# Oracle® Application Integration Architecture

Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide

Release 2.5

**E17563-03**

February 2012

ORACLE

Oracle Application Integration Architecture Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide, Release 2.5

E17563-03



Primary Author: Oracle Corporation

# Contents


Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide Preface ..... 7

Oracle Application Integration Architecture – Foundation Pack 2.5: Getting Started with the Oracle AIA Foundation Pack and Demo ..... 7

Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide ..... 8

Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide ..... 8

Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide ..... 9

Oracle Application Integration Architecture Process Integration Packs ..... 9

Additional Resources ..... 10

What's New in This Guide for Release 2.5 IPS3 ..... 11

Understanding the Order to Bill Process Integration Pack ..... 15

Part 1: Implementing the Process Integration for Product Lifecycle Management ..... 17

Chapter 1: Understanding the Process Integration for Product Lifecycle Management ..... 19

Product Lifecycle Management Overview ..... 19

Simple and Customizable Products ..... 26

Synchronizing Billing Products and Billing Discounts ..... 28

Solution Assumptions and Constraints ..... 38

Oracle BRM Interfaces ..... 40

Siebel CRM Interfaces ..... 40

Industry AIA Components ..... 40

Integration Services ..... 41

Chapter 2: Understanding the Product Bundling Methodology ..... 47

Basic Entity Mappings ..... 47

Defining Products and Discounts in Oracle BRM ..... 48

Physical Goods ..... 49

Sales Catalogs ..... 50

Recommendations for Product Definition in Siebel CRM ..... 50

Service Bundles ..... 52

Simple Service Bundles ..... 57

Marketing Bundles ..... 60
Balance Groups ..... 61
Credit Limits ..... 61
Promotion Penalty, Service Activation, and Other MACD Charges ..... 61
Supporting Friends and Family ..... 62
Supporting Time-Based Offerings ..... 68
Chapter 3: Configuring the Process Integration for Product Lifecycle Management ..... 71
Setting Up FMW ..... 71
Setting Up Oracle BRM ..... 72
Setting Up Siebel CRM ..... 74
Working with DVMs ..... 75
Working with Cross-References ..... 76
Handling Error Notifications ..... 77
Viewing EBO EIMs ..... 77
Configuring the Process Integration for Product Lifecycle Management ..... 77
Part 2: Implementing the Process Integration for Order Management ..... 81
Chapter 4: Understanding the Process Integration for Order Management ..... 83
Overview ..... 83
Business Process Flow Overview ..... 84
Data Requirements ..... 85
Submitting Orders to Order Orchestration ..... 86
Creating Customer Data in Billing ..... 87
Interfacing Orders to Billing ..... 92
Supporting Time-Based Offerings ..... 105
Supporting Friends and Family Lists ..... 106
Sending Order Updates Back to Siebel CRM ..... 108
Creating or Updating Installed Assets in Siebel CRM ..... 110
Integration Flows ..... 111
Orchestrating the Services ..... 125
Solution Assumptions and Constraints ..... 128
Oracle BRM Interfaces ..... 130
Siebel CRM Interfaces ..... 131
Industry AIA Components ..... 131
Integration Services ..... 132
Chapter 5: Configuring the Process Integration for Order Management ..... 147

Setting Up FMW ..... 147
Setting Up Oracle BRM ..... 149
Setting Up Siebel CRM ..... 149
Working with DVMs ..... 150
Working with Cross-References ..... 152
Handling Error Notifications ..... 154
Viewing EBO EIMs ..... 155
Configuring the Process Integration for Order Management ..... 155
Part 3: Implementing the Process Integration for Customer Management ..... 165
Chapter 6: Understanding the Process Integration for Customer Management ..... 167
Overview ..... 167
Solution Assumptions and Constraints ..... 167
Data Requirements ..... 168
Create/Sync Customer Account Integration Flow ..... 169
Update Customer Account Integration Flow ..... 175
Oracle BRM Interfaces ..... 180
Siebel CRM Interfaces ..... 181
Industry AIA Components ..... 182
Integration Services ..... 183
Chapter 7: Configuring the Process Integration for Customer Management ..... 193
Setting Up FMW ..... 193
Setting Up Oracle BRM ..... 194
Setting Up Siebel CRM ..... 195
Working with DVMs ..... 195
Working with Cross-References ..... 196
Handling Error Notifications ..... 198
Viewing EBO EIMs ..... 200
Configuring the Process Integration for Customer Management ..... 200
Part 4: Implementing the Process Integration for Order Fallout Management ..... 209
Chapter 8: Understanding the Process Integration for Order Fallout Management ..... 211
Overview ..... 211
Business Process Task Flow ..... 213
Solution Assumptions and Constraints ..... 218
Creating Additional Listeners to Capture Order Failure Notifications ..... 218
Creating Trouble Tickets in Siebel CRM by Oracle AIA ..... 224

Siebel CRM Interfaces ..... 226
Industry AIA Components ..... 226
Integration Services ..... 227
Order to Bill Fallout Services ..... 229
Chapter 9: Configuring the Process Integration for Order Fallout Management ..... 231
Setting Up Oracle AIA ..... 231
Setting Up FMW ..... 231
Working with DVMs ..... 233
Working with Cross-References ..... 234
Handling Error Notifications ..... 234
Viewing EBO EIMs ..... 237
Configuring the Process Integration for Order Fallout Management ..... 238
Appendix A: Order Management: Matrix of MACD Actions Supported Per Billing Product Type ..... 241
Table A ..... 241
Table B ..... 244
Appendix B: Examples of Changing the Paying Parent on Subordinate Accounts ..... 251
Appendix C: Order Fallout: Guidelines for Ensuring That Oracle AIA Processes Are Compliant ..... 257
Populating Sender Context Information in the EBM Header ..... 257
Populating the Enriched Fault Message in Case of Business Faults ..... 258
Populating the Enriched Fault Message in Services without EBMs ..... 262
Appendix D: Cross-References for the Process Integration for Product Management ..... 265
Cross-Reference Values ..... 265
Integration Solution Cross-References ..... 266
Product Synchronization Flow ..... 267
Discount Synchronization Flow ..... 276
Appendix E: Configuring Multiple Instances of Oracle BRM ..... 279
Understanding System Codes in Oracle AIA ..... 279
Adding an Additional Oracle BRM Instance – General Steps ..... 281
Receiving Customer or Billing Profile Updates and Friends and Family List Updates from Siebel CRM ..... 282
Adding a New Oracle BRM Instance for the Friends and Family List Update Flow ..... 292
Adding a New Oracle BRM instance: Configure Product Sync ..... 297
Adding a New Oracle BRM Instance: Add Routing Rules for Agent Assisted Billing Care PIP ..... 319

Appendix F: Reconfiguring AIA for Comms ..... 327
Changing the Oracle BRM Instance ..... 327
Appendix G: Using the Resequencer Feature of ESB ..... 329
Resolving Errors in Flows with Resequencer ..... 330
Appendix H: Using Session Pool Manager ..... 333
Configuring Session Pool Manager ..... 333
Appendix I: Mapping Billing Dates ..... 335
Index ..... 341

# Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide Preface

This preface discusses:

- Oracle Application Integration Architecture – Foundation Pack 2.5: Getting Started with the Oracle AIA Foundation Pack and Demo
- Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide
- Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide
- Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide
- Oracle Application Integration Architecture Process Integration Packs
- Additional resources

**Note:** Oracle Application Integration Architecture – Foundation Pack 2.5 guides can be found in My Oracle Support (MOS) Article ID: 955605.1.

## Oracle Application Integration Architecture – Foundation Pack 2.5: Getting Started with the Oracle AIA Foundation Pack and Demo

The *Oracle Integration Architecture – Foundation Pack 2.5: Getting Started with Oracle AIA Foundation Pack and Demo* provides information about how Oracle Application Integration Architecture Foundation Pack offers great opportunities to build state-of-the-art service-oriented architecture (SOA) integrations.

This guide is targeted mainly at integration architects who want to extend or adjust prebuilt integrations shipped by Oracle, or who plan to build new SOA-based integrations based on Application Integration Architecture. These concepts and components are demonstrated using the Application Integration Architecture Foundation Pack Demo. This demo is the Application Integration Architecture adaptation of the SOA Order Booking Demo.

# Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide

The *Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide* provides conceptual, setup, and usage information for the following Core Infrastructure Components:

- Business Service Repository (BSR).
- Composite Application Validation System (CAVS).
- Error handling and logging.
- The Diagnostics Framework.

# Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide

The *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide* is a companion volume to the *Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide* and *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide.* The *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide* provides definitions of fundamental Oracle Application Integration Architecture (Oracle AIA) concepts and discusses:

- Oracle AIA.
- Enterprise business objects (EBOs) and enterprise business messages (EBMs).
- Enterprise business services (EBSs).
- Application business connector services (ABCSs).
- Interaction patterns.
- Extensibility.
- Versioning.
- Business processes.
- Batch processing.
- Infrastructure services.
- Security.

# Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide

The *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide* is a companion volume to *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide* and *Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide*.

The *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide* discusses how to:

- Create an integration scenario.
- Define business service patterns.
- Design and develop EBSs.
- Design and develop enterprise business flows.
- Design and construct ABCSs.
- Work with message transformation, enrichment, and configuration.
- Develop custom xpath functions.
- Design and construct Java Message Service (JMS) Adapter services.
- Work with enterprise message headers.
- Work with message routing.
- Work with transactions.
- Develop AIA services to work with the CAVS.
- Configure Oracle AIA processes to be eligible for error handling and logging.
- Extend EBOs.

In addition, this book provides Oracle AIA naming standards.

# Oracle Application Integration Architecture Process Integration Packs

A process integration pack (PIP) is a prebuilt set of integrated orchestration flows, application integration logic, and extensible enterprise business objects and services required to manage the state and performing of a defined set of activities or tasks between specific Oracle applications associated with a given process. A PIP provides everything that you need to deploy a selected integrated business process area. The PIP product offering is suited to those customers seeking to rapidly implement a discrete business process.

# Additional Resources

This table lists available resources:

| Resource | Location |
|---|---|
| Installation Guide | My Oracle Support |
| Documentation updates | My Oracle Support |
| Release Notes | Oracle Technology Network http://www.oracle.com/technology/ |
| Known issues, workarounds, and current list of patches | My Oracle Support |

# What's New in This Guide for Release 2.5 IPS3

For release 2.5 IPS3, this guide has been updated in several ways. The following table lists the sections that have been added or changed.

| Sections | Changes Made |
|---|---|
| Chapter 2: Using the Product Bundling Methodology | |
| Section: Using Fixed Amounts versus Scaled Amounts in Oracle BRM | Section revised to document the case where both the scaled amount and the fixed amount are specified for the product. |
| Section: Promotion Penalty, Service Activation, and other MACD Charges | Section revised to include steps about how to model service activation for a service.<br>Note: Specific examples have also been updated for service activation. |
| Chapter 3: Configuring the Process Integration for Product Lifecycle Management | |
| Section: Working with Cross-References | Section revised to update the list of product management cross-references. Also included a link to *Appendix D: Cross-References for the Product Integration for Product Management*. |
| Chapter 4: Understanding the Process Integration for Order Management | |
| Section: Supporting Single Phase versus Two-Phase Billing (Billing Initiation and Billing Fulfillment), Initiate Billing Mode | Modeling and Implementation Recommendations section revised to document the default value when the property *FutureTimeThreshold* is not specified for a billing instance. |
| Section: Assumptions and Constraints for Time-Based Offerings | The assumptions and constraints for time-based offerings have been revised. |
| Section: Modifying Friends and Family Lists | Section added to document how customers can use the Special Rating Profile user interface (UI) to make chages to their lists. |

| Sections | Changes Made |
|---|---|
| Section: Integration Flows | A new section, Defining Transaction Boundaries and Recovery Details, has been added for each of the following integration flows. These sections document transactions involved, the database operations, and what actions to take in case of errors.<br>• Submitting Orders to the Order Management System<br>• Interfacing Orders to Create Customer Data in Oracle BRM<br>• Interfacing Orders to Create Transaction Data in Oracle BRM<br>• Updating Status Information on Order Lines in Siebel CRM<br>• Synchronizing Friends and Family List Updates to Oracle BRM |
| Section: Oracle BRM Services | Section has been revised to change the name of the opcode PCM_OP_CUSTOMER_UPDATE_SERVICE to PCM_OP_CUST_UPDATE_SERVICES |
| Section: Integration Services – CommunicationsBillingResponseEBSV1 | Section has been revised to document how, for error scenarios, a response message can be optionally sent back to the order management system. |
| Chapter 6: Understanding the Process Integration for Customer Management | |
| Section: Create/Sync Customer Account Integration Flow | Section has been revised with the following changes:<br>• The overall flow diagram for the create/sync customer account integration flow has been updated.<br>• A new table providing information on Siebel attributes mapped to Oracle BRM as part of the create/sync account integration flow has been added to replace the two tables in this chapter that discussed Siebel entities.<br>• The sequence diagram illustrating the create/sync account integration flow has been updated.<br>• A new section, Defining Transaction Boundaries and Recovery Details, has been added the Create/Sync Account integration flow. This section documents transactions involved, the database operations, and what actions to take in case of errors |
| Section: Update Customer Account Integration Flow | Section has been revised with the following changes:<br>• An overview of the flow has been added.<br>• The overall flow diagram for the update customer account integration flow has been updated. |
| Chapter 7: Configuring the Process Integration for Customer Management | |

| Sections | Changes Made |
|---|---|
| Section: Working with Cross-References | The name of the cross-reference CUSTOMERPARTY_ACCOUNTDEFAULTBALGRP has been changed to CUSTOMERPARTY_DEFAULTBALANCEGROUPID |
| Appendix B: Examples of Changing the Paying Parent on Subordinate Accounts | |
| Legend | The legend has been revised so that it describes all the abbreviations used in the examples. |
| Section: Example #3 | Example #3 has been revised with the following:<br>• This example has changed from being not supported to supported (with caveats)<br>• Scenario has been revised to reflect optimization changes. |

# Understanding the Order to Bill Process Integration Pack

In today's highly competitive communications industry, the rapid convergence of wireless and wire line services, prepaid and postpaid services, and IT and network all present challenges communications service providers (CSPs) to rapidly orchestrate streamlined processes across front-office and back-office applications and networks.

Oracle Communications Billing and Revenue Management: Order to Bill Process Integration Pack (Order to Bill PIP) helps with this challenge by providing CSPs efficient, accurate, order-to-bill business processes and consistent product definitions between Siebel Customer Relationship Management (Siebel CRM) and Oracle Communications Billing and Revenue Management (Oracle BRM).

By integrating Siebel CRM with the robust rating, billing, payment, and collections capabilities of Oracle BRM, the Order to Bill PIP provides a single view of the entire order-to-bill business process.

Additionally, with the Order to Bill PIP, systems can automatically create and update customer information including accounts, addresses, contacts, and billing profiles. As a result, a complete, accurate, and synchronized customer view is achieved, as well as enhanced customer visibility across CRM and Oracle BRM.

The Order to Bill PIP automates the order management process between Siebel CRM and Oracle BRM, including product and price synchronization, customer synchronization, and order processing.

The Order to Bill PIP provides the following process integrations:

- Product Lifecycle Management.
- Order Management.
- Customer Management.
- Order Fallout Management.

The process integration for product lifecycle management enables you to synchronize and administer products and discounts between Oracle BRM and Siebel CRM.

The process integration for order management enables the submission of orders from Siebel CRM to your order management system for order fulfillment. Additionally, your order management system can call the services provided by this integration to:

- Interface orders to create customer data in Oracle BRM.
- Interface orders to create transaction data in Oracle BRM.
- Update Siebel CRM with the status and other information on the order.

Note that a central fulfillment system (CFS), order management system, or order decomposition and orchestration process is not delivered in the Order to Bill PIP. However, you can interoperate this PIP with the Oracle Order to Activate PIP, which uses Oracle Communications Order and Service Management (Oracle OSM) for order orchestration and fulfillment.

**For more information** about using Oracle OSM as your orchestration process, see the *Oracle Communications Order and Service Management Application Integration Architecture Order-to-Activate Cartridge Guide*.

The process integration for customer management enables the synchronization of customer information between Siebel CRM and Oracle BRM. Customers are created in Siebel CRM and sent to Oracle BRM. Customer updates are allowed on both the Siebel CRM and Oracle BRM systems. When updates occur in one system, they are then synchronized with the other system.

The process integration for order fallout management enables you to implement a detection and notification process to handle order failures. Order fallout management uses Siebel trouble ticketing for notification and tracking of order failures.

The Order to Bill PIP is built on top of the Oracle AIA Communications Foundation Pack. Communications customers can easily leverage Oracle AIA Communications Foundation Pack to extend the delivered process integrations and build new ones.

# Part 1: Implementing the Process Integration for Product Lifecycle Management

This part includes the following chapters:

Chapter 1: Understanding the Process Integration for Product Lifecycle Management

Chapter 2: Understanding the Product Bundling Methodology

Chapter 3: Configuring the Process Integration for Product Lifecycle Management

# Chapter 1: Understanding the Process Integration for Product Lifecycle Management

This chapter provides an overview of product lifecycle management and discusses:

- Simple and customizable products.
- Synchronizing billing products and billing discounts.
- Solution assumptions and constraints.
- Oracle Communications Billing and Revenue Management (Oracle BRM) interfaces.
- Siebel Customer Relationship Management (Siebel CRM) interfaces.
- Industry Application Integration Architecture (AIA) components.
- Integration services.

## Product Lifecycle Management Overview

The process integration for product lifecycle management enables you to synchronize and administer (in real time or batch mode) billing products and billing discounts between Oracle BRM and Siebel CRM. Oracle BRM is the master for billing products and billing discounts. Creation of or updates to billing products and billing discounts occur in Oracle BRM.

The process integration for product lifecycle management delivers these integration points:

- Synchronization of billing products: Oracle BRM to Siebel CRM. The product synchronization integration flow enables you to create new products in Oracle BRM and synchronize those products in Siebel CRM. It also enables you to update existing products in Oracle BRM and then synchronize the updated products in Siebel CRM.
- Synchronization of billing discounts: Oracle BRM to Siebel CRM. The discount synchronization integration flow enables you to create discounts as products in Oracle BRM and replicate those discounts in Siebel CRM. It also enables you to update existing discounts in Oracle BRM and then synchronize the updated discounts in Siebel CRM.

Time-zone handling: Oracle AIA does not do time-zone conversion when synchronizing Oracle BRM products and discounts to Siebel. The Oracle BRM Enterprise Application Integration (EAI) property infranet.eai.date_pattern controls which time-zone Oracle BRM publishes datetime information in. As delivered, this property is not set and Oracle BRM publishes datetime information in the Oracle BRM local server time zone. If you set this property, then Oracle BRM will publish the datetime information in UTC/GMT time zone.

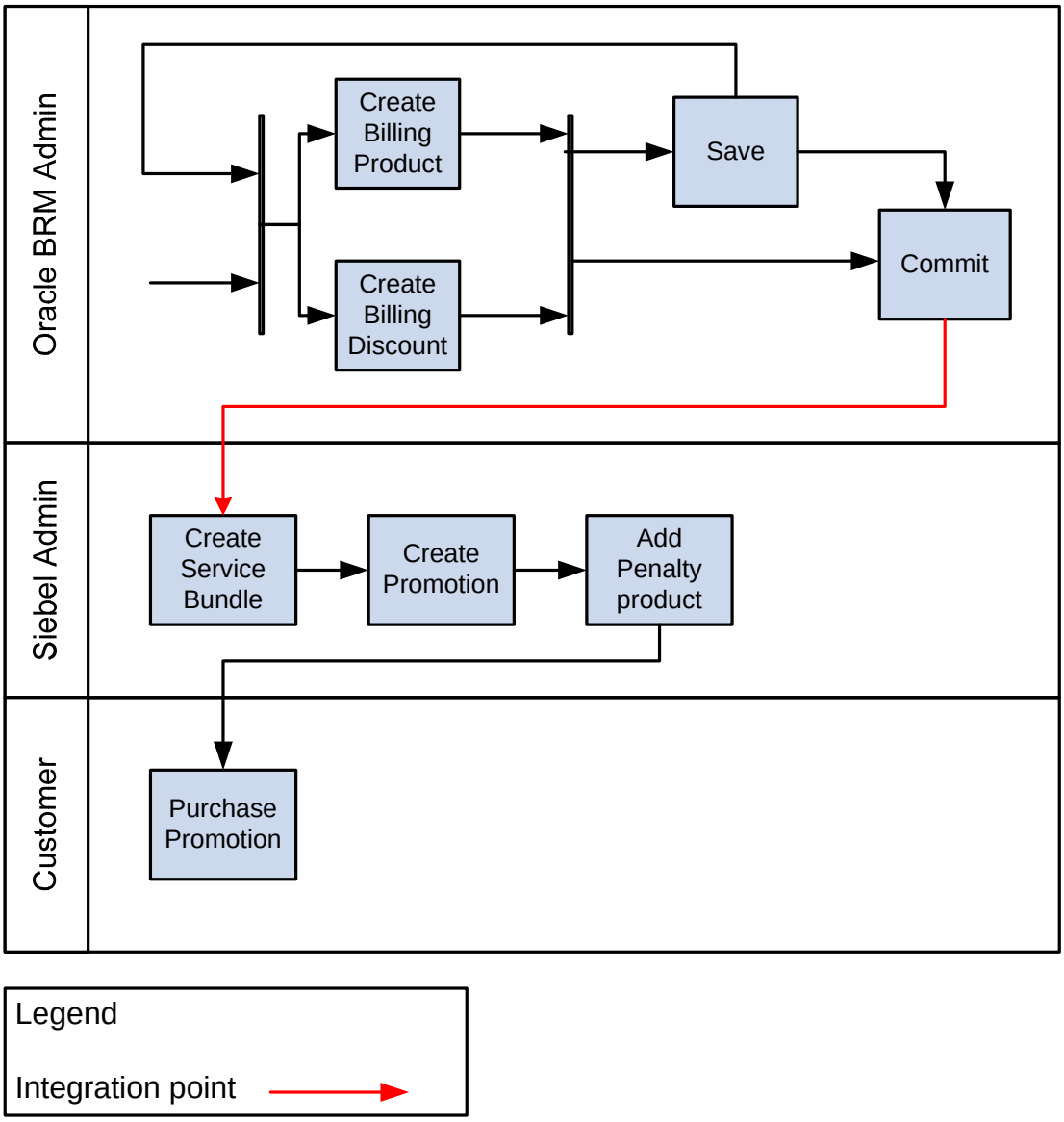
**For more information** about setting this property, see the Oracle BRM documentation (Oracle BRM patch set 3 release notes).

## Real-Time Billing Product and Billing Discount Synchronization

In this flow, the Oracle BRM administrator creates billing products and billing discounts in the Oracle BRM Pricing Center. After a new billing product or billing discount is created, the administrator can commit it to the Oracle BRM database. Alternatively, the Oracle BRM administrator can create a set of billing products and billing discounts and save them in a file. After all of the billing products and billing discounts have been created in the file, the administrator commits them to the Oracle BRM database. This instantaneously synchronizes the new billing products or billing discounts to Siebel CRM. The Siebel CRM administrator uses these billing products to create service bundles or promotions. The Siebel CRM administrator can also add charges, such as penalties, to the promotion. After the promotions are created, customers can purchase the promotions.

**For more information,** see Chapter 2: Understanding the Product Bundling Methodology.

This diagram shows the business process flow for synchronization of real-time billing products and billing discounts

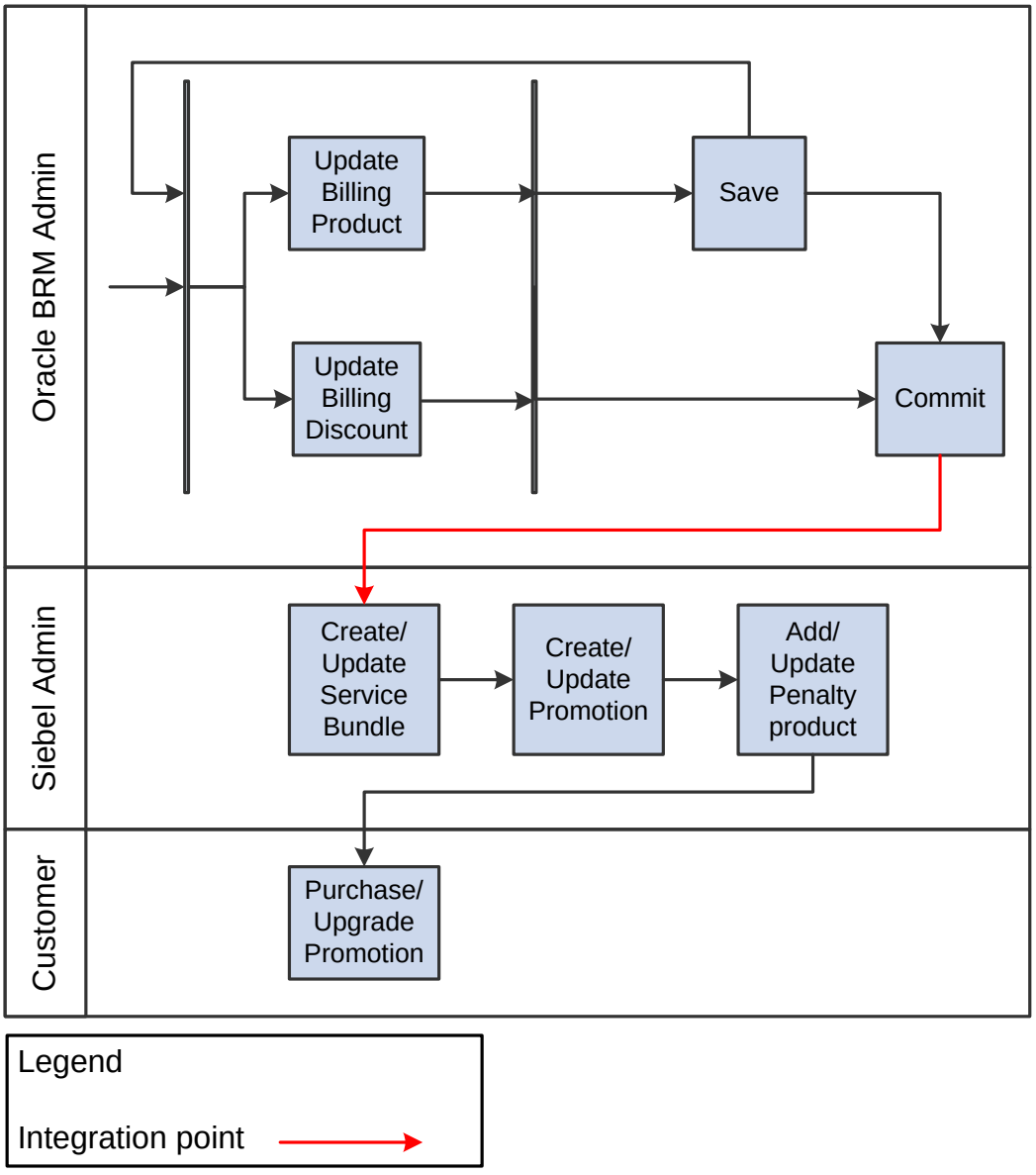



Flow for synchronization of real-time billing products and billing discounts

## Update Real-Time Billing Product and Billing Discount Synchronization

In this flow, whenever changes occur to a billing product or a billing discount attribute, the Oracle BRM administrator can update the billing products and billing discounts in the Oracle BRM Pricing Center and commit them to the Oracle BRM database. Alternatively, the Oracle BRM administrator can update a set of billing products or billing discounts and save all of them in a file. After all of the billing products and billing discounts have been updated in the file, the administrator can commit them to the Oracle BRM database. This instantaneously synchronizes the updates to Siebel CRM. The service bundles and the promotions in Siebel CRM are updated to use the latest version of the billing products. The Siebel administrator makes any necessary changes in Siebel if required. Customers who purchase the promotions receive the latest promotions.

This diagram shows the business process flow for synchronization of update real-time billing products and billing discounts:



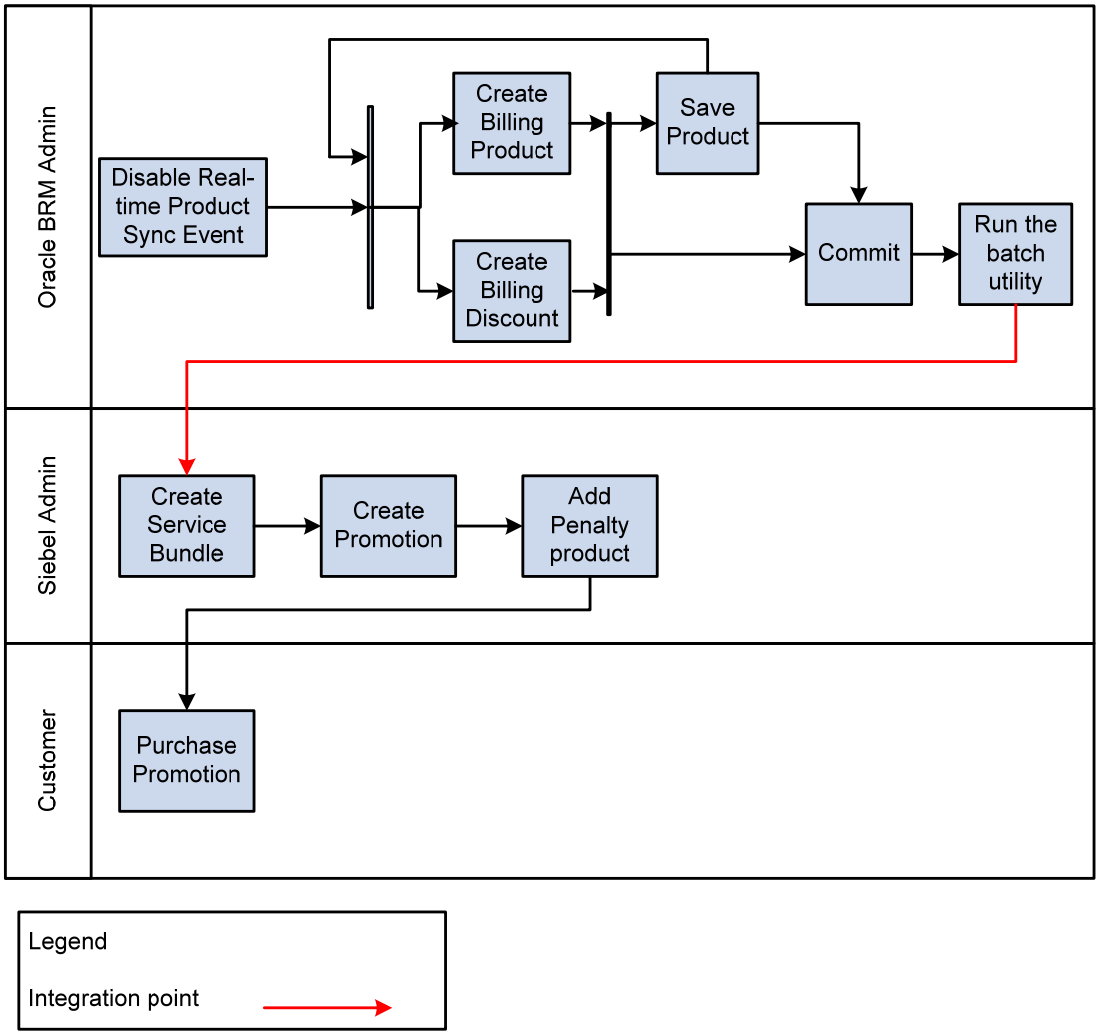
Flow for synchronization of update real-time billing products and billing discounts

## Batch Billing Product and Billing Discount Synchronization

In this flow, the Oracle BRM administrator disables the event for real-time product synchronization, and then creates a set of billing products and billing discounts. The administrator runs a batch utility to store the products in the Oracle BRM database and synchronize the products with Siebel CRM. Alternatively, the Oracle BRM administrator can create a set of products and save all of them in a file. After all of the billing products and billing discounts are created, the Oracle BRM administrator runs the batch utility. The Siebel administrator uses these billing products and billing discounts to create service bundles and promotions. The Siebel administrator can also add charges, such as penalties, to the promotion. After promotions are created, customers can purchase the promotions.

**For more information,** see Chapter 2: Understanding the Product Bundling Methodology.

This diagram shows the business process flow for synchronization of batch billing products and billing discounts:




## Flow for synchronization of batch billing products and billing discounts

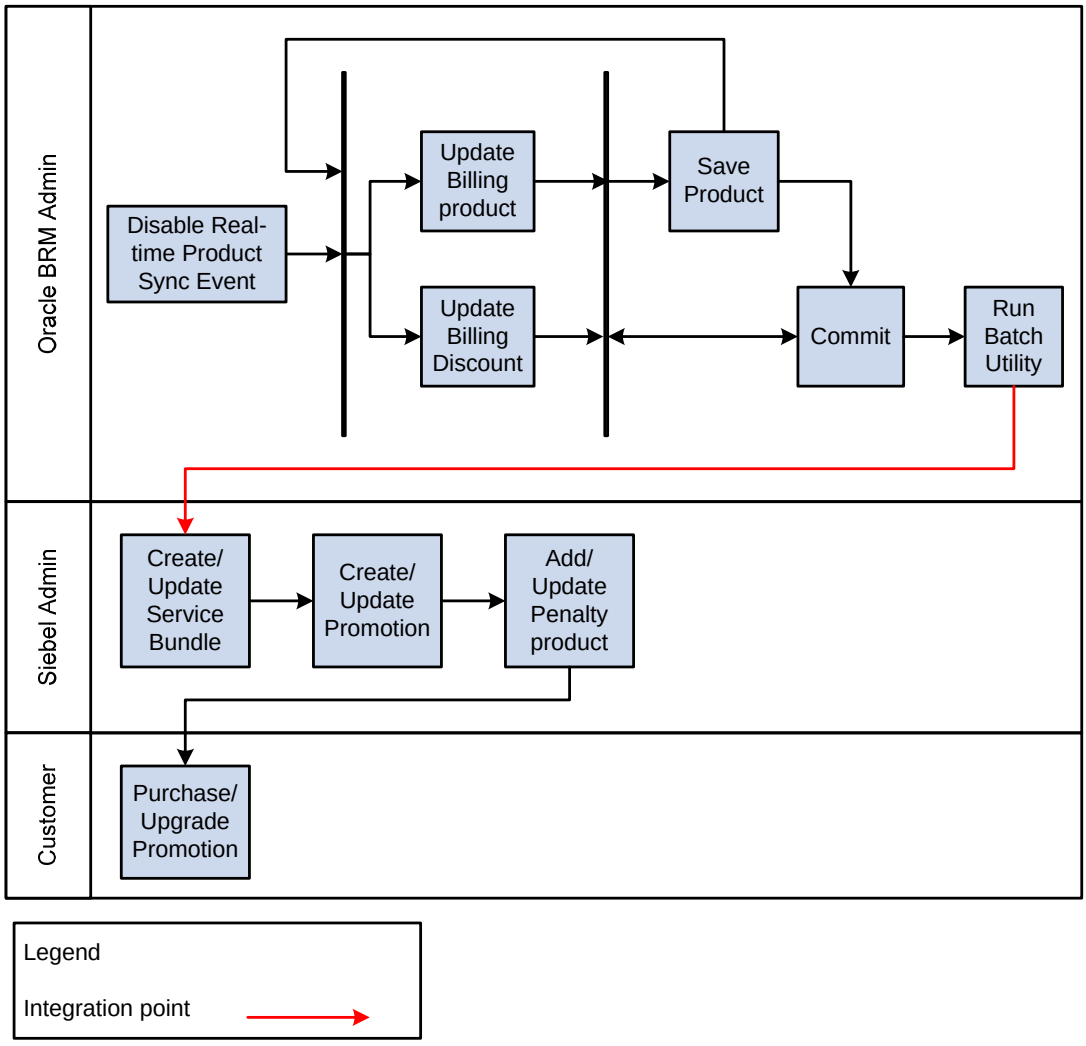
To disable the event for real-time product synchronization, see the Oracle BRM documentation.

**For more information,** see *Oracle Communications Billing and Revenue Management (BRM) Documentation*, “Service Integration Components,” Synchronization Queue Data Manager, Installing and configuring the Synchronization Queue DM, Starting and stopping the Synchronization Queue DM.

## Update Batch Billing Product and Billing Discount Synchronization

In this flow, the Oracle BRM administrator disables the event for real-time product synchronization. Whenever changes are made to the products or discount attributes, the Oracle BRM administrator updates billing products and billing discounts in the Oracle BRM Pricing Center. The administrator runs a batch utility to store the updates in the Oracle BRM database and synchronize them with Siebel CRM. Alternatively, the Oracle BRM administrator can update a set of billing products and billing discounts and save all of them in a file. After all of the billing products and billing discounts are updated, the Oracle BRM administrator runs the batch utility. The service bundles and the promotions in Siebel CRM are updated to use the latest version of the billing products and billing discounts. The Siebel administrator makes any necessary changes in Siebel if required. Customers who purchase the promotions receive the latest promotions.

This diagram shows the business process flow for synchronization of update batch billing products and billing discounts:




### Flow for synchronization of update batch billing products and billing discounts

To disable the event for real-time product synchronization, see the Oracle BRM documentation.

**For more information,** see *Oracle Communications Billing and Revenue Management (BRM) Documentation*, "Service Integration Components," Synchronization Queue Data Manager, Installing and configuring the Synchronization Queue DM, Starting and stopping the Synchronization Queue DM.

## Prerequisites

These are the prerequisites for the process integration for product management:

1. Oracle BRM must be set up before you can create billing products. The following pricing objects and data must be created in the Oracle BRM database:
   - Services.
   - Events.
   - Resources.
   - Currency exchange rates.
   - G/L IDs.
   - Tax codes and tax suppliers.
   - Ratable usage metrics (RUMs).
2. A product or price administrator defines billing products in Oracle BRM and associates them with billing events and billing rate plans.
3. Oracle BRM triggers an event that synchronizes the defined billing products with Siebel. The synchronization in this step is based on functional events available in Oracle BRM to identify changes (additions, deletions, modifications) that will trigger the integration flow to propagate those billing product changes and make the corresponding changes to Siebel CRM billing products.
4. A product or price administrator or product marketing manager defines service bundles in Siebel CRM to group all corresponding billing products.
5. A product marketing manager defines promotions in Siebel.

# Simple and Customizable Products

When products are created in Oracle BRM, they are associated with events that determine how much and how often to charge customers. These events are called billable events. Each product that is created in Oracle BRM is associated with one or more billable events. After the products are synchronized with Siebel CRM, the products that are associated with a single event are synchronized as simple products and products that are associated with multiple events are synchronized as customizable products.

This table shows how products are synchronized to Siebel:

| In Oracle BRM | In Siebel CRM |
|---|---|
| Internet<br>Monthly Cycle Forward Event - $25<br>Product Purchase Fee Event - $30 | Internet - $25<br>Internet Purchase - $30 |

| In Oracle BRM | In Siebel CRM |
|---|---|
| Delayed Telcom GSM Session Event - 0.40 | |

## Usage Charges on Products

Note the following exceptions.

If a billing product in Oracle BRM has Delayed Telco GSM Session as the only event, then the billing product is synchronized with Siebel as a simple product with no pricelist line created in Siebel.

For example, in this table, Delayed Telco GSM is an only event:

| Product in Oracle BRM | Simple Product in Siebel CRM |
|---|---|
| Wireless Usage<br>Delayed Telco GSM Session Event - 0.40 | Wireless Usage |

If a billing product in Oracle BRM has two events and one of them is Delayed Telco GSM Session, then the billing product is synchronized with Siebel CRM as a simple product. The Delayed Telco GSM Session event is not synchronized with Siebel. The list price of the simple product in Siebel is set to charge on the other event of the billing product.

For example, in this table, Delayed Telco GSM is one of two events:

| Product in Oracle BRM | Simple Product in Siebel CRM |
|---|---|
| Call Forwarding<br>Monthly Cycle Forward Event - $3.00<br>Delayed Telco GSM Session Event - $0.40 | Call Forwarding - $3.00 |

The billing product is synchronized with Siebel as a customizable product if a billing product in Oracle BRM has more than two events and one of the events is Delayed Telco GSM Session. The Delayed Telco GSM Session event is not synchronized with Siebel. The list price of the simple product in Siebel is set to charge on one of the other events of the billing product.

For example, in this table, Delayed Telco GSM is one of more than two events:

| Product in Oracle BRM | Customizable Product in Siebel CRM |
|---|---|
| Internet<br>Product Purchase Fee Event - $10.00<br>Monthly Cycle Forward Event - $20.00<br>Delayed Telco GSM Session Event - $0.40 | Internet - $20.00<br>Product Purchase Fee Event - $10.00 |

The solution is delivered with the following events mapped:

| Event Name | Event Definition |
|---|---|
| Product Purchase Fee Event (Activation) | "/event/billing/product/fee/purchase" |

| Event Name | Event Definition |
|---|---|
| Product Cancel Fee Event | "/event/billing/product/fee/cancel" |
| Monthly Cycle Arrear Event | "/event/billing/product/fee/cycle/cycle forward arrear" |
| Monthly Cycle Forward Event | "/event/billing/product/fee/cycle/cycle forward monthly" |
| Bimonthly Cycle Forward Event | "/event/billing/product/fee/cycle/cycle forward bimonthly" |
| Quarterly Cycle Forward Event | "/event/billing/product/fee/cycle/cycle forward quarterly" |
| Annual Cycle Forward Event | "/event/billing/product/fee/cycle/cycle forward annual" |
| Cycle Forward Arrear Event | "/event/billing/product/fee/cycle/cycle arrear" |

You can add more events in the PRICETYPE_EVENT domain value map. Events that are not present in this mapping will not be synchronized.

**For more information,** see Working with DVMs.

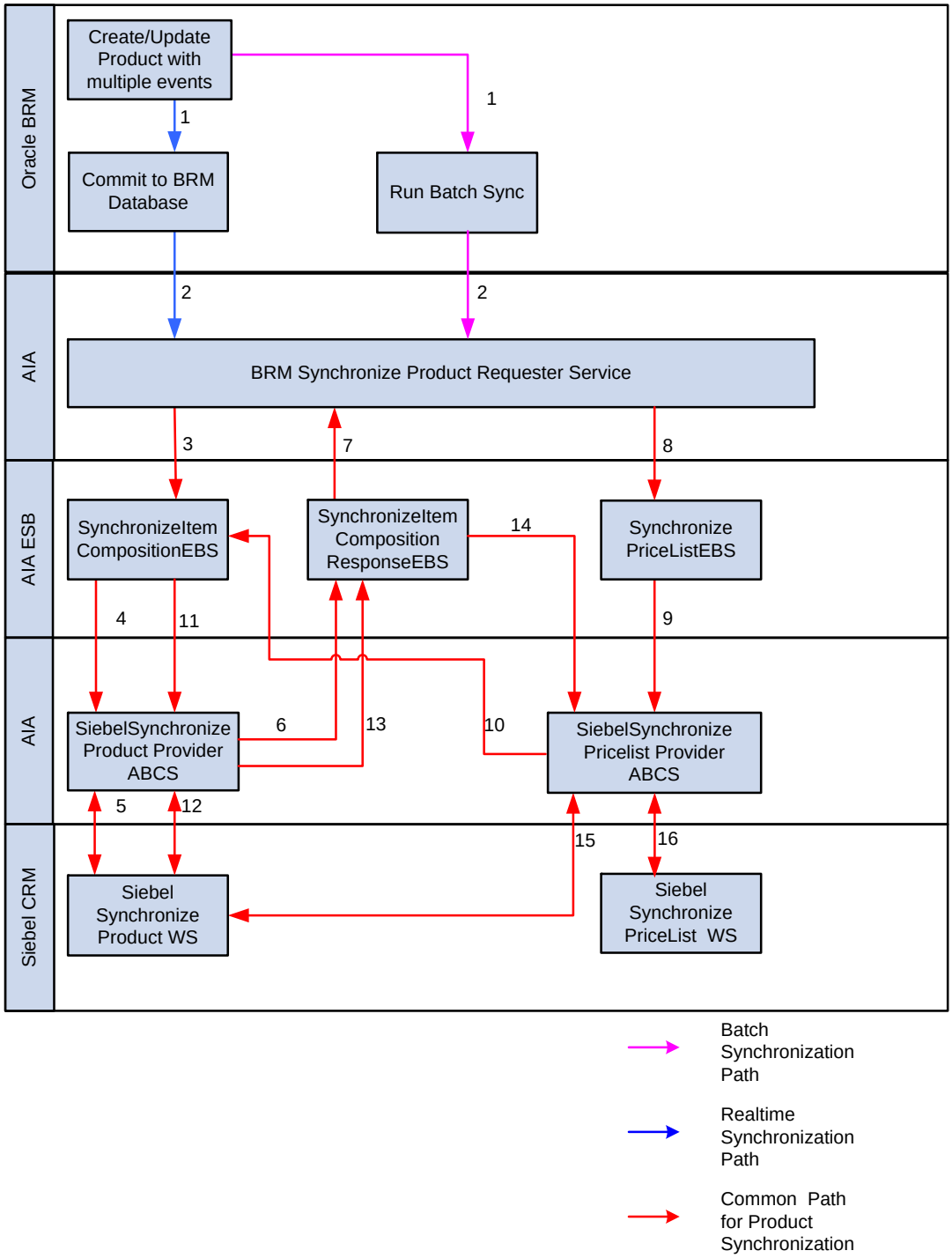
# Synchronizing Billing Products and Billing Discounts

This section describes:

- Synchronization of billing products functional overview.
- Product attributes
- Synchronization of billing discounts functional overview.
- Customizable discounts
- Synchronization of billing products technical overview.
- Synchronization of billing discounts technical overview.

## Synchronization of Billing Products Functional Overview

This diagram shows the synchronization of billing products with multiple events:




Synchronizing billing products with multiple events

For this flow, the following events occur:

1. You create billing products (single-event multi-event) in the BRM Pricing Center tool. When the new products are created they are synchronized to the target Siebel CRM by either realtime or batch synchronization.
2. The products are committed to the Oracle BRM database and realtime synchronization is invoked (or a batch utility is executed to synchronize the products as a batch). When the realtime or batch synchronization of billing products is invoked, a business event is raised in the Oracle BRM application, which also has the complete definition of the products (ProductABM).
3. The connector service (BRM Synchronize Product Requester), which is subscribed to this business event takes the input (ProductABM) and extracts all the product-related details and transforms them into a standardized representation of the product (ItemCompositionListEBM). The service invokes an enterprise business service (EBS) and provides the ItemCompositionListEBM as the input.
4. The SynchronizeItemCompositionEBS is configured with routing rules to each of the target application instances for which the product definition is published. The service routes the message to the Siebel application-specific connector service (Siebel Synchronize Product Provider).
5. The Siebel Synchronize Product Provider service transforms the standardized product definition (ItemCompositionListEBM) to a Siebel application-specific definition of the product. It invokes the Siebel application web services to create the products in the Siebel application. The status of the web service call (Success or Fail) is returned back to the caller service (Siebel Synchronize Product Provider).
6. + 7. The Siebel Synchronize Product Provider service processes the status and sends the details to the Host application connector service (BRM Synchronize Product Requester) using a standardized response message (ItemCompositionResponseEBM), which uses a response SynchronizeItemCompositionResponseEBS.
8. Once the products are successfully created, the BRM Synchronize Product Requester service extracts the pricing information from the billing products and transforms them into a standardized representation of the pricing (PriceListEBM). The service invokes the EBS and provides the PriceListEBM as input.
9. The SynchronizePriceListEBS is configured with routing rules to each of the target application instances for which the product definition is published. The service routes the message to the Siebel application-specific connector service (Siebel Synchronize Pricelist Provider).
10. The Siebel Synchronize Pricelist Provider service transforms the standardized pricelist definition (PriceListEBM) to theSiebel application-specific definition of the pricing. If there is more than one charge type associated with the pricing (Events) then simple products are created in the target CRM for each charge type. The pricing related to the charge types are assigned to the corresponding simple product. To create simple products, the connector service transforms the charge types (Events) into a standardized representation of the items (ItemCompositionListEBM) and invokes the SynchronizeItemCompositionEBS.
11. The SynchronizeItemCompositionEBS is configured with routing rules to each of the target application instances for which the product definition is published. The service routes the message to the Siebel application-specific connector service (Siebel Synchronize Product Provider).

12. The Siebel Synchronize Product Provider service transforms the standardized product definition (ItemCompositionListEBM) to a Siebel application-specific definition of the product. It invokes the Siebel application web services to create the simple products for each charge type in the Siebel application. The status of the web service call (Success or Fail) is returned back to the caller service (Siebel Synchronize Product provider).
13. + 14. The Siebel Synchronize Product Provider service processes the status and sends the details to the caller Siebel Synchronize PriceList Provider service using a standardized response message (ItemCompositionResponseEBM), which uses a response SynchronizeItemCompositionResponseEBS.
15. The Siebel Synchronize PriceList Provider service updates the simple products created earlier with the pricing attributes of the product (Price Type) by invoking the Siebel product creation web service. The status of the web service call (Success or Fail) is returned back to the caller service (Siebel Synchronize PriceList Provider).
16. The Siebel Synchronize PriceList Provider service updates the pricelist for all products with the actual pricing information (List Price, Effectivity, and so on) associated with the products. The status of the web service call (Success or Fail) is returned to the caller service (Siebel Synchronize PriceList Provider).

### Setting the Billable Flag for Products in Siebel CRM

During the product synchronization from Siebel CRM to Oracle BRM, the billable flag is set for all products of billing type Subscription. The billable flag is not set for products of billing type Event.

For service bundles, promotions, and simple products of billing type Special Rating, the billable flag must be manually set in Siebel CRM.

**For more information** about setting the billable flag in Siebel, see the *Siebel Communications Guide,* "Profiles in Siebel Communications."

## Product Attributes

These product attributes are included for all the products in the XML message that is sent to Siebel:

- Product Name
- Product Type
- Purchase Level
- Description
- Billable Events
- Rate Plan
- Effective Start Date and Effective End Date

Rate plan details (charges) go into the price list line while the remaining attributes go into the product.

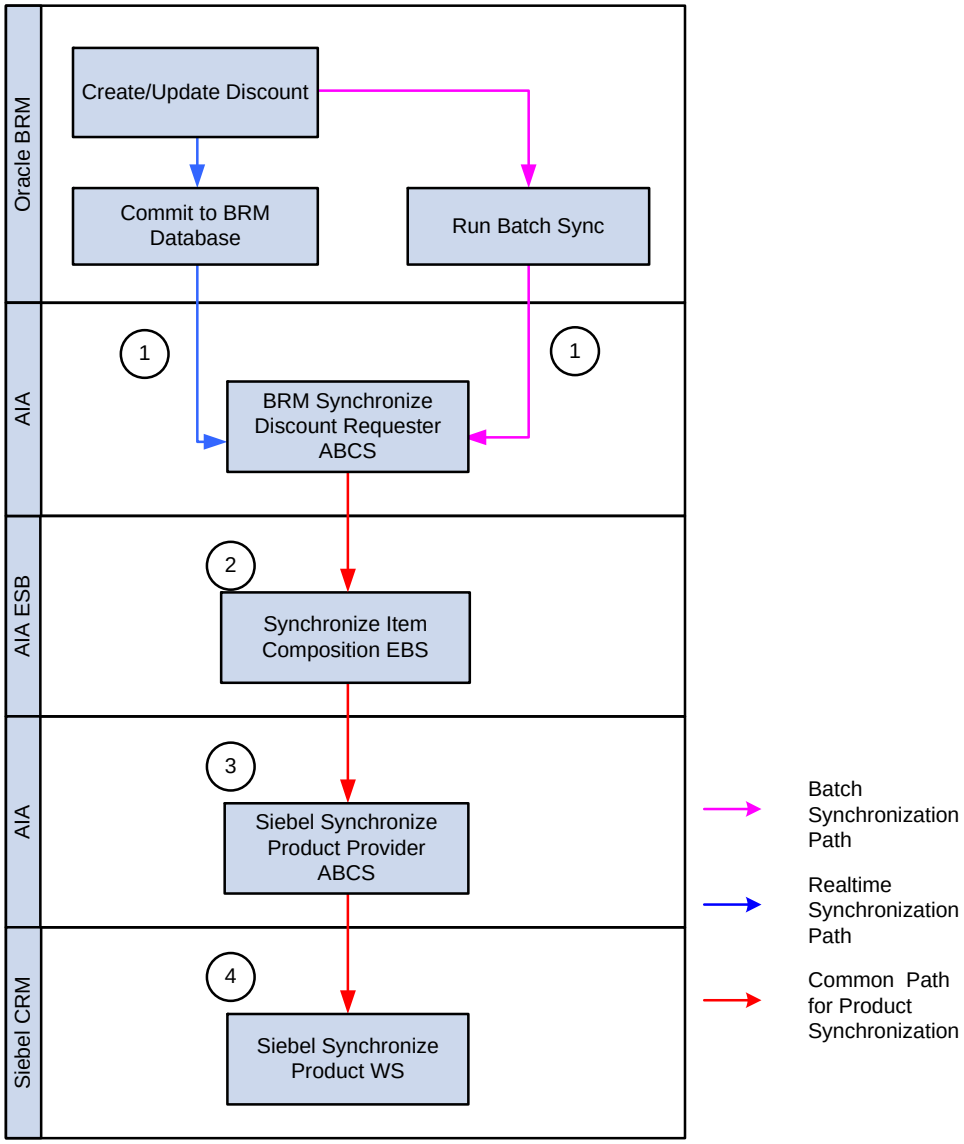
## Effective Start and End Dates

Effective Start Date - Oracle BRM publishes products or discounts with 12/31/1969 (or 01/01/1970 depending on the BRM EAI parameter setting) as the start date for cases where the product is effective immediately. The process integration sends this date as is as part of product or discount synchronization to Siebel CRM.

Effective End Date - Oracle BRM publishes products or discounts with 12/31/1969 (or 01/01/1970 depending on the BRM EAI parameter setting) as the end date for cases where the product never expires. To avoid interpretation of such products as expired in the target application in such cases, the process integration clears the end date before synchronizing the products or discounts to Siebel CRM.

## Synchronization of Billing Discounts Functional Overview

This diagram shows the synchronization of billing discounts:




### Synchronizing discounts flow

For this flow, the following events occur:

1. You create billing discounts in the BRM Pricing Center tool. Once the products are created, they are synchronized to the target Siebel CRM either realtime or using a batch synchronization. The products are committed to the Oracle BRM database and realtime synchronization is invoked. (A batch utility must be executed to synchronize the discounts as a batch). When the realtime or batch synchronization is invoked, a business event is raised in the Oracle BRM application, which also has the complete definition of the discount (DiscountABM).

2. The connector service (BRM Synchronize Discount Requestor) that is subscribed to this business event takes the input DiscountABM and extracts all the discount related details and transforms them into a standardized representation of the discount (ItemCompositionListEBM). The service invokes the enterprise business service (EBS) and provides the ItemCompositionListEBM as the input.
3. The SynchronizeItemCompositionEBS is configured with routing rules to each of the target application instances for which the discount definition is published. The service routes the message to the Siebel application specific connector service (Siebel Synchronize Product Provider). The discounts are created as simple products in Siebel.
4. The Siebel Synchronize Product Provider service transforms the standardized discount definition (ItemCompositionListEBM) to a Siebel application-specific definition of the product. It invokes the Siebel application web services to create the products in the Siebel application that corresponds to the discount that is published from Oracle BRM. The status of the web services call (Success or Fail) is returned back to the caller (Siebel Synchronize Product Provider service).

## Synchronization of Billing Products Technical Overview

The product synchronization integration flow enables you to create new products in Oracle BRM and then synchronize those products in Siebel CRM. The products created are used by the Order Capture and Asset Tracking modules in Siebel CRM. The product synchronization integration flow also enables updates to existing products in Oracle BRM. The updates are then synchronized in Siebel CRM.

When products are created in Oracle BRM, those products have multiple events, each with a price, which differs from Siebel CRM, which has only one product and a price for that product.

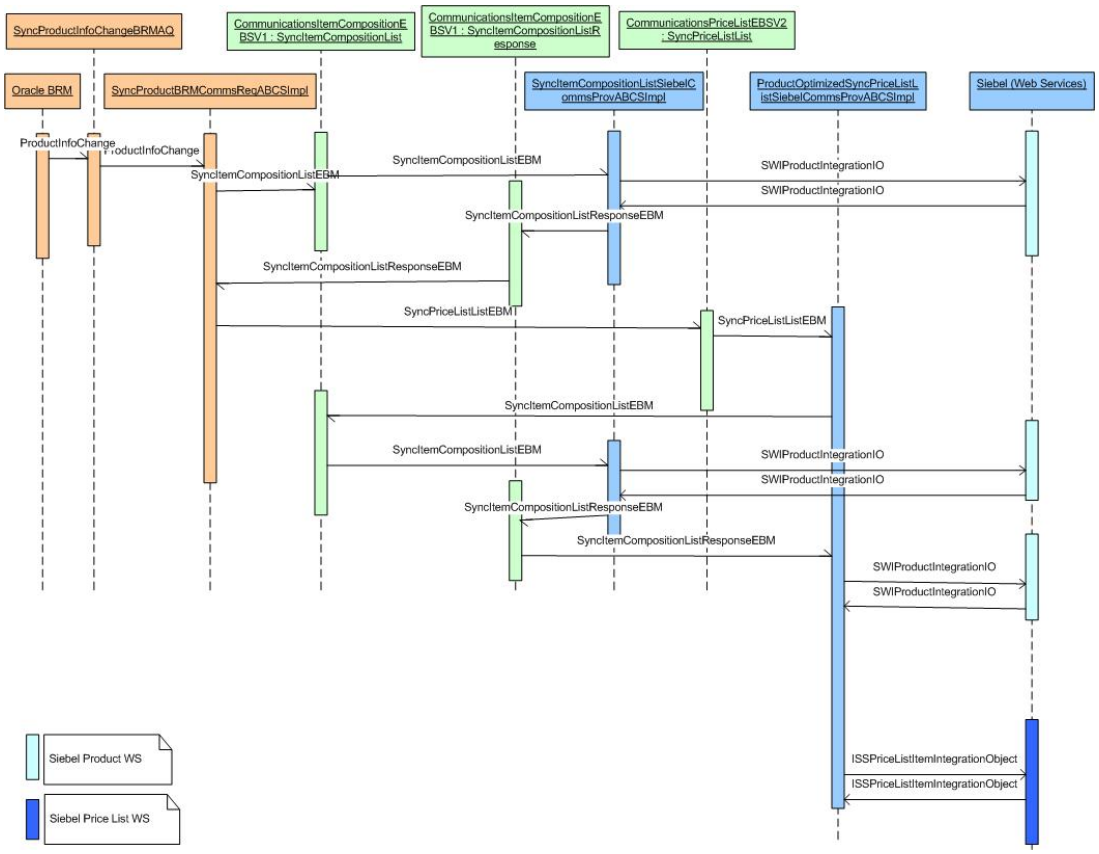
The product synchronization integration flow takes multiple events with recurring prices first. If no recurring price event exists, then the integration takes the first event and makes that the main (parent) product with a price.

After synchronizing the product data from Oracle BRM to Siebel CRM, Siebel returns a message containing both the product and the price data. The integration separates the product data from the price list data, and then synchronizes the price data in a separate process.

This integration flow delivers these services:

- SyncProductBRMCommsReqABCSImpl with operation SyncProduct
- CommunicationsItemCompositionEBSV1 with operation SyncItemCompositionList
- CommunicationsItemCompositionResponseEBSV1 with operation SyncItemCompositionListResponse
- SyncItemCompositionListSiebelCommsProvABCSImpl with operation SyncItemCompositionList
- CommunicationsPriceListEBSV2 with operation SyncPriceListList
- ProductOptimizedSyncPriceListListSiebelCommsProvABCSImpl with operation ProductOptimizedSyncPriceListList

This is the sequence diagram for the product synchronization:




## Product synchronization sequence diagram

When this process starts, the following events occur:

1. In the Oracle BRM Pricing Center, go to File, New. Drag the product that you want to edit to the right pane. Double-click the product name and edit appropriate fields. To commit the changes, click File, Commit to BRM Database. The modified product is published to the Oracle BRM product queue.
2. Dequeue the Oracle BRM product queue. The adapter SyncProductInfoChangeBRMAQ polls the Oracle BRM product queue. It is dequeued whenever it sees a message in the queue and invokes the SyncProductBRMCommsReqABCSImpl with the operation SyncProduct.
3. Invoking the SyncProductBRMCommsReqABCSImpl with the operation SyncProduct routes the Oracle BRM product message to the SyncProductBRMCommsReqABCSImpl.
4. The SyncProductBRMCommsReqABCSImpl first transforms the Oracle BRM product message into an ItemCompositionEBM and calls the CommunicationsItemCompositionEBSV1 with the operation SyncItemCompositionList.

   CommunicationsItemCompositionEBSV1 is a routing service with several operations on the ItemCompositionEBO.

5. The CommunicationsItemCompositionEBSV1 routes the ItemCompositionEBM to the SyncItemCompositionListSiebelCommsProvABCSImpl.

6. The SyncItemCompositionListSiebelCommsProvABCSImpl transforms the ItemCompositionEBM into the Siebel product message and then calls the Siebel product web service on operation SWIProductImportUpsert. The Siebel web service completes the request and returns a response message. The SyncItemCompositionListSiebelCommsProvABCSImpl then transforms the Siebel response message to an ItemCompositionResponseEBM and sends it back to CommunicationsItemCompositionResponseEBSV1.
7. The CommunicationsItemCompositionResponseEBSV1 returns the ItemCompositionResponseEBM to the SyncProductBRMCommsReqABCSImpl.
8. The SyncProductBRMCommsReqABCSImpl transforms the Oracle BRM product message into the PriceListListEBM and calls the CommunicationsPriceListEBSV2 with the operation SyncPriceListList.

   PriceListEBS is a routing Enterprise Service Bus (ESB) service with several operations on the PriceListEBO.

9. The CommunicationsPriceListEBSV2 routes the message to the ProductOptimizedSyncPriceListListSiebelCommsProvABCSImpl.
10. The ProductOptimizedSyncPriceListListSiebelCommsProvABCSImpl first identifies the event to be associated with the main product and then transforms the SyncPriceListListEBM to a SyncItemCompositionListEBM and calls the CommunicationsItemCompositionEBSV1 with the operation SyncItemCompositionList.

    CommunicationsItemCompositionEBS is a routing ESB service with several operations on the ItemCompositionEBO.

11. The CommunicationsItemCompositionEBSV1 routes the message to the SyncItemCompositionListSiebelCommsProvABCSImpl.
12. The SyncItemCompositionListSiebelCommsProvABCSImpl transforms the ItemCompositionEBM to the Siebel product message and then calls the Siebel product web service on operation SWIProductImportUpsert. The Siebel web service completes the request and returns a response message. SyncItemCompositionListSiebelCommsProvABCSImpl then transforms the Siebel response message to an ItemCompositionResponseEBM and returns it to the CommunicationsItemCompositionResponseEBSV1.
13. The CommunicationsItemCompositionResponseEBSV1 returns the ItemCompositionResponseEBM to the ProductOptimizedSyncPriceListListSiebelCommsProvABCSImpl.
14. The ProductOptimizedSyncPriceListListSiebelCommsProvABCSImpl transforms the PriceListEBM to a Siebel price list message and then calls the Siebel price list web service on operation Price_spcList_spcItem_spcInsertOrUpdate. The ProductOptimizedSyncPriceListListSiebelCommsProvABCSImpl transforms the PriceListListEBM to a Siebel product message and then calls the Siebel product web service on operation SWIProductImportUpsert. The Siebel web service completes the request and returns a response message. SWIProductImportUpsert then transforms the Siebel response message to a PriceList ListResponseEBM.

## Synchronization of Billing Discounts Technical Overview

The billing discount synchronization integration flow enables you to create billing discounts as billing products in Oracle BRM and then synchronize those billing discounts with Siebel CRM. The billing discounts created are used by the Order Capture module in Siebel CRM.

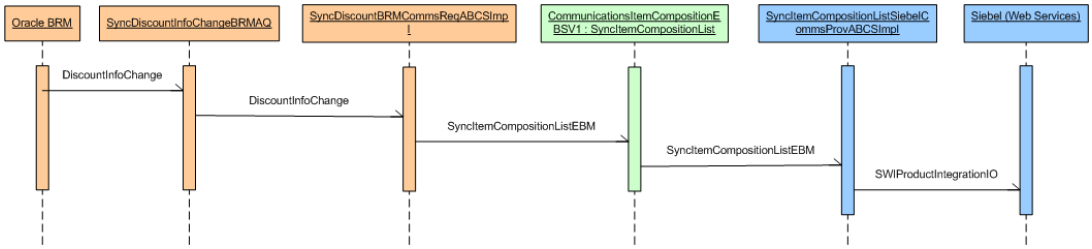
The billing discount synchronization flow also enables updates to billing discounts in Oracle BRM. The updates are then synchronized in Siebel CRM.

The billing discount synchronization integration flow synchronizes only the basic billing discount attributes. It does not synchronize any price information. Add the billing discount detail information in the description of the billing discount when creating billing discounts

This integration flow delivers these services:

- SyncDiscountBRMCommsReqABCSImpl with operation SyncDiscount
- CommunicationsItemCompositionEBSV1 with operation SyncItemCompositionList
- SyncItemCompositionListSiebelCommsProvABCSImpl with operation SyncItemCompositionList

This is the sequence diagram for the billing discount synchronization:




Billing discount synchronization sequence diagram

When this process starts, the following events occur:

1. In the Oracle BRM Pricing Center, go to File, New. Drag the discount that you want to edit to the right pane. Double-click the product name and edit the appropriate fields. To commit the changes, click File, Commit to BRM Database. The modified discount is published to the Oracle BRM discount queue.
2. Dequeue the Oracle BRM discount queue. The adapter SyncDiscountInfoChangeBRMAQ polls the Oracle BRM discount queue. It is dequeued whenever it sees a message in the queue and invokes the SyncDiscountBRMCommsReqABCSImpl with the operation SyncDiscount.
3. Invoking SyncDiscountBRMCommsReqABCSImpl with operation SyncDiscount routes the Oracle BRM discount message to the SyncDiscountBRMCommsReqABCSImpl service.
4. The SyncDiscountBRMCommsReqABCSImpl first transforms the Oracle BRM discount message into the ItemCompositionEBM and calls the CommunicationsItemCompositionEBSV1 with the operation SyncItemCompositionList.

   CommunicationsItemCompositionEBSV1 is a routing ESB service with several operations on the ItemComposition EBO.

5. The CommunicationsItemCompositionEBSV1 routes the message to the SyncItemCompositionListSiebelCommsProvABCSImpl.

6. The SyncItemCompositionListSiebelCommsProvABCSImpl transforms the ItemCompositionEBM into the Siebel product message and then calls the Siebel product web service on operation SWIProductImportUpsert. The Siebel web service completes the request and returns a response message to the SyncItemCompositionListSiebelCommsProvABCSImpl.

# Solution Assumptions and Constraints

1. Oracle BRM deals and plans are not synchronized from Oracle BRM to Siebel CRM. The service bundles and promotions are manually defined in Siebel CRM.
2. Credit limits are not synchronized from Oracle BRM to Siebel CRM.
3. Sharing groups are not synchronized from Oracle BRM to Siebel CRM.
4. Multiple brands defined within a single instance of Oracle BRM are not supported by the integration.
5. The synchronization of billing products and billing discounts is one-way. Billing products created or updated in Siebel CRM are not synchronized back to Oracle BRM. Oracle BRM is the product master.
6. The integration does not support multiple pricelists or multiple currencies. All products are created under a single pricelist and a single currency in Siebel CRM. The price list (with its specified currency) used is the one specified in the AIAConfigurationProperties.xml file.

**For more information** about configuration properties, see Chapter 3: Configuring the Process Integration for Product Lifecycle Management.

7. All of the billing products created by this synchronization are associated with one business unit in Siebel CRM. This is the business unit that is specified in the AIAConfigurationProperties.xml file.

**For more information** about business units, see the Siebel CRM product documentation.

**For more information** about configuration properties, see Chapter 3: Configuring the Process Integration for Product Lifecycle Management.

8. All of the billing products synchronized to Siebel CRM are created in a single workspace in Siebel CRM. This is the workspace specified in the AIAConfigurationProperties.xml file.

**For more information** about workspaces, see the Siebel CRM product documentation.

**For more information** about configuration properties, see Chapter 3: Configuring the Process Integration for Product Lifecycle Management.

9. If a product in Oracle BRM has multiple rate plans or multiple tiers, the integration does not synchronize the pricing information. The price is set to $0 in Siebel CRM for such products.

**For more information,** see Chapter 2: Understanding the Product Bundling Methodology.

10. We recommend you use Siebel discounts for discounting purchase fees on products. Based on the pricing commit type, Siebel discounts get applied as price or discount overrides when the order is interfaced to billing.

**For more information** about pricing commit type, see Defining Overrides on the Product Definition.

For Oracle BRM purchase fee discounts to get applied consistently, the discount must be purchased before the product that it applies to. Both the Order Management system and the AIA connector service that interfaces the order to billing must recognize this and currently, the AIA connector service does not handle this sequencing requirement.

In cases where discounts are defined in Oracle BRM and already synchronized to Siebel CRM, they must not be used in the bundling of products to create offers or promotions. Also, products, bundles, or promotions, which have purchase fee discounts must not be used to create quotes or orders.

**Note:** This guide does not address upgrade issues for customers that already have in-flight orders or transaction data with purchase fee discounts interfaced to billing.

11. The lists associated with the Special Rating products (such as Friends and Family) are defined in Siebel CRM. An external communication must exist between the Siebel product Administrator and the Oracle BRM pricing administrator to communicate the name of the lists that were created in Siebel CRM. The Oracle BRM administrator creates the labels for the corresponding list names in Oracle BRM. Oracle BRM uses labels to identify the friends and family type lists. The labels are used to associate special pricing models in Oracle BRM Pricing.
12. When a billing product is deleted in Oracle BRM, it does not publish any message. The corresponding billing product in Siebel CRM is not deleted or inactivated automatically. You must inactivate this billing product manually in Siebel CRM. If you delete a billing product in Oracle BRM that is already synchronized with Siebel CRM, then the cross-reference data for that billing product is not deleted. This has to be purged manually. We recommend that you not delete products in Oracle BRM but instead inactivate the product in Oracle BRM by setting the product end date.
13. The billable events that are associated with billing products in Oracle BRM must be included in the PRICETYPE_EVENT domain value map. If an event is not included in the DVM, the process integration ignores the event. In other words, the process integration does not create a corresponding simple product that represents the event (billing type = Event) in Siebel. The process integration does not end in error, nor does it send a notification that an event was not found in the DVM.
14. Oracle BRM is the master for usage pricing. When billing products with only one usage event are synchronized from Oracle BRM, a simple product with a price type of One-Time is created in Siebel CRM. The pricing information for such products must not be changed in Siebel CRM. For example, a price override or discount must not be specified in Siebel CRM. If the price is updated in Siebel the changes are not propagated to Oracle BRM or applied when the order is interfaced to billing.

15. A service bundle can have another service bundle as a component product. A service bundle that does not have another service bundle as one of its component products must have the same billing service type as its component products. Violation of this assumption can result in Oracle BRM grouping the billed charges under the wrong bucket (bill-item).The product synchronization sets the asset-trackable flag to Y for Oracle BRM products of type subscription and N for products of type item or system.
16. The product synchronization process ignores the effective start date and effective end date that are specified on the rate tier of the billing products. The effective start date on the price line in Siebel CRM is set to the creation date and time and the effective end date is not set.
17. Because BPEL flows are transactional in nature, they must not be used for either initial data loads or considerable-sized data loads. Instead, you should create your own data loading capability using appropriate tools or scripts. You must also create scripts to populate cross-reference data.

# Oracle BRM Interfaces

The process integration for product management uses these services:

- SyncProductInfoChangeBRMAQ: The adapter SyncProductInfoChangeBRMAQ polls the BRM Product queue. It dequeues whenever it sees a message in the queue and invokes SyncProductBRMCommsReqABCSImpl with the operation SyncProduct.
- SyncDiscountInfoChangeBRMAQ: The adapter SyncDiscountInfoChangeBRMAQ polls the BRM Discount queue. It dequeues whenever it sees a message in the queue and invokes SyncDiscountBRMCommsReqABCSImpl with the operation SyncDiscount.

**For more information,** see *Oracle Communications Billing and Revenue Management (BRM) Documentation*, “Service Integration Components,” Synchronization Queue Data Manager.

