- MQ (with Backout Threshold set to zero)
- WS

Oracle University and ORACLE CORPORATION use only

The default value of the `qualityOfService` element for a route node is `best-effort` for the following inbound transports:

- HTTP
- JMS (non-transactional)
- Tuxedo (non-transactional)
- MQ (with Backout Threshold set to greater than zero)

In addition, the default value of the `qualityOfService` element is always `best-effort` for the following:

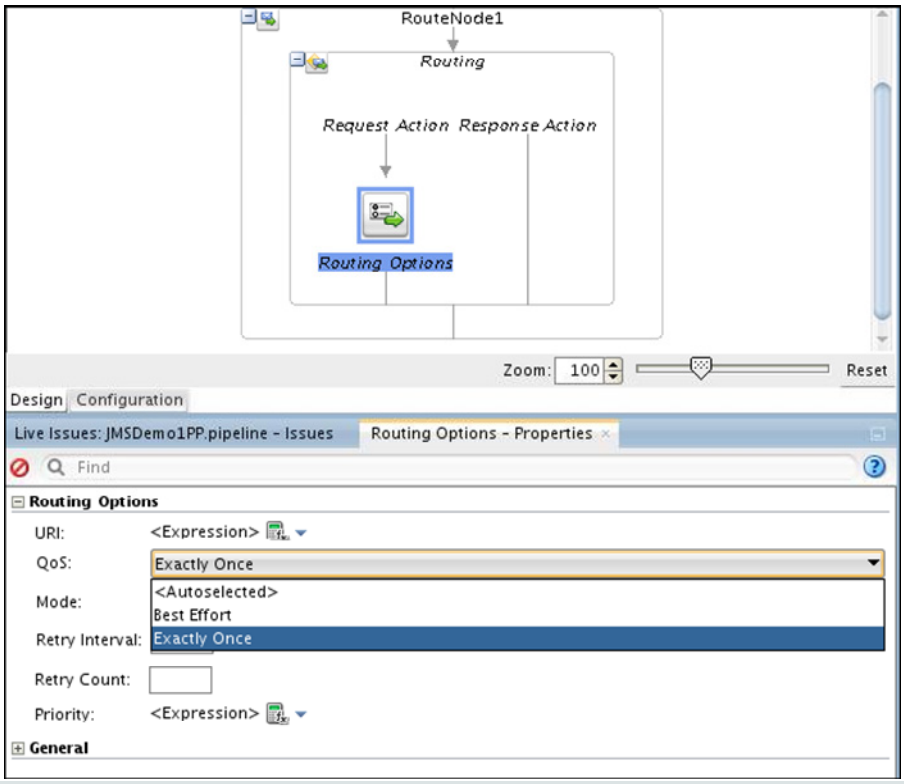
- Service callout action – always `best-effort`, but can be changed if required.
- Publish action – defaults to `best-effort`, modifiable

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Overriding the Default Value of QoS in Routing Options Action



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The inbound QoS value cannot be changed and Service Bus uses the inbound value as a default value for the outbound transport. However, you can change the outbound QoS value with a Routing Options action.

You can override the default `qualityOfService` element attribute for the following pipeline actions:

- Publish
- Dynamic Publish
- Publish Table
- Service Callout
- Routing
- Dynamic Routing
- Routing Table

To override the `qualityOfService` element attribute, add a Routing Options action to any of the previous actions, select the QoS option, and select the override value.

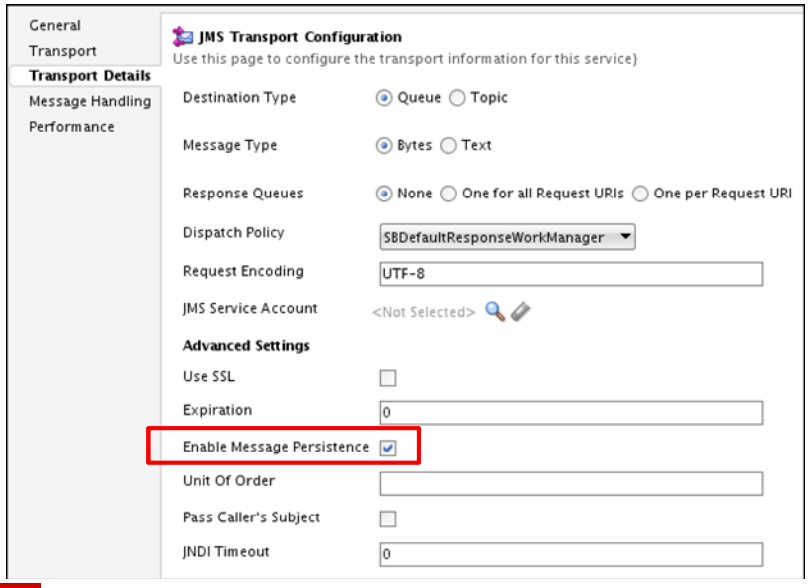
The above example shows modifying the QoS property in the outbound request of a Routing action.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Configurations for Reliable Communication

- Enabling JMS message persistence
- Configuring retry handling in JMS queue
- Using global transactions and quality of service



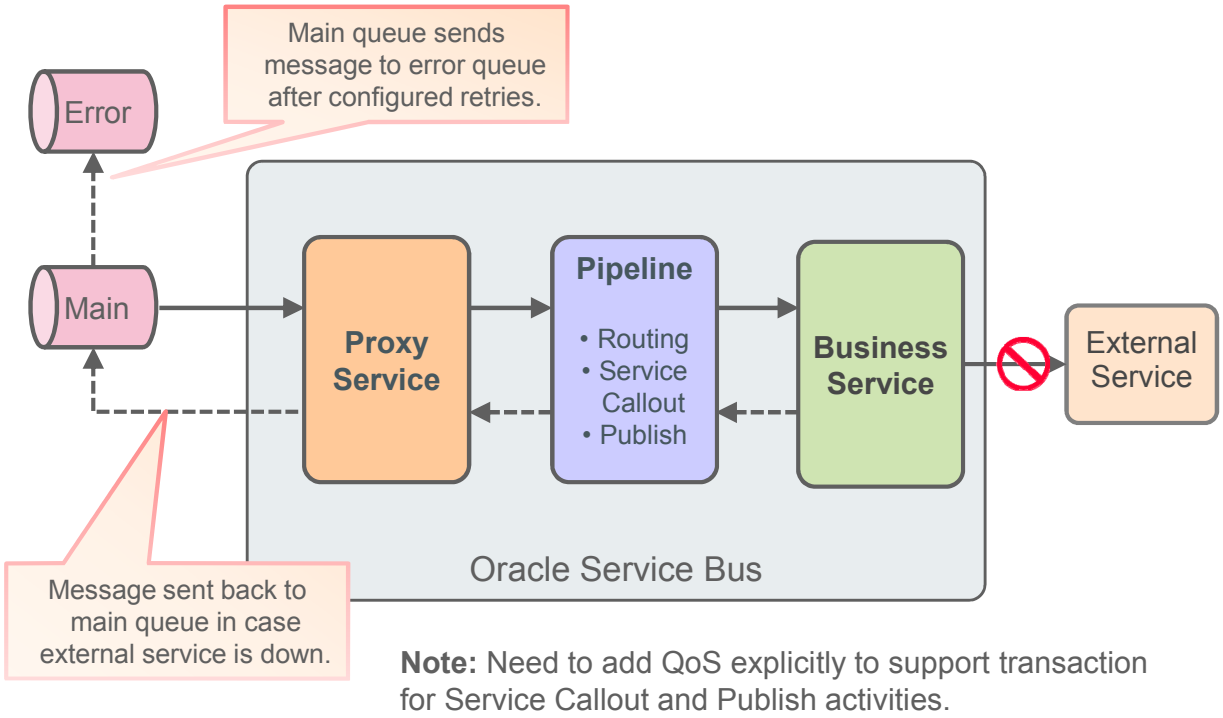
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When sending messages to a JMS queue, the Message Delivery Mode option controls if a message is guaranteed to be delivered once, and if it is safely stored in the persistent store of the JMS server. There is also a non-persistent option, where the messages are stored in memory and may be lost in case of a WebLogic or JMS server failure, or when the WebLogic server is rebooted.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Configuring Retry Handling in JMS Queue



Note: Need to add QoS explicitly to support transaction for Service Callout and Publish activities.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Consider a typical scenario where an application puts message to JMS queue; the proxy service listens on the queue; the business service routes message to an external service. In some situations it could be possible that the external service is not available. When the service is down, you want Service Bus to abort the complete transaction, redeliver the original message to the queue and pick up the message at a later moment.

The diagram in the slide shows when a proxy service reads a message from the main queue and there is an error somewhere in the process, the message will be rolled back and stays on the queue. WebLogic will detect this and wait for retry interval time configured before the proxy service can consume it again. If this fails after the retry count, then WebLogic will move the message automatically to the error queue. Such an error queue can be monitored for new messages through the WebLogic diagnostic module. An administrator might be able to solve the problem and remove the message or move it back into the original queue.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

High-Level Steps

1. Create a main queue and a corresponding error queue for the main queue.
2. Set the delivery failure option for the main queue.
3. Create a proxy service to consume the message from the main queue and enable the transaction parameters.
4. Create a business service to post the message to target queue.
5. Create a message flow to route the message to business service.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The first two steps are creating and configuring the two JMS queues (main and error) in WLS Admin console, and you need to:

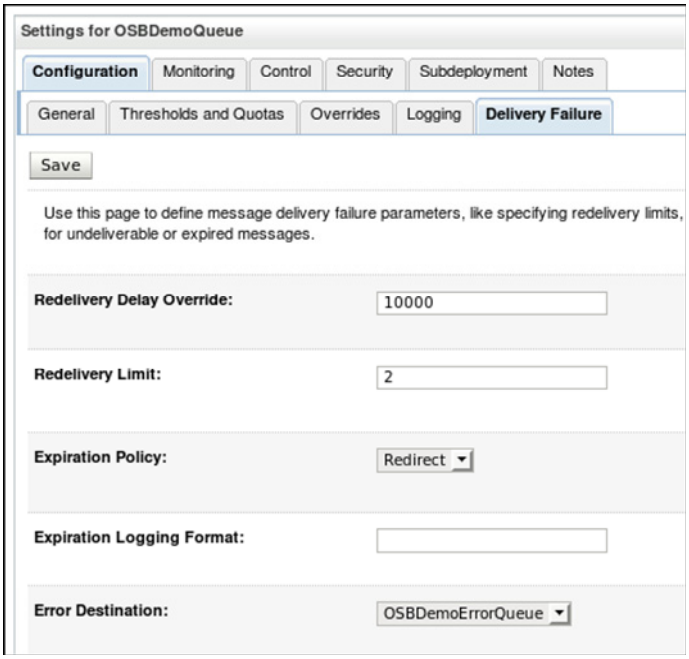
- Create a queue (main) within your JMS Module, which contains the requests
- Create a second queue (error) where the messages should go when the redelivery limit is reached
- Use the Delivery Failure tab of the main queue to configure the delivery failure option

After the configuration of the JMS queues you need to create a proxy service and a business service that represents the external service. These steps of creating Service Bus application can be performed in JDeveloper or Service Bus console.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Configuring Delivery Failure Options



