

R12.x Implement Oracle Workflow

Volume II - Student Guide

D58320GC10
Edition 1.0
October 2009
D66009

ORACLE®

Copyright © 2009, Oracle. All rights reserved.

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.

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.

Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

Authors

Clara Jaeckel

Technical Contributors and Reviewers

Leta Davis, Donna Johnson, Shivasmruthi Narayanasamy

This book was published using: oracletutor****

Table of Contents

Introduction to Oracle Workflow	1-1
Introduction to Oracle Workflow	1-3
Objectives	1-4
Enabling E-Business	1-5
Inter-Enterprise Business Processes	1-6
Traditional Workflow	1-7
Workflow-Driven Business Processes.....	1-8
Sample Workflow Process.....	1-10
Event-Based Workflow	1-11
Subscription-Based Processing.....	1-12
System Integration with Oracle Workflow	1-13
Business Process-Based Integration	1-14
Supported System Integration Types.....	1-15
Designing Applications for Change.....	1-16
Designing Applications for Integration	1-19
Summary.....	1-21
Oracle Workflow Components.....	2-1
Oracle Workflow Components	2-3
Objectives	2-4
Oracle Workflow Architecture	2-5
Oracle Workflow Components	2-9
Workflow Engine	2-10
Workflow Processes	2-11
Supported Process Constructs.....	2-13
Oracle Workflow Builder	2-17
Business Event System Architecture	2-18
Business Event System Components	2-19
Advanced Queuing, an Enabling Technology	2-20
Oracle Database Communication Alternatives.....	2-21
Accessing Oracle Workflow Web Pages	2-23
Oracle Workflow Home Pages	2-24
Notification System	2-25
Worklist Web Pages	2-26
E-Mail Notifications	2-27
Directory Services	2-28
Status Monitor Web Pages.....	2-29
Workflow Definitions Loader.....	2-30
Workflow XML Loader.....	2-31
Workflow Manager.....	2-32
Service Components	2-33
Oracle Workflow Documentation.....	2-34
Summary.....	2-35
Planning a Workflow Process.....	3-1
Planning a Workflow Process.....	3-3
Objectives	3-4
Workflow Process Components	3-5
Oracle Workflow Builder	3-7
Standard Activities	3-8
Planning a Workflow Process.....	3-10
Activity Results and Lookup Types.....	3-13
Summary.....	3-15

Diagramming a Workflow Process	4-1
Diagramming a Workflow Process.....	4-3
Objectives.....	4-4
Creating a New Workflow Process.....	4-5
Creating a New Process from Top Down	4-6
Creating a New Process from Bottom Up	4-7
Diagramming a Process	4-8
Quick Start Wizard	4-9
Defining an Item Type.....	4-12
Defining a Process Activity	4-14
Diagramming a Process	4-16
Top-Down Design	4-17
Diagramming a Process	4-18
<Default> Transition	4-19
<Any> Transition	4-20
Editing a Transition	4-21
Self-Looping Transitions	4-23
Customizing an Activity Node	4-24
Show Label in Designer Menu Options.....	4-26
Display Modes.....	4-27
Verifying a Process Definition	4-28
Validation Performed by the Verify Command.....	4-29
Saving Process Definitions.....	4-31
Summary.....	4-32
Defining Item Type Attributes and Lookup Types.....	5-1
Defining Item Type Attributes and Lookup Types.....	5-3
Objectives.....	5-4
Defining Item Type Attributes.....	5-5
Attribute Data Types	5-7
Defining Item Type Attributes.....	5-9
URL Attributes	5-10
Form Attributes	5-13
Document Attributes.....	5-16
Deleting Item Attributes	5-19
Defining Lookup Types	5-20
Defining Lookup Codes.....	5-21
Summary.....	5-22
Defining Messages and Notification Activities	6-1
Defining Messages and Notification Activities	6-3
Objectives	6-4
Notification Activities	6-5
Defining a Message	6-6
Defining a Message Attribute	6-8
Defining a Respond Attribute	6-10
Defining a Message Result	6-11
Notification Details Web Page	6-12
HTML E-mail Notifications	6-13
Plain Text E-mail Notifications Using Templatized Response	6-14
Plain Text E-mail Notifications Using Direct Response	6-15
Defining a Notification Activity.....	6-16
Standard Voting Activity	6-18
Defining a Notification Activity Node	6-19
Defining a Timeout for a Notification	6-20
Defining a Dynamic Priority for a Notification	6-21
Defining a Performer for a Notification	6-22
Special Message Attributes.....	6-23

Action History	6-30
Special Message Function	6-31
Embedding Oracle Application Framework Regions in Notifications	6-33
Summary.....	6-34
Testing and Monitoring Workflow Processes	7-1
Testing and Monitoring Workflow Processes.....	7-3
Objectives	7-4
Testing Workflow Processes	7-5
Administrator Monitor.....	7-8
Viewing Workflows in the Administrator Monitor	7-10
Viewing Activity History in the Administrator Monitor	7-12
Viewing a Status Diagram in the Administrator Monitor.....	7-13
Viewing Responses in the Administrator Monitor	7-15
Viewing Workflow Details in the Administrator Monitor.....	7-17
Viewing Error Information in the Administrator Monitor.....	7-18
Viewing Child Workflows in the Administrator Monitor	7-19
Self-Service Monitor	7-20
Viewing Workflows in the Self-Service Monitor.....	7-21
Viewing Notification History in the Self-Service Monitor.....	7-23
Viewing a Status Diagram in the Self-Service Monitor	7-24
Viewing Responses in the Self-Service Monitor	7-26
Viewing Error Information in the Self-Service Monitor	7-28
Summary.....	7-29
Viewing and Responding to Notifications.....	8-1
Viewing and Responding to Notifications.....	8-3
Objectives	8-4
Viewing Notifications from a Web Browser	8-5
Worklist Pages.....	8-6
Advanced Worklist	8-7
Notification Details.....	8-9
Reassigning Notifications.....	8-12
Responding to a Group of Notifications	8-14
Requesting More Information.....	8-15
Certificate-Based Digital Signatures	8-16
Granting Worklist Access.....	8-18
Administrator Search for Notifications.....	8-22
Reviewing Electronic Signature Details	8-24
Personal Worklist	8-26
Simple Search for Notifications.....	8-28
Advanced Search for Notifications.....	8-29
Creating a Personal Worklist View	8-31
Viewing Notifications Through E-mail	8-33
E-mail Response Methods	8-35
HTML-Formatted E-mail Notifications.....	8-36
Plain Text E-mail Notifications Using Templatized Response	8-39
Plain Text E-mail Notifications Using Direct Response	8-41
Viewing an E-mail Summary of Notifications	8-43
Vacation Rules.....	8-44
Maintaining Vacation Rules	8-45
Defining Vacation Rules	8-46
Summary.....	8-49
Worklist Flexfields.....	9-1
Worklist Flexfields	9-3
Objectives	9-4
Worklist Flexfields	9-5

Benefits of Worklist Flexfields.....	9-6
Defining a Specialized Worklist View Using Worklist Flexfields	9-7
Message Attributes in Worklist Flexfields Rules	9-8
How Worklist Flexfields Rules Operate	9-9
Phase Numbers	9-10
Customization Levels	9-11
Core Rules	9-12
Limit and User Rules	9-14
Combining Core, Limit, and User Rules	9-17
Worklist Flexfields Rules Example	9-18
Defining a Worklist Flexfields Rule: Entering General Properties	9-20
Defining a Worklist Flexfields Rule: Selecting Filter Criteria	9-22
Defining a Worklist Flexfields Rule: Selecting Message Attributes	9-23
Defining a Worklist Flexfields Rule: Mapping Attributes to Columns	9-25
Resolving Conflicts Between Worklist Flexfields Rules.....	9-27
Maintaining Worklist Flexfields Rules.....	9-28
Storing Message Attribute Values in Worklist Flexfields Columns	9-30
Performing a Worklist Flexfields Rule Simulation	9-31
Defining a Securing Function	9-34
Creating a Personalized View for the Personal Worklist.....	9-35
Restarting Oracle HTTP Server.....	9-37
Summary.....	9-38
Oracle Workflow Directory Service.....	10-1
Oracle Workflow Directory Service	10-3
Objectives	10-4
Oracle Workflow Directory Service	10-5
Predefined Directory Service.....	10-6
Directory Service Views.....	10-7
WF_USERS View	10-8
WF_ROLES View.....	10-11
WF_USER_ROLES View	10-13
WF_USER_ROLE_ASSIGNMENTS_V View	10-16
Local Directory Service Tables	10-18
Ad Hoc Users and Roles.....	10-20
Validating a Directory Service Data Model.....	10-21
Setting Workflow Preferences.....	10-22
Loading Roles.....	10-23
Summary.....	10-24
Defining Function Activities	11-1
Defining Function Activities	11-3
Objectives	11-4
Function Activities	11-5
Defining a Function Activity	11-6
External Function Activities	11-8
Assigning a Cost to a Function Activity	11-10
PL/SQL Procedures for Function Activities	11-11
Standard API for PL/SQL Procedures Called by Function Activities	11-12
Standard API Parameters	11-14
Function Activity Execution Modes	11-15
Standard API Resultout Parameter	11-17
Exception Handling	11-18
Exception Handling Example	11-19
Defining Activity Details	11-20
Error Handling	11-22
Looping	11-23
Defining an Activity Attribute	11-24

Setting Activity Attribute Values	11-25
Summary.....	11-26
Post-Notification Functions.....	12-1
Post-Notification Functions	12-3
Objectives	12-4
PL/SQL Procedures for Notification Activities	12-5
Standard API for PL/SQL Procedures Called by Notification Activities	12-6
Post-Notification Function Execution Modes.....	12-9
Standard API Resultout Parameter for a Post-Notification Function	12-11
Post-Notification Function Context Information	12-12
Exception Handling	12-13
Summary.....	12-14
Workflow Engine.....	13-1
Workflow Engine	13-3
Objectives	13-4
Overview of the Workflow Engine.....	13-5
Initiating a Workflow Process	13-7
Workflow Engine Processing	13-9
Activity Statuses	13-12
Calling the Workflow Engine.....	13-13
Background Engines.....	13-14
Stuck Processes	13-15
Timed Out Activities	13-16
Deferred Processing.....	13-17
Oracle Workflow APIs	13-19
Workflow Engine APIs	13-21
Workflow Engine Bulk APIs.....	13-28
Summary.....	13-29
Business Events	14-1
Business Events	14-3
Objectives	14-4
Business Events	14-5
Event Properties.....	14-6
Generate Functions	14-8
License Status for Events	14-10
Defining an Event.....	14-11
Event Groups	14-12
Defining an Event Group.....	14-13
Maintaining Events.....	14-14
Raising Events	14-16
Event Message Structure	14-18
Raising an Event Manually.....	14-19
Predefined Events	14-21
Summary.....	14-22
Event Subscriptions	15-1
Event Subscriptions	15-3
Objectives	15-4
Event Subscriptions	15-5
Event Manager Subscription Processing	15-6
Local Event Subscription Processing	15-7
External Event Subscription Processing	15-9
Subscription Properties.....	15-10
Subscription Actions.....	15-14
Subscription Actions: Sending an Event to a Workflow Process	15-15
Subscription Actions: Sending an Event to an Agent.....	15-17

Subscription Actions: Sending a Notification.....	15-19
Subscription Actions: Sending and Receiving Oracle XML Gateway Messages.....	15-21
Subscription Actions: Invoking a Web Service	15-22
Subscription Actions: Running a Custom Rule Function	15-23
Subscription Properties.....	15-25
License Status for Subscriptions.....	15-28
Deferred Subscription Processing.....	15-29
PL/SQL and Java Subscription Processing.....	15-31
Defining a Subscription	15-33
Maintaining Subscriptions	15-35
Predefined Subscriptions	15-36
Summary.....	15-37
Systems and Agents	16-1
Systems and Agents.....	16-3
Objectives.....	16-4
Systems.....	16-5
System Properties	16-6
Local System	16-7
External Systems	16-8
Defining a System	16-9
Maintaining Systems	16-10
Agents.....	16-12
Standard Agents.....	16-14
Agent Properties	16-16
Custom Queue Handlers.....	16-21
Agents on External Systems	16-22
Defining an Agent	16-24
Agent Groups.....	16-26
Defining an Agent Group	16-27
Maintaining Agents	16-28
External System Registration	16-29
Summary.....	16-30
Defining Event Activities.....	17-1
Defining Event Activities	17-3
Objectives.....	17-4
Event Activities	17-5
Event-Based Workflow Processes.....	17-6
Event Activity Actions	17-7
Receive Event Activities	17-8
Receive Event Activities: Event Filter.....	17-9
Receive Event Activities: Sending an Event to One Process.....	17-10
Receive Event Activities: Sending an Event to Multiple Processes	17-12
Receive Event Activities: Receiving an Event	17-13
Raise Event Activities	17-15
Send Event Activities	17-16
Defining an Event Activity	17-17
Event Details	17-19
Defining Event Details: Receive	17-20
Defining Event Details: Raise	17-21
Defining Event Details: Send	17-23
Example: Order Processing	17-25
Standard Activities	17-26
Summary.....	17-27
Business Event System APIs.....	18-1
Business Event System APIs	18-3

Objectives	18-4
Business Event System datatypes	18-5
Event Message Structure	18-6
Agent Structure	18-10
Parameter List Structure	18-12
Parameter Structure	18-13
Raising Events Programmatically	18-15
Event Data Generate Functions	18-21
Standard API for PL/SQL Event Data Generate Functions	18-22
Standard API for Java Event Data Generate Functions	18-23
Queue Handlers	18-26
Standard APIs for PL/SQL Queue Handlers	18-27
Standard APIs for Java Queue Handlers	18-28
Subscription Rule Functions	18-30
Standard API for PL/SQL Subscription Rule Functions	18-31
Standard API for Java Subscription Rule Functions	18-33
Predefined Subscription Rule Functions	18-35
Event APIs	18-37
Event Function APIs	18-39
Adding a Correlation ID to an Event Message	18-40
Business Event System Cleanup API	18-42
Summary	18-43
Error Handling	19-1
Error Handling	19-3
Objectives	19-4
Error Handling for Workflow Processes	19-5
Default Error Process	19-7
Retry-only Process	19-9
Error Handling for Subscription Processing	19-11
Stop and Rollback Error Handling	19-13
Skip to Next Error Handling	19-15
Standard Error Agents	19-17
Error Handling Subscriptions	19-18
Warning Conditions in Subscription Processing	19-19
Unexpected Events	19-20
Default Event Error Process	19-21
Event Warnings	19-22
External Event Errors	19-23
Local Event Errors	19-25
Default Event Error Process (One Retry Option)	19-26
Summary	19-27
PL/SQL Documents	20-1
PL/SQL Documents	20-3
Objectives	20-4
PL/SQL Documents	20-5
Integrating PL/SQL Documents into Workflow Processes	20-6
Including PL/SQL Documents in Messages	20-8
Standard API for a PL/SQL Document	20-9
Standard API for a PL/SQL CLOB Document	20-11
Standard API for a PL/SQL BLOB Document	20-13
Summary	20-15
Forced Synchronous Processing	21-1
Forced Synchronous Processing	21-3
Objectives	21-4
Forced Synchronous Processes	21-5

Process Definition Restrictions.....	21-6
Summary.....	21-9
Selector/Callback Functions	22-1
Selector/Callback Functions	22-3
Objectives	22-4
Item Type Selector/Callback Functions.....	22-5
Defining a Selector/Callback Function for an Item Type.....	22-6
Standard API for a Selector/Callback Function.....	22-7
Standard API Parameters	22-9
Selector/Callback Function Commands.....	22-11
Summary.....	22-13
Master/Detail Coordination Activities	23-1
Master/Detail Coordination Activities.....	23-3
Objectives	23-4
Master/Detail Coordination Activities.....	23-5
Wait for Flow Activity	23-9
Continue Flow Activity	23-11
Master Process Example.....	23-12
Detail Process Example	23-13
Continue Flow Processing	23-14
Summary.....	23-15
Customizing Workflow Processes	24-1
Customizing Workflow Processes	24-3
Objectives	24-4
Customizing Workflow Processes	24-5
Access Protection	24-7
Access Levels	24-8
Setting the Access Level.....	24-10
Setting Protection and Customization Levels	24-11
Example of Access Protection	24-13
Unsupported Customizations.....	24-15
Preserving Customizations	24-16
Summary.....	24-17
Workflow Loaders	25-1
Workflow Loaders	25-3
Objectives	25-4
Workflow Definitions Loader.....	25-5
Workflow XML Loader.....	25-7
Summary.....	25-11
Specialized Workflow Monitoring	26-1
Specialized Workflow Monitoring	26-3
Objectives	26-4
Assigning Specialized Workflow Monitoring Privileges	26-5
Granting Restricted Access to Workflow Monitoring Data.....	26-7
Granting Restricted Access Based on Item Types	26-8
Granting Restricted Access Based on Functional Criteria.....	26-10
Granting Permissions for Administrative Actions	26-12
Summary.....	26-13
Setting Up Oracle Workflow	27-1
Setting Up Oracle Workflow	27-3
Objectives	27-4
Required Setup Steps	27-5
Step 1: Setting Global Workflow Preferences	27-6
Step 2: Setting Up an Oracle Workflow Directory Service	27-9

Step 3: Running Background Engines	27-13
Step 4: Configuring the Business Event System.....	27-16
Step 4: Event Message Communication	27-17
Step 4: Setting Up Database Links and Queues.....	27-18
Step 4: Checking Database Parameters	27-20
Step 4: Scheduling Agent Listeners.....	27-21
Step 4: Scheduling Propagation.....	27-24
Step 4: Synchronizing License Statuses	27-28
Step 4: Cleaning Up the WF_CONTROL Queue.....	27-29
Optional Setup Steps	27-30
Optional Step 1: Partitioning Workflow Tables	27-32
Optional Step 2: Setting Up Additional Languages.....	27-34
Optional Step 3: Implementing Notification Mailers	27-36
Optional Step 4: Customizing Message Templates	27-39
Optional Step 5: Adding Worklist Functions to User Responsibilities.....	27-42
Optional Step 6: Setting the Notification Reassign Mode	27-44
Optional Step 7: Enabling Bulk Notification Response.....	27-45
Optional Step 8: Setting Up Notification Handling Options	27-46
Optional Step 9: Setting Up for Electronic Signatures	27-48
Optional Step 10: Customizing the Workflow Web Page Logo	27-50
Optional Step 11: Adding Custom Icons	27-51
Summary.....	27-52
Managing Service Components.....	28-1
Managing Service Components	28-3
Objectives	28-4
Oracle Workflow Manager	28-5
Service Components	28-6
Service Component Containers	28-7
Service Component Types	28-8
Accessing Service Components in Oracle Workflow Manager.....	28-9
Managing Service Components	28-10
Service Component Container Logs.....	28-12
Service Component Startup Modes	28-13
Agent Listeners	28-14
Agent Listener Configuration Wizards	28-17
Notification Mailers.....	28-20
Outbound Notification Mailer Processing	28-21
Inbound Notification Mailer Processing.....	28-23
Notification Mailer Setup	28-25
Connecting to Mail Servers Through SSL.....	28-27
Notification Mailer Basic Configuration	28-28
Notification Mailer Advanced Configuration	28-30
Component Details for Notification Mailers	28-37
Notification Mailer Throughput	28-39
Handling Notification Mailer Errors.....	28-40
Testing Mailer URL Access	28-41
Summary.....	28-44
Managing System Status and Throughput.....	29-1
Managing System Status and Throughput	29-3
Objectives	29-4
Workflow System Status	29-5
Workflow Status in Oracle Applications Manager	29-7
Oracle Workflow Administration	29-9
Work Items	29-10
Oracle Workflow Administration	29-15
Purging Workflow Data.....	29-16

Completed Work Items.....	29-21
Workflow Purge APIs	29-24
Oracle Workflow Administration.....	29-25
Background Engines.....	29-26
Oracle Workflow Administration	29-29
Control Queue Cleanup	29-30
Oracle Workflow Administration	29-32
Queue Propagation	29-33
Oracle Workflow Administration	29-34
Agent Activity	29-35
Searching Messages on an Agent	29-36
Summary.....	29-37
Student Practices	30-1
Student Practices	30-3
Lesson 1 - Introduction to Oracle Workflow	30-4
Guided Demonstration - Loading and Running a Workflow Process.....	30-5
Lesson 2 - Oracle Workflow Components	30-7
Lesson 3 - Planning a Workflow Process	30-8
Practice - Planning a Workflow Process.....	30-9
Solution – Planning a Workflow Process	30-10
Lesson 4 - Diagramming a Workflow Process	30-11
Practice - Creating a Workflow Process	30-12
Solution – Creating a Workflow Process.....	30-13
Lesson 5 - Defining Item Type Attributes and Lookup Types	30-15
Practice - Defining Item Type Attributes.....	30-16
Solution – Defining Item Type Attributes	30-17
Lesson 6 - Defining Messages and Notification Activities.....	30-20
Practice - Defining Messages	30-21
Solution – Defining Messages	30-22
Practice - Defining Notification Activities	30-25
Solution – Defining Notification Activities	30-26
Lesson 7 - Testing and Monitoring Workflow Processes	30-28
Practice - Running a Workflow Process	30-29
Solution – Running a Workflow Process.....	30-30
Lesson 8 - Viewing and Responding to Notifications	30-31
Practice - Responding to Notifications	30-32
Solution – Responding to Notifications	30-33
Practice - Modifying A Workflow Process.....	30-34
Solution – Modifying a Workflow Process	30-36
Lesson 9 - Worklist Flexfields.....	30-39
Practice - Defining a Specialized Worklist View Using Worklist Flexfields	30-40
Solution – Defining a Specialized Worklist View Using Worklist Flexfields	30-42
Lesson 10 - Oracle Workflow Directory Service	30-46
Lesson 11 - Defining Function Activities.....	30-47
Practice - Defining a Function Activity	30-48
Solution – Defining a Function Activity	30-50
Practice - Branching on a Function Activity Result	30-53
Solution – Branching on a Function Activity Result	30-55
Practice - Using the Standard Assign Activity.....	30-58
Solution – Using the Standard Assign Activity	30-60
Lesson 12 - Post-Notification Functions	30-66
Practice - Defining a Post-Notification Function.....	30-67
Solution – Defining a Post-Notification Function	30-68
Lesson 13 - Workflow Engine.....	30-71
Practice - Implementing Timeout Processing	30-72
Solution – Implementing Timeout Processing	30-73

Practice - Implementing Deferred Processing	30-76
Solution – Implementing Deferred Processing.....	30-77
Lesson 14 - Business Events.....	30-79
Practice - Defining an Event.....	30-80
Solution – Defining an Event.....	30-81
Practice - Raising an Event.....	30-83
Solution – Raising an Event	30-84
Lesson 15 - Event Subscriptions.....	30-86
Practice - Defining a Subscription	30-87
Solution – Defining a Subscription.....	30-89
Lesson 16 - Systems and Agents	30-94
Lesson 17 - Defining Event Activities.....	30-95
Practice - Defining Event Activities	30-96
Solution – Defining Event Activities.....	30-98
Lesson 18 - Business Event System APIs	30-106
Lesson 19 - Error Handling	30-107
Guided Demonstration - Error Handling	30-108
Lesson 20 - PL/SQL Documents	30-109
Practice - Using a PL/SQL Document Attribute.....	30-110
Solution – Using a PL/SQL Document Attribute	30-112
Lesson 21 - Forced Synchronous Processing.....	30-115
Lesson 22 - Selector/Callback Functions.....	30-116
Practice - Defining a Selector Function	30-117
Solution – Defining a Selector Function.....	30-119
Lesson 23 - Master/Detail Coordination Activities	30-122
Lesson 24 - Customizing Workflow Processes	30-123
Lesson 25 - Workflow Loaders	30-124
Lesson 26 - Specialized Workflow Monitoring.....	30-125
Guided Demonstration - Setting Up Specialized Workflow Monitoring.....	30-126
Lesson 27 - Setting Up Oracle Workflow	30-129
Guided Demonstration - Scheduling Agent Listeners and Propagation	30-130
Lesson 28 - Managing Service Components	30-132
Guided Demonstration - Service Components	30-133
Lesson 29 - Managing System Status and Throughput.....	30-135
Guided Demonstration - System Status and Throughput.....	30-136
Sample Solutions.....	31-1
Sample Solutions	31-3
Overview	31-4
Vacation Proposal Process Sketch.....	31-5
wfvacxx.html	31-6
wfvacxxc.sql.....	31-7
wfvacxxs.sql	31-9
wfvacxxb.sql.....	31-14
wfvacxxd.sql.....	31-28
wfslctxx.sql.....	31-30

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Preface

Profile

Before You Begin This Course

- Thorough knowledge of Oracle Database and Oracle Application Server technology
- Thorough knowledge of Oracle E-Business Suite

Prerequisites

- *Oracle Database: Introduction to SQL*
- *Oracle Database: Program with PL/SQL*

How This Course Is Organized

R12.x Implement Oracle Workflow is an instructor-led course featuring lecture and hands-on exercises. Online demonstrations and written practice sessions reinforce the concepts and skills introduced.

Related Publications

Oracle Publications

Title	Part Number
<i>Oracle Workflow Administrator's Guide</i>	E12903
<i>Oracle Workflow Developer's Guide</i>	E12905
<i>Oracle Workflow User's Guide</i>	E12906
<i>Oracle Workflow API Reference</i>	E12904
<i>Oracle Workflow Client Installation Guide</i>	E12779

Additional Publications

- System release bulletins
- Installation and user's guides
- Read-me files
- International Oracle User's Group (IOUG) articles
- *Oracle Magazine*

Typographic Conventions

Typographic Conventions in Text

Convention	Element	Example
Bold italic	Glossary term (if there is a glossary)	The algorithm inserts the new key.
Caps and lowercase	Buttons, check boxes, triggers, windows	Click the Executable button. Select the Can't Delete Card check box. Assign a When-Validate-Item trigger to the ORD block. Open the Master Schedule window.
Courier new, case sensitive (default is lowercase)	Code output, directory names, filenames, passwords, pathnames, URLs, user input, usernames	Code output: debug.set ('I', 300); Directory: bin (DOS), \$FMHOME (UNIX) Filename: Locate the init.ora file. Password: User tiger as your password. Pathname: Open c:\my_docs\projects URL: Go to http://www.oracle.com User input: Enter 300 Username: Log on as scott
Initial cap	Graphics labels (unless the term is a proper noun)	Customer address (<i>but</i> Oracle Payables)
Italic	Emphasized words and phrases, titles of books and courses, variables	Do <i>not</i> save changes to the database. For further information, see <i>Oracle7 Server SQL Language Reference Manual</i> . Enter user_id@us.oracle.com, where <i>user_id</i> is the name of the user.
Quotation marks	Interface elements with long names that have only initial caps; lesson and chapter titles in cross-references	Select "Include a reusable module component" and click Finish. This subject is covered in Unit II, Lesson 3, "Working with Objects."
Uppercase	SQL column names, commands, functions, schemas, table names	Use the SELECT command to view information stored in the LAST_NAME column of the EMP table.
Arrow	Menu paths	Select File > Save.
Brackets	Key names	Press [Enter].
Commas	Key sequences	Press and release keys one at a time: [Alternate], [F], [D]
Plus signs	Key combinations	Press and hold these keys simultaneously: [Ctrl]+[Alt]+[Del]

Typographic Conventions in Code

Convention	Element	Example
Caps and lowercase	Oracle Forms triggers	When-Validate-Item
Lowercase	Column names, table names	SELECT last_name FROM s_emp;
	Passwords	DROP USER scott IDENTIFIED BY tiger;
	PL/SQL objects	OG_ACTIVATE_LAYER (OG_GET_LAYER ('prod_pie_layer'))
Lowercase italic	Syntax variables	CREATE ROLE <i>role</i>
Uppercase	SQL commands and functions	SELECT userid FROM emp;

Typographic Conventions in Oracle Application Navigation Paths

This course uses simplified navigation paths, such as the following example, to direct you through Oracle Applications.

(N) Invoice > Entry > Invoice Batches Summary (M) Query > Find (B) Approve

This simplified path translates to the following:

1. (N) From the Navigator window, select **Invoice** then **Entry** then **Invoice Batches Summary**.
2. (M) From the menu, select **Query** then **Find**.
3. (B) Click the **Approve** button.

Notations:

(N) = Navigator

(M) = Menu

(T) = Tab

(B) = Button

(I) = Icon

(H) = Hyperlink

(ST) = Sub Tab

Typographical Conventions in Oracle Application Help System Paths

This course uses a “navigation path” convention to represent actions you perform to find pertinent information in the Oracle Applications Help System.

The following help navigation path, for example—

(Help) General Ledger > Journals > Enter Journals

—represents the following sequence of actions:

1. In the navigation frame of the help system window, expand the General Ledger entry.
2. Under the General Ledger entry, expand Journals.
3. Under Journals, select Enter Journals.
4. Review the Enter Journals topic that appears in the document frame of the help system window.

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Business Events

Chapter 14

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Business Events

Business Events

ORACLE®

Objectives

Objectives

After completing this lesson, you should be able to do the following:

- **Define events.**
- **Define event groups.**
- **Raise events.**

ORACLE®

Business Events

Business Events

- A business event is an occurrence in an application or program that might be significant to other objects in a system or to external agents.
- Define your significant business events in the Event Manager.
- When an event occurs in an application on your local system:
 - The application must assign an event key to uniquely identify that particular instance of the event.
 - The event must be raised to the Event Manager.



ORACLE®

Business Events

For instance, the creation of a purchase order is an example of a business event in a purchasing application. For an event related to a business document, the event key could be the document identifier, such as the purchase order number.

Event Properties

Event Properties

- **Name**
- **Display Name**
- **Description**
- **Status**
- **Generate Function**
- **Java Generate Function**
- **Owner Name**
- **Owner Tag**
- **Customization Level**



ORACLE®

Event Properties

A business event definition in the Event Manager includes the following properties:

- Name: The internal name for an event must be unique and is case-sensitive. The following format is suggested:
<company>.<family>.<product>.<component>.<object>.<event>
- Display Name: A user-friendly name for the event.
- Description: Optional additional information about the event's purpose.
- Status: Enabled or Disabled; no subscriptions will be executed for a disabled event.
- Generate Function: A PL/SQL function that can generate the complete event data from the event name, event key, and an optional parameter list.
- Java Generate Function: A Java function that can generate the complete event data from the event name, event key, and an optional parameter list.
- Owner Name: Full name of the program or application that owns the event.
- Owner Tag: Program ID of the program or application that owns the event.

- Customization Level: A property that indicates what changes you can make to the event definition, used to protect seed data provided by Oracle and preserve your custom data during upgrades.
 - Core: No changes can be made to the event definition. This level is used only for events seeded by Oracle.
 - Limit: The event status can be updated to Enabled or Disabled, but no other changes can be made to the event definition. This level is used only for events seeded by Oracle.
 - User: Any property in the event definition can be updated. This level is automatically set for events that you define.

Note: Some examples of event names are:

- oracle.apps.wf.event.event.create
- oracle.apps.wf.event.event.delete
- oracle.apps.wf.event.subscription.create

These events are predefined events within the Business Event System. For more information, see: Predefined Workflow Events, *Oracle Workflow Developer's Guide*.

Generate Functions

Generate Functions

- Any detail information needed to describe what occurred in an event, in addition to the event name and event key, is called the event data.
- Event data is typically structured as an XML document.
- A generate function for an event is a function that can produce the complete event data from the event name, event key, and an optional parameter list.
- Generate functions must follow a standard API.



Generate Functions

- The event data is stored as a character large object (CLOB).
- The application where an event occurs can include the event data when raising the event to the Event Manager.
- If the application will not provide the event data, you should specify a generate function for the event.
- Define only one generate function for an event, either PL/SQL or Java.
- Define the generate function independently of any subscriptions to the event.
- The Event Manager checks each subscription before executing it to determine whether the subscription requires the event data. If the event data is required but is not already provided, the Event Manager calls the generate function for the event to produce the event data.
- If the event data is required but no generate function is defined for the event, Oracle Workflow creates a default set of event data using the event name and event key.

PL/SQL Generate Functions

The Business Event System performs PL/SQL subscription processing in the database tier for an event with a PL/SQL generate function. However, if subscription processing for the event moves to the middle tier due to Java subscriptions, the middle tier Business Event System will execute the PL/SQL generate function through Java Database Connectivity (JDBC), if required.

Java Generate Functions

You can assign an event a Java generate function instead of a PL/SQL generate function if you always want to generate the event data in the middle tier. In this case the Business Event System will always perform subscription processing for the event in the middle tier to enable execution of the Java generate function, even if the subscriptions to the event are all PL/SQL subscriptions. If the event is raised from PL/SQL code or received through a PL/SQL agent listener, the Event Manager places the event on the WF_JAVA_DEFERRED queue to move subscription processing to the middle tier.

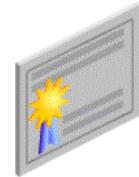
Generating Service Data Object Metadata

A service data object (SDO) represents a row of data in a service view object (SVO) in Oracle Application Framework. If you want to set the XML metadata for a service data object as the event data for an event, you can define the Java generate function for the event by using a special format to retrieve the SDO metadata. For more information, see: Managing Business Events, *Oracle Workflow Developer's Guide*.

License Status for Events

License Status for Events

- An Oracle E-Business Suite installation may include seeded events owned by products that you have not licensed.
- To avoid performing actions that are not relevant to your installation, subscriptions are executed only if both these conditions are met:
 - The subscription is owned by a licensed product.
 - The triggering event is owned by a licensed product.
- Products must be licensed with a status of Installed or Shared to be considered licensed.



License Status for Events

If a seeded event is owned by a product that you have not licensed, the Event Manager displays a notice that the event is not licensed when you view the event definition. Oracle Workflow will not execute any subscriptions to these events. Additionally, Oracle Workflow will not execute any subscriptions owned by products that you have not licensed, even if the triggering events for those subscriptions are licensed.

License status restrictions apply only to seeded events and subscriptions provided with Oracle E-Business Suite. Any custom events that you define with a customization level of User are always treated as being licensed.

Defining an Event

Defining an Event

- **Navigate to the Event Manager and select Events in the horizontal navigation.**
- **Click the Create Event button.**
- **On the Create Event page, enter the event properties and click Apply.**
- **New events that you define are automatically assigned a customization level of User.**



ORACLE

Defining an Event

Use a Web browser to navigate to the Event Manager, using a responsibility and navigation path specified by your system administrator. Some possible navigation paths in the seeded Workflow responsibilities are:

- Workflow Administrator Web Applications: Business Events
- Workflow Administrator Web (New): Business Events
- Workflow Administrator Event Manager: Business Events

You can also navigate to the Event Manager from other Oracle Workflow administrator Web pages by selecting the Business Events tab or clicking the Business Events link at the end of the page.

Note: You must have workflow administrator privileges to define events.

Event Groups

Event Groups

- Event groups let you associate any events that you want with each other and reference them as a group in event subscriptions.
- An event group is a type of event composed of a set of individual member events.
- Once you have defined an event group, you can register a subscription to the group rather than having to create separate subscriptions for each individual event within it.
- The subscription will be executed whenever any one of the group's member events occurs.



ORACLE®

Event Groups

- Event groups have the same properties as individual events, except that an event group does not have a generate function associated with it. Generate functions can be specified for member events in their individual definitions if necessary.
- The internal names of event groups should follow the same format as the names of individual events.
- Event groups cannot be used to raise events. You must raise each event individually.
- An event group can contain only individual events as its members. It cannot contain another group.

Defining an Event Group

Defining an Event Group

- **Navigate to the Event Manager and select Events in the horizontal navigation.**
- **Click the Create Event Group button.**
- **On the Create Group page, enter the event group properties and click Apply.**
- **New event groups that you define are automatically assigned a customization level of User.**
- **You can now add member events to the group.**



ORACLE®

Defining an Event Group

After you save the event group definition, the Create Group page displays the list of member events for that group.

- To add a member event to the group, click the Add Events to Group button.
- In the Add Events to Group page, enter search criteria to locate the events that you want to add. The main search option is the internal name of the event. You can enter a partial value to search for events whose internal names contain that value. You can also search by display name, status, owner name, and owner tag.
- Click the Go button to perform your search.
- Select the event or events that you want to add to your event group, and click the Add to Group button.
- After you finish adding events to the event group, click the Apply button to save the event group members.
- To delete a member event from the group, select the event on the Create Group page and click the Delete button.

Maintaining Events

Maintaining Events

Use the Events page to locate a specific event or event group definition and to maintain events and event groups.

- **Search for the events that you want to display.**
- **To view the subscriptions to an event, click the icon in the Subscription column for that event.**
- **To update an event, click the icon in the Update column for that event.**
- **To delete an event, select the event and click the Delete button.**



ORACLE

Maintaining Events

Searching for Events

In the Search region of the Events page, enter search criteria to locate the events that you want to display. The main search option is the internal name of the event. You can enter a partial value to search for events whose internal names contain that value. You can also search by the display name, status, owner name, and owner tag. Click the Go button to perform your search.

Subscriptions to an Event

For events that do not have any subscriptions, a blank subscription icon appears in the Subscription column. For events that do have subscriptions to them, a full subscription icon appears.

The Subscriptions: Event page displays the list of subscriptions to the event. If you have workflow administrator privileges, you can define a new subscription to the event by clicking the Create Subscription button. The Create Event Subscription page appears with the event name automatically entered in the Event Filter field. Also, if you have workflow administrator privileges, you can update an existing subscription by clicking the icon in the Update column for that subscription.

Updating Events

The customization level for each event or event group determines what changes you can make to its definition.

- Core: You cannot make any changes to the event definition.
- Limit: You can update the event status only.
- User: You can update any property in the event definition (except the customization level).

Deleting Events

You can only delete events that do not have any subscriptions referencing them and that are not members of any event group. Additionally, you cannot delete any event seeded by Oracle E-Business Suite that has a customization level of Core or Limit.

Raising Events

Raising Events

Events can be raised by the following methods:

- **From the application where the event occurs, using an API call**
- **From a workflow process, using a Raise event activity**
- **From the Test Business Event page, by manual submission**



Raising Events

To raise an event, you must provide the event name and an event key to identify the instance of the event. You can optionally provide a CLOB containing the event data and a list of additional header parameters for the event. If you use the WF_EVENT.Raise API or the Test Business Event page, you can optionally specify a send date for the event message.

If you are raising the event from your application code, you can use a PL/SQL API called WF_EVENT.Raise(), or a Java method called raise in the BusinessEvent class.

Raising Events

Raising Events

When an event is raised:

- **Information about the event is stored in a special datatype structure called WF_EVENT_T to form the event message.**
- **The Event Manager searches for and executes subscriptions to that event.**



Raising Events

The event message is the format in which Oracle Workflow communicates events among systems.

In Java, events are stored in a Java object called BusinessEvent, rather than in the WF_EVENT_T structure.

Event Message Structure

Event Message Structure

- Oracle Workflow uses the **WF_EVENT_T** datatype to store event messages.
- The **WF_EVENT_T** structure contains:
 - Header properties, including event name, event key, correlation ID, from agent, to agent, send date, receive date, priority, and error information
 - Event parameter list
 - Event data payload



ORACLE®

Raising an Event Manually

Raising an Event Manually

- Navigate to the Event Manager and select Events in the horizontal navigation.
- Search for the event that you want to raise.
- Click the test icon for that event.
- On the Test Business Event page, the event name is displayed by default.
- Enter an event key that uniquely identifies this instance of the event.
- Optionally, enter a send date, event parameters, and event data.
- Click either Raise in PLSQL or Raise in Java.



ORACLE

Raising an Event Manually

- Send Date: Optionally, specify the date when the event message is available for dequeuing. If you set the send date to a future date when you raise the event, the event message is placed on a deferred queue, and subscription processing does not begin until the specified date.
- Event Parameters: Optionally, enter any additional parameter name and value pairs to be stored in the parameter list within the event message. You can enter up to 100 parameters.
- Event Data: Optionally, enter an XML document to describe what occurred in the event.
 - To enter the event data manually, select Write XML in the Upload Option field and enter the XML document in the XML Content field. You can enter up to 4,000 characters.
 - To upload an XML file from your file system, select Upload XML in the Upload Option field. Enter the full path and file name of the XML file containing your event data in the XML File Name field, and click the Upload XML File button.

Note: You can also assign generate functions in the event definition to generate the event data. This method lets you provide the event data as a CLOB storing up to four gigabytes of data.

- Raise in PLSQL or Raise in Java: Choose the tier where you want to raise the event and begin event processing.
 - To raise the event in the database, select the Raise in PLSQL button.
 - To raise the event in the middle tier, select the Raise in Java button.
- Clear: After you raise an event, the page displays a confirmation message together with the values you submitted. To clear all fields except the Event Name field, select the Clear button. You can then optionally raise another event.

Predefined Events

Predefined Events

Oracle Workflow provides several predefined events for significant occurrences within the following components:

- **Business Event System**
- **Notification System**
- **Generic Service Component Framework**
- **Workflow Engine**
- **Directory Service**



ORACLE

Predefined Events

You can define subscriptions to the predefined Workflow events for replication, validation, or other purposes. For example, you can define a subscription to the Any event (oracle.apps.wf.event.any) if you want to perform some action whenever an event occurs.

Several predefined events are referenced by predefined subscriptions that are created automatically when you install Oracle Workflow. Generally, you should not disable these events, as many of them are used for Oracle Workflow internal processing. For example, the Unexpected event (oracle.apps.wf.event.unexpected) is used in error handling for subscription processing.

For the full list of events provided by Oracle Workflow, see: Predefined Workflow Events, *Oracle Workflow Developer's Guide*.

Summary

Summary

In this lesson, you should have learned how to:

- **Define events.**
- **Define event groups.**
- **Raise events.**

ORACLE®

Event Subscriptions

Chapter 15

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Event Subscriptions

Event Subscriptions

ORACLE®

Objectives

Objectives

After completing this lesson, you should be able to do the following:

- **Describe the types of processing a subscription can include.**
- **Define subscriptions.**

ORACLE®

Event Subscriptions

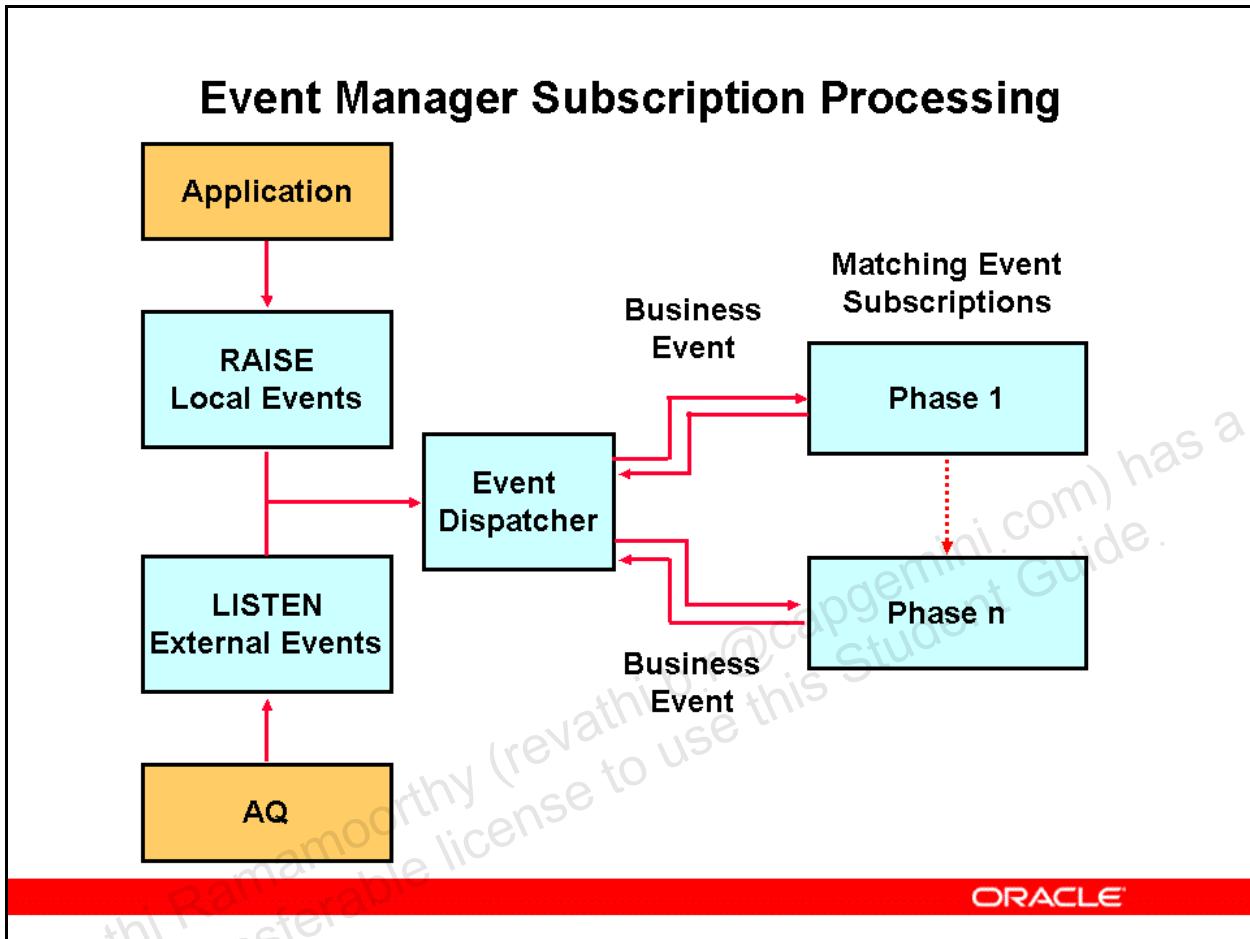
Event Subscriptions

- An event subscription is a registration that defines the processing to execute when an event occurs.
- Define subscriptions in the Event Manager.
- Whenever an event is raised locally or received from an external source, the Event Manager searches for and executes any active subscriptions by the local system to that event or to the Any event (oracle.apps.wf.event.any).