# Siebel CRM Interfaces

The process integration for product management uses these Siebel CRM interfaces:

- SWIPriceListItem
- SWI Product Import

**For more information,** see the *Siebel Order Management Guide Addendum for Communications*, “Web Services Reference.”

# Industry AIA Components

The process integration for product management uses the following delivered Industry AIA components:

- ItemCompositionEBO
- SyncItemCompositionListEBM
- SyncItemCompositionListResponseEBM

- PriceListEBO
- SyncPriceListListEBM
- SyncPriceListListResponseEBM

The industry enterprise business object (EBO) and enterprise business message XML schema (EBM XSD) files are located here: http://<server name>:<port number>/AIAComponents/EnterpriseObjectLibrary/Industry/Communications/EBO/

The industry enterprise business service (EBS) WSDL files are located here: http://<server name>:<port number>/AIAComponents/EnterpriseBusinessServiceLibrary/Industry/Communications/EBO/

For detailed documentation about individual EBOs, click the EBO Name link on the Integration Scenario Summary page of the Oracle AIA Console. You can also use the Integration Scenario Summary page to search for and view integration scenarios that use a particular EBO or EBS.

**For more information,** see *Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide,* "Using the BSR UI to View Integration Scenarios."

EBOs can be extended, for instance, to add new data elements. These extensions are protected, and they will remain intact after a patch or an upgrade.

**For more information,** see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide,* "Extensibility for Oracle AIA Artifacts."

**For more information** about the changes made for each EBO and EBM, see the appendix in the My Oracle Support document number 988374.1.

# Integration Services

These services are delivered with this integration:

- CommunicationsItemCompositionEBSV1
- CommunicationsItemCompositionResponseEBSV1
- CommunicationsPriceListEBSV2
- SyncProductBRMCommsReqABCSImpl
- SyncDiscountBRMCommsReqABCSImpl
- SyncItemCompositionListSiebelCommsProvABCSImpl
- ProductOptimizedSyncPriceListListSiebelCommsProvABCSImpl

Some of these services have been enabled to use Session Pool Manager.

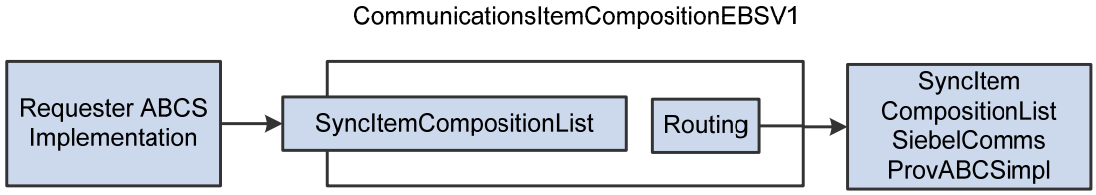
**For more information** about Session Pool Manager, see Appendix H: Using Session Pool Manager.

Use the Integration Scenario Summary page in the Oracle AIA Console to search for and view integration scenarios that use a particular ABC service.

**For more information,** see *Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide,* "Using the BSR UI to View Integration Scenarios."

## CommunicationsItemCompositionEBSV1

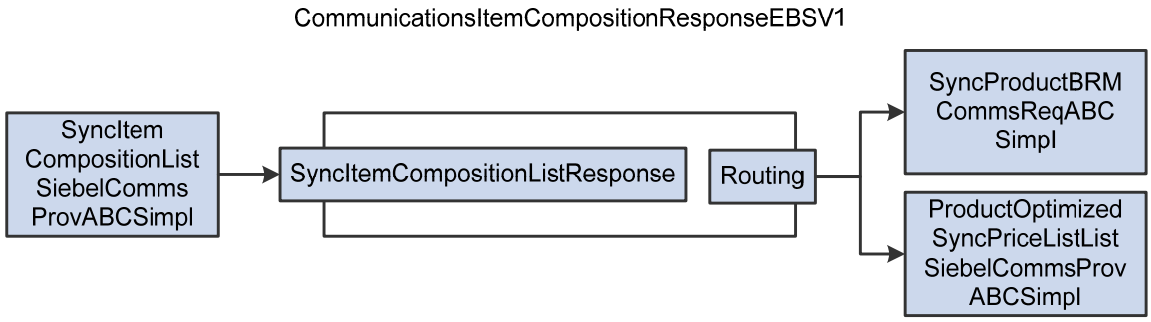
The CommunicationsItemCompositionEBSV1 performs all of the Product/Item-related actions such as Create Product/Item, Update Product/Item, and Sync Product/Item. Based on the routing rules setup, it invokes a provider application business connector service (ABCS). It has one operation: SyncItemCompositionList.




**For more information** about this EBS, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs."

## CommunicationsItemCompositionResponseEBSV1

CommunicationsItemCompositionResponseEBSV1 simply routes the ItemCompositionResponse EBM to BRM requestor ABCS implementation. It has one operation: SyncItemCompositionListResponse.




**For more information** about this EBS, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs."

## CommunicationsPriceListEBSV2

The CommunicationsPriceListEBSV2 performs all of the PriceList-related actions such as Create PriceList, Update PriceList, Sync PriceList, and Sync PriceListList. This operation has the standard create, read, update, delete (CRUD) operations. It has one operation: SyncPriceListList.

CommunicationsPriceListEBSV2


**For more information** about this EBS, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs."

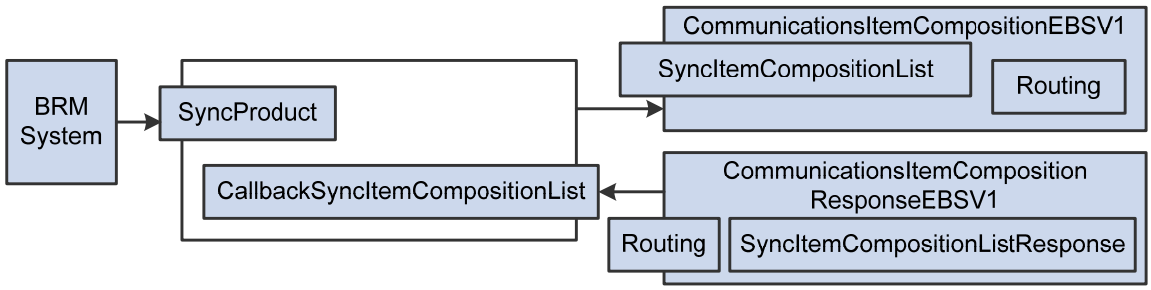
## SyncProductBRMCommsReqABCSImpl

The SyncProductBRMCommsReqABCSImpl has the operation SyncProduct. This accepts an Oracle BRM product message as a request and does not return a response.

This service accepts a BRM product message. An Oracle BRM product message has two sets of information:

- Standard product attributes.
- Pricing information that can be mapped to a PriceLine of a PriceList.

SyncProductBRMCommsReqABCSImpl




Because of this, the Oracle BRM product message is transformed into two EBMs: one for the product (SyncItemCompositionListEBM) and another for the PriceLine (SyncPriceListList EBM).

The program first prepares the SyncItemCompositionListEBM with the basic product information and invokes the CommunicationsItemCompositionEBSV1.SyncItemCompositionList operation. After this, it waits for a response from CommunicationsItemCompositionResponseEBSV1.

After the SyncItemCompositionList is complete, it prepares a SyncPriceListListEBM with the pricing information of the Oracle BRM message and invokes the CommunicationsPriceListEBSV2.SyncPriceListList operation. It fetches the PriceList name from a configuration parameter.

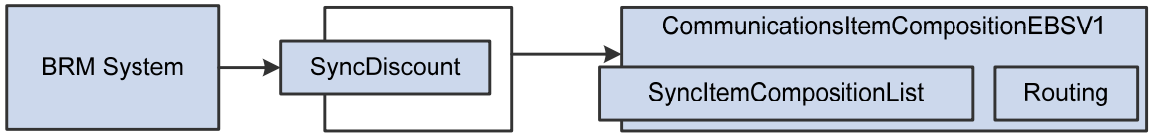
The configuration parameter is located in the AIAConfigurationProperties.xml file.

## SyncDiscountBRMCommsReqABCSImpl

The SyncDiscountBRMCommsReqABCSImpl is a BPEL service and it is the Oracle BRM discount request ABC implementation. It has the operation SyncDiscount. This accepts an Oracle BRM discount message as a request and does not return a response.

The SyncDiscountBRMCommsReqABCSImpl service accepts an Oracle BRM discount message. An Oracle BRM discount is created as a product for all of the recipients. An Oracle BRM discount message has basic discount attributes and does not contain any pricing information. The Oracle BRM discount message is transformed into the SyncItemCompositionListEBM with the basic discount information that invokes the CommunicationsItemCompositionEBSV1.SyncItemCompositionListoperation.

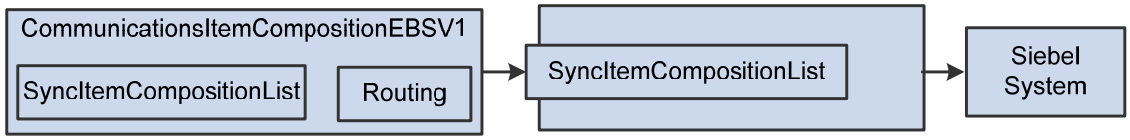
SyncDiscountBRMCommsReqABCSImpl




## SyncItemCompositionListSiebelCommsProvABCSImpl

The SyncItemCompositionListSiebelCommsProvABCSImpl process accepts the SyncItemCompositionListEBM. It transforms SyncItemCompositionListEBM into the Siebel product application business message (ABM). It then invokes the Siebel Product web service to create products and product structures in Siebel.

SyncItemCompositionListSiebelCommsProvABCSImpl

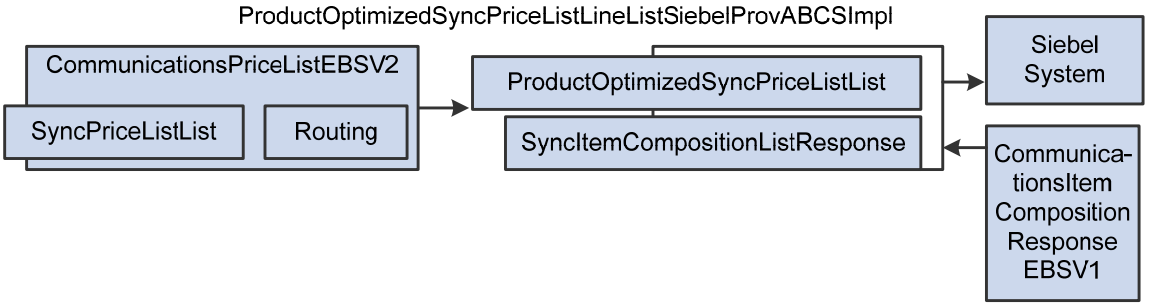



This service is now SPM (Session Pool Manager) enabled.

**For more information** about SPM, see Appendix H: Using Session Pool Manager.

## ProductOptimizedSyncPriceListListSiebelCommsProvABCSImpl

The ProductOptimizedSyncPriceListLineListSiebelProvABCSImpl service transforms the PriceListEBM into a Siebel price list message and then calls the Siebel price list web service on operation InsertOrUpdate. The ProductOptimizedSyncPriceListLineListSiebelProvABCSImpl transforms the PriceListEBM into a Siebel product message and then calls the Siebel product web service on operation Product_spcImport_spcUpdate. The Siebel web service completes the request and returns a response message. ProductOptimizedSyncPriceListLineListSiebelProvABCSImpl then transforms the Siebel response message into the PriceList response EBM and sends it back to PriceListEBS.




This service is now SPM (Session Pool Manager) enabled.

**For more information** about SPM, see Appendix H: Using Session Pool Manager.

# Chapter 2: Understanding the Product Bundling Methodology

This section describes the methodology for introducing service bundles and marketing bundles in relation to synchronizing billing products from Oracle Communications Billing and Revenue Management (Oracle BRM) to Siebel Customer Relationship Management (Siebel CRM). This chapter discusses:

- Basic entity mappings.
- Defining products and discounts.
- Physical goods.
- Sales catalogs.
- Recommendations for product definition.
- Service bundles.
- Simple service bundles.
- Marketing bundles.
- Balance groups.
- Credit limits.
- Penalty products.
- Supporting friends and family.
- Supporting time-based offerings.

## Basic Entity Mappings

This table shows the mapping between Oracle BRM and Siebel CRM entities:

| Oracle BRM Entities | Siebel CRM Entities | Description |
|---|---|---|
| Product with single event | Simple Product [automatically created] | If a product is associated with a single billable event in Oracle BRM, then a simple product is created in Siebel CRM. |
| Product with multiple events | Customizable product [automatically created] | If a product is associated with more than one billable event in Oracle BRM, then a customizable product is created in Siebel CRM. |
| Product Event Binding | Simple Product [automatically created] | Each recurring and nonrecurring type event binding is represented as a simple product. |
| Discount | Simple Product [automatically created] | A billing discount is represented as a simple product regardless of the number of event |

| Oracle BRM Entities | Siebel CRM Entities | Description |
|---|---|---|
| | | bindings. |
| Balance Impact | Price list Line [automatically created] | A balance impact defined as part of a rate plan in BRM is mapped to a price list line of a product in Siebel. |
| Deal | Service Bundle [manually created] | If existing Oracle BRM customers have previously defined deals, those deals are not synchronized as part of the Product Lifecycle Management (PLM) integration. The service bundles must be created manually in Siebel. |
| Plan | Promotion /Marketing Bundle [manually created] | If existing Oracle BRM customers have been previously defined in a plan, those plans are not synchronized as part of the PLM integration. The Promotion/ Marketing bundles must be created manually in Siebel CRM. |
| Service Instance | Service Bundle Asset [automatically created] | Purchasing a service bundle results in a service bundle asset that will be mapped to an Oracle BRM service instance to support changes to the service. |
| Purchased Products | Service Bundle Component Asset [automatically created] | Purchasing optional and mandatory components of a service bundle results in asset components that will be mapped to Oracle BRM-purchased products. |

# Defining Products and Discounts in Oracle BRM

When defining the products and discounts in Oracle BRM, use the following guidelines to fully leverage the flexibility and minimize the limitations of this integration:

- Since usage events are not synchronized when they are included as a part of multi-event product in Oracle BRM, the name and description of products should include some user-readable identity of the usage. That way the product or price administrator can distinguish the synchronized products on the Siebel side.
- Since the discount value of the BRM discount objects is synchronized to Siebel, the name and description of the discount objects should include the general intent of the discount to be conveyed on the Siebel order.
- The discountable flag on billing products in Oracle BRM must be set to Y for all charges that can be discounted when orders are interfaced to billing.

## Using Fixed Amounts versus Scaled Amounts in Oracle BRM

In Oracle BRM, the price on the billable events that are associated with the billing products can be of type Scaled or Fixed. From the user interface (UI) perspective, in the pricing center application of Oracle BRM, when the price needs to be associated to the event, two fields exist where the charge can be added.

- Scaled amount: Specifying the scaled amount allows price overrides and discounts to be applied on the price. When the scaled amount field is used then the fixed amount field must be left empty (null). Zero must not be specified. The scaled amount is specified only for billable events that represent one-time or recurring charges.
- Fixed amount: Discount override takes into consideration both a fixed and a scaled amount. However price override only overrides the scaled amount. The price overrides can still be applied for the charges but it will get added to the price specified as fixed amount. For example, if the fixed amount on the charge is $5 and a price override is $10 then the price will be $15.

Consider the case where both the scaled amount and the fixed amount are specified for the product. The product integration synchronizes the product to the Siebel CRM and the list price is the sum of the scaled and fixed amounts. If a discount override is specified for the product, when the order is interfaced to billing the discount override is applied on the sum for the purchased product instance in Oracle BRM.

For example, a billing product has a monthly cycle fee specified as: Scaled = $20 and Fixed = $10.

A discount override of 10% results in a final price of $27 and a discount override of $5 results in a final price of $25.

If a price override is specified for the product, when the order is interfaced to billing, Oracle BRM replaces only the scaled amount with the price override amount for the purchased product instance.

For example, a billing product has a monthly cycle fee specified as: Scaled = $20 and Fixed = $10.

A price override of $15 results in a final price of $25 (Scaled $15 + Fixed $10).

This behavior for the price override scenario results in a discrepancy between the final price for a product on the order in Siebel CRM and what the customer is actually charged in Oracle BRM. Therefore, we recommend that you not use fixed amounts for either one-time or recurring charges in Oracle BRM for implementations where the intent is to use the Siebel price override functionality.

**For more information** about using fixed and scaled amount fields in Oracle BRM, see the BRM Product documentation.

# Physical Goods

Customers can use one of two possible approaches:

- Physical goods can be created as a billing product in Oracle BRM at account-level or at service-level. These are synchronized to Siebel CRM and can be added to the product hierarchy when creating bundles and promotions.

- Physical goods are defined in ERP. In this case, customers are responsible for synchronizing them between ERP and Oracle BRM, The product synchronization process, which is supported by the process integration, is used to synchronize the product from Oracle BRM to Siebel CRM. If the service or marketing bundle contains one or more physical goods, then those products will be passed to Oracle BRM when the order is interfaced to billing.

## Sales Catalogs

After all of the Oracle BRM products are synchronized to Siebel CRM, you must add only those products that can be ordered to the catalogs (products whose orderable flag is set). If the customizable products are added to the catalog then the components are automatically added.

| Oracle BRM Entities | Siebel Synchronized Entities | Siebel Catalog |
|---|---|---|
| Product: Wireless (Yearly)<br>Event: YCF - $100 | Wireless – YCF - $100 | It must be added as a component to a service bundle product, which must be added to the sales catalog. |
| Product: Wireless (Monthly)<br>Event: MCF - $40<br>Event: Usage - $0.40 | Wireless - MCF -$40 | The product must be added as a component to a service bundle product, which must be added to the sales catalog. |
| Product: Wireless Activation<br>Event: Activation - $10 | Wireless - Activation - $10 | The product must be added as a component to a service bundle product, which must be added to the sales catalog. |
| Product: SMS Activation<br>Event: Activation - $10 | SMS Activation - $10 | The product must be added as a component to a service bundle product, which must be added to the sales catalog. |
| Product: SMS Usage<br>Event: Usage - $0.05 | SMS Usage | The product must be added as a component to a service bundle product, which must be added to the sales catalog. |

## Recommendations for Product Definition in Siebel CRM

These are the recommendations for defining products:

- Oracle BRM billing products that are defined with fixed charges should not be discounted in Siebel (using promotion discounts, price overrides, and so forth) because communicating such overrides to Oracle BRM will result in a price increase. For this reason it is recommended that only scaled charges be defined for the billing products of type item and subscription with one-time or recurring charges in Oracle BRM.

**For more information**, see Using Fixed Amounts versus Scaled Amounts in Oracle BRM.

- The Product Management integration maintains cross-reference information between Oracle BRM billing products and Siebel CRM products. If you delete a billing product in Oracle BRM that is already synchronized with Siebel CRM, then the cross-reference data for that billing product is not deleted. This has to be purged manually. It is recommended that instead of deleting the product you inactivate it by specifying an end date.
- If products updated in Oracle BRM result in changing the product structure in Siebel CRM, then you must release the updated product in its respective workspace. This automatically updates the service bundles and the promotions that include the updated product as one of its components.

## Recommendation for Discounts

This section describes customizable discounts that are time-based or that impact noncurrency resources and multiple event types.

### Discounts Defined in Billing Systems

Customizable discounts that are either time-based, or that impact noncurrency resources or multiple event types, must be defined in Oracle BRM. These can be account-level or service-level discounts. Because you can associate general ledger IDs (GLIDs) with them in Oracle BRM, you can account for them in the general ledger in separate accounts if needed.

These discounts are defined in Oracle BRM and synchronized to Siebel CRM as simple products (Structure type = none). The products that represent the discounts are identified using the billing type =Discount. You manually bundle the service-level discounts into the service bundles.

These can be included or excluded during promotion bundling. The account-level discounts are directly added as components of the promotions and can be made optional based on promotional bundling.

### Discounts Defined in Siebel CRM Systems

You can define simple discounts in Siebel CRM when you bundle the billing products into service bundles and promotions. These are usually matrix or promotional discounts. At run time, these discounts getting applied on the order, results in a difference in the start price or list price and the net price.

### Defining Overrides on the Product Definition

The following offers you greater control and flexibility in determining how pricing differences between the list price and the selling price are communicated to the billing system. Two new fields are on the Siebel product definition:

- Pricing commit type.
  - The value of the pricing commit type field indicates whether a price override or a discount override is being defined on the product:
  - If the pricing commit type is Committed, then a price override has been defined on the product.
  - If the pricing commit type is Dynamic, then a discount override has been defined on the product. If a discount override has been defined on the product, then the Dynamic

discount method field identifies the discount type.

- Dynamic discount method.
  - If the dynamic discount method is Amount, then an amount is defined as the discount value.
  - If the dynamic discount method is Percent, then a percent discount has been defined as the discount value.

In Oracle BRM, discount overrides can be tracked in a separate sub-bucket within the GL code that is tied to the product. With discount overrides, mass price changes can also be supported because the list price on the product remains unchanged.

# Service Bundles

Billing products (with single or multiple events) are created in Oracle BRM and are synchronized with Siebel CRM. Whenever billing products have to be bundled together, one must manually create a customizable product and set the billing type to Service Bundle in Siebel CRM. This product is called a service bundle product. You must add the billing products that are synchronized from Oracle BRM as child components to the service bundle product. Service bundles map to the run-time entity called service instances in Oracle BRM.

The service bundle products are created manually in Siebel CRM.

You can also create a simple service bundle when only one billing product is applicable for a given service.

**For more information,** see Simple Service Bundles.

The process integration for order management uses the service bundle to construct the billing service instance for billing management.

The Oracle BRM discounts that are synchronized as products with Siebel can be included in the service bundle. If they are included in the service bundle, then at run time, when the service bundle is purchased and interfaced to billing, those discounts apply to the products within the service instance. If they are not included in the service bundle, but purchased on the order, then they get applied as account level discounts.

**Note:** Any product (Oracle BRM product or discount) whose immediate parent is not a service bundle at run time gets purchased as an account-level product or discount in Oracle BRM.

The product bundling methodology gives Siebel CRM product administrators more flexibility when creating service bundles and promotions. Product administrators can nest service bundles and these nested service bundles do not need the same billing service type as the parent or root service bundle. However, within a service bundle, all of the component products must be of the same billing service type. The methodology supports a nested structure in which service bundles can be included as a component of another service bundle.

In the case of multiple billing system instances connected to the same Siebel CRM system instance, all component products within a service bundle reference products from the same billing system. Siebel CRM does not store the target billing instance details.

**For more information** about service bundles in Siebel CRM, see the *Siebel Communications Guide*.

**For more information** about multiple Oracle BRM systems, see Appendix E: Configuring Multiple Instances of Oracle BRM.

In addition to billing products and nonbilling products, the methodology gives a Siebel CRM product administrator the option to include child service bundles and nonservice bundle customizable products as components of a service bundle.

Here are some definitions of the components:

- Billing products are created by the product synchronization. They can be defined as simple products or customizable products based on the number of events. Products with one billable event are synchronized as simple products and products with more than one event are synchronized as customizable products in Siebel.
- Nonbilling products are products that are not originated or synchronized from Oracle BRM. A billing service type should not be specified for nonbilling products.
- Account-level products are associated at the account level and are not associated with any service instance in Oracle BRM; for example, a $2 monthly charge for a hard copy of the bill is charged to the account. The product definition methodology recommends not including account-level products within a service bundle.
- Service bundles can include another service bundle or nonservice bundle customizable products as a component. No limit is placed on the number of levels in the hierarchy. Child service bundles do not need to have the same billing service type as the root bundle.
- Nonservice bundle customizable products are customizable products that group service bundles. Nonservice bundle customizable products can have account-level products and non-billing products as components. They do not have a billing service type.

## Working with Products and Nested Service Bundles

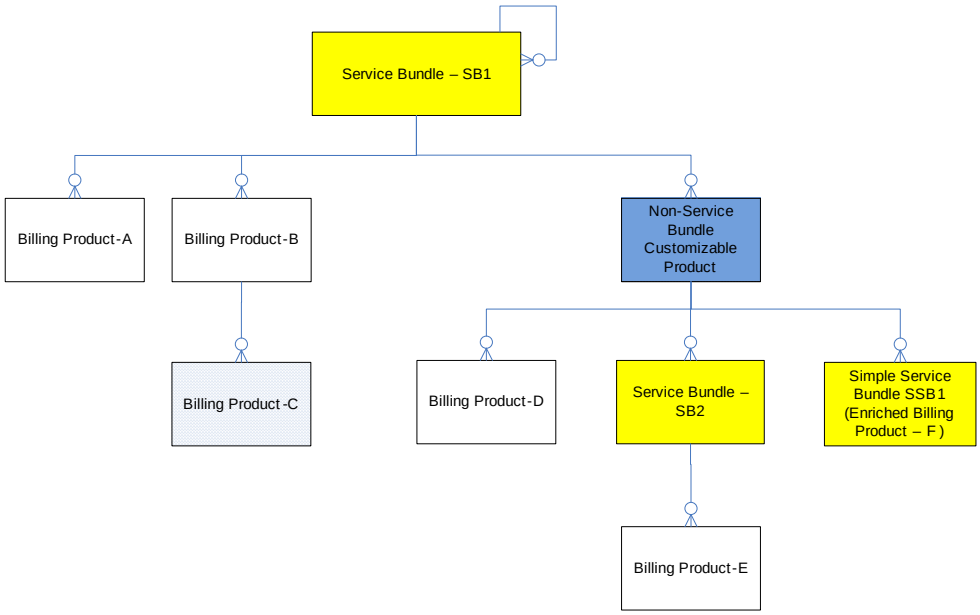
Consider the following example:

Billing Product – A, Billing Product – B, Billing Product – C, Billing Product – D, Billing Product – E, and Billing Product – F are all synchronized from Oracle BRM to Siebel CRM.

1. Create Service Bundle – SB2 with Billing Product – E as the component.
2. Enrich Billing Product – F so that it becomes a Simple Service Bundle – SSB1
3. Create a Nonservice Bundle Customizable Product (CP) with the following components:
   - Billing Product – D
   - Service Bundle – SB2
   - Simple Service Bundle – SSB1
4. Create Service Bundle – SB1 with the following components:
   - Billing Product – A
   - Billing Product – B

- Nonservice Bundle CP

This diagram shows the example of a nested service bundle described previously:




## Example of nested service bundle levels

When an order is interfaced to billing, the service bundle gets purchased as a service and the immediate children of the service bundle become purchased product or discount instances for that service instance. *Any product whose immediate parent is not a service bundle gets purchased at the account-level*.

**Note:** Dynamic or relationship classes do not get instantiated on the order and are therefore irrelevant in terms of determining a service bundle parent.

Therefore, for the above example, when Service Bundle – SB1 is purchased and the order is interfaced to billing, the following data is created in Oracle BRM:

1. Service instance for Service Bundle – SB1 with purchased product instances for Billing Product – A and Billing Product – B.

   This would be true even if Product Billing – A and Product Billing – B were members of a dynamic or relationship class.

2. Service instance for Service Bundle – SB2 with purchased product instance for Billing product – E.
3. Service instance for Simple Service Bundle – SSB1 with purchased product instance for Billing Product – F.

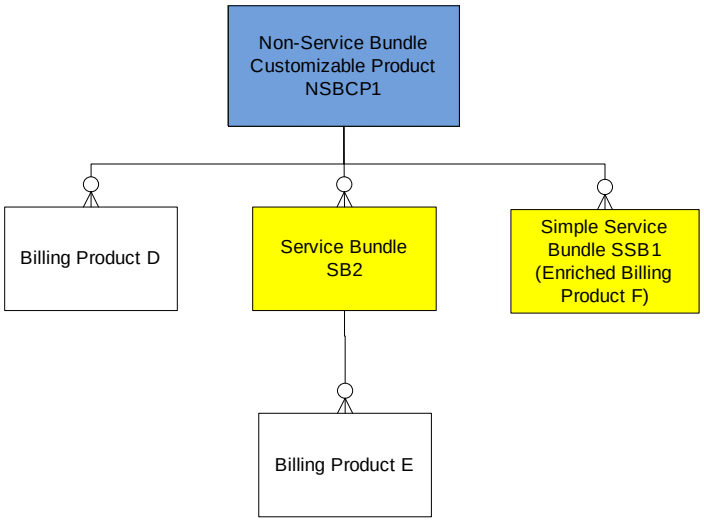
   *Billing Product – D is purchased at the account-level because its immediate parent is not a service bundle.*

4. *Billing Product – C is purchased at the account-level because its immediate parent is not a service bundle*.

**Note:** If the intention was for Billing Product – C to be purchased for Service Bundle – SB1, then it should be modeled as a sibling of Billing Product – A and Billing Product – B.

## Working with Nonservice Bundle Customizable Products

Nonservice bundle customizable products can be used to group service bundles (including nested service bundles), simple service bundles, and billing products or discounts. They serve as re-usable components for use across promotions or as is.




Using nonservice bundle customizable products as root products in Siebel CRM

**Note:** Using nonservice bundle customizable products are optional. The main benefit of using them is when you are creating promotion variants. The nonservice bundle customizable product can be used to group relevant products. When creating promotions you can add it as a component and include or exclude the components based on the promotion definition. This saves the additional overhead of adding all the components each time a new promotion is created.

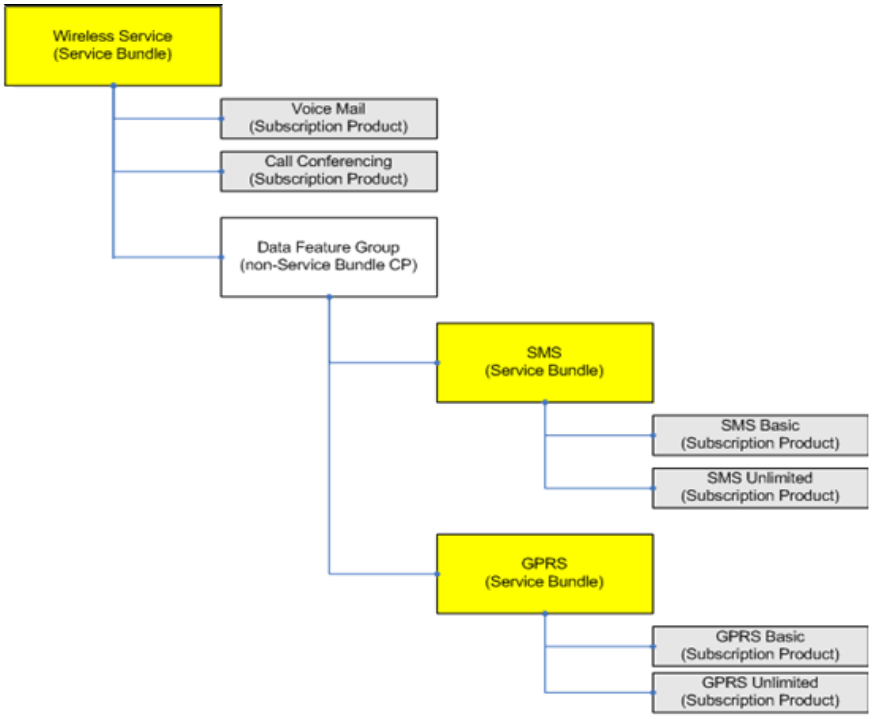
When a nonservice bundle customizable product is purchased, the same rules that are listed in the previous section apply. Therefore, using the previous diagram as an illustration, when the Nonservice Bundle Customizable Product – NSBCP1 is purchased and the order is interfaced to billing, the following data is created in Oracle BRM:

1. Service instance for Service Bundle – SB2 with purchased product instance for Billing Product – E.
2. Service instance for Simple Service Bundle – SSB1 with purchased product instance for Billing Product – F.

   *Billing Product – D is purchased at the account-level because its immediate parent is not a service bundle*.

## Working with Service Bundles with a Child Non-Service Bundle Customizable Product

This diagram shows the hierarchical relationships of service bundles with a child nonservice bundle customizable product in Siebel CRM:

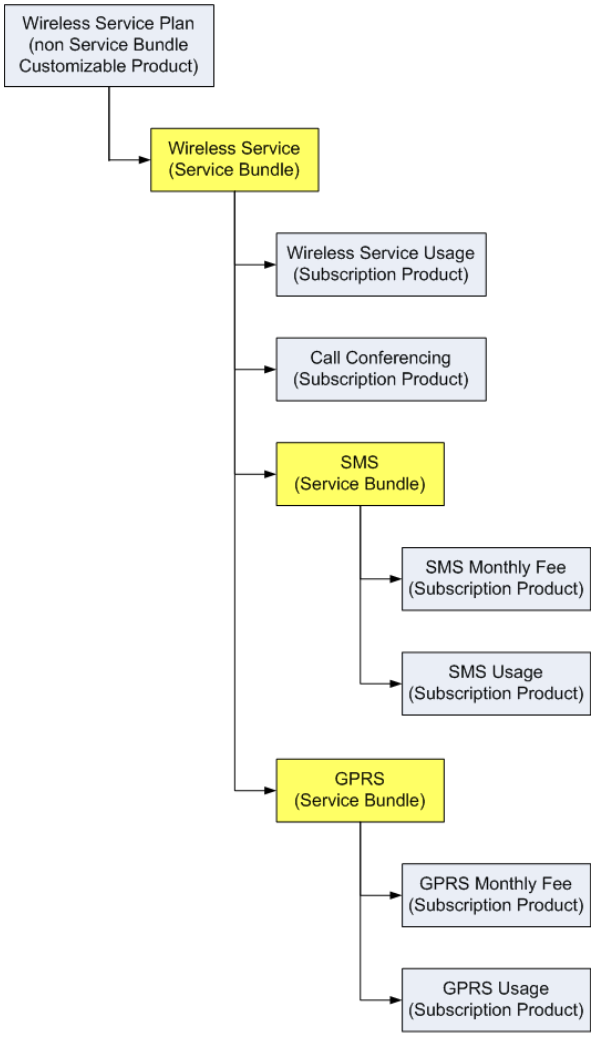



### Service bundles with a child nonservice bundle customizable product

For this diagram:

- No limits are imposed on the number of levels of service bundles. The diagram depicts three levels of bundles.
- The billing products, Voice Mail and Call Conferencing, within a service bundle share the same billing service type as the parent, Wireless Service. Similarly, general packet radio service (GPRS) Basic and GPRS Unlimited share the same billing service type as GPRS, and short message service (SMS) Basic and SMS Unlimited share the same billing service type as SMS.
- If a service bundle does not contain any billing products (for example, if it contains only a child service bundle that has component products or account-level products), then no service instance will be created in Oracle BRM for the service bundle. According to the product definition methodology, at least one subscription-based billing product must be a component product of a service bundle. If no billing products are component products, then customers can use a nonservice bundle customizable product. Note that service instances are not created in Oracle BRM for nonservice bundle customizable products.

Depending on the commercial strategy (rules, usability, and user journey), the service in the preceding diagram can also be modeled and ordered differently. The following diagram is an example of using a nonservice bundle customizable product as the root product.




Using a nonservice bundle customizable product as the root product

***Warning!*** Service bundles must not combine products from different billing systems. The order processing integration will fail because it expects all billing products within a service bundle to be interfaced to a single billing system.

# Simple Service Bundles

Using the service bundle methodology, Siebel product administrators must define a new customizable product of billing type Service Bundle and then bundle the subscription product inside it.

The simple service bundle methodology obviates the additional service bundle product definition. This alternate methodology does *not* replace the current one, but is supported in addition to the current one.

The simple service bundle can be a root-level product or can be nested within another service bundle (of billing type Service Bundle) or nested within a regular customizable product in Siebel.

This can be achieved by setting the service instance flag (serv_instance_flag in Siebel) to Y for a subscription product that has been synchronized from Oracle BRM.

Subscription products can be either:

- A customizable product (this represents a multi-event product in Oracle BRM) with billing type Subscription.
- A simple product (this represents a single-event product in Oracle BRM) with billing type Subscription.

At run time, when a simple service bundle is purchased, the integration creates both a service instance and a purchased product instance in the billing system.

Also note that:

- In release 2.4, any subscription product whose immediate parent is not a service bundle is processed as an account-level product at run time when interfaced to billing.
- In release 2.5, any subscription product whose immediate parent is not a service bundle *and* is not service instance-enabled is an account-level product.

**Note:** The Product Lifecycle Management (PLM) sync neither sets this flag when the product is synchronized from Oracle BRM to Siebel CRM as part of product creation nor updates/overwrites it as part of product updates synchronized from Oracle BRM to Siebel CRM. The Siebel product administrator sets the Service Instance flag manually.

## Guidelines for Using Service Bundles or Simple Service Bundles

Essentially the CSP's product bundling requirements determine whether the Siebel product administrator uses the classic or the new model to define service bundles.

The simple service bundle model can be used when only one billing product will be applicable for a given service. It does not have any service-level billing discounts tied to it, nor does a need exist to be switching from one product variant to another while retaining the same service. No need exists for special rating for this product either.

**For more information,** see Assumptions and Constraints in the following section.

Note that once a product is defined using the simple service bundle methodology, you cannot switch to using the other one (and vice versa) because that will adversely affect processing of change orders for existing assets. If the product bundling requirements change, requiring the use of the other methodology, then you need to define another product in billing, synchronize it to CRM, and bundle it differently.

Also, because you have a single asset representing both the service instance and billing product, you cannot upgrade a customer from a service modeled in this manner to one modeled based on the other methodology while retaining the same service instance. You can do the upgrade using a service cancellation and repurchase.

## Service Bundle versus Simple Service Bundle Example

Here is an example of the service bundles versus simple service bundles. Remember that both are supported.

| Service Bundle | Simple Service Bundle |
|---|---|
| CP: Internet Access Service (SB)<br>CP: Internet - MCF<br>Internet - Activation<br>Note: The internet product is mapped to multiple events in BRM. | CP: Internet-MCF (SSB)<br>Internet - Activation |

| Service Bundle | Simple Service Bundle |
|---|---|
| CP: Internet Service (SB)<br>Dynamic Class<br>Basic High Speed Internet MCF<br>Premium High Speed Internet MCF<br>Elite High Speed Internet MCF<br>Internet Secure Firewall<br>CP: High Speed Internet Features (NSB-CP)<br>CP: Internet email (SB)<br>Internet email<br>CP: Internet Instant Chat (SB)<br>Internet Instant Chat<br>CP: Internet Conference Chat (SB)<br>Internet Conference Chat<br>CP: Internet Media (SB)<br>Internet Content on Demand<br>Internet Video on Demand<br>High Speed Internet First Month-Free Discount<br>Note: The NSB-CP is optional; without it the four-feature SBs will have the Internet Service SB as the parent. | CP: Internet Service (SSB)<br>Dynamic Class<br>Basic High Speed Internet MCF<br>Premium High Speed Internet MCF<br>Elite High Speed Internet MCF<br>Internet Secure Firewall<br>CP: High Speed Internet Features (NSB-CP)<br>Internet email (SSB)<br>Internet Instant Chat (SSB)<br>Internet Conference Chat (SSB)<br>CP: Internet Media (SB)<br>Internet Content on Demand<br>Internet Video on Demand<br>High Speed Internet First Month-Free Discount<br>Note: The NSB-CP is optional; without it the four-feature SBs will have the Internet Service SB as the parent |

Legend:

Service bundle component product synced from Oracle BRM

Service bundle manually created in Siebel CRM, billing type set to Service Bundle (SB)

Subscription product synced from Oracle BRM, whose Service Instance flag is set to Y (SSB – Simple Service Bundle).

## Assumptions and Constraints

These are the assumptions and constraints:

1. Only products of type Subscription can be updated to be a simple-service bundle. This is enforced using Siebel validation.
2. Existing products that have pending quotes, orders, or assets in Siebel or Oracle BRM referencing them cannot be enabled as simple-service bundles because doing so impacts existing asset cross-references, and vice versa in products modeled as simple service bundles. You cannot switch from these to using the traditional one (that is, becoming a service bundle component). This is enforced using Siebel validation.
3. To switch from one methodology variant to another (that is, service bundle versus simple service bundle), you must define new products in Oracle BRM, synchronize them, and bundle them using the desired methodology.
4. Disconnecting the simple service bundle results in disconnecting both the service instance and the product in Oracle BRM. This means that customers cannot upgrade or downgrade from one simple-service bundle to another while retaining the same service instance.
5. Currently, the service ID is required on service bundle lines for the integration to successfully interface the purchase to Oracle BRM. Similarly, the service ID is required on the simple service bundle as well.
6. The methodology does *not* allow bundling additional billing product and discounts, special rating products, or other service bundles within a simple service bundle. Note that the only subcomponents that the simple-service bundle can have are products of billing type Event; these are synchronized from Oracle BRM. This is enforced using validations in Siebel.

**Note:** The order billing integration supports the simple-service bundle methodology for all supported features, within the constraints listed previously.

**For more information**, see Chapter 4: Understanding the Process Integration for Order Management.

# Marketing Bundles

After all of the service bundles are defined, the marketing manager can create marketing bundles or promotions to group services and products that are to be sold together as promotions. The promotions definition offers the flexibility to be upgraded to other promotions.

Here is an example of a marketing bundle for a wireless promotion with SMS:

| | |
|---|---|
| 1 | Nation 550 Minutes |
| 1.1 | Wireless Plan |
| 1.1.1 | Wireless Service |
| 1.1.1.1 | Basic Wireless 550 |
| 1.1.1.2 | Friends |
| 1.1.1.3 | Wireless Voice Service Feature |
| 1.1.1.3.1 | Wireless Voice Mail |
| 1.1.1.3.2 | Wireless Call Conference |
| 1.1.1.3.3 | Wireless Caller ID |

| | |
|---|---|
| 1.1.1.3.4 | Wireless Call Waiting |
| 1.1.1.3.5 | Wireless Call Forwarding |
| 1.1.1.4 | Text Messaging |
| 1.1.1.4.1 | Text Messaging SMS 200 |
| 1.1.1.4.2 | Text Messaging Usage |
| 1.2 | 50% Activation Discount |

The definition of marketing bundles is also used as a grouping for balance groups. For example, each promotion will define the boundaries of a balance group such that each included service bundle's service uses shared resources.

By using the communications product bundling methodology, promotion variants can be created by reusing the same nonservice bundle customizable products or service bundles if the bundles have options as components.

**Note:** Options are defined as a class-type relationship with the product that represents the options that are included in the relationship domain in Siebel CRM.

The same service bundle can be used to create promotion variants. This will ensure that the service is not disconnected during promotion upgrade or downgrade.

**For more information**, see section "Product Definition Methodology for Friends and Family: Example for more promotion variants created by reusing the service bundles.

# Balance Groups

Balance groups are defined at the plan level in Oracle BRM, and plans are not synchronized with Siebel CRM. As delivered, the solution does not provide design-time support for balance groups. When the order is interfaced to Oracle BRM for billing, it uses the default account-level balance group.

# Credit Limits

Because credit limits are typically defined at the billing-plan level in Oracle BRM, and such plans are not synchronized, customers can optionally define the default credit limits for each separate service type. As delivered, the solution does not support overrides of credit limits at either bundling or order capture time.

# Promotion Penalty, Service Activation, and Other MACD Charges

Penalty charges for promotion cancellation, upgrade, or downgrade need to be defined as an item type product with a charge in Oracle BRM and synchronized to Siebel CRM. The Siebel CRM promotion disconnect workflow process (ISS Promotion Disconnect Process) must be modified to use the product synchronized from Oracle BRM.

**For more information** about ISS Promotion Disconnect Process, see the *Siebel Order Management Guide Addendum for Communications*, "Workflows for Employee Asset-Based Ordering."

You can additionally define proration plans in Siebel CRM to prorate the penalty charge. During the order process when a promotion is cancelled, upgraded, or downgraded, Siebel CRM automatically adds the product (for the penalty charge) with the appropriate charge amount onto the order.

To support the application of charges for Move, Add, Change, and Disconnect (MACD) actions such as service suspend, resume, move, and cancel, the solution does not rely on the native Oracle BRM event mappings and charge application for such events. Instead, it simulates the application of such charges by relying on Siebel Related Product functionality. This facilitates visibility of the charges on the MACD and change order. Therefore, charges for suspend, resume, move, or disconnect must be defined as item type products in Oracle BRM (for every service type that you want to enable such charge application) and synchronized to Siebel CRM. You must then associate these products to the respective service bundles for the various actions (suspend, resume, move, disconnect) as related products.

**For more information** about Related Product functionality in Siebel, see the *Siebel Order Management Guide Addendum for Communications*, "Employee Asset-Based Ordering".

When a service is suspended, resumed, moved, or disconnected, Siebel CRM automatically adds the appropriate product (for the MACD charge) onto the order.

To model activation charges for a service:

1. Go to oracle BRM and define an item type product with a one-time charge.
2. Synchronize this item type product to Siebel CRM.
3. Go to Siebel CRM and set the **Track as Asset** flag to *Y*.

# Supporting Friends and Family

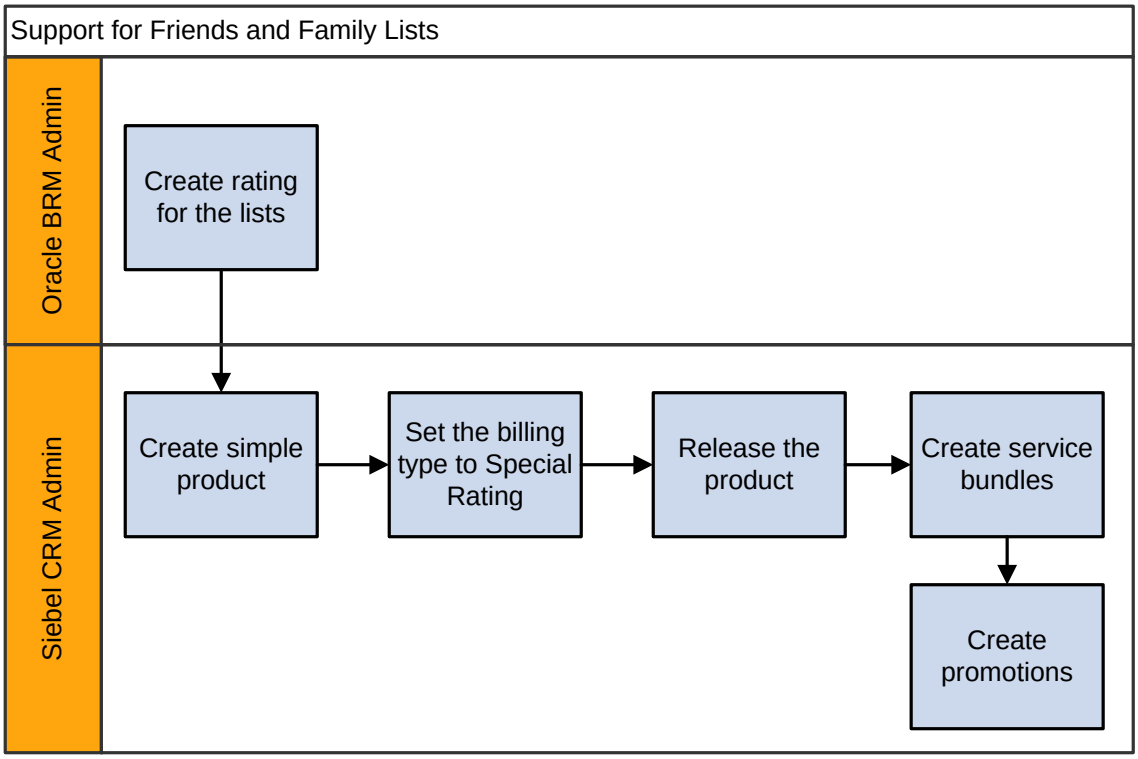
The Friends and Family feature supports the ability to rate calls to certain phone numbers differently from others.

Special rating products and Special rating profile lists in Siebel CRM are used to associate friends and family lists to services. Discounted rating for Friends and Family lists is defined in Oracle BRM.

Special rating products need to be manually defined in Siebel CRM, included in the service bundle along with the usage-based subscription product, and eventually added into the promotion during product modeling. When a promotion is purchased, the customer service representative (CSR) associates lists to the special rating products and optionally adds numbers to the lists. After the order is fulfilled and completed, the customer can update their Friends and Family lists.

**For more information** about how the lists are created and associated with the list product during run time, see Supporting Friends and Family Lists and Synchronizing Friends and Family List Updates to Oracle BRM.

This diagram shows the business process task flow for friends and family:




Business process task flow

## Design Time Setup

To enable the Friends and Family support:

You must perform the following in Oracle BRM:

- Define discounted pricing for Friends and Family lists. This involves specifying a label name for each list type defined in billing.

**Note:** The solution does not use the Oracle BRM Provisioning Tag Framework to support the Friends and Family feature.

**For more information,** see the Oracle *BRM Documentation* for “Working with Extended Rating Attributes” and “About rating based on Friends and Family ERA.”

You must perform the following in the Siebel CRM Project Workspace:

1. Create a simple product with a name that is identical to the list label name used in Oracle BRM (while defining the discounted pricing for the lists).
2. Set the billing type of the product to be *Special Rating*.
3. Leave the billing service type blank.

**Note:** This allows you to use the same special rating product across different types of services (such as Wireless and VoIP) for which you want to enable Friends and Family.

4. Set the billable flag to Y
5. Set the track as asset flag to Y
6. Add the special rating products to the service bundle that represents the service that supports friends and family lists. This service bundle must include a usage-based subscription product that is used to rate service usage.
7. Include the service bundle in the desired promotion(s) and release all the entities.

**For more information,** see the sections on friends and family plans in the "Profiles in Siebel Communications," chapter of the *Siebel Communications Guide*.

## Product Definition Methodology for Friends and Family: Example

The following is an example of the product definition methodology.

### Oracle BRM Definition

| Products in Oracle BRM | Service Type |
|---|---|
| Basic Wireless 550 | /service/telco/gsm/telephony |
| Monthly Cycle Forward Event | |
| Delayed Telco GSM Event | |
| Premium Wireless 800 | /service/telco/gsm/telephony |
| Monthly Cycle Forward Event | |
| Delayed Telco GSM Event | |
| Unlimited Wireless Voice | /service/telco/gsm/telephony |
| Monthly Cycle Forward Event | |
| Delayed Telco GSM Event | |
| Wireless Add On Line | /service/telco/gsm/telephony |
| Monthly Cycle Forward Event | |
| Product Purchase Fee Event | |
| Delayed Telco GSM Event | |
| Wireless Voice Activation | /service/telco/gsm/telephony |
| Product Purchase Fee Event | |
| Wireless Voice Mail | /service/telco/gsm/telephony |
| Wireless Call Conference | /service/telco/gsm/telephony |

| Products in Oracle BRM | Service Type |
|---|---|
| Wireless Caller ID | /service/telco/gsm/telephony |
| Wireless Call Waiting | /service/telco/gsm/telephony |
| Wireless Call Forwarding | /service/telco/gsm/telephony |
| Text Messaging SMS 200 | /service/telco/gsm/sms |
| Text Messaging SMS 400 | /service/telco/gsm/sms |
| Text Messaging SMS Unlimited | /service/telco/gsm/sms |
| Text Messaging Usage | /service/telco/gsm/sms |
| 50% Activation Discount | /account |

Define discounted pricing in Oracle BRM for rating phone numbers on the Special Rating lists. Use the labels *Friends* and *Family*.

## Siebel CRM Representation

| Product Name | Service Type | Billing type | Comments |
|---|---|---|---|
| Basic Wireless 550 | /service/telco/gsm/telephony | Subscription | Automated |
| Premium Wireless 800 | /service/telco/gsm/telephony | Subscription | Automated |
| Unlimited Wireless Voice | /service/telco/gsm/telephony | Subscription | Automated |
| Wireless Add On Line | /service/telco/gsm/telephony | Subscription | Automated |
| Product Purchase Fee Event | /service/telco/gsm/telephony | Event | Automated |
| Wireless Voice Activation | /service/telco/gsm/telephony | Item | Automated |
| Wireless Voice Mail | /service/telco/gsm/telephony | Subscription | Automated |
| Wireless Call Conference | /service/telco/gsm/telephony | Subscription | Automated |
| Wireless Caller ID | /service/telco/gsm/telephony | Subscription | Automated |
| Wireless Call Waiting | /service/telco/gsm/telephony | Subscription | Automated |
| Wireless Call Forwarding | /service/telco/gsm/telephony | Subscription | Automated |
| Text Messaging SMS 200 | /service/telco/gsm/sms | Subscription | Automated |
| Text Messaging SMS 400 | /service/telco/gsm/sms | Subscription | Automated |
| Text Messaging SMS Unlimited | /service/telco/gsm/sms | Subscription | Automated |
| Text Messaging Usage | /service/telco/gsm/sms | Subscription | Automated |
| 50% Activation Discount | /account | Discount | Automated |
| Friends | Not applicable | Special Rating | Manually Created |
| Family | Not applicable | Special Rating | Manually Created |

Here are some of the examples of the service bundles that include special rating products:

Service Bundles (SB)

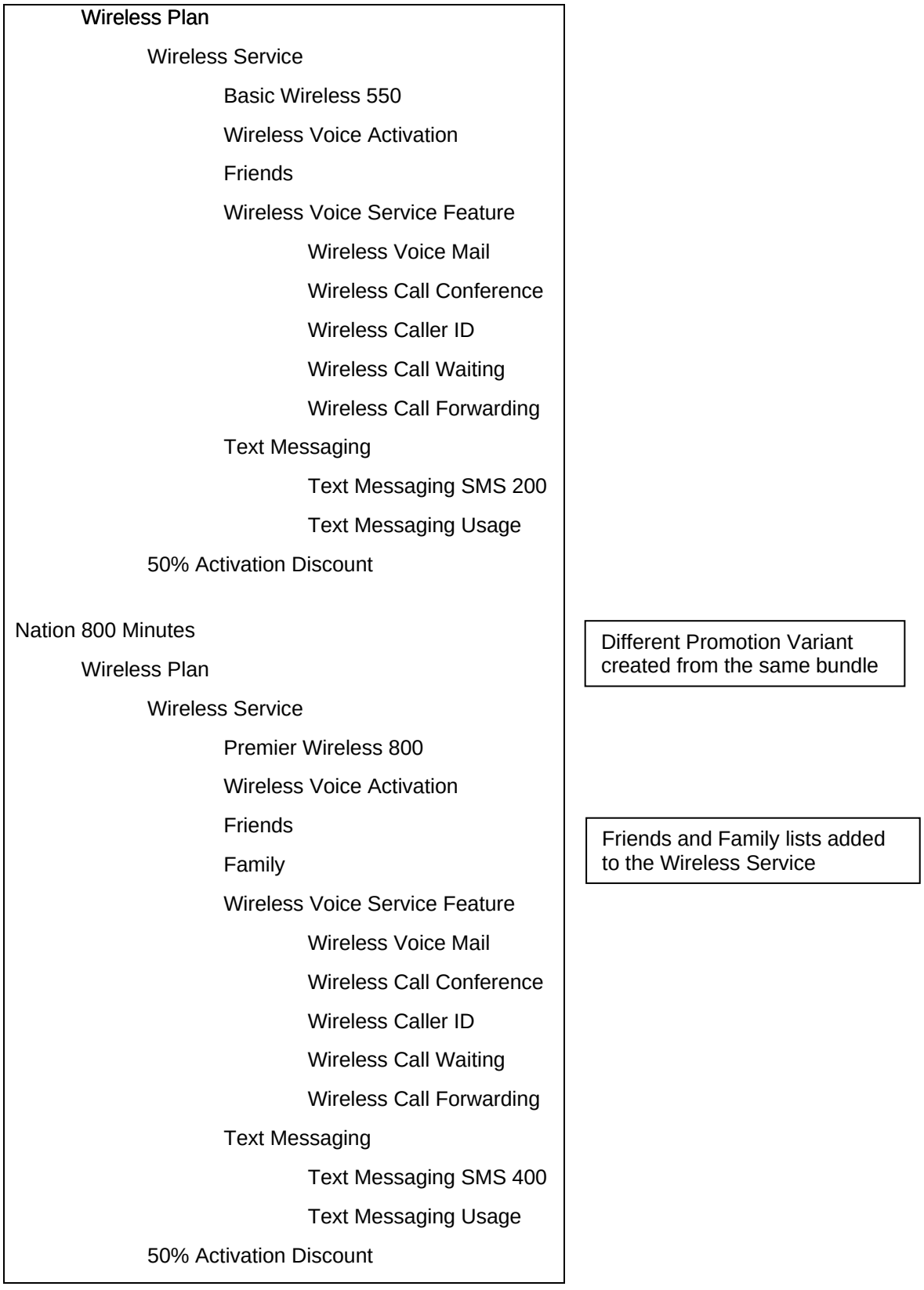
| | Comments |
|---|---|
| Wireless Service | Nested Service Bundle |
| Voice Access Options | Relationship of domain type = "Dynamic Class" and the components represent the options |
| Basic Wireless 550 | |
| Premium Wireless 800 | |
| Unlimited Wireless Voice | |
| Wireless Add On Line | |
| Wireless Voice Activation | |
| Special Rating Options | Relationship of domain type = "Dynamic Class" and the components represent the options |
| Friends | |
| Family | |
| Wireless Voice Service Feature | |
| Wireless Voice Mail | |
| Wireless Call Conference | |
| Wireless Caller ID | |
| Wireless Call Waiting | |
| Wireless Call Forwarding | |
| Text Messaging | |
| Text Messaging Options | Relationship of domain type = "Dynamic Class" and the components represent the options |
| Text Messaging SMS 200 | |
| Text Messaging SMS 400 | |
| Text Messaging SMS Unlimited | |
| Text Messaging Usage | |

**Note:** If multiple special rating products are bundled within the same service bundle, it is recommended that they be first grouped into a dynamic class and then included in the service bundle.

Here are some of the examples of the promotion definition:

## Promotions

| |
|---|
| Nation 550 Minutes |



# Supporting Time-Based Offerings

The time-based offerings (TBO) feature enables customers to define and use products and discounts in Siebel CRM that are valid only for a specific period of time, and expire after that.

Consider a use case in which a service provider wants to offer two promotions

- Gold Plan, which provides 50% discount on monthly cycle fee for the first three months
- Silver Plan, which provides 50% discount on monthly cycle fee for the first two months

Service providers can create a product in Oracle BRM that grants 1000 free minutes. This product is added to the Gold Plan by setting the attribute value for Duration to 3 and UOM to months. In case of the Silver Plan, the attribute values for Duration and UOM would be set to 2 and months respectively.

Similarly, using time-based offering, one could model absolute or percentage-based discounts with limited validity. In case of a TBO upgrade or a downgrade scenario, the attribute ValidityDurationStart is used to calculate the new service end date.

TBOs are also described in Chapter 4: Understanding the Process Integration for Order Management, "Supporting Time-Based Offerings".

## Time-Based Offerings Methodology

The two components of the solution for time-based offerings are design time and order time. They are defined as follows:

- Design Time
  - A custom product class with three validity attributes (Duration, Unit of Measure DurationUnitOfMeasure, and DurationValidityStart Validity Start) is created in Siebel.

    These attribute names must exactly match the names specified above.
  - Products and discounts that have validity requirements are manually changed from simple products to customizable products.
  - The product type for the products in Siebel is *Time Based Offering*.
  - The validity product class is associated with the products and discounts.
  - The validity attribute values are supplied in the context of bundles or promotions in Siebel.
- Order Time (for a new order or a revision order).

### Passing the End Date Value to Oracle BRM

In Siebel, a product class with new attributes must be created. The product class contains three new attributes:

- Duration: Used to define the number of days, months, or years.
- Unit DurationUnitOfMeasure: The unit used to measure the duration (days, months, or years).
- Validity Start DurationValidityStart: Indicates which date to calculate the end date from. The

valid values for this attribute are: Original Start, Order Time, and Original End.

The DurationValidityStart determines how the end date is calculated in a change order scenario. During a change order, if the duration of the product is changed, the new duration is calculated based on the original start date, the current date, or the original end date based on the value of the attribute.

**For more information** about Oracle Service Management (Oracle OSM), see *Oracle Communications Order and Service Management, Application Integration Architecture Order to Activate Cartridge Guide*.

**For more information** about Oracle BRM, see *Oracle Communications Billing and Revenue Management (BRM) Documentation*.

## Using Time-Based Offerings

Here is the methodology to use TBOs:

- The products that are valid should be changed manually to a customizable product in Siebel. This is because simple products cannot be associated with a product class.
- A custom product class with three validity attributes (Duration, DurationUnitOfMeasure, and DurationValidityStart) is created in Siebel CRM. These attribute names must exactly match the names that are specified in the following example:

| Attribute Name | ValueSet | Attribute Type | LOV |
|---|---|---|---|
| Duration | | int | -- |
| DurationUnitOfMeasure | UnitOfMeasure | enum | 0 – none<br>1 – seconds<br>2 – minutes<br>3 – hours<br>4 – days<br>5 – months |
| DurationValidityStart | ValidityStart | enum | 0 – Original Start<br>1 – Now<br>2 – Original End |

**For more information** about billing dates, see Appendix I: Mapping Billing Dates.

- Associate the previously created class with time-based products and discounts in Siebel.
- Provide values for the validity attributes for products and discounts in the context of promotions and bundles.
- Siebel supports product class hierarchy. The methodology covers the aspects of using the validity attributes in the class hierarchy scenario.

**Note:** In Siebel, a product can be associated with a single product class. If class inheritance is not used, validity class may clash with other product classes that are required to support other features. Because the Oracle Order to Activate PIP uses product class in the fulfillment flow, you need to follow the class hierarchy to support time-based offerings in the Oracle Order to Activate PIP.

- The implementer in Oracle OSM must use the attribute names used in Siebel to retrieve the values of the validity attributes.

**For more information** about how to create products and discounts in Siebel, see the Siebel product documentation.

# Chapter 3: Configuring the Process Integration for Product Lifecycle Management

This chapter discusses how to:

- Set up Fusion Middleware (FMW).
- Set up Oracle Communications Billing and Revenue Management (Oracle BRM).
- Set up Siebel Customer Relationship Management (Siebel CRM).
- Work with domain value maps (DVMs).
- Work with cross-references.
- Handle error notifications.
- View enterprise business object (EBO) implementation maps (EIMs).
- Configure the process integration for product lifecycle management.

## Setting Up FMW

Perform these steps to set up Fusion Middleware:

1. Add the -DHTTPClient.disableKeepAlives=true property in the opmn.xml file. The opmn.xml file is available here: SOA_HOME/opmn/conf/opmn.xml.

   After you add this property, service oriented architecture (SOA) does not use the same transmission control protocol (TCP) connection more than once to call a Siebel web service. Simultaneous calls to the same Siebel web service cause errors if the same TCP connection is used. This property should be added in the startup parameters of oc4j_soa. After you add the property, the opmn.xml file looks like this:

```
<process-type id="oc4j_soa" module-id="OC4J" status="enabled">
               <module-data>
                  <category id="start-parameters">
                     <data id="java-options" value="-server -
Xmx2048M -Xms2048M -XX:MaxPermSize=1024M -XX:MaxNewSize=614m -
XX:NewSize=614m -XX:AppendRatio=3 -XX:SurvivorRatio=6 -
Djava.security.policy=$ORACLE_HOME/j2ee/oc4j_soa/config/java2.policy
-Djava.awt.headless=true -Dhttp.webdir.enable=false -
Doraesb.home=/slot/ems1936/oracle/product/10.1.3.1/OracleAS_1/integr
ation/esb -Dhttp.proxySet=false -Doc4j.userThreads=true -
Doracle.mdb.fastUndeploy=60 -Doc4j.formauth.redirect=true -
Djava.net.preferIPv4Stack=true -
Dorabpel.home=/slot/ems1936/oracle/product/10.1.3.1/OracleAS_1/bpel
-
Xbootclasspath^/p:/slot/ems1936/oracle/product/10.1.3.1/OracleAS_1/b
pel/lib/orabpel-boot.jar -Dhttp.proxySet=false -
```

```
Daia.home=/slot/ems1936/oracle/product/AIA_HOME" -
DHTTPClient.disableKeepAlives=true/>
                  </category>
                  <category id="stop-parameters">
                     <data id="java-options" value="-
Djava.security.policy=$ORACLE_HOME/j2ee/oc4j_soa/config/java2.policy
-Djava.awt.headless=true -Dhttp.webdir.enable=false" />
                  </category>
               </module-data>
               <start timeout="600" retry="2" />
               <stop timeout="120" />
               <restart timeout="720" retry="2" />
               <port id="default-web-site" range="12501-12600"
protocol="ajp" />
               <port id="rmi" range="12401-12500" />
               <port id="rmis" range="12701-12800" />
               <port id="jms" range="12601-12700" />
               <process-set id="default_group" numprocs="1" />
            </process-type>
```

***Warning:*** While adding the -DHTTPClient.disableKeepAlives=true property in the opmn.xml file, you must be careful. The opmn.xml file is required for the SOA server startup and any error in this file may cause the server to fail.

2. To prevent multiple retries by the transaction manager:

   Modify the file $ORACLE_HOME/j2ee/{oc4j_instance name}/config/transaction-manager.xml to change the property value retry-count of commit-coordinator from 4 to 0:
   <commit-coordinator retry-count="4"> to <commit-coordinator retry-count="0">

3. To prevent the retries by the enterprise service bus (ESB):
   Overwrite the ESB retry entries to disable the retries in the orion-application.xml file present in the directories -
   $ORACLE_HOME/j2ee/oc4j_soa/application-deployments/esb-rt and $ORACLE_HOME/j2ee/oc4j_soa/application-deployments/esb-dt with the following values after backing up the original file.

   ```
   <property name="InboundRetryCount" value="0" />
   <property name="InboundRetryInterval" value="0" />
   <property name="InboundRetryEnabled" value="false" />
   <property name="OutboundRetryCount" value="0" />
   <property name="OutboundRetryInterval" value="0" />
   <property name="OutboundRetryEnabled" value="false" />
   ```

   Alternatively, change these property values in the $ORACLE_HOME//integration/esb/config/esb_config.ini to change the retry values

These steps are also included in the *Oracle AIA: Installation and Upgrade Guide*.

# Setting Up Oracle BRM

These are the setup steps for Oracle BRM:

1. Create services and events.

New services have to be added before a pricelist is created. Out-of-the-box (OOTB), Oracle BRM provides a set of services. A list of events must be configured to track each service. If new services are created, new events must be created to track the services.

2. Create resources.

   Each product is associated with rate plans. Resources must be created to supplement the rate plans. These include both the currency, such as USD, and the noncurrency-related resources, such as minutes.

3. Create General Ledger (GL) IDs.

   GL IDs are used to collect general ledger information from the Oracle BRM database and export it to your accounting application. Decide how to track the revenue for each type of rate, and create the appropriate GL IDs.

4. Define tax codes and tax suppliers. (Optional)

   To calculate taxes using Taxware, you need to define tax codes and tax suppliers.

5. Define ratable usage metrics (RUMs) for events.

   RUMs are used to identify the event attributes that define rates for each event. RUM definitions are stored in the Oracle BRM database.

6. Map event types to RUMs.

   Each event has to be associated with a list of RUMs. When products are created, a rate plan structure is associated with every RUM that is linked for the event.

7. Map event types to services.

   When a product is created, a set of services and events that have to be rated are selected. The events are related to the service. Not all event types are valid for all services. A mapping must be defined between the event types and the services. Creating the mapping prevents you from selecting an event that is not applicable for a given service.

8. Define zones.

   For real-time rating, zones are created as single values to represent groups of values. The representative value is used in a rate plan selector.

9. Define impact categories.

   For real time rating, impact categories are used to specify that particular groups of balance impacts within rates must be used. If the plan is to use attribute value grouping during rating, then some impact categories must be created.

10. Define pipeline data.

    If pipeline rating is used, several types of data and pricing components must be created.

11. Set up pricing for friends and family functionality.

**For more information** about setting up pricing for friends and family, see *Oracle Communications Billing and Revenue Management (BRM) Documentation*, "Setting Up Pricing and Rating," Working with extended rating attributes.

12. Install, configure, and run Synchronization Queue Data Manager (DM).

    This DM enables you to synchronize changes in the Oracle BRM database with external applications. For example, when a product is created or modified, Synchronization Queue DM sends the data to a database queue. The data in the queue can then be retrieved by an external application. You can use the Synchronization Queue DM to synchronize data in real time, and you can use it in conjunction with the pin_export_price utility to export data as a batch.

**For more information,** see *Oracle Communications Billing and Revenue Management (BRM) Documentation*, "Service Integration Components," Synchronization Queue Data Manager.

# Setting Up Siebel CRM

For some Siebel CRM interfaces, in Siebel, you must set the process property UTCCanonical to Y.

**For more information** about which Siebel CRM interfaces require you to enable the UTCCanonical process property, see instructions for ACR 474 and 508 in the *Siebel Maintenance Release Guide*.

These are the setup steps for Siebel CRM:

1. Set up a Siebel price list.

   The price list is needed for the product synchronization integration flow. Create a price list in Siebel, and then update the AIAConfigurationProperties.xml file with the name of the price list.

2. Set up a Siebel business unit.

   Identify the business unit in Siebel and update the AIAConfigurationProperties.xml file.

3. Set up a Siebel workspace.

   Identify that workspace in Siebel and update the AIAConfigurationProperties.xml file.

4. Set up friends and family products.

**For more information,** see Support for Friends and Family in Chapter 2: Understanding the Product Bundling Methodology.

5. Make workflow changes to use penalty products synchronized from Oracle BRM.

   This can be done only after you run the product synchronization integration flow.

   Onetime charge products need to be included in the Siebel Catalog. If not, you will not see the onetime charge recommended products pick list.

After products are synchronized from Oracle BRM to Siebel CRM, and after onetime charge products have been added to a Siebel Catalog, you must associate onetime charges with MACD order types.

Define simple Special Rating products and set the composition type to Partial.

6. Set up service bundles:

   Set Billing Type to Service Bundle and set Billing Service Type to the same string as the billing service bundle on the component products (that have been synced from Oracle BRM).

   For nested service bundles (the root non service bundle customizable product (NSBCP) or the root service bundle (SB)) set the success dependency.

   Set the composition type.

7. Set up promotions, bundling together service bundles, account-level products, and discounts.

   Define success dependency for promotion products.

   Set the composition type.

8. Add service bundles and promotions to the price list used by the product synchronization integration flow.

**For more information** about service bundles, see Chapter 2: Understanding the Product Bundling Methodology

# Working with DVMs

Domain value maps (DVMs) are a standard feature of the Oracle SOA Suite. They enable you to equate lookup codes and other static values across applications, for example, FOOT and FT or US and USA.

DVMs are static in nature, though administrators can add maps as needed. Transactional business processes never update DVMs; they only read from them. DVMs are stored in XML files and cached in memory at run time.

DVM types are seeded for the Order to Bill flows, and administrators can extend the list of mapped values by adding more maps. The DVM data should be synchronized with what the participating applications use. This synchronization should occur before any initial loads are run or any incremental transactional flows are initiated.

These are the DVMs for the process integration for product management:

| DVM | Description |
|---|---|
| PRICECHARGETYPE | Price Charge Type (Common Values Are One-Time, Or Recurring.) |
| PRICECHARGETYPEUOM | Price Charge Type Unit Of Measure (Common Values Are Per Day, Or Per Month.) |
| PRICETYPE_EVENT | Price Type Event (Common Values Are Purchase, Or Cancel.) |
| PRODUCTTYPECODE | Product Type Code (Common Values Are Item, Or Subscription.) |
| ITEM_BILLINGTYPECODE | Maps Billing Type from BRM to CRM |

| DVM | Description |
|---|---|
| RESOURCE | Non-Monetary resources (Free Minutes, Text Messages, and so on). |
| CURRENCY_CODE | Currency codes. |

**For more information** about DVMs, see *Oracle Application Integration Architecture – Foundation Pack 2.5: Integration Developer's Guide*, "Understanding Message Transformation, Enrichment, and Configuration," DVMs.

# Working with Cross-References

Cross-references map and connect the records within the application network, and they enable these applications to communicate in the same language. The integration server stores the relationship in a persistent way so that others can refer to it.

This table lists the product management cross-references:

| Cross-Reference Table Name | Column Names Column Values | | | Description |
|---|---|---|---|---|
| ITEM_ITEMID | COMMON | SEBL_01 | BRM_01 | Cross references the Oracle BRM (Portal) ProductID and the Siebel CRM ProductID |
| | Auto-generated GUID | ProductID of Siebel Product ABM | POID of Oracle BRM Product ABM | |
| PRICELINE_ID | COMMON | SEBL_01 | BRM_01 | Cross references the Oracle BRM (Portal) Product ID to Siebel CRM PriceLineID. Also links to the COMMON of ITEM_ITEMID. |
| | Auto-generated GUID | Siebel PriceListItemID for the main product | POID of Oracle BRM Product ABM | |
| PRICELINETYPE_ID | COMMON | SEBL_01 | BRM_01 | Cross references Oracle BRM (Portal) Product's Events to Siebel CRM PriceLineID. Also links to the COMMON of ITEM_ITEMID. |
| | Auto-generated GUID | Siebel PriceListItemID for the event product | POID of Oracle BRM Product ABM + Event Name | |
| SIEBELPRODUCTEVENTXREF | ITEM_ID_COMMON | LINEPRICETYPECODE | -- | Cross references Oracle BRM (Portal) Product's Event that is associated with the main product in Siebel CRM. |
| | From ITEM_ID.COMMON | PRICELINETYPE_ID.COMMON | -- | |

**For more information** about product management cross-references, see Appendix D: Cross-References for the Process Integration for Product Management.