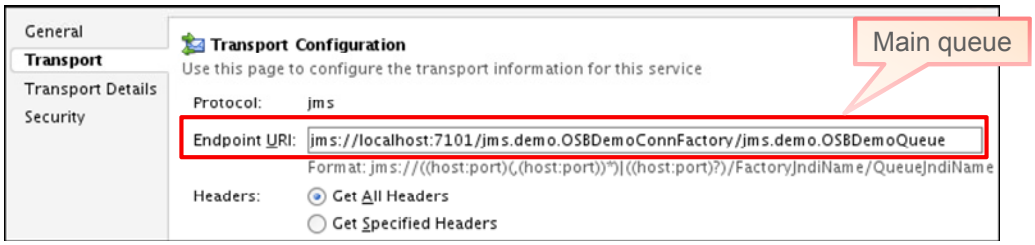
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Redelivery limit and expiration policy specified on the main queue that will take part in a reliable communication. This option prevents an infinite loop: when the redelivery limit is reached, the message will be moved to the error queue.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

JMS Transport Configuration of the Proxy Service



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When creating the proxy service make sure it is listening on the queue for the incoming messages.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Summary

In this lesson, you should have learned how to:

- Describe OSB's support for reliable messaging
- Describe OSB Quality of Service configurable settings and default behaviors
- Describe the configuration options of using JMS transport for reliable communication

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Demonstration: Overview

- Configuring Retry Handling in JMS Queue

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

12

Service Bus Security

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Objectives

After completing this lesson, you should be able to:

- Describe security concepts
- Compare transport-level and message-level security standards
- Describe Oracle Service Bus security features
- Use Service Bus and OWSM to secure web services
- Describe and assign access control policies to services

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Security concepts
  - Transport-level versus message-level security
  - WS-Policy and WS-Security
  - SAML security token
- Oracle WSM security
  - Oracle WSM concepts
  - Securing services with OWSM policies
- Access control policies

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Web Services Security: Overview

To secure your web service, you need to configure one or two different types of security:

- Transport-level security: Secures connections between service consumer and provider
- Message-level security: Secures a message throughout its journey between the sender and the intended recipient
- Access control security: Specifies which roles are allowed to access what web services

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Web services security covers a lot of territory. It is usually divided into smaller, more manageable chunks:

- **Transport-Level Security:** Security begins at the transport or wire level. It allows the client and Web service to communicate securely across a network by providing secure connection.
- **Message-Level Security:** Message-level security is end-to-end security. It means that the message is secured throughout its journey between the sender and the intended recipient. In web service communication, the message can travel through various entities before it reaches its intended recipient. The message will be secured during this journey if message-level security is applied.
- **Access Control Security:** Web services provide clients with access to resources. If a resource is secured, then a client needs the appropriate credentials to gain access. Access control security provides both authentication and authorization.

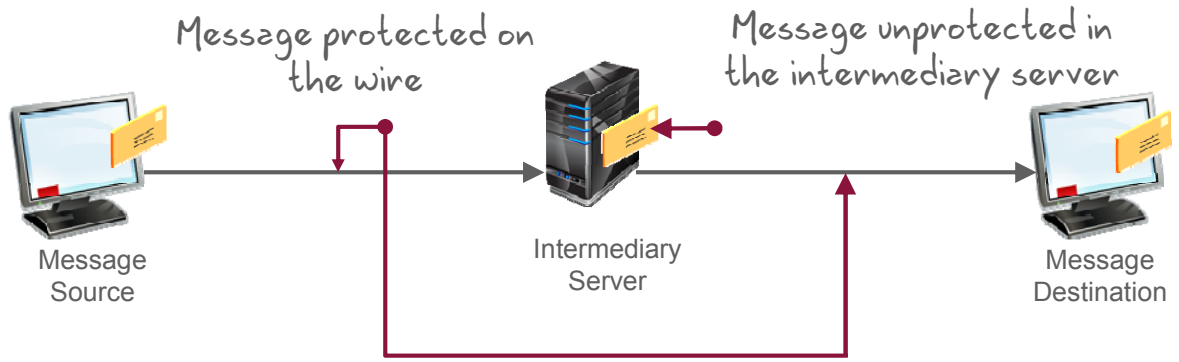
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Transport-Level Security

Transport-level security uses protocol-dependent security standards:

- HTTPS (HTTP over SSL)
- JMS over SSL
- SFTP



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Transport-level security applies security checks as part of establishing a connection between service consumers, proxy services, and business services. The type of security checks that Oracle Service Bus can apply depends on the protocol that the proxy service or business service uses to communicate. Some protocols can also encrypt the communication between client and endpoint to prevent snooping from third parties.

Transport-level security secures only the connection. Even if you use the HTTPS or JMS protocols to encrypt the communication, if there is an intermediary between a web services client and an Oracle Service Bus proxy service (such as a router, message queue, or another proxy service), the intermediary gets the SOAP message in plain text. When the intermediary sends the message to the second receiver, the second receiver does not know who the original sender was. To prevent unintended intermediaries from viewing or modifying SOAP or JMS messages, configure message-level security.

For example, each proxy service or business service that communicates to SB over the HTTPS protocol can require one of the following levels of authentication:

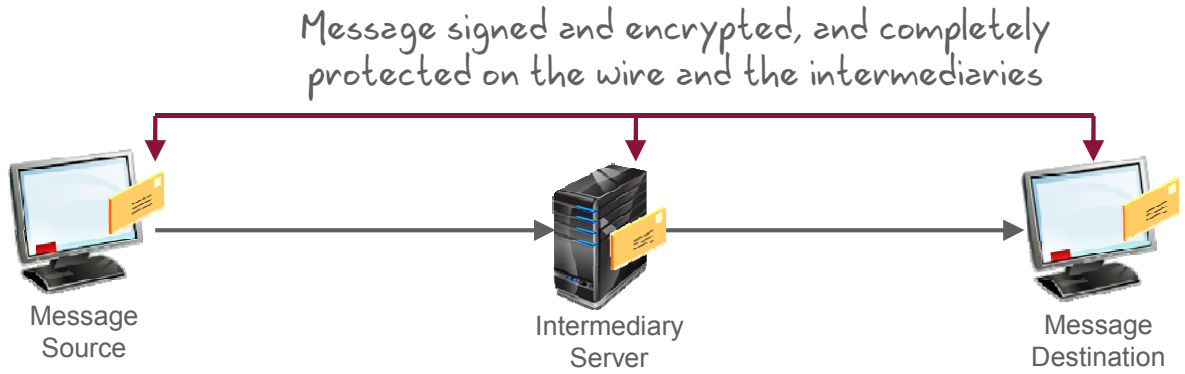
- One-way SSL, no authentication
- One-way SSL, basic authentication
- Two-way SSL, CLIENT CERT authentication

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Message Security

- Uses transport-independent security standards:
  - XML encryption
  - XML digital signatures
  - Web services security
- Has the ability to secure only portions of a message for better performance



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle Service Bus supports OASIS Web Services Security (WSS) 1.0., and 1.1 WSS defines a framework for message confidentiality, integrity, and sender authentication for SOAP messages. Using WSS, Oracle Service Bus provides support for securing messages by using digital signatures, encryption, or both. Although it is not necessarily a substitute for transport-level security, WSS is ideal for end-to-end message confidentiality and integrity.

In point-to-point situations, confidentiality and data integrity can also be enforced on web services by using Transport Layer Security (TLS) (for example, by sending messages over HTTPS). WS-Security, however, addresses the wider problem of maintaining the integrity and confidentiality of messages until after a message was sent from the originating node, providing so-called end-to-end security.

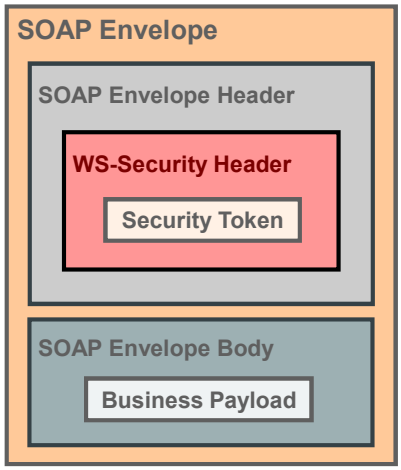
Applying TLS can significantly reduce the overhead involved by removing the need to encode keys and message signatures into ASCII before sending. A challenge in using TLS would be if messages needed to go through a proxy server, as it would need to be able to see the request for routing. In such an example, the server would see the request coming from the proxy, not the client.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

WS-Security

- Defines how to attach XML Signature and XML Encryption headers to SOAP messages
- Supports multiple security tokens for authentication:
  - Username/password
  - X.509 certificate
  - Kerberos ticket
  - SAML



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Web Services Security (WS-Security) is an OASIS standard that defines interoperable mechanisms to incorporate message-level security into SOAP messages. WS-Security supports message integrity and message confidentiality. It also defines an extensible model for including security tokens in a SOAP envelope and a model for referencing security tokens from within a SOAP envelope. WS-Security token profiles specify how specific token types are used within the core WS-Security specification. Message integrity is achieved by using XML digital signatures; message confidentiality is accomplished by using XML encryption. WS-Security allows you to specify which parts of a SOAP message are digitally signed or encrypted.

The WSS protocol includes details on the use of SAML and Kerberos, and certificate formats such as X.509. In addition, it describes how to attach security tokens, including binary security tokens such as X.509 certificates and Kerberos tickets, to messages.

Oracle Service Bus supports WS-Security over HTTP (including HTTPS) and JMS.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

SAML

Security Assertion Markup Language (SAML):

- An open framework for sharing security information on the Internet through XML documents
- The dominant standard for federated identity
- A protocol that does not define any new approaches to authentication/authorization (It simply generates appropriate tokens/assertions after authentication occurs.)

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

SAML specification defines a standard, XML-based approach for passing security tokens defining authentication and authorization rights. Originally, SAML was targeted more toward distributed authentication and single sign-on (SSO), but those concepts are also central to web services security now.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

SAML Architecture

SAML includes three parts:

- **Assertions:** How you define authentication and authorization information
- **Protocol:** How you ask (SAML `Request`) and get (SAML `Response`) the assertions you need
- **Bindings and Profiles:** How SAML assertions ride "on" (bindings) and "in" (profiles) industry-standard transport and messaging frameworks

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The SAML technology is rooted in XML. The information passed around between asserting parties (SAML authorities) and relying parties is mostly in the form of XML, and the format of these XML messages and assertions is defined in a pair of SAML XML schemas.

The full SAML specification is used in browser-based federation cases. However, web services security systems use only SAML assertions. The protocol and bindings are taken care of by WS-Security and the transport protocol (for example, HTTP).

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

WS-Security and SAML

- WS-Security and SAML work together:
  - WS-Security defines how you insert the information into a SOAP envelope.
  - SAML defines what the security information is.
  - WS-Security allows SAML assertions to be placed inside a SOAP header.
- SAML Token Profile 1.1 specifies how SAML assertions can be used for web services security.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

WS-Security is the messaging language; SAML is the security language.