ORACLE®

Event Manager Subscription Processing



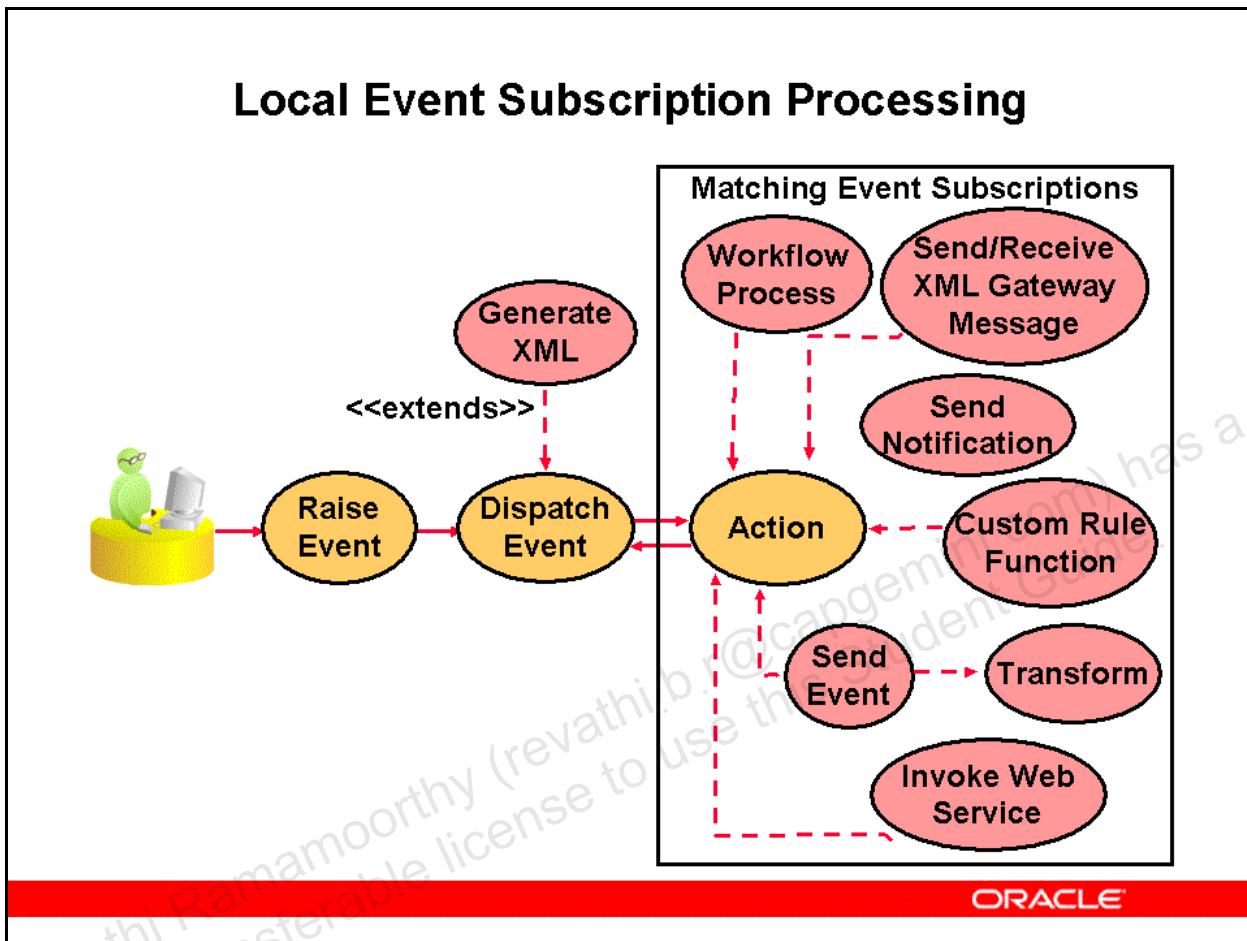
Event Manager Subscription Processing

When an event is raised by a local application, or when an event is received from an external source through Oracle Advanced Queuing (AQ), it is processed by a component of the Event Manager called the Event Dispatcher.

In both cases, the Event Dispatcher searches for and executes any subscriptions to that event. If there are multiple subscription to the same event, a subscription property called the phase value determines the order in which the subscriptions are executed. The Event Dispatcher executes only one subscription at a time, so one subscription must be completed before the next one is executed.

If no active subscriptions exist for the event that occurred, then you can handle the event by having Oracle Workflow execute any active subscriptions to the Unexpected event.

Local Event Subscription Processing



Local Event Subscription Processing

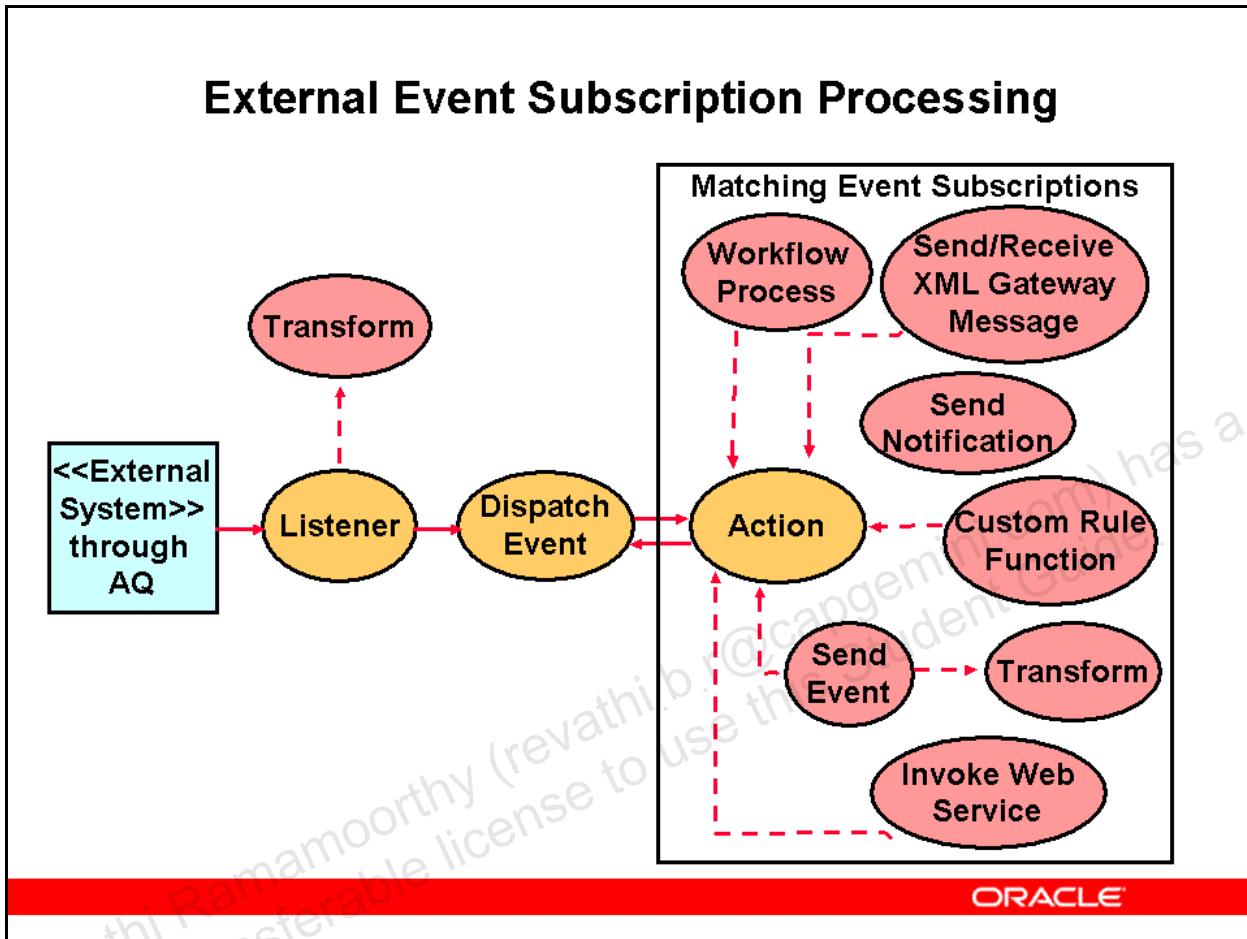
Oracle Workflow provides powerful capabilities for controlling the processing performed by your subscriptions when an event is raised.

- The Business Event System can handle event information as XML documents. The Event Dispatcher can run a generate function to generate the full XML data if the subscriptions to the event require it.
- Each event subscription has an action that specifies the type of processing it performs. The action is controlled by a function called a subscription rule function. This processing can include:
 - Sending the event message to a workflow process
 - Sending the event message to an agent on the local system or an external system
 - Sending a notification to a role
 - Receiving an Oracle XML Gateway message from your trading partner
 - Sending an Oracle XML Gateway message to your trading partner
 - Invoke a business process execution language (BPEL) process or other Web service
 - Running custom code on the event message

- The Business Event System supports performing transformations on event messages at enqueue time. For example, a message in the standard Workflow format called WF_EVENT_T can be transformed into the JMS Text message format when it is enqueued on an agent to be sent.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

External Event Subscription Processing



External Event Subscription Processing

Event subscriptions can also be triggered when an external source receives an event message through Oracle AQ.

- The Business Event System supports performing transformations on event messages when they are received. For example, if an agent receives a JMS Text message from an external source, the agent listener that handles the incoming message can transform the message from the JMS Text format into the standard Workflow format called WF_EVENT_T.
- External subscriptions can perform the same types of actions as local subscriptions.

Subscription Properties

Subscription Properties

- **Subscriber System:** The system where you want the subscription to execute
- **Source Type:** Local, External, or Error
- **Event Filter:** The triggering event
- **Source Agent:** The agent from which the triggering event is received



ORACLE

Subscription Properties

A subscription definition in the Event Manager includes the following properties:

Subscriber System

Each subscription defines an action on exactly one system, so you should define a separate subscription for each system involved in the processing that you want to perform. For example, if you want to propagate data from one system to another, you should define one subscription for the sending system, and another subscription for the receiving system.

Note: You can define and store subscriptions for multiple systems in the Event Manager on your local system, and use predefined events and subscriptions to replicate those subscriptions to other systems. However, only subscriptions with the local system as the subscriber are triggered when events are raised or received on the local system.

Source Type

Subscriptions can have the following source types:

- Local: The subscription applies only to events raised on the subscribing system.
- External: The subscription applies only to events received by an inbound agent on the subscribing system.

Note: All event messages received by an inbound agent on the subscribing system are considered to have an External source, whether the sending agent is located on a remote system or on the local system.

- Error: The subscription applies to only errored events dequeued from the WF_ERROR or WF_JAVA_ERROR agent's queue.

Note: Each subscription defines an action for exactly one source type. If you want to handle an event from both local and external sources, you should define two separate subscriptions, one with a source type of Local and the other with a source type of External.

Event Filter

You can choose either an individual event or an event group. If you choose an event group, the subscription will be triggered whenever any one of the group's member events occurs.

Source Agent

The source agent is an optional property. If you specify a source agent, then the subscription is executed only when the triggering event is received from that agent. In most cases, the Source Agent field is left blank.

Subscription Properties

Subscription Properties

- **Phase:** The order in which subscriptions to the same event are executed
- **Status:** Enabled or Disabled
- **Rule Data:** Key or Message



ORACLE

Subscription Properties

Phase

If you define multiple subscriptions to the same event, you can control the order in which the Event Manager executes those subscriptions by specifying a phase number for each subscription.

Subscriptions are executed in ascending phase order. For example, you can enter 10 for the subscription that you want to execute first when an event occurs, 20 for the subscription that you want to execute second, and so on. The Event Dispatcher executes only one subscription at a time, so one subscription must be completed before the one that comes next in the phase order is executed.

You can use phases to ensure that different types of actions are performed in the appropriate order, such as executing subscriptions that perform validation before subscriptions that perform other types of processing.

You can also use the phase number for a subscription to control whether the subscription is executed immediately or is deferred. The Event Manager treats subscriptions with a phase number of 100 or higher as deferred subscriptions.

Status

Oracle Workflow will not execute any disabled subscription.

Rule Data

Depending on the processing to be performed, a subscription may have the following rule data requirements:

- Key: Requires only the event key that identifies the instance of the event
- Message: Requires the complete set of event information contained in the event data

You can improve performance by specifying Key as the rule data for subscriptions that do not require the complete event data. For locally raised events, the Event Manager checks each subscription before executing it to determine whether the subscription requires the complete event data. If the event data is required but is not already provided, the Event Manager runs the Generate function for the event to produce the event data. However, if no subscriptions to the event require the event data, then the Event Manager will not run the Generate function, minimizing the resources required to execute the subscriptions.

For events received from an external source, any necessary event data must already be included in the event message by the source system.

Subscription Actions

Subscription Actions

- **Send the event message to a workflow process**
- **Send the event message to an agent**
- **Send a notification to a role**
- **Send an Oracle XML Gateway message to a trading partner**
- **Receive an Oracle XML Gateway message from a trading partner**
- **Invoke a Web service**
- **Run custom code on the event message**



ORACLE®

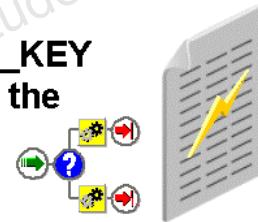
Subscription Actions

The processing performed by a subscription is called the subscription action. Code called a rule function controls the subscription action. You select an action type, which internally corresponds to a rule function.

Subscription Actions: Sending an Event to a Workflow Process

Subscription Actions: Sending an Event to a Workflow Process

- **To launch or continue a particular process:**
 - Specify the item type and process name of the process in the subscription.
 - The item key is determined either by the correlation ID in the event message, or by the event key if there is no correlation ID.
- **To continue one or more existing processes:**
 - Define the subscription to identify processes with a business key.
 - The event key must match a #BUSINESS_KEY attribute in the receive event activities in the waiting processes.



ORACLE

Subscription Actions: Sending an Event to a Workflow Process

By sending an event to a workflow process, you can model complex processing or routing logic beyond the standard action options. For example, based on the contents of the event message you can branch to different functions, initiate subprocesses, send notifications, or select recipient agents, or you can modify the event message itself.

- Any parameters included in the parameter list of the event message are set as item attribute values for the workflow process. If the corresponding item attributes do not already exist in the item type, Oracle Workflow automatically creates the item attributes when the event is sent to the process.
- Additionally, the subscription's globally unique identifier (GUID) is set as a dynamic item attribute of type text named SUB_GUID so that the workflow process can reference other information in the subscription definition.

Defining a Subscription to Send an Event to a Workflow Process

- To send an event to launch or continue a particular process, select the action type Launch Workflow and specify the item type, process name, and priority.
- To send the event to a particular workflow process only if the event message includes parameters whose names and values match all the parameters defined for the subscription,

select the action type Launch Workflow and the Launch when Parameters Match option. Then specify the item type, process name, priority, and parameters.

- To set additional parameters into the parameter list of the event message before sending the event to a particular process, select the action type Launch Workflow and the Add Subscription Parameters option. Then specify the item type, process name, priority, and parameters.
- To send the event to all existing processes that have eligible receive event activities waiting to receive it, marked by a business key attribute that matches the event key, select the action type Launch Workflow and the Launch when Business Key Matches option. Optionally, specify the priority.

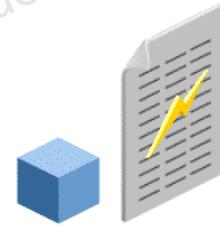
You can also include processing to send an event to a workflow process within a custom rule function.

Subscription Actions: Sending an Event to an Agent

Subscription Actions: Sending an Event to an Agent

To send an event to an agent, specify one of the following in the subscription:

- The Out Agent that should send the outbound message
- The To Agent that should receive the inbound message
- Both an Out Agent and a To Agent



ORACLE

Subscription Actions: Sending an Event to an Agent

Defining a Subscription to Send an Event to an Agent

- To send an event to an agent, select the action type Send to Agent and specify the agent details and priority.
- To send the event to an agent only if the event message includes parameters whose names and values match all of the parameters defined for the subscription, select the action type Send to Agent and the Launch when Parameters Match option. Then specify the agent details, priority, and parameters.
- To set additional parameters into the parameter list of the event message before sending the event to an agent, select the action type Send to Agent and the Add Subscription Parameters option. Then specify the agent details, priority, and parameters.

You can also include processing to send an event to an agent within a custom rule function.

Out Agent and To Agent

- If you specify both a To Agent and an Out Agent, Oracle Workflow places the event message on the Out Agent's queue for propagation, addressed to the To Agent.

- If you specify a To Agent without an Out Agent, Oracle Workflow selects an outbound agent on the subscribing system whose queue type matches the queue type of the To Agent. The event message is then placed on this outbound agent's queue for propagation, addressed to the To Agent.
- If you specify an Out Agent without a To Agent, Oracle Workflow places the event message on the Out Agent's queue without a specified recipient.
 - You can omit the To Agent if the Out Agent uses a multiconsumer queue with a subscriber list. (The standard Workflow queue handlers work only with multiconsumer queues.) In this case the queue's subscriber list determines which consumers can dequeue the message. If no subscriber list is defined for that queue, however, the event message is placed on the WF_ERROR queue for error handling.
 - You can also omit the To Agent if the Out Agent uses a single-consumer queue for which you have defined a custom queue handler. A single-consumer queue requires no specified consumer.

Future-Dated Messages

If you want an event message to become available to the recipient at a future date, rather than being available immediately as soon as it is propagated, you can set the SEND_DATE attribute within the event message to the date you want. You should set the send date during subscription processing before the event is sent, either in a prior subscription or earlier in a custom rule function before the send processing. The event message is propagated to the To Agent but does not become available for dequeuing until the specified date.

Subscription Actions: Sending a Notification

Subscription Actions: Sending a Notification

- **To send a notification:**
 - Specify a message template defined within an item type.
 - Specify the role that should receive the notification.
- **This action lets you send a notification without defining and running a complete workflow process.**



Subscription Actions: Sending a Notification

Defining a Subscription to Send a Notification

Select the action type Send Notification and specify the item type, message name, recipient role, and optional details in the subscription definition.

- You can optionally specify a custom callback function that you want the Notification System to use for communication of SEND and RESPOND source message attributes. Otherwise, Oracle Workflow uses a standard default callback function.
- You can also optionally enter context information that you want to pass to the callback function.
 - With the standard Oracle Workflow callback function, the Notification System requires a context to obtain values for item type attributes. If you do not specify a context, do not reference any item type attributes in the message attributes. Otherwise the message body will not display correctly. The context information consists of the item type, item key, and activity ID in the following format:
`<itemtype>:<itemkey>:<activityid>`
 - If you specify a custom callback function, then you can enter the context information that is appropriate for your function.

- You can also optionally enter a comment to include with the message and a priority for the message.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Subscription Actions: Sending and Receiving Oracle XML Gateway Messages

Subscription Actions: Sending and Receiving Oracle XML Gateway Messages

- If you use Oracle XML Gateway in Oracle E-Business Suite, you can:
 - Generate an Oracle XML Gateway message and send it to your trading partner
 - Receive an Oracle XML Gateway message from your trading partner
- The event must include the trading partner information in its parameter list.



ORACLE

Subscription Actions: Sending and Receiving Oracle XML Gateway Messages

Oracle Workflow and Oracle XML Gateway provide these actions to support business-to-business (B2B) integration scenarios.

- To send an Oracle XML Gateway message, select the action type Send Trading Partner Message and specify the priority.
- To receive an Oracle XML Gateway message, select the action type Receive Trading Partner Message and specify the priority.

Oracle XML Gateway then uses a standard workflow process defined in the XML Gateway Standard item type to send or receive the message. For more information, see “XML Gateway Standard Item Type,” *Oracle XML Gateway User’s Guide*.

Subscription Actions: Invoking a Web Service

Subscription Actions: Invoking a Web Service

If you use Oracle E-Business Suite Integrated SOA Gateway, you can:

- Invoke a business process execution language (BPEL) process
- Invoke other Web services



Subscription Actions: Invoking a Web Service

By defining a subscription that invokes a Web service, you can integrate Oracle E-Business Suite with service-oriented architecture (SOA) applications built using Oracle Fusion Middleware, such as Oracle BPEL Process Manager, or other standards-based technologies.

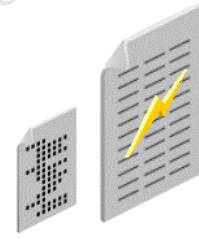
- To invoke a BPEL process, identify the triggering business event, and define the BPEL process to accept the event data payload. Then identify the Web Services Description Language (WSDL) description URL and other Web service properties for the BPEL process, and define a subscription to your event specifying those properties. When the event occurs, the Business Event System executes the subscription and invokes the BPEL process.
- To invoke any other Web service, identify the triggering business event, identify the WSDL description URL and other properties for the Web service, and define a subscription to your event specifying those properties. When the event occurs, the Business Event System executes the subscription and invokes the Web service.

For more details, see Invoking a Web Service, *Oracle Workflow Developer's Guide* and the *Oracle E-Business Suite Integrated SOA Gateway Developer's Guide*.

Subscription Actions: Running a Custom Rule Function

Subscription Actions: Running a Custom Rule Function

- You can extend your subscription processing by creating custom rule functions.
 - PL/SQL
 - Java
- Custom rule functions must be defined according to a standard API.
- You can specify additional parameters in a subscription to pass to a custom rule function.



ORACLE

Subscription Actions: Running a Custom Rule Function

The rule function specifies the processing performed for the subscription.

- Oracle Workflow provides standard rule functions that perform subscription processing for the standard subscription actions. You can also use these rule functions in a custom subscription if you choose.
- Oracle Workflow also provides some standard rule functions that you can use for testing and debugging or other purposes.
- To further extend your processing, you can write your own custom rule function.

Defining a Subscription to Run a Custom Rule Function

Select the action type Custom and specify the rule function, priority, and any parameters and workflow or agent details.

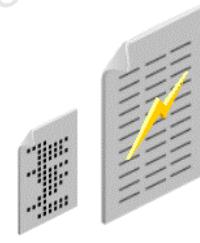
If you want to send the event message to a particular workflow process or to an agent as part of your custom processing, you can specify the workflow item type, process name, Out Agent, and To Agent in the subscription definition for your rule function to reference. Note, however, that you must explicitly include the send processing in your rule function.

Subscription Actions: Running a Custom Rule Function

Subscription Actions: Running a Custom Rule Function

Some possible uses for a custom rule function are:

- **Performing validation**
- **Processing inbound messages, acting as a Receive message handler for an application**
- **Making modifications to an outbound message, such as adding a correlation ID that associates this message with other messages**



ORACLE

Subscription Actions: Running a Custom Rule Function

If a rule function modifies the event message, any subsequent subscriptions executed on the event will access the changed message.

If the subscription processing that you want to perform for an event includes several successive steps, you may find it advantageous to define multiple subscriptions to the event with simple rule functions that you can reuse, rather than creating complex specialized rule functions that cannot be reused. You can enter phase values for the subscriptions to specify the order in which they should be executed.

Subscription Properties

Subscription Properties

On Error: The type of error handling for the subscription

- **Stop and Rollback**
- **Skip to Next**



ORACLE

Subscription Properties

You can specify the type of error handling to perform if the Event Manager encounters an error while processing the subscription.

- Stop and Rollback: The Event Manager halts all subscription processing for the event and rolls back any subscriptions already executed for the event. If the subscription trapped an error and returned a PL/SQL ERROR status code or a Java BusinessEventException, the Event Manager places the event message on the standard WF_ERROR or WF_JAVA_ERROR agent, as appropriate. If the subscription raised an unhandled exception, the Event Manager raises that exception to the calling application. If you later retry subscription processing for the event, the Event Manager re-executes all subscriptions to the event.
- Skip to Next: The Event Manager rolls back only this subscription. The Event Manager then places the event message on the standard WF_ERROR or WF_JAVA_ERROR agent, regardless of whether the subscription trapped an error or raised an unhandled exception. The exception is not raised to the calling application. Finally, the Event Manager continues processing the next subscription for the event according to the subscription phase order. If you later retry subscription processing for the event, the Event Manager re-executes only the errored subscription.

Note: Skip to Next error handling is not available for subscriptions with a source type of Error. If an additional error occurs during such a subscription, processing for that event will be halted until the error is addressed.

Subscription Properties

Subscription Properties

- **Owner Name**
- **Owner Tag**
- **Description**
- **Customization Level**



ORACLE®

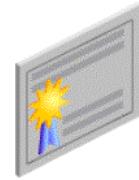
Subscription Properties

- Owner Name: Full name of the program or application that owns the event.
- Owner Tag: Program ID of the program or application that owns the event.
- Description: Optional additional information about the subscription's purpose.
- Customization Level: A property that indicates what changes you can make to the event definition, used to protect seed data provided by Oracle and preserve your custom data during upgrades.
 - Core: No changes can be made to the subscription definition. Only subscriptions seeded by Oracle use this level.
 - Limit: The subscription status can be updated to Enabled or Disabled, but no other changes can be made to the subscription definition. Only subscriptions seeded by Oracle use this level.
 - User: Any property in the subscription definition can be updated. This level is automatically set for subscriptions that you define.

License Status for Subscriptions

License Status for Subscriptions

- An Oracle E-Business Suite installation may include seeded subscriptions owned by products that you have not licensed.
- To avoid performing actions that are not relevant to your installation, subscriptions are executed only if both these conditions are met:
 - The subscription is owned by a licensed product.
 - The triggering event is owned by a licensed product.
- Products must be licensed with a status of Installed or Shared to be considered licensed.



ORACLE®

License Status for Subscriptions

If a seeded subscription is owned by a product that you have not licensed, the Event Manager displays a notice that the subscription is not licensed when you view the subscription definition. Oracle Workflow will not execute any of these subscriptions. Additionally, Oracle Workflow will not execute any subscriptions to events that you have not licensed, even if the subscriptions themselves are owned by a product that you have licensed.

License status restrictions apply only to seeded events and subscriptions provided by Oracle E-Business Suite. Any custom subscriptions that you define with a customization level of User are always treated as being licensed.

Deferred Subscription Processing

Deferred Subscription Processing

- **Deferred subscription processing lets you:**
 - Return control more quickly to the calling application
 - Execute costly subscription processing at a later time
- **To defer subscription processing for an event, you can:**
 - Define subscriptions to the event with phase numbers of 100 or higher.
 - Raise the event with a future send date.



ORACLE

Deferred Subscription Processing

To defer subscription processing, you can:

- Define subscriptions to the event with phase numbers of 100 or higher. Use this method when you want to defer processing of particular subscriptions for either local or external events.
- Raise the event with a future date in the SEND_DATE attribute within the WF_EVENT_T structure. Use this method when you want to defer all subscription processing for a locally raised event until a particular effective date.

When subscription processing for an event is deferred by either of these methods, the event message is placed on a standard deferred agent. If the subscription at which processing is deferred is a PL/SQL subscription, the event is placed on the standard WF_DEFERRED agent. If the subscription at which processing is deferred is a Java subscription, the event is placed on the standard WF_JAVA_DEFERRED agent. Oracle Workflow provides agent listeners to monitor the WF_DEFERRED and WF_JAVA_DEFERRED agents. Use Oracle Workflow Manager to ensure that these listeners are running.

The listener dequeues event messages from the WF_DEFERRED agent or WF_JAVA_DEFERRED agent in priority order. The event messages retain their original

source type, whether Local or External. The amount of time by which subscription processing for these events is deferred depends on the schedule defined for the listener, and, for future-dated events, on the specified effective date.

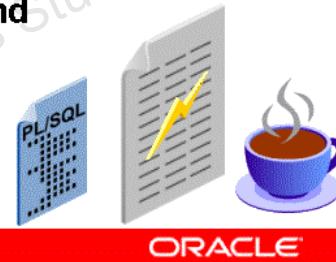
Note: If an event was deferred when the Event Manager encountered a Java subscription during subscription processing in the database, the Event Manager executes all remaining subscriptions to the event when the event is dequeued from the WF_JAVA_DEFERRED queue, including both Java and PL/SQL subscriptions, regardless of the subscription phase numbers. Subscriptions will not be deferred a second time.

PL/SQL and Java Subscription Processing

PL/SQL and Java Subscription Processing

Oracle Workflow performs subscription processing both in the database tier and in the middle tier.

- **Database tier:**
 - If an event is raised locally from PL/SQL code
 - If an event is received on an agent and processed by a PL/SQL agent listener
- **Middle tier:**
 - If an event is raised locally from Java code
 - If an event is received on an agent and processed by a Java agent listener



PL/SQL and Java Subscription Processing

If an event is raised locally from PL/SQL code, or received on an agent and processed by a PL/SQL agent listener, the Event Manager performs subscription processing in the database tier. It begins processing the subscriptions to the event in phase order and executes PL/SQL subscriptions synchronously, within the same database session, as long as the subscriptions are not deferred.

- If the Event Manager encounters a PL/SQL subscription that is deferred, it stops processing and places the event message on the WF_DEFERRED agent. Subscription processing for the event resumes when the Workflow Deferred Agent Listener or another agent listener runs on that agent.
- If the Event Manager encounters any Java subscription, or if the event has a Java generate function, the Event Manager stops processing and places the event message on the WF_JAVA_DEFERRED agent, regardless of the subscription phase number. Subscription processing for the event resumes when the Workflow Java Deferred Agent Listener or another agent listener runs on that agent.

If an event is raised locally from Java code, or received on an agent and processed by a Java agent listener, the Event Manager performs subscription processing in the middle tier. It begins

processing the subscriptions to the event in phase order, executing Java subscriptions in Java and PL/SQL subscriptions using Java Database Connectivity (JDBC), as long as the subscriptions are not deferred.

- If the Event Manager encounters a Java subscription that is deferred, it stops processing and places the event message on the WF_JAVA_DEFERRED agent. Subscription processing for the event resumes when the Workflow Java Deferred Agent Listener or another agent listener runs on that agent.
- If the Event Manager encounters a PL/SQL subscription that is deferred, it stops processing and places the event message on the WF_DEFERRED agent. Subscription processing for the event resumes when the Workflow Deferred Agent Listener or another agent listener runs on that agent.

Defining a Subscription

Defining a Subscription

- **Navigate to the Event Manager and select Subscriptions in the horizontal navigation.**
- **Click the Create Subscription button.**
- **In the Create Event Subscription page, enter the subscription properties and action type, and click Next.**
- **Enter the details for the action, depending on the action type, and click Apply.**
- **New subscriptions that you define are automatically assigned a customization level of User.**



ORACLE

Defining a Subscription

Use a Web browser to navigate to the Event Manager, using a responsibility and navigation path specified by your system administrator. Some possible navigation paths in the seeded Workflow responsibilities are:

- Workflow Administrator Web Applications: Business Events
- Workflow Administrator Web (New): Business Events
- Workflow Administrator Event Manager: Business Events

You can also navigate to the Event Manager from other Oracle Workflow administrator Web pages by choosing the Business Events tab or selecting the Business Events link at the end of the page.

Note: You must have workflow administrator privileges to define event subscriptions.

Use these guidelines when you enter subscription properties:

- **Workflow Process:** The list of values for this field includes only the runnable processes within the workflow item type you specify.
- **Out Agent:** The Out Agent must be located on the subscribing system. The list of values for the Out Agent field includes only agents with a direction of Out.

- To Agent: The list of values for the To Agent field includes only agents with a direction of In.
- Parameters: Enter the name and value for each parameter with no spaces.
- PL/SQL Rule Function: Enter the PL/SQL rule function in the following format:
<package_name>. <function_name>
- Java Rule Function: Enter the rule function in the following format:
<customPackage>. <customClass>

Note: Enter only one rule function for a subscription. If you enter a PL/SQL rule function, you must not enter a Java rule function, and vice versa.

Maintaining Subscriptions

Maintaining Subscriptions

Use the Event Subscriptions page to locate a specific subscription definition and to maintain subscriptions.

- **Search for the subscriptions that you want to display.**
- **To update a subscription, click the icon in the Update column for that subscription.**
- **To delete a subscription, select the subscription and click the Delete button.**



ORACLE

Maintaining Subscriptions

Searching for Subscriptions

In the Search region of the Event Subscriptions page, enter search criteria to locate the subscriptions that you want to display. The main search option is the subscribing system. You can also search by the source type, triggering event, or status. Click the Go button to perform your search.

Updating Subscriptions

The customization level for each subscription determines what changes you can make to its definition.

- Core: You cannot make any changes to the subscription definition.
- Limit: You can update the subscription status only.
- User: You can update any property in the subscription definition (except the customization level).

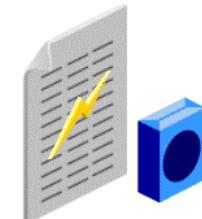
Deleting Subscriptions

You cannot delete any subscription seeded by Oracle that has a customization level of Core or Limit.

Predefined Subscriptions

Predefined Subscriptions

- Oracle Workflow provides predefined default subscriptions for several of its predefined events.
- The subscriber for all the predefined subscriptions is the local system.
- You can enable or disable subscriptions with a customization level of Limit to perform the event processing that you want.



ORACLE®

Predefined Subscriptions

Oracle Workflow includes predefined subscriptions to events including:

- Seed Event Group
- Ping Agent events
- System Signup event
- Any event
- Unexpected event
- User Entry Has Changed event
- Notification and notification mailer events
- Business Event System control events
- Generic Service Component Framework control events
- Directory service events

Many of these subscriptions are used for Oracle Workflow internal processing. Others are provided as optional features that you can enable or templates that you can copy to perform your own event processing.

Summary

Summary

In this lesson, you should have learned how to:

- **Describe the types of processing a subscription can include.**
- **Define subscriptions.**

ORACLE®

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Systems and Agents

Chapter 16

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Systems and Agents

Systems and Agents

ORACLE®

Objectives

Objectives

After completing this lesson, you should be able to do the following:

- **Define systems.**
- **Define communication agents on systems.**
- **Associate queues and queue handlers with agents.**

ORACLE®

Systems

Systems

- A system is a logically isolated software environment such as a host machine or database instance.
- Each system to which you will communicate events must be defined in the Event Manager.
 - Local system
 - External systems



ORACLE®

Systems

Systems that will communicate events with each other must store each other's system definitions in order to address event messages to each other.

System Properties

System Properties

- **Name**
- **Display Name**
- **Description**
- **Master**



ORACLE®

System Properties

A system definition in the Event Manager includes the following properties:

- Name: The internal name of the system. The internal name must be all-uppercase and should not include any single or double quotation marks (' or ") or spaces.
- Display Name: A user-friendly name for the system.
- Description: Optional additional information about the system.
- Master: Another system from which you want this system to receive Event Manager object definition updates.

Local System

Local System

- When you install Oracle Workflow in a database, that database is automatically defined as a system in the Event Manager.
- The system name is set to the database global name.
- This system is automatically set as the local system.



ORACLE®

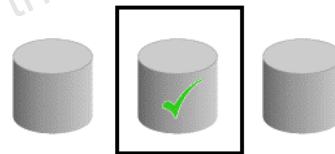
Local System

The local system is set on the Workflow Configuration page.

External Systems

External Systems

- **Workflow-enabled:** For communication with another Oracle E-Business Suite system, use the external system registration procedure to automatically copy the system definition for the other system into the Event Manager of your local system.
- **Non-Workflow:** For communication to a system that does not have Oracle Workflow installed, manually create a system definition for the other system in the Event Manager of your local system.



ORACLE®

External Systems

A Workflow-enabled system is another Oracle E-Business Suite system with Oracle Workflow configured.

A non-Workflow system can be either an Oracle system that does not have Oracle E-Business Suite with Oracle Workflow installed, or a non-Oracle system.

If you want to receive messages from a non-Workflow system on your local system, you must store the system and inbound agent information for the destination system in the source system according to that non-Workflow source system's requirements.

Defining a System

Defining a System

- You should manually create system definitions only for external systems that are not Workflow-enabled.
- Navigate to the Event Manager, select Systems in the horizontal navigation, and select System Details in the side navigation.
- Click the Create System button.
- On the Create System page, enter the system properties and click Apply.



ORACLE

Defining a System

Use a Web browser to navigate to the Event Manager, using a responsibility and navigation path specified by your system administrator. Some possible navigation paths in the seeded Workflow responsibilities are:

- Workflow Administrator Web (New): Business Events
- Workflow Administrator Web Applications: Business Events
- Workflow Administrator Event Manager: Business Events

You can also navigate to the Event Manager from other Oracle Workflow administrator Web pages by selecting the Business Events tab or clicking the Business Events link at the end of the page.

Note: You must have workflow administrator privileges to define systems.

An asterisk marks the local system when it is displayed on the System Details page.

Maintaining Systems

Maintaining Systems

Use the System Details page to locate a specific system definition and to maintain systems.

- **Search for the systems that you want to display.**
- **To view the agents belonging to a system, click the icon in the Agents column for that system.**
- **To view the subscriptions for a system, click the icon in the Subscription column for that system.**
- **To update a system, click the icon in the Update column for that system.**
- **To delete a system, select the system and click the Delete button.**



ORACLE

Maintaining Systems

Searching for Systems

On the Search region of the System Details page, enter search criteria to locate the systems that you want to display. The main search option is the internal name of the system. You can enter a partial value to search for systems whose internal names contain that value. You can also search by display name and master system. Click the Go button to perform your search.

Agents for a System

The View Agents: The System page displays a list of agents for the system. If you have workflow administrator privileges, you can update an agent by clicking the icon in the Update column for that agent.

Subscriptions for a System

For systems that do not have any subscriptions, a blank subscription icon appears in the Subscription column. For systems that do have subscriptions, a full subscription icon appears.

The Subscriptions: System page displays the list of subscriptions for the system. If you have workflow administrator privileges, you can define a new subscription for the system by clicking the Create Subscription button. The Create Event Subscription page appears with the system name automatically entered as the subscriber in the System field. Also, if you have

workflow administrator privileges, you can update an existing subscription by clicking the update icon in the Update column for that subscription.

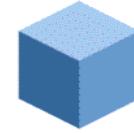
Deleting Systems

You can only delete systems that do not have any agents defined on them or any subscriptions referencing them. Also, you cannot delete the local system.

Agents

Agents

- An agent is a named point of communication within a system.
- A single system can have several different agents representing different communication alternatives, such as different protocols or propagation frequencies.
- Each agent that you will use to communicate events must be defined in the Event Manager.

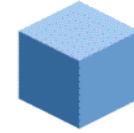


ORACLE®

Agents

Agents

- **Communication within and between systems is accomplished by sending a message from one agent to another.**
- **Each agent on a Workflow-enabled system is associated with an Oracle Advanced Queuing queue.**
- **The Business Event System interacts with the agent by enqueueing or dequeuing event messages on its queue.**



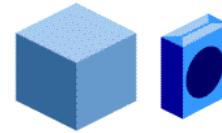
ORACLE®

Standard Agents

Standard Agents

You can use the following agents to send and receive event messages:

- **WF_IN: Default inbound agent**
- **WF_OUT: Default outbound agent**
- **WF_JMS_IN: Default inbound agent for JMS Text messages**
- **WF_JMS_OUT: Default outbound agent for JMS Text messages**



ORACLE

Standard Agents

When you install Oracle Workflow, several standard agents are automatically defined on the local system.

Oracle Workflow provides WF_IN, WF_OUT, WF_JMS_IN, and WF_JMS_OUT as default agents that you can use to receive and send events without needing to define any custom agents. You can optionally define additional inbound and outbound agents to expand your event processing. For example, you can define custom agents for:

- Propagation of event messages by different protocols or with different frequencies
- Increased scalability
- Additional levels of service - for instance, if you have five outbound agents, and one agent stops functioning, you can still send messages from the other four agents

Note: Java Message Service (JMS) is a messaging standard defined by Sun Microsystems, Oracle, IBM, and other vendors. Oracle Java Message Service (OJMS) provides a Java API for Oracle Advanced Queuing based on the JMS standard. The abstract datatype used to store a JMS Text message in an Oracle Advanced Queuing queue is called

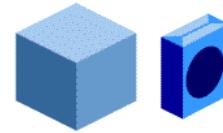
SYS.AQ\$_JMS_TEXT_MESSAGE. If you use JMS, you can use the standard WF_JMS_IN and WF_JMS_OUT agents to communicate Oracle Workflow event messages in this format.

Standard Agents

Standard Agents

The following agents are used for Workflow internal processing:

- **WF_CONTROL**: Workflow internal agent
- **WF_DEFERRED** and **WF_JAVA_DEFERRED**: Standard agents for deferred subscription processing
- **WF_ERROR** and **WF_JAVA_ERROR**: Standard agents for error handling
- **WF_NOTIFICATION_IN**: Standard inbound agent for e-mail notification responses
- **WF_NOTIFICATION_OUT**: Standard outbound agent for e-mail notifications



ORACLE

Standard Agents

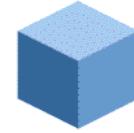
Oracle Workflow also includes the following standard agents for Oracle XML Gateway features:

- **WF_WS_JMS_IN**: Default inbound agent for Oracle XML Gateway Web service messages
- **WF_WS_JMS_OUT**: Default outbound agent for Oracle XML Gateway Web service messages

Agent Properties

Agent Properties

- **Name**
- **Display Name**
- **Description**
- **System**
- **Status**



ORACLE®

Agent Properties

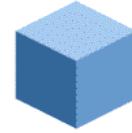
An agent definition in the Event Manager includes the following properties:

- Name: The internal name for an agent must be unique within the agent's system. The name must be all uppercase and should not include any single or double quotation marks (' or ") or spaces.
- Display Name: A user-friendly name for the agent.
- Description: Optional additional information about the agent's purpose.
- System: The system to which the agent belongs.
- Status: Enabled or Disabled. A disabled agent cannot be used in active subscriptions.

Agent Properties

Agent Properties

- **Direction:** Either inbound or outbound communication on the agent's system
- **Protocol:** Communication protocol that specifies how messages are encoded and transmitted
- **Address:** Address at which systems can communicate with an inbound agent



ORACLE

Agent Properties

Direction

An agent can support only one direction of communication on its system:

- In: The agent receives messages in a specific protocol and presents them to the system in a standard format.
- Out: The agent accepts messages from the system in a standard format and sends them using the specified protocol.

Protocol

For a message to be successfully communicated, the sending and receiving agents must use the same protocol.

You can use Oracle Advanced Queuing to perform the propagation of messages by the SQLNET protocol which it supports, or you can use the Oracle Advanced Queuing Internet access functionality to perform advanced queuing operations over the Internet using transport protocols such as HTTP and HTTPS. You can also use the Messaging Gateway and Internet access features of Oracle Advanced Queuing for integration with third-party messaging solutions, or you can implement an external service to propagate messages by a different protocol.