# Handling Error Notifications

Based on the roles defined for the services, email notifications are sent if a service ends due to an error. No Oracle Application Integration Architecture (Oracle AIA)-specific errors are caused by the process integration for product management services.

**For more information** about the errors caused by Oracle BRM or Siebel CRM, see that product's documentation.

## Describing Delivered Error Notification Roles and Users

The following roles and users are delivered as default values for issuing error notifications for the process integration for product management.

Actor roles and users:

- Role: AIAIntegrationAdmin. User: AIAIntegrationAdminUser.

The default password set for all users is welcome1.

**For more information** about setting up error notifications using these values, see *Oracle Application Integration Architecture — Core Infrastructure Components Guide,* "Understanding Error Handling and Logging."

# Viewing EBO EIMs

**For more information** about how services are mapped, see the My Oracle Support document: EBO Implementation Maps (EIMs) 881022.1.

# Configuring the Process Integration for Product Lifecycle Management

Configure these properties in the AIAConfigurationProperties.xml file. It is located here: <aia.home>/config/. Entries in the AIAConfigurationProperties.xml file are case sensitive.

**Note:** Whenever the AIAConfigurationProperties.xml file is updated, the file must be reloaded for updates to be reflected in the applications or services that use the updated properties. Click the Reload button on the Configuration page in the Oracle AIA Console to perform this reload. Alternatively, you can perform the reload by restarting the server.

**For more information**, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide*, "Loading Oracle AIA Configuration File Updates."

Settings for the SyncProductPortalReqABCSImpl service property:

| Property Name | Value/Default Values | Description |
|---|---|---|

| | | |
|---|---|---|
| Sender.Default.SystemID | BRM_01 | Default target billing system instance code (defined in the business service repository (BSR). This is used only in the event that the request message does not contain the system instance ID. |
| Routing.ItemEBS.RouteToCAVS | True/False. Default = false. | Controls whether ItemEBS routes messages to the verification system or to the Provider application business connector service (ABCS) implementation. |
| Routing.PriceListEBS.RouteToCAVS | True/False. Default = false. | Controls whether PriceListEBS routes messages to the verification system or to the Provider ABCS implementation. |

Settings for the SyncDiscountPortalReqABCSImpl service property:

| Property Name | Value/Default Values | Description |
|---|---|---|
| Sender.Default.SystemID | BRM_01 | Siebel system instance code (defined in BSR) from which messages originate. If the instance ID is present in the request message, then that takes precedence. |
| Routing.ItemEBS.RouteToCAVS | True/False. Default = false. | Controls whether ItemEBS routes messages to the verification system or to the Provider ABCS implementation. |

Settings for the SyncItemPublicationSiebelProvABCSImpl service property:

| Property Name | Value/Default Values | Description |
|---|---|---|
| Routing.Target.Default.SiebelProductService.SystemID | SEBL_01 | Siebel system instance code (defined in BSR). This is used only in the event that the request message does not contain the target system ID. |
| Routing.Target.Default.SiebelProductService.EndpointURI | No default value. | Siebel ProductImport web service end point location. This is a SOAP end point URL. If the request message contains the target URL, then that takes precedence. |
| Routing.SiebelProductService.RouteToCAVS | True/False. Default = false. | If true, it invokes the actual target system whose end point is indicated by the service-level property Routing.Target.Default.SiebelProductService.EndpointURI. If false, it invokes the verification system whose end point is indicated by the system-level property SyncResponseSimulator.Soap.EndpointURL. |

| Property Name | Value/Default Values | Description |
|---|---|---|
| Siebel.BusinessUnit | No default value. | All the products created will belong to this business unit in the Siebel system. The value for this property should be the ID of the business unit in the Siebel system. This value must be set before Product Sync is run. |
| Siebel.Product.WorkspaceName | Demo Workspace | Name of the workspace to be used by Siebel. Create a workspace and update this file with that workspace name. |
| Siebel.Product.WorkspaceReleaseFlag | Y/N. Default = N | Indicates whether the workspace needs to be released after the product is synchronized. |
| Siebel.Product.WorkspaceReuseFlag | Y/N. Default = Y | Indicates whether the workspace needs to be reused for product to be synced. |

Settings for the ProductOptimizedSyncPriceListLineListSiebelProvABCSImpl service property:

| Property Name | Value/Default Values | Description |
|---|---|---|
| Routing.Target.Default.SiebelProductService.SystemID | SEBL_01 | Siebel system instance code (defined in BSR). This is used only if the request message does not contain the target system ID. |
| Routing.Target.Default.SiebelProductService.EndpointURI | No default value. | Siebel ProductImport web service end point location. This is a SOAP end point URL. If the request message contains the target URL, then that takes precedence. |
| Routing.SiebelProductService.RouteToCAVS | True/False. Default = false. | If true, it invokes the actual target system whose end point is indicated by the service-level property Routing.Target.Default.SiebelProductService.EndpointURI. If false, it invokes the verification system whose end point is indicated by the system-level property SyncResponseSimulator.Soap.EndpointURL. |
| Routing.Target.Default.SiebelPriceListService.SystemID | SEBL_01 | Siebel system instance code (defined in BSR). This is used only if the request message does not contain the target system ID. |
| Routing.Target.Default.SiebelPriceListService.EndpointURI | No default value. | Siebel PriceList web service end point location. This is a SOAP end point URL. If the request message contains the target URL, then that takes precedence. |

| Property Name | Value/Default Values | Description |
|---|---|---|
| Routing.SiebelPriceListService.RouteToCAVS | True/False. Default = false. | If true, it invokes the actual target system whose end point is indicated by the service-level property Routing.Target.Default.SiebelPriceListService.EndpointURI. If false, it invokes the verification system whose end point is indicated by the system-level property SyncResponseSimulator.Soap.EndpointURL. |
| Siebel.BusinessUnit | No default value. | All the products created will belong to this business unit in the Siebel system. The value for this property should be the ID of the business unit in the Siebel system. This value must be set before product sync is run. |
| Siebel.Product.Workspace Name | Demo Workspace | Name of the workspace to be used by Siebel. Create a workspace and update this file with that workspace name. |
| Siebel.Product.Workspace ReleaseFlag | Y/N. Default = N | Indicates whether the workspace needs to be released after the product is synchronized. |
| Siebel.Product.WorkspaceReuseFlag | Y/N. Default = Y | Indicates whether the workspace needs to be reused for product to be synced. |
| Siebel.PriceList.ID | No default value. | All the products created by this sync belong to this price list in the Siebel system. The value for this property should be the ID of the price list in the Siebel system. This value must be set before product sync is run. |
| Siebel.PriceList.Currency | USD | Currency code of the price list mentioned in the preceding property. If the currency of the prices in PriceListEBM does not match this currency, price in Siebel will be set to 0 (zero). This value must be set before the product sync is run. |

# Part 2: Implementing the Process Integration for Order Management

This part includes the following chapters:

Chapter 4: Understanding the Process Integration for Order Management

Chapter 5: Configuring the Process Integration for Order Management

# Chapter 4: Understanding the Process Integration for Order Management

This chapter provides overview of process integration for order management and discusses:

- Business process flow overview.
- Data requirements.
- Submitting orders to order orchestration.
- Creating customer data in the billing system.
- Interfacing orders to billing.
- Supporting time-based offerings.
- Supporting friends and family lists.
- Sending order updates back to Siebel Customer Relationship Management (Siebel CRM).
- Creating or updating installed assets in Siebel CRM.
- Order management integration flows.
- Orchestrating the services.
- Solution assumptions and constraints.
- Oracle Communications Billing and Revenue Management (Oracle BRM) interfaces.
- Siebel CRM interfaces.
- Industry Application Integration Architecture (AIA) components.
- Integration services.

## Overview

The process integration for order management enables you to create an order-to-bill business process and delivers these integration points:

- Submitting orders from Siebel CRM to your central fulfillment system (CFS) for order fulfillment processing.

The following integration points can be called by your central fulfillment system to:

- Interfacing orders to create customer data in Oracle BRM.
- Interfacing orders to create transaction data in Oracle BRM.
- Updating status information on order lines in Siebel CRM.

A central fulfillment system, order management system, or order decomposition and orchestration process is not delivered in the Order to Bill PIP (though a test orchestration process (TOP) is delivered to enable installation verification.)

**Note:** You can interoperate this PIP with the Oracle Order to Activate PIP, which uses Oracle Communications Order and Service Management (Oracle OSM) for order orchestration and fulfillment.

**For more information** about using Oracle OSM as your orchestration process, see the *Oracle Communications Order and Service Management Application Integration Architecture Order-to-Activate Cartridge Guide*.

Asset integration is supported by Siebel CRM auto-asset functionality.

**For more information** about the order integration points, see Integration Flows.

Note that these terms are used in the documentation and graphics interchangeably:

- **Fulfillment or central fulfillment system (CFS)**: Order Management, Oracle OSM Central Order Management (Oracle OSM COM)
- **Provisioning**: Service Fulfillment, Oracle OSM Service Order Management (Oracle OSM SOM)

# Business Process Flow Overview

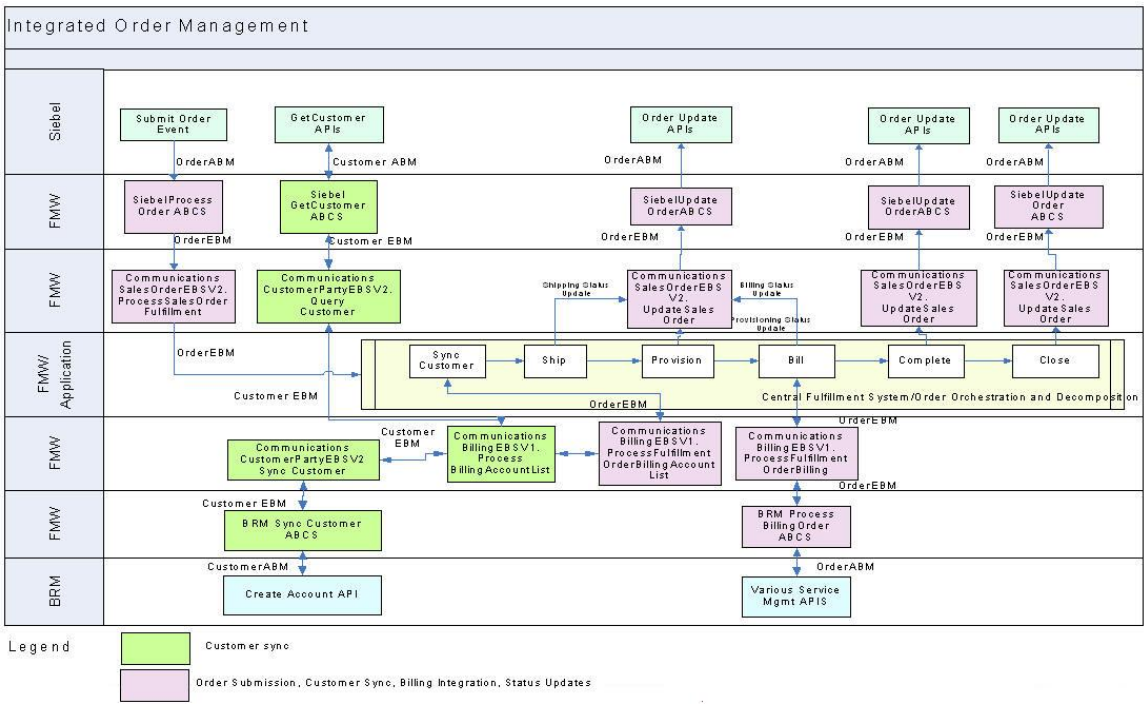
Customers contact customer service representatives (CSRs) to place orders for new services or to make changes to existing services. The CSR must first determine whether the caller is an existing customer. If the customer is new, the CSR needs to set up an account for the customer before placing an order. If a customer is calling to change an existing service, the CSR can query the asset representing the customer's existing service and then use what is known as asset-based ordering to modify or add to it. In this scenario, the CSR creates an order that references existing assets. When a CSR has captured an order, it is submitted for processing.

In Siebel CRM, the submit order event enqueues the Siebel order message (Siebel order ABM or Application Business Message) in a Java Message Service (JMS) queue. The JMSConsumer that listens to this queue dequeues the message, and then invokes the Siebel Application Business Connector Service (ABCS). After Siebel drops the message in the queue, the control is given back to the CSR, making the submit order event an asynchronous process.

The process integration for order management delivers:

- Services that can take a submitted Siebel order and call an order management system for order orchestration and fulfillment.
- Services that can be called from an order management system to perform various tasks, such as creating customer data in a billing system, interfacing the order to a billing system, and communicating status and other order updates back to Siebel CRM.

This diagram shows the overall flow for the process integration for order management:




Process integration for order management overall flow

# Data Requirements

These are the data requirements for this integration. These apply to Siebel orders submitted for processing:

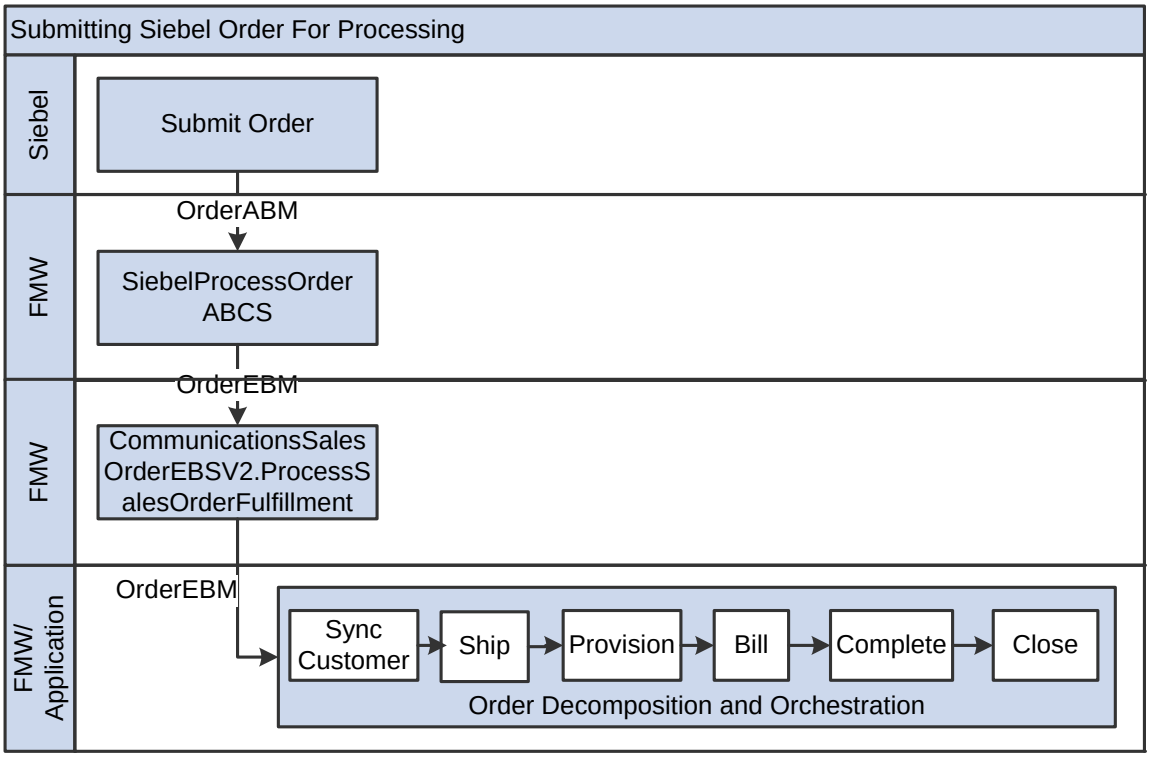
- An order must be of type Sales Order.
- The price list specified on the order must match the one created for the process integration for product lifecycle management. It is created in Siebel CRM and configured in the AIAConfigurationProperties.xml file.
- Service bundle lines or account-level product lines must have a service account, a billing account, and a billing profile.
- Service bundle lines and Simple Service bundle lines must have a service ID before they are interfaced to a billing system.
- Order lines referencing the same service account cannot reference different billing accounts. Refer to the solution constraint about having a single parent for subordinate accounts.
- On any new order or change order for a service account, if the billing account is different from the billing account used on a previous order for the same service account, then all existing services paid for by the original billing account must appear on the order as updates to be paid by the new billing account.

**For more information,** see Solution Assumptions and Constraints.

# Submitting Orders to Order Orchestration

After a new order or change order is entered and validated in the CRM system, the CSR submits it for further processing. The process integration for order management provides services that enable you to submit Siebel CRM orders to your order management system for order orchestration and fulfillment.

This diagram shows how orders are submitted to the order management system:




## Submitting orders for processing

The order orchestration process picks up and processes the orders from the queue.

These are the delivered services:

- ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer: This service dequeues the Siebel order message from the JMS queue and routes it to the Siebel connector service.
- ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl transforms a Siebel order message into an enterprise order message. It converts application-specific references to products and customer data to enterprise business object (EBO)-based references.

  The Siebel order structure does not support multiple charge types for a single order line, while the order EBO structure does. For this reason, the order lines referencing a Complex Product (CP) of billing type Subscription and its component products of billing type Event, together representing a multi-event billing product, are transformed into a single EBO order line referencing a product with multiple charge types.
- The CommunicationsSalesOrderEBSV2.ProcessSalesOrderFulfillment operation serves as a routing mechanism to decouple Siebel CRM from your Order Management system. As delivered, it calls the Test Orchestration Process (TOP). This process is meant only to serve

as a mechanism for you to test whether the PIP was successfully installed. You must configure the Enterprise Service Bus (ESB) routing rules to have the CommunicationsSalesOrderEBSV2.ProcessSalesOrderFulfillment call your order management system to orchestrate and fulfill the order.

These services delivered with the Order to Bill PIP are also used for order submission in the Oracle Order to Activate PIP.

**For more information** about these services, see the *Oracle Order to Activate Integration Pack for Siebel CRM and Oracle Communications Order and Service Management - Implementation Guide*.

# Creating Customer Data in Billing

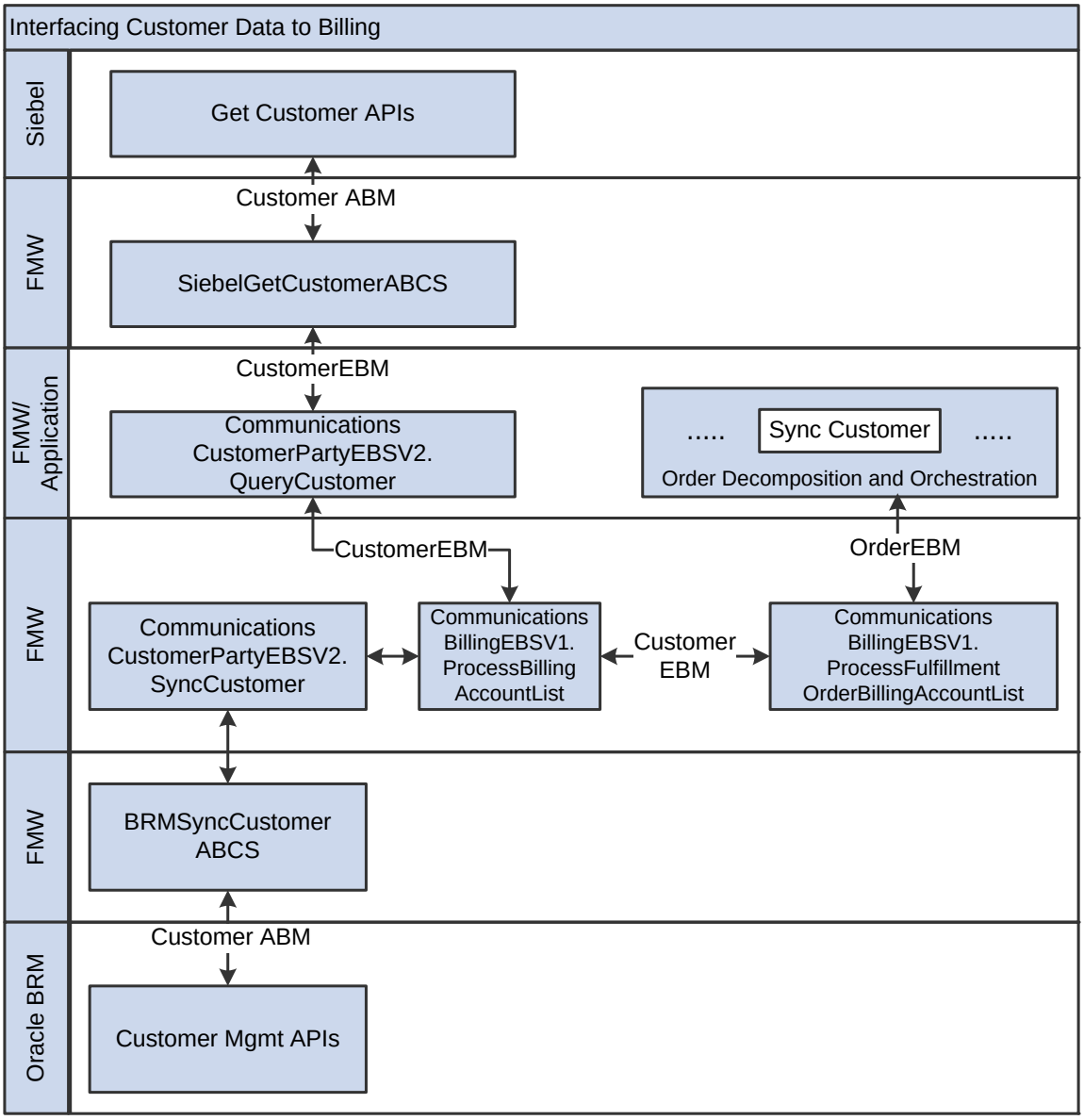
Service providers do not want to overburden their billing systems with all of the customer information in their CRM systems. Rather, they want the ability to create the necessary customer data in the billing system only when the customer places an order and the order is submitted for fulfillment.

The process integration for order management provides a service, CommunicationsBillingEBSV1.ProcessFulfillmentOrderBillingAccountList, that takes an order as input and calls the necessary enterprise billing service to create accounts and their components (such as billing preferences and payment methods) referenced on an order in a target billing system. This service can be invoked from the order management system to create customer data in the billing system.

The CommunicationsBillingEBSV1.ProcessFulfillmentOrderBillingAccountList service can be invoked from the order decomposition and orchestration process. This service calls the CommsProcessFulfillmentOrderBillingAccountListEBF using seeded ESB routing rules.

**For more information,** see Appendix B: Examples of Changing the Paying Parent on Subordinate Accounts and Appendix I: Mapping Billing Dates.

This diagram shows the order interface to customer data in billing:




Interfacing customer data in billing

The CommunicationsBillingEBSV1.ProcessFulfillmentOrderBillingAccountList service processes *only* lines with the actions of ADD, UPDATE, and MOVE-ADD and ignores the others. This service considers the following kinds of order lines for customer data collation:

- For lines whose billing type is Service Bundle, Item, Subscription, or Discount, it considers Service Account, Billing Account, and Billing Profile.
- For lines whose product type is Promotion, it includes only Billing Account.
- All other lines are ignored.

The end result of calling this service is the creation of customer data such as accounts, bill-infos, and pay-infos in Oracle BRM.

Customer creation that occurs in Oracle BRM as part of order fulfillment using the service InterfaceOrderToCustomerEBF cannot be undone:

- The service does not support the ability to inactivate or delete accounts, bill-infos, or pay-infos in Oracle BRM.
- Calling the CommunicationsBillingEBSV1.ProcessFulfillmentOrderBillingAccountList again with the same input as before has no effect.
- If the service is called with references to different customer data than before, the service detects the delta and creates just the account, bill-infos, and pay-infos that do not exist in BRM.

When the service account on a service bundle or account-level product line is different from the bill-to account, the service account is created as a nonpaying subordinate account under the bill-to account in Oracle BRM; that is, it results in the creation of a paying hierarchy in billing:

- A paying hierarchy, once created, cannot be undone simply by the cancellation of the original Siebel CRM order.
- If the service is called to update an existing paying hierarchy (for example, to set the paying account for a subordinate account to a different paying account), to undo that update (because the Siebel CRM order requesting the change was canceled), the order management system needs to rework the message such that it is a call to update the hierarchy to a previous state.

This table summarizes what is expected from calling the order management system in terms of action on the line:

| Original Action on Order Line | Is this the first time the order line is being processed by customer sync or is it a revision? | What is occurring on the revision (that is relevant to customer sync) | Expected Action on compensation order line, set by orchestration (Oracle OSM) | Comments |
|---|---|---|---|---|
| ADD | First time | Not applicable | ADD | -- |
| ADD | Revision | No changes to service account, billing account, or billing profile | NONE | No changes for customer sync to process. |
| ADD | Revision | Changes to service account, billing account, or billing profile | UPDATE | From a customer sync perspective, the fact that it is a revision is irrelevant in that it just checks whether the customer data referenced on the order exists in BRM; if not, it will create it.<br>If customer sync is using the original ADD |

| Original Action on Order Line | Is this the first time the order line is being processed by customer sync or is it a revision? | What is occurring on the revision (that is relevant to customer sync) | Expected Action on compensation order line, set by orchestration (Oracle OSM) | Comments |
|---|---|---|---|---|
| | | | | line, a billing hierarchy is created, and on the revision the attributes that affect the hierarchy are changing, then it will make the required change.<br>The calling order management system needs to indicate which attributes have changed by populating the prior value fields for the changed attributes.<br>Prior value fields are specifically used in flagging and determining that a paying hierarchy change has occurred. |
| ADD | Revision | Cancellation. Manifests as a missing line on the revision. | DELETE | This action is ignored by the customer sync.<br>If the ADD line added new account, bill-info, and pay-info, and then the request for a new purchase was canceled, then those entities are not inactivated or deleted.<br>If the ADD line created a paying hierarchy, and then the request for new purchase was canceled, then the paying hierarchy stays in place. |
| UPDATE | First time | Not applicable | UPDATE | Expects prior value fields to be populated. |
| UPDATE | Revision | No changes to service account, billing account, or billing profile | NONE | No changes for customer sync to process. |
| UPDATE | Revision | Changes to service account, billing account, or billing profile | UPDATE | From a customer sync perspective, the fact that it is a revision is irrelevant in that it just checks whether the new set of customer and billing profiles exist in Oracle BRM; if not, it will create it.<br>If customer sync is using the original UPDATE line a billing hierarchy is created or updated, and on the revision the attributes that affect the hierarchy are changing, then it will make the required change.<br>The calling order management system needs |

| Original Action on Order Line | Is this the first time the order line is being processed by customer sync or is it a revision? | What is occurring on the revision (that is relevant to customer sync) | Expected Action on compensation order line, set by orchestration (Oracle OSM) | Comments |
|---|---|---|---|---|
| | | | | to indicate which attributes have changed by populating the prior value fields for the changed attributes. |
| UPDATE | Revision | Cancellation. Manifests as a missing line on the revision or the action changing to a "-" (NONE). | UPDATE | If the original update line created a new account and billing profile in Oracle BRM, then it cannot be undone.<br>For the attributes that *had changed* on the original line, the calling order management system needs to flip the values (old, new) on the compensation line. For the case in which a hierarchy had been updated, this will in essence revert that update. |
| MOVE-ADD | First Time, but can change billing account and billing profile as part of a move-add. | Not Applicable | MOVE-ADD | Expects prior value fields to be populated for values that are changing from an existing asset. |
| MOVE-ADD | Revision | No changes to service account, billing account, or billing profile* | NONE | No changes for customer sync to process. |
| MOVE-ADD | Revision | Changes to service account, billing account, or billing profile* | MOVE-ADD | From a customer sync perspective, the fact that it is a revision is irrelevant in that it just checks whether the new set of customer and billing profiles exist in Oracle BRM; if not, it will create it.<br>If customer sync is using the original UPDATE line, a billing hierarchy is created or updated, and on the revision the attributes that affect the hierarchy are changing, then it will make the required change.<br>The calling order management system needs |

| Original Action on Order Line | Is this the first time the order line is being processed by customer sync or is it a revision? | What is occurring on the revision (that is relevant to customer sync) | Expected Action on compensation order line, set by orchestration (Oracle OSM) | Comments |
|---|---|---|---|---|
| | | | | to indicate which attributes have changed by populating the prior value fields for the changed attributes. |
| MOVE-ADD | Revision | Manifests as a missing line on the revision or the action changing to a "-"<br>(In essence the line is canceled). | MOVE-ADD | If the original MOVE-ADD line created a new account and billing profile in Oracle BRM, then it cannot be undone.<br>For the attributes that *had changed* on the original line, the calling order management system should flip the values (old, new) on the compensation line. For the case in which a hierarchy had been updated, this will in essence revert that update. |

* Billing integration supports only changes to billing account and billing profile as part of MOVE-ADD.

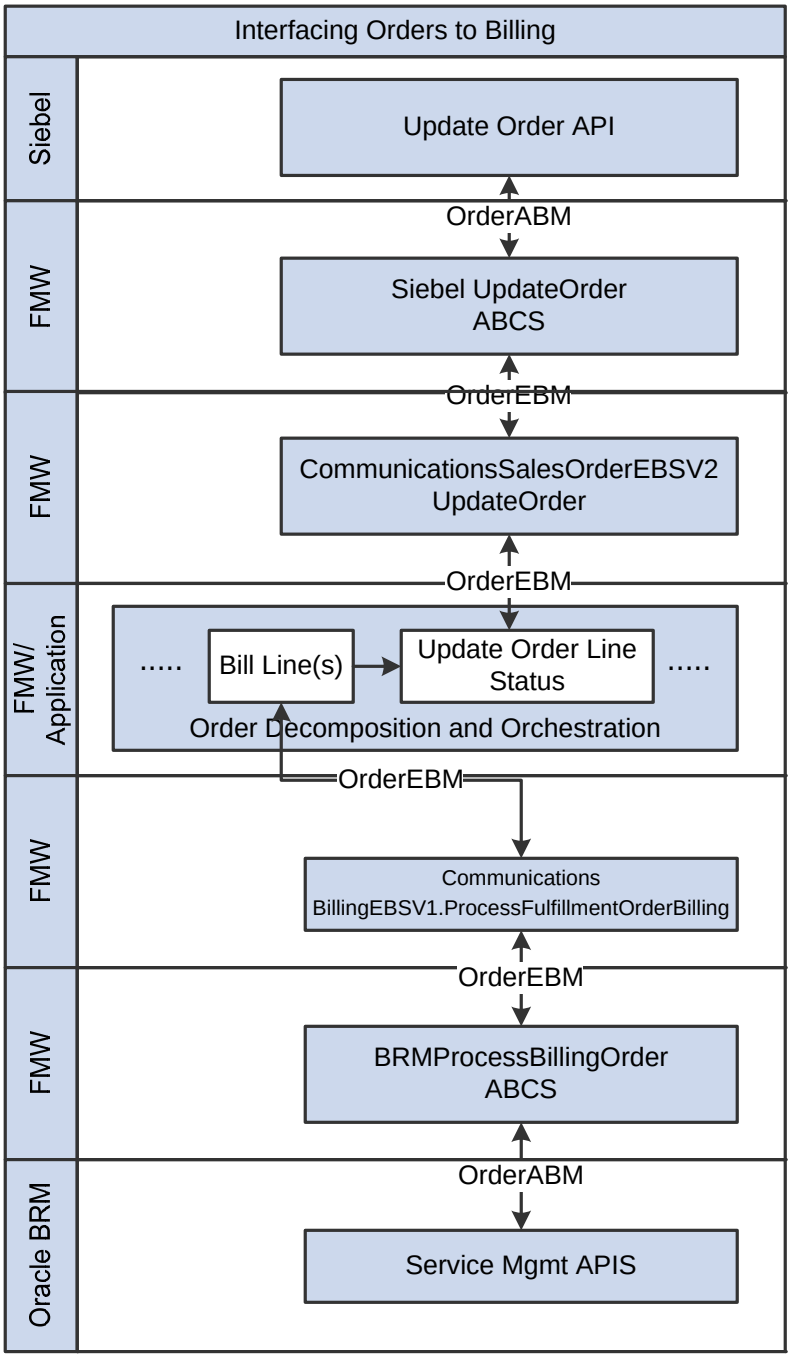
**Note:** The process integration for billing management (delivered in the Agent Assisted Billing Care PIP) assumes that a given billing profile is synchronized to a single billing system. It does not support the ability to query data for the same billing profile from more than one billing system. For that reason, if that process integration is in use, then the same billing profile should not be used on an order for services that are fulfilled in different billing systems.

**For more information** about this assumption, see the billing management chapter in the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Agent Assisted Billing Care Implementation Guide*.

# Interfacing Orders to Billing

A customer needs to be billed for service purchase and usage. The process integration for order management provides services to interface the order to billing so that the billing system can create the data it needs to bill the customer.

This diagram shows how orders interface to customer and billing:




## Interfacing orders to billing

These are the delivered services:

- CommunicationsBillingEBSV1.ProcessFulfillmentOrderBilling
- ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl

**For more information** about how AIA sets billing dates for the ProcessFulfillmentOrderBilling service, see Appendix I: Mapping Billing Dates.

As part of interfacing a new order or change order from Siebel CRM to Oracle BRM, the process integration for order management:

1. Supports the following in Oracle BRM:
   - Products of type item that apply to an account (for example, promotion penalty charges).
   - Products of type item that apply to a service (for example, one-time charges).
   - Products of type subscription that apply to an account (for example, charges for mailing a monthly paper invoice).
   - Products of type subscription that apply to a service (for example, wireless service).
   - Discounts of type subscription that apply to an account (for example, account-level discounts).
   - Discounts of type subscription that apply to a service (for example, a *free minutes* discount).

   Product synchronization occurs by means of the process integration for product lifecycle management. Examples of supported products are located in an appendix.

**For more information,** see Appendix A: Order Management: Matrix of MACD Actions Supported Per Billing Product Type.

2. Creates or updates service instances and purchased product and discount instances in the billing system as part of the order interface to billing.

   The integration supports the following actions: ADD, DELETE, UPDATE, SUSPEND, RESUME, MOVE-ADD, and MOVE-DELETE.

   It supports communicating updates to the service identifier, billing account, billing profile, and price changes on existing services.

   As part of service cancellations or promotion upgrade or downgrades, when an old product is canceled, whether the customer gets a refund for (already-billed) monthly charges or whether the refund is prorated depends on product-level controls in Oracle BRM.

   As part of a Move transaction, the integration supports changing Service Identifier, Billing Account, and Billing Profile. The integration does not support purchasing new products or canceling existing products as part of a Move transaction.

**Note:** Transferring a service from one location to another in Siebel CRM results in lines with the action of MOVE-ADD and MOVE-DELETE. (Previously referred to as “Move.”)

**For more information,** see Appendix A: Order Management: Matrix of MACD Actions Supported Per Billing Product Type, Appendix B: Examples of Changing the Paying Parent on Subordinate Accounts, and Appendix I: Mapping Billing Dates.

   The solution supports account-level default balance groups alone.

   - The account-level balance group is created when the service account or billing account referenced on the order is created in Oracle BRM. A balance group in Oracle BRM can

reference a single bill-info. When the account-level balance group is created, it uses the first billing profile referenced on the first order processed for an account. Thus all account-level products and services for a given account on the same order or subsequent orders need to reference the same billing profile. An order violating this assumption will fail billing integration with an Oracle BRM error.

- The solution does support updating an existing billing profile in Siebel; such changes are synchronized to billing outside of the order integration flow.

**For more information,** see the *Siebel CRM Integration Pack for Oracle Communications BRM: Agent Assisted Billing Care Implementation Guide*.

Because the solution supports only an account-level balance group, transfer of services (or account-level products) from one account to another is not supported.

Change orders that update the service account on existing services will fail billing integration with an Oracle BRM error.

3. Communicates pricing information such as price or discount overrides, discounts, and onetime and penalty charges as part of the order interface to billing.

   For price changes that occur mid-cycle, the integration passes the price or discount overrides on a purchased product as is, the new price goes into effect from the following billing period, and no credits or debits are issued for the current period. If the latter is desired, then the Siebel CRM user needs to explicitly disconnect and add the product with the new price versus changing the price on an existing product.

   Onetime charges*, for actions such as suspend and resume, are applied as service-level charges. Penalty charges incurred for compromising a promotion agreement are communicated to Oracle BRM as account-level charges.

   As delivered, Siebel supports defining charges for any of these actions: Suspend, Resume, Move, and Delete. One can extend Siebel to define charges for other actions such as Update.

   For example, a communications service provider (CSP) could charge a customer a fee for requesting a change to his or her phone number or billing profile. The order billing integration generically supports such charges regardless of the action that triggered the charge.

   The integration expects order lines representing such charges to be tied to the service bundle line using the related asset integration ID and due date (on the Siebel order line) and using the charge parent line (on the order enterprise business message (EBM)). Thus, any lines on the order that are tied to the service bundle line (regardless of the action on that line) using the related asset integration ID and due date (on the Siebel order**) and using the charge parent line (on the order EBM) are processed by billing interface and applied to the respective service instance.

   If the application business connector service (ABCS) that transforms the Siebel Order application business message (ABM) to the order EBM is unable to resolve the base line that a new order or change order onetime charge maps to, it will not populate the charge parent line and the charge will be applied to the account when the charge line is interfaced to billing.

* Refer to the product bundling methodology for defining onetime charge products in Oracle BRM and synchronizing them to Siebel for the purpose of tying them to new order or change order actions.

** The onetime charge points to the service bundle line using the related asset integration ID. The integration assumes that the due date on the charge line is the same as the service bundle line with the new order or change order action that triggered the charge. For example, service is suspended and resumed by means of the same order and two different charges are applied. The charge line applied for the suspend action points to the service bundle line with the SUSPEND action, and the due date on both the lines are the same. The charge applied for the resume action points to the service bundle line with the RESUME action, and the due date on both the lines will be the same.

**For more information** about service bundles, see Chapter 2: Understanding the Product Bundling Methodology.

The pricing commit type on the order line controls whether the difference between the list and selling price (due to promotion bundling discounts, matrix discounts, or manual price overrides) on a purchased product is communicated as a price or discount override to billing. Discount overrides can be accounted for in GL in Oracle BRM.

- If the pricing commit type is set to Committed, then the integration sets a price override when purchasing the product in billing.
- If the pricing commit type is set to Dynamic, then the integration sets a discount override when purchasing the product in billing.
- The Dynamic Discount method on the line controls whether the discount override is of type Percent or Amount.
- In the case in which the intent is to use Oracle BRM pricing as is, the pricing commit type on the order line should have a value of Dynamic, and neither the discount amount nor the discount percent will be set. In this case, the integration sets neither a price nor a discount override for the product purchase.

**Note:** At most, for a charge type within a given product, Oracle BRM allows a single override price. This means that if an Oracle BRM product is mapped to multiple events of the same type and is synchronized to Siebel CRM as a complex product with multiple simple products, the Siebel CRM application cannot override the price for the charge type that has multiple charges defined. If it does, it will be applied as the override value for all charges of that charge type. This same constraint applies to discount overrides as well.

**For more information** about using the pricing commit type and dynamic discount method, see the Siebel product documentation.

4. Communicates service identifiers (for example, phone number for land-based or wireless phone service) to the billing system as part of the order interface to billing. The service identifier on the service bundle line in Siebel CRM is communicated to Oracle BRM. For telephony services, it is used as the phone number. For nontelephony service, it is used as the login and password.
5. Communicates Siebel promotion information for invoice display.

To allow Oracle BRM to display promotion information on the invoice, the integration communicates the following information about the promotion when interfacing an order for billing:

- For new promotion purchases, the integration creates bundle instances (under the billing account on the order line) with the following information:

  Promotion name

  Promotion description

  Effective start date (purchase date from Promotion Order line, if available, else request date if available, else Oracle BRM defaults current date).

- The integration creates the purchased product and discount instances with reference to the respective purchased bundle instance. Such references are not created for products of type Item.
- As subsequent orders are processed, the integration creates new references as needed and maintains existing references such that the purchased products and discounts point to the bundle instance that is current.
- When a purchased promotion is canceled as part of a downgrade, upgrade, or cancellation, the integration cancels the bundle instance in Oracle BRM by specifying an effective end date. The integration uses the actual delivery date (on the order line canceling the promotion). If the actual delivery date is not available, it uses the request date.

**Note:** No support is provided for translation of promotion name or description. Changing the name and description of the promotion (design time data) in Siebel does not have any effect on transactions that have already been submitted for processing and interfaced to billing.

6. The service that interfaces the order to Oracle BRM either processes all of the lines on the incoming message or none of them. If an error occurs while it is processing the lines, then the entire transaction is rolled back.

**For more information** about order fallout, see Chapter 8: Understanding the Process Integration for Order Fallout Management.

## Supporting Simple Service Bundles

The Order to Bill PIP supports two product bundling methodologies: service bundles and simple service bundles.

**For more information** about the bundling methodologies, see Chapter 2: Understanding the Product Bundling Methodology.

The order billing integration supports the simple service bundle methodology for all supported features, within the constraints listed in that chapter.

Here is a summary of how the integration supports purchases of simple service bundles:

- Purchasing a simple service bundle creates both a service instance and a purchased product instance in Oracle BRM. If the service was purchased within the context of a promotion, the product instance in Oracle BRM will be tied to the purchased promotion (or bundle) instance. See the following cross-reference impact section.
- The quantity (if > 1) on a simple service bundle line applies to the product purchase alone. Thus a single simple service bundle line creates:
  - A single service instance *and*
  - A single purchased product instance with a quantity as specified on the order line.
- Both single-phase billing and two-phase billing are supported for the simple service bundle.

Here is a summary of how the integration supports changes to purchased simple service bundles:

- Suspending or resuming the asset that represents a simple service bundle suspends or resumes the service and product on Oracle BRM.
- Disconnecting the asset that represents a simple service bundle cancels the service and product instance in Oracle BRM. Note that when using a simple service bundle, you *cannot* cancel the product without canceling the service.
- Transferring the asset that represents a simple service bundle in Siebel (Move-add or Move-delete) results in the cross-reference being adjusted for both the service and purchased product instance.
- Updates to service instance attributes (for example, Service ID, billing account/billing profile) on the asset that represents a simple service bundle results in the appropriate updates to the service instance in Oracle BRM.
- Updates to product attributes* (for example, pricing changes, promotion reference) on the asset that represents a simple service bundle results in the appropriate updates to the purchased product instance in Oracle BRM. Changes to billing dates as part of two-phase billing are honored. (* Note that quantity changes are not propagated to Oracle BRM currently.)
- If a onetime charge was defined and applied for a Move, Add, Change, and Disconnect (MACD) action in Siebel, it will be applied in Oracle BRM to the balance group that the service instance points to.

## Cross-Reference Impact

With simple service bundles, a single Siebel asset (for the simple service bundle product) is mapped to both the service instance and the purchased product instance in Oracle BRM. To manage this, the integration creates an additional cross-reference entry in the InstalledProduct cross-reference. Here is an example:

| Cross-Reference Type | Siebel_01 | Common | BRM_01 |
|---|---|---|---|
| InstalledProduct_Id | Siebel-S01 | C-ON-01 | BRM-A01 |
| InstalledProduct_Id | -- | C-ON-01+Child | BRM-B01 |

In the previous example, BRM-A01 is the Oracle BRM portal object ID (POID) for the service instance and BRM-B01 is the Oracle BRM POID for the purchased product instance. The common ID for the purchased product instance is the same value as the common ID for the service instance with the string “+Child” appended to it.

## Supporting Single Phase versus Two-Phase Billing (Billing Initiation and Billing Fulfillment)

The solution supports both single-phase and two-phase billing. In single-phase billing, the order is interfaced to billing (or billing-fulfilled) after the service is provisioned. In two-phase billing, the order is billing-initiated before the service is provisioned, and after service activation it is billing-fulfilled.

To support various fulfillment latency requirements, the order billing interface can be called in two modes (by setting the ProcessFulfillmentOrderBillingEBM /DataArea/ProcessFulfillmentOrderBilling/FulfillmentModeCode):

- INITIATE BILLING
- FULFILL BILLING

To enable single-phase billing, you need to call the order billing interface using only the FULFILL BILLING mode. To enable two-phase billing, you need to call the order billing interface using INITIATE BILLING before the service is provisioned, and then call it using FULFILL BILLING mode.

### INITIATE BILLING Mode

An implementer can design an order orchestration flow such that it first interfaces the order to billing before the order is sent to provisioning. Calling the interface in this mode is optional.

In this mode, the billing interface can be called with either the whole order[1] or order components such as promotion lines, service bundles[2], and account-level products.

Depending on the requirements, the implementer should set some or all of the following dates on new purchase of products[3] to the future (in essence they are treated as inactive when interfaced to billing):

- Purchase Date (ProcessFulfillmentOrderBillingEBM /DataArea/ProcessFulfillmentOrderBilling/FulfillmentOrderLine/FulfillmentOrderSchedule/PurchaseDate)
- Cycle Start Date (ProcessFulfillmentOrderBillingEBM /DataArea/ProcessFulfillmentOrderBilling/FulfillmentOrderLine/FulfillmentOrderSchedule/CycleStartDate)
- Usage Start Date (ProcessFulfillmentOrderBillingEBM /DataArea/ProcessFulfillmentOrderBilling/FulfillmentOrderLine/FulfillmentOrderSchedule/ServiceUsageStartDate)

Thus to support the scenario in which a fulfillment latency exists between service activation and billing, and you want to ensure that service usage is rated as soon as the service is activated but you want to start cycle fees only as of the date that the service was requested by the customer, you need to have your order management system set the purchase and usage start dates to current and the cycle start date alone to the future when calling this service. See the subsequent section for certain modeling recommendations.

In this mode, the order interface to billing processes only new purchases of services or account-level products or new purchases of products for existing services.

If a promotion is purchased as part of the new purchase, then that is processed as well. Onetime charges for actions such as Suspend, Resume, Move, and Disconnect and promotion penalties are not processed in this mode.

**For more information** about how dates are set in Oracle BRM, see Appendix I: Mapping Billing Dates.

### Handling of Revision Orders

Oracle BRM has validation that prevents the caller from resetting purchase and cycle start dates once they become current.

The integration does not reset the purchase date as part of billing-initiation revision processing. It does reset the cycle start and usage start date if asked by the caller. However, if billing initiation is called to process a revision on order lines that are already billing-initiated, and billing initiation is asked to reset the cycle start date[4] if the previously set date is current, then billing initiation will fail due to the Oracle BRM validation error.

### Modeling and Implementation Recommendations

- General

  The interface validates that the cycle date is set to the future for products of type subscription/discount. For products of type item, the interface validates that the purchase date is set to the future. As a best practice, we recommend that when calling billing initiation, the caller set the billing date that is being set to the future to a year ahead of the due date.

  Oracle AIA deems the purchase, cycle start, or usage start as being in the future as long as the billing date in question is > (fmw current time converted to UTC + (25 or XX hours, whichever is greater)).

  XX is the value of the AIA config property ‘FutureTimeThreshold’. This property has a default value of 8640 hrs (365 days in hours).

  If an implementer is highly confident of the lead time required to activate the service, then they can lower the value of the ‘FutureTimeThreshold’ property such that the order management system does not have to call fulfill billing to reset the dates (that were set in initiate billing). This also allows the billing dates to naturally become current soon after the service is activated. This property is settable per Oracle BRM instance level.

  If the property ‘FutureTimeThreshold’ is not specified for a given billing instance, then the integration assumes the default value of 8640 hours (365 days).

**Note:** Products of billing type Item have to be purchased with a future date in billing initiation to enable the integration to cross-reference them and thus avoid repurchasing them in billing fulfillment. The 25-hour minimum threshold is hard-coded to enable this.

  BRM requires that purchase date be before or equal to usage and cycle start dates. If the caller does not comply with this for any line[5], then the billing interface (Oracle BRM ABCS) will error.

- Purchase Fees or Activation Charges

  Oracle BRM requires that the purchase date on a product be the same as or earlier than the usage start date. If activation (purchase fees) and usage charges were modeled on the same product to support the fulfillment latency scenario, you would need to set both the purchase and start usage date to current.

  However, if the customer had canceled his order before service was provisioned, you would need to manually process a refund of the activation charges to him. To avoid this manual process, you would need to model the activation (purchase) fee on a product of type item, which is a separate product from the one on which the usage and cycle charges are modeled. Now to support the fulfillment latency scenario, you would set the purchase date for products of type item to the future and set the purchase and usage start dates for the subscription products to current.

- Discounts

  If the service bundle includes products representing purchase or usage discounts, then to ensure that the customers get the discount, the purchase and usage start dates for the discount products also need to be set to current when you are modeling the flow that sets the purchase and usage start dates to current for the subscription products.

## FULFILL BILLING Mode

Once provisioning is complete, the order orchestration flow can interface the order to billing in this mode. This is the default mode that the interface supports and is required to interface an order to billing.

In this mode, the interface processes all order lines that are sent (new and change orders). Onetime charges for actions such as Suspend, Resume, Move, and Disconnect and promotion penalties are processed in this mode.

For orders (order lines) that have been interfaced in the INITIATE BILLING mode, the caller can now set a specific date[6] (based on the actual delivery date) for those new purchases whose billing dates were earlier set to the future.

Thus, for the case in which only the cycle start date was set to the future during billing initiation, it should now be reset to the actual delivery date (date when the service was delivered). For the case in which the purchase, cycle start, and usage start dates have been set to the future, the caller should now set them to the actual delivery date.

**Note:** Billing dates that were set to current in billing initiation should not be reset in billing fulfillment because Oracle BRM will end in error.

The following prior values must be supplied:

- PurchaseDate: ProcessFulfillmentOrderBillingEBM/DataArea/ProcessFulfillmentOrderBilling/ PriorFulfillmentOrder/FulfillmentOrderLine/FulfillmentOrderSchedule/ PurchaseDate
- CycleStartDate: ProcessFulfillmentOrderBillingEBM/DataArea/ProcessFulfillmentOrderBilling/ PriorFulfillmentOrder/FulfillmentOrderLine/FulfillmentOrderSchedule/ CycleStartDate
- ServiceUsageStartDate: ProcessFulfillmentOrderBillingEBM/DataArea/ProcessFulfillmentOrderBilling/ PriorFulfillmentOrder/FulfillmentOrderLine/FulfillmentOrderSchedule/ ServiceUsageStartDate

### Notes

[1] All of the lines on the order that are intended for a certain target billing system and related lines such as promotion lines.

[2] Service bundle means the service bundle line and all its component lines. The solution does not support a scenario in which some of the service bundle component lines are sent for billing initiation and billing fulfillment, while others are sent only for billing fulfillment. In such a scenario, the service bundle component lines that are sent only for billing fulfillment will not get processed.

[3] A product referenced on the Siebel CRM order line could result in the purchase of product or a discount based on how it was originally defined in Oracle BRM. For the promotion line, only the purchase date is relevant.

[4] In this case, the order management system would have set the prior values for the billing dates to indicate to the billing integration that the dates are being reset.

[5] A product referenced on the Siebel CRM order line could result in the purchase of product or a discount based on how it was originally defined in Oracle BRM. For the promotion line, only the purchase date is relevant.

[6] The interface relies on the population of prior value fields to indicate that an attribute on the line has changed. So your order management system should set the prior value fields for the billing dates.

## Assumptions and Constraints for Two-Phase Billing

1. For multi-event billing products, the integration honors billing dates (purchase start date - nrc_start_date, cycle start date - rc_start_date, usage start date - usage_start_date in Siebel) on the parent complex product alone.
2. Billing Initiation is optional, but Billing Fulfillment is mandatory if an order (or order lines) needs to be interfaced to billing.

   (Billing Initiation is defined as the billing interface called in Initiate Billing mode. Billing Fulfillment is defined as the billing interface called in Fulfill Billing mode.)
3. The product that an order line references will not change once the line has been billing-initiated.
4. The order management system sends the onetime charge associated with a MACD action (Suspend, Resume, Move, Disconnect) with the service bundle on which the action is being performed.
5. Every MOVE-ADD line on Siebel order has a matching MOVE-DELETE (and vice versa). The order management system sends MOVE-ADD lines along with the MOVE-DELETE lines to billing.
6. Once order lines are submitted for Fulfill Billing, they are assumed to have hit a hard point of no return (PONR) and cannot be revised in Siebel CRM.
7. Service ID is always sent as input to the billing interface (Initiation or Fulfillment).
8. When Initiate Billing is called as part of compensation (ADD-REDO) for revision processing, if the billing dates that were set earlier need to be recomputed for resetting in billing, then the caller (order management system) will supply the prior values for those dates to indicate to the interface that the dates need to be updated.

9. When Fulfill Billing is called to reset billing dates (assuming they had been set to the future using Initiate Billing), the caller (order management system) will additionally supply the prior values for those dates to indicate to the interface that the dates need to be updated.

**For more information** about how dates are set in Oracle BRM, see Appendix I: Mapping Billing Dates.

## Supporting Revisions

To provide support for revisions once order lines are billing-initiated but not yet billing-fulfilled, the order interface to billing expects the caller (order management system) to pass in a fulfillment mode at the line level.

- The first time that billing initiation is called for lines, the fulfillment mode should be set to DO.
- If an order line is successfully billing-initiated and subsequently the order line is revised in Siebel CRM and the order resubmitted, then the order management system compares the revised line against what was submitted to billing initiation, determines whether any changes need to be processed, and calls billing initiation with a fulfillment mode of REDO to process the delta[1].
  - Changes to the following attributes on a revised promotion line will result in updates to billing: Billing Account, Purchase Date.
  - Changes to the following attributes on a revised account-level product line will result in updates to billing: Billing Account, Bill Profile, Promotion reference, Pricing Information[4], Billing Dates[2].
  - Changes to the following attributes on a revised service bundle line will result in updates to billing: Billing Account, Bill Profile, Promotion reference, Service ID.
  - Changes to the following attributes on a revised service bundle component line will result in updates to billing: Pricing Information[4], Billing Dates[2].

**Note:** Revisions to order lines for products of type item can be interfaced to billing as long as the billing date is not current. Once it is current, the call to update Oracle BRM will fail.

**For more information** about these attributes, see Appendix A: Order Management: Matrix of MACD Actions Supported Per Billing Product Type.

- If an order line is successfully billing-initiated[5] and subsequently the order line is canceled in Siebel CRM[3] and the order resubmitted, then the order management system calls billing initiation with a fulfillment mode of UNDO.
- If no changes are made to an order line as part of a revision, but it still needs to be submitted for context (for example, the service bundle component line is revised but the service bundle line is not, the service bundle line is still be sent because the service bundle as a whole is sent to billing), then the order management system calls billing initiation with a fulfillment mode of NOOP.

### Notes

[1] Old attribute values are supplied only for delta changes.

[2] Only start cycle and start usage dates should be changed if they are not yet current. The integration ignores requests to reset the purchase date.

**For more information,** see Supporting Single Phase versus Two-Phase Billing (Billing Initiation and Billing Fulfillment).

[3] On a Siebel revised order, this manifests as lines being dropped.

[4] Pricing information includes list price, selling (or net) price, pricing commit type, dynamic discount method, discount amount, and discount percent.

[5] The Oracle AIA service that interfaces order messages to Oracle BRM processes all lines or none of the lines. It does not do partial processing. So once an order is successfully billing-initiated, if any subsequent revisions for lines on the base order have to be processed, then the calling order management system must trigger compensation as described previously (using REDO, UNDO, or NOOP mode). If the order failed billing initiation (and triggered order fallout), then a subsequent revision is submitted from Siebel CRM, and the Order Management system trigger should be sent as is for billing initiation (DO mode).

**Note:** As delivered, the integration does not check for changes to the Special Rating List reference on revision orders once the List product has been billing-initiated.

This table summarizes revision actions:

| Action on Order Line | Fulfillment Mode (expected from calling order management system) | Processed As | Comments |
|---|---|---|---|
| ADD | DO | ADD | Billing initiation processes only new purchases (lines with action of ADD). |
| ADD | REDO | UPDATE | Because billing initiation processes only new purchases (lines with action of ADD), changes to those lines are processed as updates. Prior value fields are set only for attributes that have changed on the revision. |
| ADD | UNDO | DELETE | Because billing initiation processes only new purchases (lines with action of ADD), cancellations to those lines are processed as deletes or disconnects. |
| ADD | NOOP | Ignored | Billing initiation processes only new purchases (lines with action of ADD); if on revision, those lines have not changed (from original order), then they are ignored. |

### Assumptions and Constraints for Revisions

1. Order lines are assumed to hit the PONR after they have been interfaced to billing in the FULFILL BILLING mode. Support for revisions is provided only for the case in which order lines have been billing-initiated (interfaced to billing in the INITIATE BILLING mode) but not yet billing fulfilled (interfaced to billing in the FULFILL BILLING mode).
2. Only new purchases (lines with action ADD) are processed by billing initiation; hence billing initiation will process only revisions for new purchases.
3. The billing interface detects a changed attribute by the presence of an old attribute value for that attribute on the message. This is true for change orders and revisions.

## Supporting Time-Based Offerings

The time-based offerings feature allows the definition and usage of products and discounts in Siebel CRM that are valid only for a specific duration, and expire after that.

The solution for time-based offerings has two components: design time and order time.

**Design Time**

**For more information** about the design time component, see “Chapter 2: Understanding the Product Bundling Methodology”, Supporting Time-Based Offerings.

**Order Time**

- *New Purchase* – When an order for a time-based offering is placed and processed the following occurs:
  - Siebel CRM calculates the end date, taking into account the start date (defaulted from due date), and the Duration, DurationUOM, and DurationValidityStart transaction attribute values.
  - The order is then submitted to the order management system for fulfillment. When the order is fulfilled, the order management system sets the purchase, cycle start, and usage start dates based on service actual delivery date and recalculates the end date.
  - When the order is billing fulfilled, the integration communicates the end date for the purchased product or discount to Oracle BRM.
  - As part of the order update back to Siebel CRM, the order management system, through the integration, communicates the actual start and end dates to Siebel CRM.
- *Change Order* (such as a promotion upgrade or downgrade that results in changes to the duration validity of a previously purchased time-based discount) :
  - Siebel CRM recalculates the end date based on taking into account the Duration, DurationUOM, and DurationValidityStart transaction attribute values.
  - The order is then submitted to the order management system for fulfillment. When the order is fulfilled, the order management system recalculates the end date based on the actual delivery date**.** The end date is recalculated *only if any* of the validity attributes have changed on the order (by comparing against prior values) as follows.

If DurationValidityStart = Original End:
Service End Date = Prior Value for Service End Date + Duration

If DurationValidityStart = Now:
Service End Date = Actual Delivery Date Time + Duration

If DurationValidityStart = Original Start:
Service End Date = Service Start Date + Duration

- When the order is billing fulfilled, the integration communicates the new end date for the purchased product or discount to Oracle BRM.
- As part of the order update back to Siebel CRM, the order management system, through the integration, communicates the changed end dates to Siebel CRM.

**Note:** It is recommended that end dates not be set during Billing Initiation since it is not needed and avoids the requirement to manage them as part of revisions.

## Assumptions and Constraints for Time-Based Offerings

These are the assumptions and constraints for time-based offerings (TBO):

1. When using an Order Management system other than Oracle OSM, it must behave as described above to enable support for Time Based Offerings.
2. The Implementer must schedule a recurring job (daily or some other frequency based on their requirements) in Siebel to execute the workflow (SWI Asset Status Update Workflow) to inactivate such assets (Time Based offering products whose end date has passed). This is required to ensure that change orders for services that include Time Based Offering products are successfully processed.
3. To ensure that the purchased products and discounts reflect the correct status once the expiration date is passed, the Implementer must periodically run the Oracle BRM utilities pin_cycle_fees –cancel and pin_discount_cleanup in Oracle BRM
4. When a subscription product that is duration-based (Time Based Offering) and is also marked as a simple service bundle in Siebel CRM, is purchased and fulfilled, the Siebel CRM asset changes to inactive on the duration expiring. This results in scenario where the service instance is still active in Oracle BRM, but the corresponding asset in Siebel CRM is inactive (because the same Siebel asset is mapped to both the service and the purchase product or discount instance in Oracle BRM). To handle such cases, the implementer must develop custom scripts to inactivate the respective service instances in Oracle BRM.

# Supporting Friends and Family Lists

The friends and family feature enables end customers to call certain phone numbers at discounted rates. The feature requires special rating products to be defined in Siebel CRM and included in a service bundle.

**For more information** about how special rating products are supported and the methodology, see “Chapter 2: Understanding the Product Bundling Methodology,” Supporting Friends and Family.

When orders for such service bundles are placed, the CSR can create the lists, optionally add numbers to the lists, and associate the lists with the special rating products.

**For more information,** see the sections on friends and family plans in the "Profiles in Siebel Communications," chapter of the *Siebel Communications Guide*.

When the order is interfaced to billing, the integration creates a list profile for every order line that has a special rating product. These list profiles are associated with the service instance in Oracle BRM. For the list profile to get created during order billing integration, a list (special rating profile list) must be associated to the special rating product on the order.

When the order is successfully interfaced to billing and is auto-asseted, the special rating product used to capture the list is tracked as an asset in Siebel.

**Note:** The solution assumes that if the same special rating list is referenced by multiple services, (for example, VOIP and Wireless Voice) those services are fulfilled in the same billing system.

**For more information,** see "Chapter 2: Understanding the Product Bundling Methodology," Family. and Appendix E: Configuring Multiple Instances of Oracle BRM.

## Using Change Orders and Special Rating Products

Here are some recommendations for using change orders and special rating products.

- Changing Special Rating list entries

  You can use either of the following two options to achieve this:

  - Tying a completely different list to the special rating product: A change order can be used to update the special rating list reference on the existing special rating product asset to a different list reference. When the integration processes the change, it updates the list profile in billing with contents from the new list.
  - Adding or removing entries from a list currently referenced by a special rating product: The Siebel Special Rating Profile user interface (UI) can be used to make changes to the list and synchronize them to Oracle BRM. This synchronization is enabled by the following integration services:

    ProcessInstalledProductSpecialRatingSetListSiebelCommsJMSConsumer

    ProcessInstalledProductSpecialRatingSetListSiebelCommsReqABCSImpl

    CommunicationsInstalledProductEBSV2 (Operation: ProcessInstalledProductSpecialRatingSetList)

    ProcessInstalledProductSpecialRatingSetListBRMCommsProvABCSImpl

- Promotion upgrades and downgrades

  Promotion upgrades or downgrades can result in the cancellation or addition of Special Rating products for an existing service.

  - Cancellation: When such orders are processed, the integration deletes the respective list profile in Oracle BRM.
  - Addition: When such orders are processed, the integration creates new list profiles in billing for the given service instance.

- Service cancellations

  Service cancellation results in the deletion of the list profile in Oracle BRM.

**For more information**, see Synchronizing Friends and Family List Updates to Oracle BRM.

### Modifying Friends and Family Lists

After a service that supports special rating has been purchased and the order fulfilled and asseted, the customer can use the Siebel Special Rating Profile UI to make changes to their list, and then update and synchronize the list to Oracle BRM.

The flow uses the operation ProcessInstalledProductSpecialRatingSetList on the enterprise business service CommunicationsInstalledProductEBS for this purpose. The specification group on the installed product EBM is used for communicating the list entries.

**For more information**, see Synchronizing Friends and Family List Updates to Oracle BRM.

## Sending Order Updates Back to Siebel CRM

The UpdateSalesOrder is used for two purposes:

- To update sales order data:

  Updating sales order data enables the order management system to enrich the sales order with data coming from downstream systems, such as provisioning. An example of such data is the service instance ID in cases in which network inventory has determined the data is needed during service provisioning. Order data updates should be sent to Siebel CRM only after passing the point of no return on an order line to avoid being lost due to order revisions being previously created in Siebel.

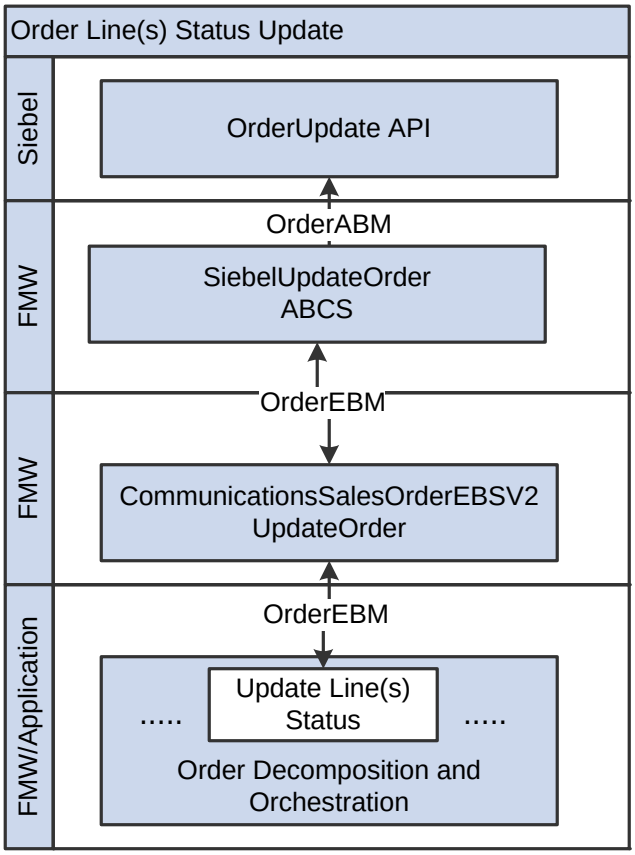
  Moreover, order data updates should be sent to Siebel at the same time or prior to the order line status changing to Complete to ensure that the data updates get included on the asset in Siebel.

- To update sales order status:

  Updating sales order status enables the order management system to send order and order-line-level status updates to keep the CSR and self-service customer updated on the progress made as the order is fulfilled. The order management system should optimize the number of updates to Siebel and limit updates to those that are significant to the Siebel user.

  The order management system is responsible for mapping statuses from different fulfillment systems' responses and aggregate status values at the order line and order header levels.

This diagram shows the order updates:




## Order line status update

These are the delivered services:

- CommunicationsSalesOrderEBSV2.UpdateSalesOrder
- UpdateSalesOrderSiebelCommsProvABCSImpl

These services delivered with the Order to Bill PIP are also used for order status updates in the Oracle Order to Activate PIP.

**For more information** about these services, see the *Oracle Order to Activate Integration Pack for Siebel CRM and Oracle Communications Order and Service Management Implementation Guides*.

The following attributes are considered part of the order-level status and can be used by your CFS:

| Sales Order EBO Component/Attribute | Delivered Seeded Values | Usage |
|---|---|---|
| Status/Code | Open, In Progress, Failed, Canceled, Complete | Used to keep CRM updated on the current status of order fulfillment at a high level. |

| | | |
|---|---|---|
| Status/Description | NA | Used to provide details for the current status. For example, in case an order is canceled, the description will be provided with the reason. |

The following attributes are considered part of the order-level status and can be used by your order management system:

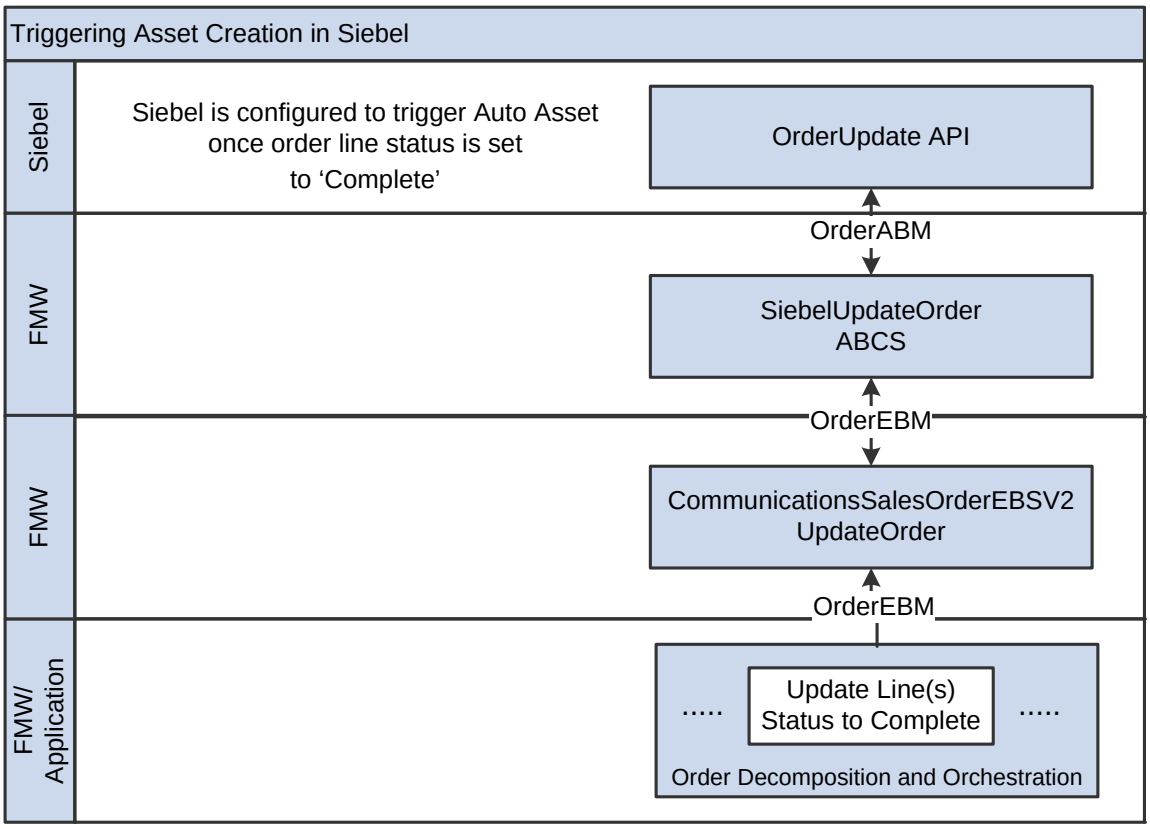
| Sales Order EBO Component/Attribute | Delivered Seeded Values | Usage |
|---|---|---|
| SalesOrderLine/Status/ Code | Open, In Progress, Failed, Canceled, Complete | Used to keep Siebel CRM updated on the current status of order-item fulfillment at a high level. |
| SalesOrderLine/Status/ Description | NA | Used to provide details for the current status. For example, in case the order line failed, the Status Context would be provided with cause. |
| SalesOrderLine/ Milestone | NA | Fulfillment passes last-reached milestone into this field |
| SalesOrderLine/ RevisionPermissibleCode | NOT YET, HARD | Determines whether Siebel allows revisions of an order item or submittal of revisions of an order item previously created. This is known as the point of no return. |
| SalesOrderLine/SalesOrderSchedule/ ActualDeliveryDateTime | NA | Determines the date when the purchased product or services are considered available to the customer. This may be the date that a physical good is shipped or delivered or a receipt is acknowledged. For service-based products, this is the date that the service is activated. |
| SalesOrderLine/SalesOrderSchedule/ ExpectedDeliveryDate | NA | The delivery date expected by the system as the result of Design and Assign. The default is the order RequestedDeliveryDate when the order is created by Siebel CRM. This is used by Fulfillment to communicate to Siebel CRM changes of the due date of specific order items |

# Creating or Updating Installed Assets in Siebel CRM

An installed asset is initially created when a customer orders a new service and that order is fulfilled and asseted. From then on, if the customer requests a change to the existing services, the CSR initiates what is known as asset-based ordering. An asset-based order (or new order or change order) has references to an existing installed asset and actions indicating how it needs to be modified to match the customer's request. After a new order or change order is fulfilled, the installed asset is updated to reflect the new desired state.

The process integration for order management relies on Siebel CRM auto-asset functionality. Siebel CRM is configured so that assets are automatically created or updated when the order line status is set to Complete.