SAML Token Profile 1.1 is part of the core set of WS-Security standards, and specifies how SAML 1.1 assertions can be used for web services security. SAML assertions and references to assertion identifiers are contained in the WS-Security Header element, which in turn is included in the SOAP envelope Header element.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

WS-Policy: Overview

WS-Policy:

- Provides a model and syntax for describing web service policies
- Enables security policies to be advertised in the WSDL
- Is broken up into several subsidiary specifications:
  - WS-Policy: Defines a grammar that explains web services policies
  - WS-PolicyAttachment: Associates policies to web services
  - WS-PolicyAssertions: Defines a set of general policy assertions

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The Web Services Policy Framework, called WS-Policy, provides a model and syntax for describing the policies of a web service. It enables advertising web service's security requirements in the WSDL using standard WS-Policy. It enables the clients to know wire-level policy requirements (such as what token to send, which algorithm to use, and so on) without the need to call service administrator.

WS-Policy includes the following specifications:

- **WS-Policy:** Defines a grammar that explains web services policies
- **WS-PolicyAttachment:** Defines policy on using attachments
- **WS-PolicyAssertions:** Defines a set of general policy assertions (The constraints and requirements of web services are expressed as policy assertions.)
- **WS-SecurityPolicy:** Defines a set of security policy assertions for use with the WS-Policy framework describing how messages are secured

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Policy Assertion

- A policy assertion:
  - Is a basic unit representing an individual requirement in a policy
  - Is domain specific (security, reliability)
- Service providers use a policy assertion to convey a condition under which they offer a web service.
- An example of policy expression:

```
<Policy>
  <sp:SamlToken sp:IncludeToken="http://schemas.xmlsoap.org/ws/
    2005/07/securitypolicy/IncludeToken/AlwaysToRecipient">
     <wsp:Policy>
          <sp:WssSamlV11Token10 />
     </wsp:Policy>
  </sp:SamlToken>
</Policy>
```

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

A policy assertion is a piece of service metadata representing an individual requirement, capability, or other property of a behavior. Assertions are domain (security, reliability, and transaction) specific and the details are specified in separate specifications. Service providers use a policy assertion to convey a condition under which they offer a web service. A policy-aware client can recognize policy assertions and engage these behaviors automatically.

The policy expression (XML form of the policy) in the example in the slide consists of a `Policy` main element and a child element `sp:WssSamlV11Token10`. Child elements of the `Policy` element are policy assertions. This example represents a specific policy assertion for SAML. A policy-aware client can recognize this policy assertion and engage text encoding automatically.

Note: The `wsp` prefix is used here to denote the Web Services Policy XML namespace.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Quiz

Which of the listed options is not correct for the following statement?

"SAML is built upon several existing standards, such as_____."

a. XML
b. XML Schema
c. XSLT
d. XML Signature
e. XML Encryption
f. HTTP

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Answer: c

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Security concepts
- Oracle WSM security
  - Oracle WSM concepts
  - Securing services with OWSM policies
- Access control policies

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

SB Security Capabilities

- Integrate with Oracle Web Services Manager (OWSM) and Oracle Platform Security Services (OPSS)
- Enforce transport and/or message security technologies:
  - Authentication
  - Confidentiality
  - Integrity
- Inbound security
- Outbound security
- Propagate security identity
- Bridge different security technologies

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle Service Bus ensures service security at all levels. A comprehensive set of components for built-in security gives customers significant flexibility and choice. Users can also plug in home-grown or third-party security components. Built-in capabilities allow flexibility in implementation by enabling security at the following levels:

- **Transport Security:** SSL/Basic Auth
- **Message Security:** Support for WS-Policy, SAML, user ID/password, X509, Signing and Encryption, and custom security credentials
- **Console Security:** Support for user web single sign-on (SSO) and role-based access
- **Policy:** Leverages WS-Policy

Inbound security ensures that SB proxy services handle only the requests that come from authorized clients. (By default, any anonymous or authenticated user can connect to a proxy service.) It can also ensure that no unauthorized user has viewed or modified the data as it was sent from the client.

Oracle University and ORACLE CORPORATION use only

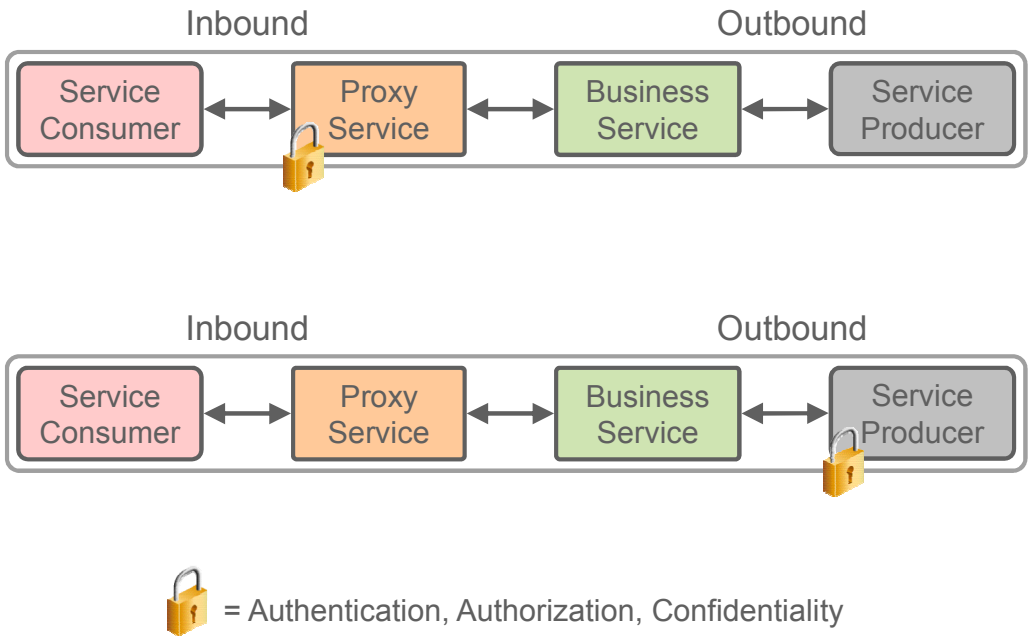
Outbound security secures communication between a proxy service and a business service. Most of the tasks involve configuring proxy services to comply with the transport-level or message-level security requirements that business services specify. If a business service requires the use of the Public Key Infrastructure (PKI) technology for digital signatures or SSL authentication, a proxy service provider is created that provides private keys paired with certificates.

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Active and Passive Intermediary



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When SB functions as an active intermediary, it plays an active role in validating and processing the incoming client security context. For example, SB might establish an encrypted secure sockets layer (SSL) channel between it and the client, or authenticate the client's credentials, or grant or deny access to the client based on custom authorization policies.

Alternatively, when SB is a passive intermediary, the client's security context is simply passed through unchanged to the back-end services, and they are responsible for performing authentication or authorization.

In an active intermediary scenario for web services, the client applies WS-Security to the request and/or response messages. The proxy service processes the security header and enforces the WS-Security policy. For example, the client encrypts and signs the message and sends it to the proxy service. The proxy decrypts the message and verifies the digital signature, and then routes the message. Before the proxy service sends the response back to the client, the proxy signs and encrypts the message. The client decrypts the message and verifies the proxy's digital signature.

Oracle University and ORACLE CORPORATION use only

In a web service pass-through scenario, the client applies WS-Security to the request and/or response messages. The proxy service does not process the security header; instead, it passes the secured request message untouched to a business service. After the business service receives the message, it processes the security header and acts on the request. The business service must be configured with WS-Policy security statements. The secured response message is passed untouched back to the client. For example, the client encrypts and signs the message and sends it to the proxy service. The proxy service does not decrypt the message or verify the digital signature; it simply routes the message to the business service.

The business service decrypts the messages and verifies the digital signature, and then processes the request. The response path is similar.

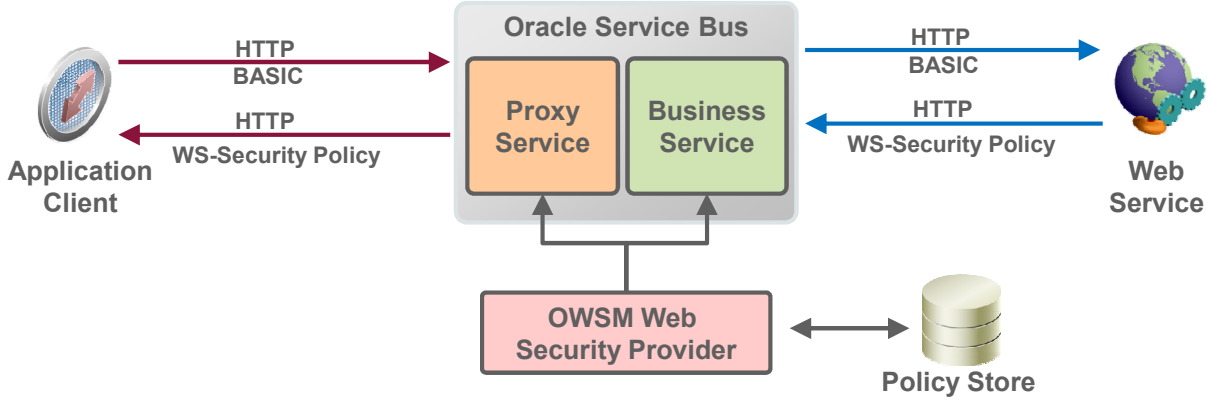
THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Securing Services with OWSM Policies

In Service Bus 12*c*, you can secure web services by using OWSM policies.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

In SB 12*c*, you use OWSM policies to secure the services. OWSM support in Oracle Service Bus is available automatically.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle Web Service Manager

- Oracle Web Service Manager (OWSM) is a security and management system that provides a common security infrastructure for web services applications.
- The Oracle Web Service Manager is based on three main operations:
  - Define
  - Enforce
  - Monitor

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Define consists of attaching security and management policies to the web services to be protected.

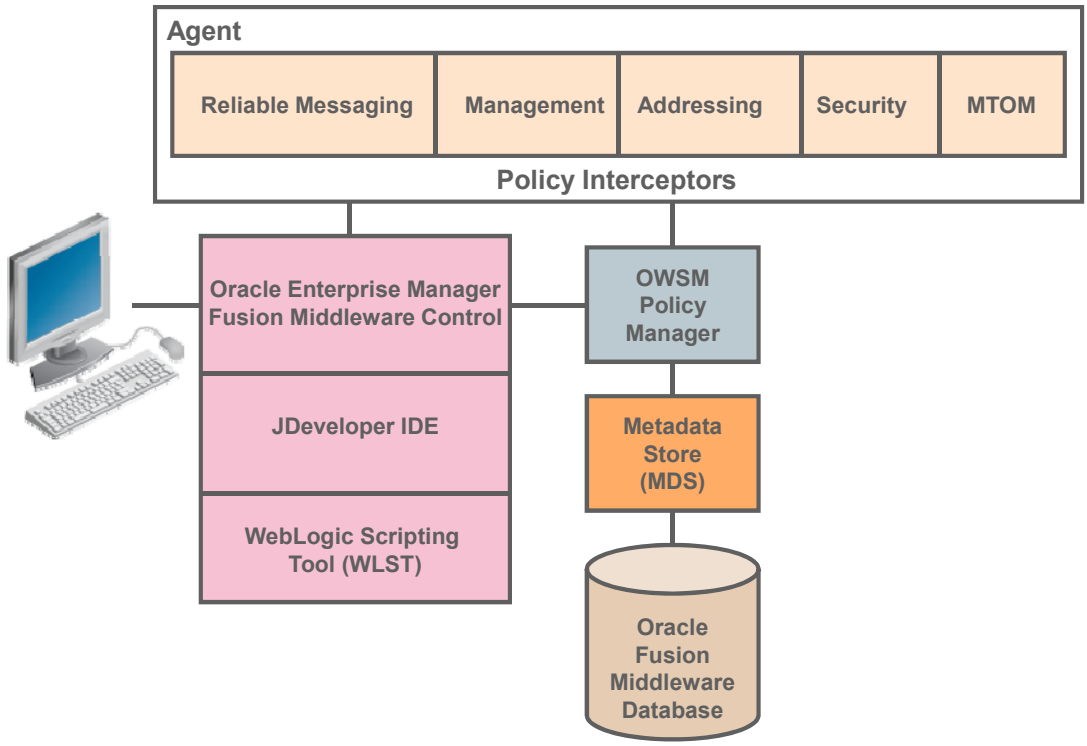
Enforce is the ability provided by Oracle Web Services Manager to distribute policies from a central policy manager to policy enforcement points that execute security and management policies at run time.