Address

The address format for an inbound agent depends on the agent's protocol. For agents that use the SQLNET protocol, the address must be in the following format:

<schema>.<queue>@<database link>

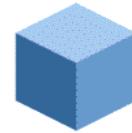
In this format, *<schema>* represents the schema that owns the queue, *<queue>* represents the name of the queue associated with the agent, and *<database link>* represents the name of the database link to the instance where the queue is located.

Note: You must enter the database link name exactly as the name was specified when the database link was created. The names of the database links that you want to use for the Business Event System should be fully qualified with the domain names.

Agent Properties

Agent Properties

- **Queue:** Oracle Advanced Queuing queue used by the Business Event System to interact with the agent
- **Queue handler:** PL/SQL or Java package that translates between the standard Workflow event message format and the format required by the agent's queue



ORACLE

Agent Properties

Queue

Each agent on a Workflow-enabled system should be associated with an Oracle Advanced Queuing queue.

- Outbound agent: To send messages, the system enqueues the messages on the queue and sets the recipient addresses.
- Inbound agent: To receive messages, the system runs a queue listener on the queue.

Specify the queue using the following format:

`<schema>.<queue>`

`<schema>` represents the schema that owns the queue and `<queue>` represents the queue name.

Queue Handler

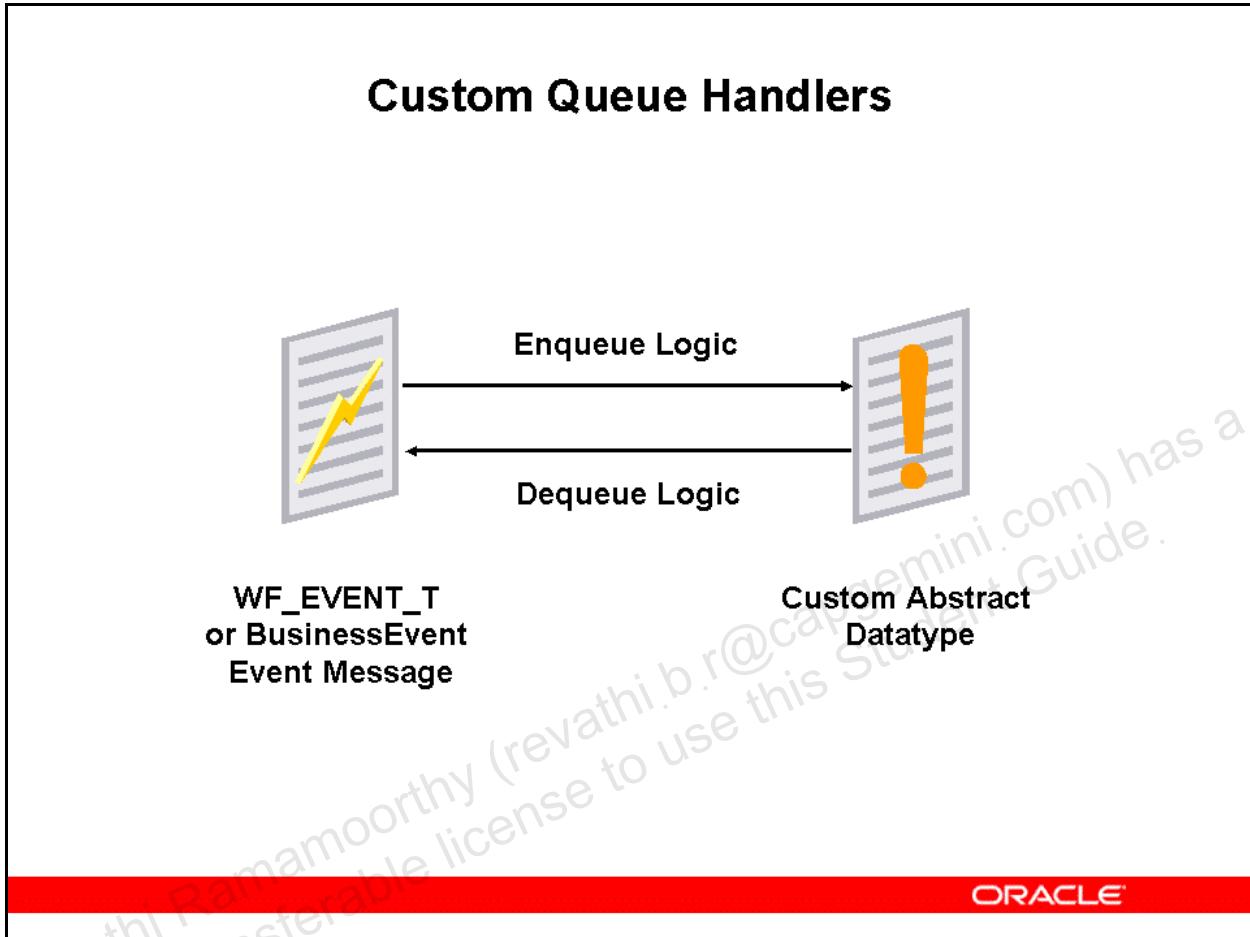
Event messages within the Business Event System are encoded in a standard format defined by the datatype WF_EVENT_T, or in Java, the BusinessEvent object. You must assign each agent a queue handler to enqueue and dequeue messages on the agent's queue, translating between the standard Workflow format and the format required by the queue.

- Oracle Workflow provides a standard queue handler named WF_EVENT_QH, which you can use for normal Business Event System processing with queues using SQLNET propagation and the payload type WF_EVENT_T.
- Oracle Workflow also provides a standard queue handler called WF_EVENT_OJMSTEXT_QH for queues that use the SYS.AQ\$_JMS_TEXT_MESSAGE datatype as their payload type. This queue handler enables communication of JMS Text messages through the Business Event System.
- You can also create your own custom PL/SQL or Java queue handler.

Define only one queue handler for an agent, either PL/SQL or Java.

- Java queue handler: Performs event message processing for the agent in the middle tier. You can run only Java agent listeners on this agent.
- PL/SQL queue handler: Performs event message processing for the agent in the database tier. You can run both PL/SQL and Java agent listeners on this agent. Java agent listeners will execute the PL/SQL queue handler through Java Database Connectivity (JDBC).

Custom Queue Handlers



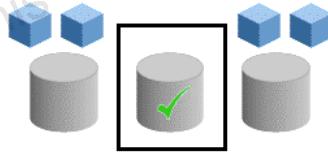
Custom Queue Handlers

A custom queue handler must translate between the standard WF_EVENT_T event message structure in PL/SQL, or the standard BusinessEvent object in Java, and your custom Abstract Datatype (ADT). Queue handler packages must include an enqueue API and a dequeue API, which must both follow a standard API format.

Agents on External Systems

Agents on External Systems

- **Workflow-enabled:** For communication with another Workflow-enabled system, use the external system registration procedure to automatically copy the inbound agent definitions for the other system into the Event Manager of your local system.
- **Non-Workflow:** For communication to a system that does not have Oracle Workflow installed, you should manually create agent definitions for the other system's inbound agents in the Event Manager of your local system.



ORACLE

Agents on External Systems

Systems that will communicate events with each other must store each other's inbound agent definitions in order to address event messages to each other. Definitions of outbound agents on other systems can optionally be stored as well.

If your local Workflow-enabled system will communicate with a non-Workflow system, the non-Workflow system must provide its own external propagation agents to handle Business Event System event messages.

- An inbound agent on a non-Workflow system must be able to dequeue event messages from a Business Event System outbound queue and process the contents of those messages. The inbound agent must have an address at which systems can communicate with the agent.
- An outbound agent on a non-Workflow system must be able to enqueue event messages in the appropriate format on a Business Event System inbound queue.

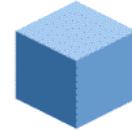
You must manually define the inbound agents for the non-Workflow system in the Event Manager of your local system. You can optionally define the non-Workflow system's outbound agents as well.

- Before defining agents for a non-Workflow system, you must define the system itself in your local Event Manager.
- You can then manually define an agent for the non-Workflow system, using the same pages as for any other agent. Follow these guidelines:
 - You must associate the agent with the non-Workflow system to which it belongs.
 - You must specify the protocol by which you will communicate with the agent.
 - For an inbound agent, you must also specify the address at which you will communicate with the agent.
 - You can leave the queue name and queue handler blank if the agent is not implemented as an Oracle Advanced Queuing queue.

Defining an Agent

Defining an Agent

- **Navigate to the Event Manager and select Agents in the horizontal navigation.**
- **Click the Create Agent button.**
- **On the Create Agent page, enter the agent properties and click Apply.**
- **Do not manually create agent definitions for inbound agents on external systems that have Oracle Workflow installed. Instead, use the external system registration procedure.**



ORACLE

Defining an Agent

Use a Web browser to navigate to the Event Manager, using a responsibility and navigation path specified by your system administrator. Some possible navigation paths in the seeded Workflow responsibilities are:

- Workflow Administrator Web (New): Business Events
- Workflow Administrator Web Applications: Business Events
- Workflow Administrator Event Manager: Business Events

You can also navigate to the Event Manager from other Oracle Workflow administrator Web pages by selecting the Business Events tab or clicking the Business Events link at the end of the page.

Note: You must have workflow administrator privileges to define agents.

Agent Properties

- Address: For agents that use the SQLNET protocol, the address must be in the following format to enable Oracle Advanced Queuing propagation: <schema>. <queue> @ <database link>

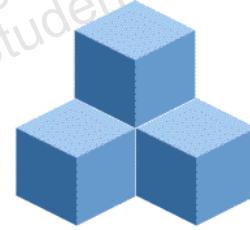
<schema> represents the schema that owns the queue, *<queue>* represents the queue name, and *<database link>* represents the database link to the instance where the queue is located.

- Queue handlers: You can assign the agent either a PL/SQL queue handler or a Java queue handler.
 - To assign the agent a PL/SQL queue handler, enter the PL/SQL package name in all uppercase in the Queue Handler field.
 - To assign the agent a Java queue handler, enter the Java package name in the Java Queue Handler field.
- Queue Name: Use the following format to specify the queue name: *<schema>. <queue>* *<schema>* represents the schema that owns the queue, and *<queue>* represents the queue name. You must enter the queue name in all uppercase.

Agent Groups

Agent Groups

- **Agent groups let you associate several inbound agents with each other and reference them as a group in event subscriptions and Send event activities.**
- **An agent group is a type of agent composed of a set of individual member agents.**
- **Once you have defined an agent group, you can send event messages to the group rather than having to send the messages separately to each individual agent within it.**



ORACLE

Agent Groups

Agent groups have the same properties as individual agents, except that an agent group does not have a protocol, address, queue, or queue handler associated with it. Those properties apply only for individual agents.

Agent groups can receive inbound messages only. All agent groups have a direction of In, and only individual agents with a direction of In can be members of an agent group. You cannot use agent groups to send outbound messages.

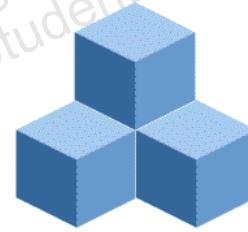
You must associate each agent group with a system to which it belongs. However, you can include agents on other systems within the group.

Ensure that you run an agent listener for each agent within the group to receive inbound messages. You cannot run an agent listener for an agent group; agent listeners can be run only for individual agents.

Defining an Agent Group

Defining an Agent Group

- **Navigate to the Event Manager and select Agents in the horizontal navigation.**
- **Click the Create Agent Group button.**
- **On the Create Agent Group page, enter the agent group properties and click Apply.**
- **You can now add member agents to the group.**



ORACLE®

Defining an Agent Group

After you save the agent group definition, the Create Agent Group page displays the list of member agents for that group.

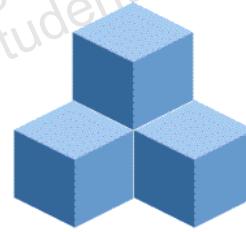
- To add a member agent to the group, click the Add Agents to Group button.
- On the Add Agents to Group page, enter search criteria to locate the agents you want to add. The main search option is the internal name of the agent. You can enter a partial value to search for agents whose internal names contain that value. You can also search by protocol, address, system, and status.
- Click the Go button to perform your search.
- Select the agent or agents that you want to add to your agent group, and click the Add Agents to Group button.
- After you finish adding agents to the agent group, click the Apply button to save the agent group members.
- To delete a member agent from the group, select the agent on the Create Agent Group page and click the Delete button.

Maintaining Agents

Maintaining Agents

Use the Agents page to locate a specific agent definition and to maintain agents.

- **Search for the agents that you want to display.**
- **To update an agent, click the icon in the Update column for that agent.**
- **To delete an agent, select the agent and click the Delete button.**



ORACLE®

Maintaining Agents

Searching for Agents

On the Search region of the Agents page, enter search criteria to locate the agents that you want to display. The main search option is the internal name of the agent. You can enter a partial value to search for agents whose internal names contain that value. You can also search by protocol, address, system, direction, individual or group type, and status. Click the Go button to perform your search.

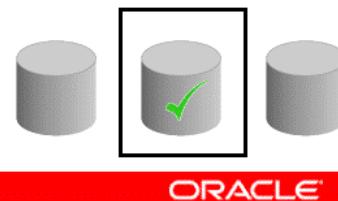
Deleting Agents

You can only delete agents that do not have any subscriptions referencing them and that do not belong to any agent groups.

External System Registration

External System Registration

- Before sending events from one system to another, you must register the destination system with the source system as a recipient of event messages.
- Registering a system means defining the destination system and its inbound agents in the Event Manager of the source system.
- When the destination system is registered, you can address event messages from the source system to the destination agents.



ORACLE

External System Registration

Registering systems is also referred to as signing up systems.

Usually when you integrate two systems, both systems should be registered with each other, so that each system can both send messages to and receive messages from the other system.

Oracle Workflow provides Web pages to help automate external system registration between two Oracle E-Business Suite systems that both have Oracle Workflow configured. For communication between a Workflow-enabled system and a non-Workflow system, you must perform manual steps to store the required destination system and agent information in the source system.

For details, see: Registering External Systems, *Oracle Workflow Developer's Guide*.

Summary

Summary

In this lesson, you should have learned how to:

- **Define systems.**
- **Define communication agents on systems.**
- **Associate queues and queue handlers with agents.**

ORACLE®

Defining Event Activities

Chapter 17

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Defining Event Activities

Defining Event Activities

ORACLE®

Objectives

Objectives

After completing this lesson, you should be able to do the following:

- **Define event activities.**
- **Define event details for event activity nodes.**
- **Send an event to a workflow process to start or continue the process.**
- **Use standard activities to manage information from event messages.**

ORACLE®

Event Activities

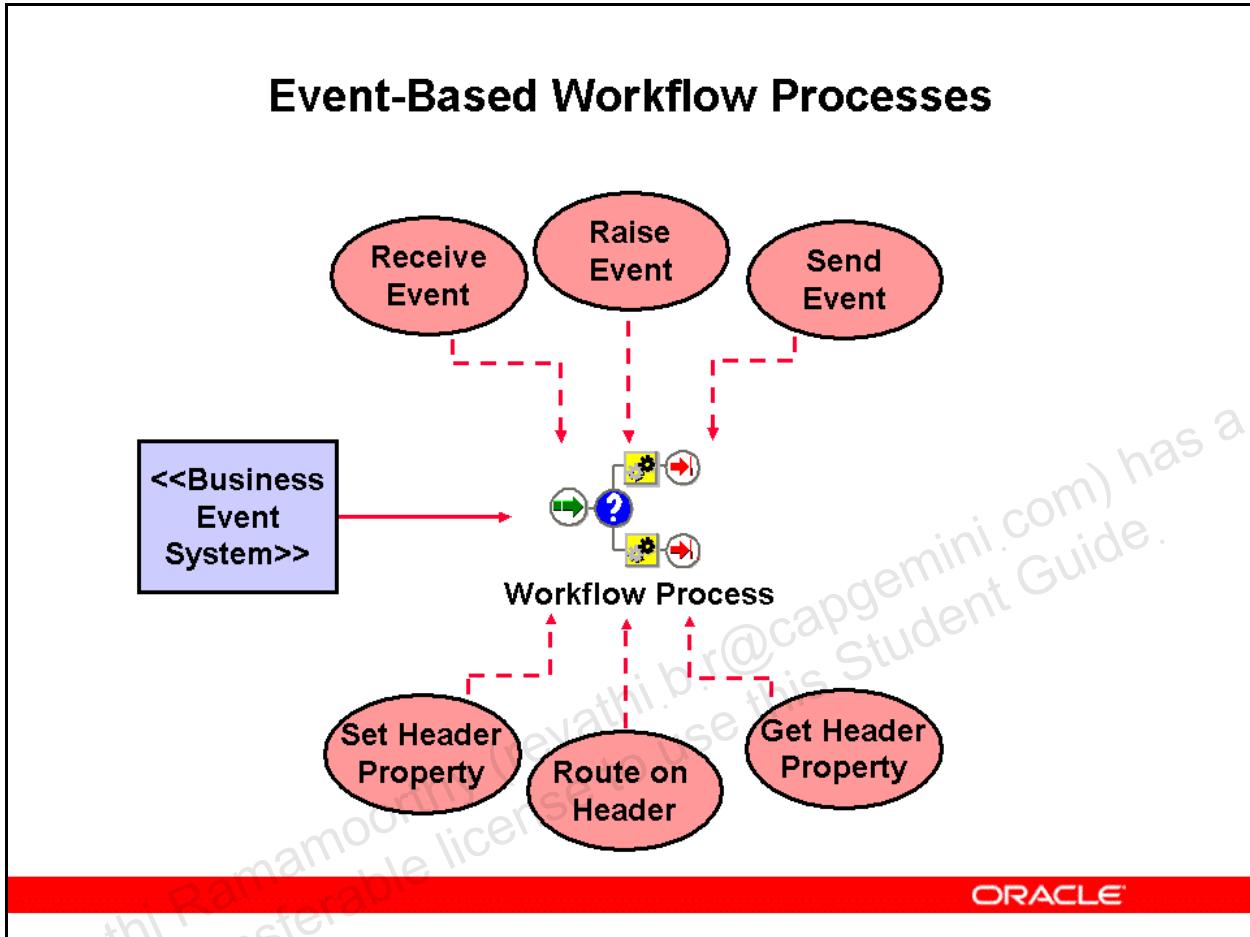
Event Activities

- An event activity represents a business event within a workflow process.
- You can include event activities in workflow processes to model complex processing or routing logic based on the content of an event.



ORACLE®

Event-Based Workflow Processes



Event-Based Workflow Processes

Event-based workflow processes control and route objects between applications according to business rules. These workflow processes support:

- Receiving business events to launch or continue processes
- Raising new business events
- Sending business event messages for intersystem communication
- Accessing and routing on header properties of event messages

By letting you model processes across different systems, event-based workflows enable business process-based integration.

Event Activity Actions

Event Activity Actions

An event activity can:

- Receive an event from the Event Manager
- Raise an event to the Event Manager
- Send an event message to an agent

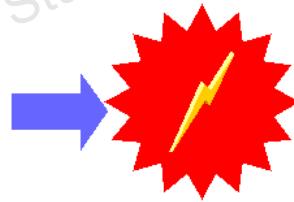


ORACLE®

Receive Event Activities

Receive Event Activities

- If you send an event to a workflow process from a subscription, the process must include a Receive event activity to accept the event.
- Depending on the subscription definition, the event can:
 - Launch one particular new process
 - Continue one particular existing process
 - Continue multiple existing processes



ORACLE®

Receive Event Activities

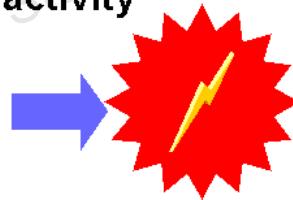
If a Receive event activity is marked as a Start activity, it is always enabled to receive events. Otherwise, the Receive event activity can only receive events after the process transitions to that activity.

Receive Event Activities: Event Filter

Receive Event Activities: Event Filter

The event filter for a Receive event activity determines which event the activity can receive.

- If you set the event filter to an individual event's internal name, the activity can receive only the specified event.
- If you set the event filter to an event group's internal name, the activity can receive any event that is a member of that group.
- If you leave the event filter blank, the activity can receive any event.

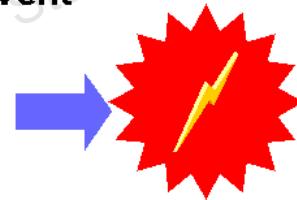


ORACLE

Receive Event Activities: Sending an Event to One Process

Receive Event Activities: Sending an Event to One Process

- A subscription can send an event to one particular process based on an item type, process name, and item key.
 - Launch a new process with the given item key
 - Continue an existing process identified by the given item key
- The correlation ID specified in the event message determines the item key.
- If no correlation ID is specified, the event key is used as the item key.



ORACLE

Receive Event Activities: Sending an Event to One Process

Defining Subscriptions

To define a subscription that sends the event to one particular process, select the action type Launch Workflow, and do not select the Launch when Business Key Matches option.

Receiving Events

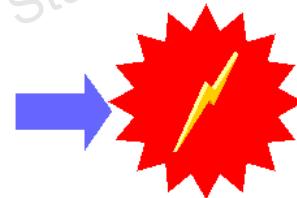
- To launch one particular new process when the event is sent, the item key must not have been previously used, and you must mark the Receive event activity as a Start activity.
- To continue one particular existing process when the event is sent, you must set the correlation ID or event key in the event message to match the item key of the running process, and the Receive event activity must have a status of NOTIFIED. A Receive event activity is set to NOTIFIED status in the following cases:
 - The process transitions to that activity.
 - A process has multiple Receive event activities marked as Start activities, and one of them receives an event to launch a new process instance. In this case, the Workflow Engine also searches for all other Receive event activities that are marked as Start activities and that do not have any incoming transitions, regardless of their event filter. For these activities, the Workflow Engine sets the activity status to NOTIFIED

so that they will be ready to receive an event if any more events are sent to this process. This feature lets you design a workflow process that requires multiple events to be received when you do not know in advance the order in which the events will arrive.

Receive Event Activities: Sending an Event to Multiple Processes

Receive Event Activities: Sending an Event to Multiple Processes

- A subscription can send an event to one or more existing processes based on a business key attribute.
- Each receive event activity that should receive the event must have an activity attribute named #BUSINESS_KEY.
- Oracle Workflow sends the event to each process with an activity whose #BUSINESS_KEY attribute value matches the event key.



ORACLE

Receive Event Activities: Sending an Event to Multiple Processes

Defining Subscriptions

To define a subscription that sends the event to multiple existing processes, select the action type Launch Workflow, and select the Launch when Business Key Matches option.

Receiving Events

- Define an activity attribute named #BUSINESS_KEY for each receive event activity that should receive the event.
- Set the default value of that activity attribute to an item type attribute.
- Include logic in the workflow process to set that item type attribute at run time to a business key value that matches the event key.
- The receive event activity must have a status of ‘NOTIFIED’ at the time the event arrives.

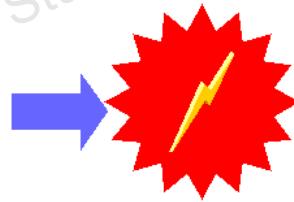
Note: You cannot launch new processes based on a business key attribute. You can only continue existing processes.

Receive Event Activities: Receiving an Event

Receive Event Activities: Receiving an Event

When a subscription sends an event to a workflow process, the Workflow Engine:

- **Sets any parameters in the event message parameter list as item type attributes for the process, creating new item type attributes if a corresponding attribute does not already exist for any parameter.**
- **Searches for eligible Receive event activities.**
- **Stores the event name, event key, and event message in the item type attributes specified for each eligible activity node.**



ORACLE

Receive Event Activities: Receiving an Event

For an activity to be eligible:

- When the event is sent to one specific process, the event must match the activity's event filter, and the activity must either be marked as a Start activity or have a status of NOTIFIED, meaning that the process has transitioned to the event.
- When the event is sent to one or more processes marked by a business key, an activity must have a status of NOTIFIED to be eligible and must have a #BUSINESS_KEY attribute that matches the event key.

If the Receive event activity has already received an event, then the On Revisit flag for the activity determines whether the Workflow Engine re-executes the activity.

If an event arrives at a Start activity to launch a new process instance, the Workflow Engine also searches for all other receive event activities that are marked as Start activities and that do not have any incoming transitions, regardless of their event filter. For these activities, the Workflow Engine sets the activity status to NOTIFIED so that they will be ready to receive an event if any more events are sent to this process. This feature lets you design a workflow process that requires multiple events to be received when you do not know in advance the order in which the events will arrive.

If the event was originally raised by a Raise event activity in another workflow process, the item type and item key for that process are included in the parameter list within the event message. In this case, the Workflow Engine automatically sets the specified process as the parent for the process that receives the event, overriding any existing parent setting.

After receiving an event, the Workflow Engine continues the thread of execution from each completed event activity node.

Raise Event Activities

Raise Event Activities

A Raise event activity:

- **Retrieves the event name, event key, and event data specified for the activity node and sets them into an event message structure.**
- **Retrieves the names and values of any activity attributes specified for the activity node and sets them as parameters in the parameter list within the event message.**
- **Also sets the item type and item key for the current workflow process as parameters in the parameter list within the event message.**
- **Raises the event to the Event Manager.**



ORACLE

Raise Event Activities

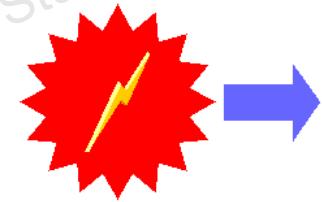
When the event is raised, it triggers any eligible subscriptions to that event by the local system. If another process later receives the event message, the Workflow Engine uses the item type and item key that were included as parameters in the parameter list to automatically set the process that raised the event as the parent for the process that receives the event.

Send Event Activities

Send Event Activities

A Send event activity:

- Retrieves the event name, event key, event message, Out Agent, and To Agent specified for the activity node.
- Sends the event message by placing it on the Out Agent's queue, addressed to the To Agent.



ORACLE

Send Event Activities

When the To Agent receives the event, it triggers any eligible subscriptions to that event on the To Agent's system.

A Send event activity sends the event directly from the Out Agent to the To Agent without raising the event to the Event Manager. No subscription processing is performed for the event on the sending system.

If no correlation ID is initially specified in the event message, the Send event activity automatically sets the correlation ID to the item key of the current workflow process.

Defining an Event Activity

Defining an Event Activity

Event activities must be associated with an item type and are created in the navigator tree beneath the Events branch of the item type.



ORACLE®

Defining an Event Activity

To define an event activity:

1. In the Oracle Workflow Builder, select the item type that you want in the navigator tree. Then select New Event from the Edit menu.
2. On the Activity property page, enter an internal name for the activity. The internal name must be all uppercase without any colons or leading or trailing spaces.
3. Enter a display name for the activity.
4. Enter a description of the activity.
5. Select an icon that identifies the activity.
6. Select the Event Action for the activity.
 - Receive
 - Raise
 - Send
7. If you are defining a Receive event activity, you can optionally enter an event filter to specify the event that the activity can receive.

- To allow the activity to accept only one specific event, enter the full internal event name.
 - To allow the activity to accept any event that is a member of a specific event group, enter the full internal event group name.
 - To allow the activity to accept any event, leave the Event Filter field blank.
8. Enter an optional cost for the activity. For Raise or Send event activities, you can use the cost to defer long running activities to a background engine.
 9. Click Apply to save your changes.
 10. Optionally, select the Details tab to display and modify additional activity details.
 11. Optionally, select the Access tab to set the access levels allowed to modify this event.

Event Details

Event Details

- When you add an event activity node to a workflow process, define event details for the node.
- The event details that are required depend on the event activity's action.



ORACLE®

Defining Event Details: Receive

Defining Event Details: Receive

For a Receive event activity, define these details:

- **Event Name**
- **Event Key**
- **Event Message**



ORACLE

Defining Event Details: Receive

To define event details for a Receive event activity node:

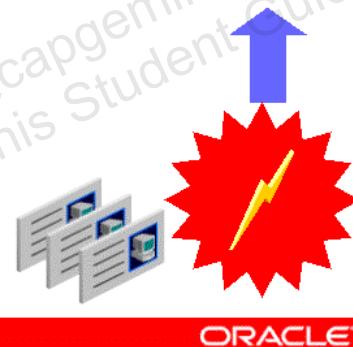
1. In the Oracle Workflow Builder, display the property pages of a Receive event activity node. Select the Event Details tab.
2. Enter the following event details:
 - Event Name: Optionally, select an item type attribute of the type text where you want to store the event name that the node receives.
 - Event Key: Optionally, select an item type attribute of the type text where you want to store the event key that the node receives.
 - Event Message: Optionally, select an item type attribute of the type event where you want to store the event message that the node receives.
3. Click Apply to save your changes.

Defining Event Details: Raise

Defining Event Details: Raise

For a Raise event activity, define these details:

- **Event Name**
- **Event Key**
- **Event Data**



Defining Event Details: Raise

To define event details for a Raise event activity node:

1. In the Oracle Workflow Builder, display the property pages of a Raise event activity node. Select the Event Details tab.
2. Enter the following event details:
 - Event Name: Enter the name of the event that the node raises. You can either specify a constant event name or select an item type attribute of the type text that dynamically determines the event name at run time.
 - Event Key: Select the item type attribute of the type text that contains the event key for the event that the node raises.
 - Event Data: Optionally, select an item type attribute that contains the event data for the event that the node raises. You can store event data in item type attributes of the type text, number, date, lookup, role, or attribute. You must not use an item type attribute of the type event, however, because the event data is only a part of the complete event message structure which is the format for the event attribute type.

Note:

- The event name and event key are required for a Raise event activity.
- The maximum length of the data that you can enter in a text attribute is 4,000 bytes. If the event data exceeds 4,000 bytes, you should assign a Generate function in the event definition to generate the event data, rather than providing the event data through a text attribute.

3. Click Apply to save your changes.

Defining Event Details: Send

Defining Event Details: Send

For a Send event activity, define these details:

- **Event Message**
- **Event Name**
- **Event Key**
- **Out Agent**
- **To Agent**



Defining Event Details: Send

To define event details for a Send event activity node:

1. In the Oracle Workflow Builder, display the property pages of a Send event activity node. Select the Event Details tab.
2. Enter the following event details:
 - Event Message: Select the item type attribute of the type event that contains the event message that the node sends.
 - Event Name: Optionally, enter the name of the event that the node sends. You can either specify a constant event name or select an item type attribute of the type text that dynamically determines the event name at run time. The event name that you enter here overrides the previous event name value in the event message.
 - Event Key: Optionally, select an item type attribute of type text that contains the event key of the event that the node sends. The event key that you enter here overrides the previous event key value in the event message.
 - Out Agent: Optionally, enter the outbound agent from which the node sends the event. Specify both the agent name and the system name for the agent using the following format: <agent_name>@<system_name>

You can either specify a constant Out Agent name or select an item type attribute of the type text that dynamically determines the Out Agent name at run time. The Out Agent that you enter here overrides the previous outbound agent value in the event message.

- To Agent: Optionally, enter the inbound agent to which the node sends the event. Specify both the agent name and the system name for the agent using the following format: <agent_name>@<system_name>

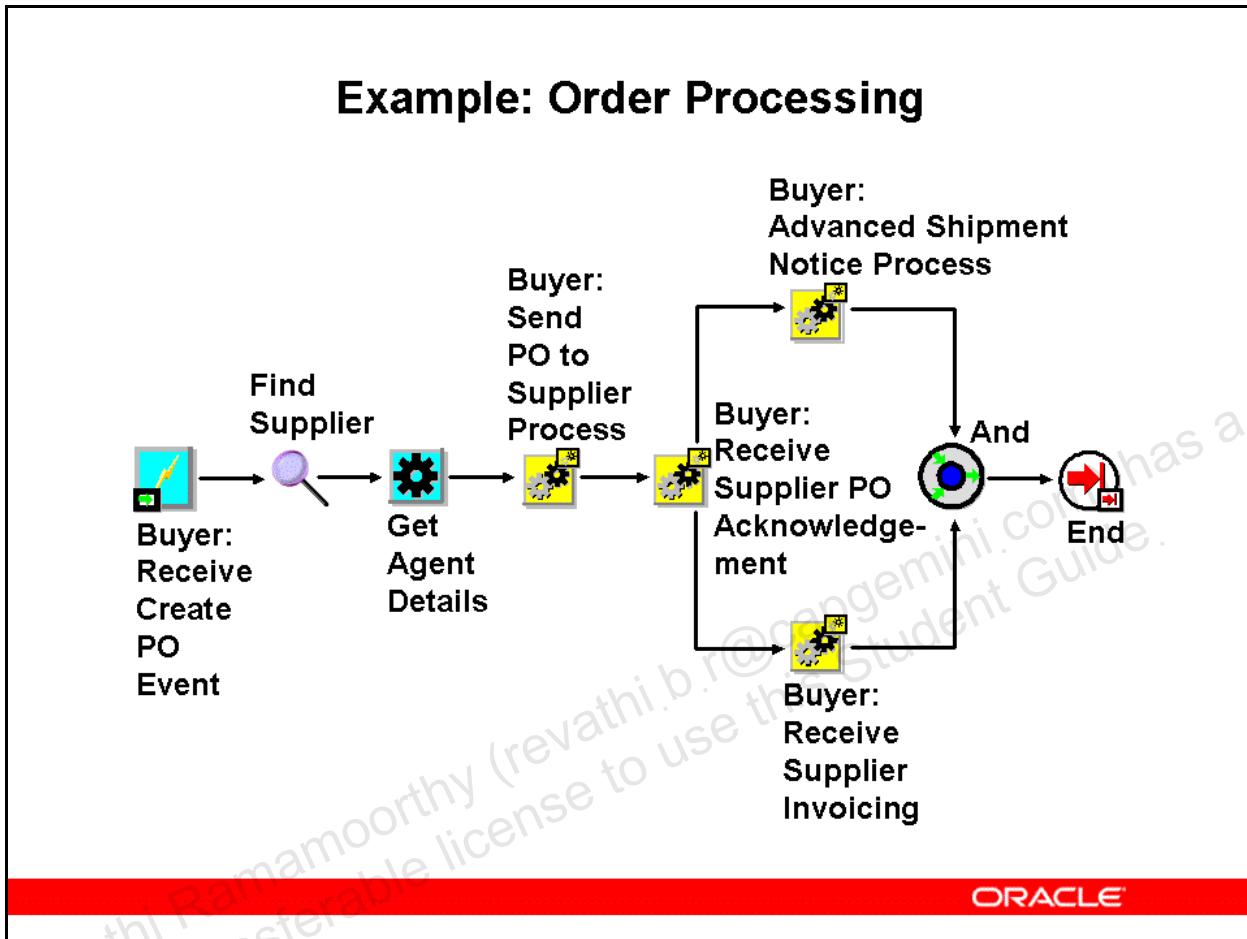
You can either specify a constant To Agent name or select an item type attribute of the type text that dynamically determines the To Agent name at run time. The To Agent that you enter here overrides the previous inbound agent value in the event message.

Note:

- The Event Message is required for a Send event activity. Additionally, you must either include a To Agent or a From Agent within the event message, or specify a To Agent or an Out Agent in the event details for this node. If you neither specify an inbound agent nor an outbound agent, the event cannot be sent.
- If no correlation ID is initially specified in the event message, Oracle Workflow automatically sets the correlation ID to the item key of the process.

3. Click Apply to save your changes.

Example: Order Processing



Example: Order Processing

This example shows a workflow process that includes business events. This process includes activities that receive a purchase order event to launch the workflow, send the purchase order event to a supplier, and wait to receive other events from the supplier in response to the order, such as an order acknowledgement, advanced shipment notice, and invoice.

Standard Activities

Standard Activities

- Oracle Workflow provides standard activities that let you access event message header properties within event messages.
- You can model routing or processing logic in your workflow process based on the properties of event messages.



Standard Activities

Use the following standard activities to manage the contents of event messages:

- Get Event Property: Retrieves a property of an event message and stores the property value in an item attribute.
- Set Event Property: Sets the value of a property in an event message.
- Compare Event Property: Compares a property of an event message with a test value.

Summary

Summary

In this lesson, you should have learned how to:

- **Define event activities.**
- **Define event details for event activity nodes.**
- **Send an event to a workflow process to start or continue the process.**
- **Use standard activities to manage information from event messages.**

ORACLE®

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Business Event System APIs

Chapter 18

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Business Event System APIs

Business Event System APIs

ORACLE®

Objectives

Objectives

After completing this lesson, you should be able to do the following:

- **Describe the abstract datatypes used by the Business Event System.**
- **Define event data generate functions.**
- **Define queue handler procedures.**
- **Define subscription rule functions.**
- **Apply Business Event System APIs.**

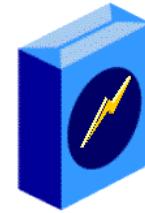
ORACLE

Business Event System datatypes

Business Event System datatypes

Oracle Workflow uses the following abstract datatypes to model the structure and behavior of Business Event System data.

- Event message structure: **WF_EVENT_T**
- Agent structure: **WF_AGENT_T**
- Parameter list structure: **WF_PARAMETER_LIST_T**
- Parameter structure: **WF_PARAMETER_T**



ORACLE

Event Message Structure

Event Message Structure

- Oracle Workflow uses the object type WF_EVENT_T to store event messages.
- WF_EVENT_T contains all the header properties of an event message as well as the event data payload.
- Internally, the Business Event System and the Workflow Engine communicate events in this format.
- The Java Business Event System communicates events as BusinessEvent objects.



ORACLE

Event Message Structure

The WF_EVENT_T type includes the following attributes:

Attribute Name: PRIORITY

Datatype: NUMBER

Description: The priority with which the message recipient should dequeue the message. A smaller number indicates a higher priority.

Attribute Name: SEND_DATE

Datatype: DATE

Description: The date and time when the message is available for dequeuing. The send date can be set to the system date to indicate that the message is immediately available for dequeuing, or to a future date to indicate future availability.

Attribute Name: RECEIVE_DATE

Datatype: DATE

Description: The date and time when the message is dequeued by an agent listener.

Attribute Name: CORRELATION_ID

Datatype: VARCHAR2(240)

Description: A correlation identifier that associates this message with other messages. This attribute is initially blank but can be set by a function. If a value is set for the correlation ID, then that value is used as the item key if the event is sent to a workflow process.

Attribute Name: PARAMETER_LIST

Datatype: WF_PARAMETER_LIST_T

Description: A list of additional parameter name and value pairs.

Attribute Name: EVENT_NAME

Datatype: VARCHAR2(240)

Description: The internal name of the event.

Attribute Name: EVENT_KEY

Datatype: VARCHAR2(240)

Description: The string that uniquely identifies the instance of the event.

Attribute Name: EVENT_DATA

Datatype: CLOB

Description: A set of additional details describing what occurred in the event. The event data can be structured as an XML document. A generate function can be defined for the event to produce the event data.

Attribute Name: FROM_AGENT

Datatype: WF_AGENT_T

Description: The agent from which the event is sent. For locally raised events, this attribute is initially null.

Attribute Name: TO_AGENT

Datatype: WF_AGENT_T

Description: The agent to which the event should be sent (the message recipient).

Attribute Name: ERROR_SUBSCRIPTION

Datatype: RAW(16)

Description: If an error occurs while processing this event, this is the subscription that was being executed when the error was encountered.

Attribute Name: ERROR_MESSAGE

Datatype: VARCHAR2(4000)

Description: An error message that the Event Manager generates if an error occurs while processing this event.

Attribute Name: ERROR_STACK

Datatype: VARCHAR2(4000)

Description: An error stack of arguments that the Event Manager generates if an error occurs while processing this event. The error stack provides context information to help you locate the source of an error.

The WF_EVENT_T object type also includes the following methods, which you can use to retrieve and set the values of its attributes.

Note: You must call the Initialize method before you can perform any further manipulation on a new WF_EVENT_T object.

- Initialize
- getPriority
- getSendDate
- getReceiveDate
- getCorrelationID
- getParameterList
- getEventName
- getEventKey
- getEventData
- getFromAgent
- getToAgent
- getErrorSubscription
- getErrorMessage
- getErrorStack
- setPriority
- setSendDate
- setReceiveDate
- setCorrelationID
- setParameterList
- setEventName
- setEventKey
- setEventData
- setFromAgent
- setToAgent
- setErrorSubscription
- setErrorMessage
- setErrorStack
- Content
- Address
- AddParameterToList
- GetValueForParameter

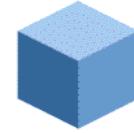
For more information, see: Business Event System APIs, *Oracle Workflow API Reference*.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Agent Structure

Agent Structure

- Oracle Workflow uses the object type WF_AGENT_T to store information about a named communication agent on a system.
- Event messages reference agent information in the format defined by the WF_AGENT_T datatype.



ORACLE

Agent Structure

The WF_AGENT_T type includes the following attributes:

Attribute Name: NAME

Datatype: VARCHAR2(30)

Description: The name of the agent.

Attribute Name: SYSTEM

Datatype: VARCHAR2(30)

Description: The system where the agent is located.

The WF_AGENT_T object type also includes the following methods, which you can use to retrieve and set the values of its attributes.

- getName
- getSystem
- setName
- setSystem

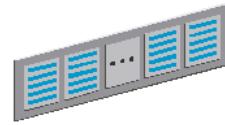
For more information, see: Business Event System APIs, *Oracle Workflow API Reference*.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Parameter List Structure

Parameter List Structure

- Oracle Workflow uses the named varying array **WF_PARAMETER_LIST_T** to store a list of parameters in a form that can be included in an event message.
- **WF_PARAMETER_LIST_T** allows custom values to be added to the **WF_EVENT_T** event message object.
- The **WF_PARAMETER_LIST_T** datatype can include up to 100 parameter name and value pairs.



ORACLE

Parameter List Structure

WF_PARAMETER_LIST_T has the following properties:

Maximum size: 100

Element datatype: **WF_PARAMETER_T**

When **WF_PARAMETER_LIST_T** is used within a **WF_EVENT_T** structure, you can use the following **WF_EVENT_T** methods to add and retrieve parameters in the list.

- AddParameterToList
- GetValueForParameter

When an event message that includes a parameter list is sent to a workflow process, the parameters in the list are created as item attributes for the workflow process.

For more information, see: Business Event System APIs, *Oracle Workflow API Reference*.

Parameter Structure

Parameter Structure

- Oracle Workflow uses the object type **WF_PARAMETER_T** to store a parameter name and value pair in a form that can be included in an event message parameter list.
- **WF_PARAMETER_T** allows custom values to be added to the **WF_EVENT_T** event message object.



ORACLE®

Parameter Structure

The **WF_PARAMETER_T** type includes the following attributes:

Attribute Name: NAME

Datatype: VARCHAR2(30)

Description: The parameter name.

Attribute Name: VALUE

Datatype: VARCHAR2(2000)

Description: The parameter value.

The **WF_PARAMETER_T** object type also includes the following methods, which you can use to retrieve and set the values of its attributes.

- getName
- getValue
- setName
- setValue

For more information, see: Business Event System APIs, *Oracle Workflow API Reference*.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Raising Events Programmatically

Raising Events Programmatically

- You can raise an event from your application by calling the WF_EVENT.Raise() or WF_EVENT.Raise3() APIs or a Java raise API.
- To raise an event, you must provide the event name and an event key to identify the instance of the event.
- You can also optionally provide the event data in a character large object (CLOB), an additional parameter list, and a send date.
- This information is stored in a WF_EVENT_T structure or BusinessEvent Java object to form the event message.



ORACLE

Raising Events Programmatically

WF_EVENT.Raise3() performs the same processing as the WF_EVENT.Raise() API, except that WF_EVENT.Raise3() passes the event parameter list back to the calling application after completing the event subscription processing.

The Java Business Event System provides the BusinessEvent class to raise events in Java and attach an event data payload. For more information, see: Managing Business Events, *Oracle Workflow Developer's Guide*.

Note: A CLOB can store up to 4 gigabytes of data.