This diagram shows how assets are created in Siebel CRM:




Creating assets in Siebel CRM

# Integration Flows

This integration delivers these services:

- ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer
- ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl
- CommunicationsSalesOrderEBSV2
- CommunicationsBillingEBSV1
- CommunicationsBillingResponseEBSV1
- CommsProcessFulfillmentOrderBillingAccountListEBF
- ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl
  - ProcessFulfillmentOrderBillingBRMCommsAddSubProcess
  - ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess

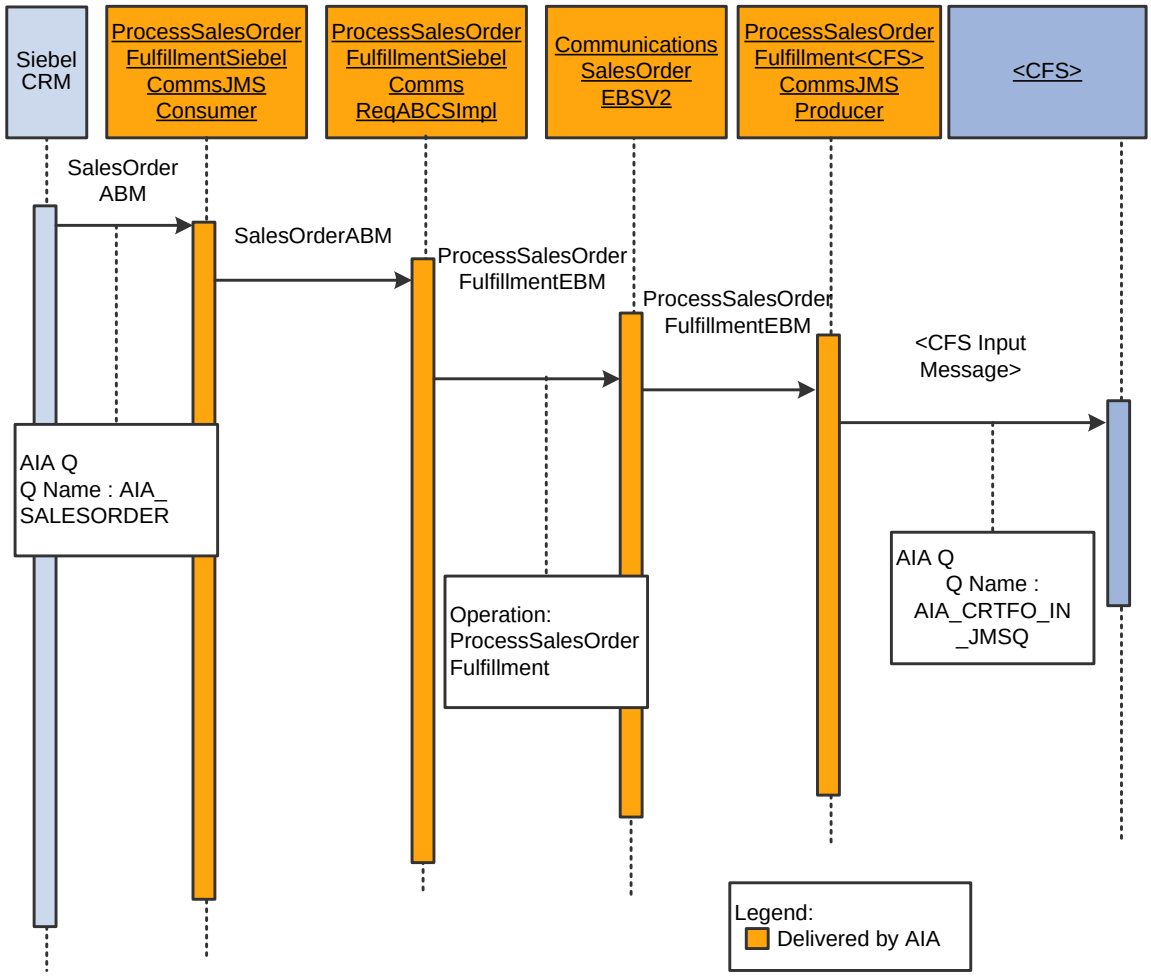
    - ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess
    - ProcessFulfillmentOrderBillingBRMCommsUpdateSubProcess
    - ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess
- UpdateSalesOrderSiebelCommsProvABCSImpl
- CommunicationsInstalledProductEBSV2
- ProcessInstalledProductSpecialRatingSetListSiebelCommsJMSConsumer
- ProcessInstalledProductSpecialRatingSetListSiebelCommsReqABCSImpl
- ProcessInstalledProductSpecialRatingSetListBRMCommsProvABCSImpl

The process integration for order management delivers these integration points:

- Submitting orders to the order management system.
- Interfacing orders to create customer data in Oracle BRM.
- Interfacing orders to create transaction data in Oracle BRM.
- Updating order information back to Siebel CRM.
- Synchronizing Friends and Family List updates to Oracle BRM.

## Submitting Orders to the Order Management System

This sequence diagram shows submitting orders to an order orchestration integration flow:




### Submitting orders to the order orchestration

This flow has the following sequence of activities. It has a one-way asynchronous pattern.

1. In Siebel CRM, a CSR enters a new order or a change order and submits it for processing.
2. The Siebel order web service enqueues the messages in the Oracle AIA queue (AIA_SALESORDERJMSQUEUE). This message is picked up by the ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer, which invokes the ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl.
3. The ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl transforms the Siebel ABM to the ProcessSalesOrderFulfillmentEBM and then invokes the ProcessSalesOrderFulfillment operation of the CommunicationsSalesOrderEBSV2.
4. You must configure the CommunicationsSalesOrderEBSV2.ProcessSalesOrderFulfillment to call the appropriate custom ProcessSalesOrderFulfillment<CFS>CommsJMSProducer, which must enqueue the message in AIA_CRTFO_IN_JMSQ. The CFS must dequeue the message from this queue and start the order orchestration process or flow to fulfill the order and close it.

## Defining Transaction Boundaries and Recovery Details

For this flow there are two transaction boundaries. The following table describes the transactions involved, the database operations, and what actions to take in case of an error.

If order submission from Siebel causes a system or business error, any further order to the account does not get processed until the error is fixed. All order submissions for that account are locked in the sequencer table. If the error is a business error then the message must be removed from the sequencer table and if the error is a system error then the message must be resubmitted.

**For more information** about system errors and business errors, see Using Error Type to Control Response to Order Fallout.

The following services are involved:

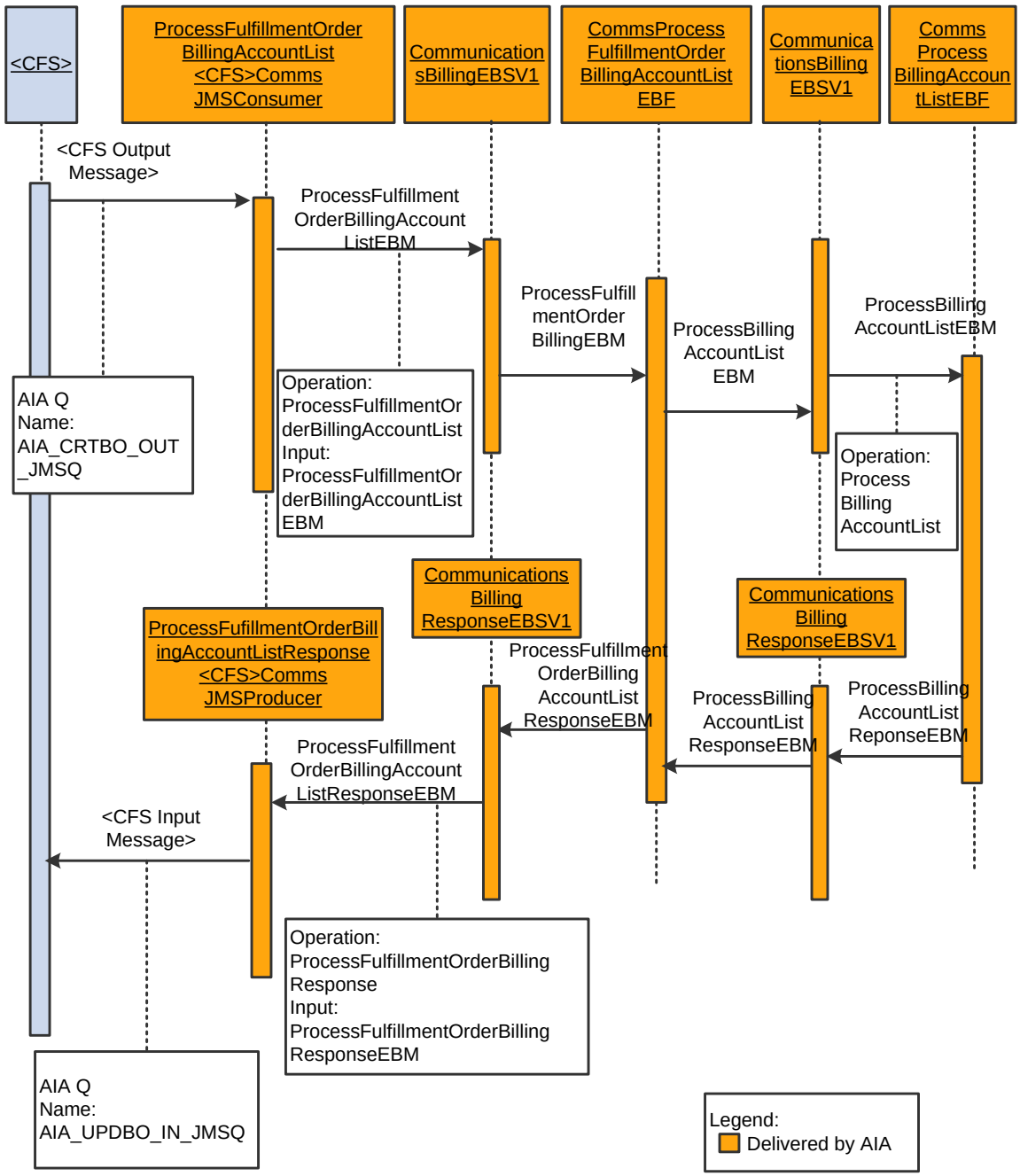
- ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer
- ProcessSalesOrderFulfillmentSiebelCommsSequencer
- ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl
- CommunicationsSalesOrderEBSV2
- ProcessSalesOrderFulfillment<CFS>CommsJMSProducer

| Transaction | DB Operations | In Case of Error | Recovery |
|---|---|---|---|
| ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer passes message to ProcessSalesOrderFulfillmentSiebelCommsSequencer | Message goes into the sequencer table. | Rollback JMS message to AIA_SALESORDERJMSQUEUE | Resubmit the order from either AIA_SALESORDERJMSQUEUE or from Siebel. |
| ProcessSalesOrderFulfillmentSiebelCommsSequencer passes the message to ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl, which invokes transformation logic and then invokes CommunicationsSalesOrderEBSV2. The message is then routed to ProcessSalesOrderFulfillment<CFS>CommsJMSProducer | AIA cross-reference entries. | Rollback cross-reference transactions.<br>Rollback the message to the sequencer table. | Resubmit the order from AIA_SALESORDERJMSQUEUE |

**For more information** about rollback procedures, see For more information about rollback procedures, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring Oracle AIA Processes for Error Handling and Trace Logging", Configuring Fault Policies to Not Issue Rollback Messages.

## Interfacing Orders to Create Customer Data in Oracle BRM

This sequence diagram shows the interface for orders to create customer data in the Oracle BRM integration flow:




### Interfacing orders to create customer data in Oracle BRM

This flow has the following set of activities. It has an asynchronous delayed response pattern.

1. The order orchestration process starts this flow by enqueueing the <CFS Output Message> message in AIA_CRTCUST_OUT_JMSQ. The Custom ProcessFulfillmentOrderBillingAccountList<CFS>CommsJMSConsumer must dequeue the message, transform it into the ProcessFulfillmentOrderBillingAccountListEBM, and invoke the CommunicationsBillingEBSV1.ProcessFulfillmentOrderBillingAccountList operation by passing the ProcessFulfillmentOrderBillingAccountListEBM.
2. The CommunicationsBillingEBSV1.ProcessFulfillmentOrderBillingAccountList routes the ProcessFulfillmentOrderBillingAccountListEBM to the CommsProcessFulfillmentOrderBillingAccountListEBF.
3. The CommsProcessFulfillmentOrderBillingAccountListEBF transforms the ProcessFulfillmentOrderBillingAccountListEBM into the ProcessBillingAccountListEBM and then invokes the CommunicationsBillingEBSV1.CommsProcessBillingAccountListEBF operation.
4. The CommunicationsBillingEBSV1.CommsProcessBillingAccountListEBF operation takes the ProcessBillingAccountListEBM and routes it to the CommsProcessBillingAccountListEBF. This invocation creates customer data in Oracle BRM.
5. The CommsProcessBillingAccountListEBF invokes the CommunicationsBillingResponseEBSV1.ProcessBillingAccountListResponse operation by passing the ProcessBillingAccountListResponseEBM back.
6. The CommunicationsBillingResponseEBSV1.ProcessBillingAccountListResponse operation routes the response back to the CommsProcessFulfillmentOrderBillingAccountListEBF.
7. The CommsProcessFulfillmentOrderBillingAccountListEBF transforms the response to the ProcessFulfillmentOrderBillingAccountListResponseEBM and invokes the CommunicationsBillingEBSV1.ProcessFulfillmentOrderBillingAccountListResponse.
8. The CommunicationsBillingEBSV1.ProcessFulfillmentOrderBillingAccountListResponse operation routes this message to the custom ProcessFulfillmentOrderBillingAccountList<CFS>CommsJMSProducer, which takes the ProcessFulfillmentOrderBillingAccountListResponseEBM, transforms it into the <CFS Input Message>, and enqueues it into AIA_UPDCUST_IN_JMSQ.
9. The central fulfillment system dequeues the message and completes the flow.

## Defining Transaction Boundaries and Recovery Details

For this flow there are two transaction boundaries. The following table describes the transactions involved, the database operations, and what actions to take in case of an error.

If any account creation causes a system or business error, any further updates to the account (and thereby processing of other orders for that account) do not happen until the error is fixed. All updates for that account are locked in the sequencer table. If the error is a business error then the message must be removed from the sequencer table and if the error is a system error then the message must be resubmitted.

**For more information** about system errors and business errors, see Using Error Type to Control Response to Order Fallout.

The following services are involved:

- ProcessFulfillmentOrderBillingAccountList<CFS>CommsJMSConsumer
- CommunicationsBilling EBSV1

- CommsProcessFulfilmentOrderBillingAccountListEBF
- CommsProcessBillingAccountListEBF
- CommunicationsCustomerPartyEBSV2
- CustomerPartyEBSV2
- QueryCustomerPartyListSiebelProvABCSImplV2
- CommunicationsCustomerPartyEBSV2Resequencer
- SyncCustomerPartyListBRMCommsProvABCSImpl
- CommunicationsCustomerPartyResponseEBSV2
- CommunicationsBillingResponseEBSV1
- ProcessFulfillmentOrderBillingAccountListResp<CFS>CommsJMSProducer

| Transaction | DB Operations | In Case of Error | Recovery |
|---|---|---|---|
| ProcessFulfillmentOrderBillingAccountList<CFS>CommsJMSConsumer invokes the CommunicationsBillingEBSV1, which routes the message to. CommsProcessFulfillmentOrderBillingAccountListEBF. CommsProcessFulfillmentOrderBillingAccountListEBF extracts relevant customer data and then invokes CommunicationsBillingEBSV1, which routes message to CommsProcessBillingAccountListEBF, which invokes CommunicationsCustomerPartyEBSV2. This invocation creates customer data in Oracle BRM. The CommunicationsCustomerPartyEBSV2 calls CustomerPartyEBSV2, which routes the message to QueryCustomerPartyListSiebelProvABCSImplV2. The ABCS fetches account details and a response is sent back to CommsProcessBillingAccountListEBF through CustomerPartyEBSV2 that invokes CommunicationsCustomerPartyEBSV2Resequencer. | AIA cross-reference entries for some of the Siebel entries.<br>Message goes into the sequencer table. | Rollback JMS message to the originating queue AIA_CRTCUST_OUT_JMSQ. | Resubmit the order from AIA_CRTCUST_OUT_JMSQ. |

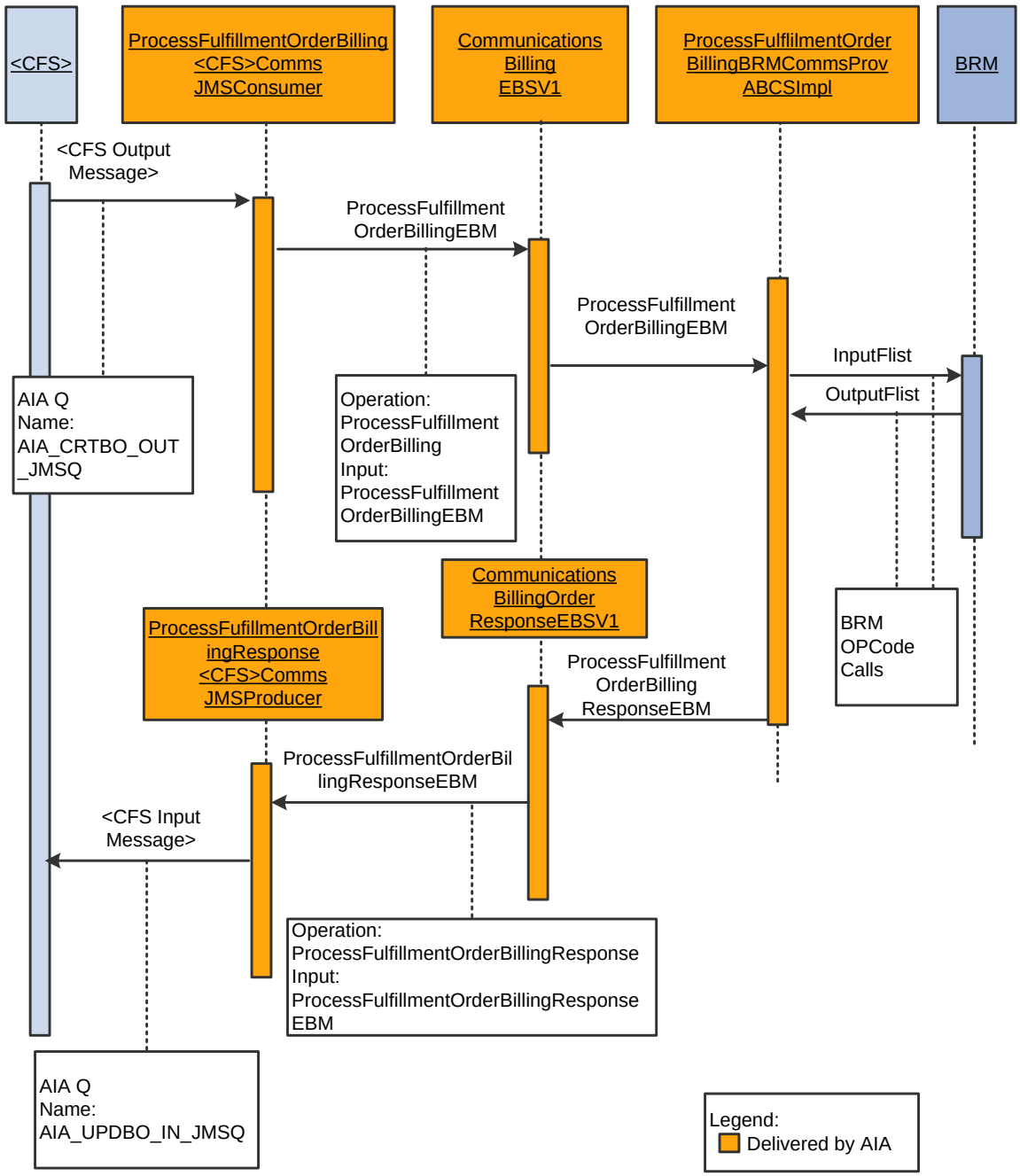
| Transaction | DB Operations | In Case of Error | Recovery |
|---|---|---|---|
| CommunicationsCustomerPartyEBSV2Resequencer calls CommunicationsCustomerPartyEBSV2, which calls SyncCustomerPartyListBRMCommsProvABCSImpl. This ABCS invokes Oracle BRM to either create or update an account. If successful a response is sent back to CommsProcessBillingAccountListEBF through CommunicationsCustomerPartyResponseEBSV2. The EBF invokes CommunicationsBillingResponseEBSV1 and a response message is then routed to CommsProcessFulfillmentOrderBillingAccountListEBF. This EBF transforms the response and invokes CommunicationsBillingResponseEBSV1, which routes the response to ProcessFulfillmentOrderBillingAccountListResp<CFS>CommsJMSProducer. The producer produces the response message to AIA_UPDCUST_IN_JMSQ. | AIA cross-reference entries created for Oracle BRM.<br>Data created in Oracle BRM.<br>Message goes to queue AIA_UPDCUST_IN_JMSQ. | Rollback cross-reference transactions.<br>Rollback data created in Oracle BRM.<br>Message goes back to the sequencer table. | Resubmit the message from the sequencer table. |

**Note:** If any order contains more than one account and a failure occurs after any account is processed successfully but the subsequent account fails, then error recovery may become difficult based on the point of failure. Customers must first examine the point of failure and then determine if it's necessary to recover the BPEL instance from the recovery console.

**For more information** about rollback procedures, see For more information about rollback procedures, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring Oracle AIA Processes for Error Handling and Trace Logging", Configuring Fault Policies to Not Issue Rollback Messages.

## Interfacing Orders to Create Transaction Data in Oracle BRM

This sequence diagram displays the interface orders to create transaction data in the Oracle BRM integration flow:




### Interfacing orders to create transaction data in Oracle BRM

This flow has the following set of activities. It has an asynchronous delayed-response pattern.

1. The order orchestration invokes this flow by pushing the <CFS Output Message> message into the AIA_CRTBO_OUT_JMSQ.

2. The custom ProcessFulfillmentOrderBilling<CFS>CommsJMSConsumer picks up this message, transforms it into the ProcessFulfillmentOrderBillingEBM, and invokes the CommunicationsBillingEBSV1.ProcessFulfillmentOrderBilling operation.
3. The CommunicationsBillingEBSV1.ProcessFulfillmentOrderBilling operation routes this message to the ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl.
4. The ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl service orchestrates the ProcessFulfillmentOrderBillingEBM into creating billing artifacts, service instances, purchased products, purchased discounts, and so on in Oracle BRM.
5. The ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl constructs the ProcessFulfillmentOrderBillingResponseEBM and sends it back to the CommunicationsBillingResponseEBSV1.ProcessFulfillmentOrderBilling.
6. The CommunicationsBillingResponseEBSV1.ProcessFulfillmentOrderBilling is routed to the custom ProcessFulfillmentOrderBillingResponse<CFS>CommsJMSProducer.
7. The custom ProcessFulfillmentOrderBillingResponse<CFS>CommsJMSProducer transforms the ProcessFulfillmentOrderBillingResponseEBM into the <CFS Input Message> and enqueues the message in AIA_UPDBO_IN_JMSQ.
8. The CFS dequeues the message and completes the flow.

## Defining Transaction Boundaries and Recovery Details

For this flow there is one transaction boundary. The following table describes the transaction involved, the database operations, and what actions to take in case of an error.

**For more information** about system errors and business errors, see Using Error Type to Control Response to Order Fallout.

The following services are involved:

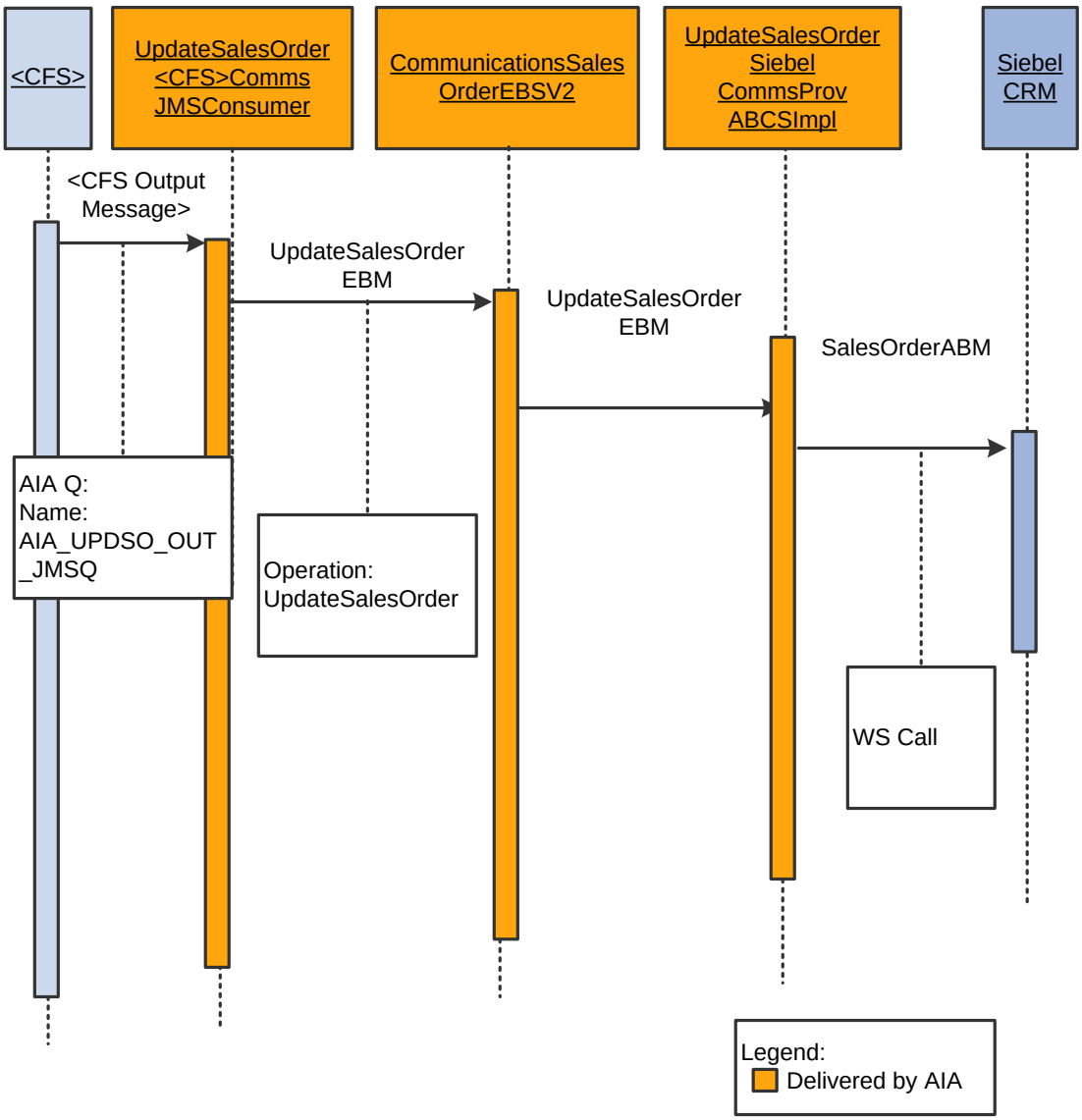
- ProcessFulfillmentOrderBilling<CFS>CommsJMSConsumer
- CommunicationsBillingEBSV1
- ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl
- ProcessFulfillmentOrderBillingBRMCommsAddSubProcess
- ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess
- ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess
- ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess
- ProcessFulfillmentOrderBillingBRMCommsUpdateSubProcess
- CommunicationsBillingResponseEBSV1
- ProcessFulfillmentOrderBillingResponse<CFS>CommsJMSProducer

| Transaction | DB Operations | In Case of Error | Recovery |
|---|---|---|---|
| ProcessFulfillmentOrderBilling<CFS>CommsJMSConsumer invokes CommunicationsBillingEBSV1, which passes the message to ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl. The ABCS creates billing artifacts and calls one or more subprocesses. The ABCS then creates the response message and sends it back to CommunicationsBillingResponseEBSV1, which routes it to ProcessFulfillmentOrderBillingResponse<CFS>CommsJMSProducer. | AIA cross-references created.<br>Oracle BRM data created.<br>Message goes to queue AIA_UPDBO_IN_JMSQ. | Rollback AIA cross-references.<br>Rollback data created in Oracle BRM.<br>Message goes back to the originating queue AIA_CRTBO_OUT_JMSQ. | Resubmit the order from AIA_CRTBO_OUT_JMSQ. |

**For more information** about rollback procedures, see For more information about rollback procedures, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring Oracle AIA Processes for Error Handling and Trace Logging", Configuring Fault Policies to Not Issue Rollback Messages.

# Updating Status Information on Order Lines in Siebel CRM

This sequence diagram displays the update status information on order lines in the Siebel CRM integration flow:




## Updating status information on order lines in Siebel CRM

This flow updates the status information on order lines in Siebel CRM. It has a one-way asynchronous pattern. This flow has the following set of activities:

1. The order orchestration invokes this flow by pushing the <CFS Output Message> message into AIA_UPDSO_OUT_JMSQ.
2. The custom UpdateSalesOrder<CFS>CommsJMSConsumer picks up this message, transforms it into the UpdateSalesOrderEBM, and invokes the CommunicationsSalesOrderEBSV2.UpdateSalesOrder operation.
3. The CommunicationsSalesOrderEBSV2.UpdateSalesOrder routes this message to the UpdateSalesOrderSiebelCommsProvABCSImpl.

4. The UpdateSalesOrderSiebelCommsProvABCSImpl orchestrates the calling of various Siebel web services to update the order information back to Siebel CRM.

## Defining Transaction Boundaries and Recovery Details

For this flow there are two transaction boundaries. The following table describes the transactions involved, the database operations, and what actions to take in case of an error.

If any update to Siebel causes a system or business error, any further updates to the account does not happen until the error is fixed. All updates for that account are locked in the sequencer table. If the error is a business error then the message must be removed from the sequencer table and if the error is a system error then the message must be resubmitted.

**For more information** about system errors and business errors, see Using Error Type to Control Response to Order Fallout.

The following services are involved:

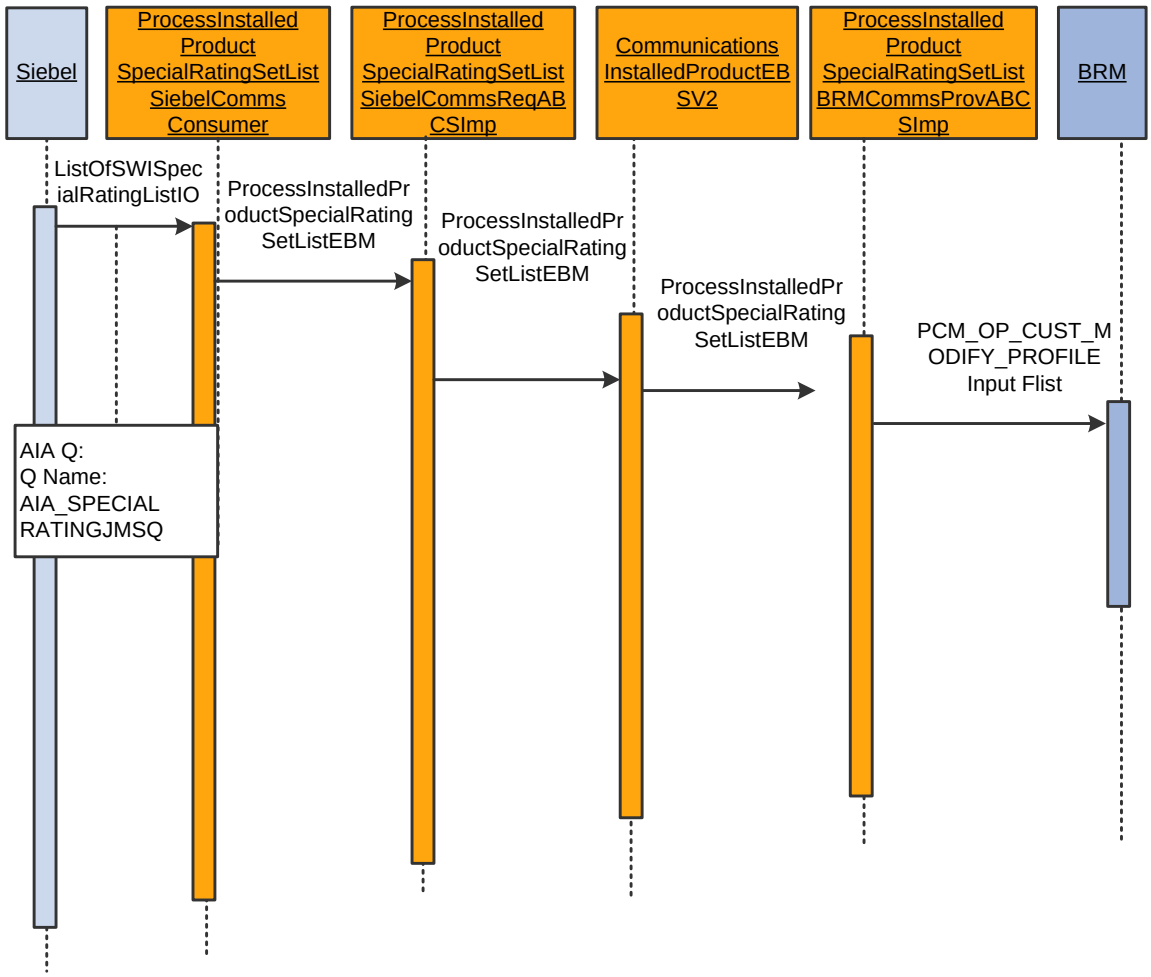
- UpdateSalesOrder<CFS>CommsJMSConsumer
- UpdateSalesOrder<CFS>CommsJMSConsumer_RS
- CommunicationsSalesOrderEBSV1
- UpdateSalesOrderSiebelCommsProvABCSImpl

| Transaction | DB Operations | In Case of Error | Recovery |
|---|---|---|---|
| UpdateSalesOrder<CFS>CommsJMSConsumer consumes the message and puts it in the sequencer table defined at the Routing Service UpdateSalesOrder<CFS>CommsJMSConsumer_RS. | Message goes into the sequencer table. | Rollback JMS message to AIA_UPDSO_OUT_JMSQ. | Resubmit the order from AIA_ UPDSO_OUT_JMSQ. |
| UpdateSalesOrder<CFS>CommsJMSConsumer_RS invokes CommunicationsSalesOrderEBSV1, which routes the message to UpdateSalesOrderSiebelCommsProvABCSImpl. The ABCS invokes the Siebel web service to update the order. | AIA cross-reference entries. | Rollback the message to the sequencer table. | Resubmit the order from the sequencer table. |

**For more information** about rollback procedures, see For more information about rollback procedures, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring Oracle AIA Processes for Error Handling and Trace Logging", Configuring Fault Policies to Not Issue Rollback Messages.

# Synchronizing Friends and Family List Updates to Oracle BRM

This sequence diagram displays the synchronizing friends and family list updates to the Oracle BRM integration flow:




## Synchronizing friends and family list updates

This flow has the following activities. It has a one-way asynchronous pattern.

1. This flow starts when, as a result of updating the Special Rating List in Siebel for an account; Siebel pushes the ListOfSWISpecialRatingListIO message into an Oracle Advanced Queuing (AQ) named AIA_SPECIALRATINGJMSQ.
2. The ProcessInstalledProductSpecialRatingSetListSiebelCommsJMSConsumer picks up this message and routes it to the ProcessInstalledProductSpecialRatingSetListSiebelCommsReqABCSImpl.
3. The ProcessInstalledProductSpecialRatingSetListSiebelCommsReqABCSImpl transforms this message into the ProcessInstalledProductSpecialRatingSetListEBM and invokes the CommunicationsInstalledProductEBSV2. ProcessInstalledProductSpecialRatingSetList operation.
4. The CommunicationsInstalledProductEBSV2. ProcessInstalledProductSpecialRatingSetList operation routes this message to the ProcessInstalledProductSpecialRatingSetListBRMCommsProvABCSImpl.

5. The ProcessInstalledProductSpecialRatingSetListBRMCommsProvABCSImpl calls the Oracle BRM opcode PCM_OP_CUST_MODIFY_PROFILE to update this information in Oracle BRM.

### Defining Transaction Boundaries and Recovery Details

For this flow there is one transaction boundary. The following table describes the transaction involved, the database operations, and what actions to take in case of an error.

**For more information** about system errors and business errors, see Using Error Type to Control Response to Order Fallout.

The following services are involved:

- ProcessInstalledProductSpecialRatingSetListSiebelCommsJMSConsumer
- ProcessInstalledProductSpecialRatingSetListSiebelCommsReqABCSImpl
- CommunicationsInstalledProductEBSV2
- ProcessInstalledProductSpecialRatingSetListBRMCommsProvABCSImpl

| Transaction | DB Operations | In Case of Error | Recovery |
|---|---|---|---|
| ProcessInstalledProductSpecialRatingSetList SiebelCommsJMSConsumer picks up message and routes it to ProcessInstalledProductSpecialRatingSetList SiebelCommsReqABCSImpl, which transforms message and invokes CommunicationsInstalledProductEBSV2. The EBS routes the message to ProcessInstalledProductSpecialRatingSetList BRMCommsProvABCSImpl, which invokes the Oracle BRM opcode to update information in Oracle BRM. | AIA cross-references updated. | Message goes back to the originating queue AIA_SPECIALRATINGJMSQ. | Resubmit from AIA_SPECIALRATINGJMSQ. |

**For more information** about rollback procedures, see For more information about rollback procedures, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring Oracle AIA Processes for Error Handling and Trace Logging", Configuring Fault Policies to Not Issue Rollback Messages.

## Orchestrating the Services

An order management system decomposes an order into suborders (lines or groups of lines) and manages the processing and fulfillment (shipping, provisioning, and billing) of each of those parts. It identifies the target fulfillment system for each line (or group of lines) and sends them to the respective system for fulfillment. Such a service or system is also known as a *central fulfillment system* (CFS).

The Order to Bill PIP does not include this order management system. It does include services (SalesOrderEBS.ProcessSalesOrderFulfillment) that can route to such an order management or central fulfillment system and services that can be invoked by the order management or central fulfillment system to create customer data in a billing system, to interface an order to billing, and to send status and other updates back to Siebel CRM.

Oracle AIA sends a sales order-based message (ProcessSalesOrderFulfillmentEBM) as input to the order management system. The order management system decomposes the input order into one or more fulfillment orders that it orchestrates across various end-fulfillment systems (such as provisioning, billing, and so on).

The AIA services that create customer data in the billing system and interface an order to billing expect FulfillmentOrder-based messages (ProcessFulfillmentOrderBillingAccountListEBM and ProcessFulfillmentOrderBillingEBM respectively). The order management system is expected to transform the incoming sales order message into fulfillment order-based messages. The process integration for order management delivers services that need to be orchestrated sequentially to run properly. Here are the assumptions and sequence:

- The services delivered in the process integration process messages for a single billing system at a time; the services CommunicationsBillingEBSV1.ProcessFulfillmentOrderBillingAccountList and CommunicationsBillingEBSV1.ProcessFulfillmentOrderBilling are destined for the same target billing system.
- The order decomposition and orchestration process needs to determine and stamp the ID of the target in the payload before calling the delivered services for this process integration.
- The order decomposition and orchestration process needs to call the CommunicationsBillingEBSV1.ProcessFulfillmentOrderBillingAccountList before calling the CommunicationsBillingEBSV1.ProcessFulfillmentOrderBilling.
- CommunicationsBillingEBSV1.ProcessFulfillmentOrderBilling requires that the customer data referenced on the order exist in the billing system before it is called. If an error occurs in customer data creation, then the CommunicationsBillingEBSV1.ProcessFulfillmentOrderBillingAccountListResponse indicates whether the call to create customer data in the billing system was successful. The order orchestration process is responsible for proceeding only if the call was successful.

**For more information,** see “Chapter 4: Understanding the Process Integration for Order Management”, Solution Assumptions and Constraints.

- The process integration provides an enterprise service, CommuncationsSalesOrderEBSV2.UpdateSalesOrder (and the respective connector service to Siebel), to allow the central fulfillment system to send updates back to Siebel CRM for fulfillment status, dates, and other attributes. The CFS is responsible for consolidating status updates and sending only updates that are significant to the CSR or end customer back to Siebel CRM.

**Note:** A central fulfillment system or order management system is not delivered in this release.

## Expectations of the Order Billing Interface from an Order Management System

This table provides a summary of the general billing integration dependencies on an order management system. For feature-specific dependencies, see the respective feature sections (for example, two-phase billing, time-based offers, and so forth).

| Number | Service | Comments |
|---|---|---|
| 1 | CommunicationsBillingEBSV1.ProcessFulfillmentOrderAccountList | The Order Management (OM) system must send all lines on the order destined *for a single billing system* to this service.<br>To handle Oracle BRM limitations on customer hierarchy updates, all the lines on the order targeted for a given billing system need to be sent to this service at the *same* time.<br>The promotion line needs to go to every billing system in which promotion components are targeted.<br>This service processes *only* lines with actions of ADD, UPDATE, and MOVE-ADD and ignores others. The OM system can choose to not send messages that do not have lines with these actions.<br>This service processes *only* lines with billing type of SERVICE BUNDLE, ITEM, SUBSCRIPTION, or DISCOUNT and lines with product type of OFFER (Promotion). It ignores the rest. The OM system can optionally filter lines based on this. |
| 2 | CommunicationsBillingEBSv1.ProcessFulfillmentOrderBilling<br>(INITIATE BILLING or FULFILL BILLING) | The OM system must send lines for promotions (product type is Promotion), account-level products (billing type is Item, Subscription, or Discount), service bundles (billing type is Service Bundle), or any combination of these destined *for a single billing system*. *Service Bundle* refers to the Service Bundle line *and all its children.* This service ignores other kinds of lines (for example, Non Service Bundle CP lines); therefore, the OM system can optionally filter them out.<br>The OM system must interface the promotion lines to billing either before the first service bundle or the account level product (including penalties) for the promotion are interfaced or along with it. This applies to both Initiate Billing and Fulfill Billing.<br>The OM system must interface MOVE-ADD lines with the corresponding MOVE-DELETE lines (linked using related line ID).The OM system must interface the one-time charge lines tied to service bundle lines with the service bundles (linked using related line ID).<br>The OM system must interface promotion penalty charges with the promotion line (linked using related line ID). |

# Solution Assumptions and Constraints

This section discusses the solution assumptions for the process integration for order management.

1. An order management system is not included as part of the Order to Bill PIP. The PIP can work with your order management system. It works with the Oracle Order to Activate PIP that includes integration with Oracle Order and Service Management (Oracle OSM).
2. The solution does not support an integration scenario in which multiple brands are defined within a single instance of Oracle BRM.
3. Once an order in Siebel is submitted for processing and successfully interfaced to billing, it *cannot be changed and resubmitted*. You should enforce this by defining rules in the Siebel state model. It can be revised and resubmitted for processing provided that the order has not reached a point of no return. The solution assumes that the order line reaches the point of no return once the line has been sent for billing fulfillment.
4. Follow-on orders are supported provided that the order management system in place handles them. That is, the order management system invokes the interface to billing for the follow-on order only after the original order has successfully been interfaced to billing and invokes requests for asset integration only after the original one has successfully interfaced to assets.
5. The Siebel Copy Orders feature does not regenerate the identifiers (asset integration Id) that uniquely identify the customer purchases on the copied order. This makes the copied orders invalid to back-end systems. Therefore, copied orders are not supported by Oracle AIA. Instead of copying orders, we recommend that you use the Siebel Favorites feature.
6. Regarding quantity support for service bundles and account-level products, the solution assumes that the auto-explode flag on service bundle products is set to Yes and that the customer is using Siebel Asset Based Ordering processes to enforce service item instantiation.
   - The service bundle line always has a quantity of 1 when the order is handed off from Siebel CRM to the integration with the integration creating a single service instance in Oracle BRM (per service bundle line on the Siebel order).

     No special handling exists for order quantity > 1 for products whose auto-explode flag in Siebel is set to No.

   - Quantity (and not extended quantity) on service bundle components or account-level products is interfaced to Oracle BRM; this creates purchased product or discount instances (one instance per product or discount purchased) with the specified quantity, which is used to determine charge calculation.
   - When an order line is interfaced to Siebel CRM assets it creates a single asset with the specified quantity.

     Additionally, the integration does *not* look at quantity changes on revisions, or change orders (for existing products) and therefore such changes are not communicated to Oracle BRM.
7. No special handling exists for shippable goods. No support is available for returns or credit orders.
8. Order lines that need to be sent to different billing systems will have different billing profiles.
   - This is a limitation only if the customer is also using the Oracle Communications Billing

and Revenue Management: Agent Assisted Billing Care process integration pack (PIP). The Billing Management flows available as part of that PIP, do not support the ability to display information from multiple billing systems for the same billing profile.

9. Order lines are interfaced to billing only after they have been provisioned.

   Based on this assumption, the service that interfaces the lines with billing creates the service instances, purchased product instances, purchased discount instances, or a combination of these as active. This applies to scenarios of single-phase billing, in which billing interface is called just once in FULFILL mode.

10. The Service Account, Billing Account, and Billing profiles are the same on all order lines (components) in a service bundle

    For service bundles, any integration logic that works on these fields looks only at the Service bundle line. This constraint also applies to onetime charges that are added for MACD, actions such as suspending or resuming a service, in that the integration ignores the Service Account, Billing Account, and Billing profiles on such a line. The charge generated by such an order line is applied to the balance group that the service instance points to.

    This is an Oracle BRM limitation and is enforced in Siebel.

11. The solution supports account-level default balance groups alone. A balance group in Oracle BRM can reference a single bill-info. This is the first billing profile that is referenced on the first order processed for an account.

    It follows that all services for a given account on the same order or subsequent orders need to reference the same billing profile; an order violating this assumption will fail billing integration with an Oracle BRM error.

    - If the order message contains more than one service being purchased, the integration (as a result of optimization), uses the billing profile on the first service for processing all of the services. In this case, the Oracle BRM validation and error are not raised.

    It follows that all account-level product purchases for a given account reference the same billing profile. Violation of this assumption does not result in failure because order billing integration ignores the billing profile specified on order lines for such products.

    The solution does support updating an existing billing profile in Siebel; such changes are synchronized with billing outside of the order integration flow.

12. In the case where an account is paying for its own services (and account-level products), the solution does not support changing the billing profile on existing services or account-level products to a different one using a change order:

    - Changing from one billing profile to another for a self paying account is not supported.
    - Changing from one paying parent to another for a subordinate account is supported.
    - Changing from one billing profile to another (while retaining the same paying parent) for a subordinate account is supported.
    - Changing from self-paying to nonpaying subordinate is not supported*.
    - Changing from nonpaying subordinate to self-paying is not supported.

This is an Oracle BRM limitation with respect to account-level balance group usage. Order integration to billing will fail with an Oracle BRM error for the preceding scenarios that are not supported. * - This specific scenario does not error but is not supported since it results in data that breaks the billing management integration flows.

13. Oracle BRM does not support a subordinate account having more than one paying parent.

    Any order changing the paying parent for a subordinate account using a new purchase has to include lines to change all the other services (and account-level products) for the subordinate account that was paid for by the old parent so that it can successfully interface customer data to Oracle BRM. .

    Transactions that do not comply with this assumption will fail with an Oracle BRM error when an order is interfacing customer data to Oracle BRM.

    Any order changing the paying parent for an existing service on a subordinate account will change the paying parent for all the other services (and account-level products) under that subordinate account. To ensure that Siebel CRM assets are in sync with Oracle BRM, we recommend that the change order to update the paying parent include an update for all the services (and account-level products) for a given subordinate account.

**For more information,** see Appendix B: Examples of Changing the Paying Parent on Subordinate Accounts.

14. Transfer of services (or account-level products) from one account to another is not supported.

**For more information**, see “Appendix A: Order Management: Matrix of MACD Actions Supported Per Billing Product Type”, Table B.

15. All lines within a service bundle reference products from the same billing system.

    Based on this assumption, a single Siebel CRM asset can be mapped to a service instance or a purchased product or discount instance in only one billing system.

16. The integration assumes that the service bundle product and its component products reference the same billing service type. This assumption applies only to component products that represent Oracle BRM products of type Subscription or BRM discounts. Violation of this assumption can result in Oracle BRM grouping the billed charges under the wrong bucket (bill-item). Nested service bundles do not have to have the same service type as the root parent service bundle.

# Oracle BRM Interfaces

The process integration for order management uses these services:

- PCM_OP_CUST_MODIFY_CUSTOMER
- PCM_OP_SUBCRIPTION_PURCHASE_DEAL
- PCM_OP_CUST_UPDATE_SERVICES
- PCM_OP_CUST_SET_STATUS

- PCM_OP_SUBCRIPTION_CANCEL_PRODUCT
- PCM_OP_SUBCRIPTION_CANCEL_DISCOUNT
- PCM_OP_SUBSCRIPTION_TRANSFER_SUBCRIPTION
- PCM_OP_SUBSCRIPTION_SET_PRODINFO
- PCM_OP_SEARCH
- PCM_OP_READ_FLDS
- PCM_OP_SUBSCRIPTION_SET_BUNDLE
- PCM_OP_SUBSCRIPTION_SET_DISCOUNT_STATUS
- PCM_OP_SUBSCRIPTION_SET_PRODUCT_STATUS
- PCM_OP_CUST_CREATE_PROFILE
- PCM_OP_CUST_DELETE_PROFILE
- PCM_OP_CUST_MODIFY_PROFILE

**For more information,** see *Oracle Communications Billing and Revenue Management (BRM) Documentation, "*BRM Documentation," Reference, API reference.

# Siebel CRM Interfaces

The process integration for order management uses these Siebel CRM interfaces:

- SISOMBillingSubmitOrderWebService: this is the outbound Siebel web service used for submitting orders.
- SWIOrderUpsert and SWIOrderUpsertSubProcess: these are the inbound Siebel web services used to update the order information back to Siebel.
- SWI Special Rating – Synchronize Process

**For more information** about the web services, see the *Siebel Order Management Guide Addendum for Communications*, "Web Services Reference."

# Industry AIA Components

The process integration for order management uses these industry components:

- SalesOrderEBO
- ProcessSalesOrderFulfillmentEBM
- FulfillmentOrderEBO
- ProcessFulfillmentOrderBillingAccountListEBM
- ProcessFulfillmentOrderBillingAccountListResponseEBM

- ProcessFulfillmentOrderBillingEBM
- ProcessFulfillmentOrderBillingResponseEBM
- UpdateSalesOrderEBM
- InstalledProductEBO
- ProcessInstalledProductSpecialRatingSetListEBM

The industry enterprise business object (EBO) and enterprise business message XML schema (EBM XSD) files are located here:

http://<server name>:<port number>/
/AIAComponents/EnterpriseObjectLibrary/Industry/Communications/EBO/

The industry enterprise business service web service description language (EBS WSDL) files are located here:

http://<server name>:<port number>/
AIAComponents/EnterpriseBusinessServiceLibrary/Industry/Communications/EBO/

For detailed documentation about individual EBOs, click the EBO Name link on the Integration Scenario Summary page of the Oracle AIA Console. You can also use the Integration Scenario Summary page to search for and view integration scenarios that use a particular EBO or EBS.

**For more information,** see *Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide,* "Using the BSR UI to View Integration Scenarios."

EBOs can be extended, for instance, to add new data elements. These extensions are protected, and they will remain intact after a patch or an upgrade.

**For more information,** see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide,* "Extensibility for Oracle AIA Artifacts."

**For more information** about the changes made for each EBO and EBM, see the appendix in the My Oracle Support document number 988374.1.

# Integration Services

These services are delivered with this integration:

- ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer
- ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl
- UpdateSalesOrderSiebelCommsProvABCSImpl
- CommunicationsSalesOrderEBSV2
- CommunicationsBillingEBSV1
- CommunicationsBillingResponseEBSV1
- CommsProcessFulfillmentOrderBillingAccountListEBF
- ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl

  - ProcessFulfillmentOrderBillingBRMCommsAddSubProcess
  - ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess
  - ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess
  - ProcessFulfillmentOrderBillingBRMCommsUpdateSubProcess
  - ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess
- CommunicationsInstalledProductEBSV2
- ProcessInstalledProductSpecialRatingSetListSiebelCommsJMSConsumer
- ProcessInstalledProductSpecialRatingSetListSiebelCommsReqABCSImpl
- ProcessInstalledProductSpecialRatingSetListBRMCommsProvABCSImpl

Some of these services have been enabled to use Session Pool Manager.

**For more information** about Session Pool Manager, see Appendix H: Using Session Pool Manager.

Use the Integration Scenario Summary page in the Oracle AIA Console to search for and view integration scenarios that use a particular ABC service.

**For more information,** see *Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide,* "Using the BSR UI to View Integration Scenarios."

## ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer

The ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer is implemented as an ESB process.

This consumer listens over the AIA_SALESORDERJMSQUEUE into which Siebel enqueues the simple object access protocol (SOAP)-wrapped Siebel Order application business message (ABM). This consumer dequeues the messages from this queue, unwraps the message from the SOAP envelope, and routes the Siebel ABM to the ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl.

## ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl

The ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl is implemented as a business process execution language (BPEL) process with a single operation: process.

This service is invoked when an order is submitted in the Siebel Application. This service is the Siebel ABCS implementation, which converts the Siebel ABM into the Order EBM before invoking the CommunicationsSalesOrderEBSV2. The service looks up the cross-reference values for the customer account ID, billing profile ID, pay profile ID, and product or discount ID to find common IDs to appropriately populate the Order EBM. In the case of promotions and service bundles, if the cross-reference values are not present, new cross-reference values are created.

This service also creates the requisite cross-reference values for the order ID, order line ID, installed product ID, account ID, bill profile ID, pay profile ID, contact ID, and address ID between Siebel values and generated common values.

## UpdateSalesOrderSiebelCommsProvABCSImpl

The UpdateSalesOrderSiebelCommsProvABCSImpl is a BPEL process with one operation: UpdateSalesOrder. It accepts the UpdateSalesOrderEBMas, the input from the CommunicationsSalesOrderEBSV2, and uses the order information in the input message to update the orders in Siebel CRM.

The main functions of this service are:

- Updating the order line status: updates the order line status back to Siebel CRM.
- Enriching the order: enriches the information back to Siebel CRM from a central fulfillment system to facilitate customer care, service, and asset-based ordering. It is also used to update or enrich the order line items with fulfillment attributes back to Siebel CRM. Among these attributes are service IDs and allocated resources such as port number and IP address.
- Updating the order header: enriches the order header to Siebel CRM.

This process is an asynchronous, one-way service.

This service is now SPM (Session Pool Manager) enabled.

**For more information** about SPM, see Appendix H: Using Session Pool Manager.

## CommunicationsSalesOrderEBSV2

The CommunicationsSalesOrderEBSV2 is implemented as an ESB service to perform routing wherever needed. The CommunicationsSalesOrderEBSV2 is the Order Entity EBS that has the following operations used in order integration:

- ProcessSalesOrderFulfillment - The ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl invokes the ProcessSalesOrderFulfillment operation to send messages to CFS.
- UpdateSalesOrder - The UpdateSalesOrder Operation calls the UpdateSalesOrderSiebelCommsProvABCSImpl, which updates the order status back to Siebel. This operation can be called by CFS to update order information back to Siebel CRM.

**For more information** about this EBS, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs."

## CommunicationsBillingEBSV1

The CommunicationsBillingEBSV1 is implemented as an ESB service to perform routing to billing systems. The CommunicationsBillingEBSV1 is the Billing EBS that has the following operations used in order integration:

- ProcessFulfillmentOrderBillingAccountList - The CFS invokes this operation to create customer information in the billing system.
- ProcessFulfillmentOrderBilling – The CFS invokes this operation to create billing transaction

data in the billing system.

**For more information** about this EBS, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Developing EBSs" *and Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs."

## CommunicationsBillingResponseEBSV1

The CommunicationsBillingResponseEBSV1 is implemented as an ESB service to perform response routing from billing systems. The CommunicationsBillingResponseEBSV1 is the billing EBS that has the following operations used in order integration:

- ProcessFulfillmentOrderBillingAccountListResponse – ProcessFulfillmentOrderBillingAccountListEBF invokes this operation to respond to CFS.
- ProcessFulfillmentOrderBillingResponse – ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl invokes this operation to respond to CFS.

For error scenarios, a response message can be optionally sent back to the order management system. The decision whether to send a response message back to the order management system is done based on the responseCode attribute of the DataArea of the incoming EBM (ProcessFulfillmentOrderBillingEBM) from the order management system.

If the responseCode value in the incoming EBM is REQUIRED_FOR_BUSINESS_AND_SYSTEM_ERRORS, the response message is sent back to the order management system for all errors. However, if the responseCode value is REQUIRED_FOR_BUSINESS_ERRORS, the response message is only sent back to the order management system for business errors.

**Caution:** With errors, Oracle OSM and the OSM AIA cartridges do not expect a response back. Instead, they use the Oracle AIA order fallout notification to both generate a trouble ticket and change the order and line status to indicate failure.

**For more information** about this EBS, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs."

## CommsProcessFulfillmentOrderBillingAccountListEBF

The CommsProcessFulfillmentOrderBillingAccountListEBF is implemented as an asynchronous BPEL process. It performs these operations:

- Receives the ProcessFulfillmentOrderBillingAccountListEBM from the CFS with the target billing system identified.
- Transforms the message into the ProcessBillingAccountListEBM appropriately.
- Invokes the CommunicationsBillingEBSV1.ProcessBillingAccountList, which in turn invokes the CommsProcessBillingAccountListEBF.

- Awaits response from CommunicationsBillingResponseEBSV1.ProcessBillingAccountListResponse.
- On receipt of response, calls CommunicationsBillingResponseEBSV1.ProcessFulfillmentOrderBillingAccountList to send the response back to CFS.


### CommsProcessFulfillmentOrderBillingAccountListEBF

This process has the following operations.

- Operation: initiate

  This is an asynchronous operation to start the CommsProcessFulfillmentOrderBillingAccountListEBF.

- Operation: CallbackResponse

  This is an asynchronous callback operation. It makes a call back to the calling process, and passes a FaultMsg in the EBMHeader in case of any error received from CommsProcessBillingAccountListEBF.

**For more information** about enterprise business flows (EBFs), see *Oracle Application Integration Architecture – Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Constructing EBFs" and *Oracle Application Integration Architecture – Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs," Enterprise Business Flow (EBF) Processes.

## ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl

The ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl consists of a BPEL process with one operation: ProcessBilling. It receives the Order EBM and then converts the message into a BRM-specific message based on which opcode needs to be invoked.

This service communicates with Oracle BRM using the custom Java EE Connector Architecture (JCA) adapter provided by Oracle BRM. It uses the default capability of the custom JCA adapter to define unit transactions for every order. (Do all or none.)

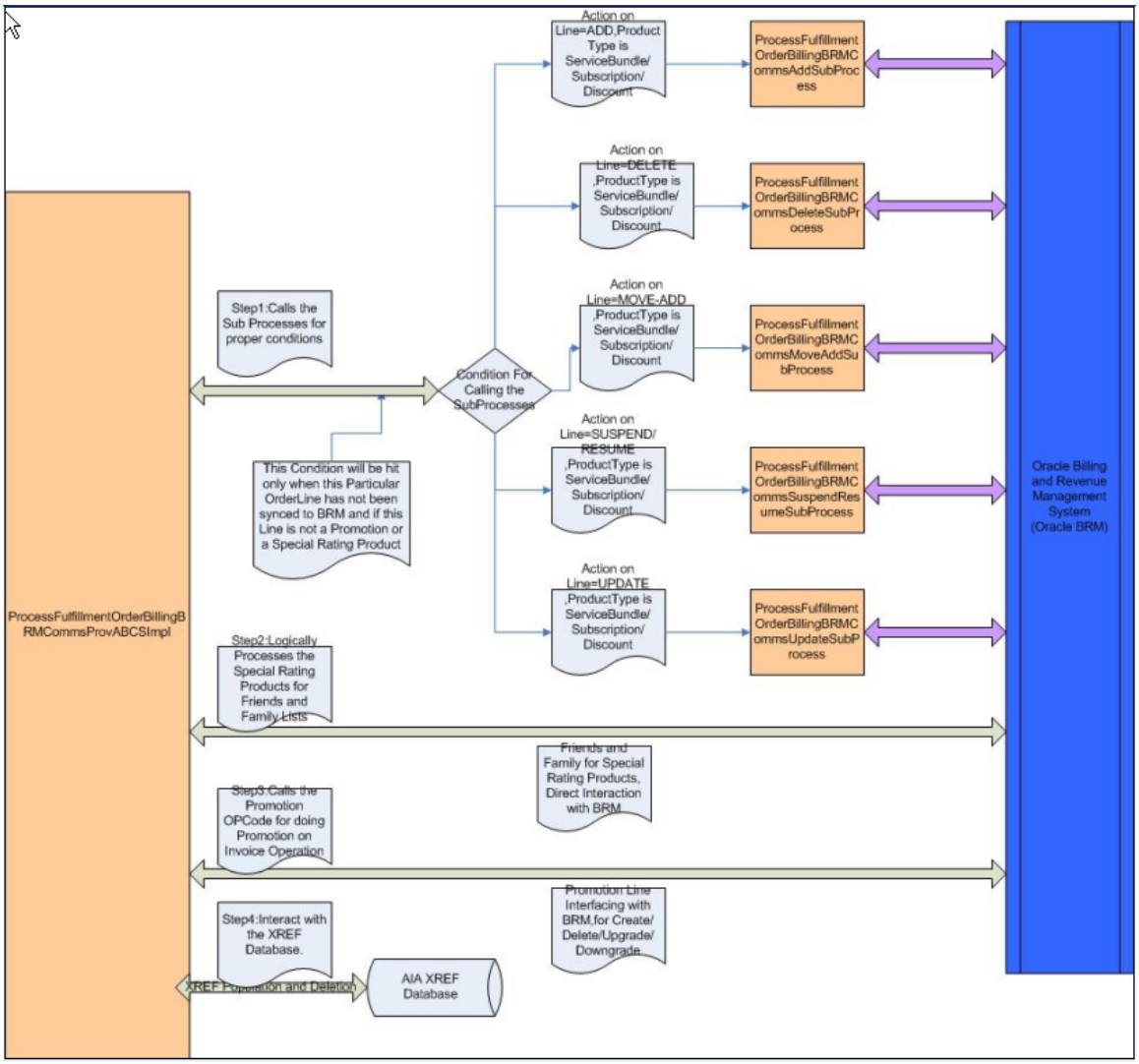
The ProcessFulfillmentOrderBilling operation in the CommunicationsBillingOrderEBSV1 invokes this server if the target billing system is Oracle BRM. The routing to the right Oracle BRM instance is done using dynamic end point binding in the BPEL process using the target application that is already decided.

This service accepts the appropriate ProcessFulfillmentOrderBillingEBM and is responsible for transforming to the relevant Oracle BRM ABM and invoking the corresponding opcode.

ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl mainly performs the following activities

- Evaluates the product type of the order line and the action code. If the particular order line is a ServiceBundle/Subscription/Discount and if this line has never been interfaced to Oracle BRM, then it proceeds to call the subprocesses:
  - For ActionCode = ‘ADD’ and BillingMode = ‘INITIATE BILLING’ or ‘FULFILL BILLING’, ProcessFulfillmentOrderBillingBRMCommsAddSubProcess is called.
  - For ActionCode = ‘SUSPEND’ or ‘RESUME’ and BillingMode = ‘FULFILL BILLING’, ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess is called.
  - For ActionCode = ‘DELETE’ and BillingMode = ‘FULFILL BILLING’, ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess is called.
  - For ActionCode = ‘UPDATE’ and BillingMode = ‘FULFILL BILLING’, ProcessFulfillmentOrderBillingBRMCommsUpdateSubProcess is called.
  - For ActionCode = ‘MOVE-ADD’ and BillingMode = ‘FULFILL BILLING’, ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess is called.
- For friends and family orders, where the order has one or more than one SpecialRatingProduct as an OrderLine, the ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl Process calls one of the following Oracle BRM opcodes depending on the nature of the action to be performed:
  - For New Order, the PCM_OP_CUST_CREATE_PROFILE opcode is called. After this call, the Oracle BRM POID is cross-referenced and populated in the AIA XREF database.
  - For Deleting the Special Rating Product, the PCM_OP_CUST_DELETE_PROFILE opcode is called. After this call, the Oracle BRM POID is cross-referenced and deleted from the AIA XREF database.
  - For Deleting the Special Rating Product, the PCM_OP_CUST_MODIFY_PROFILE opcode is called.
- For Promotion on Invoice, ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl calls the PCM_OP_SUBSCRIPTION_SET_BUNDLE opcode and different values are passed depending on the particular functional operation.
- After all of these activities, the data is cross-referenced to the AIA XREF database.

This diagram shows the data that is cross-referenced to the AIA XREF database:




## Data that is cross-referenced to the AIA XREF database

This service calls the following subprocesses in a synchronous fashion to perform various billing-related activities:

- ProcessFulfillmentOrderBillingBRMCommsAddSubProcess
- ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess
- ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess
- ProcessFulfillmentOrderBillingBRMCommsUpdateSubProcess
- ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess

## ProcessFulfillmentOrderBillingBRMCommsAddSubProcess

The ProcessFulfillmentOrderBillingBRMCommsAddSubProcess is a synchronous BPEL process that is called by the ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl. This call depends on the action code present on the order line and also the type of product.

The ProcessFulfillmentOrderBillingBRMCommsAddSubProcess is called for a service bundle, account-level product, or account-level discount that is being newly added either as a part of a new order or an update order and that has an action code of ADD.

The ProcessFulfillmentOrderBillingBRMCommsAddSubProcess is never called for any onetime penalty charges that also have an action code of ADD, but are being added as a part of the MACD operation performed on a service bundle or a promotion.

The ProcessFulfillmentOrderBillingBRMCommsAddSubProcess receives a custom message that has the ProcessFulfillmentOrderBillingEBM, XREFPopulate, and XREFDelete DataStructure.

The structure of the message coming in to the ProcessFulfillmentOrderBillingBRMCommsAddSubProcess comprises:

- ProcessFulfillmentOrderBillingEBM
- XREFPopulate
- XREFDelete

Depending on the type of product for every OrderLine, the following operations are performed in the ProcessFulfillmentOrderBillingBRMCommsAddSubProcess:

1. When the product type is service bundle, then this BPEL process accumulates all of the children inside this service bundle and calls the PCM_OP_CUST_MODIFY_CUSTOMER opcode. During this call, the ProcessFulfillmentOrderBillingBRMCommsAddSubProcess also transforms the ProcessFulfillmentOrderBillingEBM into a BRM-specific message.
2. When the product type is an account-level subscription, discount, or item, then this BPEL process calls the PCM_OP_SUBSCRIPTION_PURCHASE_DEAL opcode. During this call, the ProcessFulfillmentOrderBillingBRMCommsAddSubProcess also transforms the ProcessFulfillmentOrderBillingEBM into an Oracle BRM-specific message.
3. After these two calls are successfully carried out, this BPEL process captures the POID (ObjectIdentifier) returned by Oracle BRM and populates the XREFPopulateData.
4. For ITEM, the POID(ObjectIdentifier) is returned by Oracle BRM only during INITIATE BILLING mode.

This service communicates with Oracle BRM using the custom JCA adapter provided by BRM. It uses the default capability of the custom JCA adapter to define unit transactions for every order. (Do all or none.)

This service supports two modes of billing:

- initiatebilling
- fulfillbilling

## ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess

The ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess is a synchronous BPEL process that is called by the ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl. This call depends on the action code present on the order line and also the type of product. It has one operation: processBillingMove.

The structure of the message coming in ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess comprises:

- ProcessFulfillmentOrderBillingEBM
- XREFPopulate
- XREFDelete

When the action code on the order line is MOVE-ADD and the product type is a service bundle, an account-level product, or an AccountLevelDiscount, the ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess is called.

The following operations can be performed as a part of the ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess:

- Simple Move-Add of the service bundles from one location to another.

  During this scenario, only the XREFs are repointed from. No Oracle BRM interaction happens in this operation.

- Update of the service ID for a particular service bundle.

  During this scenario, users can update the service ID for one or more service bundles as a part of Move-Add.

  ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess converts the ProcessFulfillmentOrderBillingEBM into an Oracle BRM-specific message and calls the PCM_OP_CUST_UPDATE_SERVICES Oracle BRM opcode.

- Price Override.

  During this scenario, users can change the PriceOverride on a product line.

  ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess converts the ProcessFulfillmentOrderBillingEBM into an Oracle BRM-specific message and calls the PCM_OP_SUBSCRIPTION_SET_PRODINFO Oracle BRM opcode.

- Discount Override.

  During this scenario, users can change the DiscountOverride on a product line.

  ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess converts the ProcessFulfillmentOrderBillingEBM into an Oracle BRM-specific message and calls the PCM_OP_SUBSCRIPTION_SET_PRODINFO Oracle BRM opcode.

- All the preceding Move-Add scenarios can be accompanied with or without a onetime penalty charge.

When a onetime penalty charge is associated, the ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess converts the ProcessFulfillmentOrderBillingEBM into an Oracle BRM-specific message and calls the PCM_OP_SUBSCRIPTION_PURCHASE_DEAL Oracle BRM opcode.

## ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess

The structure of the message coming in the ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess comprises:

- ProcessFulfillmentOrderBillingEBM
- XREFPopualate
- XREFDelete

The ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess is a synchronous BPEL process that is called by the ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl. This call depends on the action code present on the order line and also the type of product. It has one operation: processBillingSuspendResume.

When the action code is SUSPEND or RESUME and the ProductType is a service bundle or an account-level subscription or account-level discount, then the ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess is called.

The following operations are done by this process:

- When the action code is SUSPEND or RESUME and the product type is a service bundle.

  ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess converts the ProcessFulfillmentOrderBillingEBM into an Oracle BRM-specific message and calls the PCM_OP_CUST_SET_STATUS Oracle BRM opcode.

  When the action code is SUSPEND, then the ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess passes the Flag= 10102.

  When the action code is RESUME, then the ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess passes the Flag= 10101.

- When the action code is SUSPEND or RESUME and the product type is Account Level Discount:

  ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess converts the ProcessFulfillmentOrderBillingEBM into an Oracle BRM-specific message and calls the PCM_OP_SUBSCRIPTION_SET_DISCOUNT_STATUS Oracle BRM opcode.

  When the action code is SUSPEND, then the ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess passes the Flag= 2.

  When the action code is RESUME, then the ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess passes the Flag= 1.

- When the action code is SUSPEND or RESUME and the product type is Account Level Subscription.

  ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess converts the ProcessFulfillmentOrderBillingEBM into an Oracle BRM-specific message and calls the PCM_OP_SUBSCRIPTION_SET_PRODUCT_STATUS Oracle BRM opcode.

  When the action code is SUSPEND, then the ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess passes the Flag= 2.

  When the action code is RESUME, then the ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess passes the Flag= 1.

  For Operation 1, a onetime penalty charge may or may not be associated.

- When a onetime penalty charge is associated with the service bundle, then depending on the action code, the onetime charge gets added in the following manner:

  When the action code is SUSPEND, the onetime charge gets added first.

  ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess converts the ProcessFulfillmentOrderBillingEBM into an Oracle BRM-specific message and calls the PCM_OP_SUBSCRIPTION_PURCHASE_DEAL Oracle BRM opcode.

  After this onetime charge is added, then Operation 1 is run to SUSPEND the service bundle.

  When the action code is RESUME, the onetime charge gets added after the service bundle is resumed.

  Operation 1 is run to RESUME the service bundle.

  After this, the onetime charge gets added:

  ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess converts the ProcessFulfillmentOrderBillingEBM into an Oracle BRM-specific message and calls the PCM_OP_SUBSCRIPTION_PURCHASE_DEAL Oracle BRM opcode.

## ProcessFulfillmentOrderBillingBRMCommsUpdateSubProcess

The ProcessFulfillmentOrderBillingBRMCommsUpdateSubProcess is a synchronous BPEL process that is called by the ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl. This call depends on the action code present on the order line and also the type of product. It has one operation: processBillingUpdate.

The structure of the message coming in ProcessFulfillmentOrderBillingBRMCommsUpdateSubProcess comprises:

- ProcessFulfillmentOrderBillingEBM
- XREFPopualate
- XREFDelete

When the action code is UPDATE and the product type is a service bundle or an account-level subscription or account-level discount, then the ProcessFulfillmentOrderBillingBRMCommsUpdateSubProcess is called.

This process supports operations 2, 3, 4, and 5 as mentioned in the ProcessFulfillmentOrderBillingBRMMoveAddSubProcess description.

## ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess

The ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess is a synchronous BPEL process that is called by ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl. This call depends on the action code present on the order line and also the type of product. It has one operation: processBillingDelete.

When the action code is DELETE and the product type is a service bundle or an account-level subscription or account-level discount, then the ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess is called.

The following operations are done by this process:

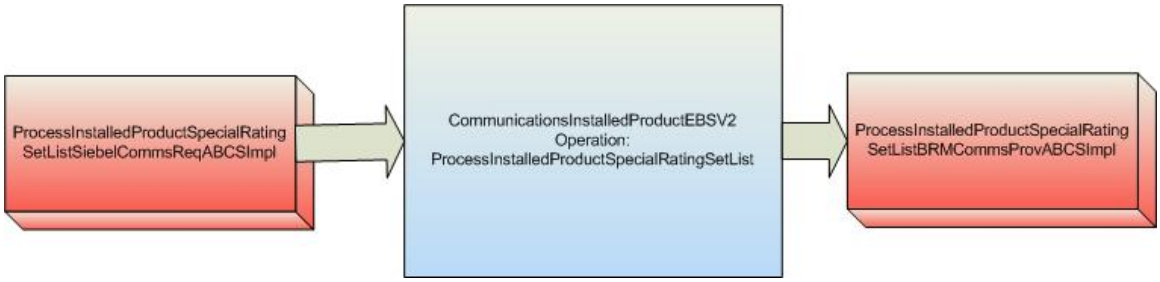
- When the action code is DELETE and the product type is Service Bundle, the ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess converts the ProcessFulfillmentOrderBillingEBM into an Oracle BRM-specific message and calls the PCM_OP_CUST_SET_STATUS Oracle BRM opcode.
- The ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess passes the StatusFlag=3 in this case.
- When the action code is DELETE and the product type is Discount, the ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess converts the ProcessFulfillmentOrderBillingEBM into an Oracle BRM-specific message and calls the PCM_OP_SUBSCRIPTION_CANCEL_DISCOUNT Oracle BRM opcode.
- When the action code is DELETE and the product type is Account Level Subscription, the ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess converts the ProcessFulfillmentOrderBillingEBM into an Oracle BRM-specific message and calls the PCM_OP_SUBSCRIPTION_CANCEL_PRODUCT Oracle BRM opcode.
- During these operations, the ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess also checks for the existence of any onetime penalty charge. If present, then the ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess converts the ProcessFulfillmentOrderBillingEBM into an Oracle BRM-specific message and calls the PCM_OP_SUBSCRIPTION_PURCHASE_DEAL Oracle BRM opcode.