Monitor is the tracking of runtime security and management events captured by the Oracle Web Services Manager.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Components of Oracle Web Services Manager Architecture



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

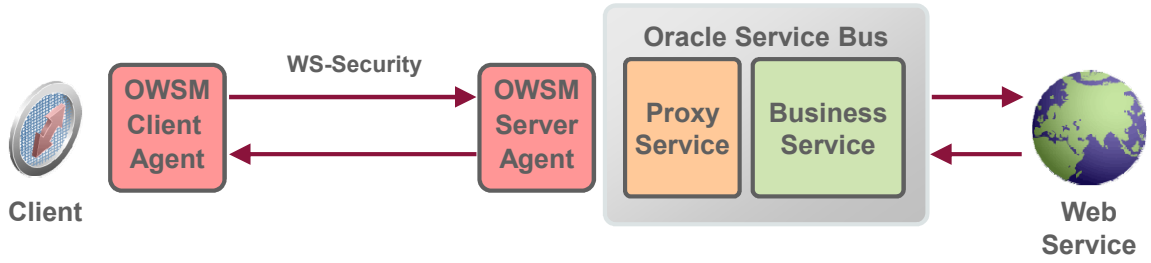
- **Oracle Enterprise Manager Fusion Middleware Control:** Enables administrators to access Oracle Web Services Manager's functionality to manage, secure, and monitor web services
- **WebLogic Scripting Tool (WLST):** Enables administrators to view and configure web services, and manage web service policies from the command line
- **Oracle Web Services Manager Policy Manager:** Reads and writes policies, including predefined and custom policies from the metadata store
- **OWSM Agent:** Manages the enforcement of policies via the Policy Interceptor Pipeline
- **Policy Interceptors:** Enforce policies, including reliable messaging, management, addressing, security, and Message Transmission Optimization Mechanism (MTOM)
- **Metadata Store:** Is used for storing policies. Policies can be stored either as files in the file system (supported for development) or to the Oracle Fusion Middleware database (supported for production).
- **Oracle Fusion Middleware Database:** Provides database support for the Metadata Services (MDS)

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Inbound Security by Using OWSM

A secured communication between a client and a proxy service by using OWSM agents



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

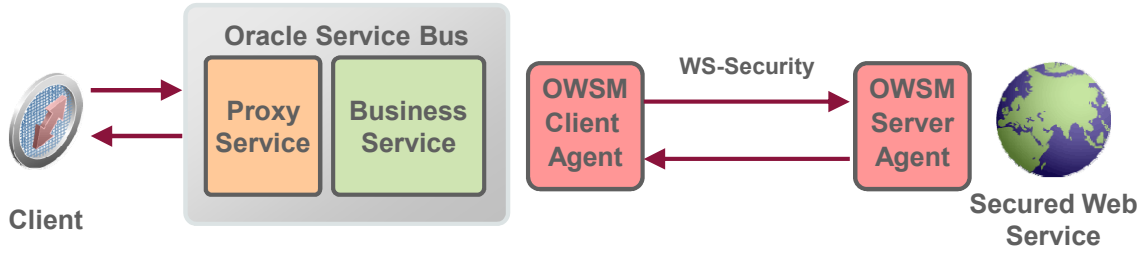
The proxy service has an inbound message protection policy by implementing an OWSM server agent. The OWSM Client Agent sends a signed and encrypted request to the proxy service. The proxy service receives the secured request and, acting as an active intermediary, decrypts and verifies signature and routes the request to the business service. The business service invokes the web service, gets the response back, and sends it to the proxy service. The proxy service signs and encrypts the response and sends it to the OWSM Client Agent. The Client Agent receives the secure response, decrypts and verifies the signature, and passes the response to the client.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Outbound Security by Using OWSM

A secured communication between a business service and a web service by using OWSM agents



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The SB business services might access different back-end services secured with different security infrastructures. The Oracle Web Services Manager Client Agent can be configured to enable the business services to securely invoke such back-end services.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Policy Support (12*c* 12.1.3.0)

- The following policies are supported:
  - WS-Policy 1.2 (default) and 1.5
  - WS-Security Policy 1.1 (default), 1.2, and 1.3
  - WS-Security: X.509 Token Profile 1.1
  - WS-Security: SAML Token Profile 1.1 (with SAML 2.0)
  - …and more
- The following policies are not supported:
  - Transport policies (HTTPS/SSL, SAML Bearer over SSL)

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

This slide outlines some of the policies that are supported and are not supported in SB 12*c*. See the documentation for other supported policies. You can find a complete list in the Securing Web Services and Managing Policies with Oracle Web Services Manager documentation: https://docs.oracle.com/middleware/1213/owsm/security/owsm-predefined-policies.htm#OWSMS5472

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

User Interface (UI) Support

Feature	SB Console	JDeveloper	FMW Console
Author OWSM policy			✓
Export custom OWSM policy			✓
Import OWSM policy			✓
Browse OWSM policies from a central policy store	✓	✓	✓
Attach/detach a OWSM policy to/from a proxy or business service	✓	✓	
Set policy overrides/configuration for service	✓	✓	
View currently attached OWSM policy on proxy/business service	✓	✓	
View effective WSDL that embeds OWSM policies	✓	✓	
View security statistics	✓		
View audit logs			✓

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Identity Propagation

- Identity Propagation: The mechanism to pass identity in the chain of interacting services.
- Service Bus's support for identity propagation includes:
  - Authenticate and authorize clients of the service bus
  - Pass security context through to service producers unchanged
  - Map credentials from one security domain to another

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

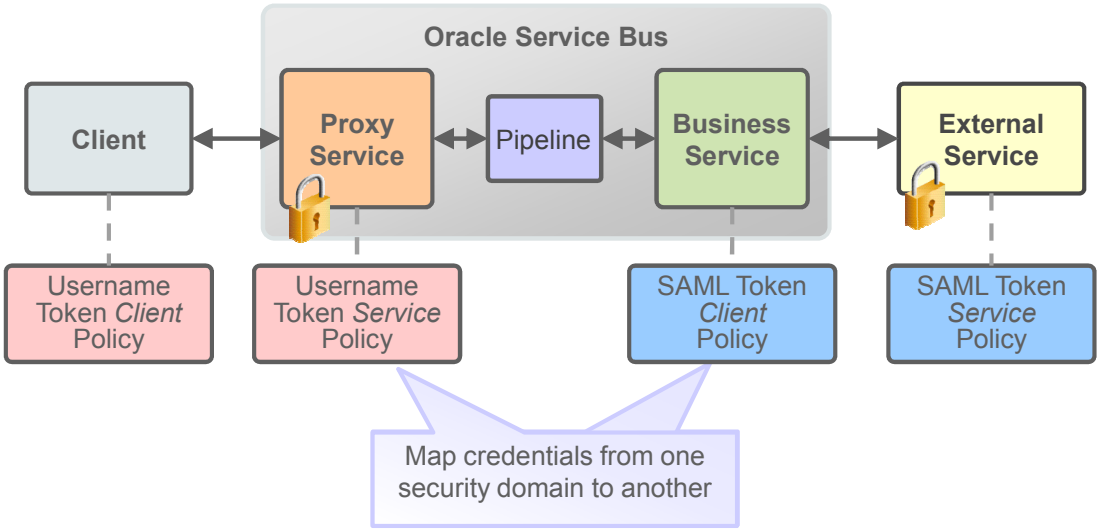
OWSM allows you to pass on the identity of the authenticated user to your OWSM protected web service, the username can then be used by your service. This will work on one or between different WebLogic domains.

The options provided by Service Bus for identity propagation allow for decision making when designing security, including how to propagate the identities that clients provide. Service Bus can be configured to authenticate the credentials provided by clients, perform authorization checks, pass credentials through as is, and map credentials.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Identity Propagation: Example



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

In the example, when the request reaches the SB Proxy service, which is secured with username token service policy, whereas the back-end service is protected by SAML token service policy. With Service Bus and OWSM, you can propagate the identity of the user through the invocation chain to the endpoint. Note that the back-end service can be a SOA service, a JAX-WS service or another OSB proxy service.

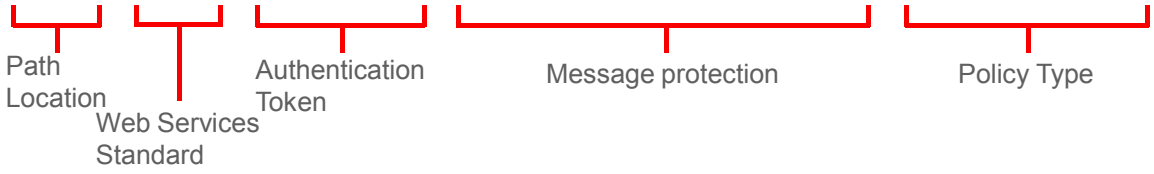
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

OWSM Predefined Policies

- A set of predefined policies are available by default.
- You can directly attach these predefined policies to your web services or clients.
- The naming conventions for security policies:

```
oracle/wss10_saml_token_with_message_protection_service_policy
```



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

There is a set of predefined policies that are automatically available when you install Oracle SOA Suite. The predefined policies are based on common best-practice policy patterns used in customer deployments.