PL/SQL Example

The following example code shows how to use the WF_EVENT.Raise API to raise an event from an application.

```
declare
  l_xmldocument varchar2(32000);
  l_eventdata clob;
  l_parameter_list wf_parameter_list_t := wf_parameter_list_t();
  l_message varchar2(10);
  l_parameter_t wf_parameter_t:= wf_parameter_t(null, null);
```

```

begin

/*
** If the complete event data is easily available,
** we can optionally test if any subscriptions to
** this event require it (rule data = Message).
*/

l_message := wf_event.test('<EVENT_NAME>');

/*
** If we do require the complete event data, and we
** have the data now, set it; else we can just rely
** on the Event Generate Function callback code.
** Then Raise the Event with the required
** parameters.
*/

```

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

```

l_parameter_t.SetName('PO_NUMBER');
l_parameter_t.SetValue('PO1234');
l_parameter_list.extend;
l_parameter_list(1) := l_parameter_t;

l_parameter_list.extend;
l_parameter_list(2) := wf_parameter_t('REQUESTOR', 'John Doe');

if l_message = 'MESSAGE' then
  if l_xmldocument is not null then
    dbms_lob.createtemporary(l_eventdata, FALSE, DBMS_LOB.CALL);

    dbms_lob.write(l_eventdata, length(l_xmldocument), 1, l_xmldocument);
    -- Raise the Event with the message
    wf_event.raise(
      p_event_name => '<EVENT_NAME>',
      p_event_key  => '<EVENT_KEY>',
      p_event_data => l_eventdata,
      p_parameters => l_parameter_list);
  else
    -- Raise the Event without the message
    wf_event.raise(
      p_event_name => '<EVENT_NAME>',
      p_event_key  => '<EVENT_KEY>',
      p_parameters => l_parameter_list);
  end if;

```

```

elsif
  l_message = 'KEY' then
    -- Raise the Event
    wf_event.raise(
      p_event_name => '<EVENT_NAME>',
      p_event_key  => '<EVENT_KEY>',
      p_parameters => l_parameter_list);
end if;

/*
** Up to your own custom code to commit the
** transaction
*/
commit;

/*
** Up to your own custom code to handle any major
** exceptions
*/
exception
when others then
null;
end;

```

Java Example

The following example code shows how to raise an event from a Java application.

```

import java.sql.Connection;
import oracle.apps.fnd.wf.bes.BusinessEvent;
import oracle.apps.fnd.wf.bes.BusinessEventException;
import oracle.apps.fnd.wf.common.WorkflowContext;
import oracle.apps.fnd.common.Log;
import oracle.apps.fnd.common.AppsContext;

public class HelloWorldPayloadRaise
{
  //Event Constants - Not Mandatory
  private static final String E_NAME =
    "oracle.apps.fnd.wf.devstd.event.1";
  private static final String E_KEY = "key0129";
  private static final String E_DATA = "This is the event data.";

  //Business Event member variable - Initialized in init() method
  private BusinessEvent mEvent;

```

```

//Connection object - needed in the raise() method
private Connection mConnection;

//Log class for logging
Log mlog;
private final String CLASS_PREFIX =
HelloWorldPayloadRaise.class.getName() + ".";

public void HelloWorldPayloadRaise() {}

// Method to get DB Connection
// Users are free to obtain DB Connection by any means.
// Sample shows how to obtain the Connection from dbc file
// using ApplicationContext
public void initConnection() {
final String METHOD_NAME = "initConnection()";

try {
String dbcfile = System.getProperty("dbcfile");
ApplicationContext ctx = new ApplicationContext(dbcfile);
mConnection = ctx.getJDBCConnection();
//Make sure AutoCommit is set to false
mConnection.setAutoCommit(false);

// use the logger from ApplicationContext for logging
mlog = ctx.getLog();

} catch (Exception e) {
e.printStackTrace();
System.exit(1);
}

mlog.write(CLASS_PREFIX + METHOD_NAME, "END", Log.PROCEDURE);
}

//Close method to release DB Connection
public void close() {
final String METHOD_NAME = "close()";
try {
mConnection.commit();
} catch (Exception e) {
mlog.write(CLASS_PREFIX + METHOD_NAME, "Exception when Committing",
Log.ERROR);
System.exit(1);
}
}
}

```

```

//Initialize BusinessEvent, Payload objects
public void init() {
    final String METHOD_NAME = "init()";
    mlog.write(CLASS_PREFIX + METHOD_NAME, "BEGIN", Log.PROCEDURE);

    //Create BusinessEvent Object
    mEvent = new BusinessEvent(E_NAME, E_KEY);
    try {
        // set the event data
        mEvent.setData(E_DATA);
        // set some string properties
        mEvent.setStringProperty("name100", "value100");
        mEvent.setStringProperty("name200", "value200");
        mEvent.setStringProperty("name300", "value300");

        try {
            //Create the Event Payload Object
            HelloWorldPayload eventPayload = new HelloWorldPayload();
            eventPayload.setPayloadValue("Uncle Sam");
            mlog.write(CLASS_PREFIX + METHOD_NAME, "Setting Object",
                Log.STATEMENT);

            //Set the Event payload with the BusinessEvent object
            mEvent.setObject(eventPayload);
        }
        catch (java.io.NotSerializableException serialExcep) {
            System.out.println("Setting Object Exception");
            mlog.write(CLASS_PREFIX + METHOD_NAME, "Object Not Serializable:" +
                serialExcep, Log.ERROR);
        }
        catch (Exception anyException) {
            mlog.write(CLASS_PREFIX + METHOD_NAME, "Exception when Setting up
                event:"+anyException, Log.ERROR);
            System.exit(1);
        }
        mlog.write(CLASS_PREFIX + METHOD_NAME, "END", Log.PROCEDURE);
    }

    //Method to Raise the event
    public void raiseEvent() {
        final String METHOD_NAME = "raiseEvent()";
        mlog.write(CLASS_PREFIX + METHOD_NAME, "BEGIN", Log.PROCEDURE);
        // raise the event

```

```
try {
mlog.write(CLASS_PREFIX + METHOD_NAME, "Raising Event",
           Log.STATEMENT);
mEvent.raise(mConnection);
}
catch (BusinessEventException exception) {
mlog.write(CLASS_PREFIX + METHOD_NAME, "Exception when raising
           event:"+exception,
Log.ERROR);
System.exit(1);
}
mlog.write(CLASS_PREFIX + METHOD_NAME, "END", Log.PROCEDURE);
}

// main()
public static void main(String[] args) {
HelloWorldPayloadRaise aBESTest = new HelloWorldPayloadRaise();
aBESTest.initConnection();
aBESTest.init();
aBESTest.raiseEvent();
aBESTest.close();
}
}
```

Event Data Generate Functions

Event Data Generate Functions

- A generate function for an event is a function that can produce the complete event data from the event name, event key, and an optional parameter list.
- The event data is stored as a CLOB and is typically structured as an XML document.
- The generate function is run if no event data is provided when the event is raised, but a subscription to the event requires the event data.
- Generate functions must follow a standard API.



ORACLE

Event Data Generate Functions

The Event Manager checks each subscription before executing it to determine whether the subscription's Rule Data property is set to Message, meaning that the complete event data is required. If the event data is required but is not already provided, the Event Manager calls the generate function for the event to produce the event data.

However, if each subscriptions to the event has the Rule Data property set to Key, then the generate function is not executed, even if the event has one defined.

Each event can have only one generate function, either PL/SQL or Java.

Oracle Workflow provides a standard generate function called WF_RULE.Default_Generate() that you can assign to events for demonstration and testing purposes.

Standard API for PL/SQL Event Data Generate Functions

Standard API for PL/SQL Event Data Generate Functions

All generate functions for events must follow a standard API format so that the Business Event System can properly generate the event data.

```
function <function_name>
  (p_event_name in varchar2,
   p_event_key in varchar2,
   p_parameter_list in wf_parameter_list_t
                      default null) return clob;
```

ORACLE

Standard API for PL/SQL Event Data Generate Functions

Arguments:

- p_event_name: The internal name of the event.
- p_event_key: A string generated when the event occurs within a program or application. The event key uniquely identifies a specific instance of the event.
- p_parameter_list: An optional list of additional parameter name and value pairs for the event.

If you define a PL/SQL generate function, then that function is used during both database and middle tier subscription processing. During middle tier subscription processing, the PL/SQL generate function will be executed through Java Database Connectivity (JDBC), if required.

For more information, see: Defining Procedures and Functions for Oracle Workflow, *Oracle Workflow Developer's Guide*.

Standard API for Java Event Data Generate Functions

Standard API for Java Event Data Generate Functions

- You can provide a Java generate function to be executed during subscription processing in the middle tier.
- A Java generate function must be a Java class using the following Java interface:

```
public interface GenerateInterface
{
    public String generate(BusinessEvent event,
                          WorkflowContext context)
        throws BusinessEventException;
}
```

ORACLE

Standard API for Java Event Data Generate Functions

Arguments:

- Event: The business event object.
- Context: Workflow context information, including the Log object which can be used for logging.

If you define a Java generate function, then the Business Event System will always perform subscription processing for the event in the middle tier to enable execution of the Java generate function, even if the subscriptions to the event are all PL/SQL subscriptions.

For more information, see: Defining Procedures and Functions for Oracle Workflow, *Oracle Workflow Developer's Guide*.

Example

The following example code shows a Java class that can be used as a generate function.

```
import oracle.apps.fnd.wf.bes.GenerateInterface;
import oracle.apps.fnd.wf.bes.BusinessEvent;
import oracle.apps.fnd.wf.bes.BusinessEventException;
import oracle.apps.fnd.wf.common.WorkflowContext;
```

```
public class HelloWorldGenerate implements GenerateInterface{  
  
    public String mXMLMessage;  
    public static String GENERATE_EXCEPTION =  
"GENERATE_EXCEPTION";  
    public static String ERROR = "ERROR";  
    public static String SUCCESS = "SUCCESS";  
  
    public void HelloWorldGenerate() {  
        mXMLMessage = new String();  
    }  
  
    public String getXMLMessage() {  
        return (mXMLMessage);  
    }  
  
    public void setXMLMessage(String aValue) {  
        mXMLMessage = aValue;  
    }  
  
    protected String goGenerate() {  
        setXMLMessage("<?xml version=\"1.0\"?>"+  
        "<content> Hello, World ! </content>");  
        return SUCCESS;  
    }  
  
    public String generate(BusinessEvent event,  
                           WorkflowContext context)  
        throws BusinessEventException {  
  
        //Generate the XML Value  
        String retValue = goGenerate();  
  
        if (retValue == SUCCESS)  
            return (mXMLMessage);  
        else if (retValue == GENERATE_EXCEPTION)  
            throw new BusinessEventException("Generate Exception");  
        else  
            return (null);  
    }  
}
```

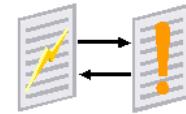
}

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Queue Handlers

Queue Handlers

- A queue handler translates between the standard **WF_EVENT_T** or **BusinessEvent** event message structure and the message format required by a particular queue.
- Oracle Workflow provides two standard PL/SQL queue handlers.
- You can create custom queue handlers for queues that use other payload types.
- Queue handler packages must include an enqueue procedure and a dequeue procedure, which must both follow a standard API format.



ORACLE

Queue Handlers

Oracle Workflow provides the following standard PL/SQL queue handlers:

- **WF_EVENT_QH**: For queues that use the **WF_EVENT_T** datatype as their payload type.
- **WF_EVENT_OJMSTEXT_QH**: For queues that use the **SYS.AQ\$_JMS_TEXT_MESSAGE** datatype as their payload type. This queue handler enables communication of JMS Text messages through the Business Event System.

You can define custom PL/SQL queue handlers or custom Java queue handlers. Each agent can have only one queue handler, either PL/SQL or Java. An agent with a PL/SQL queue handler can be monitored by both PL/SQL and Java agent listeners. However, an agent with a Java queue handler can be monitored only by Java agent listeners.

Standard APIs for PL/SQL Queue Handlers

Standard APIs for PL/SQL Queue Handlers

All queue handler APIs must follow the standard API format so that the Business Event System can properly translate messages on the queue.

- Enqueue:

```
procedure enqueue (p_event in wf_event_t,  
                  p_out_agent_override in wf_agent_t);
```

- Dequeue:

```
procedure dequeue (p_agent_guid in raw,  
                  p_event out wf_event_t);
```

ORACLE

Standard APIs for PL/SQL Queue Handlers

Enqueue arguments:

- p_event: The event message.
- p_out_agent_override: The outbound agent on whose queue the event message should be enqueued, overriding the outbound agent specified within the event message.

Dequeue arguments:

- p_agent_guid: The globally unique identifier of the inbound agent from whose queue the event message should be dequeued.
- p_event: The event message.

For more information, see: Defining Procedures and Functions for Oracle Workflow, *Oracle Workflow Developer's Guide*.

Standard APIs for Java Queue Handlers

Standard APIs for Java Queue Handlers

- You can provide a Java queue handler to be executed by a Java agent listener in the middle tier.
- A Java queue handler must be a Java class using the following Java interface:

```
public interface QueueHandlerInterface
```

ORACLE

Standard APIs for Java Queue Handlers

A Java queue handler must use the following interface:

```
public interface QueueHandlerInterface
{
    public void init(Connection conn,
                      AgentEO      agent,
                      Log          log,
                      String        uniqueLogId,
                      Properties    props)
                  throws QueueHandlerException;

    public String enqueue(BusinessEvent event)
                  throws QueueHandlerException;

    public BusinessEvent dequeue()
```

```
throws QueueHandlerException;

public void destroy();

public String getMsgId()
    throws QueueHandlerException;

public CLOB getEventData();
}
```

In addition to the enqueue and dequeue APIs, a Java queue handler must also contain methods to initialize the connection to the agent, destroy objects created during queue processing after the processing is complete, retrieve a message ID from a message on the queue, and retrieve the CLOB reference to the event data of the last business event dequeued by the queue handler.

- init arguments:
 - conn: The JDBC connection used to establish the connection with the queue.
 - agent: The AgentEO object that contains the information for this agent and its queue.
 - log: The Log object which can be used for logging.
 - uniqueLogId: The log ID for recording debug messages.
 - props: The property mapping for enqueue and dequeue operations.
- enqueue argument:
 - event: The BusinessEvent object to be enqueued.

For more information, see: Defining Procedures and Functions for Oracle Workflow, *Oracle Workflow Developer's Guide*.

Subscription Rule Functions

Subscription Rule Functions

- A subscription rule function defines the processing that the subscription performs when the triggering event occurs.
- Oracle Workflow provides some standard rule functions.
- You can extend your subscription processing by creating custom rule functions.
- Rule functions must follow a standard API.



ORACLE

Subscription Rule Functions

A rule function may read or write to the event message or perform any other database action. However, you should never commit within a rule function. The Event Manager never issues a commit as it is the responsibility of the calling application to commit.

A rule function must not change the connection context in any way, including security and NLS settings.

If you want to send an event to a workflow process from within a custom rule function, call:

- WF_ENGINE.Event() to send the event message to a workflow process only
- WF_RULE.Default_Rule() to include the default subscription processing that can send the event message both to a workflow process and to an agent

If you want to send an event to an agent from within a custom rule function, call:

- WF_EVENT.Send() to send the event message to an agent only
- WF_RULE.Default_Rule() to include the default subscription processing that can send the event message both to a workflow process and to an agent

Each subscription can have only one rule function, either PL/SQL or Java.

Standard API for PL/SQL Subscription Rule Functions

Standard API for PL/SQL Subscription Rule Functions

All subscription rule functions must follow a standard API format so that the Business Event System can properly execute the subscription.

```
function <function_name>
  (p_subscription_guid in raw,
   p_event in out wf_event_t) return varchar2;
```

ORACLE

Standard API for PL/SQL Subscription Rule Functions

A PL/SQL rule function for an event subscription must have the following standard API:

```
function <function_name>
  (p_subscription_guid in raw,
   p_event in out WF_EVENT_T) return varchar2 is

  <local declarations>

begin

  <your executable statements>

  <optional code for WARNING>
    WF_CORE.CONTEXT('<package name>', '<function name>',
                    p_event.getEventName( ),
```

```

        p_subscription_guid);
WF_EVENT.setErrorInfo(p_event, 'WARNING');
return 'WARNING';

return 'SUCCESS';

exception
when others then
WF_CORE.CONTEXT('<package name>', '<function name>',
                p_event.getEventName( ),
                p_subscription_guid);
WF_EVENT.setErrorInfo(p_event, 'ERROR');
return 'ERROR';

end;

```

Arguments:

- p_subscription_guid: The globally unique identifier for the subscription.
- p_event: The event message.

The function must return one of the following status codes:

- SUCCESS: The rule function completed successfully.
- WARNING: A warning condition occurred. The rule function reports a warning message using the Workflow Core error APIs and sets the warning information into the event message. The Event Manager places a copy of the event message on the WF_ERROR queue, but subscription processing continues.
- ERROR: An error occurred. The rule function reports an error message using the Workflow Core error APIs and sets the error information into the event message. The Event Manager halts subscription processing for this event, rolls back any subscriptions already executed for the event, and places the event message on an error queue.

For more information, see: Defining Procedures and Functions for Oracle Workflow, *Oracle Workflow Developer's Guide*.

Standard API for Java Subscription Rule Functions

Standard API for Java Subscription Rule Functions

- You can provide a Java subscription rule function to be executed in the middle tier.
- A Java subscription rule function must be a Java class using the following Java interface:

```
public interface SubscriptionInterface
{
    void onBusinessEvent(Subscription eo,
                         BusinessEvent event,
                         WorkflowContext ctx)
        throws BusinessEventException;
}
```

ORACLE

Standard API for Java Subscription Rule Functions

Arguments:

- eo: The Subscription object, which provides information about the subscription such as the phase number and any subscription parameters.
- event: The BusinessEvent object, which provides information about the business event that occurred, including the event name, event key, event data, and the optional payload object.
- ctx: Workflow context information, including the Log object which can be used for logging.

For more information, see: Defining Procedures and Functions for Oracle Workflow, *Oracle Workflow Developer's Guide*.

Example

The following example code shows a simple Java class that can be used as a subscription rule function.

```
import java.sql.Connection;
import oracle.apps.fnd.wf.bes.BusinessEvent;
```

```
import oracle.apps.fnd.wf.bes.BusinessEventException;
import oracle.apps.fnd.wf.bes.SubscriptionInterface;
import oracle.apps.fnd.wf.bes.server.Subscription;
import oracle.apps.fnd.wf.common.WorkflowContext;

public class HelloWorldSub implements SubscriptionInterface
{
    private final String CLASS_PREFIX =
        HelloWorldSub.class.getName() + ".';

    private final String HELLO_WORLD =
        "Hello World From JBES Subscription!";

    public void onBusinessEvent(Subscription eo,
                                BusinessEvent event,
                                WorkflowContext context)
        throws BusinessEventException
    {
        final String METHOD_NAME = "onBusinessEvent()";
        System.out.println(HELLO_WORLD);
    }
}
```

Predefined Subscription Rule Functions

Predefined Subscription Rule Functions

- Oracle Workflow provides some standard rule functions that you can use for basic subscription processing, testing and debugging, or other purposes.
- The following rule function APIs are defined in a PL/SQL package called WF_RULE:
 - Default_Rule
 - Default_Rule2
 - Default_Rule3
 - Default_Rule_Or
 - Error
 - Warning
 - Success
 - SetParametersIntoParameterList
 - Log
 - Workflow_Protocol
 - Error_Rule
 - SendNotification
 - Instance_Default_Rule



ORACLE

Predefined Subscription Rule Functions

PL/SQL rule functions in the WF_RULE package:

- Default_Rule: Performs default subscription processing on the event message for an event subscription, including:
 - Sending the event message to a workflow process, if specified in the subscription definition
 - Sending the event message to an agent, if specified in the subscription definition
- Default_Rule2: Performs the default subscription processing only if the PARAMETER_LIST attribute of the event message includes parameters whose names and values match all the parameters defined for the subscription.
- Default_Rule3: Sets the parameter name and value pairs from the subscription parameters into the PARAMETER_LIST attribute of the event message, and then performs the default subscription processing with the modified event message.
- Default_Rule_Or: Performs the default subscription processing only if the PARAMETER_LIST attribute of the event message includes at least one parameter whose name and value match a parameter defined for the subscription.

- Error: Returns the status code ERROR and sets a specified error message into the event message.
- Warning: Returns the status code WARNING and sets a specified error message into the event message.
- Success: Returns the status code SUCCESS.
- SetParametersIntoParameterList: Sets the parameter name and value pairs from the subscription parameters into the PARAMETER_LIST attribute of the event message, except for any parameter named ITEMKEY or CORRELATION_ID. For a parameter with one of these names, the function sets the CORRELATION_ID attribute of the event message to the parameter value.
- Log: Outputs the contents of the event message to a SQL*Plus session using DBMS_OUTPUT.put_line.
- Workflow_Protocol: Sends the event message to the workflow process specified in the subscription, which should in turn send the event message to the inbound agent specified in the subscription.
- Error_Rule: Performs the same subscription processing as Default_Rule, but reraises any exception that is encountered.
- SendNotification: Sends a notification based on message information specified in the event parameters. You can use this function to send notifications outside of a workflow process.
- Instance_Default_Rule: Sends the event to all existing workflow process instances that have eligible receive event activities waiting to receive it, marked by a business key attribute.

Oracle Workflow also provides a standard Java rule function called oracle.apps.fnd.wf.bes.WebServiceInvokerSubscription that is used in event subscriptions that invoke a Web service. You can add custom processing if necessary by extending this class and overriding its methods.

For more information, see: Business Event System APIs, *Oracle Workflow API Reference*.

Event APIs

Event APIs

- **The Event APIs can be called by an application program or a workflow process in the run-time phase to communicate with the Business Event System and manage events.**
- **The following APIs are defined in a PL/SQL package called WF_EVENT:**
 - Raise
 - Raise3
 - Send
 - NewAgent
 - Test
 - Enqueue
 - Listen
 - SetErrorInfo
 - SetDispatchMode
 - AddParameterToList
 - AddParameterToListPos
 - GetValueForParameter
 - GetValueForParameterPos
 - SetMaxNestedRaise
 - GetMaxNestedRaise
 - GetParamListFromString



ORACLE

Event APIs

- Raise: Raises a local event to the Event Manager.
- Raise3: Raises a local event to the Event Manager and returns the parameter list for the event.
- Send: Sends an event message from one agent to another.
- NewAgent: Creates a WF_AGENT_T structure for the specified agent and sets the agent's system and name into the structure.
- Test: Tests for the most costly data requirement among subscriptions to an event.
- Enqueue: Enqueues an event message onto a queue associated with an outbound agent.
- Listen: Monitors an agent for inbound event messages and dequeues messages using the agent's queue handler.
- SetErrorInfo: Retrieves error information from the error stack and sets it into the event message.
- SetDispatchMode: Sets the dispatch mode of the Event Manager to either deferred or synchronous subscription processing.

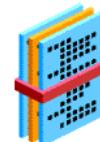
- AddParameterToList: Adds the specified parameter name and value pair to the end of the specified parameter list varray.
- AddParameterToListPos: Adds the specified parameter name and value pair to the end of the specified parameter list varray and returns the index for the position at which the parameter is stored within the varray.
- GetValueForParameter: Retrieves the value of the specified parameter from the specified parameter list varray.
- GetValueForParameterPos: Retrieves the value of the parameter stored at the specified position in the specified parameter list varray.
- SetMaxNestedRaise: Sets the maximum number of nested raises that can be performed to the specified value. A nested raise occurs when one event is raised and a Local subscription to that event is executed and raises another event.
- GetMaxNestedRaise: Returns the maximum number of nested raises that can currently be performed.
- GetParamListFromString: Parses a string of text containing parameters and returns the parsed parameters in a varray using the WF_PARAMETER_LIST_T abstract datatype.

For more information, see: Business Event System APIs, *Oracle Workflow API Reference*.

Event Function APIs

Event Function APIs

- **The Event Function APIs provide utility functions that can be called by an application program, the Event Manager, or a workflow process in the run-time phase to communicate with the Business Event System and manage events.**
- **The following APIs are defined in a PL/SQL package called WF_EVENT_FUNCTIONS_PKG:**
 - **Parameters**
 - **SubscriptionParameters**
 - **AddCorrelation**
 - **Generate**
 - **Receive**



ORACLE

Event Function APIs

- Parameters: Parses a string of text containing parameters and returns the parsed parameters in a varray.
- SubscriptionParameters: Returns the value for a parameter from a text string containing the parameters defined for an event subscription.
- AddCorrelation: Adds a correlation ID to an event message when used as a rule function for subscription processing.
- Generate: Generates the event data for events in the Seed Event Group.
- Receive: Receives Business Event System object definitions when used as a rule function for subscription processing and loads the definitions into the appropriate Business Event System tables.

For more information, see: Business Event System APIs, *Oracle Workflow API Reference*.

Adding a Correlation ID to an Event Message

Adding a Correlation ID to an Event Message

To add a correlation ID into an event message, you can:

- Define a subscription to the relevant event with the rule function `WF_EVENT_FUNCTIONS_PKG.AddCorrelation`.
- In a custom rule function, call the `setCorrelationID` method for the `WF_EVENT_T` object to set a value into the correlation ID attribute.



ORACLE

Adding a Correlation ID to an Event Message

For example, you may want to add a correlation ID to an event message to associate this message with other messages, or to set the value that will be used as the item key if the event message is sent to a workflow process.

`WF_EVENT_FUNCTIONS_PKG.AddCorrelation`

1. Define a subscription to the relevant event with the rule function

`WF_EVENT_FUNCTIONS_PKG.AddCorrelation` and a low phase number, such as 10. Enter a subscription parameter with the name `ITEMKEY` and the value `<package_name.function_name>`, replacing `<package_name.function_name>` with the package and function that will generate the correlation ID. This function should return the correlation ID as a `VARCHAR2` value.

2. Then define another subscription to the event with a higher phase number, such as 20. In this subscription, specify the main processing that you want to perform for the event, such as sending the event to a workflow process. When the event is raised, the subscription with the lower phase number will be executed first and will add a correlation ID to the event message. When the event is passed to the second subscription, that correlation ID will be used as the item key.

WF_EVENT_T setCorrelationID

The following code example shows a sample rule function that calls the setCorrelationID method in the WF_EVENT_T object type to add a correlation ID to the event message during execution of this subscription. You can use this method when you want to add a correlation ID as part of the same subscription that may also perform other processing.

Note that this sample function sets a hard-coded value for the correlation ID. However, you can expand on this example to add logic that dynamically determines the value you want to set for the correlation ID.

```
function testrule (p_subscription_guid in raw,
                  p_event in out nocopy wf_event_t)
                  return varchar2
is
  l_parameter_list wf_parameter_list_t;
  lEventData clob;
  l_corrid varchar2(200);

begin
  dbms_output.put_line('Executing Rule Function');
  p_event.setCorrelationID('TEST1');

  dbms_output.put_line('After Executing Rule Function');
  return 'SUCCESS';
...
end;
```

For more information, see: Business Event System APIs, *Oracle Workflow API Reference*.

Business Event System Cleanup API

Business Event System Cleanup API

- One of the setup steps for the Business Event System is to periodically clean up the standard WF_CONTROL queue by removing inactive subscribers from the queue.
- The Business Event System Cleanup_Subscribers API can be used to clean up WF_CONTROL.
- The Cleanup_Subscribers API is defined in a PL/SQL package called WF_BES_CLEANUP.



ORACLE

Business Event System Cleanup API

For more information, see: Business Event System APIs, *Oracle Workflow API Reference*.

Summary

Summary

In this lesson, you should have learned how to:

- **Describe the abstract datatypes used by the Business Event System.**
- **Define event data generate functions.**
- **Define queue handler procedures.**
- **Define subscription rule functions.**
- **Apply Business Event System APIs.**

ORACLE®

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Error Handling

Chapter 19

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Error Handling

Error Handling

ORACLE®

Objectives

Objectives

After completing this lesson, you should be able to do the following:

- **Describe error handling for workflow processes.**
- **Describe error handling for subscription processing.**
- **Explain the standard error handling processes provided in the System: Error item type.**

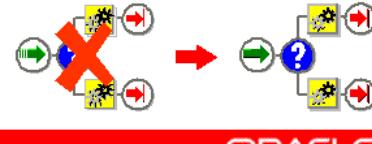
ORACLE®

Error Handling for Workflow Processes

Error Handling for Workflow Processes

If an activity error occurs during workflow process execution, the Workflow Engine:

- Rolls back to the pre-activity savepoint.
- Sets the status of the activity to ERROR.
- Attempts to run an error process.



ORACLE

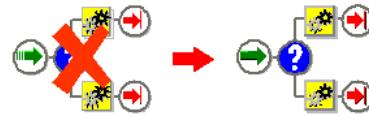
Error Handling for Workflow Processes

To identify an error process to run, the Workflow Engine checks for an associated error process in the activity details. The engine starts with the activity that caused the error and then checks each parent process activity until it finds an associated error process. You can associate an error process with an activity in the activity details property page in the Workflow Builder.

Error Handling for Workflow Processes

Error Handling for Workflow Processes

- **The System: Error item type provided by Oracle Workflow** that contains processes that you can use for generic error handling.
 - Default Error Process
 - Retry-only Process
- You cannot edit the predefined **System: Error** error processes.
- You can add custom error processes to the **System: Error** item type or to any other item type.



ORACLE

Error Handling for Workflow Processes

Although you cannot edit the predefined error processes, you can define two item attributes in your own item type (the item type during which an error may occur) to control the error processing that the processes perform. These two item attributes are:

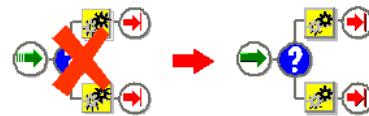
- WF_ADMINISTRATOR: Specify the role to which Oracle Workflow sends the error notification. The default is the SYSADMIN role.
- ERROR_TIMEOUT: Specify whether the error notification times out.

Default Error Process

Default Error Process

The Default Error Process:

- Sends an administrator a notification when an error occurs in a workflow process.
- Provides information about the error to the administrator.
- Enables the administrator to abort the process, retry the errored activity, or resolve the problem that caused the error.
- Provides a link to the Status Monitor for more complex processing with the administration tools there.
- Automatically terminates when the error is resolved.



ORACLE

Default Error Process

The notification to the administrator includes the following information:

- A request to either retry or abort the process
- Item type
- Item key
- User key
- Error name
- Error message
- Error stack
- Activity ID
- Activity label
- Result code
- Notification ID
- Assigned user

- Monitor URL to enable the administrator to navigate to the error process in the Status Monitor

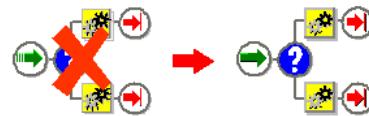
Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Retry-only Process

Retry-only Process

The Retry-only Process:

- Sends an administrator a notification when an error occurs in a workflow process.
- Provides information about the error to the administrator.
- Prompts the administrator to retry the errored activity.
- Provides a link to the Status Monitor for more complex processing with the administration tools there.
- Automatically terminates when the error is resolved.



ORACLE

Retry-only Process

The notification to the administrator includes the following information:

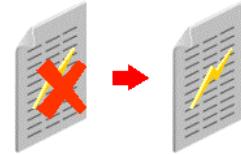
- A request to retry the process
- Item type
- Item key
- User key
- Error name
- Error message
- Error stack
- Activity ID
- Activity label
- Result code
- Notification ID
- Assigned user

- Monitor URL to enable the administrator to navigate to the error process in the Status Monitor

Error Handling for Subscription Processing

Error Handling for Subscription Processing

- The Event Manager uses the status codes returned or exceptions raised by subscription rule functions to monitor the status of subscription processing for an event.
- A PL/SQL subscription rule function can return the following status codes:
 - **SUCCESS**
 - **WARNING**
 - **ERROR**
- A Java subscription rule function can return a **BusinessEventException** for an error.



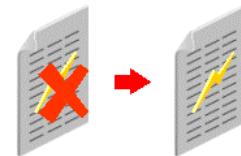
ORACLE

Error Handling for Subscription Processing

Error Handling for Subscription Processing

When you define a subscription, you can specify the type of error handling to perform if the Event Manager encounters an error while processing the subscription.

- Stop and Rollback
- Skip to Next



ORACLE

Stop and Rollback Error Handling

Stop and Rollback Error Handling

- In Stop and Rollback error handling, the Event Manager halts all subscription processing for the event and rolls back any subscriptions already run for the event.
- If the subscription trapped an error and returned a PL/SQL ERROR status code or a Java BusinessEventException, the Event Manager places the event message on the standard WF_ERROR or WF_JAVA_ERROR agent, as appropriate.
- If the subscription raised an unhandled exception, the Event Manager raises that exception to the calling application.



ORACLE

Stop and Rollback Error Handling

Stop and Rollback Error Handling

If you later retry subscription processing for the event, the Event Manager reruns all subscriptions to the event.



ORACLE®

Stop and Rollback Error Handling

Rolling back all subscription processing enables you to handle the error by retrying the event, if appropriate. In this way, Oracle Workflow ensures that no subscription is duplicated when subscription processing is restarted.

Skip to Next Error Handling

Skip to Next Error Handling

- In Skip to Next error handling, the Event Manager rolls back only this subscription.
- The Event Manager then places the event message on the standard WF_ERROR or WF_JAVA_ERROR agent, regardless of whether the subscription trapped an error or raised an unhandled exception. The exception is not raised to the calling application.
- Finally, the Event Manager continues processing the next subscription for the event according to the subscription phase order.



ORACLE

Skip to Next Error Handling

Note: Skip to Next error handling is not available for subscriptions with a source type of Error. If an additional error occurs during such a subscription, processing for that event is halted until you address the error.

Skip to Next Error Handling

Skip to Next Error Handling

If you later retry subscription processing for the event, the Event Manager reruns only the errored subscription.



ORACLE

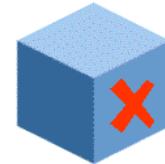
Skip to Next Error Handling

Skipping to the next subscription lets you continue processing without waiting in cases where the calling application does not depend on the successful completion of a subscription and when the subscription does not impact the running of any other subscriptions to the same event.

Standard Error Agents

Standard Error Agents

- **WF_ERROR and WF_JAVA_ERROR are standard agents for error handling provided by Oracle Workflow.**
- **Predefined agent listener service components monitor these agents and dequeue messages from their queues.**
- **All event messages that are dequeued from the WF_ERROR or WF_JAVA_ERROR queues are assigned a source type of Error.**



ORACLE®

Standard Error Agents

The WF_ERROR agent is used for error handling during subscription processing in the database. The WF_JAVA_ERROR agent is used for error handling during subscription processing in the middle tier.

Use Workflow Manager to ensure that the agent listener service components for the WF_ERROR agent and WF_JAVA_ERROR agent are running.

Error Handling Subscriptions

Error Handling Subscriptions

- When an event is dequeued from the WF_ERROR or WF_JAVA_ERROR queue, the Event Manager searches for any subscriptions by the local system to that event or to the Any event with the source type Error.
- If no subscriptions are found, the Event Manager runs any subscriptions by the local system to the Unexpected event with the source type Error.
- A predefined Error subscription to the Unexpected event sends the event message to the Default Event Error process in the System: Error item type.



ORACLE

Error Handling Subscriptions

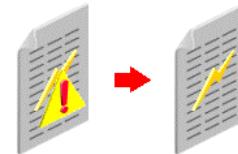
You must not change or disable the definition of the Unexpected event (oracle.apps.wf.event.unexpected) or of the predefined Error subscription to that event. If you disable this subscription, then the Event Manager cannot perform default error handling for subscription processing.

You can set up custom error handling for a particular event by defining a subscription to that event with a source type of Error and specifying the custom processing that you want to run as the subscription action.

Warning Conditions in Subscription Processing

Warning Conditions in Subscription Processing

- If a PL/SQL rule function returns a **WARNING** status code, the Event Manager places a copy of the event message on the standard **WF_ERROR** agent and then continues subscription processing for the event.
- The predefined Error subscription to the Unexpected event sends events with warnings to the Default Event Error process, just as it does for errored events.



ORACLE

Warning Conditions in Subscription Processing

The **WARNING** status code indicates that a warning condition occurred but that the subscription processing was completed without a blocking error.

You can also define custom Error subscriptions to perform custom handling for warning conditions.

Note: Both the **WARNING** status code and the Skip to Next error handling type enable subscription processing to continue. However, in Skip to Next error handling, the Event Manager rolls back the processing performed for the errored subscription before continuing, while for the **WARNING** status code, the Event Manager completes the subscription that returned the warning and continues without rolling back any subscription processing.

Unexpected Events

Unexpected Events

- If an event is received from an external source, but the local system does not have any subscriptions to that event, Oracle Workflow automatically searches for subscriptions to the predefined Unexpected event with a source type of External.
- A predefined External subscription to the Unexpected event sends the event message to the Default Event Error process in the System: Error item type.



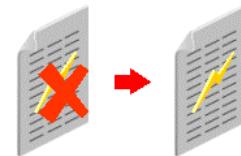
ORACLE®

Default Event Error Process

Default Event Error Process

The Default Event Error Process:

- Sends an administrator a notification when an error or warning condition occurs during event subscription processing.
- Provides information to the administrator about the error.
- Enables the administrator to abort or retry the event subscription processing, depending on the error type.



ORACLE

Default Event Error Process

The Default Event Error Process starts when the Event Manager sends an errored event message to the process. The workflow begins with the Receive Errored Queue Message activity.

Next, the process determines the error type:

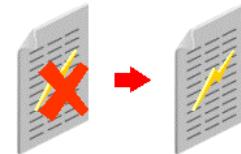
- Event Warning
- External Event Error
- Local Event Error

Then the process notifies the administrator of the error, sending different messages depending on the error type.

Event Warnings

Event Warnings

- For a warning condition, the Default Event Error Process sends the administrator a warning message.
- The warning message is an FYI notification and does not require a response.

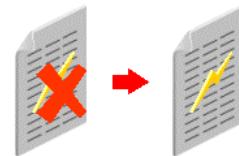


ORACLE

External Event Errors

External Event Errors

- For an error in subscription processing for an external event, the Default Event Error Process sends the administrator an error message that requests a response.
- The administrator can:
 - Abort subscription processing for the event.
 - Enqueue the event message back onto the queue where it was originally received to retry subscription processing.



ORACLE

External Event Errors

Errors in Message Groups on Transactional Queues

Before sending a notification to the administrator for an external event, the Default Event Error Process first checks whether the event message belongs to a message group on a queue that is enabled for transactional grouping. If so, the process sends the administrator an FYI notification indicating that an error occurred while processing a message that belonged to a message group. In this case the event message should not be aborted or retried individually. The administrator should investigate the error for the entire message group. For more information about transactional queues, see: *Oracle Streams Advanced Queuing User's Guide*.

Note: The standard Business Event System queues provided Oracle Workflow are not enabled for transactional grouping.

Errors in Messages on Non-transactional Queues

For an error in an external event on a non-transactional queue, the Default Event Error Process sends the administrator an error message that requests a response

- Abort - For example, if the event data contained in an event message is corrupt, then the system administrator can abort subscription processing on that event message.

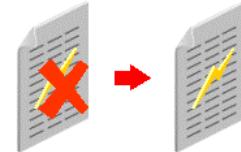
- Enqueue Event - The event message is enqueued on its original queue with a priority of -1 so that it will be the first message to be dequeued the next time that the listener runs. The system administrator can attempt to correct the error before re-enqueuing the event. For example, the system administrator can create a subscription to handle an unexpected event and then re-enqueue the event message to trigger the new subscription.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Local Event Errors

Local Event Errors

- For an error in subscription processing for a local event, the Default Event Error Process sends the administrator an error message that requests a response.
- The administrator can:
 - Abort subscription processing for the event.
 - Re-raise the event with the event name and key.
 - Re-raise the event with the event name, key, and data.
 - Re-raise the event with the event name, key, data, and parameters.



ORACLE

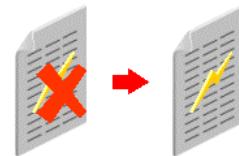
Local Event Errors

By re-raising the event, the system administrator can retry subscription processing for the event. The system administrator can choose the level of information to provide to the Event Manager when re-raising the event. For example, if an error exists in the event data that was originally provided, the event can be re-raised with only the event name and the event key, forcing the Event Manager to regenerate the event data using the event's Generate function. The system administrator can also attempt to correct the error before re-raising the event.

Default Event Error Process (One Retry Option)

Default Event Error Process (One Retry Option)

- **The Default Event Error Process (One Retry Option) is an alternative process you can optionally use in custom error handling.**
- **This process is similar to the Default Event Process, except that this process:**
 - Provides specific Web service details for any errors in subscriptions that invoke Web services.
 - Presents a simpler administrative notification for local errored events.



ORACLE

Default Event Error Process (One Retry Option)

The Default Event Error Process (One Retry Option) checks whether the error occurred during a subscription whose action is to invoke a Web service, and if so, includes specific Web service details in the error notification sent to the administrator.

The Default Event Error Process (One Retry Option) also presents a simpler administrative notification for local errored events than the standard Default Event Error Process. The Default Event Error Process provides several different retry options with different levels of information. The Default Event Error Process (One Retry Option) simplifies the notification administrators receive by providing only the option to retry the event with the event name, event key, event data, and parameters.

This process is not used in the standard Oracle Workflow error handling. However, you can define custom error handling with this process by defining a subscription to the relevant event with a source type of Error and specifying that the subscription action is to launch the Default Event Error Process (One Retry Option) workflow.

Summary

Summary

In this lesson, you should have learned how to:

- **Describe error handling for workflow processes.**
- **Describe error handling for subscription processing.**
- **Explain the standard error handling processes provided in the System: Error item type.**

ORACLE®

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

PL/SQL Documents

Chapter 20

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

PL/SQL Documents

PL/SQL Documents

ORACLE®

Objectives

Objectives

After completing this lesson, you should be able to do the following:

- **Integrate PL/SQL documents into a workflow process.**
- **Include PL/SQL documents in messages.**
- **Define procedures to generate PL/SQL documents.**

ORACLE®

PL/SQL Documents

PL/SQL Documents

- Use a PL/SQL document to represent data that you want to integrate within a workflow process when the format or content of the document may vary.
- You can define three types of PL/SQL documents:
 - PL/SQL document: Represents data from the database as a character string generated from a PL/SQL procedure (up to 32 K).
 - PL/SQL CLOB document: Represents data from the database as a character large object (CLOB) generated from a PL/SQL procedure (up to 4 GB).
 - PL/SQL BLOB document: Represents data from the database as a binary large object (BLOB) generated from a PL/SQL procedure (up to 4 GB).



ORACLE

PL/SQL Documents

Select what type of document to define depending on the data file that you want to include:

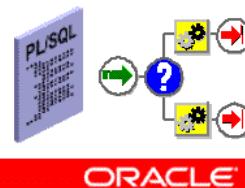
- A PL/SQL document can contain a plain text character string.
- A PL/SQL CLOB document can contain plain text, HTML, an Adobe Acrobat Portable Document Format (PDF) document, a Microsoft Rich Text Format (RTF) document, or binary data encoded to base64.
- A PL/SQL BLOB document can contain an image or other types of application files that are stored as binary data.

Note: The maximum length of the data that a PL/SQL document can contain is 32 kilobytes. If you expect your document to exceed 32 Kb, you should use a PL/SQL CLOB document to hold the data instead.

Integrating PL/SQL Documents into Workflow Processes

Integrating PL/SQL Documents into Workflow Processes

- To integrate a **PL/SQL**, **PL/SQL CLOB**, or **PL/SQL BLOB** document into a workflow process, define an attribute of type **document** for an item type or a message.
- The document attribute value tells Oracle Workflow how to construct the dynamic call to a **PL/SQL** procedure that generates the document.
- You can embed message attributes of type **Document** within the body of a message or include the attributes with the message as attachments, depending on their content.



Integrating PL/SQL Documents into Workflow Processes

The value for an attribute of type **document** must follow a specific format, depending on the type of PL/SQL document that the attribute will contain:

- PL/SQL document:
`PLSQL:<procedure>/<document_identifier>`
- PL/SQL CLOB document:
`PLSQLCLOB:<procedure>/<document_identifier>`
- PL/SQL BLOB document:
`PLSQLBLOB:<procedure>/<document_identifier>`

In these values:

- Replace `<procedure>` with the PL/SQL package and procedure name in the form: `package.procedure`.
- Replace `<document_identifier>` with the PL/SQL argument string that you want to pass directly to the procedure. The argument string should identify the document. For example: `PLSQL:PO_WF.SHOW_REQ/2034`.