## CommunicationsInstalledProductEBSV2

The CommunicationsInstalledProductEBSV2 is implemented as an ESB service to perform routing wherever needed. The Installed Product EBS is the Asset Entity EBS and has the following operation used in the order integration:

- ProcessInstalledProductSpecialRatingSetList

The ProcessInstalledProductSpecialRatingSetListSiebelCommsReqABCSImpl invokes this operation with ProcessInstalledProductSpecialRatingSetListEBM as input. This operation then routes the message to ProcessInstalledProductSpecialRatingSetListBRMCommsProvABCSImpl.




**For more information** about this EBS, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs."

## ProcessInstalledProductSpecialRatingSetListSiebelCommsJMSConsumer

The ProcessInstalledProductSpecialRatingSetListSiebelCommsJMSConsumer is implemented as an ESB process.

This consumer listens over the AIA_SPECIALRATINGJMSQ into which Siebel enqueues the SOAP-Wrapped Siebel Special Rating List ABM. This consumer dequeues the messages from this queue, unwraps the message from the SOAP envelope, and routes the Siebel ABM to ProcessInstalledProductSpecialRatingSetListSiebelCommsReqABCSImpl

## ProcessInstalledProductSpecialRatingSetListSiebelCommsReqABCSImpl

The ProcessInstalledProductSpecialRatingSetListSiebelCommsReqABCSImpl is a BPEL process with one operation: ProcessInstalledProductSpecialRatingSetList. This service accepts as input the Siebel SWISpecialRatingListIO and converts it to the ProcessInstalledProductSpecialRatingSetListEBM structure before invoking the CommunicationsInstalledProductEBSV2.

This service is invoked when an existing customer (an account already exists in Siebel CRM and is already synced to Oracle BRM) modifies the existing special rating (friends and family) profile in Siebel.

The service looks up the cross-reference values for the customer account ID and installed product ID to find common IDs to appropriately populate the EBM.

## ProcessInstalledProductSpecialRatingSetListBRMCommsProvABCSImpl

The ProcessInstalledProductSpecialRatingSetListBRMCommsProvABCSImpl is implemented as a BPEL process with a single operation: ProcessInstalledProductSpecialRatingSetList.

This service is invoked when an existing customer (an account already exists in Siebel CRM and is already synced to Oracle BRM) modifies the existing special rating (friends and family) profile in Siebel.

The CommunicationsInstalledProductEBSV2 invokes this service to synchronize the changes in a special rating profile to Oracle BRM. This service is the Oracle BRM ABCS implementation, which converts the ProcessInstalledProductSpecialRatingSetList into the Oracle BRM ABM before invoking the Oracle BRM opcode PCM_OP_CUST_MODIFY_PROFILE.

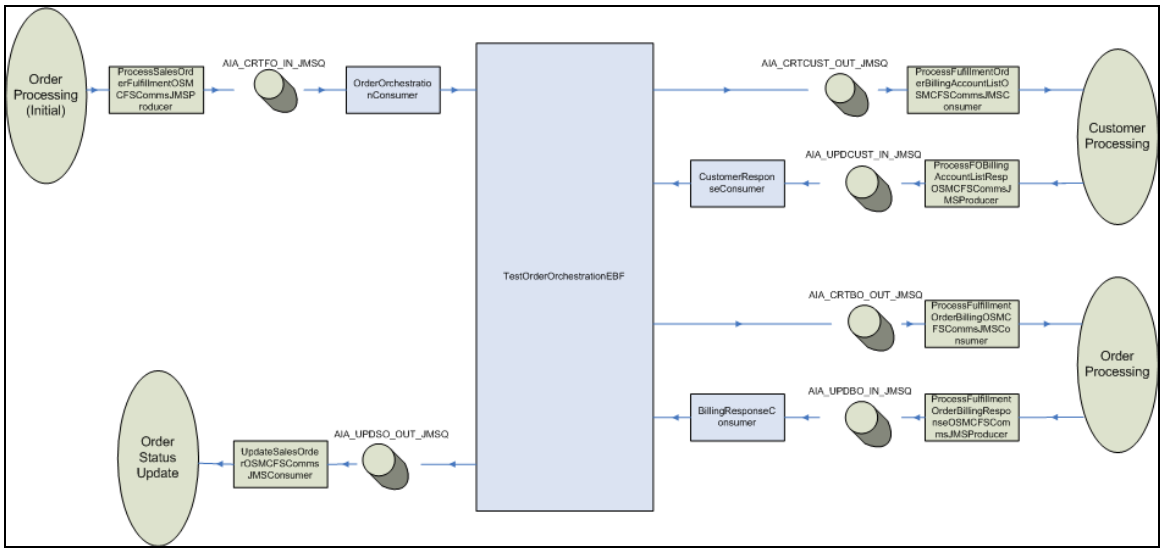
## TestOrderOrchestrationEBF

The TestOrderOrchestrationeEBF is a tool delivered so that customers can test whether the installation of the Order to Bill PIP was successful. This is not a supported product feature.

This service is a combination of:

- A main BPEL process that is the TestOrderOrchestrationEBF that orchestrates the flow using various queue interactions.
- Three ESB consumer processes to invoke or resume the main process on message received in various Oracle AIA queues: OrderOrchestrationConsumer, CustomerResponseConsumer, and BillingResponseConsumer.
- Three ESB consumers to pick up messages enqueued by the main process and pass them to various Oracle AIA flows: ProcessFufillmentOrderBillingAccountListOSMCFSCommsJMSConsumer, ProcessFulfillmentOrderBillingOSMCFSCommsJMSConsumer, and UpdateSalesOrderOSMCFSCommsJMSConsumer
- Three BPEL producers to enqueue response messages from various Oracle AIA flows for the main process: ProcessSalesOrderFulfillmentOSMCFSCommsJMSProducer, ProcessFOBillingAccountListRespOSMCFSCommsJMSProducer, and ProcessFulfillmentOrderBillingResponseOSMCFSCommsJMSProducer

This diagram shows how this service interacts with others.




TestOrderOrchestrationEBF

Here is how the TestOrderOrchestrationEBF processes orders that are submitted from Siebel CRM for the purpose of testing the delivered installed services.

1. CommunicationsSalesOrderEBSV2.ProcessSalesOrderFulfillment calls the ProcessSalesOrderFulfillmentOSMCFSCommsJMSProducer to enqueue a message in AIA_CRTFO_IN_JMSQ.
2. OrderOrchestrationConsumer picks up the message from the AIA_CRTFO_IN_JMSQ, transforms it into ProcessSalesOrderFulfillmentEBM, and invokes TestOrderOrchestrationEBF.
3. TestOrderOrchestrationEBF creates ProcessFulfillmentOrderBillingAccountListEBM and enqueues the message in AIA_CRTCUST_OUT_JMSQ.
4. ProcessFufillmentOrderBillingAccountListOSMCFSCommsJMSConsumer picks up that message and passes it to Customer Processing.
5. On response from Customer Processing, ProcessFOBillingAccountListRespOSMCFSCommsJMSProducer enqueues the message into AIA_UPDCUST_IN_JMSQ.
6. CustomerResponseConsumer picks up this message and resumes TestOrderOrchestrationEBF.
7. TestOrderOrchestrationEBF creates ProcessFulfillmentOrderBillingEBM and enqueues the message in AIA_CRTBO_OUT_JMSQ.
8. ProcessFulfillmentOrderBillingOSMCFSCommsJMSConsumer picks up that message and passes it to Order Processing for billing.
9. On response from Order Processing for billing, ProcessFulfillmentOrderBillingResponseOSMCFSCommsJMSProducer enqueues the message into AIA_UPDBO_IN_JMSQ.
10. BillingResponseConsumer picks up this message and resumes TestOrderOrchestrationEBF.
11. TestOrderOrchestrationEBF creates UpdateSalesOrderEBM and enqueues the message in AIA_UPDSO_OUTJMSQ.
12. UpdateSalesOrderOSMCFSCommsJMSConsumer picks up that message and passes it to the Order Status Update flow.

**For more information** about EBFs, see *Oracle Application Integration Architecture – Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Constructing EBFs" and *Oracle Application Integration Architecture – Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs," Enterprise Business Flow (EBF) Processes.

# Chapter 5: Configuring the Process Integration for Order Management

This chapter discusses how to:

- Set up Fusion Middleware (FMW).
- Set up Oracle Communications Billing and Revenue Management (Oracle BRM).
- Set up Siebel Customer Relationship Management (Siebel CRM).
- Work with domain value maps (DVMs).
- Work with cross-references.
- Handle error notifications.
- View enterprise business object (EBO) implementation maps (EIMs).
- Configure the process integration for order management.

## Setting Up FMW

This section discusses the post-installation configurations for the Order to Bill PIP:

- Install and configure the Oracle BRM Java EE Connector Architecture (JCA) adapter.

**For more information,** see *Oracle Communications Billing and Revenue Management (BRM) Documentation*, “Service Integration Components,” JCA Resource Adapter.

- In the BPEL Process Manager, set these properties:

  Location "'<$SOA_Home>/j2ee/oc4j_soa/config/transaction-manager.xml".'

  Property Name:transaction-timeout="10800" (all occurrences of transaction-timeout)

  [Normal default value is 180]

  Location <$SOA_Home>/j2ee/oc4j_soa/application-deployments/orabpel/ejb_ob_engine

  File Name:orion-ejb-jar.xml

  Property Name:transaction-timeout="9600" (all occurrences of transaction-timeout)

  [Normal default value is 60]

- In the BPEL Console, go to the Administration page and change this property value:

'syncMaxWaitTime'--'Delivery result receiver maximum wait time'--Give a value ='7200'(Purpose of this property- 'The maximum time the process result receiver will wait for a result before returning. Results from asynchronous BPEL processes are retrieved synchronously using a receiver that will wait for a result from the container. ')

You must add the -DHTTPClient.disableKeepAlives=true property in the opmn.xml file. The opmn.xml file is available here: SOA_HOME/opmn/conf/opmn.xml.

After you add this property, service oriented architecture (SOA) does not use the same transmission control protocol (TCP) connection more than once to call a Siebel web service. Simultaneous calls to the same Siebel web service cause errors if the same TCP connection is used. This property should be added in the startup parameters of oc4j_soa. After you add the property, the opmn.xml file looks like this:

```
<process-type id="oc4j_soa" module-id="OC4J" status="enabled">
               <module-data>
                  <category id="start-parameters">
                     <data id="java-options" value="-server -
Xmx2048M -Xms2048M -XX:MaxPermSize=1024M -XX:MaxNewSize=614m -
XX:NewSize=614m -XX:AppendRatio=3 -XX:SurvivorRatio=6 -
Djava.security.policy=$ORACLE_HOME/j2ee/oc4j_soa/config/java2.policy
-Djava.awt.headless=true -Dhttp.webdir.enable=false -
Doraesb.home=/slot/ems1936/oracle/product/10.1.3.1/OracleAS_1/integr
ation/esb -Dhttp.proxySet=false -Doc4j.userThreads=true -
Doracle.mdb.fastUndeploy=60 -Doc4j.formauth.redirect=true -
Djava.net.preferIPv4Stack=true -
Dorabpel.home=/slot/ems1936/oracle/product/10.1.3.1/OracleAS_1/bpel
-
Xbootclasspath^/p:/slot/ems1936/oracle/product/10.1.3.1/OracleAS_1/b
pel/lib/orabpel-boot.jar -Dhttp.proxySet=false -
Daia.home=/slot/ems1936/oracle/product/AIA_HOME" -
DHTTPClient.disableKeepAlives=true/>
                  </category>
                  <category id="stop-parameters">
                     <data id="java-options" value="-
Djava.security.policy=$ORACLE_HOME/j2ee/oc4j_soa/config/java2.policy
-Djava.awt.headless=true -Dhttp.webdir.enable=false" />
                  </category>
               </module-data>
               <start timeout="600" retry="2" />
               <stop timeout="120" />
               <restart timeout="720" retry="2" />
               <port id="default-web-site" range="12501-12600"
protocol="ajp" />
               <port id="rmi" range="12401-12500" />
               <port id="rmis" range="12701-12800" />
               <port id="jms" range="12601-12700" />
               <process-set id="default_group" numprocs="1" />
            </process-type>
```

***Warning:*** While adding the -DHTTPClient.disableKeepAlives=true property in the opmn.xml file, you must be careful. The opmn.xml file is required for the SOA server startup and any error in this file may cause the server to fail.

- To prevent multiple retries by the transaction manager:

  Modify the file $ORACLE_HOME/j2ee/{oc4j_instance name}/config/transaction-manager.xml to change the property value 'retry-count' of commit-coordinator from 4 to 0:
  <commit-coordinator retry-count="4"> to <commit-coordinator retry-count="0">

- To prevent the retries by ESB:

  Overwrite the ESB retry entries to disable the retries in the orion-application.xml file present in these directories:
  $ORACLE_HOME/j2ee/oc4j_soa/application-deployments/esb-rt and
  $ORACLE_HOME/j2ee/oc4j_soa/application-deployments/esb-dt with the following values after backing up the original file.

  ```
  <property name="InboundRetryCount" value="0" />
  <property name="InboundRetryInterval" value="0" />
  <property name="InboundRetryEnabled" value="false" />
  <property name="OutboundRetryCount" value="0" />
  <property name="OutboundRetryInterval" value="0" />
  <property name="OutboundRetryEnabled" value="false" />
  ```

  Alternatively, change these property values in the $ORACLE_HOME//integration/esb/config/esb_config.ini to change the retry values

- To enable resequencer, add the following properties in the esb_config.ini file present in this location:

  $ORACLE_HOME//integration/esb/config

  ```
  #ESBSequencer
  EnableResequencer = true
  ResequencerWorkerThreadPoolSize = 5
  ResequencerLockerThreadSleep = 1000
  ResequencerMaxRowsRetrieved = 100
  ```

These steps are also included in the *Oracle AIA Installation and Upgrade Guide*.

# Setting Up Oracle BRM

No setup steps are required for Oracle BRM except for installing and configuring the Oracle BRM JCA adapter.

**For more information,** see *Oracle Communications Billing and Revenue Management (BRM) Documentation*, “Service Integration Components,” JCA Resource Adapter.

# Setting Up Siebel CRM

For some Siebel CRM interfaces, in Siebel, you must set the process property UTCCanonical to Y.

**For more information** about which Siebel CRM interfaces require you to enable the UTCCanonical process property, see instructions for ACR 474 and 508 in the *Siebel Maintenance Release Guide*.

Perform the following Oracle Advanced Queuing (AQ) configurations:

- For the order flow: Configure SISOMBillingSubmitOrderWebService Siebel outbound workflow to enqueue the Siebel messages in AIA_SALESORDERJMSQUEUE.
- For updating the order information from your central fulfillment system (CFS) to Siebel CRM, SWIOrderUpsert Siebel inbound web service should be enabled.

For both of these services, in Siebel, you need to set the process property UTCCanonical to Y.

**For more information** about the UTCCanonical process property, see the "Billing in Siebel Communications" chapter of the *Siebel Communications Guide,*.

**For more information** about the web services, see the *Siebel Order Management Guide Addendum for Communications,* "Web Services Reference."

- For the Special Rating List Sync Flow: Configure SWISpecialRatingList Siebel outbound workflow to enqueue the Siebel messages in AIA_SPECIALRATINGJMSQ.

**For more information** about Siebel side configuration, see *Transports and Interfaces: Siebel Enterprise Application Integration v8.1, Process of Configuring JMS Messaging Between Siebel Business Applications and Oracle SOA Suite*.

**For more information** about the corresponding Oracle AIA side configuration, see the *Oracle Application Integration Architecture – Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Constructing JMS Adapter Services."

# Working with DVMs

Domain value maps (DVMs) are a standard feature of the Oracle SOA Suite and enable you to equate lookup codes and other static values across applications, for example, FOOT and FT or US and USA.

DVMs are static in nature, though administrators can add maps as needed. Transactional business processes never update DVMs—they only read from them. They are stored in XML files and cached in memory at run time.

DVM types are seeded for the order management flows, and administrators can extend the list of mapped values by adding more maps.

**Note:** The DVM names in the following table have an underscore. If you open the file in FTP mode, the underscore is replaced with *95*.

These are the DVMs for the process integration for order management:

| DVM | Description |
|---|---|
| SALESORDER_DYNAMICPRICEIND | Dynamic Pricing Indicator |

| DVM | Description |
|---|---|
| SALESORDER_FULFILLCOMPOSITIONTYPE | Fulfillment Composition Type Code |
| SALESORDER_FULFILLMENTMODECODE | Fulfillment Mode Code |
| SALESORDER_LINEFULFILLMENTMODECODE | Line Fulfillment Mode Code |
| SALESORDER_NETWORKINDICATOR | Network Indicator |
| SALESORDER_PARTIALFULFILLALLOWEDIND | Partial Fulfillment Mode Indicator |
| SALESORDER_PRIORITY | Priority |
| SALESORDER_PROCESSINGTYPECODE | Processing Type Code |
| SALESORDER_PRODUCTTYPECODE | Product Type Code |
| SALESORDER_REVISIONPERMISIBLECODE | Revision Permissible Code |
| SALESORDER_SERVICEINDICATOR. | Service Indicator |
| SALESORDER_STRTBILLSERVICEUSAGE | Start Billing Service Usage |
| SALESORDER_STATUS | Status |
| SALESORDER_TYPECODE | Type Code |
| STATE | State |
| PROVINCE | Province |
| ADDRESS_COUNTRYID | Country Code |
| CUSTOMERPARTY_TYPECODE | Account Type Code |
| ITEM_BILLINGTYPECODE | Examples of values are Subscription, Discount, Item, Special Rating, and so on. Billing Type Code. |
| SALESORDER_CHANGEDIND | Order Changed Indicator. Values are ‘True’ or ‘False’. Used to validate the OrderChangedIndicator attribute. For example, The Order Management system can set this attribute to ‘True’ if, as part of fulfillment, the order changes significantly such that Siebel CRM must make a copy of the customer order to preserve the customer intent before updating the working version of the order. |
| SALESORDER_ACTIONCODE | Sales Order Line Action Code |
| SALESORDER_REVISIONPERMISIBLECODE | Revision Permissible Code |
| SALESORDER_LINESTATUS | Order Line Status |
| DISCOUNT_METHODCODE | Discount Method Code |
| CURRENCY_CODE | Currency Code |
| PRICE_TYPE. | Price Type |

**For more information,** see *Oracle Application Integration Architecture – Foundation Pack 2.5: Integration Developer's Guide,* "Understanding Message Transformation, Enrichment, and Configuration," DVMs.

# Working with Cross-References

Cross-references map and connect the records within the application network, and they enable these applications to communicate in the same language. The integration server stores the relationship in a persistent way so that others can refer to it.

This table lists the order management cross-references:

| Cross-reference Table Name | Column Names Column Values | | | Description |
|---|---|---|---|---|
| SALESORDER_ID | COMMON | SEBL_01 | BRM_01 | Siebel Sales Order ID is cross-referenced. |
| | SalesOrderIdentification | Id | -- | |
| | | | | |
| SALESORDER_LINEID | COMMON | SEBL_01 | BRM_01 | Order Item ID from Siebel is mapped to SalesOrderLine Identification in EBM |
| | SalesOrderLine/Identification | OrderItem/Id | -- | |
| | | | | |
| INSTALLEDPRODUCT_ID | COMMON | SEBL_01 | BRM_01 | Siebel Asset Integration ID is mapped to Product/Service/Discount POID of Oracle BRM |
| | InstalledProductIdentification | AssetIntegrationId | PRODUCT/SERVICE/DISCOUNT OBJ | |
| | | | | |
| ITEM_ITEMID | COMMON | SEBL_01 | BRM_01 | Siebel Product ID is mapped to PRODUCT/DISCOUNT OBJ of the Oracle BRM. |
| | ItemIdentification | ProductId | PRODUCT/DISCOUNT POID | |
| | | | | |
| CUSTOMERPARTY_ACCOUNTID | COMMON | SEBL_01 | BRM_01 | Siebel Customer ID is mapped to Oracle BRM Account POID |
| | CustomerPartyAccountIdentification | AccountId | Account POID | |
| | | | | |
| CUSTOMERPARTY_CONTAC | COMMON | SEBL_01 | BRM_01 | Siebel Contact ID is mapped |

| Cross-reference Table Name | Column Names Column Values | | | Description |
|---|---|---|---|---|
| TID | CustomerPartyAccountContactIdentification | ContactId | Contact POID | to Oracle BRM Contact POID |
| | | | | |
| CUSTOMERPARTY_DEFAULTBALANCEGROUPID | COMMON | SEBL_01 | BRM_01 | Default balance group POID is mapped to common ID of account. |
| | CustomerPartyAccountContactIdentification | -- | Balance Group POID | |
| | | | | |
| CUSTOMERPARTY_PAYPROFILEID | COMMON | SEBL_01 | BRM_01 | Bill Profile ID from Siebel is mapped to Pay info POID of the Oracle BRM. |
| | PaymentProfileIdentification | BillingProfileId | Pay Info POID | |
| | | | | |
| CUSTOMERPARTY_BILLPROFILEID | COMMON | SEBL_01 | BRM_01 | Bill Profile ID from Siebel is mapped to Bill info POID of the Oracle BRM. |
| | BillingProfileIdentification | BillingProfileId | Bill Info POID | |
| | | | | |

# Handling Error Notifications

Based on the roles defined for the services, email notifications are sent if a service ends due to an error.

Order Fallout Management can generate trouble tickets for failed orders.

**For more information** about order fallout, see Chapter 8: Understanding the Process Integration for Order Fallout Management.

These are the error messages that are issued when order billing integration is called in billing initiation mode:

| Error Code | Error Text | Description |
|---|---|---|
| AIA_ERR_AIACOMOMPI_0001 | Date Validation Failed: Either one of the Purchase Date/Cycle Start Date/ Usage Start Date should be set to the future. | In Billing Initiation mode, the ProcessFulfillmentOrderBillingBRMComms AddSubProcess ends in an error when at least one of the billing dates (purchase, cycle start, usage start date) is not set to the future for lines with products of type subscription or discount. |
| AIA_ERR_AIACOMOMPI_0002 | Date Validation Failed: Purchase Date should be set to the future. | In Billing Initiation mode, the ProcessFulfillmentOrderBillingBRMComms AddSubProcess ends in an error when the purchase date is not set to the future for lines with products of type item. |
| AIA_ERR_AIACOMOMPI_0003 | Purchased promotion instance does not exist for a promotion that was previously purchased. A data upgrade script was not run. | ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl ends in an error if a change order is processed for data that was created using AIA for Communications 2.0/2.0.1 and the custom upgrade script was not run to create the necessary cross-reference and purchased promotion instances in BRM. |
| AIA_ERR_AIACOMOMPI_0004 | Promotion referenced on Sales Order &OrderNum, Line &LineNum for &Product has not been interfaced to billing. The promotion needs to be interfaced to billing, before interfacing the order line that references it. | ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl ends in an error if service bundle/acct level product with promotion reference is sent to billing before the corresponding promotion line. |

**For more information** about the errors caused by Siebel CRM or Oracle BRM, see the documentation for the product.

**For more information** about AIA error handling, see the *Oracle Application Integration Architecture – Foundation Pack 2.5: Core Infrastructure Components Guide,* "Understanding Error Handling and Logging."

## Describing Delivered Error Notification Roles and Users

The following roles and users are delivered as default values for issuing error notifications for the process integration for order management.

Actor roles and users:

- Role: AIAIntegrationAdmin. User: AIAIntegrationAdminUser.

The default password set for all users is welcome1.

**For more information** about the errors caused by Siebel CRM or Oracle BRM, see the documentation for the product.

**For more information** about Oracle AIA error handling, see the *Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide,* “Understanding Error Handling and Logging.”

# Viewing EBO EIMs

**For more information** about how services are mapped, see the My Oracle Support document: EBO Implementation Maps (EIMs) 881022.1.

# Configuring the Process Integration for Order Management

Configure these properties in the AIAConfigurationProperties.xml file. The file is located in <aia.home>/config/. Entries in the AIAConfigurationProperties.xml file are case-sensitive.

**Note:** Whenever the AIAConfigurationProperties.xml file is updated, the file must be reloaded for updates to be reflected in the applications or services that use the updated properties. Click the Reload button on the Configuration page in the Oracle AIA Console to perform this reload. Alternatively, you can perform the reload by rebooting the server.

**For more information,** see *Oracle Application Integration Architecture – Foundation Pack 2.5: Core Infrastructure Components Guide,* "Working with the BSR," and “Loading Oracle AIA Configuration File Updates.”

These are the settings for the UpdateSalesOrderSiebelCommsProvABCSImpl service name:

| Property Name | Value/Default Values | Description |
|---|---|---|
| Default.SystemID | SEBL_01 | URL for Siebel Instance web service for "Order spcLine spcItem spcUpdate spc-_spcComplex" web service. |

| Property Name | Value/Default Values | Description |
|---|---|---|
| Routing.SWI_spcOrder_spcUpsert.RouteToCAVS | true/false. Default = false. | Controls whether UpdateSalesOrderSiebelCommsProvABCSImpl routes messages to the CAVS or to the Siebel system. |
| Routing.SWI_spcOrder_spcUpsert.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link |
| Routing.SWI_spcOrder_spcUpsert.SEBL_01.EndpointURI | Target Endpoint URL for the Siebel upsert web service.<br>example:<br>http://${siebel.http.host}:${siebel.http.port}/eai_enu/start.swe?SWEExtSource=SecureWebService&SWEExtCmd=Execute&UserName=${siebel.eai.user}&Password=${siebel.eai.password} | Target Endpoint URL for the Siebel upsert web service. |

These are the settings for the ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl service name:

| Property Name | Value/Default Values | Description |
|---|---|---|
| Default.SystemID | SEBL_01 | Default Siebel CRM system instance code (defined in BSR). This is used only in the event that the Siebel Order message does not contain the EnterpriseServerName, for example, SEBL_01. |
| Routing.CommunicationsSalestOrderEBSV1.ProcessSalesOrderFulfillment.RouteToCAVS | true/false. Default = false. | Controls whether SalesOrderEBS routes messages to the CAVS or to the ProviderABCS. |
| Routing.CommunicationsSalestOrderEBSV1.ProcessSalesOrderFulfillment.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |

These are the settings for the ProcessFulfillmentOrderBillingBRMCommsAddSubProcess service name:

| Property Name | Value/Default Values | Description |
|---|---|---|
| Default.SystemID | BRM_01 | Default target billing system instance code (defined in BSR). This is used only if the request message does not contain the target information. |
| BRM_01.FutureTimeThreshold | 8640 | This property is used for future date validation in Billing Initiation. It is set to a default value of 8640 hours (365 days).<br>This property is billing-instance-specific and needs to be set for any instance that the order needs to be sent for billing integration.<br>Refer to the section on Two-Phase Billing for more information about how this property is used. |
| Routing.BRMSUBSCRIPTIONService.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link. |
| Routing.BRMSUBSCRIPTIONService.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMSUBSCRIPTIONService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |
| Routing.BRMCUSTService.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link. |
| Routing.BRMCUSTService.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMCUSTService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |
| Routing.BRMBASEService.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link |

| Property Name | Value/Default Values | Description |
|---|---|---|
| Routing.BRMBASEService.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMBASEService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |

These are the settings for the ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess service name:

| Property Name | Value/Default Values | Description |
|---|---|---|
| Default.SystemID | BRM_01 | Default target billing system instance code (defined in BSR). This is used only in the event that the request message does not contain the target information. |
| Routing.BRMSUBSCRIPTIONService.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link. |
| Routing.BRMSUBSCRIPTIONService.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMSUBSCRIPTIONService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |
| Routing.BRMCUSTService.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link. |
| Routing.BRMCUSTService.CAVS.EndpointURI | Simulator url for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMCUSTService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |

| Property Name | Value/Default Values | Description |
|---|---|---|
| Routing.BRMBASEService.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link. |
| Routing.BRMBASEService.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMBASEService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |

These are the settings for the ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess service name:

| Property Name | Value/Default Values | Description |
|---|---|---|
| Default.SystemID | BRM_01 | Default target billing system instance code (defined in BSR). This is used only if the request message does not contain the target information. |
| Routing.BRMSUBSCRIPTIONService.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link. |
| Routing.BRMSUBSCRIPTIONService.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMSUBSCRIPTIONService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |
| Routing.BRMCUSTService.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link. |
| Routing.BRMCUSTService.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |

| Property Name | Value/Default Values | Description |
|---|---|---|
| Routing.BRMCUSTService.BRM_01.EndpointURI | eis/BRM | End point for BRM Adapter. Example: eis/BRM |
| Routing. BRMBALService_ptt.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link. |
| Routing. BRMBALService_ptt.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing. BRMBALService_ptt.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |

These are the settings for the ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl service name:

| Property Name | Value/Default Values | Description |
|---|---|---|
| Default.SystemID | BRM_01 | Default target billing system instance code (defined in BSR). This is used only if the request message does not contain the target information. |
| Routing.BRMSUBSCRIPTIONService.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link. |
| Routing.BRMSUBSCRIPTIONService.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMSUBSCRIPTIONService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |
| Routing.BRMCUSTService.RouteToCAVS | False | CAVS simulator to be enabled or disabled for this partner link. |

| Property Name | Value/Default Values | Description |
|---|---|---|
| Routing.BRMCUSTService.CAVS.EndpointURI | Simulator url for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMCUSTService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |

These are the settings for the ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess service name:

| Property Name | Value/Default Values | Description |
|---|---|---|
| Default.SystemID | BRM_01 | Default target billing system instance code (defined in BSR). This is used if the request message does not contain the target information. |
| Routing.BRMSUBSCRIPTIONService.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link. |
| Routing.BRMSUBSCRIPTIONService.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMSUBSCRIPTIONService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |
| Routing.BRMCUSTService.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link. |
| Routing.BRMCUSTService.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMCUSTService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |
| Routing.BRMBASEService.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link. |

| Property Name | Value/Default Values | Description |
|---|---|---|
| Routing.BRMBASEService.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMBASEService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |

These are the settings for the ProcessFulfillmentOrderBillingBRMCommsUpdateSubProcess service name:

| Property Name | Value/Default Values | Description |
|---|---|---|
| Default.SystemID | BRM_01 | Default target billing system instance code (defined in BSR). This is used if the request message does not contain the target information. |
| Routing.BRMSUBSCRIPTIONService.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link. |
| Routing.BRMSUBSCRIPTIONService.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMSUBSCRIPTIONService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |
| Routing.BRMCUSTService.RouteToCAVS | False | CAVS simulator to be enabled or disabled for this partner link. |
| Routing.BRMCUSTService.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMCUSTService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |

| Property Name | Value/Default Values | Description |
|---|---|---|
| Routing.BRMBASEService.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link. |
| Routing.BRMBASEService.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMBASEService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |

These are the settings for the ProcessInstalledProductSpecialRatingSetListSiebelCommsReqABCSImpl service name:

| Property Name | Value/Default Values | Description |
|---|---|---|
| Default.SystemID | SEBL_01 | Default Siebel CRM system instance code (defined in BSR). This is used only if the Siebel ABM does not contain the EnterpriseServerName, for example, SEBL_01. |
| Routing.InstalledProductEBSV1.ProcessInstalledProductSpecialRatingSetList.RouteToCAVS | true/false. Default = false. | Controls whether SalesOrderEBS should route messages to the CAVS or to the ProviderABCS. |
| Routing.InstalledProductEBSV1.ProcessInstalledProductSpecialRatingSetList.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |

These are the settings for the ProcessInstalledProductSpecialRatingSetListBRMCommsProvABCSImpl service name:

| Property Name | Value/Default Values | Description |
|---|---|---|
| Default.SystemID | BRM_01 | Default target billing system instance code (defined in BSR). This is used only if the request message does not contain the target information. |
| Routing.BRMCUSTService.RouteToCAVS | false | CAVS simulator to be enabled or disabled for this partner link. |

| Property Name | Value/Default Values | Description |
|---|---|---|
| Routing.BRMCUSTService.CAVS.EndpointURI | Simulator URL for the particular CAVS simulator example: http://adc60118fems.us.oracle.com:7813/AIAValidationSystemServlet/syncresponsesimulator?simid=1051 | CAVS simulator end point URI for this partner link. |
| Routing.BRMCUSTService.BRM_01.EndpointURI | eis/BRM | End point for Oracle BRM Adapter. Example: eis/BRM |

# Part 3: Implementing the Process Integration for Customer Management

This part includes the following chapters:

Chapter 6: Understanding the Process Integration for Customer Management

Chapter 7: Configuring the Process Integration for Customer Management

# Chapter 6: Understanding the Process Integration for Customer Management

This chapter provides an overview of the process integration for customer management and discusses:

- Solution assumptions and constraints
- Data requirements
- Create/Sync account integration flow
- Update customer accounts integration flow
- Oracle Communications Billing and Revenue Management (Oracle BRM) interfaces
- Siebel Customer Relationship Management (Siebel CRM) interfaces
- Industry Application Integration Architecture AIA components
- Integration services

## Overview

The process integration for customer management enables the synchronization of customer information between Siebel CRM and Oracle BRM. Customers are created in Siebel CRM and sent to Oracle BRM. Siebel CRM is the customer master. Customer data updated in Siebel CRM is synchronized to Oracle BRM through the customer management process integration and is a one-way synchronization process.

The process integration for customer management provides two integration flows:

- The create/sync customer account integration flow, which interfaces customers to Oracle BRM (run during the Order Management processing flow).
- The update customer account integration flow, which updates account information (such as address, name, contact, and status) from Siebel CRM to Oracle BRM.

## Solution Assumptions and Constraints

Here are the solution assumptions for the process integration for customer management:

1. Siebel CRM is the customer master and manages all aspects of the lifecycle from creation to updates for a customer. The customer management process integration is a uni-directional flow from Siebel CRM to Oracle BRM.
2. Initial loading of customer data is not supported for this release.
3. An order line can have only one bill-to account.

4. If the order line items contain a service account that is different from the bill-to account, then the service account needs to be synchronized with Oracle BRM. Siebel CRM billing accounts are propagated as paying accounts in Oracle BRM, while Siebel CRM service accounts are propagated as non-paying sub-ordinate accounts in Oracle BRM.
5. Customer accounts and billing profiles are first synchronized to Oracle BRM during order processing, and not before.
6. Once synchronized to a particular billing system, a customer account is kept in sync by means of real-time integration flows.

   The Customer Account Sync integration that occurs during order processing can assume that if an account has already been created in Oracle BRM, it is current and up-to-date.

7. The Customer Account Sync process during order processing synchronizes accounts to one billing system/instance (Oracle BRM) at a time. The Order Management application can synchronize the same customer to additional billing systems/instances by calling the Customer Account Sync service multiple times.

**For more information** see Creating Customer Data in Billing and Appendix E: Configuring Multiple Instances of Oracle BRM.

8. The Siebel CRM account hierarchy is not synchronized to Oracle BRM. Instead, the billing account and service account relationship on a Siebel order line is sent to Oracle BRM as a parent account and child account, respectively. Oracle BRM supports a single parent for a child account.

# Data Requirements

The process integration for customer management requires the following data to successfully create customer data in Oracle BRM:

- Accounts must be of type Residential or Business and the account class must be Customer, Service, or Billing.
- In Siebel, accounts can have any number of contacts or addresses associated with them, but account creation in Oracle BRM requires:
  - The primary contact (must be explicitly set) and address for the account.
  - The contact and address that is associated with the billing profile that is used in the order.
  - For an account's primary address, the city, state, country, and zip code.
  - For an account's primary contact, the last name.
  - For a bill profile, all bill profiles that are synchronized for an account and its related parent and child accounts must have the same value for Bill Frequency.
  - For a bill profile address, the city, state, and zip code.
  - For a credit card bill profile, the credit card number, expiration month and year, and cardholder name are required. Card verification value (CVV) number is optional.

- For an automatic debit bill profile, the bank routing number and account number are required.

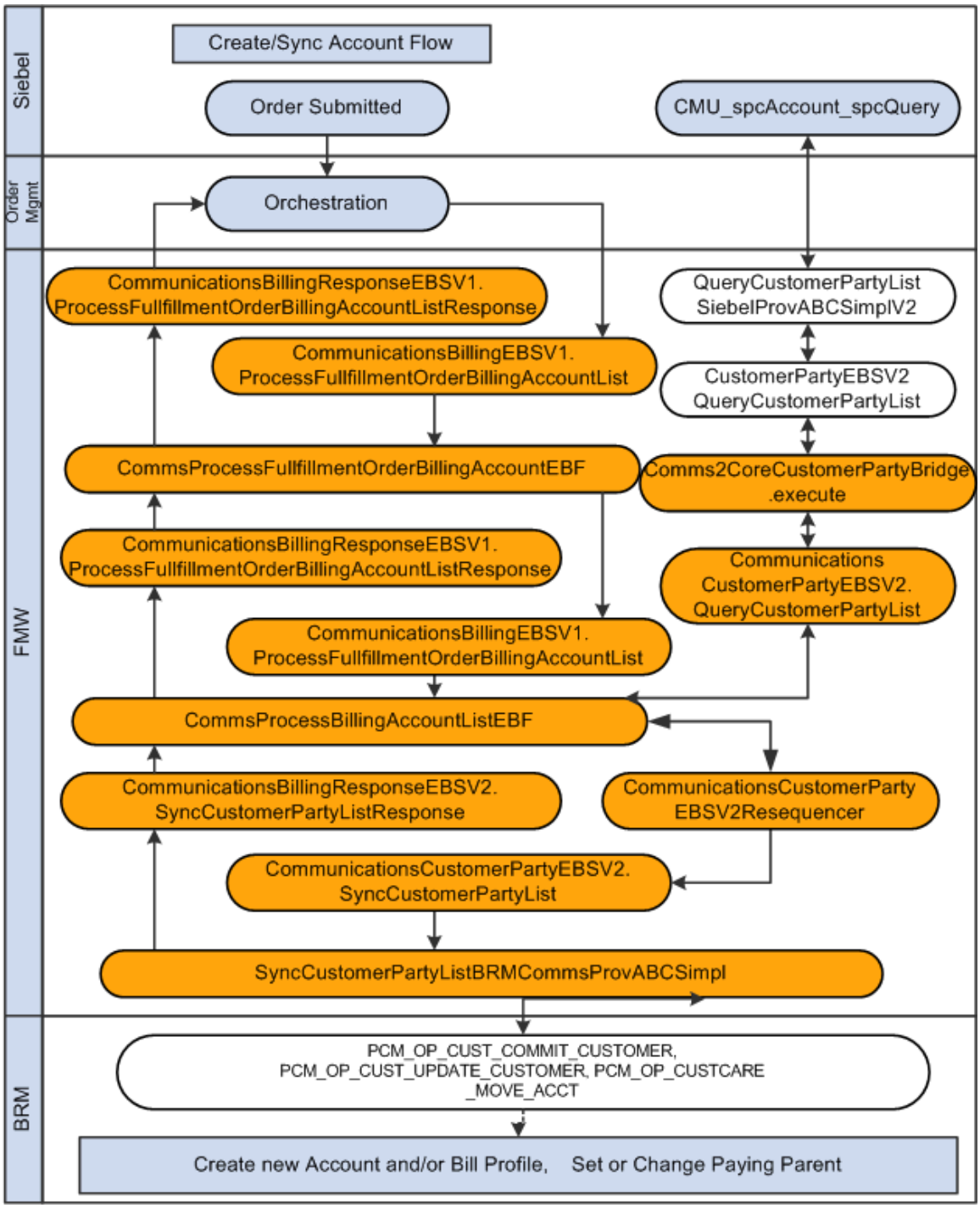
# Create/Sync Customer Account Integration Flow

Account information is captured at the beginning of the order process. When a customer places an order, the first step of the process is to determine whether the customer is new or existing. If this is an existing customer, the customer record can be found and selected, and the customer order details are captured. If this is a new customer, a new account is created.

The billing preferences (bill medium, bill frequency, payment type, billing type, billing contact, bill cycle data, and so on) are also captured. After the account information is captured, the order details are captured. The order is submitted to the order management system for processing. Customer data is created in billing as part of the Order Fulfillment flow.

**For more information** about the creating customer data in billing, see Creating Customer Data in Billing.

Subsequently, customers can call in to request changes to their contact information, address, and so on. These changes as well as updates to other attributes are supported through the Update Customer Account integration flow. This diagram illustrates the overall flow for the create/synch customer account integration flow.



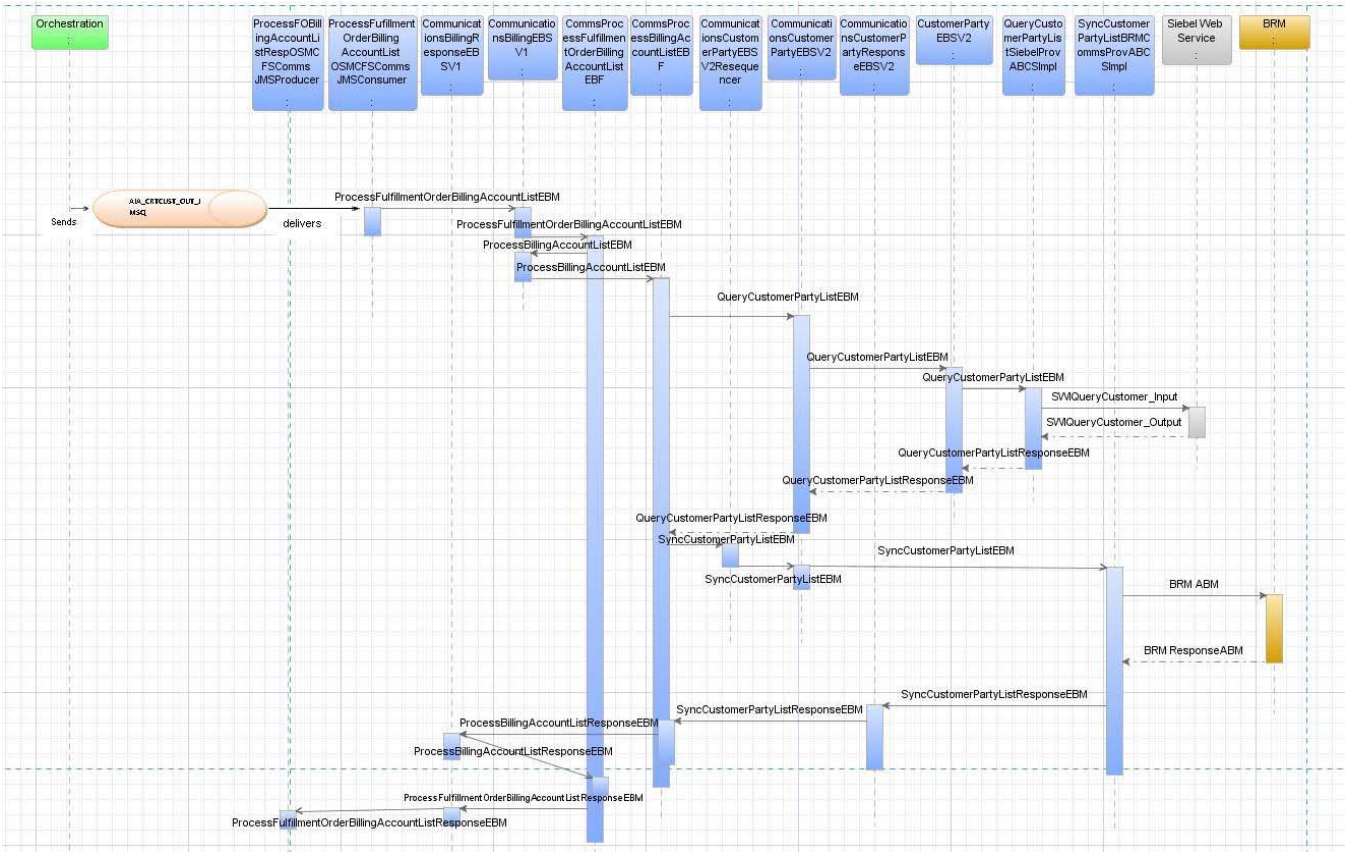

## Overall flow for the create/sync customer account integration flow

The following table provides information on Siebel attributes mapped to Oracle BRM as part of the create/sync account integration flow.

| Siebel Entity/Attributes (as labeled in Siebel UI | BRM Entity/Attributes (as labeled in Oracle BRM Customer Center) | Comments |
|---|---|---|
| Account | Account | -- |
| -- | Account Number | Integration sets this to the *Common ID*. |
| Account Type | Business Type | Only Siebel Account Type of *Residential* or *Business* is supported.<br>Uses the CUSTOMERPARTY_TYPECODE DVM. |
| Name | Company Name | Only set for the Account Type of *Business*. |
| Currency | Currency | Uses the CURRENCY_CODE DVM. |
| -- | -- | Notes:<br>Integration does not explicitly set status when creating the customer account in Oracle BRM. Oracle BRM defaults the status to *Active*.<br>The integration creates a two-level hierarchy in Oracle BRM with a paying parent and a subordinate service account when the billing account and the service account on the order line are not the same.<br>For more information, see Creating Customer Data in Billing. |
| Contact | -- | The integration only synchronizes the primary contact that is tied to the Account in Siebel CRM to Oracle BRM. |
| Mr/Mrs | Salutation | Uses the CONTACT_SALUTATION DVM. |
| First Name | First Name | -- |
| Last Name | Last Name | -- |
| Phone | Phone Number | The integration maps different Siebel phone number types (home, work, fax, mobile) to Oracle BRM Phone Type and Number using the PHONENUMBER_TYPE DVM.<br>For more information about phone number formats, see *Oracle Communications Billing and Revenue Management Concepts*, "Using BRM with Oracle Application Integration Architecture", Validating Customer Contact Information. |
| Job Title | Job Title | -- |
| Email | Email | -- |
| Address | | The integration only synchronizes the primary address that is tied to the Account in Siebel to Oracle BRM. |
| Address | Address | In addition to *Address*, the following fields are also mapped:<br>*City, State, Postal Code, Country*.<br>Uses the following DVMs:<br>ADDRESS_COUNTRYID, ADDRESS_COUNTRYSUBDIVID, PROVINCE, STATE. |
| Billing Profile | BillInfo | -- |
| Name | Name | -- |

| Siebel Entity/Attributes (as labeled in Siebel UI | BRM Entity/Attributes (as labeled in Oracle BRM Customer Center) | Comments |
|---|---|---|
| Frequency | Billing Frequency in Months | Uses the CUSTOMERPARTY_BILLPROFILE_FREQUENCYCODE DVM. |
| -- | Currency | Integration passes account-level currency.<br>Uses the CURRENCY_CODE DVM. |
| Billing Schedule | Billing Day of Month | If Billing Schedule is not set in and sent from Siebel CRM, then Oracle BRM defaults the Billing Day of Month.<br>For more information about the billing schedule, see *Oracle Communications Billing and Revenue Management Configuring and Running Billing Guide*, "Setting Business Policies for Billing. |
| -- | PayInfo | -- |
| Payment Method | Payment Method | Only *Bill Me*, *Credit Card*, or *Auto-Debit* is supported.<br>Uses the CUSTOMERPARTY_PAYPROFILE_PAYMETHODCODE DVM. |
| Contact Last Name<br>First Name | Name | When the payment method is *Bill Me*, it is the Contact Name on the Siebel Billing profile that is mapped to Oracle BRM PayInfo Contact Name.<br>When the payment method is *Credit Card* or *Auto-Debit*, either the Credit Card owner name or the Account name that is mapped to Oracle BRM PayInfo Contact Name. |
| Bill Media | Delivery Preference | Applicable only when the payment method is *Bill Me*.<br>Uses the CUSTOMERPARTY_PAYPROFILE_DELIVERYPREF DVM. |
| Email Bill To | Email Address | Applicable only when the payment method is *Bill Me*. |
| Address | Address | In addition to *Address*, the following fields are also mapped:<br>*City, State, Postal Code, Country*.<br>Uses the following DVMs:<br>ADDRESS_COUNTRYID, ADDRESS_COUNTRYSUBDIVID, PROVINCE, STATE |
| Credit Card # | Credit Card Number | Applicable only when the payment method is *Credit Card*. |
| Expiration Month & Year | Credit Card Exp | Applicable only when the payment method is *Credit Card*. |
| Security Code | Security ID | Applicable only when the payment method is *Credit Card*. |
| Account # | Debit Num | Applicable only when the payment method is *Auto-Debit.* |
| Bank Routing # | Bank No | Applicable only when the payment method is *Auto-Debit.* |
| Bank Account Type | Type | Applicable only when the payment method is *Auto-Debit*.<br>Uses the CUSTOMERPARTY_PAYPROFILE_BANKACCOUNTTYPE DVM. |

This sequence diagram illustrates the create/sync account integration flow:




## Create/sync account flow sequence diagram

When this process is initiated, the following events occur:

1. In Siebel CRM, a user navigates to the Sales Order screen, creates a sales order for the account, and clicks the Submit Billing Order button to submit the order.

   This triggers the Order flow, which in turn hands off the order to the Order Management engine.

**For more information**, see Chapter 4: Understanding the Process Integration for Order Management.

2. The Order Management/orchestration engine instantiates the customer flow.

   It calls the CommsProcessFulfillmentOrderBillingAccountListEBF, which takes the Order enterprise business message (EBM) and extracts the relevant customer data to create a Customer EBM (ProcessBillingAccountListEBM). The enterprise business flow (EBF) then calls the CommsProcessBillingAccountListEBF for further customer processing.

3. The CommsProcessBillingAccountListEBF then invokes the CommunicationsCustomerPartyEBSV2 with the operation QueryCustomerPartyList and the message QueryCustomerPartyListEBM to fetch the account data from the Siebel system before creating the account in Oracle BRM.

4. CommunicationsCustomerPartyEBSV2 with operation QueryCustomerPartyList routes the QueryCustomerPartyListEBM to the core CustomerPartyEBSV2.QueryCustomerPartyList port. It calls QueryCustomerPartyListSiebelProvABCSImplV2.

   This service transforms the QueryCustomerPartyListEBM to the Siebel-specific application business message (ABM) and invokes the SWI_Customer_Party_Service.

5. The Query web service queries the Siebel database and fetches the account data, which is then sent back as a response by means of the same Siebel ABM.
6. The response message is then transformed to the response EBM, QueryCustomerPartyListResponseEBM, by the Query Provider application business connector service (ABCS).

   This response is sent back to the CommsProcessBillingAccountListEBF service through the CustomerPartyEBSV2 and CommunicationsCustomerPartyEBSV2.

7. The CommsProcessBillingAccountListEBF then invokes the CommunicationsCustomerPartyEBSV2Resequencer. This resequencer service processes messages, grouping them by the account ID, and calls the CommunicationsCustomerPartyEBSV2.SyncCustomerPartyList operation, which calls the SyncCustomerPartyListBRMCommsProvABCSImpl.

   This service invokes the PCM_OP_CUST_COMMIT_CUSTOMER opcode if it is a new account to be synced. For an already-existing account, the PCM_OP_CUST_UPDATE_CUSTOMER or PCM_OP_CUSTCARE_MOVE_ACCT opcode is called, as required.

8. If the accounts are successfully synchronized or updates to an existing account are successfully done, the appropriate response is sent back to the CommsProcessFulfillmentOrderBillingAccountListEBF in an asynchronous, delayed-response fashion.

## Defining Transaction Boundaries and Recovery Details

For more information about defining transaction boundaries for this flow, see Defining Transaction Boundaries and Recovery Details.

# Update Customer Account Integration Flow

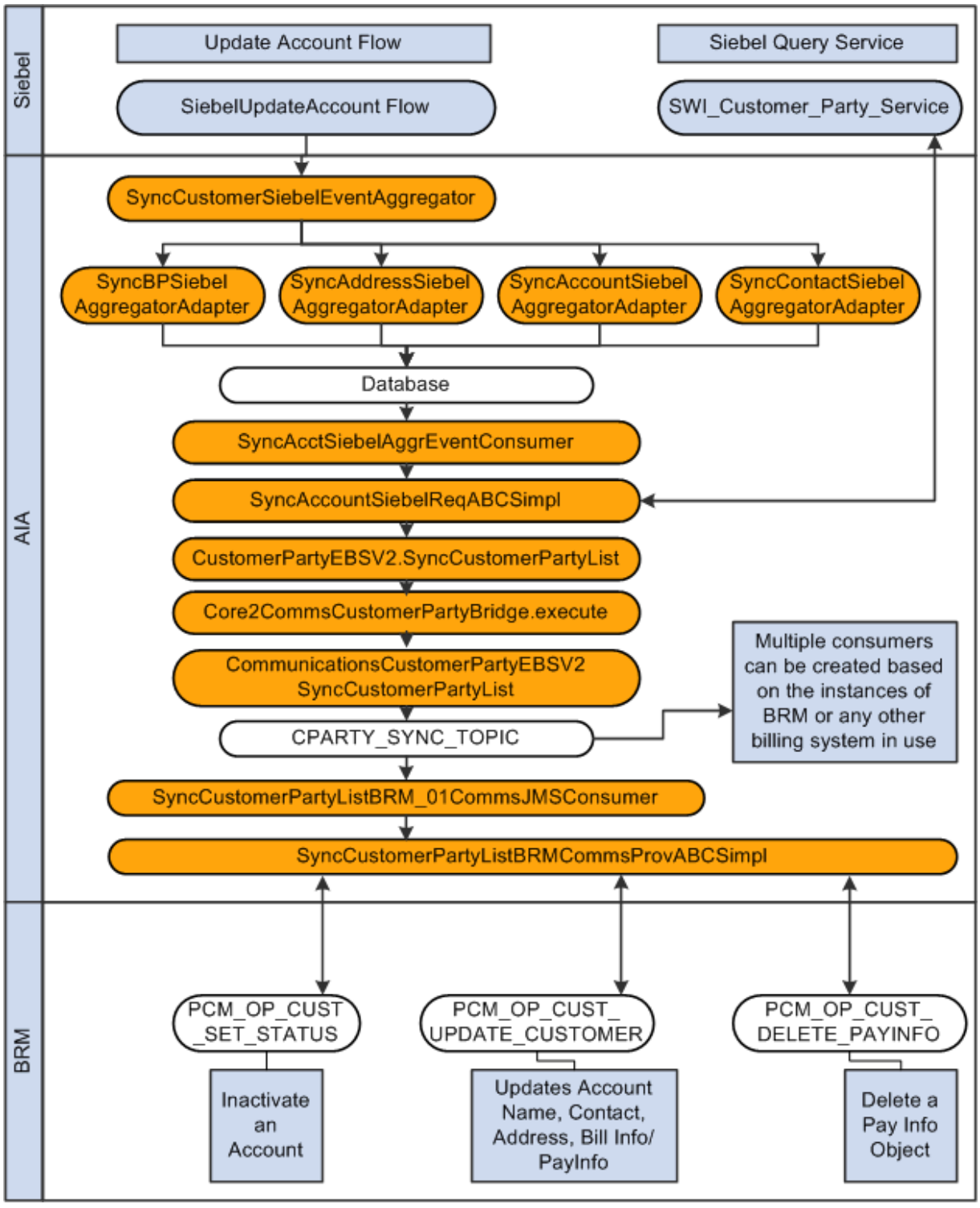
Customers can call in to make changes to their account information. The customer service representative (CSR) uses Siebel CRM as the front-end application to capture these customer data updates. The Customer Management process integration synchronizes these customer updates to Oracle BRM through the update customer account integration flow.

**Note:** Updates are synchronized to Oracle BRM only for accounts that have already been created through the order fulfillment flow,

Over time customer attributes such as name, address, contact information, billing, and payment information can change. As and when customer data is changed in Siebel CRM, the process integration ensures that these changes are synchronized to Oracle BRM in real time, thereby ensuring the customer data is both consistent and current between both the applications.

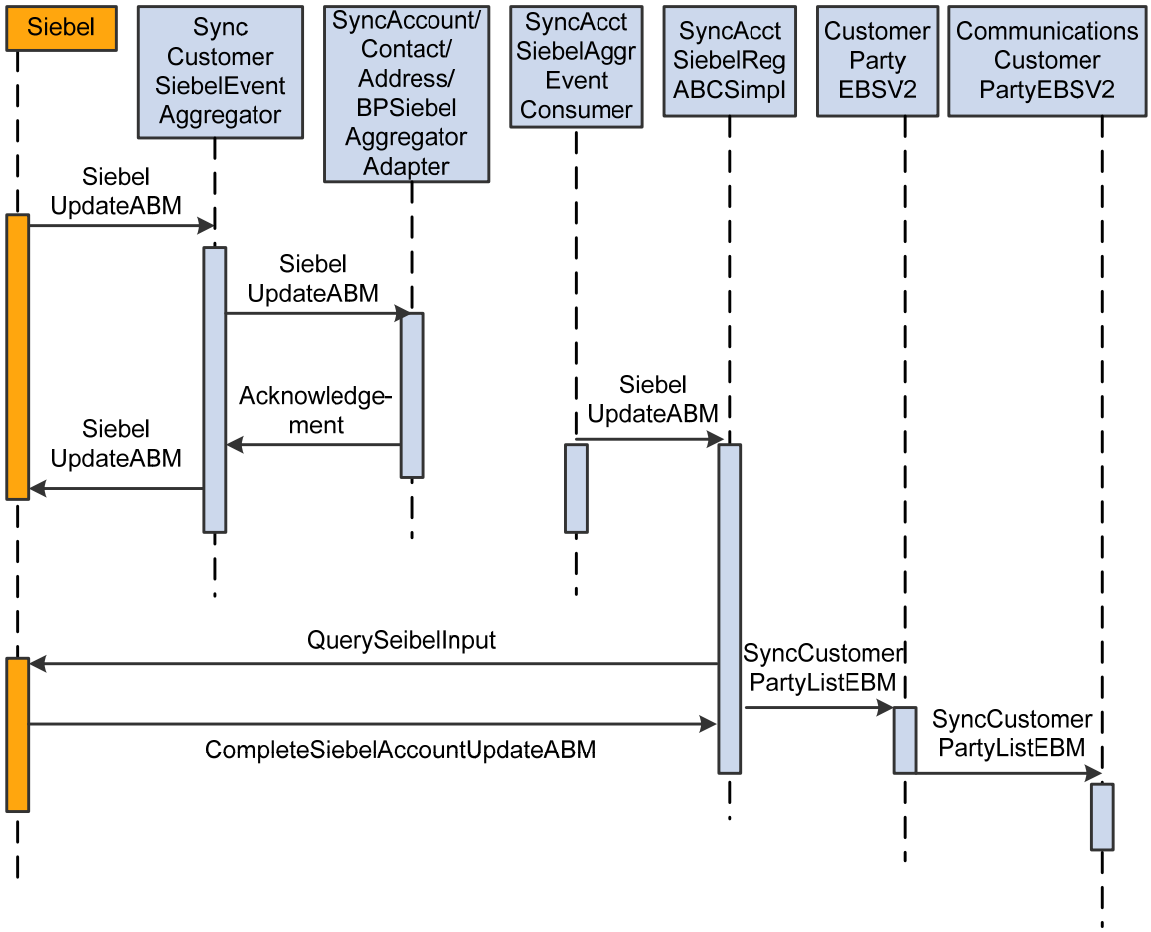
A provision exists for optionally synchronizing account status updates from Siebel CRM to Oracle BRM. For more information about the synchronization of the account status, see Account Status Synchronization Methodology.

This diagram illustrates the overall flow for the update customer account integration flow.

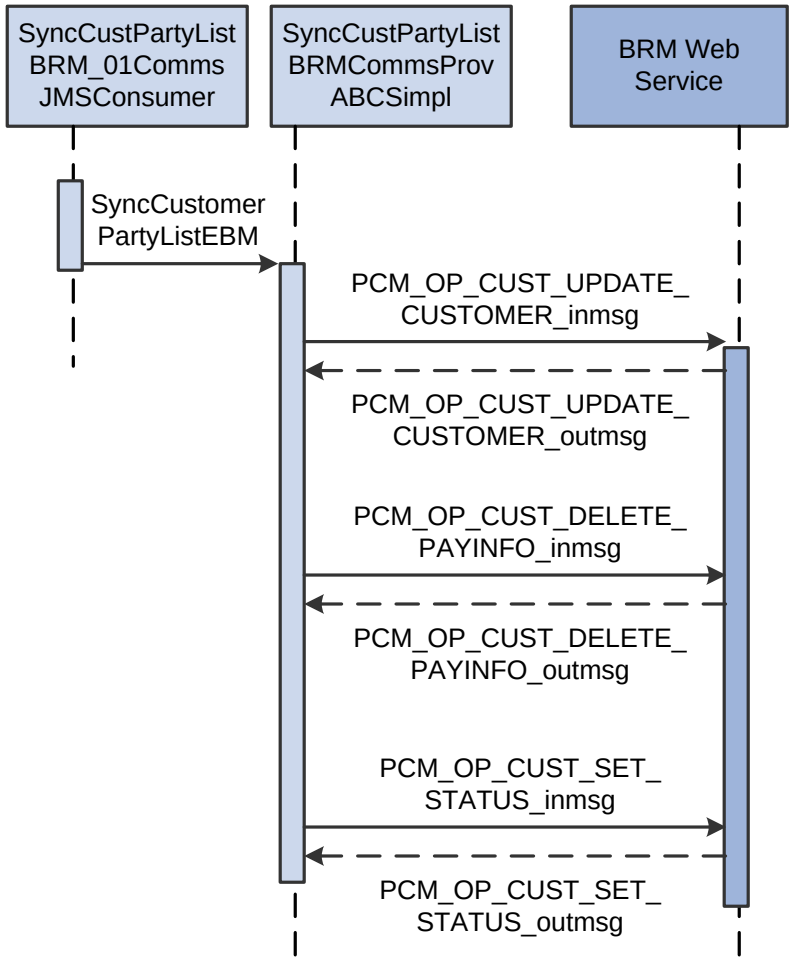



Overall flow for the update customer account integration flow

This sequence diagram illustrates the update customer accounts flow:




Update customer account flow sequence diagram (1 of 2)




## Update Customer account flow sequence diagram (2 of 2)

When this process is initiated, the following events occur:

1. In Siebel CRM, a user navigates to the Accounts screen, queries an account, and updates an account attribute (for example, address, contact, or the billing profile).

   This causes Siebel CRM to invoke the SyncCustomerSiebelEventAggregator, with the SiebelUpdateABM message containing the detail of the account that has been updated. Depending on the type of update, one of four kinds of Siebel messages can be generated: ListOfSWICustomerIO, ListOfSWIBillingProfileIO, ListOfSWIContactIO, or ListOfSWIAddressIO.

2. The SyncCustomerSiebelEventAggregator then calls a database adapter that runs a pl/sql script that extracts and stores the relevant IDs (for example, account, contact, or billing profile) in a database table AIA_AGGREGATED_ENTITIES.
3. The IDs in the database table are stored in the same manner as the hierarchy of IDs is maintained (for example, BillingProfileID will always be the child of some account ID).

   The account ID, along with its entire child IDs, is picked up from the database table by the SyncAcctSiebelEventAggrConsumer process. This consumer is sequencing-enabled to make sure that the updates for the same customer are sent in the appropriate sequence.

4. The Consumer process then calls the SyncAccountSiebelReqABCSImpl process.

This process takes all the IDs, constructs a Siebel Query Input ABM, and calls the Siebel Query web service to get the entire account data from Siebel. After getting the data, the Siebel Query Input ABM is transformed into the SyncCustomerPartyListEBM, and the CustomerPartyEBSV2.SyncCustomerPartyList operation is called.

5. The CustomerPartyEBSV2.SyncCustomerPartyList operation calls the CommunicationsCustomerPartyEBSV2.SyncCustomerPartyList operation.
6. The operation CommunicationsCustomerPartyEBSV2.SyncCustomerPartyList puts the message in the topic named CPARTY_SYNC_TOPIC.
7. Depending on the instances of Oracle BRM or any other billing system, consumers can be defined that have subscribed to the CPARTY_SYNC_TOPIC topic.

   One such consumer for the default implementation is available, named SyncCustomerPartyListBRM_01CommsJMSConsumer, which listens to the topic for messages, picks up the arriving message, and passes it on to the process SyncCustomerPartyListBRMCommsProvABCSImpl after duly checking whether the message should go to the ensuing provider ABCS, and accordingly stamping the target ID.

8. The SyncCustomerPartyListBRMCommsProvABCSImpl process then calls the PCM_OP_CUST_UPDATE_CUSTOMER, PCM_OP_CUST_DELETE_PAYINFO, or PCM_OP_CUST_SET_STATUS opcode as required to synchronize the updated data to Oracle BRM.

## Account Status Synchronization Methodology

The account status synchronization feature enables propagation of account status changes from Siebel CRM to Oracle BRM.

As delivered, the account status propagation to Oracle BRM is disabled. If implementers choose to use this feature, they must explicitly enable it by changing a configuration setting in the AIAConfiguration.xml file.

**For more information** about this configuration setting, see the EnableAccountStatusSync property in "Chapter 7: Configuring the Process Integration for Customer Management", Configuring the Process Integration for Customer Management.

The account status synchronization feature is designed as part of the collections process integration and should ideally be used in conjunction with it and not as an independent or standalone feature.

**For more information** about collections, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Agent Assisted Billing Care Implementation Guide*.

To support collections, the integration synchronizes collection actions generated by Oracle BRM as credit alerts in Siebel CRM. Various actions such as notifying the customer regarding unpaid dues or suspending or canceling services due to delinquency are delegated to Siebel.

Siebel can be extended to automate the generation of change orders for suspending or canceling services based on the generated credit alerts. Alternatively, the Siebel collection agent can manually submit change orders for suspending or canceling services. Either of these approaches ensures that changes in the service asset state are communicated correctly to Oracle BRM and both the applications are in sync with respect to the service state.

As part of the collections lifecycle, if the customer continues to be delinquent and needs to be written off and his account inactivated, this feature (if enabled) ensures that the account status change in Siebel CRM is propagated to Oracle BRM.

We recommend that the account in Siebel CRM be inactivated only after all the services (and account-level subscription products) have been canceled. This is because inactivating an account in Siebel CRM that has active services propagates that account status change to Oracle BRM resulting in the cancellation of services in Oracle BRM. This is because Oracle BRM cascades the status change from the account to all its bill-infos and services and products. An important practice is to inactivate the account in Siebel CRM only after all the services (and account-level subscription products) have been canceled (the cancellation orders fulfilled and asseted).

As delivered, Siebel does not have logic to restrict changes to account status. Therefore, it is recommended that the ability to inactivate an account be restricted to authorized users and roles in Siebel CRM. This is because inadvertently inactivating accounts with active services can result (when the account status propagation is enabled) in those services being canceled in Oracle BRM.

# Oracle BRM Interfaces

| API / Opcode | Description | Used by |
|---|---|---|
| PCM_OP_CUST_COMMIT_CUSTOMER | Create a new account with one or more bill-infos and pay-infos in Oracle BRM. | Oracle BRM Sync Customer Provider application business connector service (ABCS) as part of the Create/Update Customer Account flow. |
| PCM_OP_CUST_UPDATE_CUSTOMER | Update account information (name, address, phone), contact information, and billing and pay information. | Oracle BRM Sync Customer Provider ABCS as part of the Update Customer Account flow. |
| PCM_OP_CUST_DELETE_PAYINFO | Delete a payinfo object from an account. | Oracle BRM Sync Customer Provider ABCS, as part of the Update Customer Account flow. |
| PCM_OP_CUSTCARE_MOVE_ACCT | Move an account to a new parent account. | Oracle BRM Sync Customer Provider ABCS as part of the Update Customer Account flow. |
| PCM_OP_CUST_SET_STATUS | Used to modify the account status in Oracle BRM. | Oracle BRM Sync Customer Provider ABCS as part of the Update Customer Account flow. |

**For more information**, see *Oracle Communications Billing and Revenue Management (BRM) Documentation*, “BRM Documentation,” Reference, API reference, PCM opcode libraries.

# Siebel CRM Interfaces

### Siebel CRM Web Service Interfaces

This is the Siebel CRM web service interface:

| Web Service | Description | Used by |
|---|---|---|
| Query Account-<br>(SWICustomerParty) | Retrieves account, bill profile, contact, and address data from Siebel CRM. | Used by the Siebel Query Account Provider ABCS as part of creating and adding a new billing profile to an existing customer.<br>Also used by the Siebel Sync Account Requester ABCS to retrieve the most current account data from Siebel. |

**For more information** about web services, see the *Siebel Order Management Guide Addendum for Communications*, "Web Services Reference."

### Siebel CRM Workflow Event Interfaces

These are the Siebel CRM workflow event interfaces:

| Event | Description | Consumed by |
|---|---|---|
| SWI Account Updated | This workflow event is started when an account is updated in Siebel CRM. | This event message is consumed by the SyncCustomerSiebelEventAggregator.aggregateaccount event service, which extracts all the relevant IDs from the input payload and stores them in a database table (AIA_AGGREGATED_ENTITIES). |
| SWI Bill Profile Updated | This workflow event is started when a bill profile is updated in Siebel CRM. | The event message is consumed by the SyncCustomerSiebelEventAggregator.aggregatebpevent service, which extracts all the relevant IDs from the input payload and stores them in a database table (AIA_AGGREGATED_ENTITIES). |
| SWI Contact Updated | This workflow event is started when a contact is updated in Siebel CRM. | The event message is consumed by the SyncCustomerSiebelEventAggregator.aggregatecontact event service, which extracts all the relevant IDs from the input payload and stores them in a database table (AIA_AGGREGATED_ENTITIES). |
| SWI Address Updated | This workflow event is started when an address is updated in Siebel CRM. | The event message is consumed by the SyncCustomerSiebelEventAggregator.aggregateaddress event service, which extracts all the relevant IDs from the input payload and stores them in a database table (AIA_AGGREGATED_ENTITIES). |

**For more information**, *see Siebel Order Management Guide Addendum for Communications*, "Workflows for Employee Asset-Based Ordering."

# Industry AIA Components

This is the list of the enterprise business objects (EBOs) and enterprise business messages (EBMs) used by the process integration for customer management:

- CustomerPartyEBO
- QueryCustomerPartyListEBM
- QueryCustomerPartyListResponseEBM
- SyncCustomerPartyListEBM
- SyncCustomerPartyListResponseEBM
- ProcessBillingAccountListEBM
- ProcessBillingAccountListResponseEBM
- FulfillmentOrderEBO
- ProcessFulfillmentOrderBillingAccountListEBM
- ProcessFulfillmentOrderBillingAccountListResponseEBM

These industry EBO and EBM XML schema (XSD) files are located here:

http://<host name>:<portnumber>/AIAComponents/EnterpriseObjectLibrary/Industry/Communications/EBO/CustomerParty/V2/

http:// <host name>:<portnumber>/AIAComponents/EnterpriseObjectLibrary/Core/EBO/CustomerParty/V2/

These industry enterprise business service web service description language (EBS WSDL) files are located here:

http:// <host name>:<portnumber>/AIAComponents/EnterpriseBusinessServiceLibrary/Industry/Communications/EBO/CustomerParty/V2/CommunicationsCustomerPartyEBSV2.wsdl

http:// <host name>:<portnumber>/AIAComponents/EnterpriseBusinessServiceLibrary/Core/EBO/CustomerParty/V2/CommunicationsCustomerPartyEBSV2.wsdl

For detailed documentation for individual EBOs, click the View EBO Documentation link on the Integration Scenario Summary page in the Oracle AIA Console. You can also use the Integration Scenario Summary page to search for and view integration scenarios that use a particular EBO or EBS.

**For more information,** see *Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide*, "Using the BSR UI to View Integration Scenarios," Viewing Integration Scenarios.

EBOs can be extended, for instance, to add new data elements. These extensions are protected and will remain intact after a patch or an upgrade.

**For more information,** see *Oracle Application Integration Architecture - Foundation Pack Integration Developer's Guide*, "Extensibility for Oracle AIA Artifacts," Extending EBOs.

**For more information** about the changes made for each EBO and EBM, see the appendix in the My Oracle Support document number 988374.1.