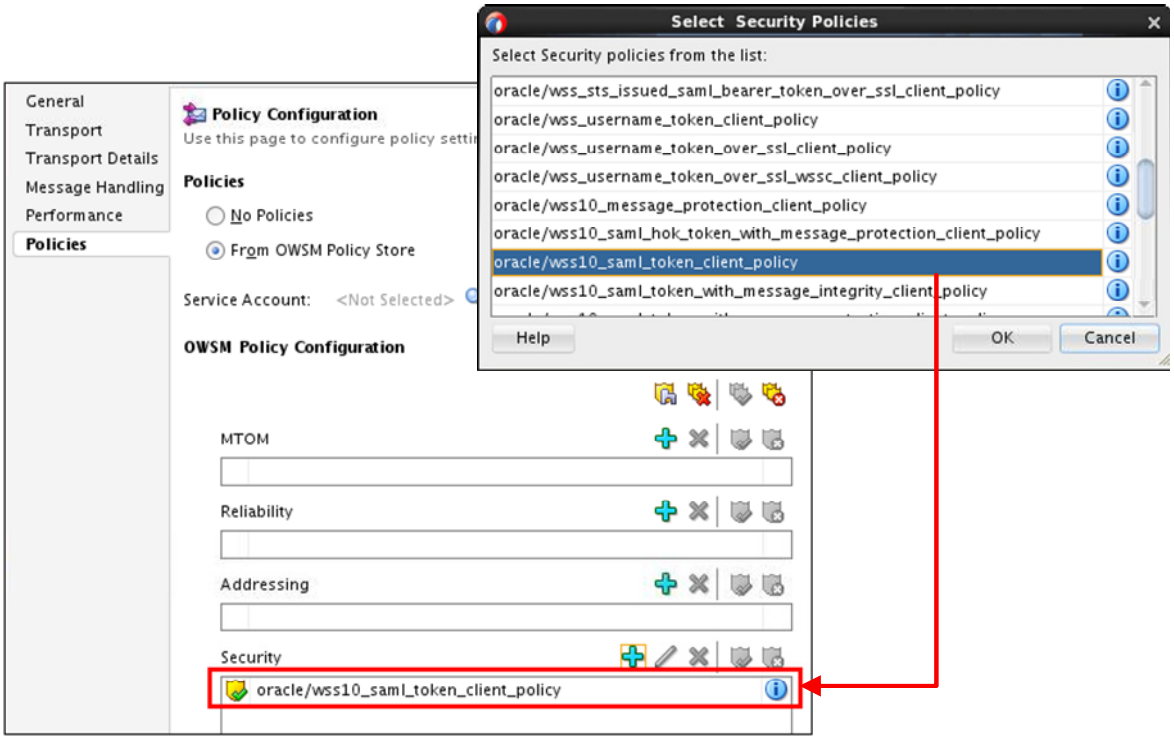
You can attach these predefined policies to your web services or clients, and configure them, or create a new policy by making a copy of one of the predefined policies.

Predefined policies are constructed by using assertions based on predefined assertion templates.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Applying OWSM Policies to Services in JDeveloper



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

OSB 12*c* supports attaching security policies:

- To proxy service and business service
- In JDeveloper and Service Bus Console

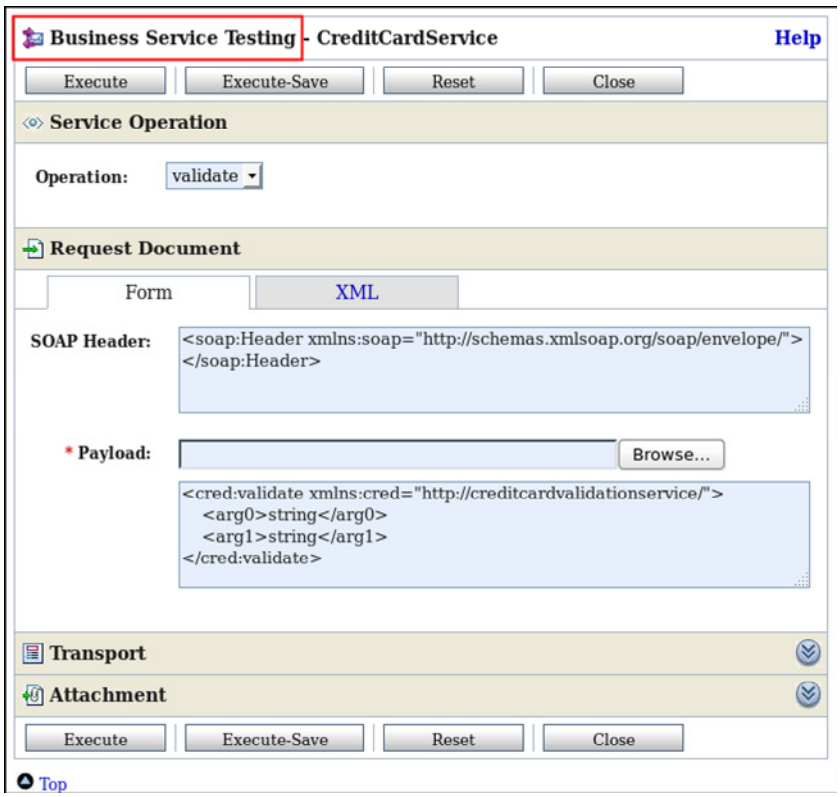
The example shows a policy attachment to a business service in JDeveloper.

All supported policies in policy store are visible in the Select Security Policy wizard. Even disabled policies are visible because they can be enabled on EM at a later stage.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Test Console Support: Business Service



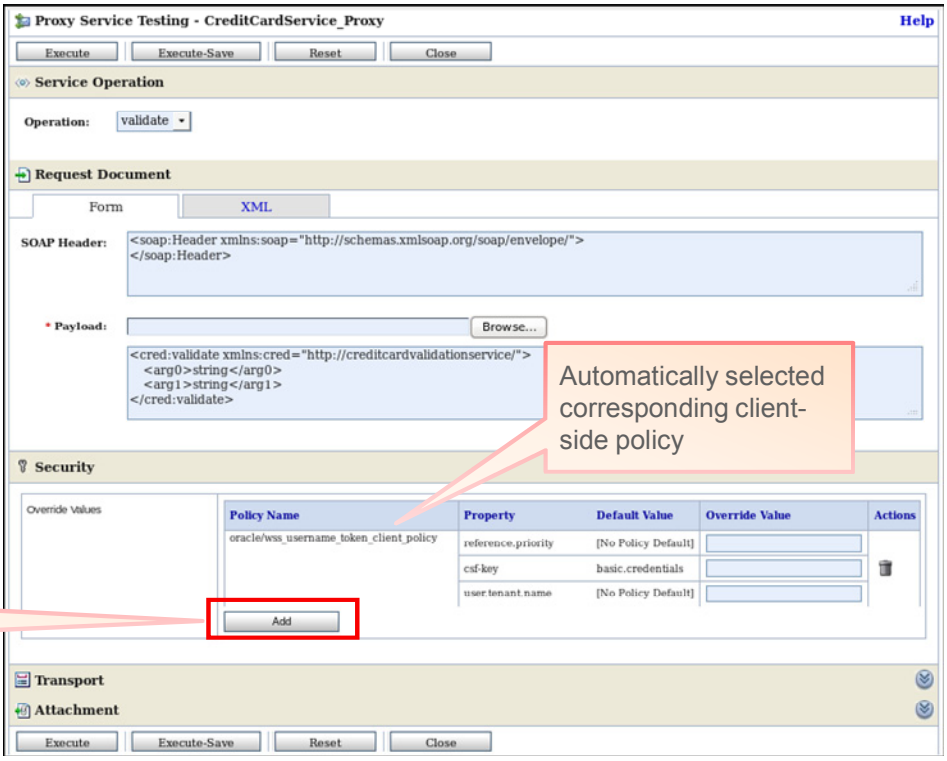
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When testing a business service by using the SB Test Console, no client-side policy is shown for the business service. The SB Test Console selects the policies attached on the business service. It also uses the override configuration set on the business service.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Test Console Support: Proxy Service



Automatically selected corresponding client-side policy

Click to change the policy

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When testing a proxy service by using the SB Test Console, the corresponding client-side policy is automatically selected and shown for the proxy service. You can change the default policy configuration by providing the override values for the specific properties, as shown in the slide.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Security concepts
- Oracle WSM security
- Access control policies

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Service Access Control Policies

- Policies determine which clients are authorized to access a proxy service, based on:
  - Role membership
  - Any of the conditions available to roles
- By default, all clients are granted access to a service.
- SOAP proxy services also support operation-level access policies if they are active intermediaries:
  - Configured to process WS-Security
  - Assigned a WS-Policy that requires authentication

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

An access control policy specifies conditions under which users, groups, or roles can access a proxy service. For example, you can create a policy that always allows users in the `GoldCustomer` role to access a proxy service and allows users in the `SilverCustomer` role to access the proxy service only after 12 PM on weeknights.

For all proxy services, you can create a transport-level policy, which applies a security check when a client attempts to establish a connection with the proxy service. Only requests from users who are listed in the transport-level policy are allowed to proceed.

A message-level access control policy applies a security check when a client attempts to invoke a proxy service with message-level security. Only users who are listed in the message-level policy are allowed to invoke the operation. You can create a message-level access control policy in the following cases:

- For proxy services that are web service security intermediaries that are active
- For proxy services that have custom authentication at the message level

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Role-Based Access: Application Security Roles

Application security roles are divided into three access types:

- Resource Management
- Administration Functions
- Session Management

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Application security roles provide access to Fusion Middleware Control and Oracle Service Bus Console features as long as the users are also members of the Oracle WebLogic Server Monitors group. You can assign application roles to users from the Service Bus Security page in Fusion Middleware Control.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Summary

In this lesson, you should have learned how to:

- Describe security concepts
- Compare transport-level and message-level security standards
- Describe Oracle Service Bus security features
- Use Service Bus and OWSM to secure web services
- Describe and assign access control policies to services

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Practice 12: Propagating Identity from Service Bus to a Web Service

This practice covers the following topics:

12-1: Configuring the Security Environment

12-2: Securing Back-end Web Service and Attaching Security Policy to Business Service

12-3: Applying a Security Policy to Proxy Service

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

13

Advanced Topics

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Objectives

After completing this lesson, you should be able to:

- List the best practices of using Service Bus
- Improve performance by caching service results
- Define SLA alert rules
- Explain how Oracle MFT works with OSB to handle large message transfers
- Describe message resequencing
- Build and deploy Service Bus projects with Maven

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Best practices
- Service result caching
- SLA alerts
- Integrating with MFT
- Message resequencing
- Continuous integration with Maven

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Principles and Best Practices of Using Service Bus

- Keeping business logic out of the OSB layer
- Implementing well-defined integration patterns in the OSB, such as message transformation and routing
- Exposing standards-based interfaces from the OSB, wrapping proprietary interfaces with web services or JMS interfaces
- Transforming proprietary message formats to common business objects and hiding proprietary interface behavior
- Always configuring the service-level error handler
- Using Actions like Delete, Insert, Replace for minor update to a document

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Always configure the service-level error handler. For non-SOAP services the built-in error handler only returns a transport error code because there is no standard error response document like a SOAP Fault that can be generated automatically.

Actions like Delete, Insert, Replace, and Rename are more efficient for minor fixes to a document than using Assign with an XQuery. The Assign action regenerates the entire document, which is not efficient especially if the document is large.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

BPEL Versus Service Bus

BPEL

- Orchestrating service calls
- Primarily for stateful and long running processes
- Implementing SCA composite services
- Integrating Business Rules and Human Workflow

Service Bus

- Implementing VETRO pattern
- Value mapping and cross-reference tables for canonical data models
- Stateless messaging

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

BPEL and Service Bus have some functional overlap between the two products, such as both can work with adapters, transform data, support the composite service pattern, and so on. However, each of them has its own strengths that makes it a more suitable candidate than the other to solve certain business problems.