If you want to generate the PL/SQL argument string dynamically, create another item attribute and specify `&ITEM_ATTRIBUTE` in place of the *PL/SQL* argument string. The item attribute name must be in upper case.

For example: `PLSQL:PO_WF.SHOW_REQ/&POREQ_NUMBER`

Before you execute any activity that references this other item attribute, call the `WF_ENGINE.SetItemAttribute` API to set the PL/SQL argument string value dynamically.

If you want to reference multiple item attributes to generate the PL/SQL argument string, separate them with a colon.

For example: `PLSQL:PO_WF.SHOW_REQ/&POREQ_NUMBER:&PO_REQ_SITEID`

Including PL/SQL Documents in Messages

Including PL/SQL Documents in Messages

- You can display the following types of documents within the text of a notification message:
 - PL/SQL documents
 - PL/SQL CLOB documents that contain text or HTML
- You can include all types of PL/SQL documents with a notification message as an attachment, including:
 - PL/SQL documents
 - PL/SQL CLOB documents
 - PL/SQL BLOB documents



Including PL/SQL Documents in Messages

Oracle Workflow supports two ways to include message attributes of type Document in a notification message. The attributes must have a source of Send.

- Use token substitution to embed a PL/SQL document or a PL/SQL CLOB document containing text or HTML within the body of a notification message. Enter the token in the message body in the following format: &MESSAGE_ATTRIBUTE_NAME
- Select the Attach Content check box to attach the content of any type of PL/SQL document attribute to the notification message. In the Notification Details Web page, a document icon appears after the notification message body. A user can click that icon to display the contents of the attached message attribute. In e-mail, the presentation of the attachment will vary depending on the user's e-mail notification preference setting.

Note: A notification message can have both embedded and attached PL/SQL documents included in it at the same time. You are not limited to one or the other.

Standard API for a PL/SQL Document

Standard API for a PL/SQL Document

```
procedure <procedure name>
    (document_id      in varchar2,
     display_type     in varchar2,
     document         in out varchar2,
     document_type    in out varchar2)
```

ORACLE

Standard API for a PL/SQL Document

The procedure for generating a PL/SQL document must follow a standard API format:

- document_id: A string that uniquely identifies a document. This string is the same as the string that you specify in the document attribute value (PLSQL:<procedure>/<document_identifier>).
- display_type: Represents how the document appears in the notification (also referred to as the *content type* or *requested type*):
 - text/plain: The document is embedded inside a plain text representation of the notification. Note that the entire e-mail message including the document must be less than or equal to 32K.
 - text/html: The document is embedded inside an HTML representation of the notification as viewed from the Notification Details Web page or an HTML-formatted version of the e-mail message. The entire HTML representation of the document must be less than or equal to 32K and should not include top-level HTML tags like <HTML> or <BODY>, because the HTML page that the document is being inserted into already contains these tags. If the tags are included, Oracle Workflow removes them for you when the document attribute is referenced in a message body.

The procedure can alternatively generate a plain text document; in this case, the Notification System automatically surrounds the plain text with the appropriate HTML tags to preserve formatting.

- ‘ ‘: The document is presented as a separate attachment to the notification. Any content type can be returned.
- document: Outbound text buffer where up to 32K of document text is returned.
- document_type: Outbound text buffer where the document content type is returned. Also referred to as the returned type. If no type is supplied, then *text/plain* is assumed.

Standard API for a PL/SQL CLOB Document

Standard API for a PL/SQL CLOB Document

```
procedure <procedure name>
    (document_id      in varchar2,
     display_type     in varchar2,
     document         in out clob,
     document_type    in out varchar2)
```

ORACLE

Standard API for a PL/SQL CLOB Document

A PL/SQL CLOB document that you include as an attachment to a notification can contain a PDF or RTF document or other binary data that is encoded to base64. You should first store the document in the database as a binary large object (BLOB) and then convert the document into a CLOB as part of the PL/SQL procedure that generates the CLOB. You can use the UTL_RAW.Cast_To_VARCHAR2 function to convert the data from the BLOB into VARCHAR2 data that you write to a CLOB. You can optionally use the WF_MAIL_UTIL.EncodeBLOB procedure to encode the binary data to base64. See: UTL_RAW, *Oracle Supplied PL/SQL Packages and Types Reference* and EncodeBLOB, *Oracle Workflow API Reference*.

The procedure for generating a PL/SQL CLOB document must follow a standard API format:

- document_id: A string that uniquely identifies a document. This string is the same as the string that you specify in the document attribute value (PLSQLCLOB:<procedure>/<document_identifier>).
- display_type: Represents how the document appears in the notification (also referred to as the *content type* or *requested type*):

- text/plain: The document is embedded inside a plain text representation of the notification.
- text/html: The document is embedded inside an HTML representation of the notification as viewed from the Notification Details Web page or an HTML-formatted version of the e-mail message. The HTML representation of the document should not include top level HTML tags like <HTML> or <BODY> because the HTML page that the document is being inserted into already contains these tags. If the tags are included, Oracle Workflow removes them for you when the document attribute is referenced in a message body. The procedure can alternatively generate a plain text document; in this case the Notification System automatically surrounds the plain text with the appropriate HTML tags to preserve formatting.
- ‘ ‘: The document is presented as a separate attachment to the notification. Any content type can be returned.
- document: The outbound LOB locator pointing to where the document text is stored. This locator is a temporary LOB locator, so you must write your document text to this locator rather than replacing its value. If this value is overwritten, the temporary LOB is *not* implicitly freed. For more information, see Temporary LOBs, *Oracle Application Developer’s Guide – Large Objects (LOBs)*.
- document_type: Outbound text buffer where the document content type is returned. Also referred to as the returned type. If no type is supplied, then *text/plain* is assumed. For a PDF or RTF document, this argument should specify an appropriate Multi-purpose Internet Mail Extension (MIME) type, such as *application/pdf* or *application/rtf*. You can also optionally specify a file name for the attachment as part of this argument. Use a semicolon (;) to separate the file name from the preceding value in the argument, and specify the file name in the format *name=<filename>* with no spaces before or after the equal sign (=). For example, you can set a value for the document_type with the following command:

```
document_type := 'application/pdf; name=filename.pdf' ;
```

Note: If you are using the WF_MAIL_UTIL.EncodeBLOB API to encode binary data to base64 in order to store the data in this PL/SQL CLOB document, then the document_type parameter should specify an appropriate MIME type with a primary type of either application or image, such as *application/doc*, *application/pdf*, *image/jpg*, or *image/gif*. The MIME type must be followed by a semicolon (;) and then by the encoding specification *encoding=base64* with no spaces before or after the equal sign. You can also optionally specify a file name for the attachment as part of this argument. Use a semicolon (;) to separate the file name from the preceding value in the argument, and specify the file name in the format *name=<filename>* with no spaces before or after the equal sign (=). For example, you can set a value for the document_type with the following command:

```
document_type := 'image/jpg; encoding=base64;
name=filename.jpg' ;
```

Standard API for a PL/SQL BLOB Document

Standard API for a PL/SQL BLOB Document

```
procedure <procedure name>
    (document_id      in varchar2,
     display_type    in varchar2,
     document        in out blob,
     document_type   in out varchar2)
```

ORACLE

Standard API for a PL/SQL BLOB Document

The procedure for generating a PL/SQL BLOB document must follow a standard API format:

- document_id: A string that uniquely identifies a document. This string is the same as the string that you specify in the document attribute value (PLSQLBLOB:<procedure>/<document_identifier>).
- display_type: Represents how the document appears in the notification (also referred to as the “content type” or “requested type”). For a PL/SQL BLOB document, this value should be ‘ ’ to indicate that the document is presented as a separate attachment to the notification. Any content type can be returned.
- document: The outbound LOB locator pointing to where the document text is stored. This locator is a temporary LOB locator, so you must write your document text to this locator rather than replacing its value. If this value is overwritten, the temporary LOB is *not* implicitly freed. For more information, see Temporary LOBs, *Oracle Application Developer’s Guide – Large Objects (LOBs)*.
- document_type: The outbound text buffer where the document content type is returned. Also referred to as the returned type. If no type is supplied, then *text/plain* is assumed. This argument should specify an appropriate MIME type with a primary type of either

application or image, such as *application/doc*, *application/pdf*, *image/jpg*, or *image/gif*. You can also optionally specify a file name for the attachment as part of this argument. Use a semicolon (;) to separate the file name from the preceding value in the argument, and specify the file name in the format *name=<filename>* with no spaces before or after the equal sign (=). For example, you can set a value for the *document_type* with the following command:

```
document_type := 'image/jpg; name=filename.jpg';
```

Summary

Summary

In this lesson, you should have learned how to:

- **Integrate PL/SQL documents into a workflow process.**
- **Include PL/SQL documents in messages.**
- **Define procedures to generate PL/SQL documents.**

ORACLE®

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Forced Synchronous Processing

Chapter 21

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Forced Synchronous Processing

Forced Synchronous Processing

ORACLE®

Objectives

Objectives

After completing this lesson, you should be able to do the following:

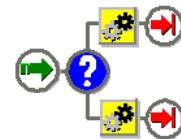
- **Describe forced synchronous processing.**
- **Follow process definition restrictions for forced synchronous processes.**

ORACLE®

Forced Synchronous Processes

Forced Synchronous Processes

- A synchronous process consists of consecutive activities in a single thread that are not deferred to the background engine.
- A forced synchronous process completes in a single SQL session and never inserts or updates workflow information in any database tables.
- Consequently, a forced synchronous process generates a result more quickly than a synchronous process.



ORACLE

Forced Synchronous Processing

To create a forced synchronous process, you must set the item key of your process to #SYNCH, or to wf_engine.eng_synch, which returns the #SYNCH constant when you call the necessary WF_ENGINE APIs. Because a forced synchronous process never writes to the database, it is not a problem to use a non-unique item key such as #SYNCH.

Also, because a forced synchronous process never writes workflow information to the database, a forced synchronous process completes faster than a normal synchronous process. However, for the same reason, you cannot view the process from the Status Monitor and no auditing is available for the process.

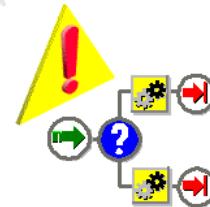
Note: Although the Workflow Engine does not write workflow information to the database during a forced synchronous process, you can still use custom function activities that write to database tables as part of a forced synchronous process.

Process Definition Restrictions

Process Definition Restrictions

Process definitions for forced synchronous processes must adhere to various restrictions:

- **No notification activities**
- **Limited blocking activity use**
- **No error processing**
- **No master/detail coordination activities**
- **No parallel flows**



ORACLE

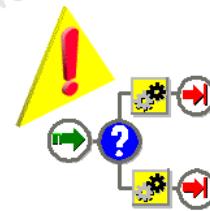
Process Definition Restrictions

- No notification activities are allowed.
- Limited blocking-type activities are allowed. A process can block and restart with a call to WF_ENGINE.CompleteActivity only if the blocking and restarting activities:
 - Occur in the same database session
 - Contain no intervening calls to Oracle Workflow
 - Contain no intervening commits
- No error processes can be assigned to the process or the process's activities.
- Each function activity behaves as if On Revisit is set to Loop and is executed in non-canceling mode, regardless of its actual On Revisit setting. Loops are allowed in the process.
- No master/detail coordination activities are allowed.
- No parallel flows are allowed in the process because transitions from each activity must have a distinct result. This also means that no <Any> transitions are allowed because they cause parallel flows.

Process Definition Restrictions

Process Definition Restrictions

- **Some standard activities are not supported.**
- **No deferred processing is allowed.**
- **No process status information is saved.**



ORACLE

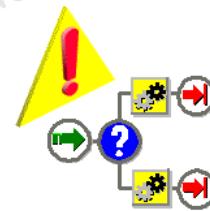
Process Definition Restrictions

- The following Standard activities are not allowed:
 - And
 - Block (restricted by the conditions stated for limited blocking)
 - Defer Thread
 - Wait
 - Continue Flow/Wait for Flow
 - Role Resolution
 - Voting
 - Compare Execution Time
 - Notify
- No use of the background engine is allowed; that is, activities are never deferred.
- No data is ever written to the Oracle Workflow tables, and as a result:
 - The process cannot be viewed from the Status Monitor.
 - No auditing is available for the process.

Process Definition Restrictions

Process Definition Restrictions

- Limited WF_ENGINE API calls:
 - CreateProcess
 - StartProcess
 - SetItemAttribute
 - GetItemAttribute
 - GetActivityAttribute
 - CompleteActivity



ORACLE

Process Definition Restrictions

- Only the following WF_ENGINE API calls are allowed to be made, and in all cases, the item key supplied to these APIs must be specified as #SYNCH or wf_engine.eng_synch:
 - WF_ENGINE.CreateProcess
 - WF_ENGINE.StartProcess
 - WF_ENGINE.GetItemAttribute
 - WF_ENGINE.SetItemAttribute
 - WF_ENGINE.GetActivityAttribute
 - WF_ENGINE.CompleteActivity (for the limited usage of blocking-type activities)
- WF_ENGINE API calls for any item except the current synchronous item are not allowed.

Summary

Summary

In this lesson, you should have learned how to:

- **Describe forced synchronous processing.**
- **Follow process definition restrictions for forced synchronous processes.**

ORACLE®

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Selector/Callback Functions

Chapter 22

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Selector/Callback Functions

Selector/Callback Functions

ORACLE®

Objectives

Objectives

After completing this lesson, you should be able to do the following:

- Define a selector/callback function for an item type.
- Create a selector/callback function, following a standard API.
- Call a selector/callback function.

ORACLE®

Item Type Selector/Callback Functions

Item Type Selector/Callback Functions

- An item type can have more than one runnable process activity associated with it.
- PL/SQL selector functions are used to determine which process activity to run in a particular situation.
- You can extend a selector function to be a general callback function so that you can reset the item type context information as needed if the SQL session is interrupted.



Item Type Selector/Callback Functions

A selector function is a PL/SQL procedure that automatically identifies the specific process definition to run when a workflow is initiated for a particular item type, but when no process name is provided. If your item type has more than one runnable process activity associated with it, define a PL/SQL selector function that determines which process activity to run in a particular situation.

For example, you may have two different requisition approval process activities associated with the same item type. The process that Oracle Workflow runs varies depending on how and where the requisition originates. Your selector function determines which process is appropriate in any given situation. This functionality gives you more flexibility at run time to determine dynamically which process to run.

You can also extend the selector function to be a general callback function so that you can reset item type context information as needed if the SQL session is interrupted during the execution of a process.

Defining a Selector/Callback Function for an Item Type

Defining a Selector/Callback Function for an Item Type

- You can associate a selector function with an item type in the item type property page when you define the item type in the Oracle Workflow Builder.
- Enter the selector function in the following format:
`<package_name>.<procedure_name>`



Defining a Selector/Callback Function for an Item Type

For example, the WFDEMO item type has a selector function named WF_REQDEMO.SELECTOR.

Standard API for a Selector/Callback Function

Standard API for a Selector/Callback Function

You can define one PL/SQL procedure that includes both selector and callback functionality by following a standard API.

```
procedure <procedure name>
  (itemtype    in    varchar2,
   itemkey     in    varchar2,
   actid       in    number,
   command     in    varchar2,
   resultout   out   varchar2)
```

ORACLE

Standard API for a Selector/Callback Function

```
procedure <procedure name>(
  itemtype    in    varchar2,
  itemkey     in    varchar2,
  actid       in    number,
  command     in    varchar2,
  resultout   out   varchar2) is
<local declarations>
begin
if (command = 'RUN') then
  <your RUN executable statements>
  resultout:='<Name of process to run>';
  return;
endif;
if (command = 'SET_CTX') then
  <your executable statements for establishing context information>
  resultout:=' ';
  return;
endif;
```

```
if (command = 'TEST_CTX') then
    <your executable statements for testing validity of current context
    information>
    resultout:='<TRUE or FALSE or NOTSET>';
    return;
endif;
if (command = '<other commands>') then
    resultout:=' ';
    return;
endif;
exception
when others then
    WF_CORE.CONTEXT ('<package name>', '<procedure name>', <itemtype>,
                      <itemkey>, to_char(<actid>), <command>);
    raise;
end <procedure name>;
```

Standard API Parameters

Standard API Parameters

- **itemtype:** The internal name for the item type.
- **itemkey:** A string that represents a primary key generated by the workflow-enabled application for the work item. The string uniquely identifies the item within an item type.
- **actid:** The ID number of the activity from which this procedure is called. This parameter is always null when the procedure is called with the RUN command to execute the selector function.



Standard API Parameters

Standard API Parameters

- **command:** The mode in which to execute the selector/callback function:
 - RUN
 - TEST_CTX
 - SET_CTX
- **resultout:** The result that is returned for the RUN command or the TEST_CTX command. The result value depends on the command used to call the selector/callback function.



Standard API Parameters

The different commands for a selector/callback function require different return values.

- If the function is called with the RUN command, the name of the process to run must be returned through the resultout parameter.
- If the function is called with the TEST_CTX command, then the code must return TRUE if the context is correct, FALSE if the context is incorrect, or NOTSET if the context has not been initialized yet.
- If the function is called with the SET_CTX command, then no return value is expected.

Selector/Callback Function Commands

- **RUN:** Selects the appropriate process to start when either of the following conditions occurs:
 - A process is not explicitly passed when `WF_ENGINE.CreateProcess` is called.
 - A process is implicitly started by `WF_ENGINE.CompleteActivity` with no prior call to `WF_ENGINE.CreateProcess`.
- **TEST_CTX:** Determines if the current item type context is correct before executing a function.
- **SET_CTX:** Establishes any context information that a function activity in an item type requires before it can be executed (before a new database session begins).



Selector/Callback Function Commands

Use the TEST_CTX command when you want to check that the user context is correct before permitting processing to start on a workflow process. Use this mode to check that the organization is correctly set in a multi-organization environment.

- If the function returns the result value TRUE in TEST_CTX mode when the workflow is being processed by a background engine, this result indicates that the application context is set correctly, and the Workflow Engine keeps the current context.
- If the function returns the result value NOTSET in TEST_CTX mode, then the Workflow Engine executes the function again in SET_CTX mode to set the context.
- If the function returns the result value FALSE in TEST_CTX mode, and the Workflow Engine permits context switching at this point in its processing, then the Workflow Engine executes the function again in SET_CTX mode to set the correct context. However, if the result value is FALSE, but the Workflow Engine requires the current context to be preserved, the Workflow Engine defers the activity to be processed by a background engine instead.

Use the SET_CTX mode to set up all your PL/SQL variables. In this way, you can ensure that PL/SQL variables are always set for asynchronous processes when you restart these processes.

For example, you can use this mode to set a bind variable so that specific views are correct, or to set global variables. If you are developing a custom process, you should use this mode to set your organization context.

Note: Oracle Workflow also uses the TEST_CTX mode to determine whether a form can be launched with the current item type context information just before the Notification Details Web page launches a reference form. When you view a notification from the Notification Details Web page and attempt to launch a form that is associated with the notification, Oracle Workflow calls the selector/callback function for your item type in TEST_CTX mode to test the context before turning the form launch over to the Oracle Application Object Library function security system. In TEST_CTX mode, the selector/callback function can perform whatever logic is necessary to determine whether it is appropriate to launch the form.

Summary

Summary

In this lesson, you should have learned how to:

- **Define a selector/callback function for an item type.**
- **Create a selector/callback function, following a standard API.**
- **Call a selector/callback function.**

ORACLE®

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Master/Detail Coordination Activities

Chapter 23

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Master/Detail Coordination Activities

Master/Detail Coordination Activities

ORACLE®

Objectives

Objectives

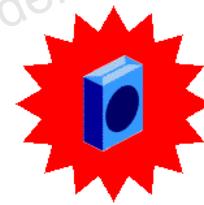
After completing this lesson, you should be able to coordinate master and detail processes using standard activities.

ORACLE

Master/Detail Coordination Activities

Master/Detail Coordination Activities

- Standard activities provided by Oracle Workflow let you coordinate the flow of master and detail processes:
 - Wait for Flow activity: Pauses a process.
 - Continue Flow activity: Signals the halted process to continue.
- Use the WF_ENGINE.SetItemParent API to associate master and detail processes with each other.



ORACLE

Master/Detail Coordination Activities

When you spawn a detail process from a master process, you are creating a separate process with its own unique item type and item key. You define the master/detail relationship between the two processes by calling the Workflow Engine SetItemParent API after you call the CreateProcess API and before you call the StartProcess API when you create the detail process.

Two activities are used to coordinate the flow in the master and detail processes: the Wait for Flow activity and the Continue Flow activity. One activity is placed in the master process; the other is placed in the detail process. Each of the activities contains two attributes used to identify the coordinating activity in the other process.

Example

The following code sample shows an example of launching a set of detail processes from a master process. This API is the PL/SQL procedure for the Spawn Detail Processes function activity in the Workflow Agent Ping/Acknowledge process. The procedure is named WF_EVENT_PING_PKG.Launch_Processes.

```
procedure LAUNCH_PROCESSES (
    ITEMTYPE    in  varchar2,
    ITEMKEY     in  varchar2,
```

```

ACTID      in  number,
FUNCMODE   in  varchar2,
RESULTOUT  out  varchar2
) is
-----
CURSOR c_external_in_agents IS
SELECT
    wfa.name AGENT,
    wfs.name SYSTEM
FROM
    wf_systems wfs,
    wf_agents wfa
WHERE
    wfa.name != 'WF_ERROR'
and  wfa.status = 'ENABLED'
and  wfa.direction = 'IN'
and  wfa.system_guid = wfs.guid
and  upper(wfa.queue_handler) = 'WF_EVENT_QH';

l_eventname varchar2(100);
l_eventkey  varchar2(240);
l_itemkey   varchar2(100);
l_event_t   wf_event_t;
l_msg       varchar2(32000);
l_clob      clob;

begin

if (funcmode = 'RUN') then
    l_eventname := wf_engine.GetActivityAttrText(
                    itemtype => itemtype,
                    itemkey  => itemkey,
                    actid    => actid,
                    aname    => 'EVNTNAME') ;

for x in c_external_in_agents loop
    --
    -- For every agent launch detail process
    --

```

```

l_eventkey := x.agent|| '@' || x.system|| '@' || itemkey;
l_itemkey := l_eventkey;

wf_engine.CreateProcess(
    itemtype => itemtype,
    itemkey   => l_itemkey,
    process    => 'WFDTLPNG') ;

wf_engine.SetItemAttrText(
    itemtype => itemtype,
    itemkey   => l_itemkey,
    fname     => 'EVNTNAME',
    avalue    => l_eventname) ;

wf_engine.SetItemAttrText(
    itemtype => itemtype,
    itemkey   => l_itemkey,
    fname     => 'EVNTKEY',
    avalue    => l_eventkey) ;

wf_engine.SetItemAttrText(
    itemtype => itemtype,
    itemkey   => l_itemkey,
    fname     => 'TOAGENT',
    avalue    => x.agent|| '@' || x.system) ;

-- Initialize the wf_event_t
wf_event_t.initialize(l_event_t);
l_event_t.setcorrelationid(l_itemkey);
l_msg := '<PING>Test Ping</PING>';
dbms_lob.createtemporary(l_clob, FALSE, DBMS_LOB.CALL);
dbms_lob.write(l_clob, length(l_msg), 1, l_msg);
l_event_t.SetEventData(l_clob);

wf_engine.SetItemAttrEvent(
    itemtype => itemtype,
    itemkey   => l_itemkey,
    name      => 'EVNTMSG',
    event     => l_event_t) ;

```

```

wf_engine.SetItemParent (
    itemtype => itemtype,
    itemkey  => l_itemkey,
    parent_itemtype => itemtype,
    parent_itemkey  => itemkey,
    parent_context   => null);

wf_engine.StartProcess (
    itemtype => itemtype,
    itemkey  => l_itemkey);

end loop;

resultout :=
    wf_engine.eng_completed || ':' || wf_engine.eng_null;

return;

elsif (funcmode = 'CANCEL') then
    null;
end if;

exception
    when others then
        WF_CORE.CONTEXT ('WF_EVENT_PING_PKG', 'LAUNCH_PROCESSES',
            ITEMTYPE, ITEMKEY, to_char(ACTID), FUNCMODE);
        raise;
end LAUNCH_PROCESSES;

```

For more information, see: Workflow Agent Ping/Acknowledge, *Oracle Workflow Developer's Guide*.

Wait for Flow Activity

Wait for Flow Activity

- Place the Wait for Flow activity in a master or detail process to pause the flow until the other corresponding detail or master process completes a specified activity.
- The Wait for Flow activity calls the PL/SQL procedure `WF_STANDARD.WAITFORFLOW`.



ORACLE

Wait for Flow Activity

The Wait for Flow activity contains two attributes:

- Continuation Flow: Specify whether this activity is waiting for a corresponding Master or Detail process to complete.
- Continuation Activity: Specify the label of the activity node that must complete in the corresponding process before the current process continues. The default value is `CONTINUEFLOW`.

For more information, see: Predefined Workflow Activities, *Oracle Workflow Developer's Guide*.

Using Multiple Wait for Flow Activities

You can use multiple Wait for Flow activities in the same workflow process. For example, if a master process launches two different sets of detail processes at different stages, the master process should include a separate Wait for Flow activity node for each set of detail processes. In this case, when you call the `WF_ENGINE.SetItemParent` API to associate a detail process with the master process, you must specify the parent context as the activity label name for the Wait for Flow activity node that corresponds to this detail process. (If the parent process

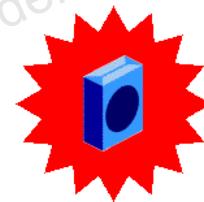
contains only one Wait for Flow activity, however, you can leave the parent context parameter of the WF_ENGINE.SetItemParent API null.)

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Continue Flow Activity

Continue Flow Activity

- Use the Continue Flow activity to mark the position in the corresponding detail or master process where, upon completion, the halted process is to continue.
- The Continue Flow activity calls the PL/SQL procedure `WF_STANDARD.CONTINUEFLOW`.



ORACLE®

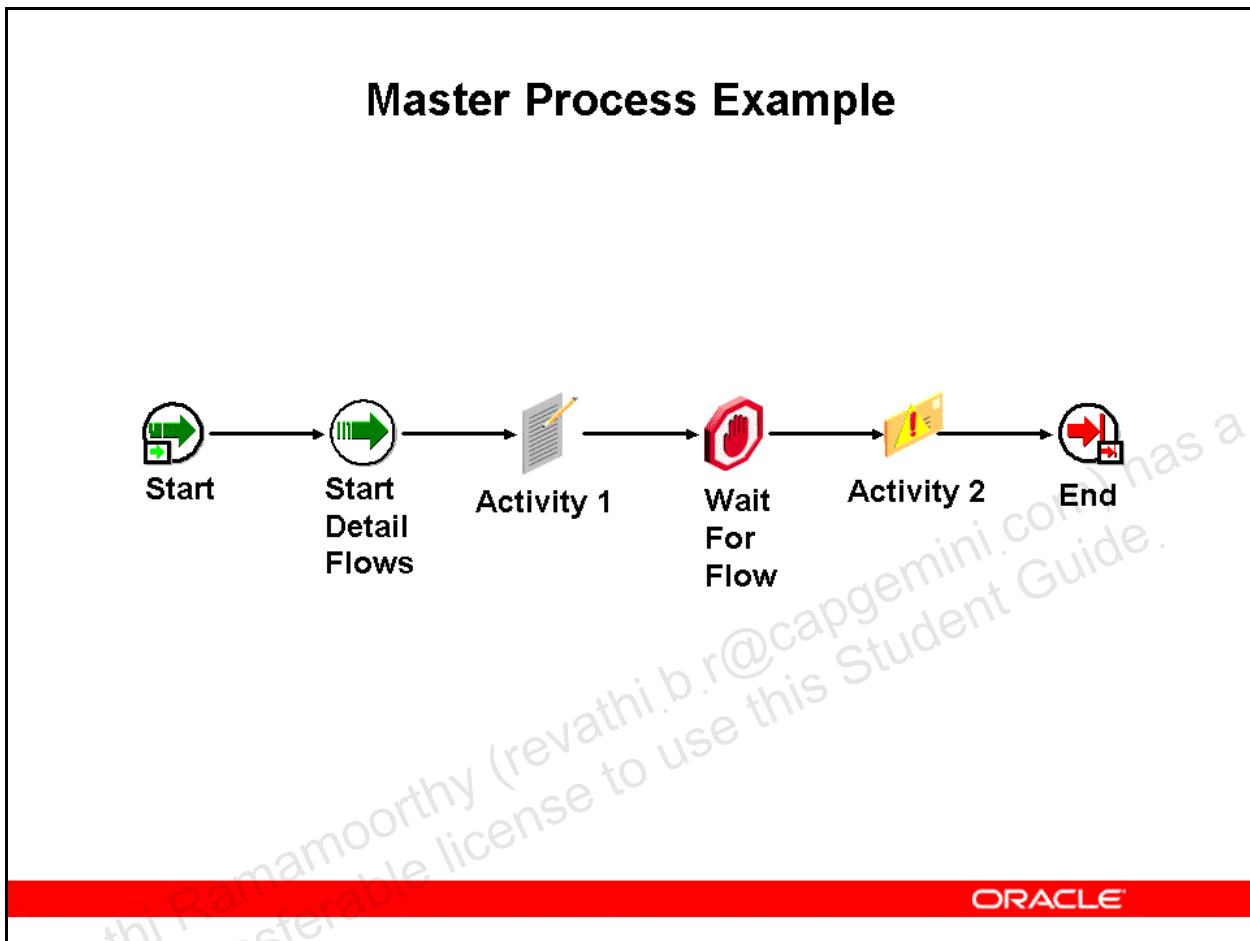
Continue Flow Activity

The Continue Flow activity contains two attributes:

- Waiting Flow: Specify whether the halted process which is waiting for this activity to complete is a Master or Detail flow.
- Waiting Activity: Specify the label of the activity node in the halted process that is waiting for this activity to complete.

For more information, see: Predefined Workflow Activities, *Oracle Workflow Developer's Guide*.

Master Process Example

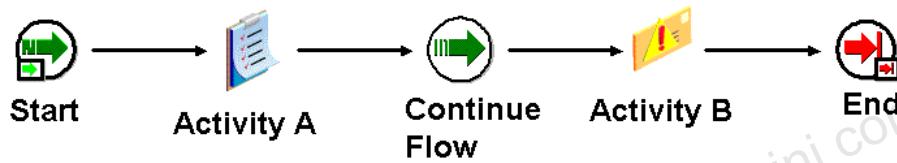


Master Process Example

In the master process above, the Start Detail Flows activity initiates several detail processes. The master process then completes Activity 1 before it pauses at the Wait for Flow activity. The Wait for Flow activity is defined to wait for all its detail processes to complete a Continue Flow activity before allowing the master process to transition to Activity 2.

Detail Process Example

Detail Process Example

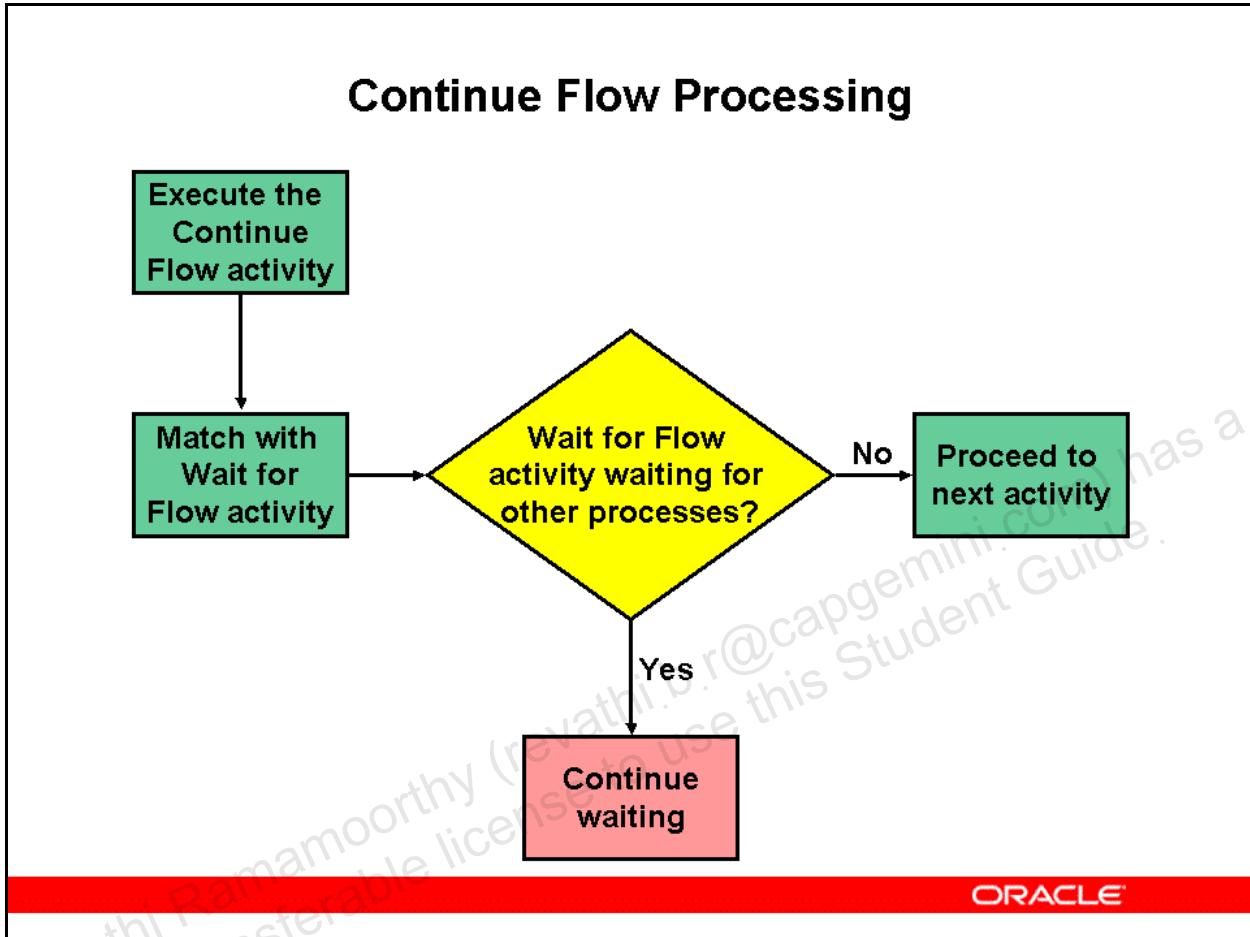


ORACLE

Detail Process Example

When a detail process begins, it completes Activity A. When the process reaches the Continue Flow activity, it signals to the Workflow Engine that the master process can now continue from the Wait for Flow activity. The detail process itself then transitions to Activity B.

Continue Flow Processing



Continue Flow Processing

When a Continue Flow activity is executed, the WF_STANDARD.CONTINUEFLOW procedure checks whether the corresponding Wait for Flow activity is associated with any other processes that have not yet completed their Continue Flow activities. If so, the waiting process continues to wait for those other processes. If the Wait for Flow activity is not waiting for any other processes, then the WF_STANDARD.CONTINUEFLOW procedure completes the Wait for Flow activity so that the process that was waiting now continues to the next activity.

In the master and detail process example, when a detail process executes its Continue Flow activity, it will check whether the Wait for Flow activity in the master process is still waiting for any other detail processes. If all the other detail processes have already completed, then the master process will proceed to Activity 2. If some of the other detail processes have not completed yet, the master process will continue waiting until these processes are complete. For more information, see: Predefined Workflow Activities, *Oracle Workflow Developer's Guide*.

Summary

Summary

In this lesson, you should have learned how to coordinate master and detail processes using standard activities.

ORACLE

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Customizing Workflow Processes

Chapter 24

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Customizing Workflow Processes

Customizing Workflow Processes

ORACLE®

Objectives

Objectives

After completing this lesson, you should be able to do the following:

- **Discuss the guidelines for customizing predefined workflow processes.**
- **Understand the access protection feature of Oracle Workflow.**
- **Describe the types of customizations that are not supported.**
- **Preserve customizations during an upgrade.**

ORACLE®

Customizing Workflow Processes

Customizing Workflow Processes

- You can use Oracle Workflow Builder to easily modify or extend an existing business process without changing its application's code.
- When modifying any predefined workflow process provided by Oracle, you must follow the customization guidelines.
- Develop and test all customizations against a test system first and save your custom definitions from Oracle Workflow Builder to flat files.
- Use the Workflow Definitions Loader to deploy your customizations from the saved flat files to your production system.



ORACLE

Customizing Workflow Processes

Follow these guidelines when customizing workflow processes provided by an Oracle product:

1. Verify that all setup steps have been completed as documented in the *Oracle Workflow Administrator's Guide* and the product-specific user's guides.
2. Test the unmodified seeded workflow, event, or subscription on a test database, and ensure that it runs successfully with the setup and data specific to your environment.
3. Refer to the product-specific user's guide and any documentation updates for the specific workflow, event, or subscription of interest. These documentation sources specifically mention what you should *not* modify. Oracle Support Services will not support modifications to any object that is specifically documented as not modifiable.
4. Gradually build in customizations step-by-step, and test the customized workflow or subscription after each step.
5. When creating PL/SQL procedures, conform to the standard PL/SQL API templates documented in the *Oracle Workflow Developer's Guide*. Be sure to handle exceptions in the event of an error so that you can track down the procedure where the error has occurred.

6. Do not implement the customized workflow, event, or subscription in production without fully ensuring that it works successfully on a test database that is a replica of your production setup. Modifying workflow processes, events, and subscriptions can fundamentally change the behavior of the applications that use them, so you should thoroughly test your changes before deploying them in production.

The following types of customizations are supported:

1. Oracle supports any customization that is stated as required in the seeded workflow's product-specific user's guide or documentation update.
2. Oracle supports customization examples documented in the product-specific user's guide or documentation update. Any issues that arise are fully supported to resolution, to the extent that the customization example was followed as documented. Any deviation from what is documented amounts to a custom development issue that needs further evaluation. See number 3 below.
3. Customizations that are not explicitly stated as unsupported customizations, as required customizations, or as supported customization examples are supported to the extent that the customer must first isolate the problem following the customization guidelines listed above. If upon evaluation, Oracle Support Services deems that the isolated problem stems from an Oracle product, Oracle will supply a solution. Otherwise, it is the responsibility of the customer to correct the custom development issue.

Access Protection

Access Protection

- **Oracle Workflow uses a feature called access protection to control modification of workflow definitions.**
- **Access protection:**
 - Enables “customers” of a workflow definition to modify objects to meet their needs.
 - Prevents “customers” of a workflow definition from modifying “seed data” objects.
 - Preserves legitimate customizations of workflow objects during a workflow definition upgrade.



ORACLE

Access Protection

As a workflow developer, you can use access protection to enable or discourage “customers” of your workflows from modifying your “seed data” workflow definitions.

As a customer of predefined workflows provided by Oracle, you can use access protection to preserve valid customizations that you have made to a predefined workflow during a workflow definition upgrade.

All workflow objects except lookup codes, function attributes, and message attributes contain an Access tab on their property pages. Lookup codes, function attributes, and message attributes inherit their access settings from their parent lookup type, function, or message, respectively.

The Access tab lets you define the following properties:

- Whether future customizations to the object are preserved during a workflow definition upgrade.
- Whether the object can be edited by users operating at a higher access level.

Access Levels

Access Levels

- **Each user of Oracle Workflow Builder operates the system at a certain access level.**
- **The access levels are defined as follows:**
 - **0-9: Reserved for Oracle Workflow**
 - **10-19: Reserved for Oracle Application Object Library**
 - **20-99: Reserved for Oracle E-Business Suite**
 - **100-999: Reserved for customer organizations**
 - **1000: Public**



ORACLE

Access Levels

Protection Levels

- If you protect an object against customization, you effectively assign the object a protection level equal to your current access level.
- Objects protected against customizations are considered “seed data.”
- Only users operating at an access level equal to or lower than the protection level of the object can modify the object.
- Users operating at an access level greater than the protection level of the object will see a small lock on the icon for the object in the navigator tree, indicating that the object is read-only.

Customization Levels

- If you set an object to be customizable, its protection level is set to 1000.
- The customization level of an object is set to the access level of the initial user who customizes the object.
- A customized object is locked from further modification except from users with access levels equal to the customization level of the object.

- The customization level is relevant only with respect to unprotected workflow objects.
- If an object is protected at a certain level, it should not be modified at all except by an access level equal to or less than the protected level of the object.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Setting the Access Level

Setting the Access Level

- **The access level defaults to 100 when Oracle Workflow Builder is installed.**
- **Oracle E-Business Suite customers should always operate the Oracle Workflow Builder at an access level of 100.**



ORACLE

Setting Access Levels

You can view your access level in the About Oracle Workflow Builder dialog box available from the Help menu.

The Allow Modifications of Customized Objects check box also appears in the About Oracle Workflow Builder dialog box. Usually you do not need to change this setting.

- Deselected: The Workflow Builder saves edits only to protected objects that you have access to change and does not overwrite previously customized objects (equivalent to Workflow Definitions Loader Upgrade mode). This option is recommended.
- Selected: The Workflow Builder saves your edits, overwriting protected objects that you have access to modify, as well as any previously customized objects (equivalent to Workflow Definitions Loader Upload mode).

Setting Protection and Customization Levels

Setting Protection and Customization Levels

Select the Access tab for an object in Oracle Workflow Builder to display the Access property page.

- Use the indicator bar to view the range of access levels that can edit the object.
- Use the Levels region to view the customization, access, and protection levels of the object based on the settings in the Options region.
- Use the Options region to set the protection and customization levels of an object.
 - Preserve Customizations
 - Lock at this Access Level



ORACLE

Setting Protection and Customization Levels

- The indicator bar provides a visual range of access levels that can edit the object:
 - Black vertical line: Current access level.
 - White range: Cannot edit the object.
 - Solid green: Can edit the object.
 - Cross-hatch green: Usually cannot modify the object because it has been customized, but can now do so because Oracle Workflow Builder is set to Allow Modifications of Customized Objects mode so that you can modify customized objects.
- The Levels region shows the Customization, Access, and Protection levels of the object based on how you set the check boxes in the Options region.
- Use the Options region to set the protection and customization levels of an object:
 - Preserve Customizations: Prevents you from overwriting customized objects during a workflow definition upgrade.
 - Lock at this Access Level: Protects the object at the current access level and does not allow higher access levels to customize the object.

- Oracle E-Business Suite customers should select both the Preserve Customizations and Lock at this Access Level check boxes to protect your workflow objects during upgrades.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Example of Access Protection

Example of Access Protection		
The example assumes an access level of 100.		
Selected Options	Resulting Level	Edit Range
None	Customization = 0 Access = 100 Protection = 1000	0-1000
Preserve Customizations	Customization = 100 Access = 100 Protection = 1000	100-1000
Lock at this Access Level	Customization = 0 Access = 100 Protection = 100	0-100
Both	Customization = 100 Access = 100 Protection = 100	100

ORACLE®

Protection and Customization Levels

Assuming an access level of 100, these protection and customization levels result when the following check boxes are selected in the Access properties region:

- None: Any access level can update an object at any time.
- Preserve Customizations: Prohibits anyone from overwriting customized objects during a workflow upgrade.
 - Access levels 100-1000 can update an object.
 - If the Allow modifications of customized objects check box is selected, access levels 0-99 (as represented by green crosshatches in the indicator bar) can also update customized objects.
- Lock at this Access Level: Protects the object at the current access level and allows only access levels 0-100 to customize the object.
- Both: Only the access level at which the object is protected can update the object.
 - Only access level 100 can update the object.

- If the Allow Modifications of Customized Objects check box is selected, access levels 0-99 (as represented by green crosshatches in the indicator bar) can update customized objects.

Unsupported Customizations

Unsupported Customizations

The following types of customizations are *not supported*:

- **Modifying a workflow object that has a protection level less than 100.**
- **Altering a workflow object's protection level if its original protection level is less than 100.**
- **Changing your access level to an unauthorized level of less than 100 for the purpose of modifying workflow objects that are protected at levels less than 100.**



Unsupported Customizations

Oracle does *not support* the following types of customizations:

- Oracle does not support customizations that are explicitly documented as being unsupported in the seeded workflow's product-specific user's guide or documentation update notes. These customizations include modifying processes, attributes, function activities, event activities, notifications, lookup types, messages, events, or subscriptions that are specifically documented as not to be modified.
- Oracle does not support manual modifications of Workflow tables with a prefix of WF_ or FND_, unless these modifications are documented in the Oracle Workflow guides or required by Oracle Support Services.