# Integration Services

These services are delivered with the customer management integration flow:

- CommunicationsCustomerPartyEBSV2
- CommunicationsCustomerPartyEBSV2Resequencer
- CommsProcessFulfillmentOrderBillingAccountListEBF
- CommsProcessBillingAccountListEBF
- SyncCustomerSiebelEventAggregator
- SyncAccountSiebelAggregatorAdapter
- SyncContactSiebelAggregatorAdapter
- SyncAddressSiebelAggregatorAdapter
- SyncBPSiebelAggregatorAdapter
- SyncAcctSiebelAggrEventConsumer
- SyncAccountSiebelReqABCSImpl
- CustomerPartyEBSV2
- QueryCustomerPartyListSiebelProvABCSImplV2
- CommunicationsCustomerPartyResponseEBSV2
- SyncCustomerPartyListBRMCommsProvABCSImpl
- SyncCustomerPartyListBRM_01CommsJMSConsumer

You can use the Integration Scenario Summary page in the Oracle AIA Console to search for and view integration scenarios that use a particular ABCS.

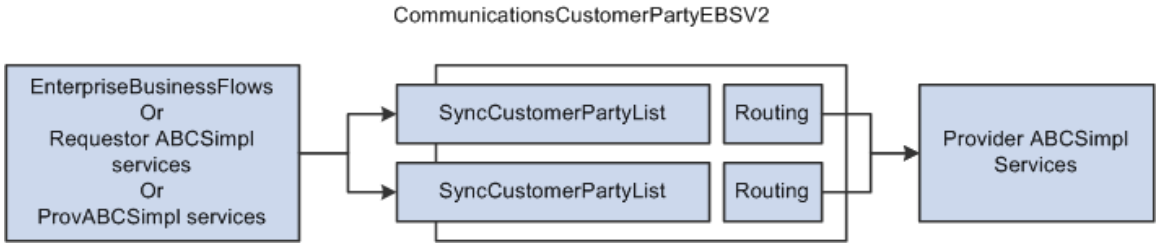
**For more information,** see *Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide*, "Using the BSR UI to View Integration Scenarios."

## CommunicationsCustomerPartyEBSV2

The CommunicationsCustomerPartyEBSV2 exposes all of the enterprise operations that can be performed with a CustomerParty enterprise object. The integration uses the following operations provided by the CommunicationsCustomerPartyEBSV2:

- QueryCustomerPartyList.
- SyncCustomerPartyList.

This diagram illustrates the relationship of CommunicationsCustomerPartyEBSV2 with the other services in the integration flow:




### CommunicationsCustomerPartyEBSV2

In one of the routing rules for the SyncCustomerPartyList operation of the CommunicationsCustomerPartyEBSV2, the process checks whether the incoming message has a target system identifier. If the target system identifier is not present, then the delivered rule assumes multiple Oracle BRM systems and routes the incoming requests to a Java message service (JMS) producer service SyncCustomerPartyListBRMCommsJMSProducer::Produce_Message.

If the implementing customer doesn't have multiple Oracle BRM systems and operates only on a single Oracle BRM system, then they can change the routing rule to route incoming requests to the SyncCustomerPartyListBRMCommsProvABCSImpl_1_0::SyncCustomerPartyList directly. Additionally, customers must apply a transformation before routing to stamp the target system identifier in the EBM. The transformation file name is esb:///ESB_Projects/Customer_CommunicationsCustomerPartyEBSV2/AddTargetID_BRM01.xsl.

**For more information** about this EBS, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs."

## CommunicationsCustomerPartyEBSV2Resequencer

The CommunicationsCustomerPartyEBSV2Resequencer enterprise business service sequences the account message from CommsProcessBillingAccountListEBF. The messages are grouped by account ID. This process receives the customer EBM and passes it to CommunicationsCustomerPartyEBSV2 enterprise business service for routing to Oracle BRM.

**For more information** about this EBS, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs."

**For more information** about the resequencer, see Appendix G: Using the Resequencer Feature of ESB.

## CommsProcessFulfillmentOrderBillingAccountListEBF

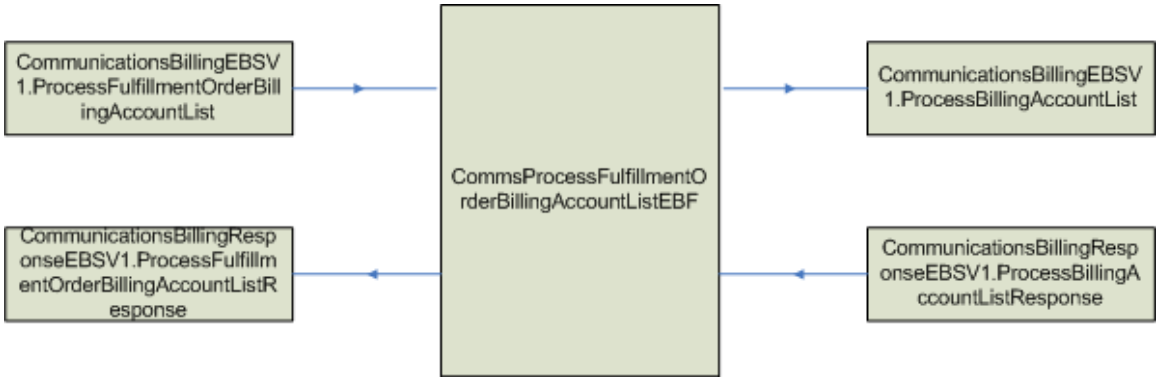
This enterprise business flow (EBF) extracts the Customer Data from OrderEBM. The process loops through every order line and extracts any customer account or billing profile that it encounters.

This service has two operations. One accepts the ProcessFulfillmentOrderBillingAccountListEBM and is used by the process to order data. The other is used by the process to send the response back to the calling process (using the ProcessFulfillmentOrderBillingAccountListEBM).

The transformations include:

- ProcessFulfillmentOrderBillingAccountList to ResponseEBM.xsl
- ProcessFulfillmentOrderBillingAccountListEBM to ProcessBillingAccountListEBM.xsl

This diagram illustrates the relationship of CommsProcessFulfillmentOrderBillingAccountListEBF with the other services in the integration flow:




CommsProcessFulfillmentOrderBillingAccountListEBF

The CommsProcessFulfillmentOrderBillingAccountListEBF enterprise business flow is implemented as an asynchronous delayed response Business Process Execution Language (BPEL) process.

**For more information** about EBFs, see *Oracle Application Integration Architecture – Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Constructing EBFs" and *Oracle Application Integration Architecture – Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs," Enterprise Business Flow (EBF) Processes.

## CommsProcessBillingAccountListEBF

This EBF service creates or synchronizes all the customer accounts and billing profiles in an appropriate billing system. The Order Processing integration flow invokes this service with a list of customer account IDs, billing profile IDs, and the target system ID. When the process is complete, a response is sent back to the order flow confirming that all accounts have been set up in the target billing system, and the order processing can continue.

This service provides two operations. One accepts the ProcessBillingAccountListEBM and is used by the process to accept the customer data to be synced. The other one is used by the process to send the response back to the calling process (using the SyncCustomerPartyListResponseEBM). The data area of the message contains one or more customer account IDs. For each account, one or more bill profile IDs must be synced to the target billing system. The customer data indicates both the hierarchical and the paying relationships between the accounts.

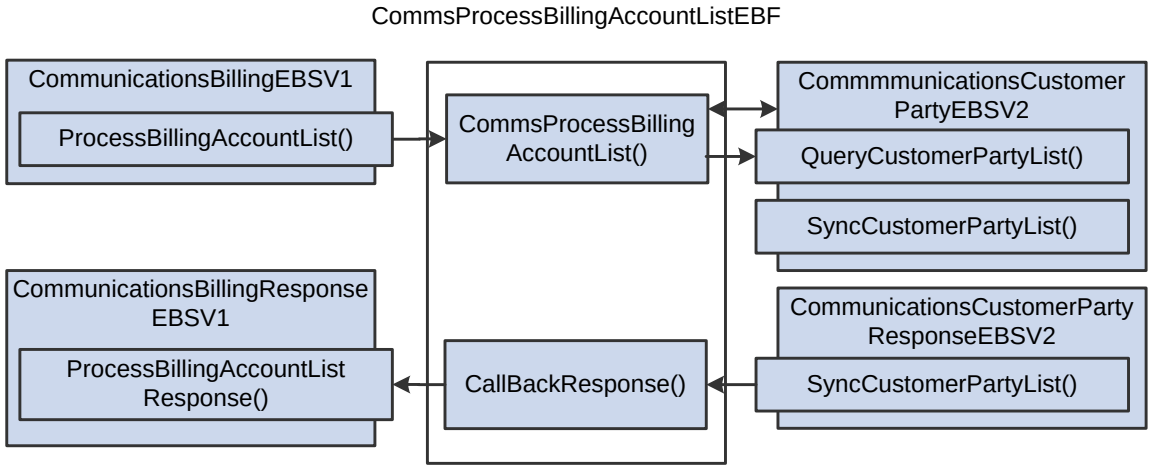
This service creates or synchronizes one or more customers (identified by ID only) and their billing profiles to a particular target billing system (identified in the EBM header). Therefore, the responsibilities of this service include:

- Determining whether the customer already exists and is up to date in the target billing system.

  If so, optimize and do not try to create or synchronize.

- Retrieving the customer data from the appropriate Siebel CRM system using the provided IDs, if necessary.
- Optimizing, if possible, the number and size of queries back into Siebel CRM for the customer data.
- Creating or updating the customers and billing profiles in the target billing system, reflecting the customer hierarchy, and paying relationships among the customers.

The following diagram illustrates the relationship of the CommsProcessBillingAccountListEBF with the other services in the integration flow:



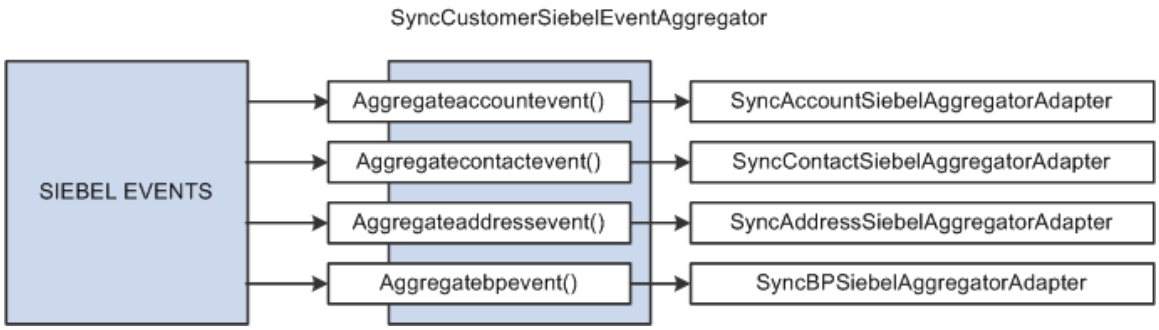

## CommsProcessBillingAccountListEBF

**For more information** about EBFs, see *Oracle Application Integration Architecture – Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Constructing EBFs" and *Oracle Application Integration Architecture – Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs," Enterprise Business Flow (EBF) Processes.

## SyncCustomerSiebelEventAggregator

This service is responsible for receiving Siebel update account events and collating them into an AIA database table.

This diagram illustrates the relationship of the SyncCustomerSiebelEventAggregator with the other services in the integration flow:




### SyncCustomerSiebelEventAggregator

This service provides four operations, one for each of the object types that are updated:

- Aggregateaccountevent:

  Receives the Account Updated Siebel message
  Extracts the account ID, contact IDs, and address IDs from the message
  Invokes the SyncAccountSiebelAggregatorAdapter to store these IDs into the AIA_AGGREGATED_ENTITIES database table

- Aggregatecontacttevent:

  Receives the Contact Update Siebel message
  Extracts the account IDs, bill profile IDs, and contact IDs from the message
  Invokes the SyncContactSiebelAggregatorAdapter to store these IDs in the AIA_AGGREGATED_ENTITIES database table

- Aggregateaddresstevent:

  Receives the Address Update Siebel message
  Extracts the account IDs, bill profile IDs, and address IDs from the message
  Invokes the SyncAddressSiebelAggregatorAdapter to store these IDs into the AIA_AGGREGATED_ENTITIES database table

- Aggregatebpevent:

  Receives the BillingProfile Updated Siebel message
  Extracts the BillingProfile ID and the associated account ID from the message
  Invokes the SyncBPSiebelAggregatorAdapter to store these IDs in the AIA_AGGREGATED_ENTITIES database table

**For more information** about the Event Aggregation programming model, see *Oracle Application Integration Architecture – Foundation Pack 2.5: Integration Developer's Guide,* "Describing the Event Aggregation Programming Model."

## SyncAccountSiebelAggregatorAdapter

This service aggregates the account events generated in Siebel CRM when an account is created or updated. This service invokes a PL/SQL procedure, AIA_AGGREGATOR_PUB.SIEBEL_AGGREGATE_ACCOUNT, which does the actual aggregation in the AIA aggregator table.

## SyncContactSiebelAggregatorAdapter

This service aggregates the account events generated in Siebel CRM when an account is created or updated. This service invokes a PL/SQL procedure, AIA_AGGREGATOR_PUB.SIEBEL_AGGREGATE_CONTACT, which does the actual aggregation in the AIA aggregator table.

## SyncAddressSiebelAggregatorAdapter

This service aggregates the account events generated in Siebel CRM when an account is created or updated. This service invokes a PL/SQL procedure, AIA_AGGREGATOR_PUB.SIEBEL_AGGREGATE_ADDRESS, which does the actual aggregation in the AIA aggregator table.

## SyncBPSiebelAggregatorAdapter

This service aggregates the account events generated in Siebel CRM when an account is created or updated. This service invokes a PL/SQL procedure, AIA_AGGREGATOR_PUB.SIEBEL_AGGREGATE_BP, which does the actual aggregation in the AIA aggregator table.

## SyncAcctSiebelAggrEventConsumer

This service extracts the account IDs stored in the AIA_AGGREGATED_ENTITIES database table and sends them forward to the SyncAccountSiebelReqABCSImpl service.

Sequencing is enabled for this service. When this consumer calls the requestor for further processing and the requestor fails, any subsequent update for that customer will not be processed until proper action is taken on the messages in the sequencer. If the failure is due to a business error then messages must be removed from the queue for the subsequent messages to process. If the failure is system related then messages in the resequencer can be retried to move the message from the resequencer queue and thereby enabling subsequent messages to be processed. Any updates for other errors will be processed as usual.

## SyncAccountSiebelReqABCSImpl

This service is responsible for transforming the Siebel message into the SyncCustomerPartyList EBM format and invoking the SyncCustomerPartyList operation of the CustomerPartyEBSV2.

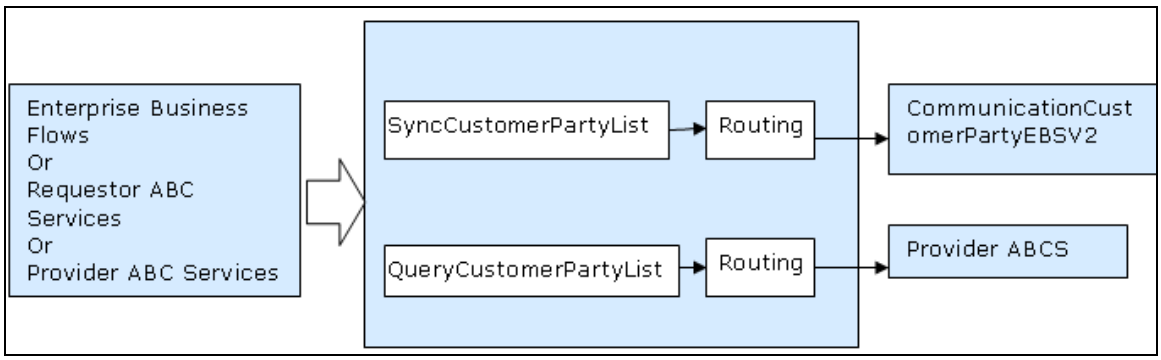
## CustomerPartyEBSV2

CustomerPartyEBSV2 exposes all of the enterprise operations that can be performed with a CustomerParty enterprise object.

CustomerPartyEBSV2 service uses the following operations:

- SyncCustomerPartyList
- QueryCustomerPartyList

This diagram illustrates the relationship of QueryCustomerPartyListSiebelProvABCSImplV2 with the other services in the integration flow:




### CustomerPartyEBSV2

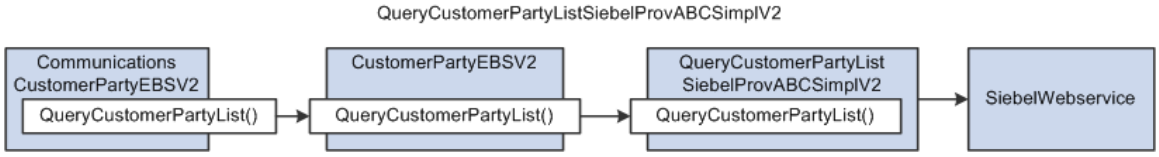
The CustomerPartyEBSV2 is implemented as a lightweight EBS routing service.

**For more information** about this EBS, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs."

## QueryCustomerPartyListSiebelProvABCSImplV2

CustomerPartyEBSV2 invokes the QueryCustomerPartyListSiebelProvABCSImplV2 service when the routing rules determine that Siebel CRM is to be the service provider for the QueryCustomerPartyList EBS operation.

This diagram illustrates the relationship of QueryCustomerPartyListSiebelProvABCSImplV2 with the other services in the integration flow:




### QueryCustomerPartyListSiebelProvABCSImplV2

This service has one synchronous request and reply operation, QueryCustomerPartyList.

## CommunicationsCustomerPartyResponseEBSV2

CommunicationsCustomerPartyResponseEBSV2 exposes all of the enterprise response operations that can be performed with a CustomerParty enterprise object. All of the customer management integration flows use the operations provided by this enterprise business service.

This diagram illustrates the relationship of CommunicationsCustomerPartyResponseEBSV2 with the other services in the integration flow:


CommunicationsCustomerPartyResponseEBSV2

**For more information** about this EBS, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs."

## SyncCustomerPartyListBRMCommsProvABCSImpl

The CommsProcessBillingAccountListEBF or SyncAccountSiebelReqABCSImpl service invokes SyncCustomerPartyListBRMCommsProvABCSImpl. It performs the following actions:

1. Receives the SyncCustomerPartyListEBM.
2. Loops through each data area:

   If the current account is a child account, it checks whether the parent account has been synced. Child should be synced only when the parent has been synced.

   Based on the action code associated with each account, it goes to the Create block (used when a new account needs to be synced) or the Update block (used when an existing account is to be updated).

3. Creates the block:

   Transforms the SyncCustomerPartyListEBM to the Oracle BRM-specific ABM (PCM_OP_CUST_COMMIT_CUSTOMER_Inmsg).

   Calls the PCM_OP_CUST_COMMIT_CUSTOMER opcode with the Oracle BRM ABM.

   Transforms the response from the PCM_OP_CUST_COMMIT_CUSTOMER opcode call to SyncCustomerPartyListResponseEBM.

   While transforming, the service populates the following cross-reference tables with the Oracle BRM IDs obtained:

   - CUSTOMERPARTY_ACCOUNTID
   - CUSTOMERPARTY_ADDRESSID

- CUSTOMERPARTY_CONTACTID
- CUSTOMERPARTY_BILLPROFILEID
- CUSTOMERPARTY_PAYPROFILEID

4. Updates the block:

   If the account is a child account:

   - Gets the corresponding parent account by calling the PCM_OP_BILL_GROUP_GET_PARENT.
   - If the parent obtained from the EBM is different from the parent obtained from the opcode call, then it moves the child account to the new parent (as directed by the EBM) by calling the PCM_OP_CUSTCARE_MOVE_ACCT.

   Creates the PCM_OP_CUST_UPDATE_CUSTOMER input message by a transformation from the SyncCustomerPartyListEBM.

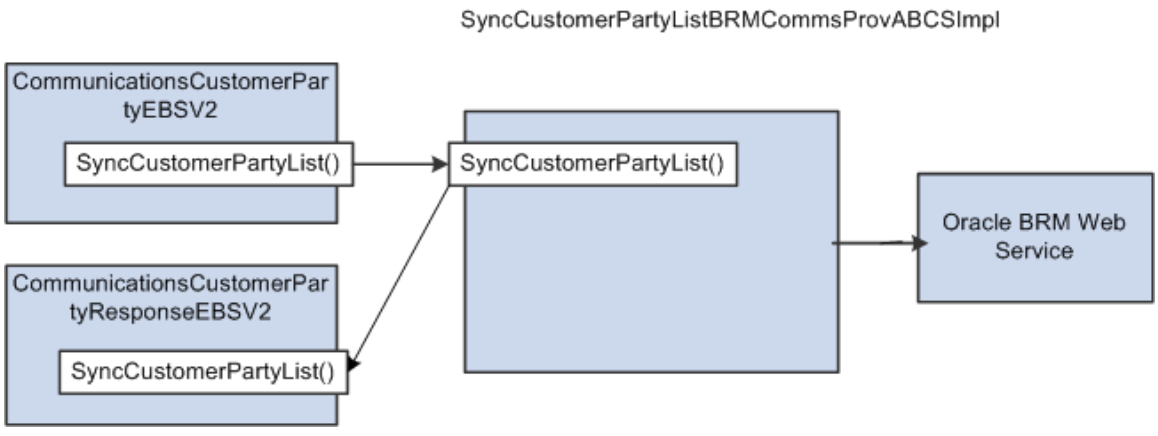
   Checks whether the AIAConfig property EnableAccountStatusSync is set to true. If set to true, then it creates the PCM_OP_CUST_SET_STATUS input message from the SyncCustomerPartyListEBM. Calls the opcode PCM_OP_CUST_SET_STATUS to synchronize the status mentioned in the EBM to BRM.

   If the result of an account update, in which the PayProfile of the account was changed, is SyncCustomerPartyListEBM, then after calling the PCM_OP_CUST_COMMIT_CUSTOMER, it calls the PCM_OP_CUST_DELETE_PAYINFO to delete the earlier PAYINFO object from BRM.

   Transforms the SyncCustomerPartyListEBM to SyncCustomerPartyListResponseEBM.

5. Calls CommunicationsCustomerPartyResponseEBSV2.SyncCustomerPartyListResponse.

   This diagram illustrates the relationship of SyncCustomerPartyListBRMCommsProvABCSImpl with the other services in the integration flow:



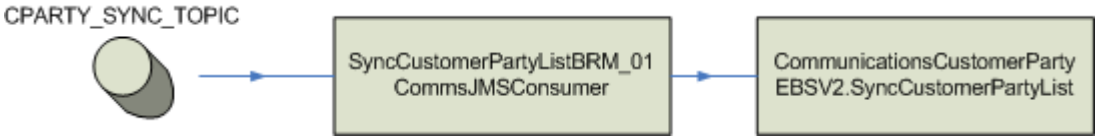

SyncCustomerPartyListBRMCommsProvABCSImpl

## SyncCustomerPartyListBRM_01CommsJMSConsumer

This process listens to the topic CPARTY_SYNC_TOPIC and as soon as a message is picked up, forwards it to the CommunicationsCustomerPartyEBSV2.SyncCustomerPartyList operation.

The following diagram illustrates the relationship of SyncCustomerPartyListBRM_01CommsJMSConsumer with the other services in the integration flow:

The diagram also depicts where the service lies in relation to the other services in the overall integration flow:




### SyncCustomerPartyListBRM_01CommsJMSConsumer

This service performs the following actions:

- Receives the SyncCustomerPartyListEBM.
- Does an xref lookup to determine whether for the given common ID, the corresponding Oracle BRM ID (for BRM_01 or BRM_02, based on the consumer name) exists.

  If it exists, then the service stamps the message with the particular target system ID and passes it forward to the CommunicationsCustomerPartyEBSV2.SyncCustomerPartyList operation.

This process is implemented as an enterprise service bus (ESB) process. This consumer process is intended for multiple Oracle BRM system type installation. If multiple Oracle BRM systems exist, then for each system one such consumer must be deployed.

**For more information** about multiple BRM systems, see Appendix E: Configuring Multiple Instances of Oracle BRM.

# Chapter 7: Configuring the Process Integration for Customer Management

This chapter discusses how to:

- Set up Fusion MiddleWare (FMW).
- Set up Oracle Communications Billing and Revenue Management (Oracle BRM).
- Set up Siebel Customer Relationship Management (Siebel CRM).
- Work with domain value maps (DVMs).
- Work with cross-references.
- Handle error notifications.
- View enterprise business object EBO implementation maps (EIMs).
- Configure the process integration for customer management.

## Setting Up FMW

Perform these steps to set up Fusion Middleware:

1. Add the -DHTTPClient.disableKeepAlives=true property in the opmn.xml file. The opmn.xml file is available here: SOA_HOME/opmn/conf/opmn.xml.

   After you add this property, service oriented architecture (SOA) does not use the same transmission control protocol (TCP) connection more than once to call a Siebel web service. Simultaneous calls to the same Siebel web service cause errors if the same TCP connection is used. This property should be added in the start-up parameters of oc4j_soa. After you add the property, the opmn.xml file looks like this:

```
<process-type id="oc4j_soa" module-id="OC4J" status="enabled">
               <module-data>
                  <category id="start-parameters">
                     <data id="java-options" value="-server -
Xmx2048M -Xms2048M -XX:MaxPermSize=1024M -XX:MaxNewSize=614m -
XX:NewSize=614m -XX:AppendRatio=3 -XX:SurvivorRatio=6 -
Djava.security.policy=$ORACLE_HOME/j2ee/oc4j_soa/config/java2.policy
-Djava.awt.headless=true -Dhttp.webdir.enable=false -
Doraesb.home=/slot/ems1936/oracle/product/10.1.3.1/OracleAS_1/integr
ation/esb -Dhttp.proxySet=false -Doc4j.userThreads=true -
Doracle.mdb.fastUndeploy=60 -Doc4j.formauth.redirect=true -
Djava.net.preferIPv4Stack=true -
Dorabpel.home=/slot/ems1936/oracle/product/10.1.3.1/OracleAS_1/bpel
-
Xbootclasspath^/p:/slot/ems1936/oracle/product/10.1.3.1/OracleAS_1/b
pel/lib/orabpel-boot.jar -Dhttp.proxySet=false -
Daia.home=/slot/ems1936/oracle/product/AIA_HOME" -
DHTTPClient.disableKeepAlives=true/>
                  </category>
```

```
                  <category id="stop-parameters">
                     <data id="java-options" value="-
Djava.security.policy=$ORACLE_HOME/j2ee/oc4j_soa/config/java2.policy
-Djava.awt.headless=true -Dhttp.webdir.enable=false" />
                  </category>
               </module-data>
               <start timeout="600" retry="2" />
               <stop timeout="120" />
               <restart timeout="720" retry="2" />
               <port id="default-web-site" range="12501-12600"
protocol="ajp" />
               <port id="rmi" range="12401-12500" />
               <port id="rmis" range="12701-12800" />
               <port id="jms" range="12601-12700" />
               <process-set id="default_group" numprocs="1" />
            </process-type>
```

***Warning:*** While adding the -DHTTPClient.disableKeepAlives=true property in opmn.xml file, you must be careful. The opmn.xml file is required for the SOA server start-up and any error in this file may cause the server to fail.

2. To prevent multiple retries by the transaction manager:

   Modify the file $ORACLE_HOME/j2ee/{oc4j_instance name}/config/transaction-manager.xml to change the property value 'retry-count' of commit-coordinator from 4 to 0.
   <commit-coordinator retry-count="4"> to <commit-coordinator retry-count="0">

3. To prevent the retries by enterprise service bus (ESB):

   Overwrite the ESB retry entries to disable the retries in the orion-application.xml file that are present in the directories:
   $ORACLE_HOME/j2ee/oc4j_soa/application-deployments/esb-rt and
   $ORACLE_HOME/j2ee/oc4j_soa/application-deployments/esb-dt with the following values after backing up the original file.
   <property name="InboundRetryCount" value="0" />
   <property name="InboundRetryInterval" value="0" />
   <property name="InboundRetryEnabled" value="false" />
   <property name="OutboundRetryCount" value="0" />
   <property name="OutboundRetryInterval" value="0" />
   <property name="OutboundRetryEnabled" value="false" />

   Alternatively, change these property values in the $ORACLE_HOME//integration/esb/config/esb_config.ini file to change the retry values.

These steps are also included in the *Oracle AIA Installation and Upgrade Guide*.

# Setting Up Oracle BRM

You must add an additional phone number validation format to Oracle BRM so that the nonformatted phone numbers coming from Siebel CRM are not rejected by Oracle BRM. The format you must add is this: [0-9] [0-9] [0-9] [0-9] [0-9] [0-9] [0-9] [0-9] [0-9] [0-9].

**For more information**, see *Oracle Communications Billing and Revenue Management (BRM) Documentation*, "BRM Documentation," Managing Customer Accounts, Registering customers, Customizing registration, Specifying how to validate customer contact information.

No other setup steps are required for Oracle BRM, except for configuring the Oracle BRM adapter. See the configuration section.

# Setting Up Siebel CRM

No setup steps are required for Siebel CRM.

For some Siebel CRM interfaces, in Siebel, you must set the process property UTCCanonical to Y.

**For more information** about which Siebel CRM interfaces require you to enable the UTCCanonical process property, see instructions for ACR 474 and 508 in the *Siebel Maintenance Release Guide*.

# Working with DVMs

Domain value maps (DVMs) are a standard feature of the Oracle SOA Suite that enables you to equate lookup codes and other static values across applications, for example, FOOT and FT or US and USA.

DVMs are static in nature, though administrators can add maps as needed. Transactional business processes never update DVMs—they only read from them. They are stored in XML files and cached in memory at run time.

DVM types are seeded for the customer management flows, and administrators can extend the list of mapped values by adding more maps.

These are the DVMs for the process integration for customer management:

| DVM | Columns | Description |
|---|---|---|
| CUSTOMERPARTY_ACCOUNTTYPECODE | SEBL_01,COMMON,BRM_01 | Used to get the type of the account, such as Business or Customer. |
| PROVINCE | SEBL_01,COMMON,BRM_01 | Province name. |
| STATE | SEBL_01,COMMON,BRM_01 | State name. |
| ADDRESS_COUNTRYID | SEBL_01,COMMON,BRM_01 | Country codes. |
| ADDRESS_COUNTRYSUBDIVID | SEBL_01,COMMON,BRM_01 | State codes. |
| CONTACT_SALUTATION | SEBL_01,COMMON,BRM_01 | Salutation (such as Mr., Mrs.) In Oracle BRM, Salutation is not a language-independent code. If Oracle BRM requires salutations in a language other than English, then you must update the DVM with the appropriate Oracle BRM values. |

| DVM | Columns | Description |
|---|---|---|
| CURRENCY_CODE | SEBL_01,COMMON,BRM_01 | Currency codes. |
| CUSTOMERPARTY_BILLPROFILE_BILLTYPECODE | SEBL_01,COMMON,BRM_01 | Bill type (summary and detailed). |
| CUSTOMERPARTY_BILLPROFILEFREQUENCYCODE | SEBL_01,COMMON,BRM_01 | Billing frequency (monthly, yearly, quarterly, and so on). |
| CUSTOMERPARTY PAYPROFILE BANKACCOUNTTYPE | SEBL_01,COMMON,BRM_01 | Bank account type (checking, savings, and so on) |
| CUSTOMERPARTY PAYPROFILE CREDIT CARDTYPE | SEBL_01,COMMON | Credit Card type (Visa, MasterCard, and so on). |
| CUSTOMERPARTY_PAYPROFILE_DELIVERYPREF | COMMON,BRM_01 | Bill media/delivery preference (Email and Mail). |
| CUSTOMERPARTY_PAYPROFILE_PAYMETHODCODE | SEBL_01,COMMON,BRM_01 | Payment profile payment method types (credit card, direct debit, and invoice/bill me). |
| CUSTOMERPARTY_PAYPROFILE_PAYTERMCODE | COMMON,BRM_01 | Payment term codes. |
| CUSTOMERPARTY_STATUSCODE | SEBL_01,COMMON,BRM_01 | Account status codes. |
| PHONENUMBER_TYPE | SEBL_01,COMMON,BRM_01 | Phone number type codes (home, work, mobile, fax, and so on) |

**For more information,** see *Oracle Application Integration Architecture – Foundation Pack 2.5: Integration Developer's Guide,* "Understanding Message Transformation, Enrichment, and Configuration," DVMs.

# Working with Cross-References

Cross-references map and connect the records within the application network, and they enable these applications to communicate in the same language. The integration server stores the relationship in a persistent way so that others can refer to it.

This table lists the customer management cross-references:

| Cross-Reference Table Name | Column Names: Column Values | | | Description |
|---|---|---|---|---|
| | | | | |
| CUSTOMERPARTY_ACCOUNTID | COMMON | SEBL_01 | BRM_01 | Siebel Account ID is mapped one-to-one to the Oracle BRM Account ID. |
| | Account ID | Account ID | Account POID | |
| | | | | |
| CUSTOMERPARTY_BI | COMMON | SEBL_01 | BRM_01 | Siebel Bill Profile ID is mapped one- |

| Cross-Reference Table Name | Column Names: Column Values | | | Description |
|---|---|---|---|---|
| LLPROFILEID | Bill Profile ID | Bill Profile ID | bill-info POID | to-one to the Oracle BRM bill-info ID. |
| | | | | |
| CUSTOMERPARTY_PAYPROFILEID | COMMON | SEBL_01 | BRM_01 | Siebel Bill Profile ID is mapped one-to-one to the Oracle BRM pay-info ID. |
| | Payment Profile ID | Bill Profile ID | Pay-info POID | |
| | | | | |
| CUSTOMERPARTY_ADDRESSID | COMMON | SEBL_01 | BRM_01* | Oracle BRM Account ID is cross-referenced here if the address is used as the billing address (name-info[1]) on that account.<br>Oracle BRM pay-info ID is cross-referenced if the address is used as the pay-info address on an account.<br>The ACCOUNT and PAYINFO codes are prefixed to each ID to indicate the type of the ID. |
| | Address ID | Address ID | Account POID<br>pay-info POID | |
| | | | | |
| CUSTOMERPARTY_CONTACTID | COMMON | SEBL_01 | BRM_01* | Oracle BRM Account ID is cross-referenced if the contact is used as the name (name-info[1]) on that account.<br>Oracle BRM pay-info ID is cross-referenced if the contact is used as the name on the pay-info on an account.<br>The ACCOUNT and PAYINFO codes are prefixed to each ID to indicate the type of the ID. |
| | Contact ID | Contact ID | Account POID<br>pay-info POID | |
| | | | | |
| CUSTOMERPARTY_DEFAULTBALANCEGROUPID | COMMON* | BRM_01 | | This cross-reference map the default balance group to the common account ID This is populated after account creation in the CreateCustomerPartyProviderABCSImpl service, and is referenced by the order flow during service creation. |
| | Account ID | Balance Group POID | | |
| | | | | |
| CUSTOMERPARTY_PARTYID | SEBL_01,COMMON,EBIZ_01,UCM_01, SAP_01 | | | Customer Party IDs |

| Cross-Reference Table Name | Column Names: Column Values | Description |
|---|---|---|
| CUSTOMERPARTY_PARTYLOCATIONID | SEBL_01,COMMON,EBIZ_01,UCM_01, SAP_01 | Customer Party Location IDs |
| CUSTOMERPARTY_CONTACTID | SEBL_01,COMMON,EBIZ_01,UCM_01, BRM_01, SAP_01 | Customer Party contact IDs.<br>Oracle BRM account ID is cross-referenced here if the contact is used as the name (name-info[1]) on that account.<br>Oracle BRM pay-info ID is cross-referenced here if the contact is used as the name on the pay-info on an account.<br>The ACCOUNT and PAYINFO codes are prefixed to each ID to indicate what type of ID it is. |
| CUSTOMERPARTY_LOCATIONREFID | SEBL_01,COMMON,EBIZ_01,UCM_01 | Customer Party Location Reference IDs. |
| CUSTOMERPARTY_ACCOUNT_PHONECOMMID | SEBL_01,COMMON,EBIZ_01,UCM_01, SAP_01 | Customer Party Account's Phone contact points. |
| CUSTOMERPARTY_ACCOUNT_FAXCOMMID | SEBL_01,COMMON,EBIZ_01,UCM_01, SAP_01 | Customer Party Account's Fax contact points. |
| CUSTOMERPARTY_ACCOUNT_WEBCOMMID | SEBL_01,COMMON,EBIZ_01,UCM_01 | Customer Party Account's Email/Web contact points. |
| CUSTOMERPARTY_CONTACT_PHONECOMMID | SEBL_01,COMMON,EBIZ_01,UCM_01 | Customer Party Contact's Phone contact points. |
| CUSTOMERPARTY_CONTACT_FAXCOMMID | SEBL_01,COMMON,EBIZ_01,UCM_01 | Customer Party Contact's Fax contact points. |
| CUSTOMERPARTY_CONTACT_EMAILCOMMID | SEBL_01,COMMON,EBIZ_01,UCM_01 | Customer Party Contact's Email/Web contact points. |

# Handling Error Notifications

Based on the roles defined for the services, email notifications are sent if a service ends due to an error. This table lists the errors that are caused by the process integration for customer management services:

| Service Name | Error Code | Possible Cause |
|---|---|---|
| SyncCustomerPartyListBRMCommsProvABCSImpl | AIA_ERR_AIACOMCMPI_0004 | Subordinate account cannot have multiple parent accounts. |
| SyncCustomerPartyListBRMCommsProvABCSImpl | AIA_ERR_AIACOMCMPI_0005 | Ambiguous subordinate bill profile update: Multiple distinct Pay-From-Party billing profile references are associated with a single Prior Pay-From-Party billing profile reference. |
| SyncCustomerPartyListBRMCommsProvABCSImpl | AIA_ERR_AIACOMCMPI_0006 | None of the existing subordinate bill profiles are included in the move account request. |
| CommsProcessBillingAccountListEBF | AIA_ERR_AIACOMCMPI_0001 | EBMHeader/Sender/ID is required. |
| CommsProcessBillingAccountListEBF | AIA_ERR_AIACOMCMPI_0002 | EBMHeader/Target/ID is required. |
| CommsProcessBillingAccountListEBF | AIA_ERR_AIACOMCMPI_0003 | Account sequence error: Pay-From accounts and billing profiles must appear before dependent and subordinate accounts and billing profiles. |

## Describing Delivered Error Notification Roles and Users

The following roles and users are delivered as default values for issuing error notifications for the process integration for customer management.

Actor roles and users:

- Role: AIAIntegrationAdmin. User: AIAIntegrationAdminUser.

The default password set for all users is welcome1.

**For more information** about setting up error notifications using these values, see *Oracle Application Integration Architecture Core Infrastructure Components Guide*, "Setting Up Error Notifications and Trace Logging."

## Order Fallout Management

When an order is submitted from Siebel CRM, the order may fail while customer-related information is being interfaced to Oracle BRM. In that case, a trouble ticket is generated by the Order Fallout flow. The trouble ticket generated is displayed in Siebel. This helps the customer service representative (CSR) in getting notified about any error while processing the order without checking the instances in the Business Process Execution Language (BPEL) Console.

Whenever an error occurs during customer synchronization, it is propagated to the CommsProcessFulfillmentOrderBillingAccountListEBF. This enterprise business flow (EBF) generates an error notification in the error topic (similar to any other Oracle Application Integration Architecture (Oracle AIA) process). From the error topic, the order fallout flow is triggered only for the CommsProcessFulfillmentOrderBillingAccountListEBF (among all the processes in customer management process integration), thus generating one trouble ticket for any error.

**For more information** about order fallout, see Chapter 8: Understanding the Process Integration for Order Fallout Management.

# Viewing EBO EIMs

**For more information** about how services are mapped, see the My Oracle Support document: EBO Implementation Maps (EIMs) 881022.1.

# Configuring the Process Integration for Customer Management

Configure these properties in the AIAConfigurationProperties.xml file. The file is located in <aia.home>/config/. Entries in the AIAConfigurationProperties.xml file are case-sensitive.

**Note:** Whenever the AIAConfigurationProperties.xml file is updated, the file must be reloaded for updates to be reflected in the applications or services that use the updated properties. Click the Reload button on the Configuration page in the Oracle AIA Console to perform this reload. Alternatively, you can perform the reload by rebooting the server.

**For more information**, see *Oracle Application Integration Architecture – Foundation Pack 2.5: Core Components Guide*, "Loading Oracle AIA Configuration File Updates."

These are the settings for the SyncCustomerPartyListBRMCommsProvABCSImpl service property:

| Property Name | Value/Default Values | Description |
|---|---|---|
| EnableAccountStatusSync | true/false. Default = false | This property when set to true, updates the status (active or inactive) of the account from Siebel to Oracle BRM. |
| ABCSExtension.prexformEBMtoABM | true/false. Default = false | This property governs whether the application business connector service (ABCS) Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined in AIA ABCS Extension guidelines) is invoked. The name of the property indicates which extension point will be enabled |
| ABCSExtension.PreInvokePCM_OP_BILL_GROUP_GET_PARENTABM | true/false. Default = false | This property governs whether the ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined in Oracle AIA ABCS Extension guidelines) is invoked. |

| Property Name | Value/Default Values | Description |
|---|---|---|
| | | This property is required for extensibility. The name of the property clearly suggests which extension point will be enabled. |
| ABCSExtension.PostInvokePCM_OP_BILL_GROUP_GET_PARENTABM | true/false. Default = false | This property governs whether the ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked. This property is required for extensibility. The name of the property indicates which extension point will be enabled. |
| ABCSExtension.reInvokePCM_OP_SEARCHABM | true/false. Default = false | This property governs whether the ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined in AIA ABCS Extension guidelines) is invoked. This property is required for extensibility. The name of the property indicates which extension point will be enabled. |
| ABCSExtension.PostInvokePCM_OP_SEARCHABM | true/false. Default = false | This property governs whether the ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined along AIA ABCS Extension guidelines) is invoked. This property is required for extensibility. The name of the property indicates which extension point will be enabled. |
| ABCSExtension.PreInvokeABSPCM_OP_CUST_COMMIT_CUSTOMERABM | true/false. Default = false | This property governs whether the ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined in AIA ABCS Extension guidelines) is invoked. The name of the property indicates which extension point will be enabled. |
| ABCSExtension.PostInvokeABSPCM_OP_CUST_COMMIT_CUSTOMERABM | true/false. Default = false | This property governs whether the ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined in AIA ABCS Extension guidelines) is invoked. The name of the property indicates which extension point will be enabled. |

| Property Name | Value/Default Values | Description |
|---|---|---|
| ABCSExtension.PreInvokePCM_OP_CUSTCARE_MOVE_ACCTABM | true/false. Default = false | This property governs whether the ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined in AIA ABCS Extension guidelines) is invoked. The name of the property indicates which extension point will be enabled. |
| ABCSExtension.PostInvokePCM_OP_CUSTCARE_MOVE_ACCTABM | true/false. Default = false | This property governs whether the ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined in AIA ABCS Extension guidelines) is invoked. The name of the property indicates which extension point will be enabled. |
| ABCSExtension.PreInvokePCM_OP_CUST_UPDATE_CUSTOMERABM | true/false. Default = false | This property governs whether the ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined in AIA ABCS Extension guidelines) is invoked. The name of the property indicates which extension point will be enabled. |
| ABCSExtension.PostInvokePCM_OP_CUST_UPDATE_CUSTOMERABM | true/false. Default = false | This property governs whether the ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined in AIA ABCS Extension guidelines) is invoked. The name of the property indicates which extension point will be enabled. |
| ABCSExtension.PreInvokePCM_OP_CUST_DELETE_PAYINFOABM | true/false. Default = false | This property governs whether the ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined in AIA ABCS Extension guidelines) is invoked. The name of the property indicates which extension point will be enabled. |
| ABCSExtension.PostInvokePCM_OP_CUST_DELETE_PAYINFOABM | true/false. Default = false | This property governs whether the ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined in AIA ABCS Extension guidelines) is invoked. The name of the property indicates which |

| Property Name | Value/Default Values | Description |
|---|---|---|
| | | extension point will be enabled. |
| ABCSExtension.postexformABMtoEBM | true/false. Default = false | This property governs whether the ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined in AIA ABCS Extension guidelines) is invoked. The name of the property indicates which extension point will be enabled. |
| AccountLevelBalanceGroupName | Account Level Balance Group | This property is used to name the default balance group created in BRM when an account is created. |
| Default.SystemID | BRM_01 | This property specifies the default target system ID to be populated in the enterprise business message (EBM) Header in case the value is not coming from the Requestor. |
| Routing.BRMCUSTService.BRM_01.EndpointURI | eis/BRM | This property specifies the Connection factory to connect to the Oracle BRM Java EE Connector Architecture (JCA) adapter for the first instance of the Oracle BRM in case of multiple Oracle BRM instances.<br>For more information about multiple Oracle BRM systems, see Appendix E: Configuring Multiple Instances of Oracle BRM. |
| Routing.BRMCUSTService.BRM_02.EndpointURI | eis/BRM1 | This property specifies the Connection factory to connect to the Oracle BRM JCA adapter for the second instance of the Oracle BRM in case of multiple Oracle BRM instances.<br>For more information about multiple BRM systems, see Appendix E: Configuring Multiple Instances of Oracle BRM. |
| Routing.BRMCUSTService.RouteToCAVS | True/false | This property specifies whether the end point should route to Composite Application Validation System (CAVS). |
| Routing.BRMCUSTService_ptt.BRM_01.EndpointURI | eis/BRM | This property specifies the Connection factory to connect to the BRM JCA adapter for the first instance of the BRM in case of |

| Property Name | Value/Default Values | Description |
|---|---|---|
| | | multiple BRM instances. For more information about multiple BRM systems, see Appendix E: Configuring Multiple Instances of Oracle BRM. |
| Routing.BRMCUSTService_ptt.BRM_02.EndpointURI | eis/BRM1 | This property specifies the Connection factory to connect to the BRM JCA adapter for the second instance of the BRM in case of multiple BRM instances.<br>For more information about multiple BRM systems, see Appendix E: Configuring Multiple Instances of Oracle BRM. |
| Routing.BRMCUSTService_ptt.RouteToCAVS | True/false | This property specifies whether the CAVS service needs to be invoked. |
| Routing.BRMCUSTService_ptt.CAVS.EndpointURI | http://${http.hostname}:${http.port}/AIAValidationSystemServlet/asyncrequestrecipient?simid=1000 | This property specifies the end point URL for the CAVS Service. |
| Routing.BRMCUSTService.CAVS.EndpointURI | http://${http.hostname}:${http.port}/AIAValidationSystemServlet/syncresponsesimulator?simid=1000 | This property specifies the end point URL for the CAVS Service. |
| Routing.BRMCUSTCAREService.BRM_01.EndpointURI | eis/BRM | This property specifies the Connection factory to connect to the Oracle BRM JCA adapter for the first instance of the Oracle BRM in case of multiple Oracle BRM instances for the CUSTCare opcode of Oracle BRM.<br>For more information about multiple BRM systems, see Appendix E: Configuring Multiple Instances of Oracle BRM. |
| Routing.BRMCUSTCAREService.BRM_02.EndpointURI | eis/BRM1 | This property specifies the Connection factory to connect to the Oracle BRM JCA adapter for the second instance of the Oracle BRM in case of multiple Oracle BRM instances for the CUSTCare opcode of Oracle BRM.<br>For more information about multiple BRM systems, see Appendix E: Configuring Multiple Instances of Oracle BRM. |
| Routing.BRMCUSTCAREService.RouteToCAVS | True/false | This property specifies whether to route to CAVS Service. |

| Property Name | Value/Default Values | Description |
|---|---|---|
| Routing.BRMCUSTCAREService.CAVS.EndpointURI | http://${http.hostname}:${http.port}/AIAValidationSystemServlet/syncresponsesimulator | This property specifies the end point URL for the CAVS Service. |
| Routing.BRMBILLService.BRM_01.EndpointURI | eis/BRM | This property specifies the Connection factory to connect to the Oracle BRM JCA adapter for the first instance of the Oracle BRM in case of multiple Oracle BRM instances for the BillService opcode.<br>For more information about multiple Oracle BRM systems, see Appendix E: Configuring Multiple Instances of Oracle BRM. |
| Routing.BRMBILLService.BRM_02.EndpointURI | eis/BRM1 | This property specifies the Connection factory to connect to the Oracle BRM JCA adapter for the second instance of the Oracle BRM in case of multiple Oracle BRM instances for the BillService opcode.<br>For more information about multiple Oracle BRM systems, see Appendix E: Configuring Multiple Instances of Oracle BRM. |
| Routing.BRMBILLService.RouteToCAVS | True/false | This property specifies whether to Route to CAVS service. |
| Routing.BRMBILLService.CAVS.EndpointURI | http://${http.hostname}:${http.port}/AIAValidationSystemServlet/syncresponsesimulator | This property specifies the end point URL for the CAVS Service. |
| Routing.BRMBASEService.BRM_01.EndpointURI | eis/BRM | This property specifies the Connection factory to connect to the Oracle BRM JCA adapter for the first instance of the Oracle BRM in case of multiple Oracle BRM instances for the BRMBASEService.<br>For more information about multiple Oracle BRM systems, see Appendix E: Configuring Multiple Instances of Oracle BRM. |
| Routing.BRMBASEService.BRM_02.EndpointURI | eis/BRM1 | This property specifies the Connection factory to connect to the Oracle BRM JCA adapter for the second instance of the Oracle BRM in case of multiple Oracle BRM instances for the BRMBASEService.<br>For more information about multiple Oracle BRM systems, see |

| Property Name | Value/Default Values | Description |
|---|---|---|
| | | Appendix E: Configuring Multiple Instances of Oracle BRM. |
| Routing.BRMBASEService.RouteToCAVS | True/false | This property specifies whether the CAVS service should be invoked. |
| Routing.BRMBASEService.CAVS.EndpointURI | http://${http.hostname}:${http.port}/AIAValidationSystemServlet/syncresponsesimulator | This property specifies the end point URL for the CAVS Service. |
| Routing.BRMTXNService.BRM_01.EndpointURI | eis/BRM | This property specifies the Connection factory to connect to the Oracle BRM JCA adapter for the first instance of the Oracle BRM in case of multiple Oracle BRM instances for the TXNService opcode.<br>For more information about multiple Oracle BRM systems, see Appendix E: Configuring Multiple Instances of Oracle BRM. |
| Routing.BRMTXNService.BRM_02.EndpointURI | eis/BRM1 | This property specifies the Connection factory to connect to the Oracle BRM JCA adapter for the second instance of the Oracle BRM in case of multiple Oracle BRM instances for the TXNService opcode.<br>For more information about multiple Oracle BRM systems, see Appendix E: Configuring Multiple Instances of Oracle BRM. |
| Routing.BRMTXNService.RouteToCAVS | True/false | This property specifies whether to route to CAVS Service. |
| Routing.BRMTXNService.CAVS.EndpointURI | http://${http.hostname}:${http.port}/AIAValidationSystemServlet/syncresponsesimulator | This property specifies the end point URL for the CAVS Service. |

These are the settings for the SyncAccountSiebelReqABCSImpl service property:

| Property Name | Value/Default Values | Description |
|---|---|---|
| ABCSExtension.PreXformABMtoEBMABM | true/false. Default = false | This property governs whether the ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined in AIA ABCS Extension guidelines) is invoked. This property is required for extensibility. The name of the property indicates which extension point will be enabled. |

| Property Name | Value/Default Values | Description |
|---|---|---|
| ABCSExtension.PreInvokeEBSEBM | true/false. Default = false. | This property governs whether the ABCS Extension is enabled at the predefined plug-into point. If set to true, then the Extension process (defined in AIA ABCS Extension guidelines) is invoked. This property is required for extensibility. The name of the property indicates which extension point will be enabled. |
| Account.ProcessUpdateEventsOnly | true/false. Default = true | Customers must set this property to true. This is required to optimize the flow. By setting this property to true, the Siebel connector will not propagate create events onwards. The OOTB solution supports creation of customers only as part of the order flow.<br>Setting the flag to false will result in a less optimized flow, but OOTB behavior where customer creation occurs as part of the order flow remains the same.<br>For more information, see the *PIP Functional Interoperability Configuration Guide*. |

Settings for the QueryCustomerPartyListSiebelProvABCSImplV2 service property:

**For more information**, see *Siebel CRM Integration Pack for Oracle Order Management: Order to Cash Implementation Guide*.

Settings for the SyncAcctSiebelAggrEventConsumer service property:

**For more information**, see *Siebel CRM Integration Pack for Oracle Order Management: Order to Cash Implementation Guide*.

# Part 4: Implementing the Process Integration for Order Fallout Management

This part includes the following chapters:

Chapter 8: Understanding the Process Integration for Order Fallout Management

Chapter 9: Configuring the Process Integration for Order Fallout Management

# Chapter 8: Understanding the Process Integration for Order Fallout Management

This chapter provides an overview and discusses:

- Business process task flow.
- Solution assumptions and constraints.
- Creating additional listeners to capture order failure notifications.
- Creating trouble tickets.
- Siebel Customer Relationship Management (CRM) interfaces.
- Industry Application Integration Architecture (AIA) components.
- Integration services.

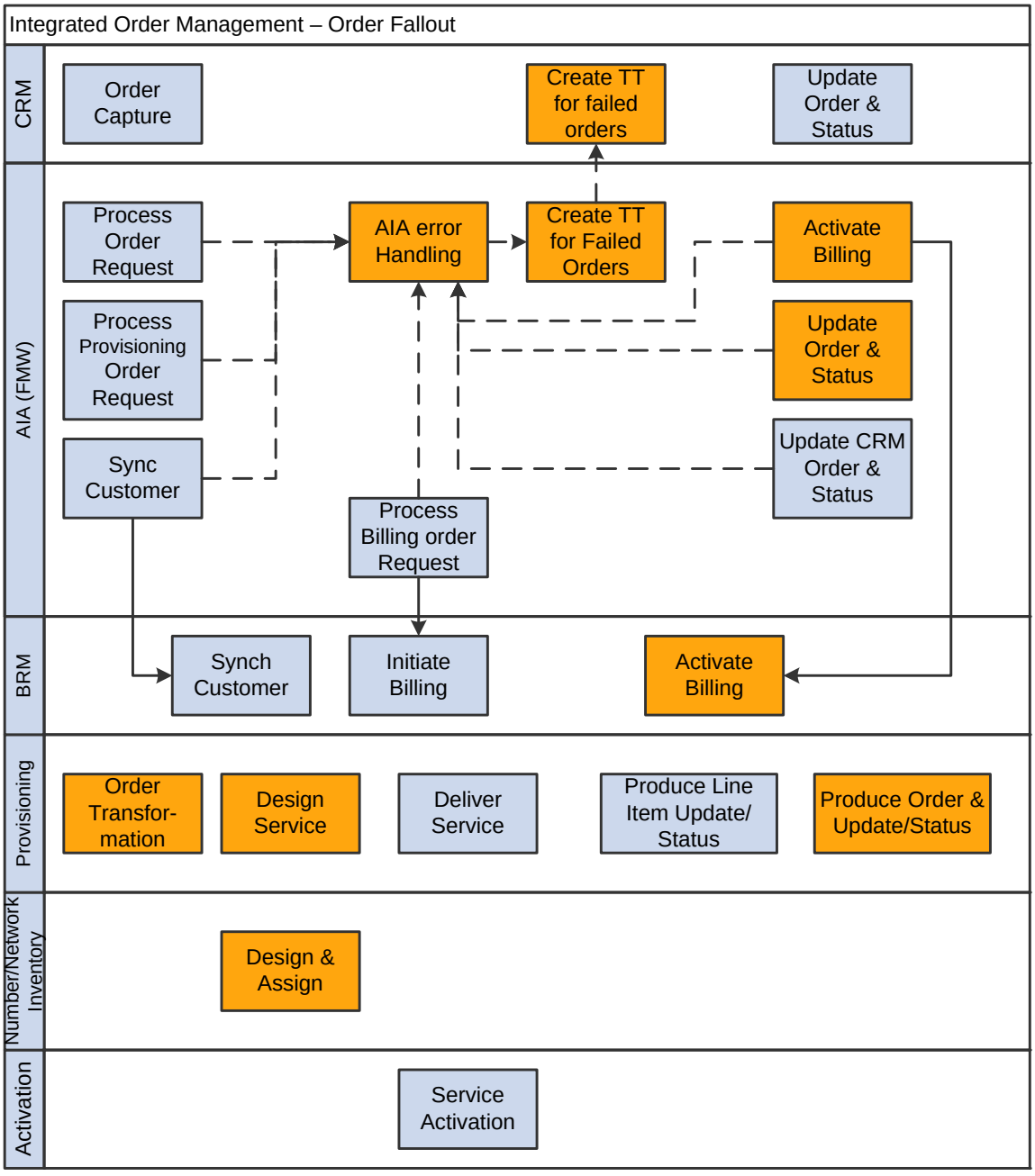
## Overview

Orders that have been submitted in Siebel to reflect a customer's intent to use or purchase services provided by a communications service provider (CSP) are passed to downstream systems for fulfillment and provisioning. Because an order is likely to traverse multiple stages before completion, it could fail during the process. The process integration for order fallout management provides a comprehensive, delivered solution that handles such exceptions by implementing a detection and notification process, making the Order to Bill process integration pack (PIP) more robust. Order fallout uses trouble ticketing for notification and tracking of order failures.

This chapter discusses an alternative solution for order fallout management in which Oracle Service Management (Oracle OSM) is not the central fulfillment system and is not used for order fulfillment and fallout management. The approach adopted for this alternate solution assumes that as delivered, the integration handles a subset of order fallout management functionalities by providing delivered services and artifacts that handle order fallout detection and notification.

This chapter also discusses the functional design required to implement trouble ticket creation in Siebel CRM by the integration when an order fails and an error is detected by the Oracle AIA Error Handler.

This diagram shows the high-level flow of order fulfillment and order fallout management within the capacity of the integration.




High-level order fulfillment and order fallout flows

**Note:** This diagram shows only the interactions for order fallout flow. Additional interactions are part of the order fulfillment flow.

As illustrated in the diagram, orders can fail at various stages while in process. Some of these are:

- Fulfillment systems (Oracle Communications Billing and Revenue Management (Oracle BRM), provisioning, and so on).

- AIA services.
- Fusion middleware (FMW) infrastructure.

If an error occurs at one of the AIA service calls (enterprise business service (EBS), application business connector service (ABCS), and so on), then the service creates an error by invoking the services provided by the Oracle AIA Error Handling Framework to generate a fault message that contains information about the error and also order-specific information that can then be used to create a trouble ticket. The Oracle AIA order fallout management services are then called to create a trouble ticket in Siebel using a Siebel web service. After the trouble ticket is available within Siebel, an order fallout specialist or customer service representative (CSR) can open the trouble ticket and address it either by resubmitting the order after correcting it, or by canceling the order.

**For more information** about how to resolve errors in flows where sequencing is enabled, see Appendix G: Using the Reseuencer Feature of ESB, "Resolving Sequencing Errors".

**For more information** about the resequencer, see the *Oracle AIA Foundation Pack - Integration Developer's Guide*, “Designing and Constructing ABC Services,” Interacting with Participating Applications, Oracle Enterprise Service Bus Resequencer.

**For more information** about the Message Resubmission Utility API, see the *Oracle Enterprise Service Bus Developer's Guide*, “Using the Message Resubmission Utility API.”

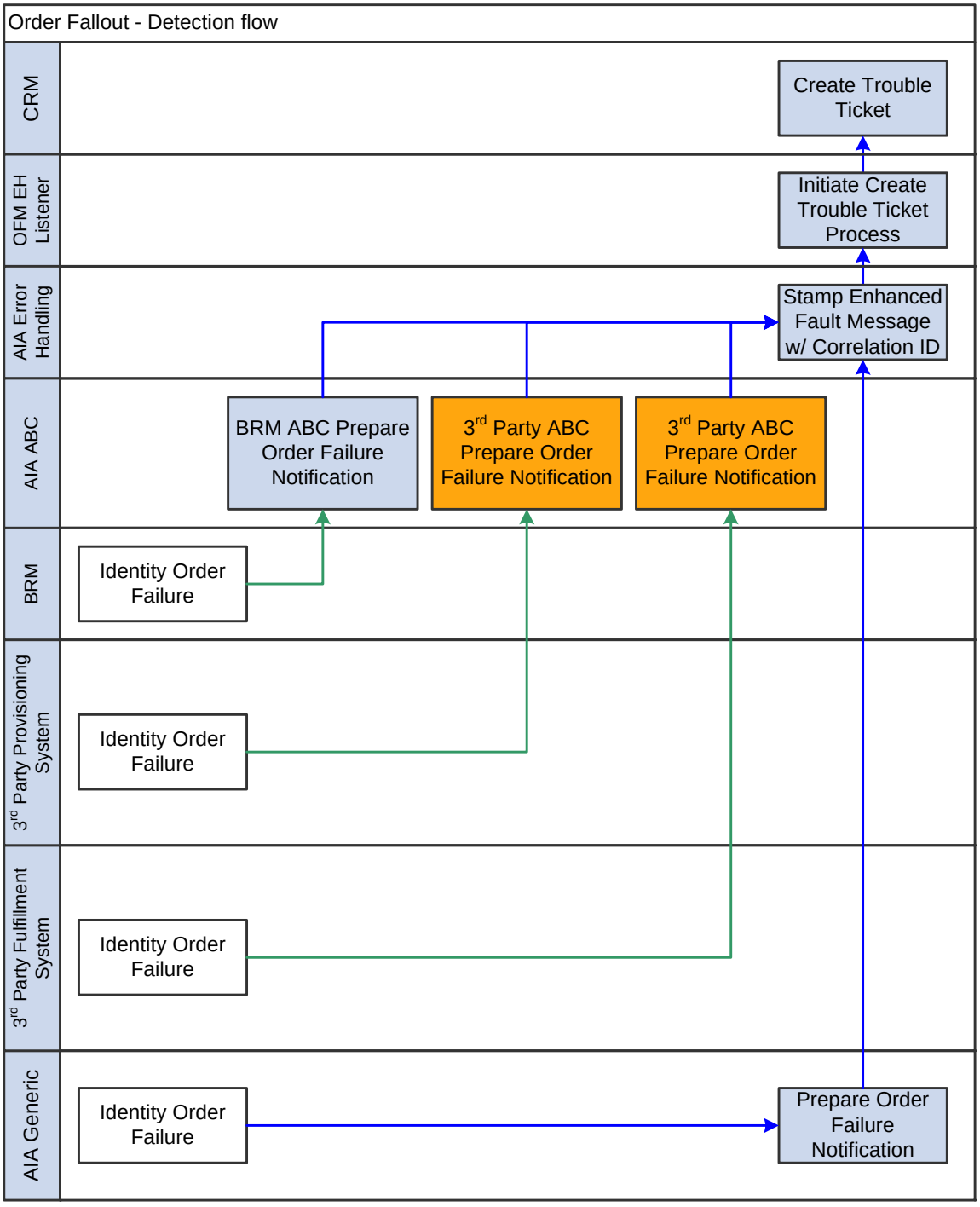
**For more information** about the AIA Error Handling Framework, see *Oracle Application Integration Architecture – Foundation Pack 2.5: Core Infrastructure Components Guide,* "Understanding Error Handling and Logging."

# Business Process Task Flow

This section describes the various integration flows that are part of the process integration for order fallout management. The order fallout process is broadly categorized into three subprocesses:

1. Order Fallout Detection
2. Order Fallout Notification
3. Order Correction

This diagram illustrates the detection subprocess within the order fallout process:




Detection flow

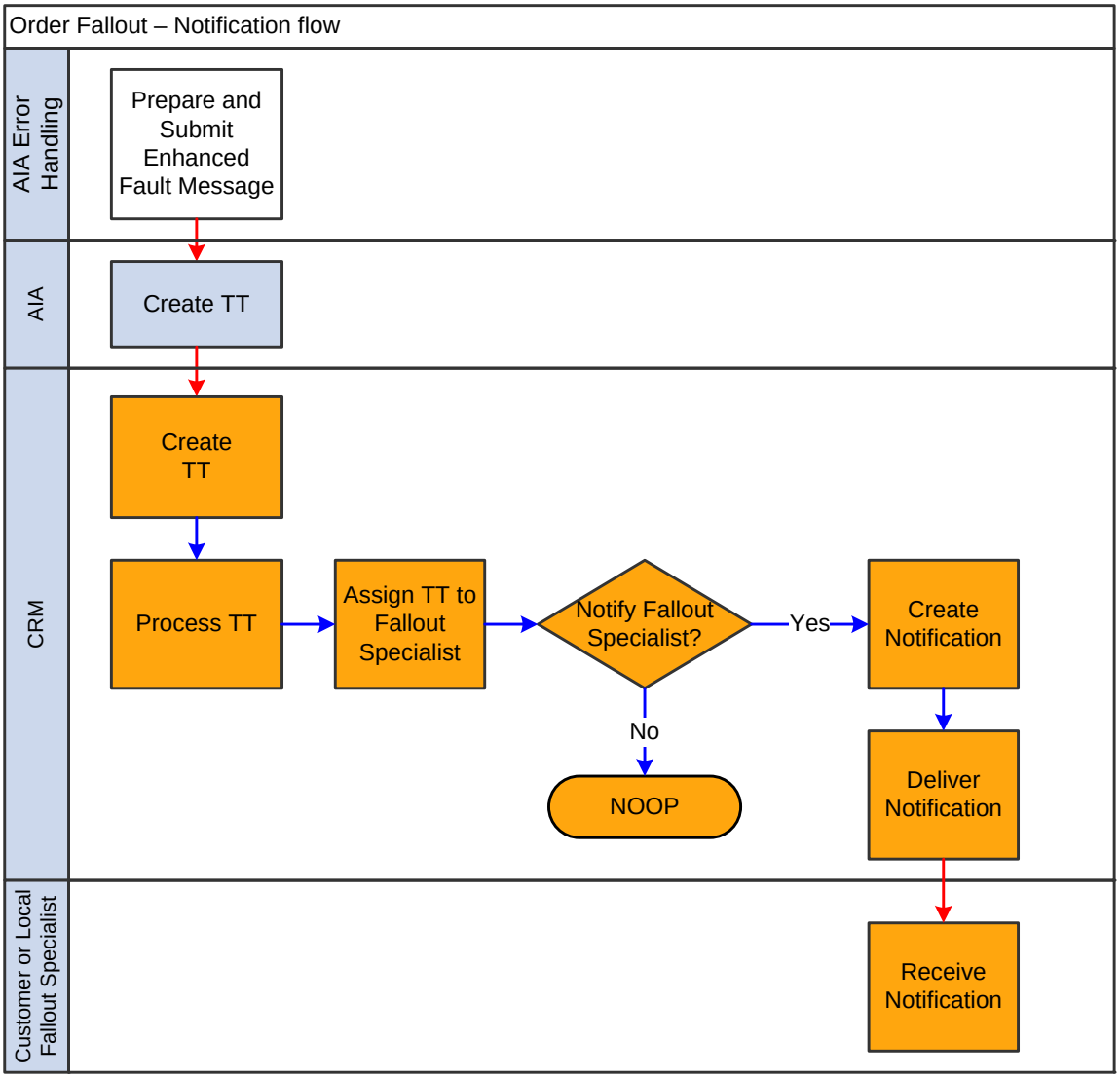
**Note:** The order can fail in any of the preceding application tiers, but this chapter discusses order failure only within Oracle AIA. Other applications and systems are outside the scope of this solution.

When an error occurs within any of the order services, the ABCS (in this case) creates an error in Oracle AIA that is detected by the Oracle AIA Error Handling framework. The framework then creates an enhanced fault message that contains information about the fault and the failed order and publishes it to the AIA Error Java Message Service (JMS) topic. The Oracle AIA Order Fallout Management Error Handling Listener detects the AIA Error Handling Enhanced Fault Message, picks up the message from the queue, and submits it to the order fallout function within Oracle AIA for further processing (creation of trouble ticket).

The AIA Enhanced Fault Message has some of the following key error and order failure information:

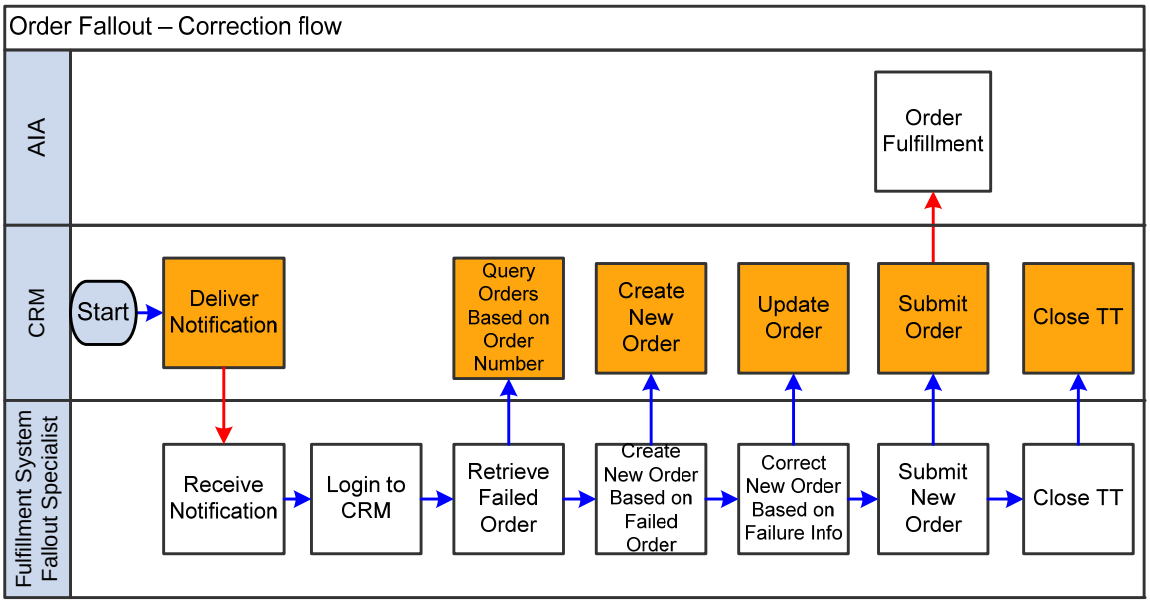
- Faulting Service
- Error Code
- Error Severity
- Error Text
- Time Of Failure
- Order ID
- Order Number
- Order Originating System Code
- Account ID
- Account Name

This diagram illustrates the notification subprocess within the order fallout process:




Notification flow

This diagram illustrates the Siebel CRM correction flow subprocess within the order fallout management process:

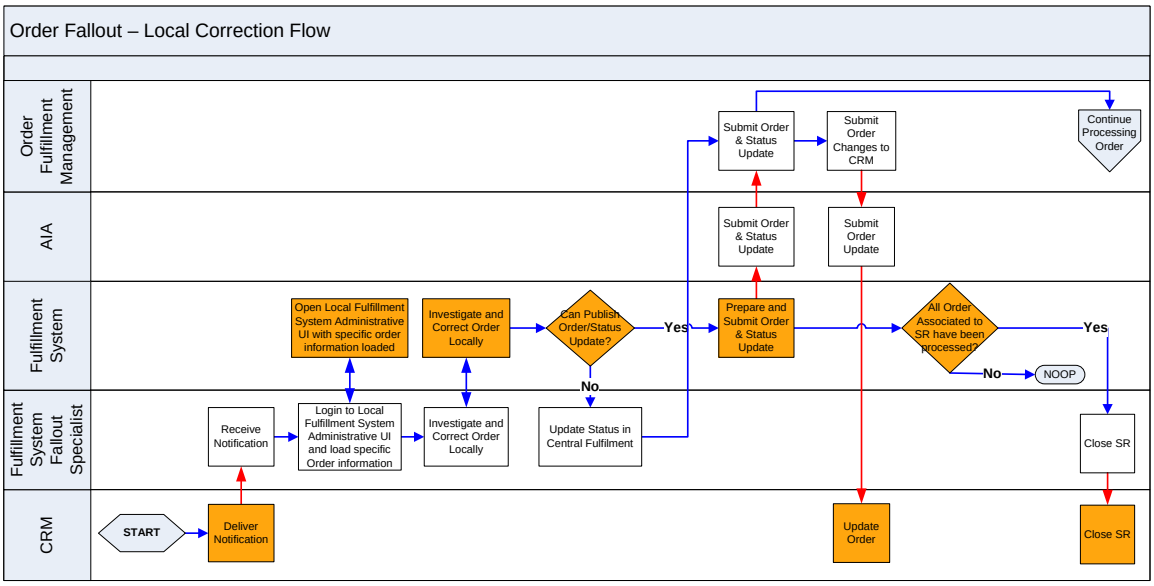



## Siebel CRM correction flow

After the trouble ticket is created in Siebel through Oracle AIA, the request is assigned to a fallout specialist by an assignment rule set in Siebel CRM. The fallout specialist can then log in the system, pick up the trouble ticket from the queue, and resolve the ticket. After the specialist identifies the failure aspects of the order, he or she can create a new order to correct the failed order and then submit it for processing.

If changes were applied in a fulfillment system because of the failed order, then the order fallout specialist may need to first correct the partial processing in a local fulfillment system before resubmitting the order. For example, this is the case when charges are applied to the customer account in the billing system for a failed order. In this case the charges first need to be undone locally to avoid double billing for the services ordered by the client.