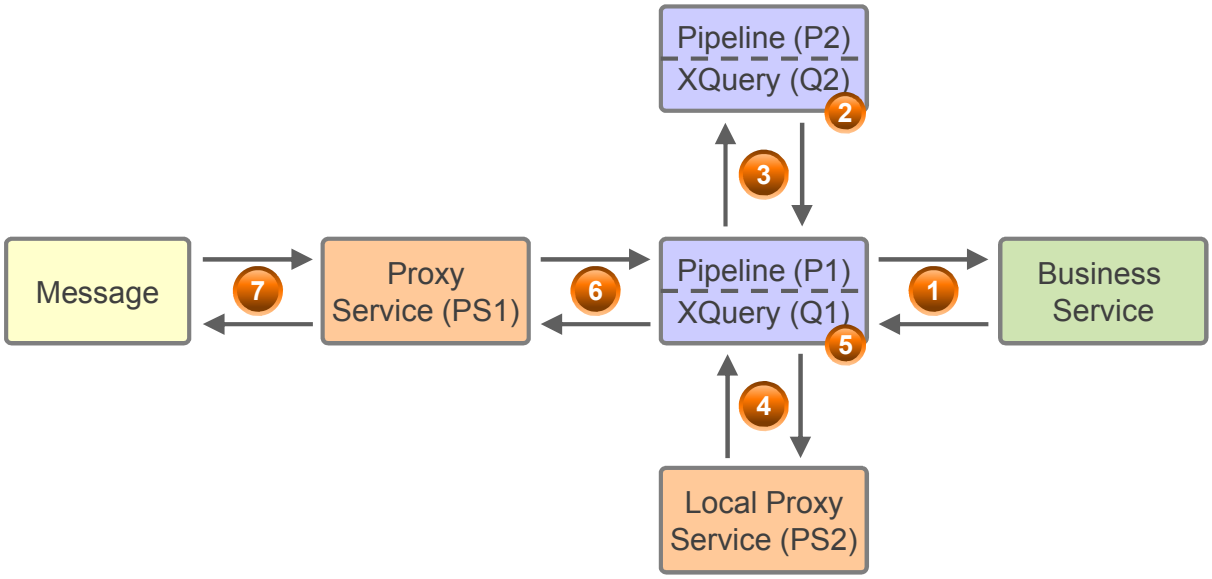
As shown in the table in the slide, BPEL is used for bringing together multiple services. It has much more functionality (for example, Business Rules) and allows implementation of complex business logic. Long-running business processes are stateful and can best be implemented using BPEL and/or BPMN.

One of the main strengths of Service Bus is processing messages. Message in, message out, perhaps with protocol or format mediations applied. It is good for routing messages to multiple destinations. It is also good for doing transformations that have little to no business rules. The footprint is much smaller and incurs minimal overhead; therefore, the performance is much better. In addition, Service Bus is designed to be able to handle large volumes of messages given its stateless transactional nature.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Testing Approaches



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Complete the testing of interfaces other than the initial proxy service first. This means that you complete the testing of interfaces 1 through 6 first, then test interface 7.

Generally, you want to test in the reverse order of the way a message would flow through the system. In this way, the message flow logic in the pipelines can be iteratively changed and tested knowing that the downstream interfaces function correctly.

- Test the pipeline (P1) to business service interface (1).
- Validate and test all the XQuery expressions in the pipelines before testing the pipelines themselves. In the figure in the slide, interfaces 2 and 5 refer to XQuery expression tests.
- Test the pipeline (P1) to pipeline (P2) interface (3).
- Test the pipeline (P1) to local proxy service (PS2) interface (4).
- Test the initial proxy service (PS1) to pipeline (P1) interface (6).
- Make your final system test simulate the client invoking the proxy service PS1. This test is represented by interface 7.

Save the message state after executing successful interface tests to facilitate future troubleshooting efforts on the system. Testing interface 6 is in fact a test of the complete system. Knowing that all other interfaces in the system work correctly can help narrow the troubleshooting effort when system errors arise.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Best practices
- Service result caching
- SLA alerts
- Integrating with MFT
- Message resequencing
- Continuous integration with Maven

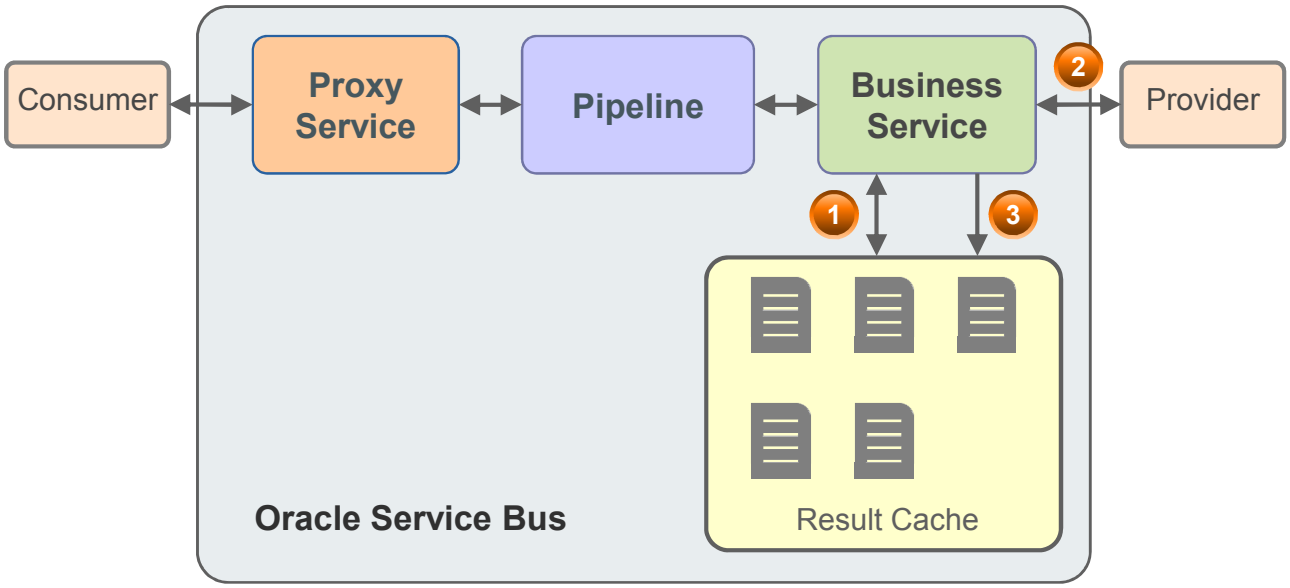
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Service Result Caching

Requirement: Handle a lot of read-only requests to a service with limited capability.



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

You can store the result of a given request in a cache instead of calling the back-end system for each product request. The cache is then used to retrieve data that has already been requested when another request for the same information arrives.

Oracle Service Bus allows for declaratively adding a result cache on a Business Service. This way, every request on the Business Service first checks if the desired information is already in the cache (1). If so, the data is directly returned to the consumer. If not, the request is sent to the back-end system (2), and the response is stored in the cache (3) and returned to the consumer.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Use Cases of Result Caching

- Synchronous services that return results that do not change often
- Handle a lot of read-only calls to a system with limited scalability

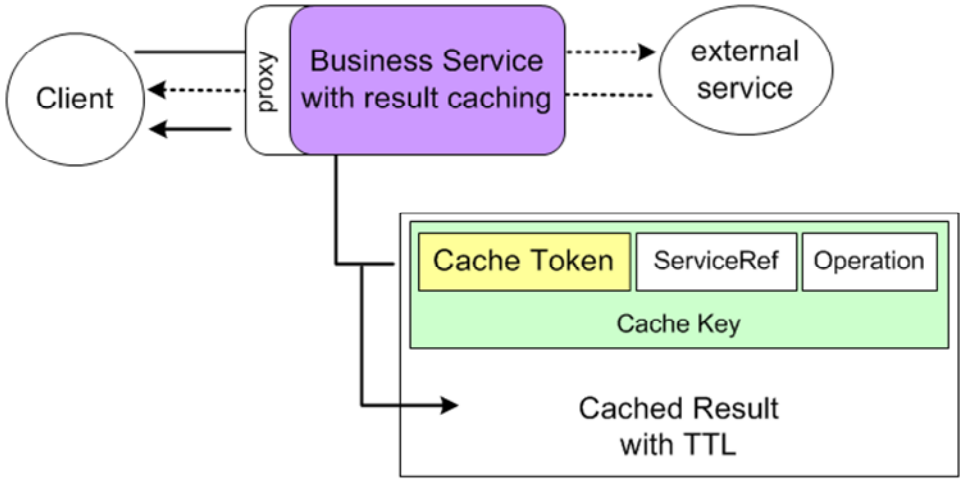
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Result Caching improves performance by reducing network overhead to access the back-end service. Result caching can help to improve scalability by reducing the load on the back-end servers that host the service. Service Result caching is used when we have business service that connects to external service, which returns somewhat static response. So by using Service Result Caching we do not hit external service for the same request, instead it takes the response from cache, which improves the OSB performance.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

How Does Result Catching Work?



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

The result cache uses a cache key to identify cached results, and an expiration time to determine when to flush cached results.

A cache key is made up of the ServiceRef (the unique identifier for the service, which is the fully qualified path name of the service), the operation being invoked, and a cache token string.

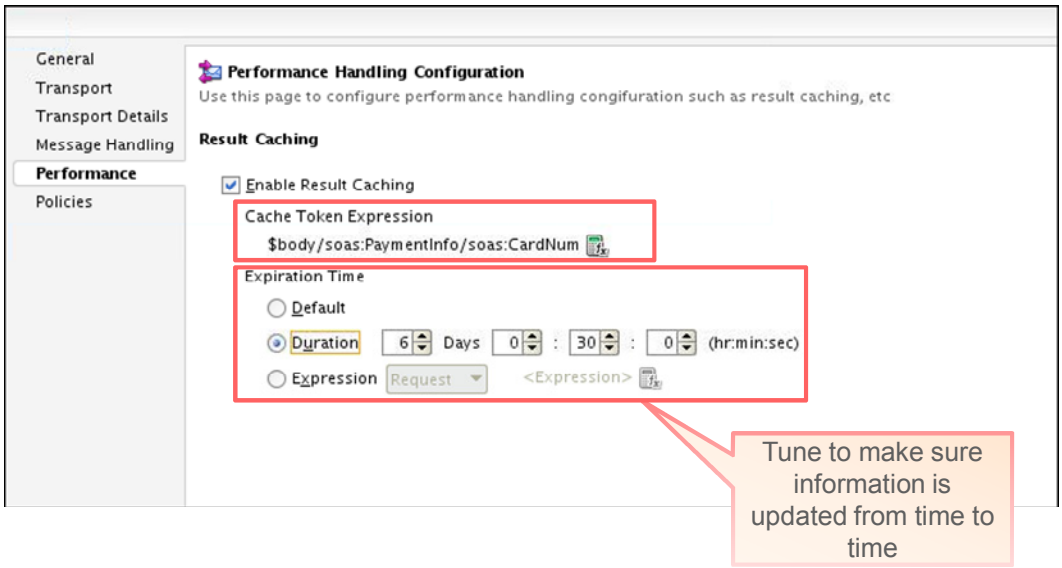
For cache expiration, cached results have a time-to-live (TTL) attribute. If Coherence finds that the TTL has expired, it flushes the cache, and the business service invokes the external service for a result. That result is then stored in the cache (if there is no error in the result), and the result is available in the cache so that it can be returned to the next request.