Preserving Customizations

Preserving Customizations

To ensure that your customizations are preserved during an upgrade of Oracle Workflow:

- **Check that your access level is set to 100 before you make your modifications to the predefined workflow process.**
- **Select both the Preserve Customizations and Lock at this Access Level options in the Access property page for any object that you modify.**



ORACLE®

Preserving Customizations

During an Oracle Workflow seed data upgrade, the Workflow Definitions Loader is always run in Upgrade mode at an access level less than 100. As a result, the upgrade will not overwrite any object with a customization level of 100 or higher.

Summary

Summary

In this lesson, you should have learned how to:

- **Discuss the guidelines for customizing predefined workflow processes.**
- **Understand the access protection feature of Oracle Workflow.**
- **Describe the types of customizations that are not supported.**
- **Preserve customizations during an upgrade.**

ORACLE

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Workflow Loaders

Chapter 25

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Workflow Loaders

Workflow Loaders

ORACLE®

Objectives

Objectives

After completing this lesson, you should be able to do the following:

- **Use the Workflow Definitions Loader to transfer workflow process definitions between a database and flat file.**
- **Use the Workflow XML Loader to transfer XML definitions for Business Event System objects between a database and a flat file.**

ORACLE®

Workflow Definitions Loader

Workflow Definitions Loader

- Use the Workflow Definitions Loader program to transfer workflow definitions between a flat file and a database.
 - Upload customizations from a saved flat file to a production environment.
 - Download and back up your process definitions to a flat file prior to a database upgrade.
 - Upload the definitions back into your database when the database upgrade is complete.
- Workflow definition files are identified by the file extension .wft.



Workflow Definitions Loader

Oracle Workflow also uses the Workflow Definitions Loader internally to perform seed data upgrades.

Workflow Definitions Loader

Workflow Definitions Loader

Run the Workflow Definitions Loader concurrent program using one of the following modes:

- **Upgrade**
- **Upload**
- **Force**
- **Download**



ORACLE

Workflow Definitions Loader

Specify one of the following modes:

- Upgrade: Upgrade to a definition in an input file, preserving customizations, using the access level in the input file. This mode is recommended.
- Upload: Upload the definition from an input file, ignoring preserve customizations settings and using the access level in the input file. This mode is useful to someone who is developing a workflow process on a test system. It enables the developer to save definitions to the database without concern that accidental customizations to existing objects might prevent the upload of some process definition elements.
- Force: Force the upload of a definition from an input file to a database regardless of an object's protection level. This mode is useful for fixing data integrity problems in a database with a known, reliable file backup.
- Download: Download the specified item type definition from the database to an output file.

Workflow XML Loader

Workflow XML Loader

- Use the Workflow XML Loader program to transfer XML definitions for Business Event System objects between a flat file and a database.
- Workflow object XML definition files are identified by the file extension .wfx.



Workflow XML Loader

When you download Business Event System object definitions from a database, Oracle Workflow saves the definitions as an XML file.

When you upload object definitions to a database, Oracle Workflow loads the definitions from the source XML file into the Business Event System tables in the database, creating new definitions or updating existing definitions as necessary.

Workflow XML Loader

Workflow XML Loader

To run the Workflow XML Loader manually, run JRE against `oracle.apps.fnd.wf.WFXLoad`.



Workflow XML Loader

To run the Workflow XML Loader, you must have Java Development Kit (JDK) installed in a version supported by Oracle E-Business Suite. See: *Overview of Using Java with Oracle E-Business Suite Release 12*, My Oracle Support Knowledge Document 418664.1.

For detailed information on the commands to run the Workflow XML Loader, see: *Setting Up Oracle Workflow, Oracle Workflow Administrator's Guide*.

Workflow XML Loader

Workflow XML Loader

When you run the Workflow XML Loader, specify the mode that you want:

- **-d: Normal download mode**
- **-de: Exact download mode**
- **-u: Normal upload mode**
- **-uf: Force upload mode**



ORACLE

Workflow XML Loader

You can download Business Event System object definitions in either normal download mode or exact download mode:

- Normal download mode: Saves a generic copy of object definitions from one system that you can use to create similar definitions in other systems. In normal download mode, the Workflow XML Loader replaces certain system-specific data within the object definitions with tokens. Select normal download mode, for example, when you want to save Business Event System object definitions from a development system as seed data that you can upload to a production system.
- Exact download mode: Saves object definitions exactly as they are specified in the database. In exact download mode, the Workflow XML Loader does not convert any data to tokens; instead, all values, including system-specific values, are copied to the XML file. Select exact download mode, for example, when you want to save Business Event System object definitions from one production system so that you can replicate them to another production system that communicates with the first.

You can upload Business Event System object definitions in either normal upload mode or force upload mode:

- Normal upload mode: Loads the object definitions from the source XML file into the Business Event System tables in the database, but does not update any event or subscription definition with a customization level of User. Oracle Workflow uses this mode to preserve your customizations during upgrades.
- Force upload mode: Loads the object definitions from the source XML file into the Business Event System tables in the database and overwrites any existing definitions, even for events or subscriptions with a customization level of User. Use this mode when you want to update the definitions for your own custom events and subscriptions.

Note: To report more extensive debugging information in the program output, you can run the Workflow XML Loader in debug mode by including the DebugMode argument just before the -d, -de, -u, or -uf option.

Summary

Summary

In this lesson, you should have learned how to:

- **Use the Workflow Definitions Loader to transfer workflow process definitions between a database and flat file.**
- **Use the Workflow XML Loader to transfer XML definitions for Business Event System objects between a database and a flat file.**

ORACLE®

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Specialized Workflow Monitoring

Chapter 26

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Specialized Workflow Monitoring

Specialized Workflow Monitoring

ORACLE®

Objectives

Objectives

After completing this lesson, you should be able to do the following:

- **Grant restricted access to workflow monitoring data.**
- **Grant permissions for administrative actions within the Status Monitor.**

ORACLE®

Assigning Specialized Workflow Monitoring Privileges

Assigning Specialized Workflow Monitoring Privileges

- You can designate certain users as administrators only for particular types of workflows by assigning those users specialized workflow monitoring privileges with restricted access to workflow data.
- You can base the restrictions on a defined set of item types or on criteria specific to a particular functional area.



ORACLE®

Assigning Specialized Workflow Monitoring Privileges

Assigning Specialized Workflow Monitoring Privileges

- **Users who act as specialized workflow administrators must have access to the administrator version of the Status Monitor in one of their responsibilities.**
- **You must also grant permissions to specialized workflow administrators to enable them to perform administrative actions within the Status Monitor for workflows to which they have access.**



ORACLE

Assigning Specialized Workflow Monitoring Privileges

Users cannot perform any actions on workflows that they own themselves, irrespective of any permissions granted to them. Only users with full workflow administrator privileges assigned in the Workflow Configuration page can perform administrative actions on workflows that they own themselves.

If a user has full workflow administrator privileges assigned on the Workflow Configuration page, then those privileges override any specialized workflow monitoring privileges assigned to that user. That is, a user with full workflow administrator privileges can access all workflows, irrespective of any restrictions defined for any specialized privileges.

Granting Restricted Access to Workflow Monitoring Data

Granting Restricted Access to Workflow Monitoring Data

You can restrict access to workflow monitoring data based on the following:

- Item types only
- Functional criteria only
- Both item types and functional criteria



ORACLE

Granting Restricted Access to Workflow Monitoring Data

Grants based on functional criteria depend on item attribute values. Consequently, these grants are most effective when combined with grants for item types that share the same item attributes.

Granting Restricted Access Based on Item Types

Granting Restricted Access Based on Item Types

- **Create an instance set on the object WORKFLOW_ITEMS with a predicate that includes the parameters in which you will specify the item types that you want.**
- **Create a grant using the instance set that you created.**



Granting Restricted Access Based on Item Types

The following excerpt shows a sample predicate for granting access to a single item type.

```
&TABLE_ALIAS.ITEM_TYPE = &GRANT_ALIAS.PARAMETER1
```

You can also grant access to multiple item types, up to a maximum of ten, which is the maximum number of parameters you can specify for a grant. For example, the following excerpt shows a sample predicate for granting access to ten item types:

```
&TABLE_ALIAS.ITEM_TYPE = &GRANT_ALIAS.PARAMETER1 OR  
&TABLE_ALIAS.ITEM_TYPE = &GRANT_ALIAS.PARAMETER2 OR  
&TABLE_ALIAS.ITEM_TYPE = &GRANT_ALIAS.PARAMETER3 OR  
&TABLE_ALIAS.ITEM_TYPE = &GRANT_ALIAS.PARAMETER4 OR  
&TABLE_ALIAS.ITEM_TYPE = &GRANT_ALIAS.PARAMETER5 OR  
&TABLE_ALIAS.ITEM_TYPE = &GRANT_ALIAS.PARAMETER6 OR  
&TABLE_ALIAS.ITEM_TYPE = &GRANT_ALIAS.PARAMETER7 OR  
&TABLE_ALIAS.ITEM_TYPE = &GRANT_ALIAS.PARAMETER8 OR  
&TABLE_ALIAS.ITEM_TYPE = &GRANT_ALIAS.PARAMETER9 OR  
&TABLE_ALIAS.ITEM_TYPE = &GRANT_ALIAS.PARAMETER10
```

The following excerpt shows another alternative for a sample predicate for granting access to ten item types:

```
&TABLE_ALIAS.ITEM_TYPE in (&GRANT_ALIAS.PARAMETER1,
&GRANT_ALIAS.PARAMETER2, &GRANT_ALIAS.PARAMETER3,
&GRANT_ALIAS.PARAMETER4, &GRANT_ALIAS.PARAMETER5
&GRANT_ALIAS.PARAMETER6, &GRANT_ALIAS.PARAMETER7,
&GRANT_ALIAS.PARAMETER8, &GRANT_ALIAS.PARAMETER9,
&GRANT_ALIAS.PARAMETER10)
```

To create a grant, first specify appropriate security context information such as grantee and responsibility. Then specify the following data context information:

- Object: WORKFLOW_ITEMS
- Data Context Type: Instance Set
- Instance Set: The instance set that you created on WORKFLOW_ITEMS
- Parameter 1 through Parameter 10: The internal names of the item types to which you want to grant access, such as WFDEMO. You can specify one item type name in each parameter that is referenced in the predicate of your instance set, up to the maximum of ten.
- Set: Business workflow item permission set

For more information about creating instance sets and grants, see: Defining Data Security Policies, *Oracle Applications System Administrator's Guide - Security* and Assigning Permissions to Roles, *Oracle Applications System Administrator's Guide - Security*.

Granting Restricted Access Based on Functional Criteria

Granting Restricted Access Based on Functional Criteria

- Create an instance set on the object **WORKFLOW_ITEM_ATTR_VALUES** with a predicate that defines criteria specific to a particular functional area using item attributes.
- Create a grant using the instance set that you created.



Granting Restricted Access Based on Functional Criteria

The following excerpt shows a sample predicate for an instance set on the object **WORKFLOW_ITEM_ATTR_VALUES**, defining criteria for HR data using the **CURRENT_PERSON_ID** item attribute.

```
&TABLE_ALIAS.NAME = 'CURRENT_PERSON_ID'  
and EXISTS (SELECT 'Y' FROM per_people_f  
WHERE person_id = &TABLE_ALIAS.TEXT_VALUE  
AND TRUNC (SYSDATE) BETWEEN effective_start_date AND  
effective_end_date)
```

As another example, if the workflows for a particular organization are marked with an item attribute named **ORG_ID**, the following excerpt shows a sample predicate that allows access only to workflows associated with the user's current organization context.

```
&TABLE_ALIAS.NAME = 'ORG_ID'  
and &TABLE_ALIAS.TEXT_VALUE = substr  
(sys_context ('USERENV', 'CLIENT_INFO') ,1,10)
```

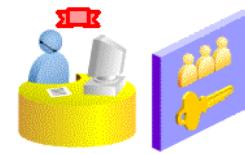
To create a grant, first specify appropriate security context information such as grantee and responsibility. Then specify the following data context information:

- Object: WORKFLOW_ITEM_ATTR_VALUES
- Data Context Type: Instance Set
- Instance Set: The instance set that you created on WORKFLOW_ITEM_ATTR_VALUES
- Permission Set: Business workflow item attribute permission set

Granting Permissions for Administrative Actions

Granting Permissions for Administrative Actions

- If you want to assign a user privileges for all administrative actions within the Status Monitor, assign that user the role WF_ADMIN_ROLE.
- If you want to assign a user privileges only for specific administrative actions, create a custom permission set with the permissions you want to assign, and grant that permission set to the user.



ORACLE

Granting Permissions for Administrative Actions

The role WF_ADMIN_ROLE by default is granted the seeded permission set "Business workflow item permission set" (WF_ADMIN_PSET), which includes the permissions for all the administrative actions within the Status Monitor.

You can include the following permissions for administrative actions in a custom permission set:

- Skip Workflow Activity (WF_SKIP)
- Retry Activity (WF_RETRY)
- Rewind Workflow (WF_REWIND)
- Suspend Workflow (WF_SUSPEND)
- Cancel Workflow (WF_CANCEL)
- Update Workflow Item Attributes (WF_UPDATE_ATTR)
- Monitor Data (WF_MON_DATA)

See: Assigning Permissions to Roles, *Oracle Applications System Administrator's Guide - Security*.

Summary

Summary

In this lesson, you should have learned how to:

- **Grant restricted access to workflow monitoring data.**
- **Grant permissions for administrative actions within the Status Monitor.**

ORACLE®

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Setting Up Oracle Workflow

Chapter 27

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Setting Up Oracle Workflow

Setting Up Oracle Workflow

ORACLE®

Objectives

Objectives

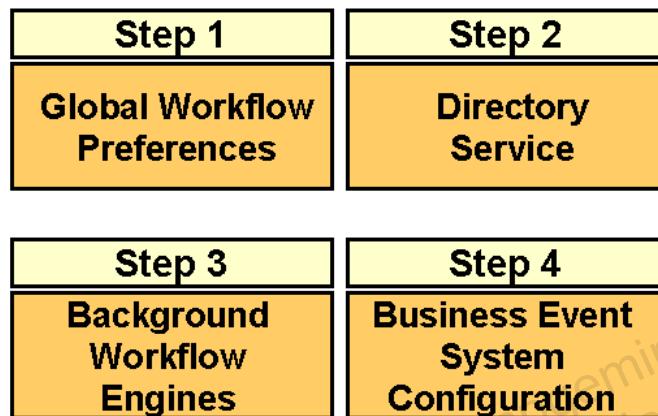
After completing this lesson, you should be able to do the following:

- **Describe the required setup steps for Oracle Workflow.**
- **Describe the optional setup steps for Oracle Workflow.**

ORACLE®

Required Setup Steps

Required Setup Steps



ORACLE®

Required Setup Steps

1. Set system-wide preferences and default user preferences for your installation of Oracle Workflow using the Workflow Configuration page.
2. Ensure that a directory service is set up to provide information about the individuals and roles in your organization who may utilize Oracle Workflow functionality and receive workflow notifications.
3. Set up background Workflow engines to manage the load on the primary Workflow Engine by processing deferred and timed out activities and stuck processes.
4. Configure the Business Event System for event communication.

For more information, see: Setting Up Oracle Workflow, *Oracle Workflow Administrator's Guide*.

Step 1: Setting Global Workflow Preferences

Step 1: Setting Global Workflow Preferences

- Set your preferences on the Workflow Configuration page.
 - Define the workflow administrator role.
 - Set system-wide preference values.
 - Set default user preference values.
- Individual user preferences set by users override the defaults set on the Workflow Configuration page.

ORACLE

Step1: Setting Global Workflow Preferences

Use a Web browser to navigate to the Workflow Configuration page, using a responsibility and navigation path specified by your system administrator. Some possible navigation paths in the seeded Workflow responsibilities are:

- Workflow Administrator Web Applications: Administration > Workflow Preferences
- Workflow Administrator Web (New): Administration > Workflow Preferences
- Workflow Administrator Event Manager: Administration > Workflow Preferences

You can also navigate to the Workflow Configuration page from other Oracle Workflow administrator Web pages by choosing the Administration tab or selecting the Administration link at the end of the page.

You must have workflow administrator privileges to set global workflow preferences on the Workflow Configuration page. If you do not have administrator privileges, you can view global workflow preferences, but you cannot modify them.

Workflow System Administrator

Select the role to which you want to assign workflow administrator privileges. If you want all users and roles to have workflow administrator privileges, such as in a development environment, enter an asterisk (*).

LDAP

If you are integrating with Oracle Internet Directory, specify the Lightweight Directory Access Protocol (LDAP) server information for the LDAP directory to which you will connect. If you already configured these parameters while installing Oracle Application Server with Oracle E-Business Suite, Oracle Workflow displays those values here. For more information, see: *Installing Oracle Application Server 10g with Oracle E-Business Suite Release 12 (OracleMetaLink note 376811.1)* and *Oracle Single Sign-On Integration, Oracle Applications System Administrator's Guide - Security*.

- Host: The host on which the LDAP directory resides.
- Port: The port on the host.
- Username: The LDAP user account used to connect to the LDAP server. This user name must have write privileges and is required to bind to the LDAP directory.
- Old Password: Enter your current LDAP password. Oracle Workflow validates this password before letting you change it.
- New Password: Enter the new LDAP password you want to use. The password must be at least five characters long.
- Repeat Password: Enter your new LDAP password again in this field to confirm it.
- Change Log Base Directory: The LDAP node under which change logs are located.
- User Base Directory: The LDAP node under which user records can be found.

Business Event Local System

Specify the system name for the database where this installation of Oracle Workflow is located, to identify it in the Business Event System. Oracle Workflow automatically creates the system definition for this database in the Event Manager during installation.

Select the execution status for the local system.

- Enabled: Subscriptions are executed on all events. Oracle Workflow sets the system status to Enabled by default.
- Local Only: Subscriptions are executed only on events raised on the local system.
- External Only: Subscriptions are executed only on events received by inbound agents on the local system.
- Disabled: No subscriptions are executed on any events.

Notification Style

Specify whether Oracle Workflow should send e-mail notifications to users, and if so, in what format. A user can override this default setting by specifying a different notification style in his or her individual Oracle E-Business Suite preferences.

- HTML mail with attachments
- Plain text mail with HTML attachments
- Plain text mail
- Plain text summary mail
- Do not send me mail

- HTML mail
- HTML summary mail

Browser Signing DLL Location

Specify the location of the Capicom.dll file that is used for Web page operations with encryption in the Microsoft Internet Explorer browser. This preference is required only if you plan to use certificate-based digital signatures to confirm notification responses, and your users access Oracle E-Business Suite with Microsoft Internet Explorer. By default, this preference is set to a URL at which the Capicom.dll file can be downloaded from Microsoft's Web site. In most cases, you do not need to change this setting. However, you can update this preference if the location of the Capicom.dll file changes, or if you choose to store a copy of the file on your local network and point to that location instead.

JInitiator

Review details about the JInitiator plugin in your Oracle E-Business Suite installation. Oracle Workflow uses JInitiator to launch Oracle E-Business Suite forms linked to notifications.

- Class ID: The class identifier for this version of JInitiator.
- Download Location: The location where the JInitiator executable is staged for download to users' client machines.
- Version: The JInitiator version number.

Step 2: Setting Up an Oracle Workflow Directory Service

- ### Step 2: Setting Up an Oracle Workflow Directory Service
- Oracle Workflow requires a directory service to provide information about the individuals and roles in your organization who may use Oracle Workflow functionality and receive workflow notifications.
 - A predefined directory service for users and roles from the unified Oracle E-Business Suite environment is automatically implemented for you during installation.

ORACLE®

Step 2: Setting Up an Oracle Workflow Directory Service

In the predefined directory service implementation, Oracle Workflow automatically creates directory service views to integrate with the appropriate source tables.

You can also create your own directory service by defining custom views with the required columns. However, note that only the predefined directory service provided by Oracle Workflow is supported by Oracle.

Step 2: Setting Up an Oracle Workflow Directory Service

- ### Step 2: Setting Up an Oracle Workflow Directory Service
- Oracle Workflow uses a directory service model in which denormalized information is maintained in the Workflow local tables for performance gain.
 - You should maintain synchronization between the user and role information stored in application tables by the source modules and the information stored in the Workflow local tables.

ORACLE®

Step 2: Setting Up an Oracle Workflow Directory Service

Step 2: Setting Up an Oracle Workflow Directory Service

- **Each Oracle E-Business Suite module that stores user and role information in its application tables automatically synchronizes that information with the information in the Workflow local tables on an incremental basis, using the Workflow local synchronization APIs.**
- **For troubleshooting or diagnostic purposes, you can run a concurrent program named Synchronize WF LOCAL Tables or a request set named Synchronize Workflow LOCAL Tables to perform synchronization in bulk.**
- **The concurrent program periodically refreshes the information in the Workflow local tables for the specified modules.**

ORACLE

Step 2: Setting Up an Oracle Workflow Directory Service

You only need to run the bulk synchronization program for products that are not successfully performing incremental synchronization.

The Synchronize Workflow LOCAL Tables request set contains ten instances of the Synchronize WF Local Tables program, one for each originating system. You can use this request set to submit requests for all the originating systems at once.

Use the Submit Requests form to submit the request set or concurrent program. Enter the following parameters:

- Orig System (single concurrent program only): Select the name of the originating system whose user and role information you want to synchronize with the WF_LOCAL tables.
- Parallel Processes: Enter the number of parallel processes to run. The default value for this parameter is 1. However, if your hardware resources allow, you can optionally set this parameter to a higher value in order to parallelize the queries during execution of the program.
- Logging: Select the logging mode you want. This mode determines whether redo log data is generated for database operations performed by the bulk synchronization process. The default value for this parameter is LOGGING, which generates redo log data normally.

You can optionally set the logging mode to NOLOGGING to suppress redo log data, obtaining a performance gain. Without this redo log data, however, no media recovery is possible for the Workflow directory tables and indexes.

- Temporary Tablespace: Select the temporary tablespace the program should use.
- Raise Errors: Select Yes or No to indicate whether the program should raise an exception if it encounters an error. Usually you can leave this parameter set to the default value, which is Yes.

Note: Products that use role hierarchies do not participate in bulk synchronization. These products must perform incremental synchronization.

Step 3: Running Background Engines

Step 3: Running Background Engines

- Set up background engines to handle:
 - Costly deferred activities, to allow the Workflow Engine to continue to the next available activity
 - Timed out notifications
 - Stuck processes
- Set up background engines to run periodically by scheduling the Workflow Background Process concurrent program to resubmit periodically.

ORACLE®

Step 3: Running Background Engines

Ensure that you have at least one background engine that can check for timed out activities, one that can process deferred activities, and one that can handle stuck processes. At a minimum, you need to set up one background engine that can handle all three types of issues. However, for performance reasons we recommend that you run three separate background engines at different intervals.

- Run a background engine to handle only deferred activities every 5 to 60 minutes.
- Run a background engine to handle only timed out activities every 1 to 24 hours as needed.
- Run a background engine to handle only stuck processes once a day to once a month, when the load on the system is low.

Step 3: Running Background Engines

Step 3: Running Background Engines

- **Use Oracle Workflow Manager in Oracle Applications Manager to run the Workflow Background Process concurrent program.**
- **For testing purposes, you can also run the WF_ENGINE.Background API manually through SQL*Plus.**

ORACLE®

Step 3: Running Background Engines

Step 3: Running Background Engines

The Background Engine API is as follows:

```
WF_ENGINE.BACKGROUND (
    itemtype          in varchar2,
    minthreshold     in number   default null,
    maxthreshold     in number   default null,
    process_deferred in boolean  default TRUE,
    process_timeout  in boolean  default FALSE,
    process_stuck    in boolean  default FALSE);
```

ORACLE

Step 3: Running Background Engines

The Background Engine API parameters are:

- itemtype: Optional item type to restrict this engine to activities associated with that item type.
- minthreshold: Optional minimum cost that an activity must have for this background engine to execute it, in hundredths of a second.
- maxthreshold: Optional maximum cost an activity can have for this background engine to execute it, in hundredths of a second.
- process_deferred: Specify TRUE or FALSE to indicate whether the engine should check for deferred activities.
- process_timeout: Specify TRUE or FALSE to indicate whether the engine should check for timed out activities.
- process_stuck: Specify TRUE or FALSE to indicate whether the engine should check for stuck processes.

Step 4: Configuring the Business Event System

Step 4: Configuring the Business Event System

To communicate business events between systems:

- Set up database links and queues.
- Check database initialization parameters.
- Schedule listeners for local inbound agents.
- Schedule propagation for local outbound agents.
- Synchronize event and subscription license statuses with product license statuses.
- Periodically clean up the WF_CONTROL queue.

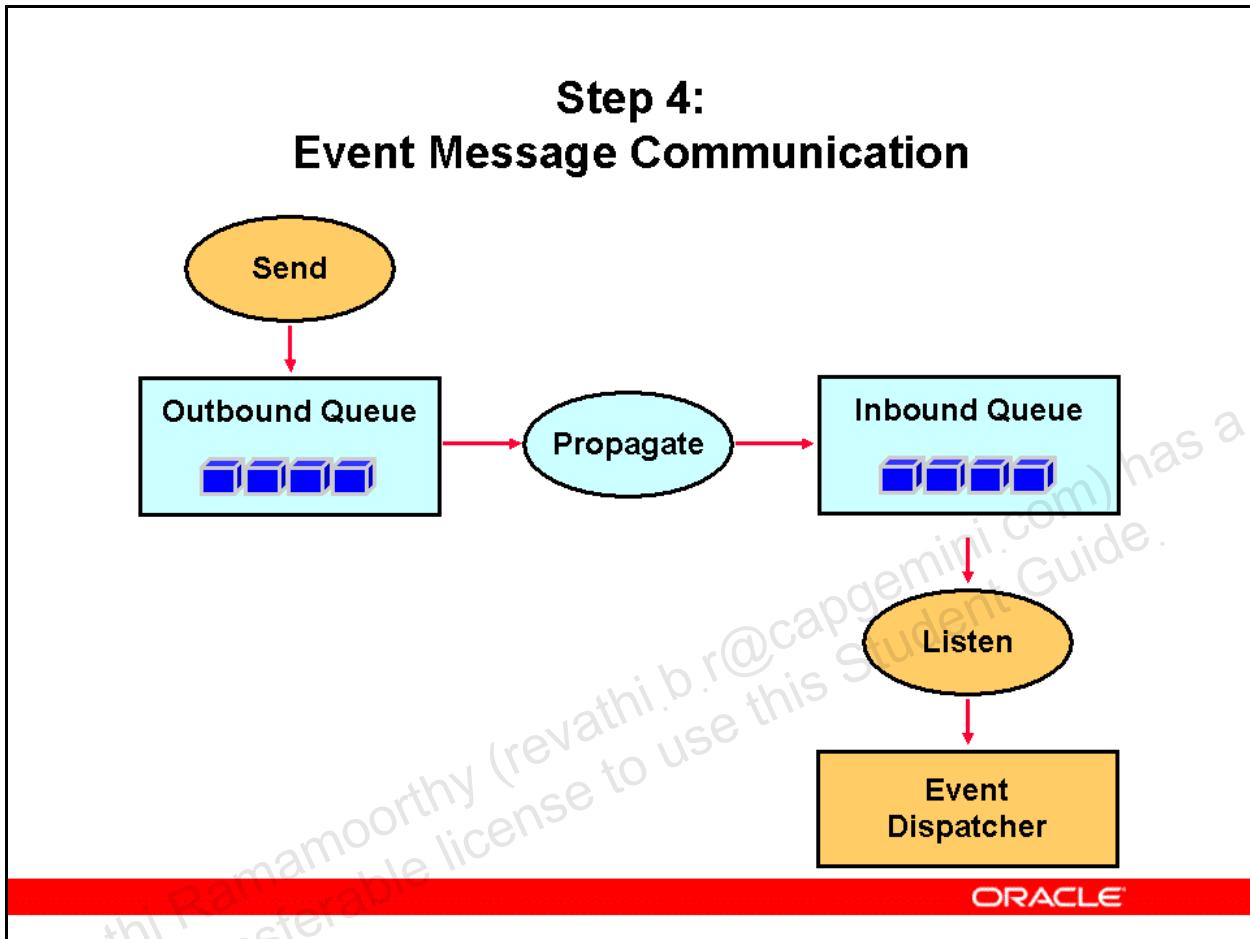
ORACLE®

Step 4: Configuring the Business Event System

You only need to set up database links and schedule propagation to external systems if you want to communicate event messages between those systems and your local system.

For more information about additional optional configuration steps, see: Setting Up the Business Event System, *Oracle Workflow Administrator's Guide*.

Step 4: Event Message Communication



Step 4: Event Message Communication

To send an event message, Oracle Workflow places the event message on a local outbound agent's queue. You must schedule propagation to deliver the message from there to the designated inbound agent's queue.

To receive an event message in Oracle Workflow, you must run an agent listener to dequeue the message from the inbound agent's queue for Oracle Workflow to process. A component of the Event Manager called the Event Dispatcher then searches for and executes subscriptions to the event.

Step 4: Setting Up Database Links and Queues

Step 4: Setting Up Database Links and Queues

- If you want to communicate business events between the local system and external systems using the SQLNET protocol, create database links to those external systems.
- If you want to use custom queues for propagating events, set up your queues.
- You can either create database links and set up queues manually, or you can use Oracle Enterprise Manager.

ORACLE

Step 4: Setting Up Database Links and Queues

Database Links

When you set up database links for use by the Business Event System, you should fully qualify each database link name with the domain name.

Note: Database links let systems communicate with each other through the SQLNET protocol. For communication using other protocols such as HTTP or HTTPS, or for communication between Oracle and non-Oracle systems, you should provide connections as appropriate. For more information, refer to the *Oracle Streams Advanced Queuing User's Guide and Reference*.

Queues

In addition to the standard queues provided by Oracle Workflow, you can also set up your own queues for event message propagation. To set up a queue, you must create the queue table, create the queue, and start the queue.

Oracle Workflow provides a sample script called wfevquc2.sql, which you can modify to set up your queues, as well as a sample script called wfevqued.sql, which you can modify to drop queues. These scripts are located on your server in the \$FND_TOP/sql directory.

If you define a queue with a payload type other than the standard WF_EVENT_T Workflow format or the JMS Text message format, you must create a queue handler to translate between the standard Workflow format and the format required by the queue.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Step 4: Checking Database Parameters

Step 4: Checking Database Parameters

- Use Oracle Workflow Manager to verify the **JOB_QUEUE_PROCESSES** database initialization parameter.
- Oracle Workflow requires job queue processes to handle propagation of Business Event System event messages by Oracle Advanced Queuing queues.

ORACLE®

Step 4: Checking Database Parameters

The **JOB_QUEUE_PROCESSES** parameter defines the number of job queue processes for your instance. You must start at least one job queue process to enable message propagation. The minimum recommended number of processes for Oracle Workflow is ten.

You can either modify the parameter in the init.ora file and restart the database, or you can use the ALTER SYSTEM statement to dynamically modify the parameter for the duration of the instance.

Note: If you want to review more detailed information about Oracle Advanced Queuing processing, you can optionally use another initialization parameter, EVENT, for detailed database level tracing of issues related to Oracle Advanced Queuing. Add the following line to your init.ora file:

```
event = "24040 trace name context forever, level 10"
```

Then restart your database to make this change effective. Be aware that using this parameter may generate large trace files.

Step 4: Scheduling Agent Listeners

Step 4: Scheduling Agent Listeners

- An agent listener checks an agent for inbound event messages and dequeues messages for the Event Manager to process.
 - PL/SQL agent listeners process event subscriptions in the database.
 - Java agent listeners process event subscriptions in the middle tier.
- The PL/SQL and Java agent listener programs are defined as service component types in the Generic Service Component Framework.
- Use Oracle Workflow Manager to submit and manage agent listener service components.

ORACLE®

Step 4: Scheduling Agent Listeners

The Generic Service Component Framework is a facility that helps to simplify and automate the management of background Java services.

Agent listener service components, managed through Oracle Workflow Manager, are the most automated and robust way to run agent listeners. However, Oracle Workflow also provides an administrative script named wfagtlst.sql that you can use to run an agent listener, or you can run the WF_EVENT.Listen API directly in SQL*Plus. These methods are intended primarily for testing and debugging purposes.

Step 4: Scheduling Agent Listeners

Step 4: Scheduling Agent Listeners

- **Oracle Workflow provides seeded agent listeners:**
 - **Workflow Deferred Agent Listener**
 - **Workflow Deferred Notification Agent Listener**
 - **Workflow Error Agent Listener**
 - **Workflow Inbound Notifications Agent Listener**
 - **Workflow Java Deferred Agent Listener**
 - **Workflow Java Error Agent Listener**
- **You can optionally create additional agent listeners.**

ORACLE

Step 4: Scheduling Agent Listeners

Seeded PL/SQL agent listeners:

- Workflow Deferred Agent Listener: Handles messages on WF_DEFERRED to support deferred subscription processing.
- Workflow Deferred Notification Agent Listener: Handles notification messages on WF_DEFERRED to support outbound notification processing.
- Workflow Error Agent Listener: Handles messages on WF_ERROR to support error handling for the Business Event System.
- Workflow Inbound Notifications Agent Listener: Handles messages on WF_NOTIFICATION_IN to support inbound e-mail notification processing.

Seeded Java agent listeners:

- Workflow Java Deferred Agent Listener: Handles messages on WF_JAVA_DEFERRED to support deferred subscription processing in the middle tier.
- Workflow Java Error Agent Listener: Handles messages on WF_JAVA_ERROR to support error handling for the Business Event System in the middle tier.

Oracle XML Gateway also provides PL/SQL agent listeners named ECX Inbound Agent Listener and ECX Transaction Agent Listener, and a Java agent listener named Web Services IN Agent. See: *Oracle XML Gateway User's Guide*.

You can optionally create additional agent listener service components. For example, you can configure agent listeners for other inbound agents that you want to use for event message propagation, such as the standard WF_IN and WF_JMS_IN agents, or any custom agents. You can also configure an agent listener service component that only processes messages on a particular agent that are instances of a specific event.

Step 4: Scheduling Propagation

Step 4: Scheduling Propagation

- When you send an event message to an agent, the Event Manager places the message on the queue associated with the outbound agent.
- The message is then asynchronously delivered to the recipient by propagation.
- You should schedule propagation for each outbound agent on your local system.

ORACLE®

Step 4: Scheduling Propagation

If you want to use the standard WF_OUT and WF_JMS_OUT agents or custom agents for event message propagation, ensure that you schedule propagation for those agents.

Step 4: Scheduling Propagation

Step 4: Scheduling Propagation

- You can schedule Oracle Advanced Queuing propagation for agents that use the SQLNET protocol by the following methods:
 - Distributed Database Management feature to manage Oracle Advanced Queuing through Oracle Enterprise Manager
 - DBMS_AQADM.Schedule_Propagation API
- For agents that use other protocols, you must provide external propagation logic.
- You can use Oracle Workflow Manager to review the propagation schedules for your local outbound agents.

ORACLE®

Step 4: Scheduling Propagation

For information about using Oracle Enterprise Manager to schedule Oracle Advanced Queuing propagation, see: Oracle Enterprise Manager Support, *Oracle Streams Advanced Queuing User's Guide and Reference* and the Oracle Enterprise Manager online help.

Step 4: Scheduling Propagation

Step 4: Scheduling Propagation

The Schedule_Propagation API is as follows:

```
DBMS_AQADM.Schedule_Propagation (
    queue_name    IN VARCHAR2 ,
    destination   IN VARCHAR2 DEFAULT NULL ,
    start_time    IN DATE      DEFAULT SYSDATE ,
    duration      IN NUMBER    DEFAULT NULL ,
    next_time     IN VARCHAR2 DEFAULT NULL ,
    latency       IN NUMBER    DEFAULT 60) ;
```

ORACLE

Step 4: Scheduling Propagation

The Schedule_Propagation API parameters are:

- queue_name: The queue associated with the local outbound agent for which you want to schedule propagation. Specify the queue name prefaced by the schema that owns the queue, in the following format: <schema>. <queue>
The standard Workflow queues are usually owned by the APPLSYS schema.
- destination: The database link to the remote system to which you want to propagate messages. To propagate messages to another queue on the local system, enter the destination NULL. The default value is NULL.
- start_time: The initial start time for the propagation window.
- duration: The duration of the propagation window, in seconds
- next_time: A date function to compute the start of the next propagation window from the end of the current window. For example, to start the window at the same time every day, next_time should be specified as 'SYSDATE + 1 - duration/86400'. If this value is NULL, then propagation is stopped at the end of the current window and is not run repeatedly. The default value is NULL.

- latency: A latency time in seconds to specify how long you want to wait, after all messages have been propagated, before rechecking the queue for new messages to the destination. The latency represents the maximum wait time during the propagation window for a message to be propagated after it is enqueued. To propagate messages as soon as possible after they are enqueued, enter a latency of zero. The default latency is 60 seconds.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Step 4: Synchronizing License Statuses

Step 4: Synchronizing License Statuses

- Some Oracle E-Business Suite products provide seeded events and subscriptions.
- Oracle Workflow executes subscriptions only if the triggering event and the subscription are both owned by products that you have licensed with a status of Installed or Shared.
- To ensure that the license status of the seeded events and subscriptions is updated for the products that you have licensed, run the Synchronize Product License and Workflow BES License concurrent program.

ORACLE

Step 4: Synchronizing License Statuses

You can use the License Manager utility to review which products you currently have licensed. See: License Manager, *Oracle Applications System Administrator's Guide - Maintenance*.

Use the Submit Requests form to submit the Synchronize Product License and Workflow BES License concurrent program (FNDWFLIC). This program does not require any parameters.

If you upgrade from an Oracle E-Business Suite release earlier than Release 11.5.9, you should run the Synchronize Product License and Workflow BES License concurrent program once after the upgrade to update the license status of the existing events and subscriptions in your Event Manager. Otherwise, subscriptions may not be correctly processed after the upgrade. Subsequently, when you license a product, Oracle Workflow automatically updates the license status for all the events and subscriptions owned by that product.

Note: Any events and subscriptions that you define with a customization level of User are always treated as being licensed.

Step 4: Cleaning Up the WF_CONTROL Queue

Step 4: Cleaning Up the WF_CONTROL Queue

- The Generic Service Component Framework uses the standard WF_CONTROL queue to handle control events for containers and service components.
- The subscribers to the WF_CONTROL queue need to be cleaned up periodically.
- The Workflow Control Queue Cleanup concurrent program is automatically scheduled to perform this cleanup.

ORACLE

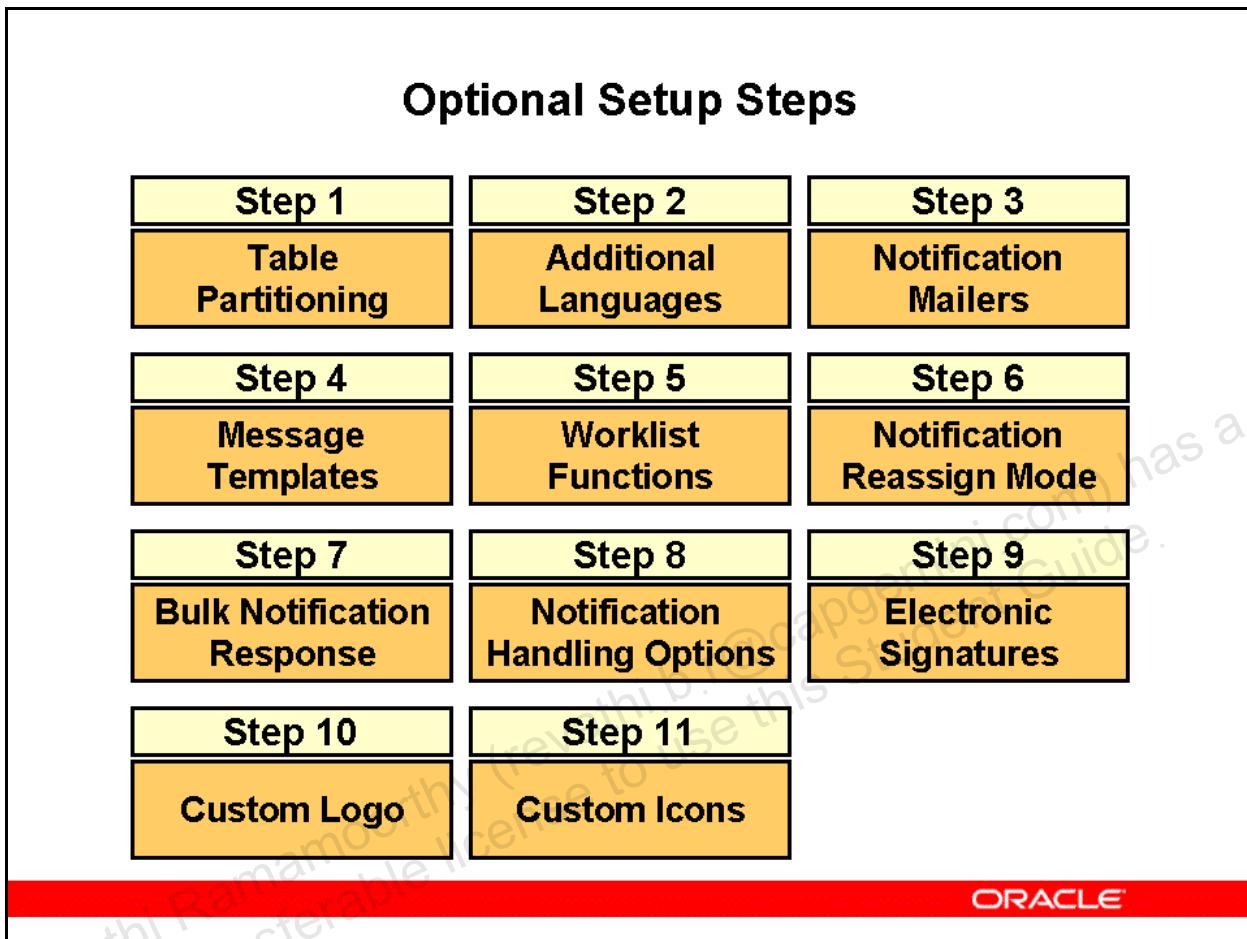
Step 4: Cleaning Up the WF_CONTROL Queue

The WF_CONTROL agent is used for internal processing only. You should not place custom event messages on this queue. You do not need to schedule propagation for the WF_CONTROL agent, because the middle tier processes that use WF_CONTROL dequeue messages directly from its queue. However, this queue should be cleaned up periodically.

When a middle tier process for Oracle E-Business Suite starts up, it creates a JMS subscriber to the WF_CONTROL queue. If a middle tier process dies, however, the corresponding subscriber remains in the database. For more efficient processing, you should ensure that WF_CONTROL is periodically cleaned up by removing the subscribers for any middle tier processes that are no longer active. The recommended frequency for performing cleanup is every 12 hours.

Use Oracle Workflow Manager to manage the Workflow Control Queue Cleanup (FNDWFBES_CONTROL_QUEUE_CLEANUP) concurrent program, which uses the *WF_BES_CLEANUP.Cleanup_Subscribers()* API to perform the necessary cleanup. This concurrent program is scheduled to run every 12 hours by default. The program does not require any parameters.

Optional Setup Steps



Optional Setup Steps

1. Partition certain Workflow tables for performance gain.
2. Set up additional languages if you want to use Oracle Workflow in languages other than English.
3. Set up one or more notification mailers if you want to allow your users to receive notifications by e-mail.
4. Customize the templates for your e-mail notifications.
5. Give users access to the Advanced Worklist, Personal Worklist, Notifications administrator search, and Workflow Mailer URL Access Tester Web pages from any responsibility you choose.
6. Use the WF: Notification Reassign Mode profile option to control which reassign modes are available to users from the Notification Details page.
7. Use the WF: Enable Bulk Notification Response profile option to specify whether users can respond to a group of notifications collectively from the Worklist and Notification Search pages, without navigating to the Notification Details page for each notification individually.