This diagram illustrates the local correction flow subprocess within the order fallout management process that will need to take place to undo, compensate, or otherwise fix changes that were committed locally within a fulfillment system for a failed order:




Local correction flow

# Solution Assumptions and Constraints

These are the assumptions and constraints for the process integration for order fallout management:

1. The order fallout management functionality manages orders that failed after being submitted by Siebel CRM.
2. One trouble ticket is created in Siebel CRM for every fault message notification. The process flow must make sure that no multiple notifications are generated for the same order failure.

# Creating Additional Listeners to Capture Order Failure Notifications

The order fallout management solution leverages the existing Oracle AIA Error Handling Framework to capture order failure notifications when an ABCS or an AIA service ends due to error.

A fault message is created when an order fails in an AIA service, an ABCS, or in the fulfillment system. The fault message is enhanced with additional information to capture pertinent data about the order failure.

The messages used by the Oracle AIA Error Handling Framework to capture the errors must be extended to capture order failure information. The following two tables describe additional fields that must be added to the AIA error handling messages to capture order fallout information.

**For more information** about extending error handling, see *Oracle Application Integration Architecture – Foundation Pack 2.5: Integration Developer's Guide,* "Configuring Oracle AIA Processes for Error Handling and Trace Logging," Extending Error Handling.

If a fault happens within Oracle AIA, the fault message will have all the required details of the failed order and will not require additional enrichment by the Error Handling Framework. In this case, the common error handler will stamp the correlation ID to the fault message and publish it to the Error Topic (JMS Correlation ID is set to the value indicated in the business services repository (BSR) Error Notifications table) so that it can be uniquely identified as an order fallout fault message.

**For more information** about extending fault messages, see *Oracle Application Integration Architecture – Foundation Pack 2.5: Integration Developer's Guide,* "Configuring Oracle AIA Processes for Error Handling and Trace Logging," Extending Fault Messages.

This table shows the order header-level data that is passed from a fulfillment system or AIA service to the Order Fallout Management functionality over the Oracle AIA Error Handling Framework (order header-level fields):

| Field Name | Type | Description | Source | Optional |
|---|---|---|---|---|
| Order Originating System Code | ID | The system code of the Siebel CRM system from which the order was placed. It is needed to cross-reference the IDs back to the appropriate Siebel CRM IDs. | AIA service | No |
| Sales Order Number | Alphanumeric | Alphanumeric identifier for the sales order number (Siebel CRM value). | Siebel CRM | Yes |
| Sales Order Revision Number | Numeric | Numeric field storing the sales order number (Siebel CRM value). | Siebel CRM | Yes |
| SalesOrderID | ID | Siebel CRM Sales Order ID. Needed to create trouble tickets for the orders that fail even before hitting the central fulfillment system. | Siebel CRM | Yes |
| Account Name | AlphaNumeric | AlphaNumeric value identifying the Siebel CRM account name. | Siebel CRM | Yes |
| Account ID | ID | Siebel CRM Account ID. Needed to create trouble tickets for the orders that fail even before hitting the central fulfillment system. | Siebel CRM | Yes |
| SalesOrderID (Common) | ID | Common Order ID. (Required when Oracle AIA creates the trouble tickets). | AIA service | No |
| AccountID (Common) | ID | Common Account ID. | AIA service | Yes |
| Order ID | ID | Alphanumeric identifier for the order. Assigned by fulfillment system to the order. The fulfillment system uses it to | FulfillmentSystem | No |

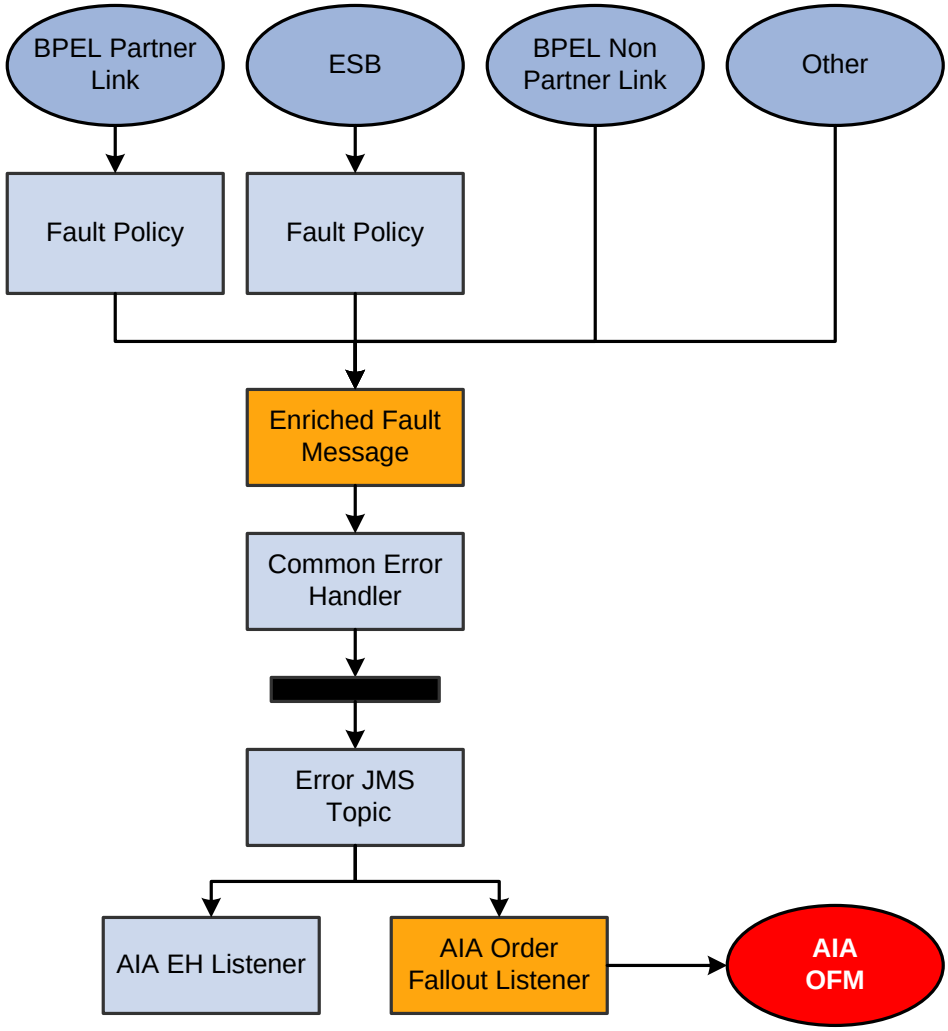
| Field Name | Type | Description | Source | Optional |
|---|---|---|---|---|
| | | correlate the order back to the common order ID received for the original order. The common order ID is then mapped to the Siebel order ID by the Siebel ABCS. | | |
| Order Number | AlphaNumeric | User-friendly identifier for the order in the fulfillment system. | Fulfillment System | Yes |
| ProductID | AlphaNumeric | Alphanumeric identifier for the product used for the failed line or the product for the first order line in case of multiple line failures. | Siebel CRM or AIA service | Yes |
| Fulfillment System of Failure for Order | LOV | Part of the enterprise business object (EBO) header. Set to the Fulfillment System in which the order failed.<br>The AIA identifier for the Fulfillment System will be used. | Fulfillment system of Failure or AIA service | No |
| Service of Failure / FailureSubSystem | LOV | Identifies the AIA service, web service, application programming interface (API), or SubSystemCode (if available) where the order failed. | Fulfillment System of Failure or AIA service | Yes |
| Message | Alphanumeric | Used for the message (error, warning, or other). It can also be used to return notification to customers or other systems.<br>Not to be confused with the original input order message. | Fulfillment System of Failure or AIA Service | Yes |
| Error Code | Alphanumeric | Used to return the error code from the downstream fulfillment system (if any). | Fulfillment System of Failure or AIA service | No |
| Error Severity | LOV | Used to return the error severity from the downstream fulfillment system (if any). | Fulfillment System of Failure or AIA service | Yes |
| Processing Number | ID | Identifier of the job ID assigned in case of batch or bulk orders. | Siebel CRM | Yes |
| Processing Type Code | Code | Code to identify the job type. | Siebel CRM | Yes |
| Processing Quantity | Quantity | Job cardinality - Total number of orders within the job. | Siebel CRM | Yes |

**For more information** about how to pass this information, see Appendix C: Order Fallout: Guidelines for Ensuring That AIA Processes Are Compliant.

This table shows the order fallout information passed from a fulfillment system or AIA service to the order fallout management functionality over the Oracle AIA Error Handling Framework (order-line item-level fields). This is supplied only if the AIA service or the fulfillment system identifies a particular order line item as responsible for the order failure. In the case of system faults caused by network issues or system unavailability, the order lines may not actually add value to the trouble ticket and in those cases you need not populate these fields.

| Field Name | Type | Description | Source | Optional |
|---|---|---|---|---|
| Order Line Item ID | ID | Unique identifier for the order item. | Siebel CRM | No |
| Message | Alphanumeric | Used for error message. It can also be used to return notification to customers or other systems. | Fulfillment system of failure or AIA service. | Yes |
| Error Code | Alphanumeric | Used to return the error code from the downstream fulfillment system (if any). | Fulfillment system of failure or AIA service. | No |
| Error Severity | Alphanumeric | Used to return the error severity from the downstream fulfillment system (if any). | Fulfillment system of failure or AIA service. | Yes |
| StatusContext | LOV | Used to capture status-related display information or status-related information that is product-dependent. It can also be used to capture the current milestone within the provisioning system for the service associated with the order item. | Fulfillment system of failure or AIA service. | Yes |
| FailureSubSystem Code | LOV | Subsystem code or API where the order line has failed. Applicable in the case of participating applications. If the fault is within Oracle AIA, the service which faulted is assumed as the subsystem of failure. | Fulfillment system of failure or AIA service. | Yes |

This diagram shows how the existing Oracle AIA error-handling infrastructure is leveraged to submit an order failure notification to the order fallout management functionality within Oracle AIA:




Fault message for order failure: creation, submission, and delivery

This is a high-level description of the flow:

1. The fault message containing the failed order information is created and submitted within an AIA service (EBS or application business service (ABS)). If the order fails within a fulfillment application, this returns an error to its ABCS, which produces the fault message.
2. The fault message is then submitted to the AIA Common Error Handler, which recognizes that the fault message is related to an order failure and posts it to the AIA Error JMS Topic (AIA_ERROR_TOPIC) with JMSCorrelation set to AIA_ORDERFALLOUT_TTS (as indicated in the ERROR_TYPE column in the BSR_ERROR_NOTIFICATIONS page).
3. The AIA order fallout listener (AIAOrderFalloutJMSBridgeService) picks up the fault message from the AIA Error Topic and pushes it to the Fallout Queue (AIA_ORDERFALLOUT_JMSQ)

4. The AIACOMOrderFalloutNotificationConsumer process picks up the fault message from the Fallout Queue and invokes AIA order fallout services to create the order failure notification within AIA.

**For more information** about extending error handling, see *Oracle Application Integration Architecture – Foundation Pack 2.5: Integration Developer's Guide,* "Configuring Oracle AIA Processes for Error Handling and Trace Logging," Extending Error Handling and Extending Fault Messages.

The overall solution includes:

1. Extending the AIA fault message to be able to capture the additional information identified in the tables described previously.
2. Extending the common error handler to be able to:
   - Identify when a fault message is related to order failures.
   - Stamp the error type in the fault message as a correlation ID and invoke the appropriate fault extension handlers (in case of a partner link fault).
   - Publish to the AIA Error JMS Topic.
3. Creating the AIA order fallout listener (AIAOrderFalloutJMSBridgeService). The order fallout listener (AIAOrderFalloutJMSBridgeService):
   - Listens to all messages published to the AIA Error JMS Topic.
   - Picks up the messages that are specific to order fallout by looking at the correlation ID that contains the error type stamped by the AIA Common Error Handler.
   - Persists the fault message into a fallout queue (AIA_ORDERFALLOUTJMSQ).
4. Creating a listener to the Order Fallout Queue, AIACOMOrderFalloutNotificationConsumer that routes the fault message appropriately to the process integration for order fallout management to create the trouble ticket.

**For more information** about extending error handling, see *Oracle Application Integration Architecture – Foundation Pack 2.5: Integration Developer's Guide*, , "Configuring Oracle AIA Processes for Error Handling and Trace Logging," Extending Error Handling.

## Exception Handling

The types of operation conducted by the AIA Order Fallout Listeners are quite straightforward; thus, the exception handling is also straightforward: If an error occurs while the listeners are preparing the message for the invocation of the AIA service, then standard Oracle AIA Error Handling Framework notification is posted to the Oracle AIA Error Handling Framework.

# Creating Trouble Tickets in Siebel CRM by Oracle AIA

After the Order Fallout Listener (AIACOMOrderFalloutNotificationConsumerProcess) picks up the fault message from the AIA Error JMS Topic based on an error notification from a downstream system or AIA service that has ended due to an error, an AIA Requestor Service provides an interface to invoke an EBS for the creation of trouble tickets in Siebel.

The implementation of this feature enables Oracle AIA to invoke the TroubleTicketEBS so that a trouble ticket can be created.

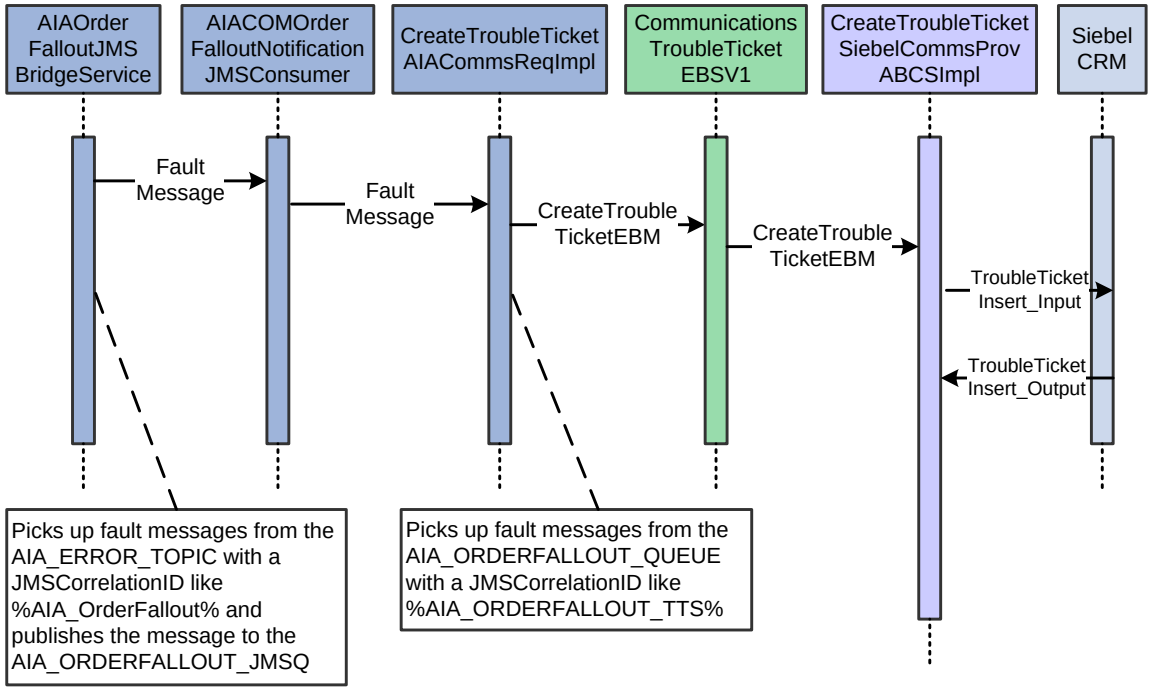
This feature is composed of the following services:

- An AIA Requestor ABCS (CreateTroubleTicketAIACommsReqImpl).
- An AIA Provider ABCS (Create TroubleTicketSiebelCommsProvABSCImpl) that will be invoked by the AIA Trouble Ticket EBS to create a trouble ticket in Siebel.
- An AIA EBS (CommunicationsTroubleTicketEBSV1) that is invoked to call the Siebel Provider ABCS to create the trouble ticket.

On an error, the order fallout process (detection) within Oracle AIA passes the order fault message that is queued in the AIA Error JMS Topic to the CreateTroubleTicketAIACommsReqImpl ABCS. The service then invokes the CommunicationsTroubleTicketEBSV1, which routes the AIA message to the Siebel provider, which in turn calls the Siebel web service to create the trouble ticket in Siebel.

## Creating a Trouble Ticket in Siebel CRM

This is the create trouble ticket sequence diagram:




Create trouble ticket sequence diagram

This flow creates a trouble ticket and has the following set of activities:

1. The enriched fault messages that contain the details of the order are pushed to the AIA_ERROR_TOPIC using the Oracle AIA Error Handling Framework. These messages are stamped with a JMS Correlation ID = AIA_ORDERFALLOUT_TTS in case the trouble tickets are created by Oracle AIA directly, based on the ERROR_TYPE set in the Business Services Repository (BSR) Error Notifications page.
2. The AIAOrderFalloutJMSBridgeService picks up the messages with the JMSCorrelationID such as AIA_ORDERFALLOUT (AIA_ORDERFALLOUT_TTS in this case) and publishes them to the AIA_ORDERFALLOUT_JMSQ JMS Queue.
3. The AIACOMOrderFalloutNotificationJMSConsumer picks up the messages stamped with the JMS Correlation ID AIA_ORDERFALLOUT_TTS from the AIA_ORDERFALLOUT_JMSQ.
4. The AIACOMOrderFalloutNotificationJMSConsumer invokes the CreateTroubleTicketAIACommsReqImpl.
5. The CreateTroubleTicketAIACommsReqImpl service parses the fault message, prepares the CreateTroubleTicketEBM, and invokes the CommunicationsTroubleTicketEBSV1 with the CreateTroubleTicket operation.
6. The EBS routes the message to the CreateTroubleTicketSiebelCommsProvABCSImpl.
7. The CreateTroubleTicketSiebelCommsProvABCSImpl synchronously invokes the Siebel web service (SWITroubleTicketIO.wsdl: SWITroubleTicketInsert) and the response trouble ticket ID is received in the form of SWITroubleTicketInsert_Output message. This application business message (ABM) is transformed to the CreateTroubleTicketResponseEBM depending on the Response Code set in the EBM.

## Defining Transaction Boundaries and Recovery Details

For this flow there are two transaction boundaries. The following table describes the transactions involved, the database operations, and what actions to take in case of an error.

**For more information** about system errors and business errors, see Using Error Type to Control Response to Order Fallout.

The following services are involved:

- AIAOrderFalloutJMSBridgeService
- AIACOMOrderFalloutNotificationJMSConsumer
- CreateTroubleTicketAIACommsReqImpl
- CommunicationsTroubleTicketEBSV1
- CreateTroubleTicketSiebelCommsProvABCSImpl

| Transaction | DB Operations | In Case of Error | Recovery |
|---|---|---|---|
| The AIAOrderFalloutJMSBridgeService picks up the messages with the JMSCorrelationID and publishes to AIA_ORDERFALLOUT_JMSQ. | Message enqueued in AIA_ORDERFALLOUT_JMSQ. | Rollback JMS message to AIA_ERROR_TOPIC | Resubmit from AIA_ERROR_TOPIC. |

| Transaction | DB Operations | In Case of Error | Recovery |
|---|---|---|---|
| AIACOMOrderFalloutNotificationJMSConsumer picks up messages with the JMS Correlation ID AIA_ORDERFALLOUT_TTS and invokes CreateTroubleTicketAIACommsReqImpl, which parses fault message and invokes CommunicationsTroubleTicketEBSV1. The EBS routes the message to CreateTroubleTicketSiebelCommsProvABCSImpl. | AIA cross-reference entries. | Rollback the message to AIA_ORDERFALLOUT_JMSQ. | Resubmit from AIA_ORDERFALLOUT_JMSQ. |

**For more information** about rollback procedures, see For more information about rollback procedures, see *Oracle Fusion Middleware Developer's Guide for Oracle Application Integration Architecture Foundation Pack*, "Configuring Oracle AIA Processes for Error Handling and Trace Logging", Configuring Fault Policies to Not Issue Rollback Messages.

### Exception Handling

These are the exception handling notes for creating trouble tickets in Siebel CRM:

- If validation of the message fails because of missing mandatory data, incorrect formatting, or other problems, then an error message identifying the validation issue is returned to the invoking application.
- In case of any errors in the flow, a standard AIA Error Handling Framework notification is posted to the Oracle AIA Error Handling Framework.

## Siebel CRM Interfaces

The process integration for order fallout management uses this Siebel CRM interface:

- SWI Trouble Ticket Service: This service is invoked by the Siebel ABCS to create or update a trouble ticket in Siebel. If the request is for creating a new trouble ticket, a new trouble ticket is created and the trouble ticket number is returned. If the request is to update a particular trouble ticket, typically to close the trouble ticket, the trouble ticket is updated.

**For more information,** see the *Siebel Order Management Guide Addendum for Communications*, “Web Services Reference.”

## Industry AIA Components

The process integration for order fallout management uses these industry components:

- TroubleTicketEBO
- CreateTroubleTicketEBM
- CreateTroubleTicketResponseEBM

- CommunicationsTroubleTicketEBSV1.wsdl

The industry EBO and EBM XML schema (XSD) files are located here: http://<server name>:<port number>/AIAComponents/EnterpriseObjectLibrary/Industry/Communications/EBO/TroubleTicket/V1/

The industry EBS web service description language (WSDL) files are located here: http://<server name>:<port number>/AIAComponents/EnterpriseBusinessServiceLibrary/Industry/Communications/EBO/TroubleTicket/V1/

For detailed documentation about individual EBOs, click the EBO Name link on the Integration Scenario Summary page of the Oracle AIA Console. You can also use the Integration Scenario Summary page to search for and view integration scenarios that use a particular EBO or EBS.

**For more information,** see *Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide*, "Using the BSR UI to View Integration Scenarios."

EBOs can be extended, for instance, to add new data elements. These extensions are protected, and they will remain intact after a patch or an upgrade.

**For more information,** see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Extensibility for AIA Artifacts."

**For more information** about the changes made for each EBO and EBM, see the appendix in the My Oracle Support document number 988374.1.

# Integration Services

These services are delivered with this integration:

- CommunicationsTroubleTicketEBSV1
- CommunicationsTroubleTicketResponseEBSV1
- CreateTroubleTicketSiebelCommsProvABCSImpl
- AIAOrderFalloutJMSBridgeService
- AIACOMOrderFalloutNotificationJMSConsumer
- CreateTroubleTicketAIACommsReqImpl
- AIAOrderFalloutErrorHandlerExtension.java

Some of these services have been enabled to use Session Pool Manager.

**For more information** about Session Pool Manager, see Appendix H: Using Session Pool Manager.

Use the Integration Scenario Summary page in the Oracle AIA Console to search for and view integration scenarios that use a particular ABC service.

**For more information,** see *Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide*, “Using the BSR UI to View Integration Scenarios.”

## CommunicationsTroubleTicketEBSV1

The CommunicationsTroubleTicketEBSV1 service is implemented as an enterprise service bus (ESB) routing service. It provides the basic request operations that can be performed against the TroubleTicketEBO. This service is invoked as part of the create trouble ticket flow. It has one operation: CreateTroubleTicket

**For more information** about this EBS, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer’s Guide*, “Designing and Developing EBSs” and *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, “Understanding EBSs.”

## CommunicationsTroubleTicketResponseEBSV1

The CommunicationsTroubleTicketResponseEBSV1 service is implemented as an ESB routing service. It provides the basic request operations that can be performed against the TroubleTicketEBO. This service is invoked as part of the create trouble ticket flow. It has one operation: CreateTroubleTicketResponse.

**For more information** about this EBS, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer’s Guide*, “Designing and Developing EBSs” and *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, “Understanding EBSs.”

## CreateTroubleTicketSiebelCommsProvABCSImpl

The CreateTroubleTicketSiebelCommsProvABCSImpl is implemented as an asynchronous business process execution language (BPEL) process. This service takes CreateTroubleTicketEBM as the input. It invokes the Siebel web service to create the trouble ticket and after the trouble ticket is created in Siebel, the trouble ticket ID is passed back to this service.

This service acts as the provider for the CreateTroubleTicket operation of the CommsTroubleTicketEBS. When complete, this service invokes the CreateTroubleTicketResponse operation of the CommuinicationsTroubleTicketResponseEBS.

This process acts both as a fire-and-forget one-way flow or a request response flow depending on a couple of configurable parameters. CreateTroubleTicketSiebelCommsProvABCSImpl creates a trouble ticket response message (creates a cross-reference for the trouble ticket ID with the Siebel ID) and invokes the CommunicationsTroubleTicketResponseEBSV1 if the property TroubleTicket.GenerateTroubleTicketResponse is set to True or if the response code attribute (CreateTroubleTicketEBM/DataArea/Create/@responseCode) is not null. Otherwise, this service just acts as a fire-and-forget flow and ignores the response.

This service is now SPM (Session Pool Manager) enabled.

**For more information** about SPM, see Appendix H: Using Session Pool Manager.

### AIAOrderFalloutJMSBridgeService

The AIAOrderFalloutJMSBridgeService service is an ESB service that picks up the fault message from the AIA Error Topic and publishes the message to the AIA_ORDERFALLOUT_JMSQ. This service is introduced to persist the enhanced fault message into a fallout queue and retry in case of errors in the downstream process. The message can either be picked from this queue by Oracle AIA to directly create a trouble ticket in Siebel CRM or to send an order failure notification to Oracle Order and Service Management Central Fulfillment System (Oracle OSM CFS).

### AIACOMOrderFalloutNotificationJMSConsumer

The AIAOrderFalloutNotificationJMSConsumer service is implemented as an ESB service and picks up the fault message from the AIA Error Topic. The fault message is passed to the CreateTroubleTicketAIACommsReqImpl process. This service acts as the consumer, listening to the messages produced in the AIA Error Topic.

### CreateTroubleTicketAIACommsReqImpl

The CreateTroubleTicketAIACommsReqImpl service is implemented as a one-way asynchronous BPEL process. This service picks up the fault message from the AIACOMOrderFalloutNotificationJMSConsumer. The fault message is parsed and then the CreateTroubleTicketEBM is constructed.

It invokes the CreateTroubleTicket operation of the CommunicationsTroubleTicketEBSV1.

### AIAOrderFalloutErrorHandlerExtension - Java Class

This module is the Java action that is specified for enhancing the fault message. In case of a Java action in the bpel/esb fault policy, the control is handed to this application module to enrich the fault message with business-specific content. The enriched fault message is returned to the AIA Error Handling Framework Common Error Handler.

- oracle.apps.aia.industry.comms.eh.AIAOrderFalloutErrorHandlerExtension.java implements oracle.apps.aia.core.eh. IAIAErrorHandlerExtension interface.
- This class implements the IAIAErrorHandlerExtension interface, which has two methods exposed: one for treating a BPEL fault and the other for an ESB fault.

This class constructs the ApplicationFaultData element of the fault message with the order-related details.

## Order to Bill Fallout Services

These Order to Bill services are fallout-enabled:

1. ProcessFulfillmentOrderBillingBRMCommsAddSubProcess
2. ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess
3. ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess
4. ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl

5. ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess
6. ProcessFulfillmentOrderBillingBRMCommsUpdateSubProcess
7. CommsProcessFulfillmentOrderBillingAccountListEBF
8. CommsProcessBillingAccountListEBF
9. QueryCustomerPartyListSiebelProvABCSImplV2
10. EBS.CommunicationsCustomerPartyResponseEBSV2
11. SyncCustomerPartyListBRMCommsProvABCSImpl
12. EBS.CustomerPartyEBSV2
13. EBS.CommunicationsCustomerPartyEBSV2
14. EBS.CommunicationsCustomerPartyEBSV2Resequencer

**For more information** about Order to Activate services enabled for fallout, see the *Oracle Order to Activate Integration Pack for Siebel CRM and Oracle Communications Order and Service Management - Implementation Guide*, “Order to Activate Fallout Services”.

# Chapter 9: Configuring the Process Integration for Order Fallout Management

This chapter discusses how to:

- Set up Oracle Application Integration Architecture (Oracle AIA).
- Set up Fusion Middleware (FMW).
- Work with domain value maps (DVMs).
- Work with cross-references.
- Handle error notifications.
- View enterprise business object (EBO) implementation maps (EIMs).
- Configure the process integration for order fallout management.

## Setting Up Oracle AIA

- The installation preseeds the services that participate in the Oracle Order Fallout Framework in the BSR_ERROR_NOTIFICATIONS table.

**For more information** about how to update the seeded data in the BSR_ERROR_NOTIFICATIONS table, see Using Error Type to Control Response to Order Fallout.

- The SystemType for the applications configured in the BSR SYSTEMS table must match the COMMON value of the TROUBLETICKET_AREA DVM.

## Setting Up FMW

Perform these steps to set up Fusion Middleware:

1. Add the -DHTTPClient.disableKeepAlives=true property in the opmn.xml file. The opmn.xml file is available here: SOA_HOME/opmn/conf/opmn.xml.

   After you add this property, service oriented architecture (SOA) does not use the same transmission control protocol (TCP) connection more than once to call a Siebel web service. Simultaneous calls to the same Siebel web service cause errors if the same TCP connection is used. This property should be added in the startup parameters of oc4j_soa. After you add the property, the opmn.xml file looks like this:

```
<process-type id="oc4j_soa" module-id="OC4J" status="enabled">
               <module-data>
                  <category id="start-parameters">
                     <data id="java-options" value="-server -
```

```
Xmx2048M -Xms2048M -XX:MaxPermSize=1024M -XX:MaxNewSize=614m -
XX:NewSize=614m -XX:AppendRatio=3 -XX:SurvivorRatio=6 -
Djava.security.policy=$ORACLE_HOME/j2ee/oc4j_soa/config/java2.policy
-Djava.awt.headless=true -Dhttp.webdir.enable=false -
Doraesb.home=/slot/ems1936/oracle/product/10.1.3.1/OracleAS_1/integr
ation/esb -Dhttp.proxySet=false -Doc4j.userThreads=true -
Doracle.mdb.fastUndeploy=60 -Doc4j.formauth.redirect=true -
Djava.net.preferIPv4Stack=true -
Dorabpel.home=/slot/ems1936/oracle/product/10.1.3.1/OracleAS_1/bpel
-
Xbootclasspath^/p:/slot/ems1936/oracle/product/10.1.3.1/OracleAS_1/b
pel/lib/orabpel-boot.jar -Dhttp.proxySet=false -
Daia.home=/slot/ems1936/oracle/product/AIA_HOME" -
DHTTPClient.disableKeepAlives=true/>
                  </category>
                  <category id="stop-parameters">
                     <data id="java-options" value="-
Djava.security.policy=$ORACLE_HOME/j2ee/oc4j_soa/config/java2.policy
-Djava.awt.headless=true -Dhttp.webdir.enable=false" />
                  </category>
               </module-data>
               <start timeout="600" retry="2" />
               <stop timeout="120" />
               <restart timeout="720" retry="2" />
               <port id="default-web-site" range="12501-12600"
protocol="ajp" />
               <port id="rmi" range="12401-12500" />
               <port id="rmis" range="12701-12800" />
               <port id="jms" range="12601-12700" />
               <process-set id="default_group" numprocs="1" />
            </process-type>
```

***Warning:*** While adding the -DHTTPClient.disableKeepAlives=true property in the opmn.xml file, you must be careful. The opmn.xml file is required for the SOA server startup and any error in this file may cause the server to fail.

2. To prevent multiple retries by the transaction manager:

   Modify the file $ORACLE_HOME/j2ee/{oc4j_instance name}/config/transaction-manager.xml to change the property value 'retry-count' of commit-coordinator from 4 to 0.
   <commit-coordinator retry-count="4"> to <commit-coordinator retry-count="0">

3. To prevent the retries by the enterprise service bus (ESB):

Overwrite the ESB retry entries to disable the retries in the orion-application.xml file present in the directories:

$ORACLE_HOME/j2ee/oc4j_soa/application-deployments/esb-rt and

$ORACLE_HOME/j2ee/oc4j_soa/application-deployments/esb-dt with the following values after backing up the original file.

```
<property name="InboundRetryCount" value="0" />
<property name="InboundRetryInterval" value="0" />
<property name="InboundRetryEnabled" value="false" />
<property name="OutboundRetryCount" value="0" />
<property name="OutboundRetryInterval" value="0" />
<property name="OutboundRetryEnabled" value="false" />
```

Alternatively, change these property values in the $ORACLE_HOME//integration/esb/config/esb_config.ini to change the retry values

These steps are also included in the *Oracle AIA Installation and Upgrade Guide*.

# Working with DVMs

Domain value maps (DVMs) are a standard feature of the Oracle SOA Suite. They enable you to equate lookup codes and other static values across applications, for example, FOOT and FT or US and USA.

DVMs are static in nature, though administrators can add additional maps as needed. Transactional business processes never update DVMs; they only read from them. DVMs are stored in XML files and cached in memory at run time.

DVM types are seeded for the order fallout flows, and administrators can extend the list of mapped values by adding more maps. The DVM data should be synchronized with what the participating applications use.

These are the DVMs for the process integration for order fallout:

| DVM | Description |
|---|---|
| TROUBLETICKET_AREA | DVM to map the Area of the trouble ticket<br>SEBL_01 column maps to the Area element in Siebel.<br>COMMON column points to the SystemCode column of the corresponding system in the Business Service Repository (BSR) Systems page. |
| TROUBLETICKET_SUBAREA | DVM to map the SubArea of the trouble ticket.<br>SEBL_01 column maps to the Sub-Area element in Siebel.<br>COMMON column points to the appropriate FailureSubSystemCode or the faulting service. |
| TROUBLETICKET_STATUS | DVM to map the status of the trouble ticket.<br>SEBL_01 column maps to the Status element in Siebel.<br>COMMON column maps to the appropriate status in Oracle AIA. |
| TROUBLETICKET_SEVERITY | DVM to map the severity of the trouble ticket.<br>SEBL_01 column maps to the Severity element in Siebel.<br>COMMON column maps to the appropriate severity (1-5) in Oracle AIA. |

| DVM | Description |
|---|---|
| TROUBLETICKET_PRIORITY | DVM to map the recovery priority of the trouble ticket.<br>SEBL_01 column maps to the Priority element in Siebel.<br>COMMON column maps to the appropriate priority (1-4) in AIA. |

**For more information,** see *Oracle Application Integration Architecture – Foundation Pack 2.5: Integration Developer's Guide*, "Understanding Message Transformation, Enrichment, and Configuration," DVMs.

# Working with Cross-References

Cross-references map and connect the records within the application network, and they enable these applications to communicate in the same language. The integration server stores the relationship in a persistent way so that others can refer to it.

This table lists the order fallout cross-reference:

| Cross-reference Table Name | Column Names Column Values | | Description |
|---|---|---|---|
| TROUBLETICKET_TROUBLETICKETID | COMMON | SEBL_01 | The trouble ticket ID returned by the Siebel web service is cross-referenced to the BusinessComponentID of the TroubleTicket Response enterprise business message (EBM).<br>The idea is to use this cross-referenced value for making any updates to this trouble ticket. So this cross-referencing is done only when the response is sought from the process CreateTroubleTicketSiebelCommsProvABCSImpl |
| | CreateTroubleTicketResponseEBM/DataArea/CreateTroubleTicketResponse/Identification/BusinessComponentID stores this value.<br>A randomly generated ID is used as the COMMON value for the trouble ticket and referenced with the Siebel value. | The row ID for the trouble ticket created in Siebel, which is returned in the ListOfSWITroubleTicketIO/TroubleTicket/Id element of the response of the web service, is cross-referenced. | |

# Handling Error Notifications

Based on the roles defined for the services, email notifications are sent if a service ends due to an error.

The following localized custom errors are caused by the order fallout management services for data insufficiency for creating a trouble ticket:

| Error Code | Message Text |
|---|---|
| AIA_ERR_AIACOMOFMPI_0001 | Data Insufficient for Trouble Ticket Creation. Order Originating System Code not available. |

| Error Code | Message Text |
|---|---|
| AIA_ERR_AIACOMOFMPI_0002 | Data Insufficient for Trouble Ticket Creation. Order ID not available. |

**For more information** about the errors caused by Siebel CRM or Oracle BRM, see the documentation for that product.

**For more information** about Oracle AIA error handling, see the *Oracle Application Integration Architecture – Foundation Pack 2.5: Core Infrastructure Components Guide,* "Understanding Error Handling and Logging."

## Describing Delivered Error Notification Roles and Users

The following roles and users are delivered as default values for issuing error notifications for the process integration for order fallout management.

Actor roles and users:

- Role: AIAIntegrationAdmin. User: AIAIntegrationAdminUser.

The default password set for all users is welcome1.

**For more information** about the errors caused by Siebel CRM or Oracle BRM, see the documentation for that product.

**For more information** about AIA error handling, see the *Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide,* "Understanding Error Handling and Logging."

## Using Error Type to Control Response to Order Fallout

**Important:** The content contained in this section is applicable only to the Comms 2.5 RUP#3 release.

The ERROR_TYPE column in the AIA Error Notifications table (BSR_ERROR_NOTIFICATION) determines what happens when there is a failure during order processing.

The supported values for ERROR_TYPE are:

- AIA_EH_DEFAULT – Generates the standard Oracle AIA error notification.
- AIA_ORDERFALLOUT_CFS – Results in Oracle AIA notifying an Order Management System or Central Fulfillment System (such as Oracle OSM) regarding the order fallout so that it can create and manage the trouble ticket. This value enables the default Order Fallout handling for the Order to Activate PIP.
- AIA_ORDERFALLOUT_TTS – Results in Oracle AIA creating a trouble ticket for the order fallout. This value enables the default Order Fallout handling for the Order to Bill PIP.

The value AIA_EH_DEFAULT can be combined with the value AIA_ORDERFALLOUT_CFS or AIA_ORDERFALLOUT_TTS, using a comma as the separator. For example, AIA_EH_DEFAULT,AIA_ORDERFALLOUT_CFS results in the actions associated with both the values being triggered.

**Note:** You cannot have both AIA_ORDERFALLOUT_CFS and AIA_ORDERFALLOUT_TTS values specified for a given record.

If the Order to Activate PIP is installed, the seeded value for ERROR_TYPE is AIA_ORDERFALLOUT_CFS. If the Order to Bill PIP is installed alone (without the Order to Activate PIP) the seeded value for ERROR TYPE is AIA_ORDERFALLOUT_TTS. If the Order to Bill PIP is installed with the Order to Activate PIP, the seeded value for ERROR TYPE is AIA_ORDERFALLOUT_CFS.

Different ERROR_TYPE values can be given for different combinations of BPEL and ESB service, business process, system code, and error code. As delivered, AIA seeds these values for all order services. In cases where a service is used in multiple business processes, it is separately seeded for each business process.

In any given order service, there can be two categories of errors:

**Business errors** - A business error is usually due to invalid or incomplete data on the order or missing setup in the end fulfillment system, which results in the request to process an order failing. In this case, for the order to be successfully processed, either the order must be corrected or revised and resubmitted, or the setup in the end fulfillment system must be corrected and the order resubmitted. For this type of error, Order Fallout should be triggered.

This type of error usually happens when an order reaches either the participating or the edge application (such as Oracle BRM). The expectation is that the fault coming from the application is a BPEL error code: "{http://schemas.oracle.com/bpel/extension}bindingFault". Oracle BRM 7.4 returns a bindingFault when it sees a business error in the order.

**All other errors** – This includes system errors. System errors can arise when a certain system (such as Oracle BRM or BRM JCA Adapter) is down. The assumption is that there is actually nothing wrong with the order data itself and once system errors are addressed, the order can be resubmitted without any changes. For these types of errors, Order Fallout should not occur.

Order services are delivered seeded with two entries in the BSR_NOTIFICATIONS table:

- Error Code - "{http://schemas.oracle.com/bpel/extension}bindingFault

  The seeded value for Error Type is either AIA_EH_DEFAULT,AIA_ORDERFALLOUT_TTS or AIA_EH_DEFAULT,AIA_ORDERFALLOUT_CFS. The expected behavior for this case is both standard AIA error notification and order fallout processing occurs.

- Error Code – null or no value

  The seeded value for Error Type is AIA_EH_DEFAULT. The expected behavior for this case is only standard AIA error notification occurs.

This table is an example entry for the ProcessFulfillmentOrderBillingBRMCommsAddSubProcess order service.

| Error Code | Service Name | Error Type | Error Extn Handler |
|---|---|---|---|
| | ProcessFullmentOrderBillingBRMCommsAddSubProcess | AIA_EH_DEFAULT | AIACOM_OFM_EXT |
| {http://schemas.oracle.com/bpel/extension}bindingFault | ProcessFullmentOrderBillingBRMCommsAddSubProcess | AIA_EH_DEFAULT,AIA_ORDERFALLOUT_CFS | AIACOM_OFM_EXT |

If additional error codes are also classified as business errors, you can add new entries into the BSR_NOTIFICATIONS table with the appropriate Error Code value.

**Note:** The Error Extn Handler value for all order service entries must be AIACOM_OFM_EXT. This is required so that the correct information will be in the fallout as well as the standard error notification.

To update ERROR_TYPE for seed data in the Error Notifications table:

1. Open the Application Integration Architecture homepage by logging in to http://<httphost>:<httpport>/AIA
2. Click the Setup link. This automatically direct s you to the Setup > Error Notifications page.
3. Update the Error Type column with the appropriate value for each service for which you want to change the configuration:

   For example, if you want system errors to trigger order fallout, update this column on the respective records to AIA_EH_DEFAULT,AIA_ORDERFALLOUT_TTS. This indicates that if a particular service errors out, a standard Oracle AIA error notification is created and the error message is sent to Oracle AIA for fallout.

4. Click Save to save your changes.
5. Restart FMW.

If you need to perform a bulk update for all of the processes, you can use a SQL script to update the table ERROR_TYPE column in the BSR_ERROR_NOTIFICATIONS table with the appropriate values. See the AIA_CreateOrderFalloutBSRErrorNotificationsData.sql for reference. After the table is updated, you must restart FMW.

**For more information** about setting up error notifications for Oracle AIA process, see *Oracle Application Integration Architecture Foundation Pack 2.5: Core Infrastructure Components Guide,* "Setting Up Error Notifications and Trace Logging."

# Viewing EBO EIMs

**For more information** about how services are mapped, see the My Oracle Support document: EBO Implementation Maps (EIMs) 881022.1.

# Configuring the Process Integration for Order Fallout Management

The process integration for order fallout management provides two business process execution language (BPEL) services with configuration options to create trouble tickets in Siebel Customer relationship Management (Siebel CRM).

Configure the properties for these services in the AIAConfigurationProperties.xml file. It is located here: <aia.home>/config/. All the property values are case-sensitive. All Boolean values are in lowercase.

**Note:** Whenever the AIAConfigurationProperties.xml file is updated, the file must be reloaded for updates to be reflected in the applications or services that use the updated properties. Click the Reload button on the Configuration page in the Oracle AIA Console to perform this reload. Alternatively, you can perform the reload by rebooting the server.

**For more information**, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide*, "Loading Oracle AIA Configuration File Updates."

Here are the settings for the CreateTroubleTicketAIACommsReqImpl service name:

serviceName="{http://xmlns.oracle.com/ABCSImpl/AIA/Industry/Comms/CreateTroubleTicketAIACommsReqImpl/V1}CreateTroubleTicketAIACommsReqImpl

| Property Name | Value/Default Values | Description |
|---|---|---|
| Sender.Default. SystemID | COMMON | This is used only if the request message does not contain the system instance ID. This is always COMMON because this service is triggered by Oracle AIA. |
| Routing.TroubleTicketEBSV1.CreateTroubleTicket.RouteToCAVS | true/false. Default = false. | Controls whether TroubleTicketEBS routes messages to the validation system or to the Provider application business connector service (ABCS) implementation. |
| Routing.TroubleTicketEBSV1.CreateTroubleTicket.CAVS.EndpointURI | http://${http.hostname}:${http.port}/AIAValidationSystemServlet/asyncrequestrecipient | The end point URI of the Composite Application Validation System (CAVS) simulator. |
| Routing.TroubleTicketEBSV1.CreateTroubleTicket.MessageProcessingInstruction.EnvironmentCode | CAVS/PRODUCTION / Any Value<br>Default: PRODUCTION | If CAVS, the message is routed to CAVS.<br>For other values, the message is routed to the Provider ABCS Implementation. |
| TroubleTicket.DefaultSeverity | Any number from 1 to 5<br>Default – 2 | If the fault message does not have any severity associated with it, the default severity is assigned to the fault message and the same is reflected in the trouble ticket. |
| TroubleTicket.DefaultPriority | Any number from 1 to 4<br>Default – 2 | This service by default assigns the recovery priority for the trouble ticket to the value specified in this configuration property. |

Here are the settings for the following CreateTroubleTicketSiebelCommsProvABCSImpl service name:

{http://xmlns.oracle.com/ABCSImpl/Siebel/Industry/Comms/CreateTroubleTicketSiebelCommsProvABCSImpl/V1}CreateTroubleTicketSiebelCommsProvABCSImpl

| Property Name | Value/Default Values | Description |
|---|---|---|
| Default.SystemID | SEBL_01 | Siebel system instance code (defined in the Business Service Repository (BSR)). This is used when the target system cannot be identified from the request message or if the configuration property TroubleTicket.UseDefaultInstance is set to true. |
| ABCSExtension.PreXformEBMtoABMTroubleTicketEBM | true/false<br>Default: false | Value used to determine whether the ABCS should invoke the Extension service (before the enterprise business message (EBM) to application business message (ABM) transformation). |
| ABCSExtension.PostXformABMtoEBMTroubleTicketEBM | true/false<br>Default: false | Value used to determine whether the ABCS should invoke the Extension service (after the ABM to EBM transformation). |
| ABCSExtension.PreInvokeABSSWITroubleTicketIOABM | true/false<br>Default: false | Value used to determine whether the ABCS should invoke the Extension service (PreInvoke Application). |
| ABCSExtension.PostInvokeABSSWITroubleTicketIOABM | true/false<br>Default: false | Value used to determine whether the ABCS should invoke the Extension service (PostInvoke Application). |
| Routing.SWI_spcTrouble_spcTicket_spcService.RouteToCAVS | true/false<br>Default: false | Indicates whether the Partner link SWI_spcTrouble_spcTicket_spcService should be routed to CAVS or the actual application. |
| Routing.SWI_spcTrouble_spcTicket_spcService.CAVS.EndpointURI | http://${http.hostname}:${http.port}/AIAValidationSystemServlet/asyncresponsesimulator | End point URI of the CAVS simulator for this partner link - SWI_spcTrouble_spcTicket_spcService. |
| Routing.SWI_spcTrouble_spcTicket_spcService.SEBL_01.EndpointURI | End point URI of the SEBL_01 Siebel instance | End point URI of the SEBL_01 Siebel instance. |
| Routing.SWI_spcTrouble_spcTicket_spcService.MessageProcessingInstruction.EnvironmentCode | CAVS/PRODUCTION/Any Value | Acts as a reference and is not used in the service. |
| TroubleTicket.GenerateTroubleTicketResponse | true/false<br>Default: false | CreateTroubleTicketSiebelCommsProvABCSImpl creates a trouble ticket response message (creates a cross-reference for the trouble ticket ID with the Siebel ID) and invokes the CommunicationsTroubleTicketResponseEBSV1 if this property is set to true or if the response code attribute is not null. Otherwise, this service acts only as a fire-and-forget flow and ignores the |

| Property Name | Value/Default Values | Description |
|---|---|---|
| | | response. |
| TroubleTicket.UseDefaultInstance | true/false<br>Default: false | If set to true, the target Siebel instance is overwritten to the default instance indicated by the property Default.SystemID.<br>This gives the user an option to create a trouble ticket in a Siebel instance different from the one from where the order was placed. |
| TroubleTicket.SR_TYPE | Order Failure | SR_TYPE to identify that the trouble ticket is for Order Failure. Siebel web service expects this value to be Order Failure for order failure trouble tickets. |
| Routing.TroubleTicketEBSResponseV1.CreateTroubleTicketEBSResponse.RouteToCAVS | true/false<br>Default: false | Indicates whether the ResponseEBS should route the message to CAVS or the designated target service. |
| Routing.TroubleTicketEBSResponseV1.CreateTroubleTicketEBSResponse.CAVS.EndpointURI | http://${http.hostname}:${http.port}/AIAValidationSystemServlet/asyncresponserecipient | End point URI for the CAVS simulator. |
| Routing.TroubleTicketEBSResponseV1.CreateTroubleTicketEBSResponse.MessageProcessingInstruction.EnvironmentCode | CAVS/PRODUCTION / Any Value<br>Default : PRODUCTION | If CAVS, the message is routed to CAVS.<br>For other values, the message is routed to the target service. |

The following fields extract the localized values using the aia:getAIALocalizedString xpath function:

**EBM Field Name: DataArea / CreateTroubleTicket / Description**

Siebel Field Name: Description

ResourceBundle – oracle.apps.aia.core.i18n.AIAListResourceBundle

ResourceBundle Key - TROUBLETICKET_DESCRIPTION

Resource Bundle Value: **SalesOrder - {OrderNumber} # {OrderRevision} for Account {AccountName} failed at {Timestamp}**

**EBM Field Name: EBMHeader/BusinessScope/ID**

Siebel Field Name: Abstract

ResourceBundle – oracle.apps.aia.core.i18n.AIAListResourceBundle

ResourceBundle Key - TROUBLETICKET_ABSTRACT

Resource Bundle Value: **[{Timestamp}] Trouble Ticket for (Sales)Order - {OrderNumber} # {OrderRevision}**

# Appendix A: Order Management: Matrix of MACD Actions Supported Per Billing Product Type

## Table A

The process integration for order management supports the following Move, Add, Change, Disconnect (MACD) line actions for a given product type:

| Product Type | Add | Delete | Suspend | Resume | Update | Move-Add | Move-Delete |
|---|---|---|---|---|---|---|---|
| -- | -- | -- | -- | -- | See Table B for a list of attributes | -- | -- |
| Marketing Bundle ('Promotion' in Siebel, no representation in Oracle Communications Billing and Revenue Management (Oracle BRM) | Yes | Yes | Not Applicable. Does not affect purchased bundle in Oracle BRM. | Not Applicable. Does not affect purchased bundle in Oracle BRM. | Yes | Xref updated to reflect new Siebel Customer relationship Management (Siebel CRM) asset. | Ignored other than to determine original Oracle BRM asset. |
| Service Bundle | Yes | Yes | Yes | Yes | Yes | Yes. Same as UPDATE with respect to communicating changes to line attributes. | Ignored other than to determine original Oracle BRM asset. |
| Service Bundle Component – Billing Subscription product | Yes. Can communicate price/discount override as | Yes | Not supported by either Siebel or Oracle BRM, | Not supported by either Siebel or Oracle BRM, | Yes | Yes. Same as UPDATE with respect to communica | Ignored other than to determine original Oracle |

| Product Type | Add | Delete | Suspend | Resume | Update | Move-Add | Move-Delete |
|---|---|---|---|---|---|---|---|
| (applies to service) | part of this action. | | hence ignored by Billing Integration. | hence ignored by Billing Integration. | | ting changes to line attributes. | BRM asset. |
| Service Bundle Component – Billing Discount product (applies to service) | Yes | Yes | Not supported by either Siebel or Oracle BRM, hence ignored by Billing Integration. | Not supported by either Siebel or Oracle BRM, hence ignored by Billing Integration. | Yes | Yes. Same as UPDATE with respect to communicating changes to line attributes. | Ignored other than to determine original Oracle BRM asset. |
| Service Bundle Component – Billing Item product (applies to service). Example: One-Time charge | Yes. Can communicate price/discount override as part of this action. | Not Applicable, because no asset or purchased product instance is created. | Not Applicable, because no asset or purchased product instance is created. | Not Applicable, because no asset or purchased product instance is created. | * Not Applicable, because no asset or purchased product instance is created. | Not Applicable, because no asset or purchased product instance is created. | Not Applicable, because no asset or purchased product instance is created. |
| Billing Subscription product (applies to account) | Yes. Can communicate price/discount override as part of this action. | Yes | Yes | Yes | Yes | Ignored | Ignored |
| Billing Discount (applies to account) | Yes | Yes | Yes | Yes | Yes | Ignored | Ignored |
| Billing Item product (applies to an account). Example: Penalty charge. | Yes. Can communicate price/discount override as part of this action. | Not Applicable, because no asset or purchased product instance is created. | Not Applicable, because no asset or purchased product instance is created. | Not Applicable, because no asset or purchased product instance is created. | * Not Applicable, because no asset or purchased product instance is created. | Ignored | Ignored |

* If a line is billing-initiated and a revision is processed for a service-level product of type Item, then pricing information and billing dates can change. If a line is billing-initiated and a revision is processed for an account-level product of type item, then Billing Account, Bill Profile, Promotion Reference, Pricing Information, and Billing Dates can change.

# Table B

The process integration for order management communicates changes to billing for the following attributes on change orders (action of UPDATE (for service bundle and its components)) for a given product type.

**Note:** Attributes could be updated on a revision as well as a change order, unless comments indicate otherwise.

| Product Type | Service Acct (transfer) | Billing Account and Billing Profile | Pricing Info | Promotion Ref | Service ID | F&F List Ref | Billing Dates and End Date | Comments |
|---|---|---|---|---|---|---|---|---|
| Marketing Bundle (Promotion in Siebel, no representation in Oracle BRM | NA | Yes<br><br>For Billing Account, see note 4 | NA<br><br>See note 1 | NA | NA | NA | Yes<br><br>See note 3 | The billing interface creates purchased bundle instances under billing accounts in Oracle BRM based on promotion lines. The purchase date on promotion lines is used as the start effective date for the bundle instance.<br><br>When a billing account on a promotion line is updated to a different one, the purchased bundle instance is repointed to the new billing account.<br><br>Billing Profile is irrelevant for promotions.<br><br>Updates to a billing account on a revision or a change order result in the bundle instance being repointed to the new billing account.<br><br>Updates to a purchase date on a revision result in the start effective date on the |

| Product Type | Service Acct (transfer) | Billing Account and Billing Profile | Pricing Info | Promotion Ref | Service ID | F&F List Ref | Billing Dates and End Date | Comments |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | bundle instance being reset. |
| Service Bundle | NA<br>See note 2 | Yes | NA | NA | Yes | NA | NA | No pricing on service bundle itself. Pricing is on service bundle components.<br>A service can be transferred from one account to another provided it is the only service in a balance group and that balance group is not an account-level default balance group.<br>Because this release supports only account-level balance groups, service transfers will fail. |
| Service Bundle Component – Billing Subscription product (applies to service) | NA | NA | Yes | Yes | NA | NA | Yes | When subscription product is bundled in a service bundle.<br>Service Acct, Billing Acct/Billing Profile, & Service Identifier – Integration looks only at service bundle line for these attributes.<br>Promotion reference changing is communicated to billing. The purchased product instance is repointed to the new bundle instance in billing.<br>Billing Dates – Cannot be reset using change orders.<br>Cycle Start and Usage Start dates can be reset using revisions on billing initiation, provided that the dates that were previously set are not current.<br>In two-phase billing, once cycle start and usage start dates have been set using billing initiation, they can be reset using billing fulfillment provided that the dates that were previously set are not |

| Product Type | Service Acct (transfer) | Billing Account and Billing Profile | Pricing Info | Promotion Ref | Service ID | F&F List Ref | Billing Dates and End Date | Comments |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | current.<br>End dates can be updated by change orders (promotion upgrade/downgrades) that change duration for products/discounts. |
| Service Bundle Component – Billing Discount product (applies to service) | NA | NA | NA | Yes | NA | NA | Yes | When discount is bundled in a service bundle.<br>Discount products are not priced.<br>Service Acct, Billing Acct/Billing Profile and Service Identifier – Integration looks only at Service bundle line for these attributes.<br>Promotion reference changing is communicated to billing. The purchased discount instance will be repointed to the new bundle instance in billing.<br>Billing Dates cannot be reset using change orders.<br>Cycle Start and Usage Start dates can be reset using revisions on billing initiation provided that the dates that were previously set are not current.<br>In two-phase billing, once cycle start and usage start dates have been set using billing initiation, they can be reset using billing fulfillment provided that the dates that were previously set are not current.<br>End dates can be updated by change orders (promotion upgrade and downgrades) that change duration for products and discounts. |

| Product Type | Service Acct (transfer) | Billing Account and Billing Profile | Pricing Info | Promotion Ref | Service ID | F&F List Ref | Billing Dates and End Date | Comments |
|---|---|---|---|---|---|---|---|---|
| Service Bundle Component – Billing Item product (applies to service). | NA | NA | Yes | NA | NA | NA | Yes | Example: Onetime Charge.<br>No purchased product instance, hence no change orders.<br>Pricing information, Promotion, and Quantity can be updated only on new purchase revisions.<br>Billing dates or end date – Cannot be reset using change orders.<br><br>In two-phase billing, once billing dates have been set using billing initiation, they can be reset using billing fulfillment provided that the dates that were previously set are not current. |
| Service Bundle Component – Special Rating product (applies to service). | NA | NA | NA | NA | NA | Yes | NA | On a change order, change in list reference results in list values being updated to new values from new F&F list.<br>As delivered, the integration does not check for changes to the Special Rating List reference on revision orders once the list product has been billing-initiated. The assumption is that for wireless services (that F&F is targeted towards) no fulfillment latency exists between provisioning and billing and therefore they will not undergo two-phase billing.<br>See the *Understanding the Process Integration for Product Lifecycle Management* chapter. |

| Product Type | Service Acct (transfer) | Billing Account and Billing Profile | Pricing Info | Promotion Ref | Service ID | F&F List Ref | Billing Dates and End Date | Comments |
|---|---|---|---|---|---|---|---|---|
| Billing Subscription product (applies to account) | NA<br>See comments | Yes | Yes | Yes | NA | NA | Yes | Subscription product that is not bundled into a service bundle.<br>Service Acct – Oracle BRM does not support transferring account-level product from one account to another.<br>Siebel disallows this change.<br>Promotion reference changing is communicated to billing. The purchased product instance is repointed to the new bundle instance in billing.<br>Billing Dates – Cannot be reset using change orders.<br>Cycle start and usage start dates can be reset using revisions on billing initiation provided that the dates that were previously set are not current.<br>In two-phase billing, once cycle start and usage start dates have been set using billing initiation, they can be reset using billing fulfillment provided that the dates that were previously set are not current.<br>End dates can be updated by change orders (promotion upgrade and downgrades) that change duration for products and discounts. |
| Billing Discount (applies to an account) | NA<br>See Comments | Yes | NA | Yes | NA | NA | Yes | When discount is *not* bundled in a service bundle.<br>Discount products are not priced.<br>Service Acct – Oracle BRM does not support transferring account-level products from |

| Product Type | Service Acct (transfer) | Billing Account and Billing Profile | Pricing Info | Promotion Ref | Service ID | F&F List Ref | Billing Dates and End Date | Comments |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | one account to another.<br>Siebel disallows this change.<br>Promotion reference changing is communicated to billing. The purchased discount instance is repointed to the new bundle instance in billing.<br>Billing dates – Cannot be reset using change orders.<br>Cycle start and usage start dates can be reset using revisions on billing initiation provided that the dates that were previously set are not current.<br>In two-phase billing, once cycle start and usage start dates have been set using billing initiation, they can be reset using billing fulfillment provided that the dates that were previously set are not current.<br>End dates can be updated by change orders (promotion upgrade/downgrades) that change duration for products/discounts. |
| Billing Item product (applies to an account). | NA | Yes | Yes | NA | NA | NA | Yes | Example: Penalty. No purchased product instance, hence no MACD.<br>Billing Acct/Profile, Pricing info, Promotion reference, can be changed on revisions.<br>Service Acct – Oracle BRM does not support transferring account-level products from one account to another.<br>Siebel disallows this change. |

| Product Type | Service Acct (transfer) | Billing Account and Billing Profile | Pricing Info | Promotion Ref | Service ID | F&F List Ref | Billing Dates and End Date | Comments |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | Billing dates or end date – Cannot be reset using change orders.<br>Cycle start and usage start dates can be reset using revisions on billing initiation, provided that the dates that were previously set are not current.<br>In two-phase billing, once cycle start and usage start dates have been set using billing initiation, can be reset using billing fulfillment provided that the dates that were previously set are not current. |

## Table B Notes

- Pricing information: This includes selling price, pricing commit type, dynamic discount method, discount amount, and discount percent.
- Because this release supports only account-level balance groups, and Oracle BRM disallows transferring a service pointing to the account-level balance group. This will fail.
- Billing dates: This refers to purchase date, cycle start date, and usage start date.
- With the Oracle BRM 7.4 change: Billing profile change alone, which results in the balance group being repointed to a different bill-info, is not supported. In the case of a nonpaying subordinate account, changing the paying parent (billing account *and* billing profile) is supported.

# Appendix B: Examples of Changing the Paying Parent on Subordinate Accounts

Legend:

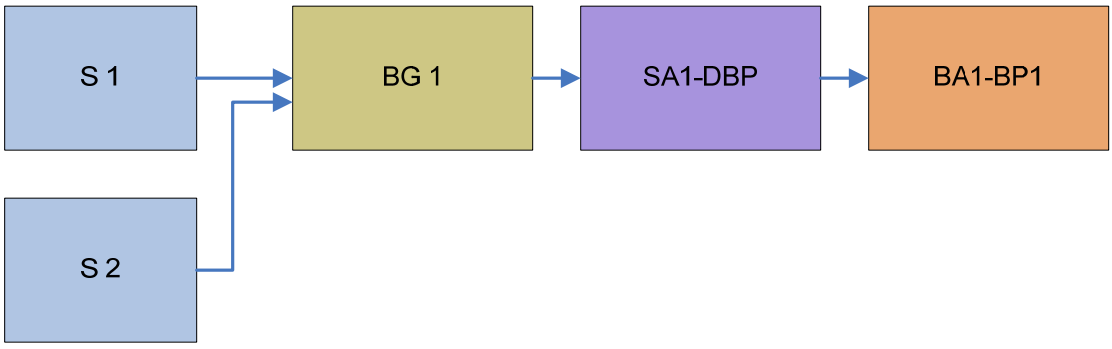
| Abbreviation | Description |
|---|---|
| S | Service |
| SA | Service Account |
| BA | Billing Account |
| BP | Billing Profile |
| BG | Balance Group |
| DBP | Dummy Bill Info |

## 1. Supported

This is the base scenario:

| Action | # | SA | BA | BP |
|---|---|---|---|---|
| ADD | S1 | SA1 | BA1 | BA1-BP1 |
| ADD | S2 | SA1 | BA1 | BA1-BP1 |

This scenario results in the following after the order is processed to billing:



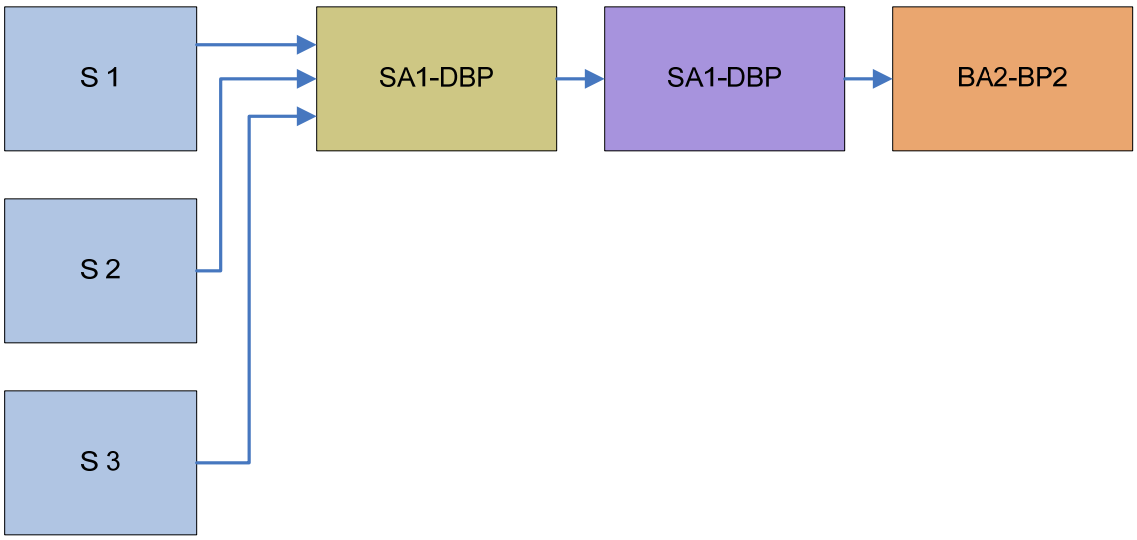

The following is supported, reparenting subordinate to a different parent.

| Action | # | SA | BA | BP |
|---|---|---|---|---|
| UPD | S1 | SA1 | BA2 | BA2-BP2 |
| UPD | S2 | SA1 | BA2 | BA2-BP2 |

This will work even if S2 is not included on the order. However, to keep Siebel Customer Relationship Management (Siebel CRM) assets in sync, S2 should be updated as well. Or like this:

| Action | # | SA | BA | BP |
|---|---|---|---|---|
| UPD | S1 | SA1 | BA2 | BA2-BP2 |
| UPD | S2 | SA1 | BA2 | BA2-BP2 |
| ADD | S3 | SA1 | BA2 | BA2-BP2 |

The preceding scenario results in the following:



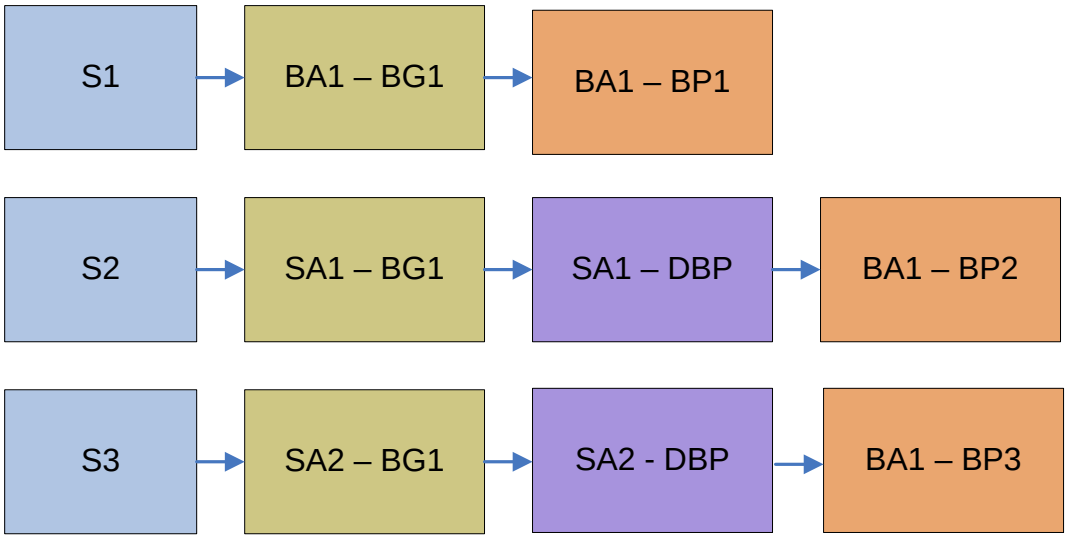

## 2. Supported

This is the base scenario:

A parent using different billing profiles to pay for their account and not each of the child accounts. Note that a parent cannot use more than one billing profile to pay for services of the same child account because the child account can have only one balance group, which can point to only one bill-info.

| Action | # | SA | BA | BP |
|---|---|---|---|---|
| ADD | S1 | BA1 | BA1 | BA1-BP1 |
| ADD | S2 | SA1 | BA1 | BA1-BP2 |
| ADD | S3 | SA2 | BA1 | BA1-BP3 |

The preceding scenario results in the following after the order is processed to billing:

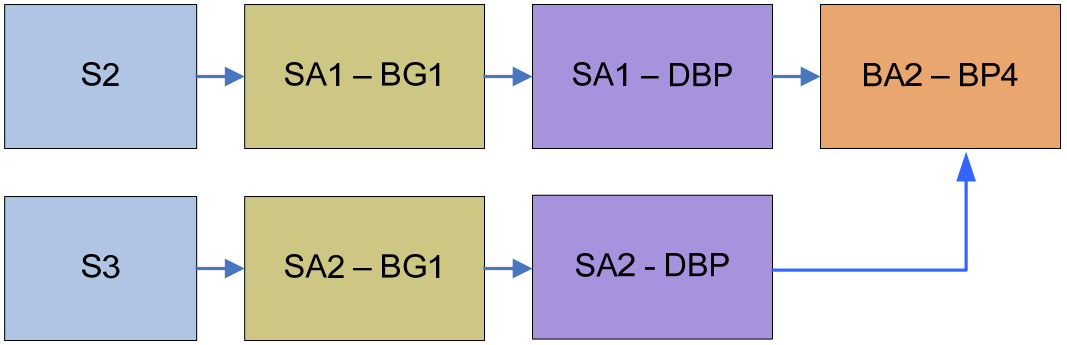



**Note:** That the account-level balance group for the parent account (BA1) references the first billing profile that is created for that account. In the previous scenario it is BP1. If the ADD line for the service purchase for the parent account (BA1) is not the first line on the order, then the account-level balance group references billing profile BP2, and the purchase of S1 fails because it is using BP1.

The following change order is processed, which reparents both the subordinate accounts to a new parent, and the same billing profile under the new parent.

| Action | # | SA | BA | BP |
|---|---|---|---|---|
| UPDATE | S1 | SA1 | BA2 | BA2-BP4 |
| UPDATE | S2 | SA2 | BA2 | BA2-BP4 |

The following is visible in Oracle Communications Billing and Revenue Management (Oracle BRM):

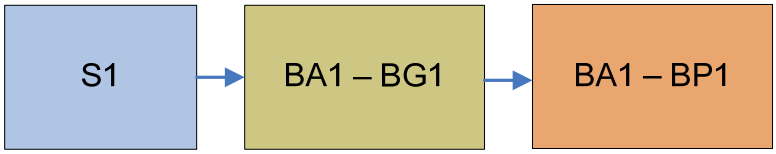



The repointing of the dummy bill-infos to the new billing profile is handled by the services that interface customer data to billing.

Note on alternates:

- A variant in which two different billing profiles under BA2 are used to pay for the different child accounts is also supported.
- A variant in which the billing profile changes but not the paying parent (such as in the base scenario, updating S3 to be paid for by BA1-BP2) is not supported. It will fail in order billing integration because that involves repointing the balance group to a new dummy bill-info (pointing to BA1-BP2), which Oracle BRM does not allow.

This diagram remains unchanged:




## 3. Supported (with caveats)

This is the base scenario:

| Action | # | SA | BA | BP |
|---|---|---|---|---|
| ADD | S1 | SA1 | BA1 | BA1-BP1 |
| ADD | S2 | SA1 | BA1 | BA1-BP2 |

After customer sync, the integration creates two dummy bill-infos:

SA1-DBP1 pointing to BA1-BP1

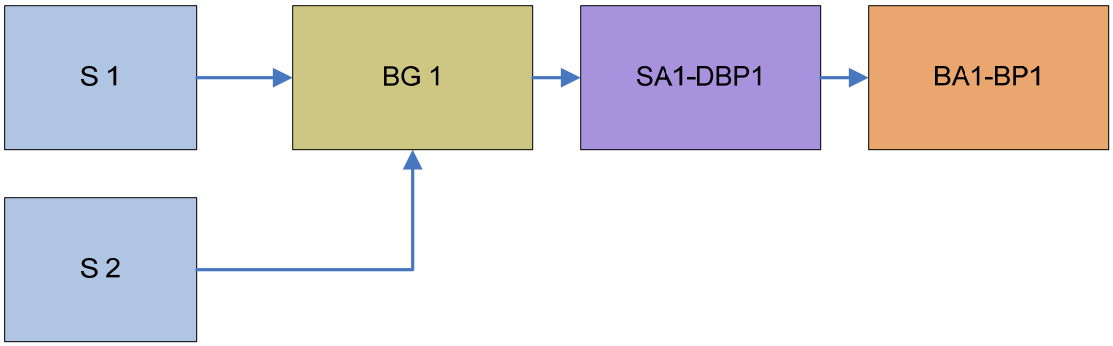
SA1-DBP2 pointing to BA1-BP2

The default account-level balance group points to BA1-BP1. (It points to the billing profile referenced on the first line of the order.)

Order billing integration behaves one of two ways depending on which lines are sent for processing:

- **If both the service bundles are sent for order billing integration at the same time (in the same call):**

  Since OOTB the integration supports a single balance group (and therefore only a single bill-info for a self-paying account), for a given service account. Oracle AIA uses billing account and billing profile on the first service bundle purchase for all the remaining service bundles in the incoming request. The data in Oracle BRM is as follows:




While this transaction is processed successfully, it results in a mismatch between Siebel CRM assets and Oracle BRM in terms of the billing profile and bill-info used to pay for a given service. ***It is therefore recommended, that if you are using OOTB functionality with respect to balance group support, you ensure that orders in Siebel CRM comply with the constraints of a single billing profile for a self-paying account***.