In the diagram in the slide, the solid arrows represent the message path between the client and a cached result. The dotted arrows show the message path if no cached result exists.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Configuring a Business Service for Result Caching



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

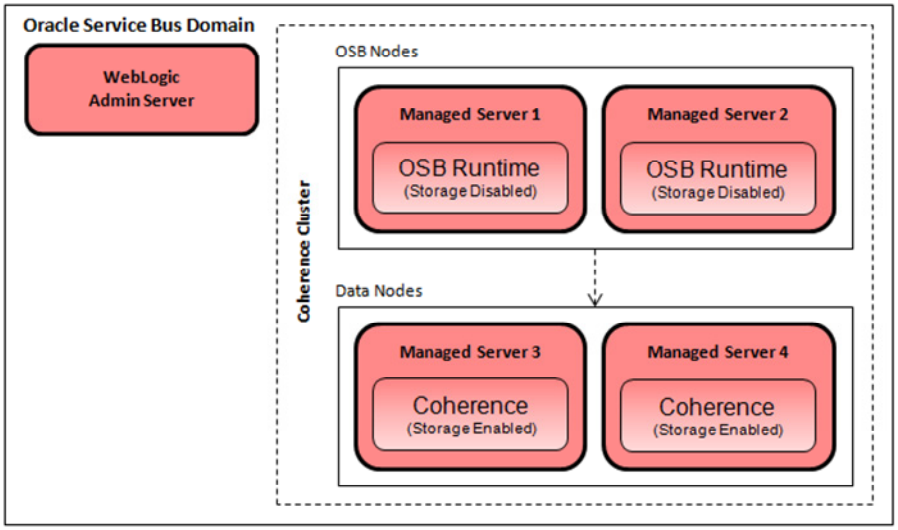
You can set the cache token either by configuring the cache-token expression in the result caching configuration for the business service or by using the cache-token metadata element in `$transportMetaData` using the pipeline.

You can configure cache expiration either with the Expiration Time property in the result caching configuration on the business service or the `cache-ttl` element in `$transportMetaData` using the pipeline.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Using an Out-of-Process Coherence Cache Server



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Using Result Catching has side effects. When this feature is activated, all results are cached in the JVM heap. That means that the heap can rapidly become full after several service invocations occur. This could lead to serious garbage collection issues once the JVM starts to reclaim the used space when it hits the high-water mark of 80% of the heap size. To avoid using too much heap space with Result Caching, out-of-process Coherence servers can be set up to run in their own JVMs to hold the cached results. The technique here is to allocate data off the OSB JVM letting the Coherence servers use their own heap space without affecting the heap space OSB uses to process messages. The figure in the slide shows an OSB domain using out-of-process Coherence servers to hold the cached results.

Besides avoiding GC issues in the OSB JVM, another great advantage of using out-of-process Coherence servers is data density increase. Instead of having each OSB cluster member storing data on its own JVM heap space, data can be stored in a shared storage layer distributed across the cluster. Considering the total amount of data to be cached per service, this means a considerable reduction in the amount of memory required.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Coherence Configuration Files

- The configuration files are:
  - `osb-coherence-override.xml`:
    - Defines the operational settings for Coherence
    - Enables configuring Coherence clusters
  - `osb-coherence-cache-config.xml`:
    - Defines the cache configuration
- Both files are located in the *`DOMAIN_HOME`*`/config/osb/coherence` directory.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle Service Bus provides its own default Oracle Coherence configuration file for the servers in a domain. The `osb-coherence-override.xml` configuration file provides different properties (such as Unicast/Multicast address and Unicast/Multicast port to configure Coherence cluster) that control server access to Oracle Coherence cache.

You can tune cache schemes for the Oracle Service Bus result cache by modifying the `osb-coherence-cache-config.xml` file. By default, a distributed cache scheme is used for the Oracle Service Bus result cache.

You must be well acquainted with Oracle Coherence to modify these configuration files.

For more information, refer to the *Oracle Coherence Developer's Guide*.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Best practices
- Service result caching
- SLA alerts
- Integrating with MFT
- Message resequencing
- Continuous integration with Maven

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

SLA Alert in OSB

- A service-level agreement (SLA) is a contract between a service provider and a service consumer:
  - Min/max response time
  - Message count
  - Error count
- SLA alert provides insight to health metrics of services, such as response time, errors, or even load.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

A service-level agreement (SLA) is a contract between a service provider and a service consumer. In OSB monitoring framework, you have SLA alerts that come into picture when there is violation of SLAs.

SLA alerts are automated responses to violations of SLAs. These alerts are displayed on the OSB dashboard. They are generated when the service violates the SLA or a predefined condition.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

SLA Alert Rules

- SLA alerts:
  - Consist of a set of conditions, or “rules”
  - Are automatically evaluated after each service’s aggregation interval
- Define SLA alert rule to measure service health metrics:
  - Min/max response time
  - Message count
  - Error count

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

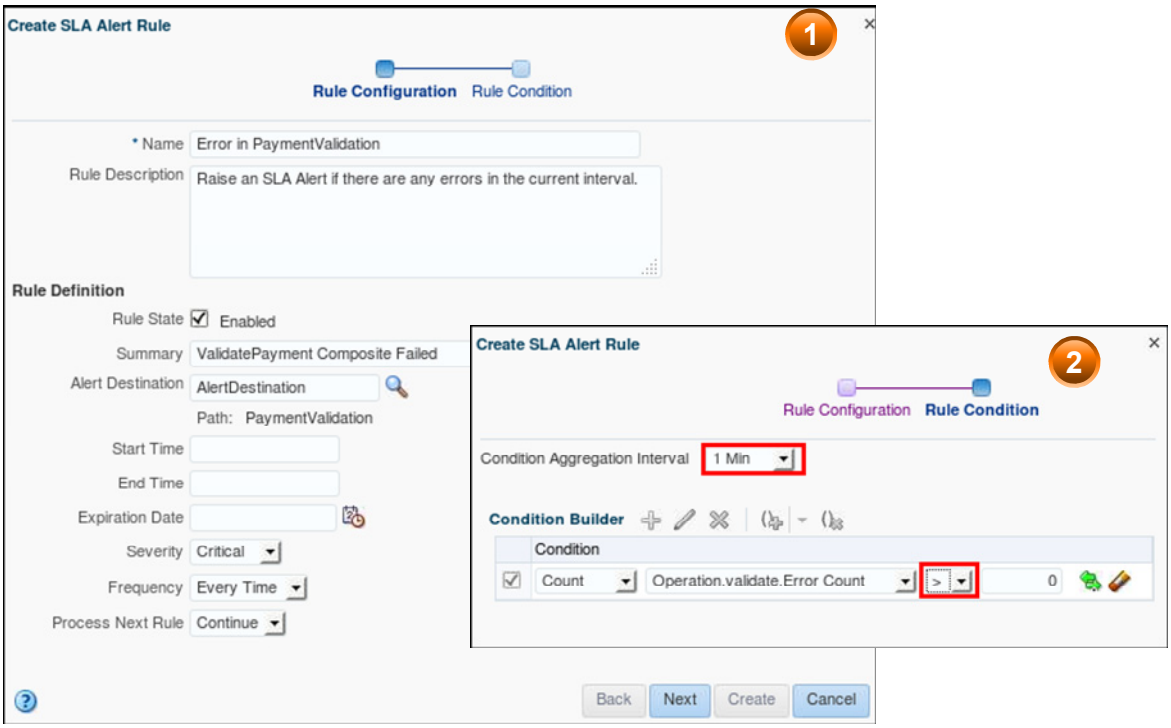
SLA alert rules define conditions under which an alert is generated. These conditions are typically indicators of the overall health of the service Bus application or of a specific service component.

You can define an SLA alert rule based on throttling metrics, such as the average time spent by all messages in the throttling queue.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Creating SLA Alert Rules



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Best practices
- Service result caching
- SLA alerts
- Integrating with MFT
- Message resequencing
- Continuous integration with Maven

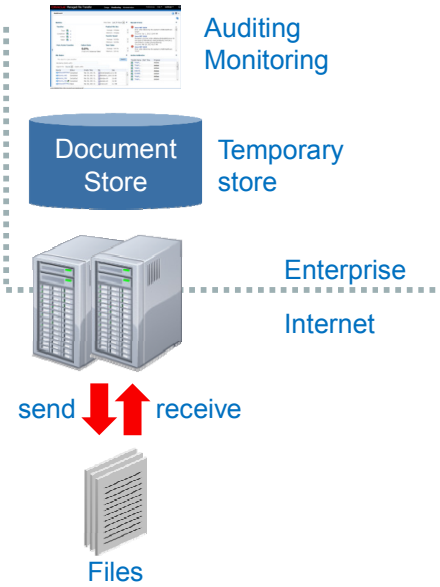
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Managed File Transfer (MFT)

Simple and secure end-to-end managed file gateway



- Centralized file transfer infrastructure
- Supports:
  - Large file transfer
  - Pass-by-reference
  - Encryption
  - Auditing and monitoring
  - Scheduling
- Standard-based integration with SOA, OSB, B2B
- Highly-available/clusterable

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Traditional OSB implementations used File/FTP adapters for file transfers. There is no visibility into the transfers and scheduling functionality.

Oracle Managed File Transfer (MFT) is a simple and secure end-to-end managed file gateway that provides global visibility into all file exchanges.

MFT:

- Share files between applications. It supports event-driven file delivery, batch, scheduled, and large files.
- Consolidates and simplifies file transfer integrations, and avoids custom scripts or incorrect tooling
- Enforces security and policy compliance with partners through built-in auditing and encryption/decryption
- Provides standards-based middleware integration with SOA, B2B, service bus, and so on

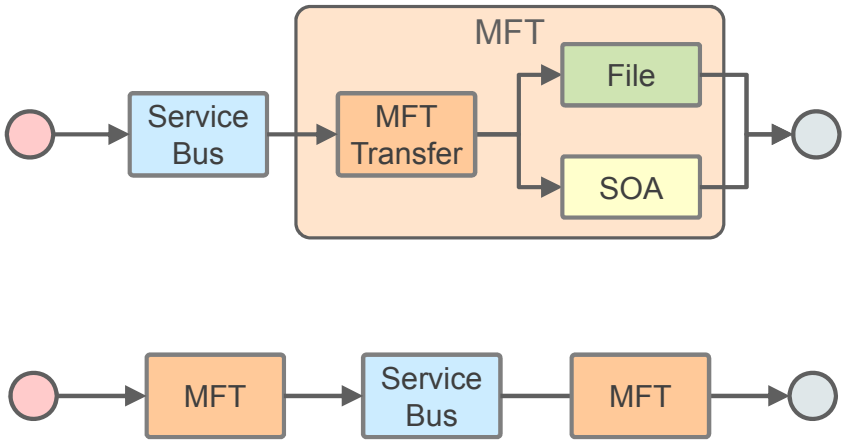
Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

OSB and MFT Integration

Oracle Service Bus interface can be:

- The source or target of a transfer
- The common endpoint for the target of one transfer and the source of another



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle Managed File Transfer can help integrate applications by transferring files between them in complex use case patterns. This slide shows a couple common OSB and MFT integration patterns.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Best practices
- Service result caching
- SLA alerts
- Integrating with MFT
- Message resequencing
- Continuous integration with Maven

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Resequencing Message

- The resequencer in Service Bus rearranges a stream of related but out-of-sequence messages into a sequential order.
- Resequencing is configured inside a pipeline component.
  - Only supports one-way WSDL-based service
- Resequencing order:
  - **Standard:** Is used when incoming message contain numeric identifier
  - **FIFO:** Is based on message arrival time
  - **Best Effort:** Is used for applications that produce a large number of messages in a short period and cannot provide identifier information.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

When incoming messages arrive, they may be in a random order. The resequencer orders the messages based on sequential or chronological information, and then sends the messages to the target services in an orderly manner.