8. Control the item types for which users can define vacation rules and grant worklist access, using the WF: Vacation Rule Item Types lookup type and the WF: Vacation Rules - Allow All profile option.
9. Set up users to enable electronic signatures in notification responses.
10. Customize the company logo that appears in Oracle Workflow Web pages.
11. Include additional icons in your Oracle Workflow icons subdirectory to customize the diagrammatic representation of your workflow processes.

For more information, see: Setting Up Oracle Workflow, *Oracle Workflow Administrator's Guide*.

Optional Step 1: Partitioning Workflow Tables

Optional Step 1: Partitioning Workflow Tables

- You can optionally run the `wfupartb.sql` script to partition certain Workflow tables for performance gain.
- The script is located in the `$FND_TOP/admin/sql` directory.
- The script partitions four Workflow tables and re-creates the associated indexes.

ORACLE

Optional Step 1: Partitioning Workflow Tables

The `wfupartb.sql` script partitions the following tables:

- `WF_ITEM_ACTIVITY_STATUSES`
- `WF_ITEM_ACTIVITY_STATUSES_H`
- `WF_ITEM_ATTRIBUTE_VALUES`
- `WF_ITEMS`

The script recreates the following indexes:

- `WF_ITEM_ACTIVITY_STATUSES_PK`
- `WF_ITEM_ACTIVITY_STATUSES_N1`
- `WF_ITEM_ACTIVITY_STATUSES_N2`
- `WF_ITEM_ACTIVITY_STATUSES_H_N1`
- `WF_ITEM_ACTIVITY_STATUSES_H_N2`
- `WF_ITEM_ATTRIBUTE_VALUES_PK`
- `WF_ITEMS_PK`
- `WF_ITEMS_N1`

- WF_ITEMS_N2
- WF_ITEMS_N3

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Optional Step 2: Setting Up Additional Languages

Optional Step 2: Setting Up Additional Languages

You can:

- **Display Oracle Workflow Web pages in other languages.**
- **Create and view workflow definitions in other languages using Oracle Workflow Builder.**
- **Load workflow definitions in other languages to a database.**
- **Send e-mail notifications in other languages.**

ORACLE®

Optional Step 2: Setting Up Additional Languages

Note: You can display languages that require a multibyte character set only if your database uses a character set that supports these languages, such as UTF8.

Displaying Oracle Workflow Web Pages in Other Languages

You select and install additional languages as part of the Oracle E-Business Suite installation. Users can set their language preference to an installed language through the Personal Homepage.

Creating and Viewing Workflow Definitions in Other Languages using Oracle Workflow Builder

- Set the NLS_LANG environment variable for the new language, territory, and encoded character set that you want to use for the workflow definition. Specify the value for NLS_LANG in the following format: LANGUAGE_TERRITORY.CHARSET Use the Registry Editor on your PC to set the NLS_LANG environment variable:
 - Select Run from the Start menu on your PC.
 - Enter the command to run the Registry Editor and click OK. The command depends on your version of Windows; for example, it may be regedit or regedit32.

- Drill down to My Computer > HKEY_LOCAL_MACHINE > Software > Oracle, and then to the Oracle home where the Workflow Builder is installed.
- Select the NLS_LANG variable and select Modify from the Edit menu.
- Enter the value you want and click OK.
- Exit the Registry Editor.
- Start Oracle Workflow Builder. Create a translated version of your workflow definition and save it as a flat file (.wft), or open and view a workflow definition that is already translated.

Loading Workflow Definitions in Other Languages to a Database

- Ensure that the language you want is set up in the database. You select and install additional languages as part of the Oracle E-Business Suite installation.
- Before running the Workflow Definitions Loader program to load a translated workflow definition to your database, you must set the NLS_LANG environment variable to the appropriate territory and character set for the workflow definition you want to load. The character set must match the character set encoding used to create the workflow definition file, which is determined by the NLS_LANG value that was set on the client PC before the .wft file was created in the Workflow Builder. To set NLS_LANG before running the Workflow Definitions Loader, use the following format: _TERRITORY.CHARSET
Note that it is important to include the underscore (_) before the territory name and the period (.) between the territory name and the character set name in the NLS_LANG value. You do not need to include the language in this NLS_LANG value because the Workflow Definitions Loader uses the language specified within the .wft file to determine the language to load.
- Before using the Workflow Builder to save a translated workflow definition to your database, you must set the NLS_LANG environment variable to the appropriate language, territory, and character set. If you are saving several workflow definitions in different languages, you must reset NLS_LANG for each language.

Sending E-mail Notifications in Other Languages

- Determine whether Oracle has translated the e-mail notification templates to the language you want to set by checking for the file containing the templates in the appropriate language subdirectory, \$FND_TOP/import/<lang>. The standard templates are delivered in a file called wfmail.wft.
- If the e-mail templates are available for the desired language, Oracle Workflow uses the language preference for the notification recipient to determine the language for an e-mail notification.

Optional Step 3: Implementing Notification Mailers

Optional Step 3: Implementing Notification Mailers

- **Implement one or more notification mailers if you want users to receive notifications by e-mail as well as from the Worklist Web pages.**
- **A notification mailer is a Java program that performs e-mail send and response processing for the Oracle Workflow Notification System, using the JavaMail API.**
- **Use Oracle Workflow Manager to configure and run notification mailers.**

ORACLE®

Optional Step 3: Implementing Notification Mailers

Optional Step 3: Implementing Notification Mailers

- The notification mailer program is defined as a service component type in the Generic Service Component Framework.
- Oracle Workflow provides one seeded notification mailer service component, called Workflow Notification Mailer.
- You can optionally create additional notification mailer service components.

ORACLE®

Optional Step 3: Implementing Notification Mailers

Optional Step 3: Implementing Notification Mailers

- Oracle Workflow supports the following protocols:
 - Simple Mail Transfer Protocol (SMTP): For outbound messages
 - Internet Message Access Protocol (IMAP): For inbound messages
- You must have an SMTP server and an IMAP server set up in order to send and receive Oracle Workflow notification e-mail messages from your Workflow server.

ORACLE®

Optional Step 3: Implementing Notification Mailers

Users can receive e-mail notifications using various e-mail clients, although notifications may be displayed differently in different clients, depending on the features each client supports.

Oracle Workflow fully supports Multipurpose Internet Mail Extensions (MIME) encoded messages. This means that a notification mailer can exchange messages with workflow users containing languages with different character sets and multimedia encoded content.

Optional Step 4: Customizing Message Templates

Optional Step 4: Customizing Message Templates

- **Notification mailers use message templates to generate e-mail notifications.**
- **Oracle Workflow provides a set of standard default templates, as well as some alternative templates for certain types of messages.**
- **You can optionally customize your message templates:**
 - **Use the alternative templates provided by Oracle Workflow.**
 - **Create your own custom message templates using the Workflow Builder.**

ORACLE®

Optional Step 4: Customizing Message Templates

Message templates are defined in the System: Mailer item type, stored in a file named wfmail.wft. It is not recommended to modify the standard templates. However, you can choose to use the alternative templates provided by Oracle Workflow instead of the default templates, or you can also create your own custom templates in the System: Mailer item type. For more information, see: Setting Up Oracle Workflow, *Oracle Workflow Administrator's Guide*.

You can implement alternative standard or custom templates in the following ways:

- Assign the templates that you want to a particular notification mailer service component in the mailer configuration parameters. The templates assigned to a mailer override the default System: Mailer templates.
- Assign the templates that you want to a particular notification in a workflow process by defining special message attributes. In this case the templates assigned to the notification override both the templates assigned to a mailer and the default System: Mailer templates.

Optional Step 4: Customizing Message Templates

Optional Step 4: Customizing Message Templates

- **Open Mail (Templated)**
- **Open Mail (Direct)**
- **Open Mail (Outlook Express)**
- **Open FYI Mail**
- **View From UI**
- **View FYI From UI**
- **URL Attachment**
- **Canceled Mail**
- **Invalid Mail**
- **Closed Mail**
- **Summary Mail (HTML)**
- **Warning Mail**
- **Signature Required Mail**
- **Signature Warning Mail**
- **Secure Mail Content**
- **Open Mail (More Information Request)**
- **More Information Request (Outlook Express)**
- **Invalid Open Mail (More Information Request)**
- **More Info Answered Mail**
- **User Notification Preference Update Report**

ORACLE

Optional Step 4: Customizing Message Templates

The following standard message templates are used by default. An asterisk (*) marks the templates for which Oracle Workflow also provides an alternative version.

- * Open Mail (Templated): For notifications that require a response when you are using the templated response method
- * Open Mail (Direct): For notifications that require a response when you are using the direct response method
- * Open Mail (Outlook Express): For notifications that require a response, if you use an e-mail application such as Microsoft Outlook Express as your e-mail client
- * Open FYI Mail: For notifications that do not require a response
- View From UI: For response-required notifications whose content you do not want to send in e-mail
- View FYI From UI: For notifications that do not require a response, whose content you do not want to send in e-mail
- URL Attachment: Creates the Notification References attachment for HTML-formatted notification messages that include URL attributes with Attach Content checked

- * Canceled Mail: Informs the recipient that a notification is canceled
- * Invalid Mail: Informs the recipient that the response to the notification is invalid
- * Closed Mail: Informs the recipient that a previously sent notification is now closed
- Summary Mail (HTML): For notification summaries
- Warning Mail: Informs the recipient of unsolicited mail that he or she sent
- Signature Required Mail: For notifications that require an electronic signature in the user's response; users must respond to such notifications through the Notification Details Web page rather than by e-mail
- Signature Warning Mail: Informs the recipient that an e-mail notification response was not valid because the notification required an electronic signature to be entered through the Notification Details Web page
- Secure Mail Content: For notifications that include sensitive content that cannot be sent in e-mail for security reasons; users must view and respond to such notifications through the Notification Details Web page rather than by e-mail
- * Open Mail (More Information Request): For requests for more information about a notification from one user to another user
- More Information Request (Outlook Express): For requests for more information about a notification from one user to another user, if you use an e-mail application such as Microsoft Outlook Express as your e-mail client
- Invalid Open Mail (More Information Request): Informs the recipient that the response to a request for more information is invalid
- More Info Answered Mail: Informs the recipient that no further response to a request for more information is required, if the user sent another response to a request for more information that was already answered, or if the user sent an unsolicited e-mail formatted as a more information response.
- User Notification Preference Update Report: Informs the administrator that an e-mail notification could not be sent to one or more recipients, that the notification preference for those recipients has been set to DISABLED, and that those recipients' original notification preferences, which are listed, should be reset after the issues that caused the failures are corrected

Optional Step 5: Adding Worklist Functions to User Responsibilities

Optional Step 5: Adding Worklist Functions to User Responsibilities

- You can optionally give users access to these Web pages from any responsibility that you choose:
 - Advanced Worklist
 - Personal Worklist
 - Notifications administrator search page
 - Workflow Mailer URL Access Tester
- To make a page available from a particular responsibility, you must add the appropriate function to the menu associated with that responsibility.
- You can then assign that responsibility to your users.

ORACLE

Optional Step 5: Adding Worklist Functions to User Responsibilities

Add the following functions to the responsibilities from which you want users to access the corresponding pages:

- Advanced Worklist: WF_WORKLIST
- Personal Worklist: WF_WORKLIST_CUSTOM
- Notifications administrator search page: WF_WORKLIST_SEARCH
- Workflow Mailer URL Access Tester page: WF_TEST_MAILER_URL

The Advanced Worklist is seeded on the menu for the Workflow User Web Applications responsibility by default. You can also add its function to other responsibilities from which you want users to access notifications.

The Personal Worklist is an optional feature that is not seeded on any Oracle E-Business Suite menu. If you want users to access this version of the Worklist, you must first add its function to the menu for a responsibility assigned to those users.

If you add the Personal Worklist, you can use worklist flexfields to define specialized worklist views that display information specific to particular types of notifications. If you define a

securing function for a view, add the securing function to the same menu as the Personal Worklist function. Your specialized worklist view will appear in the list of views only when users access the Personal Worklist from that responsibility.

The Notifications administrator search page is seeded on the menu for the Workflow Administrator Web Applications responsibility by default. You can also add its function to other responsibilities from which you want users to be able to search for notifications. For example, if you want users with the Workflow User Web Applications responsibility to have access to the Notifications administrator search page, you can add this function to the FND_WFUSER (Workflow User) menu with a prompt such as Notification Search.

A user must have workflow administrator privileges to access other users' notifications on the Notifications administrator search page. If a user does not have administrator privileges, that user can only search for and access his or her own notifications.

The Workflow Mailer URL Access Tester page is an optional feature that is not seeded on the menu for any Oracle Workflow responsibility. If you want users to access this page, you must first add either the WF_TEST_MAILER_URL function or the Workflow Mailer URL Access Test Menu (WF_TEST_MAILER_APPLICATION) to the menu for a responsibility assigned to those users.

Optional Step 6: Setting the Notification Reassign Mode

Optional Step 6: Setting the Notification Reassign Mode

Use the WF: Notification Reassign Mode profile option to control which reassign modes are available to users.

- **Delegate**
- **Transfer**
- **Reassign**

ORACLE

Optional Step 6: Setting the Notification Reassign Mode

The WF: Notification Reassign Mode profile option controls the reassign modes available to users from the Advanced Worklist, the Personal Worklist, and the Response section of the Notification Details page.

- Delegate: This mode lets users give another user authority to respond to a notification on their behalf, while still retaining ownership of the notification themselves. For example, a manager might delegate all vacation scheduling approvals to an assistant.
- Transfer: This mode lets users give another user complete ownership of and responsibility for a notification. For example, users might select this option if they should not have received a certain notification and they want to send it to the correct recipient or to another recipient for resolution. A transfer may have the effect of changing the approval hierarchy for the notification. For example, a manager might transfer a notification about a certain project to another manager who now owns that project.
- Reassign: This setting provides users access to both the Delegate and Transfer reassign modes. The Reassign setting is the default value for the WF: Notification Reassign Mode profile option.

Optional Step 7: Enabling Bulk Notification Response

Optional Step 7: Enabling Bulk Notification Response

- Use the **WF: Enable Bulk Notification Response** profile option to specify whether users can respond to a group of notifications collectively.
- If you set this profile option to Yes, the **Respond** button appears on the following pages:
 - Advanced Worklist
 - Personal Worklist
 - Personal Worklist Simple Search
 - Personal Worklist Advanced Search
 - Administrator Notification Search

ORACLE®

Optional Step 7: Enabling Bulk Notification Response

If you enable bulk notification response, then users can select several notifications in their worklist and click the Respond button to respond to all the notifications at once, without navigating to the Notification Details page for each notification individually.

If you set the profile option to No, the Respond button is hidden, and users must always respond to each notification individually from the Notification Details page. The default value is No.

Optional Step 8: Setting Up Notification Handling Options

Optional Step 8: Setting Up Notification Handling Options

- **Vacation rules are defined according to the item type with which notifications are associated.**
- **Users can choose to grant worklist access based on the item type with which notifications are associated.**
- **You can control what item types are available for vacation rules and worklist access.**

ORACLE

Optional Step 8: Setting Up Notification Handling Options

Adding Item Types for Vacation Rules and Worklist Access

By default, the list of item types a user can select when creating a vacation rule or when granting worklist access displays those item types for which the user has previously received at least one notification. You can also choose to add item types that you want to appear in the lists for all users. In this way you can allow users to create rules or grant worklist access to handle any notifications they may receive from those item types in the future.

To add an item type to the list, define the internal name of the item type as a lookup code for the WF: Vacation Rule Item Types lookup type.

- Navigate to the Application Object Library Lookups window in the Application Developer responsibility.
- Query the WF_RR_ITEM_TYPES lookup type with the meaning WF: Vacation Rule Item Types in the Application Object Library application.
- Define the item type you want as a new lookup code for this lookup type. Ensure that you enter the item type internal name in the Code field exactly as the name is defined in your database.

Allowing Vacation Rules That Apply to All Item Types

Use the WF: Vacation Rules - Allow All profile option to determine whether the list of item types for vacation rules includes the “All” option. The “All” option lets users create a generic rule that applies to notifications associated with any item type.

- Enabled: The “All” option appears in the list of item types for vacation rules. This is the default value.
- Disabled: Users must always specify the item type to which a vacation rule applies.

After changing the value of this profile option, you must stop and restart Oracle HTTP Server for the change to take effect.

Optional Step 9: Setting Up for Electronic Signatures

Optional Step 9: Setting Up for Electronic Signatures

- **Certificate-based digital signatures:** Load users' certificates into your Oracle E-Business Suite database using the Workflow Certificate Loader.
- **Password-based signatures:** If you implement single sign-on, set the Applications SSO Login Types profile option to either Local or Both for users who enter password-based signatures.

ORACLE

Optional Step 9: Setting Up for Electronic Signatures

Implementing Password-based Signatures with Single Sign-On

Oracle Workflow supports password-based signatures for notifications based on Oracle Application Object Library (FND) passwords. If you maintain your directory service based on Oracle Application Object Library users and passwords, no additional setup is required. However, if you implement single sign-on for your site through Oracle Internet Directory, you must perform the following steps.

- Set the Applications SSO Login Types profile option to either Local or Both at user level for all users who need to enter password-based signatures.
- Ensure that these users have valid passwords defined in Oracle Application Object Library.

For more information, see: Oracle Single Sign-On Integration, *Oracle Applications System Administrator's Guide - Security*.

Loading Certificates for Digital Signatures

Before users can sign responses with their X.509 certificates, you must load these certificates into your Oracle E-Business Suite database using the Workflow Certificate Loader. When you load a certificate, you must also specify the Oracle E-Business Suite user to whom that

certificate is assigned. Oracle Workflow uses this information to validate that the user attempting to sign with a certain certificate is the same user to whom that certificate is assigned.

A user can have more than one certificate assigned to him or her. However, each certificate can only be assigned to one user. Additionally, after you have loaded a certificate for a user, you cannot delete it from the database or assign it to a different user. If a certificate is incorrectly assigned, the user to whom it belongs must revoke it and obtain a new certificate instead.

You must load a user's personal certificate, the root certificate of the certificate authority that issued the personal certificate, and any intermediate certificates required for this type of personal certificate.

Run the loader by running Java against `oracle.apps.fnd.wf.DigitalSignature.loader.CertificateLoader`. You can load several certificates at once by listing the information for all the certificates in a data file for the loader. You can also load a single certificate by specifying the certificate information in the command line for the loader.

Note: Oracle Workflow does not provide a framework for certificate provisioning. Oracle Workflow's digital signature support assumes that certificate provisioning is already part of your infrastructure.

Optional Step 10: Customizing the Workflow Web Page Logo

Optional Step 10: Customizing the Workflow Web Page Logo

- You can customize the company logo that appears on Oracle Workflow Web pages.
- Create your company logo file in GIF format and save it as FNDLOGOS.gif.
- Move the gif file to the physical directory associated with your Web server's /OA_MEDIA/ virtual directory.

ORACLE

Optional Step 10: Customizing the Workflow Web Page Logo

Note: /OA_MEDIA/ is a virtual directory mapping defined in your Web server when Oracle Workflow is installed.

Optional Step 11: Adding Custom Icons

Optional Step 11: Adding Custom Icons

- Oracle Workflow provides a variety of icons that you can use with your activities and processes.
- You can add additional icon files if they are in the appropriate format.
 - Oracle Workflow Builder looks for Windows icon files (.ico) in the Icon subdirectory of the Oracle Workflow area on your PC.
 - The Status Monitor looks for GIF files (.gif) in your Web server's /OA_MEDIA/ virtual directory.

ORACLE

Optional Step 11: Adding Custom Icons

If you create custom icons to include in your Oracle Workflow Builder process definition, and you want the custom icons to appear in the Status Monitor when you view the process, you must do the following:

1. Convert the custom icon files (.ico) to GIF format (.gif).
2. Copy the .gif files to the physical directory associated with your Web server's /OA_MEDIA/ virtual directory, so that the Status Monitor can access them.

Note: /OA_MEDIA/ is a virtual directory mapping defined in your Web server when Oracle Workflow is installed.

Summary

Summary

In this lesson, you should have learned how to:

- **Describe the required setup steps for Oracle Workflow.**
- **Describe the optional setup steps for Oracle Workflow.**

ORACLE®

Managing Service Components

Chapter 28

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Managing Service Components

Managing Service Components

ORACLE®

Objectives

Objectives

After completing this lesson, you should be able to do the following:

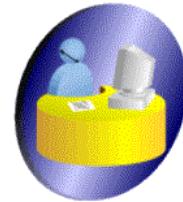
- **Configure and run agent listeners.**
- **Configure and run notification mailers.**

ORACLE®

Oracle Workflow Manager

Oracle Workflow Manager

- **Oracle Workflow Manager is a component that allows administrators to manage Workflow system-level status and features from a central console.**
- **Access Oracle Workflow Manager through Oracle Applications Manager.**



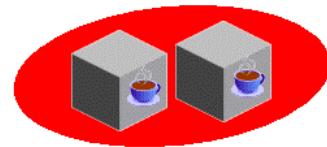
Oracle Workflow Manager

To access Oracle Workflow Manager, log in to Oracle Applications Manager, select Workflow Manager from the Navigate To menu on the Applications Dashboard page, and click the Go button.

Service Components

Service Components

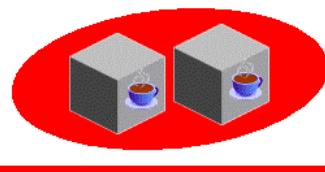
- **The Generic Service Component Framework (GSCF) helps to simplify and automate the management of background Java services.**
- **A service component is an instance of a Java program defined according to GSCF standards so that it can be managed through this framework.**
- **Use Oracle Workflow Manager to manage service components.**



Service Component Containers

Service Component Containers

- A service component container manages the running of the individual service components that belong to it.
 - Monitors the status of its components
 - Handles control events for itself and for its components
 - Records its actions in a log for the container
- A service component container must be running in order for the service components inside it to run.



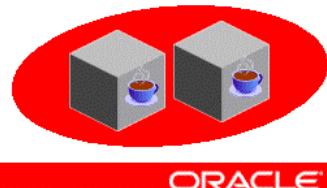
Service Component Containers

Service component containers and their service components are run through Generic Service Management (GSM), which you can control through Oracle Applications Manager.

Service Component Types

Service Component Types

- **Workflow Mailer:** Perform send and respond e-mail processing for the Notification System.
- **Workflow Agent Listener:** Process inbound messages on Business Event System agents in the database.
- **Workflow Java Agent Listener:** Process inbound messages on Business Event System agents in the middle tier.



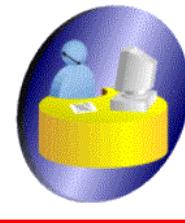
Service Component Types

Oracle E-Business Suite also includes the service component type Workflow Web Services Outbound. These service components process outbound Oracle XML Gateway Web service messages. See: *Oracle XML Gateway User's Guide*.

Accessing Service Components in Oracle Workflow Manager

Accessing Service Components in Oracle Workflow Manager

- On the Workflow System page, click a feature status icon to manage that feature.
 - Service Components
 - Agent Listeners
 - Notification Mailers
- You can also click the Service Components link in the Configuration section of the Related Links region.



Accessing Service Components in Oracle Workflow Manager

If you click the Service Components status icon or link, the Service Components page displays all types of service components.

If you click the Agent Listeners status icon, the Service Components page automatically filters the list to display only PL/SQL agent listeners and Java agent listeners.

If you click the Notification Mailers status icon, the Service Components page automatically filters the list to display only notification mailers.

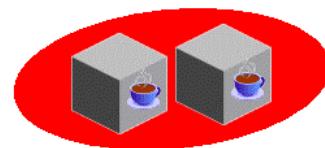
If the status of a service component changes to Stopped with Error or System Deactivated, Oracle Workflow posts a system alert to the System Alerts and Metrics page in Oracle Applications Manager.

Managing Service Components

Managing Service Components

On the Service Components page, use the administrative buttons to:

- **Verify the status of all service components**
- **Create a new service component**
- **Edit a service component's configuration**
- **View the service component container log**
- **View details about a service component**
- **Review any control events scheduled to control the running of a service component**
- **Delete a service component**



ORACLE

Managing Service Components

You can also perform the following actions on the Service Components page.

- To manually filter the service components displayed in the list, select a service component property from the Filter pull-down menu, enter a filter value in the text field, and click the Go button. You can filter by the following properties:
 - Service component name
 - Service component status
 - Service component type display name
 - Service component type internal name
- You can automate the running of a service component by assigning it an Automatic or On-Demand startup type, or by scheduling control events in advance. However, if you want to manually control the running of a service component, select the service component, choose the command that you want from the command pull-down menu, and click the Go button. You can choose the following control commands:
 - Refresh
 - Resume

- Start
- Stop
- Suspend
- Launch Summary Notifications (Workflow Mailer service components only)
- To review details for the container to which a service component belongs, click the container link in the Container column.

When you choose to create a new service component, a page appears to let you select the type of service component that you want to create. After you select a service component type, the configuration wizard for that type appears.

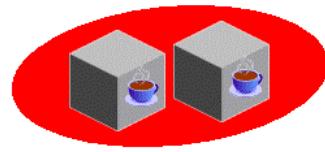
Similarly, when you choose to edit an existing service component, the appropriate configuration wizard appears.

On the Component Details page, you can review the configuration parameters and scheduled control events for the selected service component. You can also review the history of control events scheduled for the service component, review the service component container log, access the configuration wizard to edit the service component, or perform other actions appropriate to that type of service component.

Service Component Container Logs

Service Component Container Logs

- **Oracle Workflow Manager records a log for each service component container, which includes the service components within that container.**
- **You can assign a log level to a service component container to specify how much detail to record for the container's actions.**
- **You can also assign a log level separately to each service component within the container to control the details logged for that component.**



ORACLE

Service Component Container Logs

The default container log level, which is also the recommended setting, is Error. You can optionally specify a different log level. The log levels that you can select, in order from most detailed to least detailed, are as follows:

- 1 - Statement
- 2 - Procedure
- 3 - Event
- 4 - Exception
- 5 - Error
- 6 - Unexpected

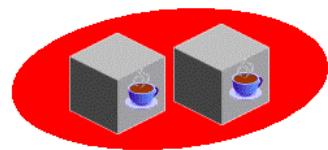
To specify a log level for a service component container, click the container link on the Service Components page, select the container, and click the View Status button to navigate to the Service Status page. Then select the log level that you want to assign to the container from the Change Log Level To pull-down menu, and click Go.

You can specify a log level for an individual service component in the configuration wizard for that component.

Service Component Startup Modes

Service Component Startup Modes

- **Automatic**
- **On-Demand**
- **Manual**



ORACLE

Service Component Startup Modes

A service component can have one of three startup modes.

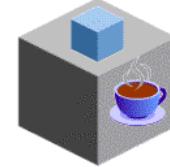
- Automatic: When a component container is started, it will automatically start its automatic service components. It will also monitor these components and restart them automatically when necessary.
- On-Demand: A component container will start its on-demand service components if those components have messages waiting to be processed. For example, an on-demand notification mailer service component will be started if there are messages waiting on the WF_NOTIFICATION_OUT queue. The container will stop an on-demand service component when that component's maximum idle time has been exceeded.
- Manual: You must manually start and stop the service component through Oracle Workflow Manager. The component container does not start or stop its manual service components.

Agent Listeners

Agent Listeners

An agent listener monitors a Business Event System agent for incoming messages and dequeues messages using the agent's queue handler.

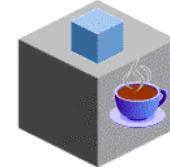
- PL/SQL agent listeners: Process event subscriptions with a PL/SQL rule function in the database
- Java agent listeners: Process event subscriptions with a Java rule function or subscriptions to events with a Java generate function in the middle tier



Agent Listeners

Agent Listeners

- **Seeded PL/SQL agent listeners:**
 - Workflow Deferred Agent Listener
 - Workflow Deferred Notification Agent Listener
 - Workflow Error Agent Listener
 - Workflow Inbound Notifications Agent Listener
- **Seeded Java agent listeners:**
 - Workflow Java Deferred Agent Listener
 - Workflow Java Error Agent Listener
- **You can optionally create additional agent listener service components.**



Agent Listeners

Oracle Workflow provides the following seeded PL/SQL agent listeners:

- Workflow Deferred Agent Listener: Handles messages on WF_DEFERRED to support deferred subscription processing.
- Workflow Deferred Notification Agent Listener: Handles notification messages on WF_DEFERRED to support outbound notification processing.
- Workflow Error Agent Listener: Handles messages on WF_ERROR to support error handling for the Business Event System.
- Workflow Inbound Notifications Agent Listener: Handles messages on WF_NOTIFICATION_IN to support inbound e-mail notification processing.

Oracle XML Gateway also provides seeded PL/SQL agent listeners named ECX Inbound Agent Listener and ECX Transaction Agent Listener. See: *Oracle XML Gateway User's Guide*.

Oracle Workflow provides the following seeded Java agent listeners:

- Workflow Java Deferred Agent Listener: Handles messages on WF_JAVA_DEFERRED to support deferred subscription processing in the middle tier.

- Workflow Java Error Agent Listener: Handles messages on WF_JAVA_ERROR to support error handling for the Business Event System in the middle tier.

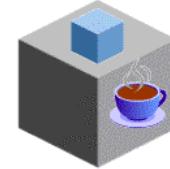
Oracle XML Gateway also provides a seeded Java agent listener named Web Services IN Agent. See: *Oracle XML Gateway User's Guide*.

You can configure additional agent listeners for other inbound agents that you want to use for event message propagation, such as the standard WF_IN and WF_JMS_IN agents, or any custom agents. You can also configure an agent listener that only processes messages on a particular agent that are instances of a specific event.

Agent Listener Configuration Wizards

Agent Listener Configuration Wizards

- In the agent listener configuration wizards, you can:
 - Define general and detail attributes for the agent listener.
 - Optionally schedule control events to control in advance when the agent listener runs.
 - Review the configuration that you have just defined.
- PL/SQL and Java agent listeners have the same configuration parameters.



ORACLE

Agent Listener Configuration Wizards

Use the Next and Back buttons to navigate among the pages of the configuration wizards.

Define Page

- ID: When you edit a previously created service component, the configuration wizard displays the identifier for the service component.
- Status: When you edit a previously created service component, the configuration wizard displays the status of the service component.
- Name: The unique name of the service component.
- Startup Mode: Select Automatic, Manual, or On-Demand.
- Container Type: The container type to which this service component belongs. In Oracle E-Business Suite, the container type is always Oracle Applications Generic Service Management (Oracle Applications GSM).
- Inbound Agent: The Business Event System agent that you want to monitor for inbound event messages.
- Outbound Agent: Leave this field blank. Agent listeners do not use an outbound agent.

- Correlation ID: Optionally specify the Oracle Advanced Queuing (AQ) correlation ID of the event messages that you want the agent listener to process. The AQ correlation ID for an event message in the Business Event System is usually specified in the following format: <event name>

Consequently, by specifying a correlation ID in this attribute, you can dedicate the agent listener to listen only for messages that are instances of the specified event. You can also specify a partial value to listen for messages that are instances of any event whose name begins with the specified value.

Details Page

- ID: When you edit a previously created service component, the configuration wizard displays the identifier for the service component.
- Status: When you edit a previously created service component, the configuration wizard displays the status of the service component.
- Name: The configuration wizard displays the name defined for the service component.
- Container: The container to which the service component will belong. The container for agent listener service components is a predefined container called Workflow Agent Listener Service.
- Maximum Idle Time: If you selected the On-Demand startup mode for the service component, enter the maximum time in minutes that the service component can remain idle before it is stopped. An on-demand component that is stopped in this way will be restarted by its container when it is needed again to process new messages.
- Max Error Count: The number of consecutive errors the service component can encounter before its container stops it and changes its status to Stopped with Error. If an error is resolved and processing can continue, the error count is reset.
- Inbound Thread Count: Set the inbound processing thread count to 1 or higher to enable inbound message processing with this agent listener. Select 0 to disable this agent listener. The default value is 1. If this agent listener receives a high volume of inbound messages, you can set the inbound thread count to a higher value to increase throughput.
- Outbound Thread Count: Leave this parameter set to the default value of 0. Agent listener service components do not perform outbound message processing.
- Log Level: Select the level of detail for the information you want to record in the service component container log.
- Processor Read Wait Timeout: Specify the amount of time in seconds that the service component's processing thread continues to wait, after reading the last message from its assigned queue, before timing out. If another message is received before this time expires, that message is processed and the timeout period begins again. If the timeout period expires and no more messages have been received, the service component stops reading and its sleep time begins.
- Processor Min Loop Sleep: Specify the minimum sleep time in seconds during which the service component waits, after its read timeout period expires, before it checks its queue for messages again.
- Processor Max Loop Sleep: Specify the maximum sleep time in seconds if you want to increase the sleep time between read attempts when no messages are received. If you specify a maximum sleep time that is greater than the minimum sleep time, then the

service component initially waits for the minimum sleep time after it finishes reading messages from its queue. If no messages are read in subsequent attempts, then the sleep time between read attempts gradually increases until the maximum sleep time is reached. Increasing the sleep time can help enhance performance if messages are received infrequently. You can also specify 0 for this parameter to indicate that the sleep time should not be increased. In this case, the service component always waits for the minimum sleep time between read attempts.

- Processor Error Loop Sleep: Specify the sleep time in seconds during which the service component waits, after an error occurs, before it attempts to begin processing again.
- Processor Close on Read Timeout: Select this parameter to specify that the service component should close its connections after its read timeout period expires, when its sleep time begins. Deselect this parameter to specify that the connections should remain open until the processing thread stops.

Scheduling Events Page

You can schedule the following events to be raised to control the running of an agent listener service component.

- Start
- Refresh
- Suspend
- Resume
- Stop

Click the Add a Row or Add Another Row button to add a new row to the list of events and enter the information for the event.

- Select the event for the command you want to schedule.
- Select the date when you want the event to be raised first.
- Select the hour and minute to specify the time on the specified date when you want the event to be raised first. The hour values are in a twenty-four hour format. For example, select 00 for midnight, or 23 for 11 PM.
- If you want to raise the event periodically, enter the time interval in minutes at which you want to raise the event. If you do not specify a repeating interval, the event is raised only once.
- If you choose the refresh event, you can optionally enter parameters to refresh the agent listener configuration parameters with those values when the event is raised.

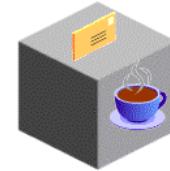
Review Page

If you want to change any of these settings, return to the appropriate step in the agent listener configuration wizard to make your changes. To save these settings and finish the configuration, click the Finish button.

Notification Mailers

Notification Mailers

- A notification mailer performs e-mail send and response processing for the Oracle Workflow Notification System, using the JavaMail API.
 - Simple Mail Transfer Protocol (SMTP) for outbound messages
 - Internet Message Access Protocol (IMAP) for inbound messages
- Oracle Workflow provides one seeded notification mailer service component, the Workflow Notification Mailer.
- You can optionally create additional notification mailer service components.



ORACLE®

Notification Mailers

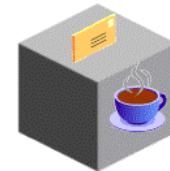
For example, you can create a notification mailer that processes only messages that belong to a particular workflow item type, or create additional mailers that process the same types of message to increase throughput. The correlation ID for a notification mailer can be set to a specific item type to determine which messages it can process.

You can also configure any notification mailer service component to process only inbound messages, or only outbound messages. You associate inbound and outbound mailers with each other by assigning them the same mailer node name. The mailer node name indicates which inbound mailer can process incoming responses to outbound messages sent by a particular outbound mailer.

Outbound Notification Mailer Processing

Outbound Notification Mailer Processing

- **The Workflow Engine raises an `oracle.apps.wf.notification.send` event, which is placed on the `WF_DEFERRED` agent.**
- **When the Workflow Deferred Notification Agent Listener dequeues the event, an event subscription places an XML representation of the notification message on the `WF_NOTIFICATION_OUT` agent.**
- **A notification mailer service component dequeues the event and sends the e-mail message to the recipient.**



Outbound Notification Mailer Processing

When the event is dequeued from `WF_DEFERRED` and the subscription is processed, the subscription requires the event data for the event, causing the Generate function for the event to be executed. The Generate function for this event performs the following actions:

- Resolves the notification recipient role to a single e-mail address, which itself can be a mail list.
- Checks the notification preference of the recipient to determine whether an e-mail notification is required, and in what type of format.
- Switches its database session to the recipient role's preferred language and territory as defined in the directory service.
- Generates an XML representation of the notification message and any optional attachments using the appropriate message template.

Finally, the subscription places the event message on the standard `WF_NOTIFICATION_OUT` agent.

A notification mailer service component polls the `WF_NOTIFICATION_OUT` agent for messages that must be sent by e-mail. When the notification mailer dequeues a message from this agent, it uses a Java-based notification formatter to convert the XML representation of the

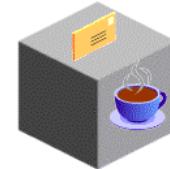
notification into a MIME (Multipurpose Internet Mail Extensions) encoded message and sends the message by the Simple Mail Transfer Protocol (SMTP).

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Inbound Notification Mailer Processing

Inbound Notification Mailer Processing

- A notification mailer service component checks its IMAP e-mail account for incoming messages.
- After validating a message, the notification mailer places an XML representation of the message on the WF_NOTIFICATION_IN agent as an event called oracle.apps.wf.notification.receive.message.
- When the Workflow Inbound Notifications Agent Listener dequeues the event, an event subscription calls the appropriate notification response function to record the response and complete the notification.



Inbound Notification Mailer Processing

A notification mailer processes e-mail responses from users using the Internet Message Access Protocol (IMAP). The notification mailer uses a Java-based e-mail parser to interpret the text of each message and create an XML representation of it.

A notification mailer uses three folders in your response mail account for response processing: one to receive incoming messages, one to store processed messages, and one to store discarded messages.

A notification mailer does the following to process response messages:

- Logs into its IMAP e-mail account.
- Checks the inbox folder for messages. If a message exists, the notification mailer reads the message, checking for the notification ID (NID) and node identifier in the NID line.
- If the message is not a notification response, meaning it does not contain an NID line, the notification mailer moves the message to the discard folder and treats it as an unsolicited message.
- If the message is a notification response, but for the wrong node, the notification mailer leaves the message in the inbox and adds the e-mail's Unique Message ID (UID) to its ignore list.

- If the message is a notification response for the current node, meaning it contains an NID line including the node identifier of the current node, the notification mailer processes the message.

The notification mailer performs the following steps for messages that belong to its node.

- Retrieves the notification ID.
- Checks to see if the message bounced by referring to the tags specified in the configuration parameters. If the message bounced, the notification mailer reroutes it or updates the notification's status and stops any further processing, depending on the specifications of the tag list.
- Checks the Oracle Workflow database for this notification based on the NID line.
 - If the notification does not exist, meaning the notification ID or the access key in the NID line is invalid, the notification mailer moves the message to the discard folder. If the NID line is incorrectly formatted, the notification mailer moves the message to the discard folder and treats it as an unsolicited message.
 - If the notification exists, but is closed or canceled, the notification mailer moves the message to the discard folder and sends a Workflow Closed Mail or Workflow Canceled Mail message to the recipient role, respectively.
 - If the notification exists and is open, the notification mailer generates an XML representation of the message and places it on the WF_NOTIFICATION_IN agent. The notification mailer then moves the message for the completed notification to the processed folder.

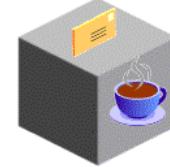
Finally, if there are no more unprocessed messages in the inbox, the notification mailer logs out of the mail and database accounts.

When an event message is dequeued from WF_NOTIFICATION_IN, Oracle Workflow executes a seeded subscription that calls the appropriate notification response function. This function verifies the response values with the definition of the notification message's response attributes in the database. If a response value is invalid, or if no response value is included, the notification mailer sends a Workflow Invalid Mail message to the recipient role, or, for an invalid response to a request for more information, the notification mailer sends a Workflow Invalid Open Mail (More Information Request) message to the recipient role. If the responses are valid, the notification response function records the response and completes the notification.

Notification Mailer Setup

Notification Mailer Setup

- Set up an **SMTP mail server** and an **IMAP mail server**.
- Set up an **IMAP e-mail account** for the notification mailer with three folders: one to use as an inbox, one to store processed messages, and one to store discarded messages.
- Check that the service component containers and the required agent listeners are running.
- Configure the notification mailer using the configuration wizard.



Notification Mailer Setup

You should also ensure that the Business Event System status is set to Enabled on the Configuration page, and that the JOB_QUEUE_PROCESSES database initialization parameter, which is required for the Business Event System, is set to an appropriate value. The Business Event System status is set to Enabled by default, and usually you do not need to change the status. If notification processing is not being completed, however, you should check this preference value.

You can optionally set the WF: Workflow Mailer Framework Web Agent profile option to the host and port of the Web server that notification mailers should use to generate the content for Oracle Application Framework regions that are embedded in notifications. If this profile option is not set, notification mailers will use the same Web agent specified in the Application Framework Agent profile option. However, if necessary for load balancing purposes, you can optionally specify a different Web agent for notification mailers to use. The WF: Workflow Mailer Framework Web Agent profile option should be set at site level.

Before a service component can run, the container which manages it must first be started. Check that the containers for agent listeners and notification mailers are running. Also check that the Workflow Deferred Notification Agent Listener, Workflow Error Agent Listener, and Workflow Inbound Notifications Agent Listener are running.

By default, the seeded Workflow Notification Mailer has a Launch Summary Notifications event scheduled to send summary notifications once a day. You can optionally use the notification mailer configuration wizard to modify the start time and interval for this event's schedule, or to schedule the Launch Summary Notifications event at the interval you choose for any notification mailer service component. When this event is processed, a summary notification is sent to each role with a notification preference of SUMMARY or SUMHTML. You can optionally configure a notification mailer to connect to the SMTP server and IMAP server through Secure Sockets Layer (SSL) to encrypt the data exchanged.

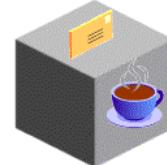
You can optionally set the internal mailer parameter named HTML_DELIMITER to specify which characters the notification mailer uses to delimit response values in response templates for HTML-formatted e-mail notifications. See: Setting Up a Notification Mailer, *Oracle Workflow Administrator's Guide*.

The seeded Workflow Notification Mailer uses the Automatic startup mode by default and will be started automatically when you complete its configuration. If you select the Manual startup mode for a notification mailer service component, use the Service Components page to start that notification mailer.

Connecting to Mail Servers Through SSL

Connecting to Mail Servers Through SSL

- You can optionally configure a notification mailer to connect to the SMTP server and IMAP server through the SSL protocol.
- SSL provides encrypted connections for sending data between the notification mailer and the mail servers, for enhanced security.



Connecting to Mail Servers Through SSL

To use SSL, you must have loaded on the SMTP and IMAP servers an X.509 certificate and private key that were issued by a certificate authority. You can use the same certificate for both the SMTP server and the IMAP server.

Additionally, to connect to the SMTP server through SSL, you must have an Stunnel process running on the SMTP server, with the location of the certificate file specified in the Stunnel arguments. Stunnel is a program that lets you encrypt connections inside SSL. For more information, see: <http://www.stunnel.org>

You can then enable SSL for the SMTP and IMAP servers in the notification mailer configuration wizard.

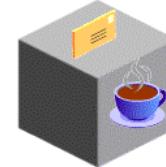
After completing the SSL setup, stop and restart the service component container named Workflow Mailer Service for the changes to take effect.

Notification Mailer Basic Configuration

Notification Mailer Basic Configuration

On the Basic Configuration page you can enter the minimum required parameters.

- **Details - Name**
- **Outbound E-mail Account (SMTP)**
 - **Server Name**
 - **Outbound SSL Enabled**
- **Inbound E-mail Account (IMAP)**
 - **Inbound Processing**
 - **Server Name**
 - **Username**
 - **Password**
 - **Reply-To Address**
 - **Inbound SSL Enabled**



ORACLE®

Notification Mailer Basic Configuration

The Basic Configuration page is the first page of the notification mailer configuration wizard.

- Details
 - Name: The unique name of the service component.
- Outbound E-mail Account (SMTP)
 - Server Name: The name of the outbound SMTP mail server.
 - Outbound SSL Enabled: Select this parameter to enable the notification mailer to use SSL for connections to the SMTP server. You must also complete additional setup steps to use SSL.
- Inbound E-mail Account (IMAP)
 - Inbound Processing: Select this parameter to enable inbound e-mail processing with this notification mailer. Deselect this parameter to disable inbound e-mail processing for this notification mailer and dedicate the notification mailer solely to outbound processing. If you disable inbound processing, you can leave the other inbound parameters blank.
 - Server Name: The name of the inbound IMAP mail server.

- Username: The user name of the mail account that the notification mailer uses to receive e-mail messages.
- Password: The password for the mail account specified in the Username parameter.
- Reply-To Address: The address of the e-mail account that receives incoming messages, to which notification responses should be sent.
- Inbound SSL Enabled: Select this parameter to enable the notification mailer to use SSL for connections to the IMAP server. You must also complete additional setup steps to use SSL.

To send a test message, click the Test Mailer button. On the Test Notification Mailer page, select the recipient role to which the message should be sent, and click the Send Test Message button. Then check the e-mail account for the recipient role to verify that the test message was received.

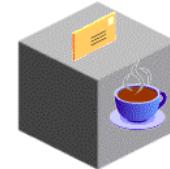
To set additional parameters for this notification mailer in the advanced configuration wizard, click the Advanced button.

Notification Mailer Advanced Configuration

Notification Mailer Advanced Configuration

In the advanced configuration wizard, you can:

- Define general and detail attributes, as well as e-mail server and message generation parameters, for the notification mailer.
- Optionally schedule control events to control in advance when the notification mailer runs.
- Define tags to assign statuses to unusual messages.
- Send a test message.
- Review the configuration you have just defined.



Notification Mailer Advanced Configuration

Use the Next and Back buttons to navigate among the pages of the configuration wizard.

Define Page

- ID: The configuration wizard displays the identifier for the service component.
- Status: The configuration wizard displays the status of the service component.
- Name: The unique name of the service component.
- Startup Mode: Select Automatic, Manual, or On-Demand.
- Container Type: The container type to which this notification mailer belongs, which is always Oracle Applications Generic Service Management (Oracle Applications GSM).
- Inbound Agent: The Business Event System agent for inbound processing. The inbound agent for a notification mailer is always WF_NOTIFICATION_IN.
- Outbound Agent: The Business Event System agent for outbound processing. The outbound agent for a notification mailer is always WF_NOTIFICATION_OUT.
- Correlation ID: Optionally select an item type to specify that this notification mailer should only process messages that belong to that item type. If you enter a partial value, this notification mailer will process messages that belong to any item type whose internal

name begins with that value. You can enter an item type as the correlation ID if you want to increase throughput for that particular item type by dedicating a notification mailer to it. If you leave the correlation ID blank, this notification mailer will process messages from any item type. The seeded Workflow Notification Mailer does not have any correlation ID specified, so that it can operate generally to process all messages.

Details Page

- ID: The configuration wizard displays the identifier for the service component.
- Status: The configuration wizard displays the status of the service component.
- Name: The configuration wizard displays the name defined for the service component.
- Container: The container to which the service component will belong. The container for notification mailer service components is a predefined container called Workflow Mailer Service.
- Maximum Idle Time: If you selected the On-Demand startup mode for the service component, enter the maximum time in minutes that the service component can remain idle before it is stopped. An on-demand component that is stopped in this way will be restarted by its container when it is needed again to process new messages.
- Max Error Count: The number of consecutive errors the service component can encounter before its container stops it and changes its status to Stopped with Error. If an error is resolved and processing can continue, the error count is reset.
- Inbound Thread Count: Set the inbound processing thread count to 1 to enable inbound message processing with this notification mailer. Select 0 to disable inbound message processing for this notification mailer.
- Outbound Thread Count: Specify the number of outbound processing threads you want to execute simultaneously with this notification mailer. You can set the outbound thread count to 1 or more depending on the volume of outbound messages you need to send. Specify 0 to disable outbound message processing for this notification mailer.
- Log Level: Select the level of detail for the information you want to record in the service component container log.
- Processor Read Wait Timeout: Specify the amount of time in seconds that the service component's processing thread continues to wait, after reading the last message from its assigned queue, before timing out. If another message is received before this time expires, that message is processed and the timeout period begins again. If the timeout period expires and no more messages have been received, the service component stops reading and its sleep time begins.
- Processor Min Loop Sleep: Specify the minimum sleep time in seconds during which the service component waits, after its read timeout period expires, before it checks its queue for messages again.
- Processor Max Loop Sleep: Specify the maximum sleep time in seconds if you want to increase the sleep time between read attempts when no messages are received. If you specify a maximum sleep time that is greater than the minimum sleep time, then the service component initially waits for the minimum sleep time after it finishes reading messages from its queue. If no messages are read in subsequent attempts, then the sleep time between read attempts gradually increases until the maximum sleep time is reached. Increasing the sleep time can help enhance performance if messages are received

infrequently. You can also specify 0 for this parameter to indicate that the sleep time should not be increased. In this case, the service component always waits for the minimum sleep time between read attempts.

- Processor Error Loop Sleep: Specify the sleep time in seconds during which the service component waits, after an error occurs, before it attempts to begin processing again.
- Processor Close on Read Timeout: Select this parameter to specify that the service component should close its connections after its read timeout period expires, when its sleep time begins. Deselect this parameter to specify that the connections should remain open until the processing thread stops.

E-mail Servers Page

General

- Mailer Node Name: The node identifier name used by this notification mailer. The maximum length of the node name is eight characters. The node name is included with the outgoing notification ID in outbound messages, and is used in inbound messages to identify the notification mailer that should process the messages.
- Email Parser: The Java class used to parse an incoming notification response e-mail and create an XML document for the response. The standard default e-mail parser provided by Oracle Workflow is named oracle.apps.fnd.wf.mailer.TemplatedEmailParser. Usually you do not need to change this value.
- Alternate Email Parser: The Java class used to parse an incoming notification response e-mail formatted according to the direct response method and to create an XML document for the response. The notification mailer uses this parser when the Direct Response parameter is selected. The default alternate e-mail parser provided by Oracle Workflow is named oracle.apps.fnd.wf.mailer.DirectEmailParser. Usually you do not need to change this value.
- Expunge Inbox on Close: Select this parameter to purge deleted messages from the inbox folder when the notification mailer closes this folder. If you do not select this parameter, copies of messages that were moved to the discard or processed folders remain in the inbox, in a deleted state, until you manually expunge them using your e-mail application.

Inbound E-mail Account

- Inbound Protocol: Oracle Workflow supports the IMAP protocol for inbound e-mail.
- Inbound Server Name: The name of the inbound mail server.
- Username: The user name of the mail account that the notification mailer uses to receive e-mail messages.
- Password: The password for the mail account specified in the Username parameter.
- Inbox Folder: The name of the folder from which the notification mailer receives inbound messages. This value is case-insensitive. The default value is INBOX.
- Inbound Connection Timeout: The maximum amount of time in seconds that the notification mailer will wait to establish a connection to the inbound server before timing out.
- Inbound Message Fetch Size: The maximum number of messages that the notification mailer can fetch from the inbox at one time.

- Maximum Ignore List Size: The maximum number of notification IDs that the notification mailer can store in its ignore list, indicating that this notification mailer will make no further attempts to process them after encountering errors. Usually you do not need to change this value.
- Inbound SSL Enabled: Select this parameter to enable the notification mailer to use SSL for connections to the IMAP server. You must also complete additional setup steps to use SSL.

Outbound E-mail Account

- Outbound Protocol: Oracle Workflow supports the SMTP protocol for outbound e-mail.
- Outbound Server Name: The name of the outbound mail server.
- Test Address: This parameter has been replaced by the override e-mail address, which is available through the Component Details page for a notification mailer.
- Outbound Connection Timeout: The maximum amount of time, in seconds, that the notification mailer will wait to establish a connection to the outbound server before timing out.
- Outbound SSL Enabled - Select this parameter to enable the notification mailer to use Secure Sockets Layer (SSL) for connections to the SMTP server. You must also complete additional setup steps to use SSL.

E-mail Processing

- Processed Folder: The name of the mail folder where the notification mailer places successfully processed notification messages. This value is case-insensitive. The default value is PROCESS.
- Discard Folder: The name of the mail folder where the notification mailer places incoming messages that are not recognized as notification messages. This value is case-insensitive. The default value is DISCARD.
- Allow Forwarded Response: Indicate whether to allow a user to respond by e-mail to an e-mail notification that has been forwarded from another role.

Message Generation Page

Send

- From: The value that appears in the From field of the message header of a notification e-mail.
- Reply-to Address: The address of the e-mail account that receives incoming messages, to which notification responses should be sent.
- HTML Agent: The base URL that identifies the HTML agent that handles HTML notification responses. This URL is required to support e-mail notifications with HTML attachments.
- Message Formatter: Oracle Workflow uses the `oracle.apps.fnd.wf.mailer.NotificationFormatter` Java class to generate notification messages.
- Framework User: The numerical user ID for the user through which a notification mailer accesses Oracle Application Framework content for inclusion in e-mail notifications. The Framework user must have workflow administrator privileges in order to access the content for every user's notifications. If you change the Workflow System Administrator

preference, check the Framework User parameter to ensure that the user accessed by the notification mailer has workflow administrator privileges. Set the Framework User parameter to a user that is a member of the Workflow System Administrator role, or, if you set the Workflow System Administrator preference to a responsibility, set the Framework User parameter to a user that has that responsibility.

- Framework Responsibility: The numerical responsibility ID for the responsibility through which a notification mailer accesses Oracle Application Framework content for inclusion in e-mail notifications. The user specified in the Framework User parameter must have this responsibility assigned.
- Framework Application ID: The numerical application ID for the application through which a notification mailer accesses Oracle Application Framework content for inclusion in e-mail notifications. The responsibility specified in the Framework Responsibility parameter must be assigned to this application.
- Framework URL Timeout: The maximum amount of time, in seconds, that the notification mailer will wait to access a URL for Oracle Application Framework content before timing out.
- Attach Images to Outbound Emails: Select this parameter to attach any images referenced in HTML content included in a message, such as Oracle Application Framework content, to outbound notification e-mail messages. Deselect this parameter to display the image references as off-page URLs.
- Attach Stylesheet to Outbound Email: Select this parameter to attach any stylesheet referenced in HTML content included in a message, such as Oracle Application Framework content, to outbound notification e-mail messages. Deselect this parameter to display the stylesheet reference as a URL.
- Autoclose FYI: Indicate whether this notification mailer automatically closes notifications that do not require a response, such as FYI (For Your Information) notifications, after sending the notifications by e-mail.
- Direct Response: By default, notification mailers require a response format for plain text notifications called the templated response method. Select this parameter to use the direct response method instead.
- Reset NLS: Indicate whether the notification mailer should convert the NLS codeset for a notification message according to the notification recipient's preferences before composing the message.
- Inline Attachments: Select this parameter to set the Content-Disposition MIME header to inline for attachments to notification messages, including the Notification Detail Link, HTML Message Body, Notification References containing attached URLs, and attached PL/SQL documents. Deselect this parameter to set the Content-Disposition MIME header to attachment for these attachments. For example, if your e-mail application cannot display HTML content such as the Notification Detail Link inline, deselect this parameter to display this link as an attachment instead.
- Send Warning for Unsolicited E-mail: Select this parameter to allow the notification mailer to send back a warning message the first time it receives an unsolicited e-mail message from a particular e-mail address.

- Send E-mails for Canceled Notifications: Select this parameter to allow the notification mailer to send cancellation messages to users when previously sent notifications are canceled.

Templates

This region lets you specify the message templates you want to use to generate e-mail notifications. The template for a particular type of e-mail notification determines the basic format of the notification, including what header information to include, and whether and where to include details such as the message due date and priority.

Oracle Workflow provides a set of standard templates in the System: Mailer item type, which are used by default. It is not recommended to modify the standard templates. However, you can customize the message templates used to send your e-mail notifications by assigning a notification mailer alternative templates provided by Oracle Workflow or custom templates that you create in Oracle Workflow Builder. The templates assigned to a mailer override the default System: Mailer templates.

Scheduling Events Page

You can schedule the following events to be raised to control the running of a notification mailer.

- Start
- Refresh
- Suspend
- Resume
- Stop
- Launch Summary Notifications

Click the Add a Row or Add Another Row button to add a new row to the list of events and enter the information for the event.

- Select the event for the command you want to schedule.
- Select the date when you want the event to be raised first.
- Select the hour and minute to specify the time on the specified date when you want the event to be raised first. The hour values are in a twenty-four hour format. For example, select 00 for midnight, or 23 for 11 PM.
- If you want to raise the event periodically, enter the time interval in minutes at which you want to raise the event. If you do not specify a repeating interval, the event is raised only once.
- If you choose the refresh event, you can optionally enter parameters to refresh the notification mailer configuration parameters with those values when the event is raised.

Tags Page

This page lets you enter patterns of text found in unusual messages and the status you want to assign to an inbound message if it contains any of those patterns. For example, unusual messages include bounced or returned messages and auto-reply messages such as those sent by vacation daemons, mass mailing lists, and so on. Since different mail systems vary in how they identify bounced, undeliverable, or otherwise invalid messages, you can use notification mailer tags to specify how your mail system identifies those stray messages and how you want the notification mailer to handle those messages should it come across them.

Oracle Workflow provides several predefined tags for text commonly found in undeliverable or auto-reply messages. The notification mailer handles messages according to these mail status values, as follows:

- UNDELVRD: Moves the message to the discard folder and updates the notification's mail status to FAILED. Additionally, the notification preference of the recipient of the notification is updated to DISABLED. No error process is initiated for this notification activity. However, after correcting the issues that prevented the e-mail from being sent, you can reset the user's notification preference and then run the Resend Failed Workflow Notifications program to re-enqueue failed notifications on the notification mailer's outbound queue.
- Unavailable: Moves the message to the discard folder and continues waiting for a reply to the notification since the notification's status is still OPEN, but its mail status is updated to UNAVAIL. This status is purely informative, as no further processing occurs with this notification.
- Ignore: Moves the message to the discard folder and continues waiting for a valid reply to the open notification. The notification's status is still OPEN and its mail status is still SENT.

You can define additional tags for other patterns you want the notification mailer to handle automatically. To add a new tag, click the Add Another Row button, enter the text pattern in the Pattern column, and select the status you want to assign to messages containing that pattern in the Action column.

Test Page

This page lets you test the notification mailer configuration by sending a sample notification message. Select the recipient role to which the message should be sent, and click the Send Test Message button. Then check the e-mail account for the recipient role to verify that the test message was received.

Review Page

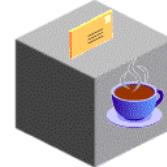
If you want to change any of these settings, return to the appropriate step in the notification mailer configuration wizard to make your changes. To save these settings and finish the configuration, click the Finish button.

Component Details for Notification Mailers

Component Details for Notification Mailers

The Component Details page shows the following additional buttons for notification mailers:

- **Test Mailer**
- **Set Override Address**



ORACLE

Component Details for Notification Mailers

Click the Test Mailer button to send a test message. On the Test Notification Mailer page, select the recipient role to which the message should be sent, and click the Send Test Message button. Then check the e-mail account for the recipient role to verify that the test message was received.

Click the Set Override Address button to set an override address where you want to send all outgoing e-mail notifications. Use an override address when you are testing workflow definitions or mailer processing so that you can automatically receive all the test notifications at one e-mail address, instead of having to check or change each individual recipient's e-mail address. To ensure that the override address is accessible and that its use is authorized, you must verify the request before the notification mailer can use the address.

- On the Set Override Address page, review the current override address, if any. Enter the e-mail address you want to set as the new override address, and select Submit. Then check the e-mail account that you specified for the verification e-mail message.
- On the Verify Override Address page, enter the verification code shown in the e-mail message, and click Apply. If necessary, you can use the link provided in the verification e-

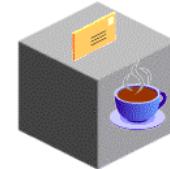
mail message to navigate back to the Verify Override Address page. You must log in to Oracle Applications Manager before you can access this page.

- To remove the override address, navigate to the Set Override Address page and click the Clear Override Address button. The notification mailer then resumes sending e-mail notifications to the individual recipients' e-mail addresses.

Notification Mailer Throughput

Notification Mailer Throughput

- On the Workflow System page, click the Notification Mailer link in the Throughput section of the Related Links region.
- The notification mailer throughput graph displays the distribution of notifications in the WF_NOTIFICATIONS table with these statuses:
 - Processed: An e-mail message has been sent by a notification mailer service component.
 - Waiting: An e-mail message has not yet been sent.

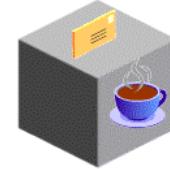


Handling Notification Mailer Errors

Handling Notification Mailer Errors

Oracle Workflow provides scripts, diagnostic tests, and concurrent programs to help handle notification mailer errors, including:

- **wfmlrdbg.sql script or Mailer Diagnostic Test:** Check the status of a particular notification or investigate errors.
- **Resend Failed Workflow Notifications program:** Resend e-mail notifications with a mail status of FAILED after correcting the issues that prevented the e-mails from being sent.



ORACLE®

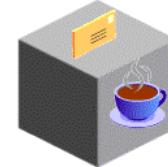
Handling Notification Mailer Errors

For additional information about error handling scripts, diagnostic tests, and concurrent programs, see: Handling Mailer Errors, *Oracle Workflow Administrator's Guide*.

Testing Mailer URL Access

Testing Mailer URL Access

- If a notification contains an embedded Oracle Application Framework region, the notification mailer fetches the Oracle Application Framework content from the Web tier.
- Use the Workflow Mailer URL Access Tester page to test whether that content can be generated correctly in the context that the notification mailer uses.



Testing Mailer URL Access

Before you can access the Workflow Mailer URL Access Tester page, your system administrator must add it to the menu for an Oracle E-Business Suite responsibility that is assigned to you.

You can test either the notification details for a single notification, or the notification summary of all open notifications for a particular role.

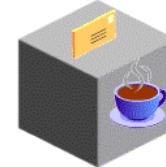
The notification mailer uses the context values specified in the Framework User, Framework Responsibility, and Framework Application ID parameters in the notification mailer configuration wizard. For example, the default values for these parameters are:

- Framework User - SYSADMIN (0)
- Framework Responsibility - System Administrator (20420)
- Framework Application ID - System Administration (1)

Testing Mailer URL Access

Testing Mailer URL Access

- In the Workflow Mailer URL Access Tester page, select the region type and parameters to test.
- Choose Go.
- Oracle Workflow displays a test URL to generate the notification content that you specified.
- Establish a browser session with the same context values as the notification mailer.
- In this browser session, navigate to the test URL and verify that the generated content appears correctly.



Testing Mailer URL Access

Navigate to the Workflow Mailer URL Access Tester page using a responsibility and navigation path specified by your system administrator.

You can test either a Notification Details region or a Notification Summary region.

- For a Notification Details region, specify the following parameters:
 - Notification ID - Specify the unique ID for the notification to test.
 - Content Type - Specify whether to test the notification content in HTML format or as plain text.
 - Language - Select the language in which to display the notification content.
- For a Notification Summary region, specify the following parameters:
 - Recipient Role - Select the type of role for which you want to test the notification summary. Then select the role you want within that type.
 - Content Type - Specify whether to test the notification summary content in HTML format or as plain text.

If the content generated by the test URL in your browser session does not appear correctly, adjust the code for your Oracle Application Framework region to ensure that it does not depend on the user session context.

Summary

Summary

In this lesson, you should have learned how to:

- **Configure and run agent listeners.**
- **Configure and run notification mailers.**

ORACLE®

Managing System Status and Throughput

Chapter 29

Unauthorized reproduction or distribution prohibited. Copyright© 2017, Oracle and/or its affiliates.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Managing System Status and Throughput

Managing System Status and Throughput

ORACLE®

Objectives

Objectives

After completing this lesson, you should be able to do the following:

- **Review Workflow system status information.**
- **Monitor work items in different statuses and purge completed work items.**
- **Run background engines.**
- **Run Workflow control queue cleanup.**
- **Review Business Event System queue propagation and agent activity details.**

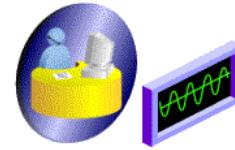
ORACLE

Workflow System Status

Workflow System Status

On the Workflow System page of Oracle Workflow Manager:

- **Select a feature status icon to manage that feature.**
 - **Background Engines**
 - **Purge**
 - **Control Queue Cleanup**
- **Monitor system activity using the Workflow Metrics graphs.**
 - **Work Items**
 - **Agent Activity**



ORACLE

Workflow System Status

The Workflow System page also includes the following:

System Status

You can also click the Notification Mailers, Agent Listeners, or Service Components status icons to manage those components.

Additionally, you can also select an Oracle Workflow feature that runs as a concurrent program from the Submit Request For pull-down menu and click the Go button. You can submit requests for background engines, purging, and control queue cleanup.

Related Database Parameters

This region lets you review actual and recommended values for the database initialization parameters that are required for Oracle Workflow.

Workflow Metrics

- Work Items: To view the distribution of item types within a status, either select the bar for that status in the graph, or click the status name link.

Note: A work item can be counted in more than one status. For example, all work items that do not have an end date are counted as Active work items, including deferred,

suspended, and errored work items as well as running work items. Also, if an activity within an item is deferred, and the work item as a whole is suspended, the work item is included in the count for both the Deferred and Suspended statuses. Consequently, the total of the counts for all the statuses is greater than the actual number of work items.

- Agent Activity: To view the distribution of event messages with different statuses on different agents, either select the bar for a status in the graph, or select an event message status name link.

Note: Messages are not explicitly assigned a status of Error. The Error bar in the graph represents messages of any status on the WF_ERROR agent.

Related Links

- Configuration
 - Click the Service Components link to configure service components, including notification mailers and agent listeners.
 - Click the Queue Propagation link to view a list of propagation schedules for Business Event System agents.
- Throughput
 - Click the Work Items link to view the distribution of completed work items across different item types.
 - Click the Notification Mailer link to view the notification mailer throughput.

Gathering Oracle Workflow Statistics

Some Oracle Workflow Manager graphs and lists may summarize large volumes of data, depending on the level of activity in your Oracle E-Business Suite instance. To enhance performance in displaying these statistics, Oracle Workflow Manager periodically runs concurrent programs to gather the statistics and displays the graphs and lists based on the latest data from the concurrent programs.

- Workflow Agent Activity Statistics program: Gathers statistics for the Agent Activity graph on the Workflow System status page and for the agent activity list on the Agent Activity page.
- Workflow Mailer Statistics program: Gathers statistics for the throughput graph on the Notification Mailer Throughput page.
- Workflow Work Items Statistics program: Gathers statistics for the Work Items graph on the Workflow System status page, for the Completed Work Items list on the Workflow Purge page, and for the work item lists on the Active Work Items, Deferred Work Items, Suspended Work Items, and Errored Work Items pages.

These concurrent programs are scheduled to run every 24 hours by default. They do not require any parameters. You can optionally cancel the default scheduled requests and run the programs with a different schedule if you want to gather statistics at a different frequency.

Each of these graphs and lists displays the date and time when its statistics were last updated, as well as a refresh icon that you can click to refresh the statistics immediately if necessary. However, note that if your Oracle E-Business Suite instance contains very large volumes of workflow data, you may encounter delays or page timeouts when refreshing the data.

Workflow Status in Oracle Applications Manager

Workflow Status in Oracle Applications Manager

You can also use other Oracle Applications Manager features to help manage Oracle Workflow.

- Use Oracle Diagnostics to run diagnostic tests that check the setup of your Oracle Workflow installation and review debugging information.
- Use Oracle Applications Logging to review Oracle Workflow logs.



Oracle Applications Manager

Diagnostics

Oracle Workflow provides the following diagnostic tests.

- Duplicate User Test: Checks the Oracle Workflow directory service to verify that there are no roles in the WF_LOCAL_ROLES table with the same internal name, originating system, and originating system ID.
- Notification Preference Validation Test: Checks the Oracle Workflow directory service to ensure that all roles with a notification preference for receiving e-mail notifications have an e-mail address defined.
- Rule Function Validation Test: Checks the rule functions defined for subscriptions and the generate functions defined for events in the Business Event System.
- GSM Setup Test: Checks the Generic Service Management (GSM) setup required for Oracle Workflow in Oracle Applications Manager.
- BES Clone Test: Checks certain standard agents and subscriptions required for internal Business Event System and notification mailer processing to verify that they are enabled and that their definitions include the correct local system.

- GSC Control Queue Test: Verifies that the Workflow control queue, WF_CONTROL, is properly accessible.
- Workflow Advanced Queue Rule Validation Test: Checks the standard WF_ERROR and WF_DEFERRED queues to verify that only one subscriber rule is defined for each queue.
- Workflow Agents/AQ Status Test: Checks the Business Event System agents for Oracle Workflow and Oracle XML Gateway, as well as the queues associated with these agents.
- Workflow Objects Validity Test: Checks the Oracle Workflow and Oracle XML Gateway database objects to verify that all the objects are valid.
- XML Parser Installation Test: Checks your Oracle E-Business Suite installation to verify that the Oracle XML parser is installed and valid.
- Mailer Component Test: Checks your notification mailer service components to verify that at least one notification mailer has been configured with all the parameters needed to run it.
- Mailer Component Parameter Test: Checks your notification mailer service components to validate their configuration parameters.
- Event Diagnostic Test: Reports details about the Business Event System, including the statuses of the local system and agent listeners, details about the definitions of the specified event and any subscriptions to that event, and details about the specified instance of the event if it appears on a standard deferred or error queue.
- Event Raise Test: Checks the basic operation of the Business Event System by raising a test event from Java and from PL/SQL and executing synchronous and asynchronous subscriptions to that event.
- Mailer Diagnostic Test: Reports details about a notification and about the notification mailer that sent the notification.

Logging

Oracle Workflow uses the Oracle Applications Logging framework to standardize and centralize in the database logging activities related to the Oracle Workflow Business Event System and Oracle XML Gateway.

Note: The Java middle tier components of Oracle Workflow, including notification mailers and agent listeners, also use Oracle Applications Logging; however, due to the high volume of messages that pass through these components, their information is logged to the file system by default.

Oracle Workflow Administration

Oracle Workflow Administration

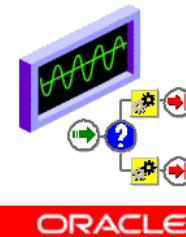
- **Work Items** ←
- Purging
- Background Engines
- Control Queue Cleanup
- Queue Propagation
- Agent Activity

ORACLE®

Work Items

Work Items

- Drill down from the Work Items graph on the Workflow System Status page to review work items in these statuses:
 - Active
 - Deferred
 - Suspended
 - Errorred
- The detail pages for these statuses are similar.



Work Items

Review the number of work items in different statuses to monitor Workflow Engine processing and identify any possible bottlenecks. For example, an abnormal number of activities with a deferred status may indicate that there are not enough background engines available.

Information about completed work items is available through the Workflow Purge page.

Work Items

Work Items

- **Work Items**
 - Review the number of work items of different item types that are in the selected status.
 - To drill down, select an item type and select View Details.
- **Work Item Details**
 - Review the number of work items of the selected item type that are currently at various activity stages within the workflow process.
 - To drill down, select an activity stage and select View Details.



Work Items

Work Items

To filter the item types displayed in the list, select an item type property and an operator from the Filter pull-down menus, enter a filter value in the text field, and click the Go button. You can filter by the following properties:

- Work item type display name
- Work item type internal name
- Number of work items of this type

Note: When you drill down to active work items, all work items that do not have an end date are counted as active work items in this page, including deferred, suspended, and errored work items as well as running work items.

Work Item Details

By default, the lists of active or deferred work items show only work items that started within the last 30 days, because there may be large numbers of work items in these statuses. However, the lists of suspended or errored work items show work items that started at any time, by default.

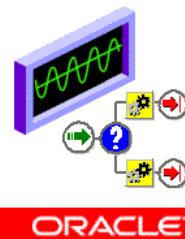
To view work items that started within a specific period, enter a number of days in the Filter: Start Date Within Last _ Days option and click the Go button.

Note: When you drill down to active work items, only activities with a status of Active, Waiting, or Notified are included on this page. Activities with a status of Deferred, Suspended, or Error are not included on this page, although the work items to which they belong are counted as active work items.

Work Items

Work Items

- **Work Item Activity Details**
 - **Review details about work item activities of the selected activity stage within the selected item type.**
 - **Use the administrative buttons to manage work items. Depending on the work item status, you can abort, suspend, resume, or retry work items.**
 - **Select an activity and select Launch Workflow Monitor to review the work item in the Status Monitor.**



Work Items

Work Item Activity Details

By default, the lists of active or deferred work items show only work items that started within the last 30 days, because there may be large numbers of work items in these statuses. However, the lists of suspended or errored work items show work items that started at any time, by default.

To filter the work items displayed in the list, select an activity property from the Filter By pull-down menu, enter a filter value in the text field, and click the Go button. You can filter by the following properties:

- Internal name of the work item activity
- Start date within a specified number of days
- Due date within a specified number of days
- User assigned to perform the activity
- Item key of the work item

Note: When you drill down to active work items, only activities with a status of Active, Waiting, or Notified are included on this page. Activities with a status of Deferred, Suspended,

or Error are not included on this page, although the work items to which they belong are counted as Active work items.

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.

Oracle Workflow Administration

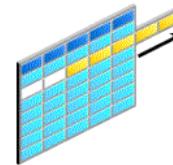
- Work Items
- **Purging** ←
- Background Engines
- Control Queue Cleanup
- Queue Propagation
- Agent Activity

ORACLE

Purging Workflow Data

Purging Workflow Data

- Oracle Workflow tables can grow quite large with obsolete workflow run-time information that is stored for all completed workflow processes.
- The size of these tables and indexes can adversely affect performance.
- You should purge these tables on a regular basis, using:
 - Purging in Oracle Workflow Manager
 - Purge APIs



ORACLE

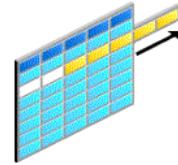
Purging Workflow Data

Note: The Purge APIs are provided to let developers and administrators perform manual purging when necessary. However, it is recommended that you submit purge requests from Oracle Workflow Manager because this tool centralizes workflow management functionality and lets you review the completed work items eligible for purging before you perform the purge.

Purging Workflow Data

Purging Workflow Data

- **Oracle Workflow Manager lets you review completed work items to determine what data is available for purging.**
- **To perform purging, submit the Purge Obsolete Workflow Runtime Data concurrent program.**



ORACLE

Purging Workflow Data

The persistence type of an item type controls when runtime status information for work items of that type becomes eligible for purging. The persistence values are:

- Temporary: Item will be eligible to be deleted in n days
- Permanent: Item will be deleted only when forced

Note: For a work item to be considered eligible for purging, all activities must be complete. This requirement includes FYI notifications, which must be closed.

This program purges obsolete runtime information associated with work items, including status information, any associated notifications, and, if the ECX: Purge ECX data with WF profile option is set to Y, any associated Oracle XML Gateway transactions. By default, it also purges obsolete design information, such as activities that are no longer in use and expired ad hoc users and roles, and obsolete runtime information not associated with work items, such as notifications that were not handled through a workflow process and, if the ECX: Purge ECX data with WF profile option is set to Y, Oracle XML Gateway transactions that were not handled through a workflow process. You can optionally choose to purge only core runtime information associated with work items for performance gain during periods of high activity,

and purge all obsolete information as part of your routine maintenance during periods of low activity.

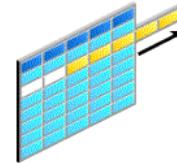
To preserve electronic signature evidence for future reference, this program by default does not delete any notifications that required signatures or their associated signature information. If you do not need to maintain signature evidence, you can choose to delete signature-related information as well.

Note: You can also use the Purge Obsolete ECX Data concurrent program to purge Oracle XML Gateway transactions according to Oracle XML Gateway-specific parameters. See: Purge Obsolete ECX Data Concurrent Program, *Oracle XML Gateway User's Guide*.

Purging Workflow Data

Purging Workflow Data

- On the Workflow System page, select the Work Items link in the Throughput region.
- Review the next scheduled and last completed purge requests.
- Review completed work items.
- To submit a request for the Purge Obsolete Workflow Runtime Data concurrent program:
 - Click the Purge button.
 - Enter the purge parameters and other concurrent request options.
- To review purge concurrent request details, click the View Purge Requests button.



ORACLE

Purging in Oracle E-Business Suite

You can also access the Workflow Purge page from the Workflow System page by selecting the Purge status icon.

You can also submit a request for the Purge Obsolete Workflow Runtime Data concurrent program by selecting Purge from the Submit Request For pull-down menu on the Workflow System page and clicking the Go button.

Parameters

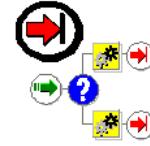
- Item Type: Specify the item type to purge. Leave this field blank to purge the run-time data for all item types.
- Item Key: Specify the item key to purge. The item key is a unique identifier for an item within an item type. Leave this field blank to purge the run-time data for all items of the specified item type.
- Age: Specify the minimum age of data to purge, in days, if you are purging items with a Temporary persistence type. The default is 0 days.
- Persistence Type: Specify the persistence type of the data you want to purge, either Permanent or Temporary. The default is Temporary.

- Core Workflow Only: Enter Y to purge only obsolete run-time data associated with work items, or N to purge all obsolete run-time data as well obsolete design data. The default is N.
- Commit Frequency: Enter the number of records to purge before the program commits data. To reduce rollback size and improve performance, set this parameter to commit data after a smaller number of records. The default is 500 records.
- Signed Notifications: Enter N to preserve signature evidence, including notifications that required electronic signatures and their associated signature information. Enter Y to purge signature-related information. The default is N.

Completed Work Items

Completed Work Items

- **Completed Work Items**
 - Review the number of completed work items of different item types and the number of work items available for purging.
 - To drill down, select an item type and select View Details.
- **Completed Work Item Details**
 - Review the number of work items of the selected item type that ended at various activity stages within the workflow process.
 - To drill down, select an activity stage and select View Details.



ORACLE

Completed Work Items

Note: These pages show information for completed work items that have not yet been purged. Work items that have already been purged will no longer appear, because their information is no longer stored in the Oracle Workflow tables.

Completed Work Items

To filter the item types displayed in the list, select an item type property and an operator from the Filter pull-down menus, enter a filter value in the text field, and click the Go button. You can filter by the following properties:

- Work item type display name
- Work item type internal name
- Persistence type
- Retention period
- Number of completed work items of this type
- Number of items of this type available for purging

You can also drill down by clicking the item type link in the Work Item Type column.

Completed Work Item Details

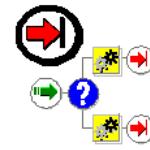
By default, the list shows unpurged, completed work items that ended within the last 30 days, because there may be large numbers of work items in this status. To view unpurged, completed work items that ended within a different period, enter a number of days in the Filter: End Date Within Last _ Days option and click the Go button.

You can also drill down by clicking the activity stage link in the Work Item Activity Stage column.

Completed Work Items

Completed Work Items

- **Completed Work Item Activity Details**
 - **Review details about completed work items of the selected item type that ended at the selected activity stage.**
 - **Select an activity and choose Launch Workflow Monitor to review the work item in the Status Monitor.**



ORACLE

Completed Work Items

Note: This page shows information for completed work items that have not yet been purged. Work items that have already been purged will no longer appear, because their information is no longer stored in the Oracle Workflow tables.

Completed Work Item Activity Details

By default, the list shows unpurged, completed work items that ended within the last 30 days, because there may be large numbers of work items in this status.

To filter the work items displayed in the list, select an activity property from the Filter pull-down menu, enter a filter value in the text field, and click the Go button. You can filter by the following properties:

- Internal name of the activity at which the work item ended
- Start date within a specified number of days
- End date within a specified number of days
- User assigned to perform the activity
- Item key of the work item

Workflow Purge APIs

Workflow Purge APIs

- **Purge APIs delete obsolete run-time and design data.**
- **The most commonly used WF_PURGE APIs include:**
 - **Items:** Purge all run-time data associated with completed items, their processes, and notifications sent by them.
 - **Activities:** Purge obsolete design versions of activities that are no longer in use by any item.
 - **Total:** Purge both run-time and design data.
 - **Directory:** Purge expired users and roles in the Workflow local tables that are not referenced in any notification.



ORACLE

Workflow Purge APIs

Many of the purge APIs in the WF_PURGE package accept the following parameters:

- Item Type: The item type associated with the obsolete run-time data that you want to delete. Leave this parameter null to delete obsolete data for all item types.
- Item Key: The string generated from the application object's primary key that uniquely identifies the item within an item type. Leave this parameter null to purge all items in the specified item type.
- End Date: A specified date to delete up to.

Note: Most of the WF_PURGE APIs only purge data associated with Temporary item types whose persistence, in days, has expired. Use the WF_PURGE.TotalPERM API to delete all eligible obsolete run-time data associated with item types of with a persistence type of Permanent.

Oracle Workflow Administration

Oracle Workflow Administration

- Work Items
- Purging
- **Background Engines** ←
- Control Queue Cleanup
- Queue Propagation
- Agent Activity

ORACLE®

Background Engines

Background Engines

- **Background engines handle:**
 - Activities deferred by the Workflow Engine
 - Timed out activities
 - Stuck processes
- **To run a background engine, submit the Workflow Background Process concurrent program.**



Background Engines

You can set up as many background engines as you need, but if you set up only one, then that background engine must handle both deferred and timed out activities as well as stuck processes.

Generally, you should run a separate background engine to check for stuck processes at less frequent intervals than the background engine that you run for deferred or timed out activities, normally not more often than once a day. Run the background engine to check for stuck processes when the load on the system is low. Do not run more background engines concurrently than your server has CPU processors.

Background Engines

Background Engines

- To submit a request for the Workflow Background Process concurrent program:
 - Select Background Engine from the Submit Request For pull-down menu on the Workflow System page and click Go.
 - Enter the background engine parameters and other concurrent request options.
- To review Workflow Background Process concurrent requests, click the Background Engines status icon on the Workflow System page.



Background Engines

When you start a new background engine, you can restrict the engine to handle activities associated with specific item types, and within specific cost ranges. You can submit the Workflow Background Process concurrent program several times to schedule different background engines to run at different times.

Parameters

- Item Type: Specify an item type to restrict this engine to activities associated with that item type. If you do not specify an item type, the engine processes any activity regardless of its item type.
- Minimum Threshold: Specify the minimum cost that an activity must have for this background engine to execute it, in hundredths of a second.
- Maximum Threshold: Specify the maximum cost that an activity can have for this background engine to execute it, in hundredths of a second. By using Minimum Threshold and Maximum Threshold you can create multiple background engines to handle very specific types of activities. For example, if you only want to process activities with a certain cost, you can set both the Minimum Threshold and the Maximum Threshold to that

value. The default values for these arguments are 0 and 100 so that the background engine runs activities regardless of cost.

- Process Deferred: Specify whether this background engine checks for deferred activities.
- Process Timeout: Specify whether this background engine checks for activities that have timed out.
- Process Stuck: Specify whether this background engine checks for stuck processes.

Oracle Workflow Administration

- Work Items
- Purging
- Background Engines
- **Control Queue Cleanup** ←
- Queue Propagation
- Agent Activity

ORACLE

Control Queue Cleanup

Control Queue Cleanup

- Oracle Workflow uses the WF_CONTROL queue for internal processing.
 - When a middle tier process for Oracle Workflow starts up, it creates a subscriber to the WF_CONTROL queue.
 - You must periodically clean up the queue by removing inactive subscribers.
- To perform control queue cleanup, submit the Workflow Control Queue Cleanup concurrent program.



ORACLE

Control Queue Cleanup

The recommended frequency for performing cleanup is every 12 hours.

Control Queue Cleanup

Control Queue Cleanup

- To submit a request for the Workflow Control Queue Cleanup concurrent program:
 - Select Control Queue Cleanup from the Submit Request For pull-down menu on the Workflow System page and click Go.
 - Enter the concurrent request options.
- To review Workflow Control Queue Cleanup concurrent requests, click the Control Queue Cleanup status icon on the Workflow System page.



ORACLE

Control Queue Cleanup

The Workflow Control Queue Cleanup concurrent program is automatically scheduled to run every 12 hours by default. You can optionally submit this program with a different schedule. This program does not require any parameters.

Oracle Workflow Administration

Oracle Workflow Administration

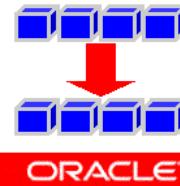
- Work Items
- Purging
- Background Engines
- Control Queue Cleanup
- **Queue Propagation** ←
- Agent Activity

ORACLE®

Queue Propagation

Queue Propagation

- **Review database initialization parameters required for queue propagation.**
- **Review the list of propagation schedules for local outbound agents.**
- **Select a propagation schedule and choose View Details to review detailed information about that schedule.**



Queue Propagation

You should schedule propagation for your local outbound Business Event System agents to send event messages to their destinations. A propagation schedule is defined for an outbound queue and a specified destination, which can be either a remote database link or the local system. If you want to use the standard WF_OUT and WF_JMS_OUT agents or custom agents for event message propagation, ensure that you schedule propagation for those agents.

Note: You do not need to schedule propagation for the WF_CONTROL or WF_NOTIFICATION_OUT agents, however, because the middle tier processes that use WF_CONTROL dequeue messages directly from its queue, and a notification mailer sends messages placed on the WF_NOTIFICATION_OUT queue.

The Queue Propagation page shows information to let you determine the status of a propagation schedule, including the job queue process executing the schedule, whether the schedule is enabled or disabled, and the error date and error message of the last unsuccessful execution. For example, if no process is allocated to execute the schedule, you may need to increase the JOB_QUEUE_PROCESSES database initialization parameter to ensure that processes are available for propagation. If the propagation schedule is disabled, you must enable it before it can be executed.

Oracle Workflow Administration

- Work Items
- Purging
- Background Engines
- Control Queue Cleanup
- Queue Propagation
- Agent Activity

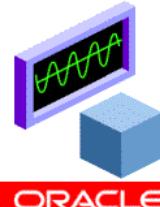


ORACLE

Agent Activity

Agent Activity

- **Review the distribution of event messages with different statuses on different local agents:**
 - Ready
 - Waiting
 - Processed
 - Expired
 - Undeliverable
- **Click a link in the Agent column to review details about that agent's queue.**
- **Select an agent and click the Search Agent Entry Details button to view messages on the agent.**



Agent Activity

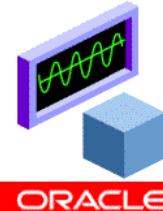
Review the number of messages in different statuses on your local agents to monitor event message processing and identify any possible bottlenecks. For example, if an inbound agent has an abnormally large number of messages with a status of Ready, you may need to check the status of the agent listener processing message for that agent, or configure a new agent listener service component for that agent. Similarly, if an outbound agent has an abnormally large number of messages with a status of Ready, you may need to check the status of the propagation schedule for that agent's queue, or schedule propagation if necessary.

Note: The Agent Activity page displays event messages on the WF_ERROR agent according to their explicitly assigned status on the WF_ERROR queue, unlike the Agent Activity graph in the Workflow System page which summarizes all messages on the WF_ERROR agent in an Error status.

Searching Messages on an Agent

Searching Messages on an Agent

- In the Search Queue page:
 - Enter filter criteria to locate the messages you want.
 - Click the Go button.
 - Review message details.
 - Click the icon in the View XML column to review the event data for the message as an XML document.
- You can search queues with these payload types:
 - WF_EVENT_T
 - SYS.AQ\$_JMS_TEXT_MESSAGE



Searching Messages on an Agent

The icon in the View XML column is disabled if the event data for a message is empty.

You can also search Oracle XML Gateway queues with these payload types:

- SYSTEM.ECXMSG
- SYSTEM.ECX_INENGOBJ

For the Oracle XML Gateway payload types, the Search Queue page provides different filter criteria and message details that are specific to Oracle XML Gateway messages.

Summary

Summary

In this lesson, you should have learned how to:

- **Review Workflow system status information.**
- **Monitor work items in different statuses and purge completed work items.**
- **Run background engines.**
- **Run Workflow control queue cleanup.**
- **Review Business Event System queue propagation and agent activity details.**

ORACLE®

Revathi Ramamoorthy (revathi.b.r@capgemini.com) has a
non-transferable license to use this Student Guide.