Pipelines with the following service types are supported:

- **WSDL:** Resequencing is available for operations with only request type.
- **Message Type:** The request message type should be XML, and the response message type should be None.

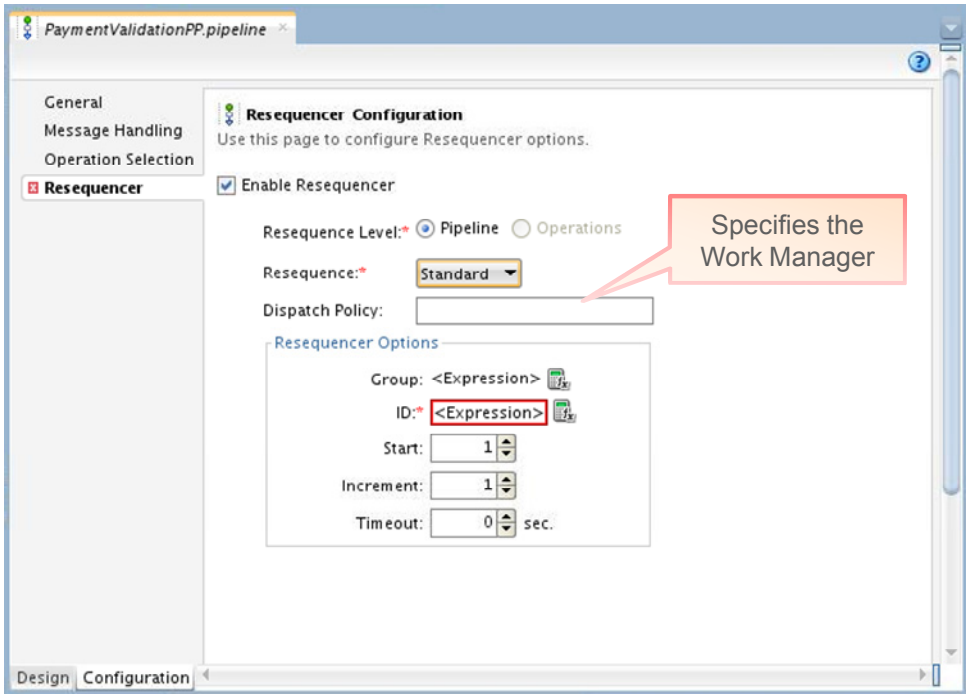
Service Bus resequencing allows you to specify three types of resequencing orders:

- **Standard:** Is used when incoming message contains numeric identifier. This numeric identifier is termed as Sequence ID. The standard resequencer receives a stream of messages that might not arrive in order; it then stores the out-of-sequence messages until a complete sequence based on the Sequence IDs is received. The in-sequence messages are then processed asynchronously based on their sequence ID.
- **FIFO:** Is based on message arrival time. It supports a standard first in, first out (FIFO) pattern. Here you need to specify only Group. So all the messages that are there in one group are processed based on message receive time.
- **Best Effort:** The best effort pattern is useful for applications that produce a large number of messages in a short period and cannot provide information to the resequencer about the identifier to use for sequencing. Typically, the identifier used for sequencing in such scenarios is of a `dateTime` type or `numeric` type.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Configuring the Resequencer



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Resequencing is a new feature that is provided in Oracle Service Bus 12*c* release. You can set this feature either at pipeline level or at operation level. Configuring the Resequencer is done on the Configuration tab of the Pipeline Editor.

- **Resequence Level:**
  - **Pipeline:** A common configuration specified at the component level is used to resequence all messages. If a component has multiple operations, then messages for each operation are sequenced separately using the common component configuration.
    Component-level resequencing is allowed only when all the operations of the pipeline component support request one-way messages. If only a subset of operations support request one-way messages, then you can individually specify operation-level resequencing for these operations.
  - **Operations:** For a WSDL-based pipeline, resequencing can be configured at the operation level. Each operation can have a different resequencer configuration. Only operations supporting request one-way messages can be resequenced. Non-WSDL pipelines cannot have resequencer configured at the operation level.

Oracle University and ORACLE CORPORATION use only

- **Resequence Mode:**
  - Standard
  - FIFO
  - Best Effort
- **Dispatch Policy:** Specifies the Work Manager to use

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Work Manager in Service Bus

- WebLogic Server uses Work Manager to prioritize work and allocate threads for the applications/components.
- In Service Bus (OSB), several transports for proxy and business services provide a configuration option called *Dispatch Policy* that enables a Work Manager to associate a service to prioritize service work.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Service Bus uses Oracle WebLogic Server Work Managers to optimize performance and to maintain SLAs. Work Managers prioritize work and allocate threads based on rules you define and based on runtime performance. When you create and configure a Work Manager, you define the maximum and minimum number of threads to use, server capacity, and request and response classes that express scheduling guidelines. One default Work Manager is provided, but you can create as many Work Managers as you need to optimize your services. In Service Bus, you specify a Work Manager for a proxy service or business service in the Dispatch Policy property of the transport configuration.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Agenda

- Best practices
- Service result caching
- SLA alerts
- Integrating with MFT
- Message resequencing
- Continuous integration with Maven

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

About Maven

Maven is essentially a project management and comprehension tool and provides a way to help with managing:

- Builds
- Documentation
- Reporting
- Dependencies
- SCMs
- Releases
- Distribution

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Maven is inherently project-centric in that everything resolves around the notion of a project. A pom.xml contains the Project Object Model (POM) for the project. The POM is the basic unit of work in Maven.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Concepts and Terminology

- **Build Life Cycle:** Made up of phases, like validate, compile, test, package, and so on
- **Repositories:** Hold build artifacts and dependencies of varying types
- **Plug-in:** Maven is a plug-in execution framework; all work is done by plug-ins
- **Goals:** A specific task, which can be mapped to one or more phases
- **Artifact** (deployment unit)**:** Library or plug-in, identified by groupID + artifact ID + version + signed (signature)

Example:

```
mvn com.oracle.maven(groupID):oracle-maven-
sync(artifact ID):push(goal)
-DoracleHome=$ORACLE_HOME
```

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

- Maven is based around the central concept of a build life cycle. A build life cycle is made up of "phases." There are three built-in build life cycles: default, clean, and site. For example, the "default" life cycle comprises the following phases (not a complete list): validate, compile, test, package, and so on. The phases are executed sequentially to complete the "default" life cycle. It means one only needs to call the last desired build phase, for example, "mvn package", then all phases before "package" in the default life cycle including "validate," "compile," "test," as well as the last phase "package" will all be executed one by one.
- A repository in Maven is used to hold build artifacts and dependencies of varying types. There are two types of repositories: local and remote.
- Maven is—at its heart—a plug-in execution framework; all work is done by plug-ins. In addition to Maven-provided plug-ins, there are many third party plug-ins too. Plug-in behavior can be customized via plug-in goal(s), which are bound to specific stages in the life cycle.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Concepts and Terminology

- POM files: Define Maven projects with a description of the project's artifacts:
  - Plug-ins to use
  - Inheritance
  - Dependencies on other artifacts
- Archetypes: A template for creating a specific type of project

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Maven's library and plug-in is called artifact (deployment unit), which is identified by groupID + artifact ID + version + signed (signature).

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

OSB Maven Support

- OSB provides:
  - A plug-in for pulling/downloading all requires libraries and plug-ins into local repository
  - A plug-in to package and deploy an OSB project
  - An archetype to create an OSB project
- Service Bus Maven plug-in goals:
  - Package
  - Deploy

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

With Oracle SOA Suite 12*c*, you can build all SOA and Service Bus projects with Maven natively, without calling a utility like configjar or ANT from Maven.

The Oracle Service Bus development Maven plug-in provides goals that are specific to Service Bus projects and that can be used in conjunction with standard Maven goals to build and manage Service Bus projects and applications throughout the lifecycle phases.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Summary

In this lesson, you should have learned how to:

- List best practices of using Service Bus
- Improve performance by caching service results
- Define SLA alert rules
- Explain how Oracle MFT works with OSB to handle large message transfers
- Describe message resequencing
- Build and deploy Service Bus projects with Maven

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Practice 13: Overview

- 13-1: Configuring Result Caching to Improve Service Performance
- 13-2: Adding a Service Level Agreement Alert
- 13-3: Building and Deploying Service Bus Projects with Maven (Optional)

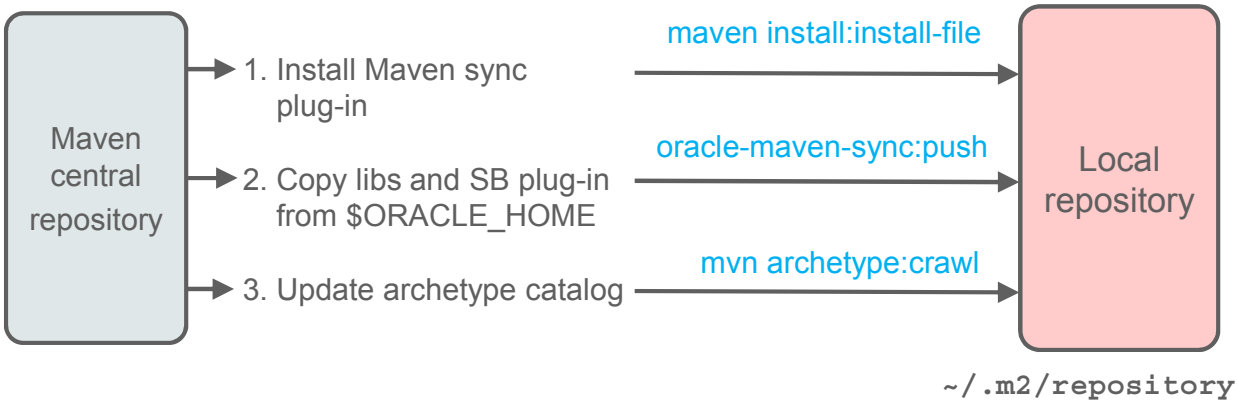
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Building Service Bus Projects with Maven

- Maven 3.0.5 bundled in JDeveloper 12.1.3:
  `$FMW_DEV_HOME/oracle_common/modules/org.apache.maven_3.0.5`
- OSB plug-ins:
  `$FMW_DEV_HOME/oracle_common/plugins/maven/com/oracle/maven/oracle-maven-sync/12.1.3`



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Oracle University and ORACLE CORPORATION use only

THESE eKIT MATERIALS ARE FOR YOUR USE IN THIS CLASSROOM ONLY. COPYING eKIT MATERIALS FROM THIS COMPUTER IS STRICTLY PROHIBITED

Oracle University and ORACLE CORPORATION use only