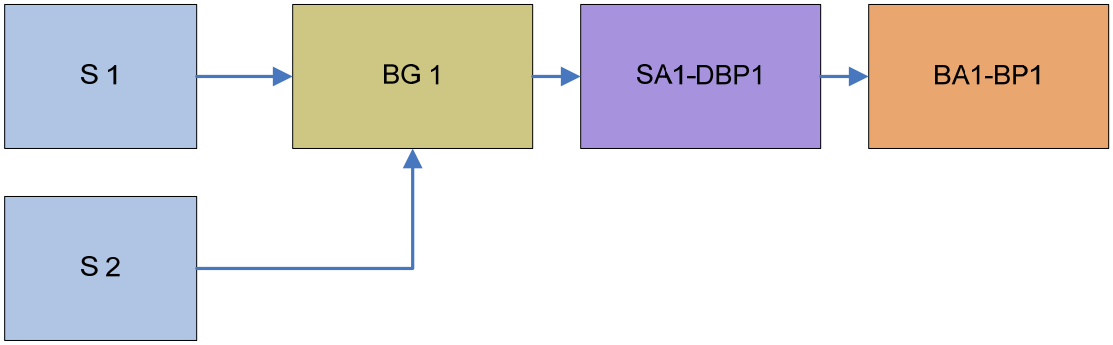
- **If the two service bundles are sent for order billing integration at different times (different calls):**

  Independent of whether service bundle S1 is sent first or second, it is successfully processed. However, order billing integration fails in its attempt to repoint the default account-level balance group SA1-DBP2 when purchasing service bundle S2.

  If the order is resubmitted (through a revision) as follows:

| Action | # | SA | BA | BP |
|---|---|---|---|---|
| ADD | S1 | SA1 | BA1 | BA1-BP1 |
| ADD | S2 | SA1 | BA1 | BA1-BP1 |

The following occurs after the order is processed to billing:



However, you still have the hanging subordinate bill-info SA1-DBP2 that is not used by any service. It will cause a problem in the future (customer synchronization fails; it has validation that finds that a dummy bill profile is not being reparented). If any attempts are made to reparent SA1, because no asset is using SA1-DBP2, it cannot be repointed as part of any reparenting operation.

***No automated workaround is available for this issue. You must manually move SA1 to be under the new parent (thus repointing all the subordinate dummy bill-infos) and then process the order to have assets reflect the new state***.

# Appendix C: Order Fallout: Guidelines for Ensuring That Oracle AIA Processes Are Compliant

This appendix describes the fields and attributes that must be passed to make Oracle AIA processes fallout-compliant.

New services that need to be included to participate in the order fallout notification mechanism must be included in the Business Service Repository (BSR) Error Notifications table with the appropriate Error_Type and Error_Ext_Handler.

## Populating Sender Context Information in the EBM Header

For all system faults (binding and remote) and Enterprise Service Bus (ESB) faults, the fault policy is initiated and publishes a notification message. By ensuring that your process has the following context information supplied, the order fallout management extension handler application programming language (API) constructs an enriched fault message.

All the enterprise business messages (EBMs) for order processing will pass the following information as a sender reference in the EBMHeader. This list shows the information that you need to pass for fallout:

- Order ID - Business Component ID of the Order – SalesOrder / Provisioning Order / Fulfillment Order / Fulfillment Billing Order
- Order Number – ID of the order - FulfillmentOrder#/ProvisioningOrder#/SalesOrder# (optional – needed only if available)
- SalesOrderID – Alternate Object Key – storing the Sales Order Common ID
- Sales Order Number - Alternate Object Key – storing the Sales Order Number (Siebel value)
- Sales Order Revision Number - Alternate Object Key – storing the Sales Order Number (Siebel value)
- Common Account ID – Alternate Object key – storing the Common Account ID
- Account ID – Alternate Object key – storing the Siebel Account ID (only for Sales Order EBM because the account information in the Xref would have been rolled back)
- Account Name – Alternate Object Key – storing the Siebel Account Name

Along with these fields, you would populate the SchemeID field indicating the name and the SchemeAgencyID indicating the column name.

The attribute value for schemeAgencyId of SALESORDER_NUMER is considered as the system code of the system from which the order was placed (Order Originating System Code)

This information should be entered in the EBM Header in the following path:

EBMHeader / Sender / ObjectCrossReference / SenderObjectIdentification /

Here is a sample EBMHeader section:

```
<EBMHeader>
<Sender>
<ObjectCrossReference>
 <SenderObjectIdentification>
<BusinessComponentID> OrderId</BusinenessComponentID>
            <ID> Order# (if any)</ID>
<ApplicationObjectKey>
<ID
schemeID="SALESORDER_ID"schemeAgencyID="SEBL_01">SalesOrderID</ID>
</ApplicationObjectKey>
<AlternateObjectKey>
<ID schemeID="SALESORDER_ID "
schemeAgencyID="COMMON">SalesOrderCommonID</ID>
</AlternateObjectKey>
<AlternateObjectKey>
<ID schemeID="SALESORDER_NUMBER"
schemeAgenyId="SEBL_01”>SalesOrderNumber</ID>
</AlternateObjectKey>
<AlternateObjectKey>
<ID schemeID="SALESORDER_REVISION"
schemeAgenyId="SEBL_01”>SalesOrderRevision</ID>
</AlternateObjectKey>
<AlternateObjectKey>
<ID schemeID="CUSTOMERPARTY_ACCOUNTID"
schemeAgenyId="COMMON”>CommonAccountID</ID>
</AlternateObjectKey>
<AlternateObjectKey>
<ID schemeID="CUSTOMERPARTY_ACCOUNTID"
schemeAgenyId="SEBL_01”>Siebel Account ID</ID>
</AlternateObjectKey>
<AlternateObjectKey>
<ID schemeID="CUSTOMERPARTY_ACCOUNTNAME"
schemeAgenyId="SEBL_01”>Account Name</ID>
</AlternateObjectKey>
</SenderObjectIdentification>
</ObjectCrossReference>
</Sender>
</EBMHeader>
```

Only the underlined elements are required for the SalesOrder EBM.

# Populating the Enriched Fault Message in Case of Business Faults

In case nonpartner link errors or business faults are in the business process execution language (BPEL) processes (where the BPEL process is creating the fault message and calling the AIA Async Error handling process), the expectation is that the ApplicationFaultData is populated as well.

ApplicationFaultData is an xsd: Any field in the fault message:

Fault/FaultNotification/FaultMessage/ApplicationFaultData

The BPEL processes are expected to construct a variable of element type ApplicationFaultData defined in this xsd: http://{httphostname}:{http portname}/AIAComponents/PIPS/Communications/Schemas/OrderFailureData.xsd

The fields defined in the xsd and how they have to be used are listed here.

- ApplicationFaultData / OrderFailureData / OrderID

  BusinessComponentID - SalesOrder / Provisioning Order / Fulfillment Order / Fulfillment Billing Order

  ID - SalesOrder # / Provisioning Order # / Fulfillment Order # / Fulfillment Billing Order # (If available)

  ApplicationObjectKey – If available

  AlternateObjectKey – SALESORDER_ID

  AlternateObjectKey – SALESORDER_NUMBER

  AlternateObjectKey – SALESORDER_REVISION

  AlternateObjectKey - FULFILLMENTSYSTEM_ID

  Sample:

  <BusinessComponentID> Order ID </BusinessComponentID>

  <ID> Order# (if any)</ID>

  <ApplicationObjectKey>

  <ID schemeID="**SALESORDER_ID**"schemeAgencyID="**SEBL_01**">SalesOrderID</ID>

  </ApplicationObjectKey>

  <AlternateObjectKey>

  <ID schemeID="SALESORDER_ID" schemeAgencyID="**COMMON**">SalesOrderCommonID</ID>

  </AlternateObjectKey>
  <AlternateObjectKey>

  <ID schemeID="SALESORDER_NUMBER" schemeAgenyId="**SEBL_01**”>SalesOrderNumber</ID>

  </AlternateObjectKey>

  <AlternateObjectKey>

  <ID schemeID="SALESORDER_REVISION" schemeAgenyId="**SEBL_01**”>SalesOrderRevision</ID>

  </AlternateObjectKey>

  <AlternateObjectKey>

<ID schemeID="**FULFILLMENTSYSTEM_ID** "
schemeAgenyId="**FulfillmentSystemAppID**">OrderID in the Fulfillment System</ID>

</AlternateObjectKey>

- ApplicationFaultData / OrderFailureData / AccountID

  BusinessComponentID – CommonAccountID

  ID – Account Name

  ApplicationObjectKey – Siebel AccountID (required only in the case of SalesOrder EBM)

  Sample:

  <BusinessComponentID schemeID="**CUSTOMERPARTY_ACCOUNTID**"
  schemeAgencyID="**COMMON**">AccountID</BusinessComponentID>

  <ID schemeID="CUSTOMERPARTY_ACCOUNTNAME"
  schemeAgencyID="SEBL_01">AccountName</ID>

  <ApplicationObjectKey>

  <ID schemeID="**CUSTOMERPARTY_ACCOUNTID**" schemeAgencyID="**SEBL_01**">88-878PX</ID>

  </ApplicationObjectKey>

- ApplicationFaultData / OrderFailureData / ProductID

  Information regarding the Product / Discount of the failed order line.

  In case of an entire order failure, this can be mapped for the product corresponding to the first line item of the order.

  Sample:

  <BusinessComponentID schemeID="**ITEM_ID**" schemeAgencyID="**COMMON**">Item ID

  </ BusinessComponentID>

  <ApplicationObjectKey>

  <ID schemeID="**ITEM_ID**" schemeAgencyID="**SEBL_01**">SiebelID</ ID>

  <ApplicationObjectKey>

- ApplicationFaultData / OrderFailureData / ProcessingNumber

Job ID – String type

- ApplicationFaultData / OrderFailureData / ProcessingTypeCode

  Common Value of the Processing Type Code

- ApplicationFaultData / OrderFailureData / ProcessingQuantity

  Processing Quantity as available in the EBM

- ApplicationFaultData / OrderFailureData / FailureSystemCode

  System where the fault occurred – 'AIA' in case the error is internal to the ABCS or BPEL.

  Target System ID in case the fault is identified from the target application system

- ApplicationFaultData / OrderFailureData / FailureSubSystemCode

  The code of either the subsystem or the API, where the order has failed. This is applicable in the case of participating applications. If the fault is within Oracle AIA, the service that faulted is assumed as the subsystem of failure

- ApplicationFaultData / OrderFailureData / OrderLineItemFailureDataList

  This is required if you are handling faults at the line level or if the BPEL fails while it is trying to process a particular order line.

  OrderLineItemID

  Structure similar to OrderID

  BusinessComponentID - SalesOrder / Provisioning Order / Fulfillment Order / Fulfillment Billing Order Line IDs (if any)

  ID - SalesOrder Liner # / Provisioning Order Line # / Fulfillment Order Line # / Fulfillment Billing Order Line # (If available)

  ApplicationObjectKey - If available (at the Siebel end at least if the LineId is not yet cross-referenced)

  AlternateObjectKey – SALESORDER_LINEID (COMMON)

  Sample:

  <BusinessComponentID> Order Line ID </BusinessComponentID>

  <ID> Order Line# (if any)</ID>

  <ApplicationObjectKey>

  <ID
  schemeID="SALESORDER_LINEID"schemeAgencyID="SEBL_01">SalesOrderLineID</ID>

</ApplicationObjectKey>

<AlternateObjectKey>

<ID schemeID="**SALESORDER_LINEID** " schemeAgencyID="**COMMON**">

SalesOrderLineCommonID

</ID>

</AlternateObjectKey>

ErrorCode

Error code associated with the failure

ErrorMessage

Error message associated with the failure

ErrorSeverity

Error severity associated with the failure

Status Context

Status context of the order line

FailureSubSystemCode

Code of the subsystem or API where the order line has failed. This is applicable in the case of participating applications. If the fault is within Oracle AIA, the service that faulted is assumed to be the subsystem of failure.

# Populating the Enriched Fault Message in Services without EBMs

In the Requestor ABCS Implementation services, populating the EBM_HEADER variable typically is the last step of this process and the chances of an error occurring (nonsystem fault error) is more likely during this last step.

For the nonpartner link faults or business faults, the application business connector service (ABCS) should follow the guidelines as stated in Populating the Enriched Fault Message in Case of Business Faults. The intention is to capture as many fields as possible here in this case. No common IDs can be available.

In case of system faults or enterprise service bus (ESB) faults, the extension handler feature of the Oracle AIA Error Handling Framework can be used to enrich the fault message.

As delivered, the system faults for the Siebel Requestor ABCS are handled by the Extension Handler - oracle.apps.aia.industry.comms.eh.AIAOrderFalloutErrorHandlerExtension.java to parse the Siebel order message and enrich the fault message (Fault/FaultNotification/FaultMessage/ApplicationFaultData) with the appropriate available data (OrderID and the AccountID).

**For more information** about extending error handling, see *Oracle Application Integration Architecture – Foundation Pack 2.5: Integration Developer's Guide,* "Configuring Oracle AIA Processes for Error Handling and Trace Logging," Extending Error Handling and Extending Fault Messages.

# Appendix D: Cross-References for the Process Integration for Product Management

This section describes the cross references used in the process integration for product management and provides information about:

- Cross-reference values.
- Integration Solution cross-references.
- Product synchronization flow.
- Discount synchronization flow.

## Cross-Reference Values

The following values denote the entries made to the Xref table and what they mean.

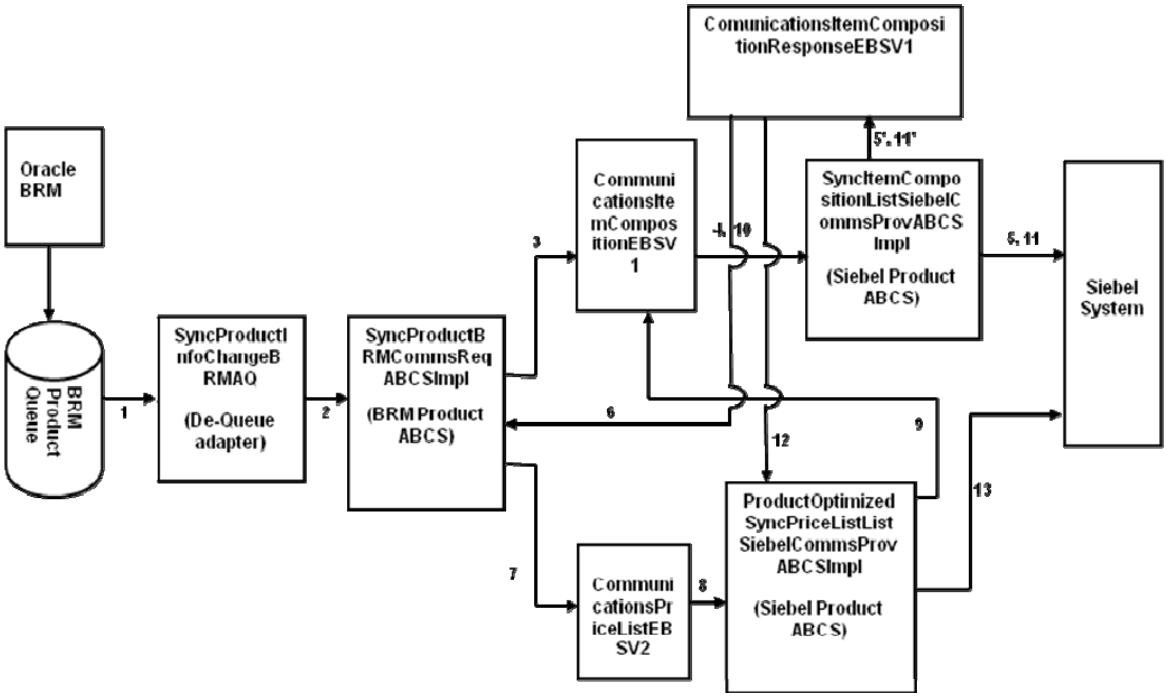
```
ITEM_ITEMID    : cross references the BRM (Portal) ProductID and the
             Siebel ProductID.
COMMON   : auto generated GUID.
BRM_01   : POID of BRM Product ABM.
SEBL_01  : ProductID of Siebel Product ABM.

PRICELINE _ID  : cross references the BRM (Portal) Product ID to
                   Siebel PriceLineID. Also links to the COMMON of
                   ITEM_ITEMID.
COMMON         : auto generated GUID.
BRM_01         : POID of BRM Product ABM.
SEBL_01        : Siebel PriceListItemID for the main product.
ITEM_ID_COMMON : From ITEM_ID.COMMON.

PRICELINETYPE _ID    : cross references BRM (Portal) Product's
Events to
                   Siebel PriceLineID. Also links to the COMMON of
                   ITEM_ITEMID.
COMMON         : Auto generated GUID.
BRM_01         : POID of BRM Product ABM + Event Name.
SEBL_01        : Siebel PriceListItemID for the event product.
ITEM_ID_COMMON : From ITEM_ID.COMMON.
SIEBELPRODUCTEVENTXREF     :     cross references BRM (Portal)
Product's Event   that is Associated with the main product in
Siebel.
ITEM_ID_COMMON       : From ITEM_ID.COMMON
LINEPRICETYPECODE          : PRICELINETYPE _ID.COMMON
```

# Integration Solution Cross-References

These are the cross-references:

| Operation | Entity | Siebel CRM ID | Oracle BRM ID |
|---|---|---|---|
| Inserts/Refers | ITEM_ITEMID | Product ID | Product ID |
| Inserts/Refers | PRICELINE_ID (main products only) | Price Line ID to Common ITEM_ITEMID of main product | Product ID |
| Inserts/Refers | PRICELINETYPE_ID (for event/special type products) | Price Line ID to Common ITEM_ITEMID (both only if exits, may not be in case of optimized mode) | Generated Product ID for Event products (ProductIDEvent Name) |
| Inserts/Refers | SIEBELPRODUCTEVENTXREF | Common ITEM_ITEMID for the parent product to Common PRICELINETYPE_ID for event product | |

# Product Synchronization Flow




Product synchronization flow

1. Before the call 3, which SyncProductBRMCommsIReqABCSImpl makes to CommunicationsItemCompositionEBSV1, the following entries are made in the XREF_DATA table:

| XREF_TABLE_NAME | VALUE |
|---|---|
| ITEM_ITEMID | <POID of BRM product> |
| ITEM_ITEMID | COMMON GUID1 |

2. During the response back from Siebel to SyncItemCompositionListSiebelCommsProvABCSImpl, the following entry is made in the XREF_DATA table:

| XREF_TABLE_NAME | VALUE |
|---|---|
| ITEM_ITEMID | <ProductID in Siebel > |

3. Before the call 7 from SyncProductBRMCommsIReqABCSImpl to CommunicationsPriceListEBSV2 is made, the following entries are made in XREF_DATA table:

| XREF_TABLE_NAME | VALUE |
|---|---|
| PRICELINE_ID | POID of BRM product |

| | |
|---|---|
| PRICELINE_ID | COMMON GUID2 |
| PRICELINETYPE_ID | POID of BRM Event product |
| PRICELINETYPE_ID | COMMON GUID2 |

4. Before the call 9 from ProductOptimizedSyncPriceListLineListSiebelProvABCSImpl to CommunicationsItemCompositionEBSV1 is made, the following entries are made in the XREF_DATA table:

| XREF_TABLE_NAME | VALUE |
|---|---|
| SIEBELPRODUCTEVENTXREF | LINEPRICETYPECODE GUID2 |
| SIEBELPRODUCTEVENTXREF | ITEM_ID_COMMON GUID1 |

5. During the response from SyncItemCompositionListSiebelCommsProvABCSImpl for the call 9, the following entries are made in the XREF_DATA table:

| XREF_TABLE_NAME | XREF_COLUMN_NAME |
|---|---|
| ITEM_ITEMID | COMMON GUID2 |
| ITEM_ITEMID | < ProductID in Siebel of Event Product > |

**Note:** For the simple product sync, the previous call is not made because the main product is already synced as an Item.

6. Before the call 13 from ProductOptimizedSyncPriceListLineListSiebelProvABCSImpl to Siebel System is made, the following entries are made in the XREF_DATA table:

| XREF_TABLE_NAME | VALUE |
|---|---|
| PRICELINE_ID | ITEM_ID_COMMON |
| PRICELINETYPE_ID<br>(in case of multi-event product) | ITEM_ID_COMMON |

7. During the response from the Siebel system, ProductOptimizedSyncPriceListLineListSiebelProvABCSImpl, the following entries are made in the XREF_DATA table:

| XREF_TABLE_NAME | VALUE |
|---|---|
| PRICELINE_ID | < ProductID in Siebel for Event Product > |
| PRICELINETYPE_ID<br>(in case of multi-event product) | < ProductID in Siebel for Event Product > |

## Examples

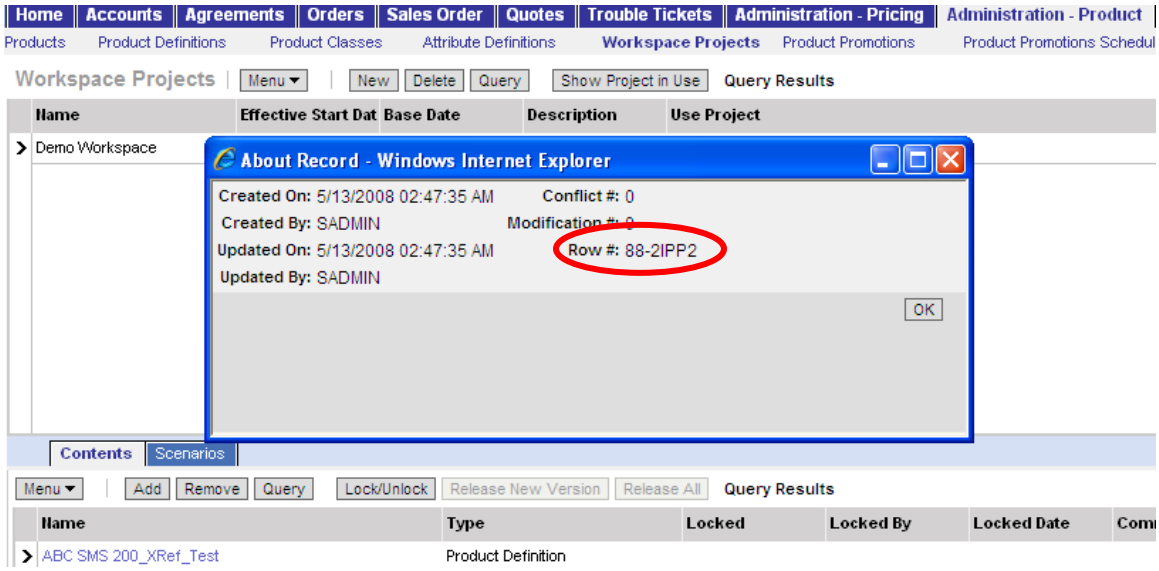
Here are two examples, one for simple product synchronization and one for complex product synchronization.

### Example 1

Consider an actual scenario in which a simple product is being synced from Oracle Communications Billing and Revenue Management (Oracle BRM) to Siebel Customer Relationship Management (Siebel CRM).

Step1: Create products in Oracle BRM to be synced with Siebel CRM:


Step 2: Verify the synced records in Siebel:




Step 3: The following examples show how data is entered into the cross-reference table corresponding to the points 1 through 7 explained previously.

| RESULTS: | | | | |
|---|---|---|---|---|
| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE |
| 1 | ITEM_ITEMID | BRM_01 | <ROWNUM_1> | <BRM_PROD_01> |
| 2 | ITEM_ITEMID | COMMON | <ROWNUM_1> | <COMMON_PROD_01> |

Table corresponding to point 1

| RESULTS: |
|---|

| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE |
|---|---|---|---|---|
| 1 | ITEM_ITEMID | SEBL_01 | <ROWNUM_1> | <CRM_PROD_01> |

Table corresponding to point 2

| RESULTS: | | | | |
|---|---|---|---|---|
| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE |
| 1 | PRICELINE_ID | BRM_01 | <ROWNUM_2> | <BRM_PROD_01> |
| 2 | PRICELINE_ID | COMMON | <ROWNUM_2> | <COMMON_PRICE_ID1> |
| 3 | PRICELINETYPE_ID | BRM_01 | <ROWNUM_3> | <BRM_PROD_01_EVENT 1> |
| 4 | PRICELINETYPE_ID | COMMON | <ROWNUM_3> | <COMMON_PRICETYPE _ID1> |

Table corresponding to point 3

| RESULTS: | | | | |
|---|---|---|---|---|
| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE |
| 1 | SIEBELPRODUCTEVE NTXREF | LINEPRICETYPECODE | <ROWNUM_4> | <COMMON_PRICETYPE_I D1> |
| 2 | SIEBELPRODUCTEVE NTXREF _ID | ITEM_ID_COMMON | <ROWNUM_4> | <COMMON_PROD_01> |

Table corresponding to point 5

| RESULTS: | | | | |
|---|---|---|---|---|
| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE |
| 1 | PRICELINE_ID | ITEM_ID_COMMON | <ROWNUM_2> | <COMMON_PROD_01> |
| 2 | PRICELINE_ID | SEBL_01 | <ROWNUM_2> | <CRM_PRICE_01> |

Table corresponding to point 7

This table shows the complete entry for the product sync:

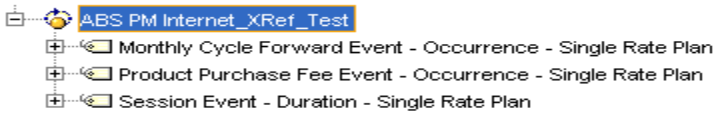
| RESULTS: | | | | |
|---|---|---|---|---|
| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE |
| 1 | ITEM_ID | BRM_01 | <ROWNUM_1> | <BRM_PROD_01> |
| 2 | ITEM_ID | COMMON | <ROWNUM_1> | <COMMON_PROD_01> |
| 3 | ITEM_ID | SEBL_01 | <ROWNUM_1> | <CRM_PROD_01> |
| 4 | PRICELINE_ID | BRM_01 | <ROWNUM_2> | <BRM_PROD_01> |

| | | | | |
|---|---|---|---|---|
| 5 | PRICELINE_ID | COMMON | <ROWNUM_2> | <COMMON_PRICE_ID1> |
| 6 | PRICELINETYPE_ID | BRM_01 | <ROWNUM_3> | <BRM_PROD_01_EVENT1> |
| 7 | PRICELINETYPE_ID | COMMON | <ROWNUM_3> | <COMMON_PRICETYPE_ID1> |
| 8 | SIEBELPRODUCTEVENTXREF | LINEPRICETYPECODE | <ROWNUM_4> | <COMMON_PRICETYPE_ID1> |
| 9 | SIEBELPRODUCTEVENTXREF _ID | ITEM_ID_COMMON | <ROWNUM_4> | <COMMON_PROD_01> |
| 10 | PRICELINE_ID | ITEM_ID_COMMON | <ROWNUM_2> | <COMMON_PROD_01> |
| 11 | PRICELINE_ID | SEBL_01 | <ROWNUM_2> | <CRM_PRICE_01> |

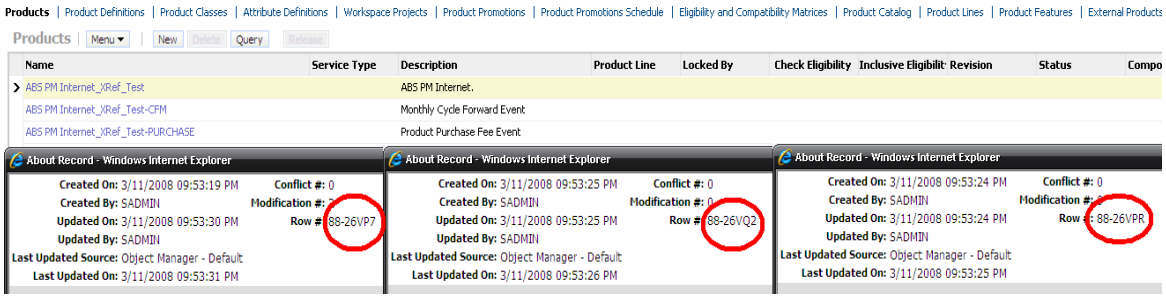
## Example 2

Now consider the scenario in which a complex product is being synced from Oracle BRM to Siebel CRM.

Step1: Create products in Oracle BRM to be synced with Siebel CRM.




Step 2: Verify the synced records in Siebel.




Step 3: The following examples show how data is entered into the cross-reference table corresponding to the points 1 through 7 explained previously.

| RESULTS: | | | | |
|---|---|---|---|---|
| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE |
| 1 | ITEM_ID | BRM_01 | <ROWNUM_1> | <BRM_PROD_01> |
| 2 | ITEM_ID | COMMON | <ROWNUM_1> | <COMMON_PROD_01> |

Table corresponding to point 1

| RESULTS: | | | | |
|---|---|---|---|---|
| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE |
| 1 | ITEM_ID | SEBL_01 | <ROWNUM_1> | <CRM_PROD_01> |

Table corresponding to point 2

| RESULTS: | | | | |
|---|---|---|---|---|
| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE |
| 1 | PRICELINE_ID | COMMON | <ROWNUM_2> | <BRM_PROD_01> |
| 2 | PRICELINE_ID | BRM_01 | <ROWNUM_2> | <COMMON_PRICE_01> |

| | | | | |
|---|---|---|---|---|
| 3 | PRICELINETYPE_ID | COMMON | <ROWNUM_3> | <COMMON_PRICETYPE_01> |
| 4 | PRICELINETYPE_ID | BRM_01 | <ROWNUM_3> | <BRM_PROD_01_EVENT1> |
| 5 | PRICELINETYPE_ID | BRM_01 | <ROWNUM_4> | <BRM_PROD_01_EVENT2> |
| 6 | PRICELINETYPE_ID | COMMON | <ROWNUM_4> | <COMMON_PRICETYPE_02> |

Table corresponding to point 3

| RESULTS: | | | | |
|---|---|---|---|---|
| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE |
| 1 | SIEBELPRODUCTEVENTXREF | LINEPRICETYPECODE | <ROWNUM_4> | <COMMON_PRICETYPE_01> |
| 2 | SIEBELPRODUCTEVENTXREF _ID | ITEM_ID_COMMON | <ROWNUM_4> | <COMMON_PROD_01> |

Table corresponding to point 4

| RESULTS: | | | | |
|---|---|---|---|---|
| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE |
| 1 | ITEM_ID | COMMON | <ROWNUM_5> | <COMMON_PRICETYPE_01> |
| 2 | ITEM_ID | SEBL_01 | <ROWNUM_5> | <CRM_PROD_02> |

Table corresponding to point 5

| RESULTS: | | | | |
|---|---|---|---|---|
| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE |
| 1 | PRICELINE_ID | ITEM_ID _COMMON | <ROWNUM_3> | <COMMON_PRICETYPE_01> |
| 2 | PRICELINETYPE_ID | ITEM_ID _COMMON | <ROWNUM_4> | <COMMON_PRICETYPE_02> |

Table corresponding to point 6

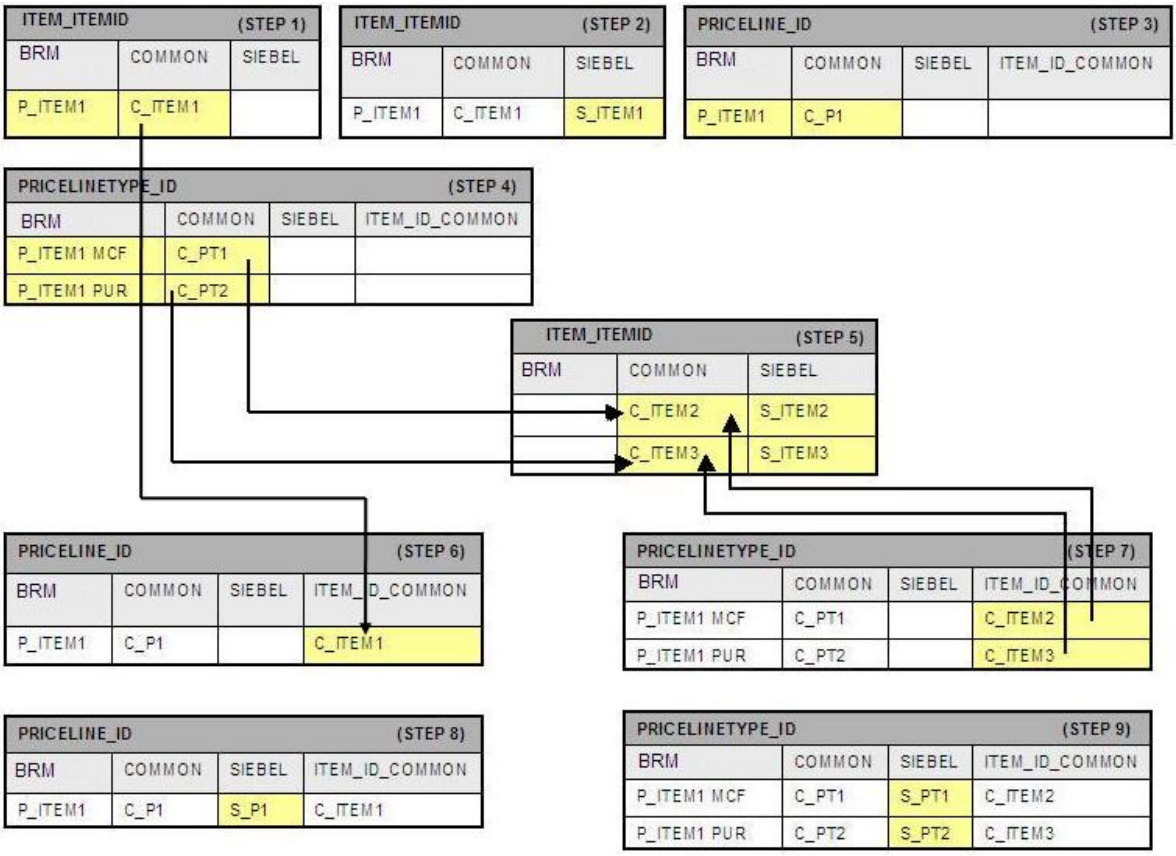
| RESULTS: | | | | |
|---|---|---|---|---|
| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE |
| 1 | PRICELINE_ID | SEBL_01 | <ROWNUM_3> | <CRM_ITEM_PRICE_01> |
| 2 | PRICELINETYPE_ID | SEBL_01 | <ROWNUM_4> | <CRM_ITEM_PRICE_02> |

Table corresponding to point 7

This table shows the complete entry for the product sync:

| RESULTS: | | | | |
|---|---|---|---|---|
| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE |
| 1 | ITEM_ID | BRM_01 | <ROWNUM_1> | <BRM_PROD_01> |
| 2 | ITEM_ID | COMMON | <ROWNUM_1> | <COMMON_PROD_01> |
| 3 | ITEM_ID | SEBL_01 | <ROWNUM_1> | <CRM_PROD_01> |
| 4 | PRICELINE_ID | COMMON | <ROWNUM_2> | <BRM_PROD_01> |
| 5 | PRICELINE_ID | BRM_01 | <ROWNUM_2> | <COMMON_PRICE_01> |
| 6 | PRICELINETYPE_ID | COMMON | <ROWNUM_3> | <COMMON_PRICETYPE_01> |
| 7 | PRICELINETYPE_ID | BRM_01 | <ROWNUM_3> | <BRM_PROD_01_EVENT1> |
| 8 | PRICELINETYPE_ID | BRM_01 | <ROWNUM_4> | <BRM_PROD_01_EVENT2> |
| 9 | PRICELINETYPE_ID | COMMON | <ROWNUM_4> | <COMMON_PRICETYPE_02> |
| 10 | SIEBELPRODUCTEVENTXREF | LINEPRICETYPECODE | <ROWNUM_4> | <COMMON_PRICETYPE_01> |
| 11 | SIEBELPRODUCTEVENTXREF _ID | ITEM_ID_COMMON | <ROWNUM_4> | <COMMON_PROD_01> |
| 12 | ITEM_ID | COMMON | <ROWNUM_5> | <COMMON_PRICETYPE_02> |
| 13 | ITEM_ID | SEBL_01 | <ROWNUM_5> | <CRM_PROD_02> |
| 14 | PRICELINE_ID | ITEM_ID _COMMON | <ROWNUM_3> | <COMMON_PRICETYPE_01> |
| 15 | PRICELINETYPE_ID | ITEM_ID _COMMON | <ROWNUM_4> | <COMMON_PRICETYPE_02> |
| 16 | PRICELINE_ID | SEBL_01 | <ROWNUM_3> | <CRM_ITEM_PRICE_01> |
| 17 | PRICELINETYPE_ID | SEBL_01 | <ROWNUM_4> | <CRM_ITEM_PRICE_02> |

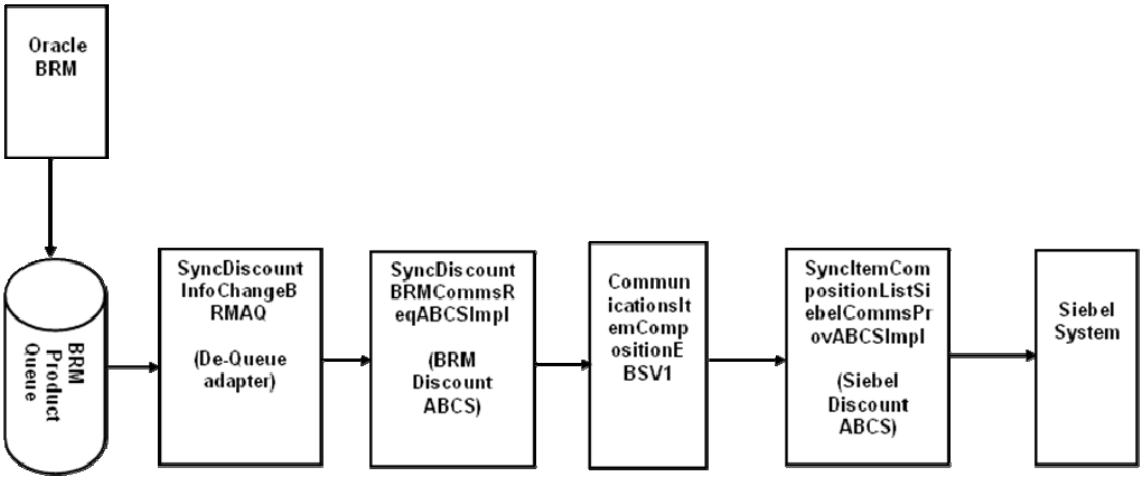
This diagram shows a high-level overview of how the mappings are maintained in the cross-reference table:




Cross-reference table

# Discount Synchronization Flow

This diagram shows the discount synchronization flow:




## Discount synchronization flow

1. Before the call 3, which SyncDiscountBRMCommsReqABCSImpl makes to CommunicationsItemCompositionEBSV1, the following entries are made in the XREF_DATA table: ITEM_ITEMID, COMMON, POID of BRM product.

| XREF_TABLE_NAME | VALUE |
|---|---|
| ITEM_ITEMID | COMMON GUID |
| ITEM_ITEMID | <POID OF BRM PRODUCT> |

2. During the response from the Siebel system, a SyncItemCompositionListSiebelCommsProvABCSImpl the following entry is made in the XREF_DATA table with the value: ProductID of Siebel Product.

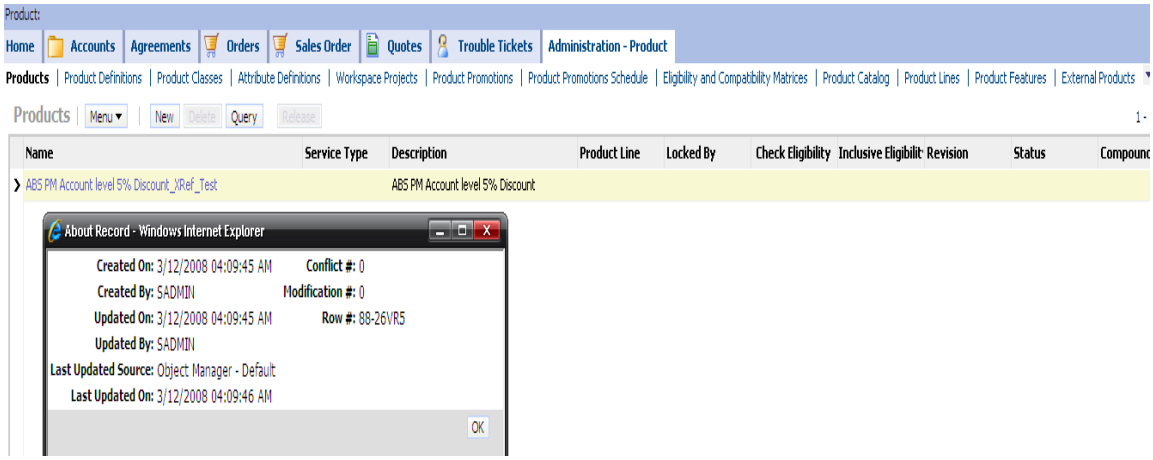
| XREF_TABLE_NAME | VALUE |
|---|---|
| ITEM_ITEMID | <PRODUCTID OF SIEBEL PRODUCT> |

## Example

Now consider an actual scenario in which a discount is being synced from Oracle BRM to Siebel CRM.

Step 1: Create discounts in Oracle BRM to be synced with Siebel.


Step 2: Verify the synced records in Siebel.




Step 3: The following examples show how data is entered into the cross-reference table corresponding to the points 1 and 2 explained previously.

Results:

| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE | IS_DELETED | LAST_MODIFIED | LAST_ACCESSED |
|---|---|---|---|---|---|---|---|
| 1 | ITEM_ID | BRM_01 | 2E60E990F02D11DCBFCAF1F293F06D61 | 0.0.0.1 /discount 600048 0 | N | 1205323775657 | 1205323775657 |
| 2 | ITEM_ID | COMMON | 2E60E990F02D11DCBFCAF1F293F06D61 | 2d31373437313438343183534303233 | N | 1205323775657 | 1205323775657 |

Figure corresponding to point 1

Results:

| | XREF_TABLE_NAME | XREF_COLUMN_NAME | ROW_NUMBER | VALUE | IS_DELETED | LAST_MODIFIED | LAST_ACCESSED |
|---|---|---|---|---|---|---|---|
| 1 | ITEM_ID | SEBL_02 | 2E60E990F02D11DCBFCAF1F293F06D61 | 88-26VR5 | N | 1205323788212 | 1205323788212 |

Figure corresponding to point 2

# Appendix E: Configuring Multiple Instances of Oracle BRM

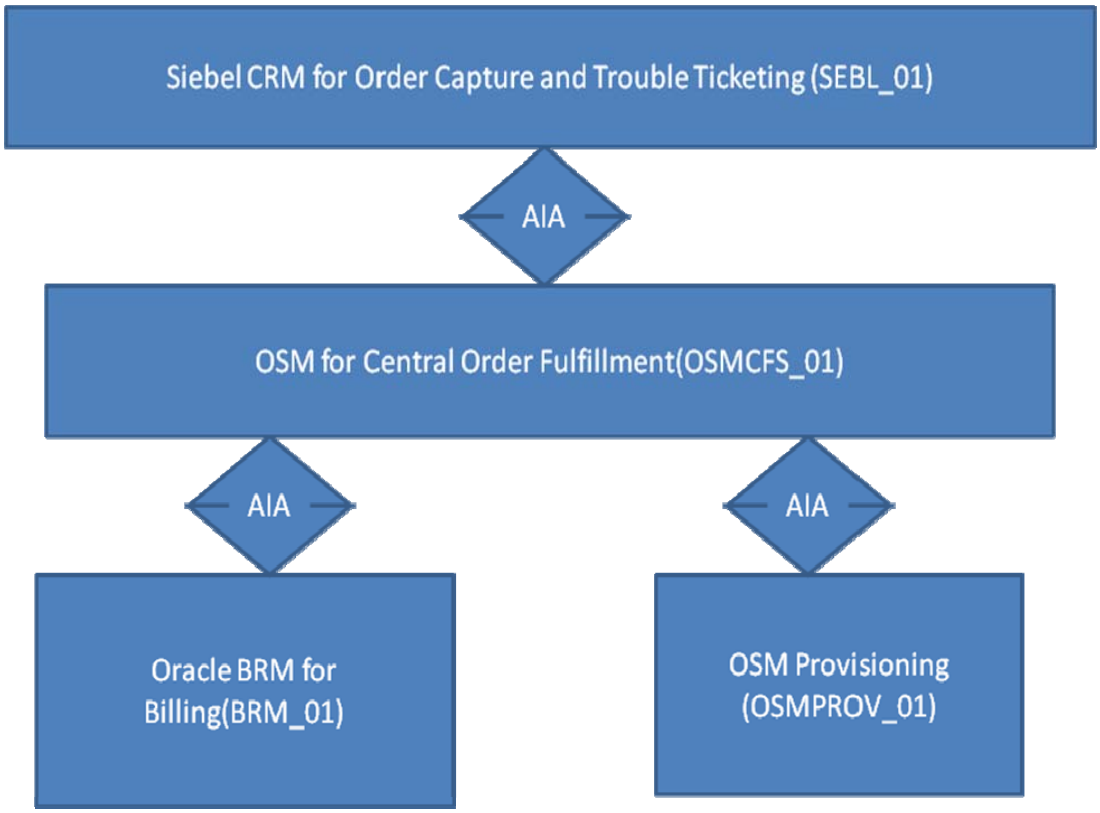
This section provides examples of how to configure additional Oracle Communications Billing and Revenue Management (Oracle BRM) instances for the process integrations in Oracle Application Integration Architecture (Oracle AIA) for Communications.

## Understanding System Codes in Oracle AIA

In Oracle AIA, each system instance is identified in Oracle AIA by a unique identifier, called a *system code*. The system codes help Oracle AIA identify the source or destination of a message. As delivered, Oracle AIA comes with following system codes:

| | |
|---|---|
| **SEBL_01** | The Siebel Customer Relationship Management (Siebel CRM) instance for order capture and trouble ticketing. |
| **BRM_01** | One Oracle BRM instance for order billing. |
| **OSMCSF_01** | The central fulfillment system instance of Oracle OSM (Oracle Order and Service Management) system. This is the instance responsible for customer orders order management. |
| **OSMPROV_01** | The provisioning system instance of Oracle OSM. |

This diagram shows the topology as delivered:




## Oracle AIA topology

Oracle AIA uses cross-reference (xref) tables to maintain mapping of system-specific identifiers (account ID, product ID, and so on). One xref table exists per entity. In an xref table, columns are created for each system instance. System codes are used as column names.

Oracle AIA uses domain value maps (DVMs) to map values of enumeration type attributes (country code, state Code, price type). One DVM exists for each enumeration type attribute. Columns are created for each system instance. System codes are used as column names.

System codes are also used to identify the sender and target in the enterprise business message (EBM) header for a given EBM message. Also in AIAConfigurationProperties.xml, system code values are used to name the properties that require instance-specific values such as EndPointURI (each system has a different end point URI). An example of such a property is:

<Property name="Routing.BRMSUBSCRIPTIONService.BRM_01.EndpointURI">eis/BRM</Property>.

Because Oracle OSM communicates to Oracle AIA using AIA EBMs, AIA Common IDs, and AIA DVM values, you do not need to have separate columns for OSMCFS_01 and OSMPROV_01 in DVMs and xrefs. Also, because Oracle AIA-OSM communications is using automatic queue synchronizations, no Oracle OSM-specific properties are in AIAConfigurationProperties.xml.

**For more information,** see *Oracle Application Integration Architecture – Foundation Pack 2.5: Core Infrastructure Components Guide*, “Managing the Oracle AIA Application Registry.”

Oracle OSM also recognizes the fulfillment topologies and assigns logical names to each system instance. These logical names should match the system codes configured in Oracle AIA.

# Adding an Additional Oracle BRM Instance – General Steps

Use this sample information as an overview of the process.

Assume that you have two billing instances. As shown in the previous section, the installation as delivered configures one Oracle BRM instance. To configure the second billing system, follow these steps. These steps guide you through the process to add one billing instance. Repeat them for each additional Oracle BRM instance.

1. Determine the system code for the new Oracle BRM instance, for example, BRM_02.
2. Add the information about this Oracle BRM instance into the Oracle AIA Application Registry (BSR system table). In the registry entry, add the system code as BRM_02 and the internal ID as the value coming from Oracle BRM into brmproductabo:InstanceId during, for instance, the product sync. The following is an example of the product sync message:

```
<SyncProductReqMsg><part
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
name="SyncProduct"><brmproductabo:ProductInfoChange
xmlns:brmproductabo="http://www.portal.com/schemas/CRMSync"
brmproductabo:InstanceId="brm_instance_2">
```

**For more information** about the Oracle AIA Application Registry*, see Oracle Application Integration Architecture – Foundation Pack 2.5: Core Infrastructure Components Guide*, "Managing the Oracle AIA Application Registry."

**For more information** about the Oracle BRM instance ID, see *Oracle Communications Billing and Revenue Management (BRM) Documentation*, "BRM Documentation," Service Integration Components, Synchronization Queue Data Manager. Look for the payloadconfig_crm_sync.xml. In this file you need to ensure that the InstanceId tag has the same value for all the EventDefs. Also, it has to be unique for each Oracle BRM instance.

3. Find all the xrefs and DVMs that have a BRM_01 column. Refer to the implementation guides of your Oracle BRM-interfacing PIPs (Order to Bill, Agent Assisted Billing Care) for the complete list of DVMs and xrefs. Add a new column BRM_02 for these xrefs and DVMs. For DVMs, also provide the values for this new column, which can typically be the same as the values for the preseeded BRM_01 column.

**For more information** about adding columns into cross-references, see the *Oracle Cross-Reference User Guide*.

**For more information** about DVMs, see the *Oracle Enterprise Service Bus Developer's Guide* available on oracle.com.

4. Configure the Oracle BRM JCA Adapter to add a new connection factory for this Oracle BRM instance. Example: eis/BRM2.

**For more information,** see *Oracle Communications Billing and Revenue Management (BRM) Documentation*, "Service Integration Components," JCA Resource.

5. In AIAConfigurationProperties.xml, look for all properties with name BRM_01 in it (for example, <Property name="Routing.BRMCUSTService.BRM_01.EndpointURI">eis/BRM1</Property>). Make a copy of each one of them and rename the new entry to have BRM_02 instead of BRM_01 in its name (example: <Property name="Routing.BRMCUSTService.**BRM_02**.EndpointURI">**eis/BRM2**</Property>) and specify the correct value for it.
6. For receiving Customer/Billing Profile updates from Siebel CRM, see the instructions in Receiving Customer or Billing Profile Updates and Friends and Family List Updates from Siebel.
7. If products from this new Oracle BRM instance need to be synchronized to Siebel, then see the instructions in Adding a New BRM instance: Configure Product Sync.
8. If the Agent Assisted Billing Care PIP is installed, then see the section Adding a New BRM instance: Add Routing Rules for Agent Assisted Billing Care PIP to determine how to add new routing rules for the Oracle BRM instance.

# Receiving Customer or Billing Profile Updates and Friends and Family List Updates from Siebel CRM

Any changes related to a customer or its billing profile starts the Customer Sync flow.

**For more information,** see "Chapter 6: Understanding the Process Integration for Customer Management," Update Customer Accounts Integration Flow.

Any changes in the Friends and Family list in Siebel CRM starts the Friends and Family List Update flow

**For more information,** refer to "Chapter 4: Understanding the Process Integration for Order Management," Supporting Friends and Family Lists.

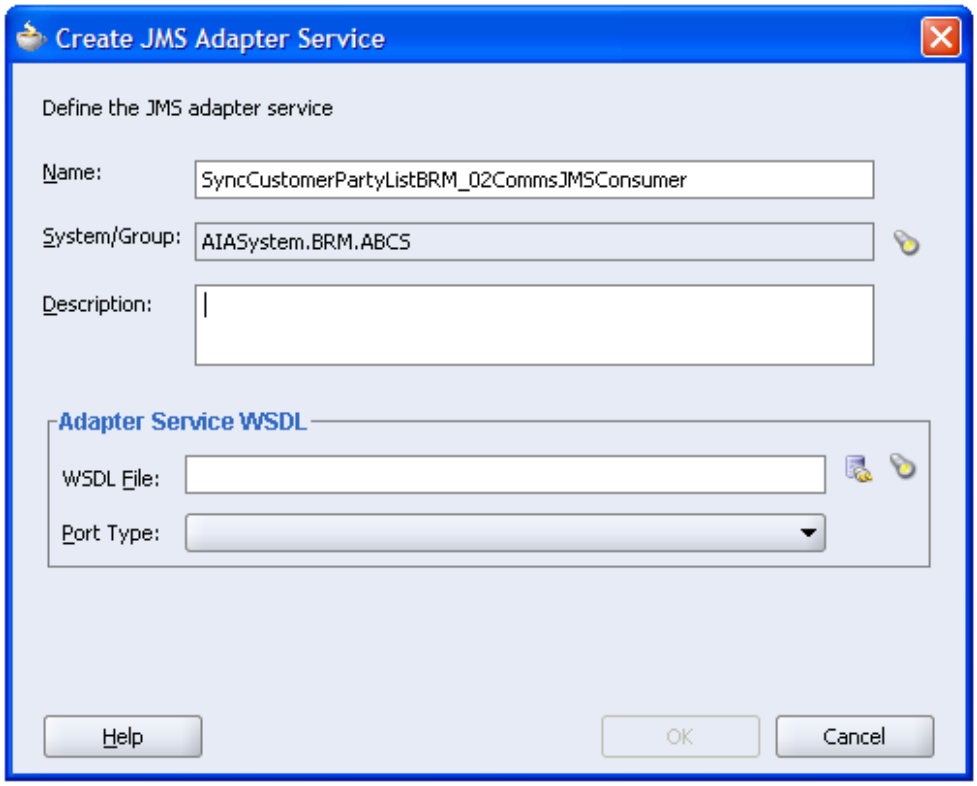
## Customer and Billing Profile Sync flow

This flow syncs customer and bill profile updates from Siebel CRM. To support multiple recipients for the updates in a *publish and subscribe* mode, a Java Message Service topic (JMS TOPIC) is used. For each recipient, a separate consumer process is needed to consume messages from this topic. As delivered, Oracle AIA ships the consumer for BRM_01 with the name SyncCustomerPartyListBRM_01CommsJMSConsumer.

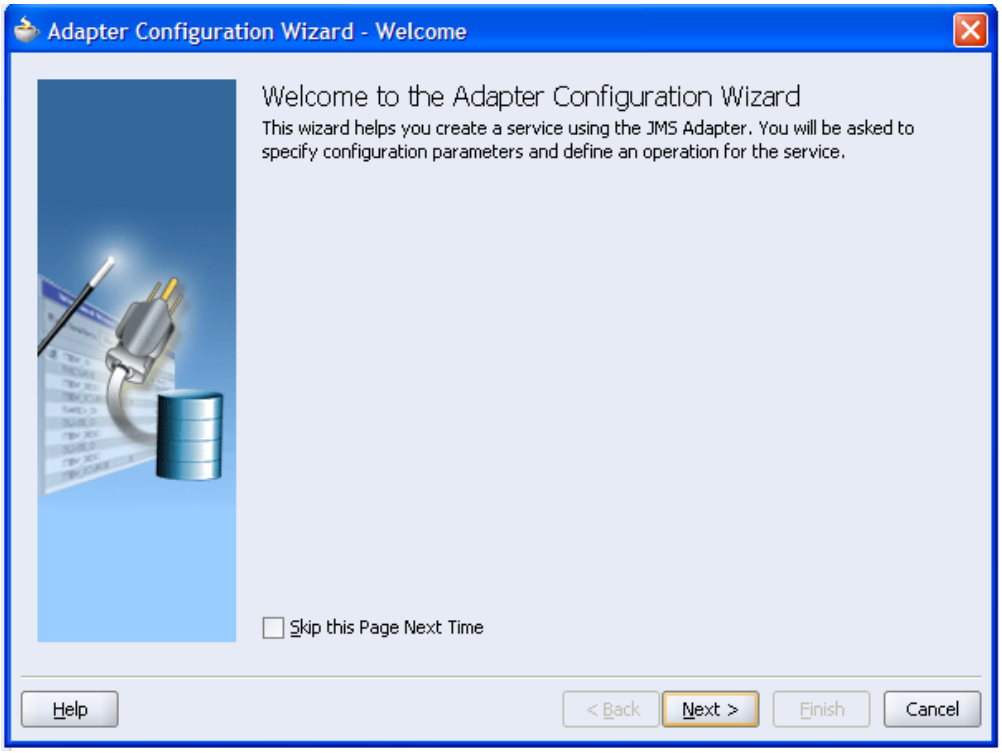
For each additional Oracle BRM instance (such as BRM_02), a new consumer needs to be created. The purpose of this Oracle BRM instance-specific consumer topology is to give an update EBM for an account. Check in the customer account cross-reference topology whether a BRM ID exists for this account (which means Oracle BRM has this account), and pass this message to the Oracle BRM instance. Otherwise, ignore the message.

Follow these steps to create a new consumer for a new Oracle BRM instance. The example here assumes BRM_02 as the system code for the new instance.

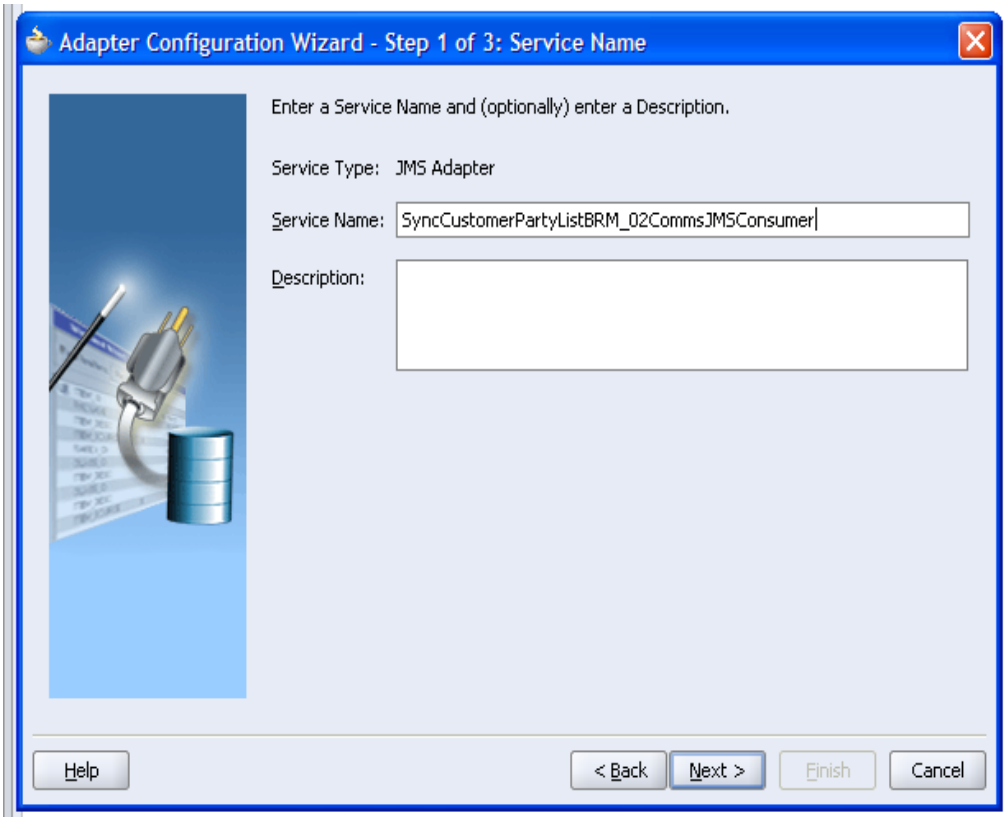
1. Create a new ESB project in JDeveloper by the name SyncCustomerPartyList[BRMInstanceID]CommsJMSConsumer, where BRMInstanceID is the ID of the new Oracle BRM instance to be added, for example, SyncCustomerPartyListBRM_02CommsJMSConsumer for BRM_02 as the ID.
2. In the ESB file generated, create a new JMS Adapter. The System/Group should be AIASystem.BRM. The name should be SyncCustomerPartyList[BRM_02]CommsJMSConsumer. The screen looks like this:



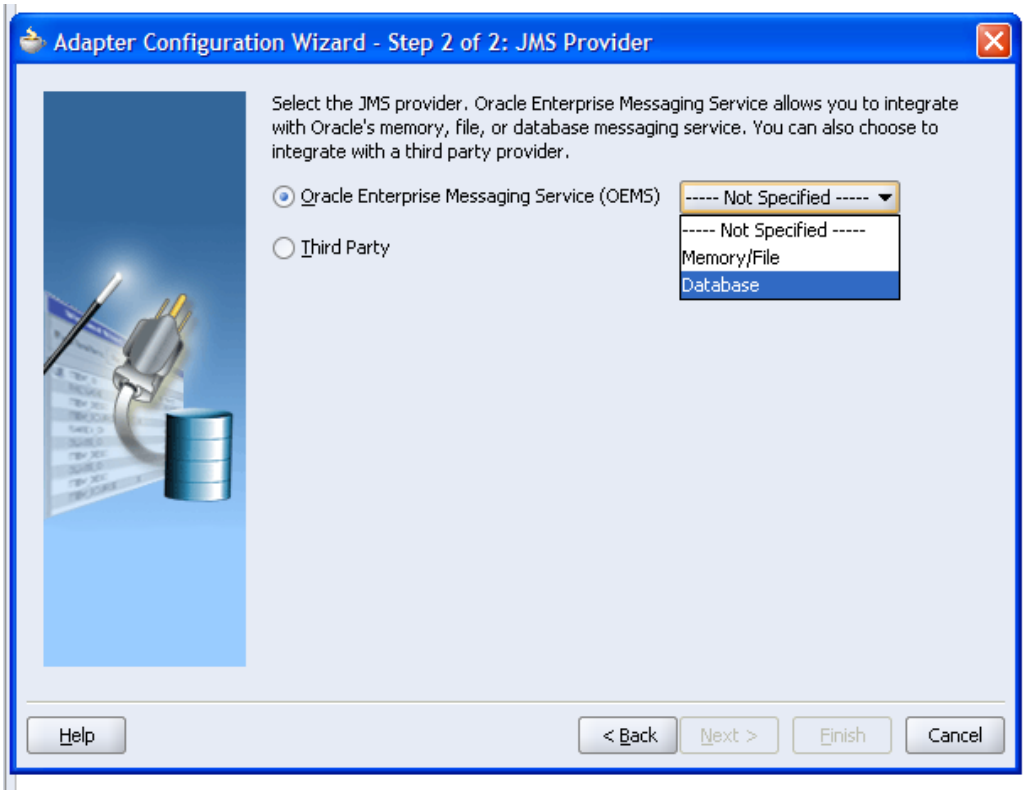

3. Click the Configure Adapter Service WSDL button next to the WSDL file location.
4. This screen appears:



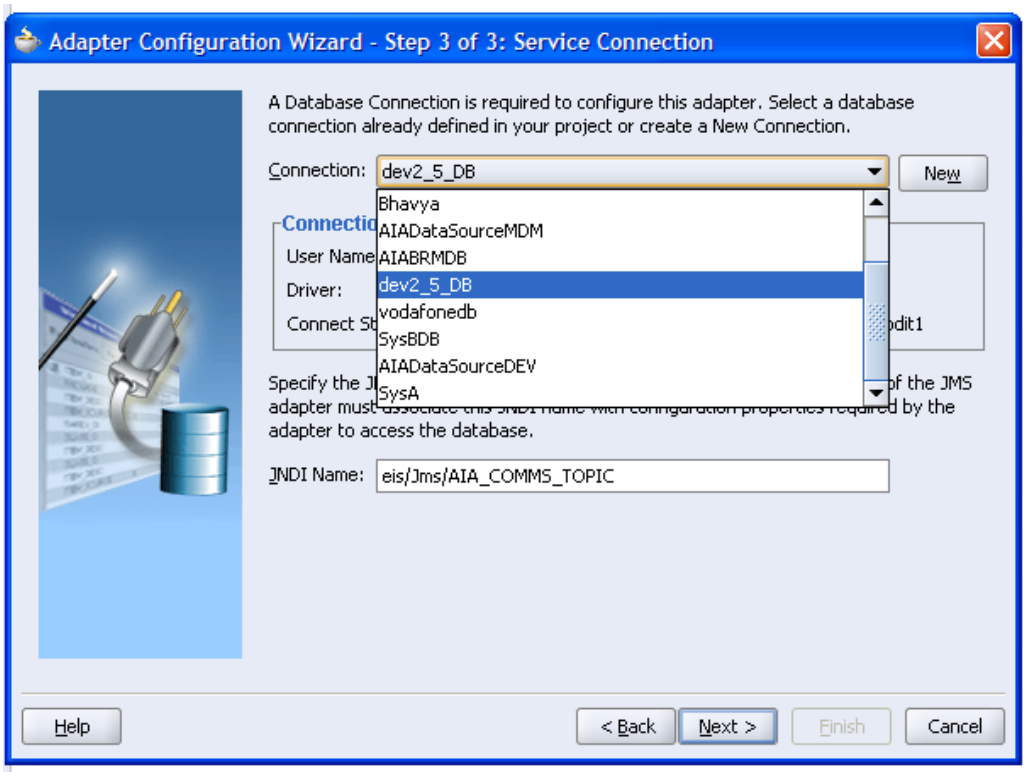

5. Click Next.
6. This screen appears:



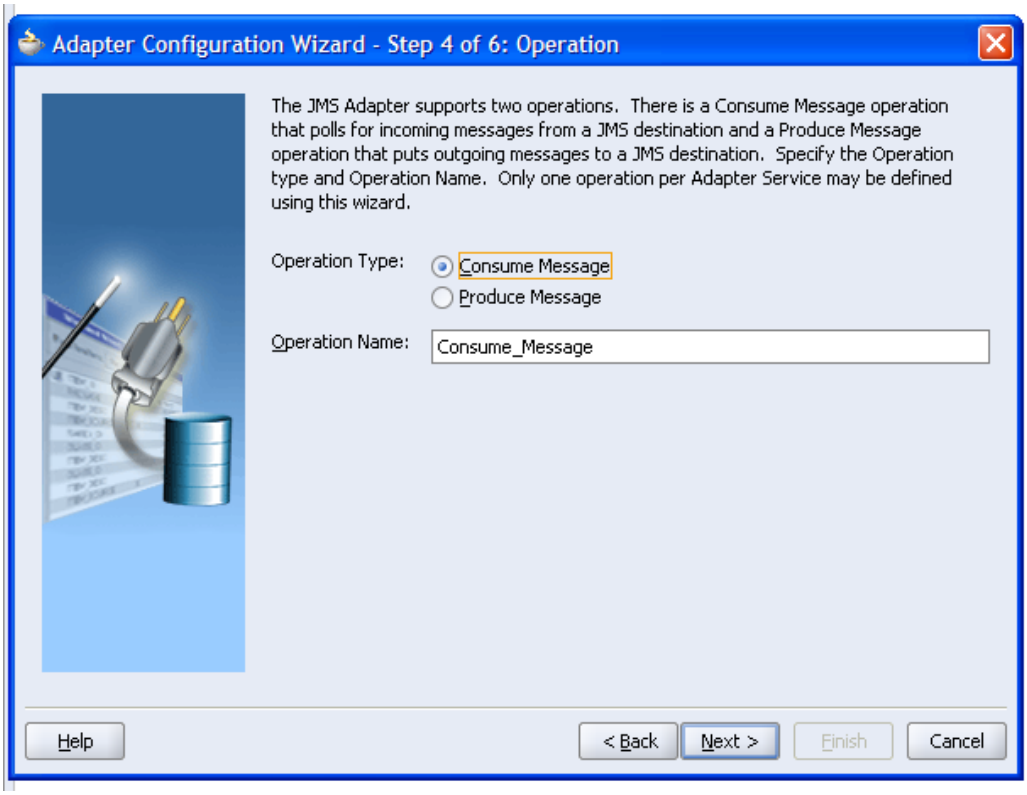
7. Click Next.
8. This screen appears:



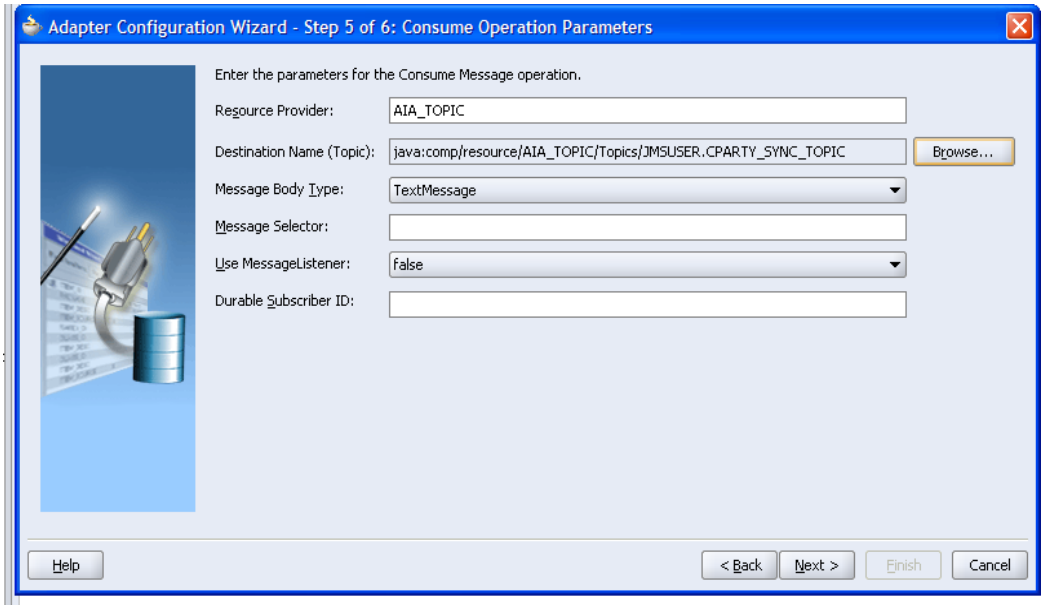

9. Select Oracle Enterprise Messaging Service (OEMS) as the database, and click Next.
10. Create a database connection that points to the database of the Fusion Middleware (FMW) server. Select the database from the Connection available options, and enter the JNDI name **eis/Jms/AIA_COMMS_TOPIC.** Click Next.




11. Select Consume Message for the operation name and click Next:

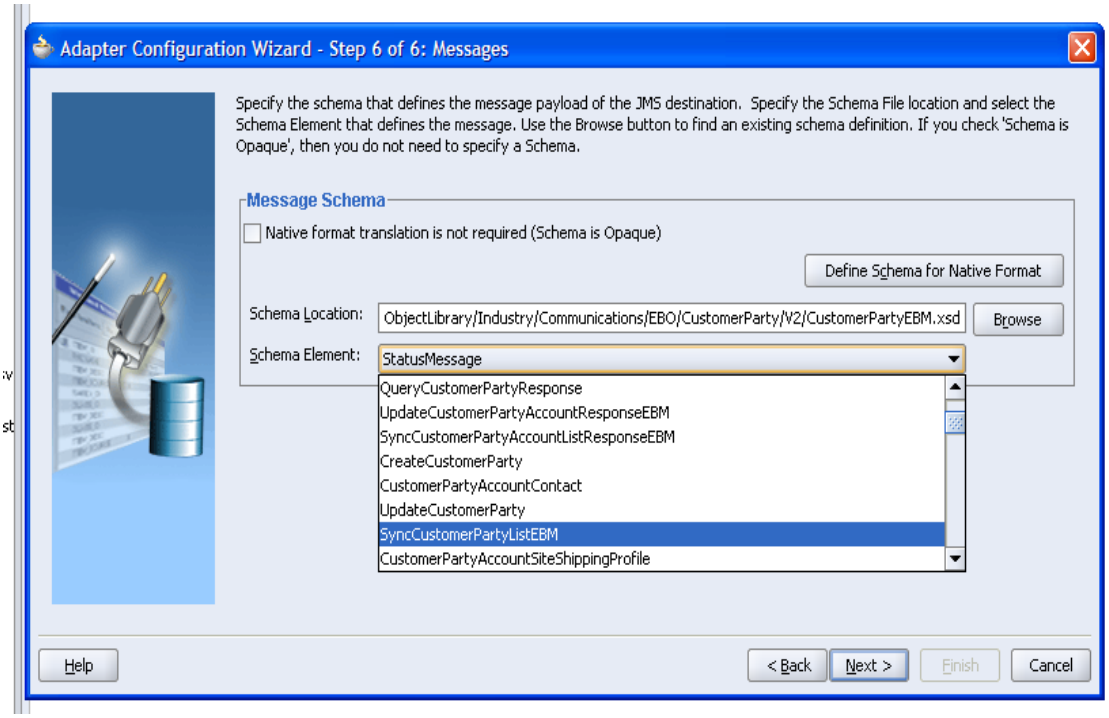


12. Enter AIA_TOPIC for the resource provider name and select CPARTY_SYNC_TOPIC for the destination name:



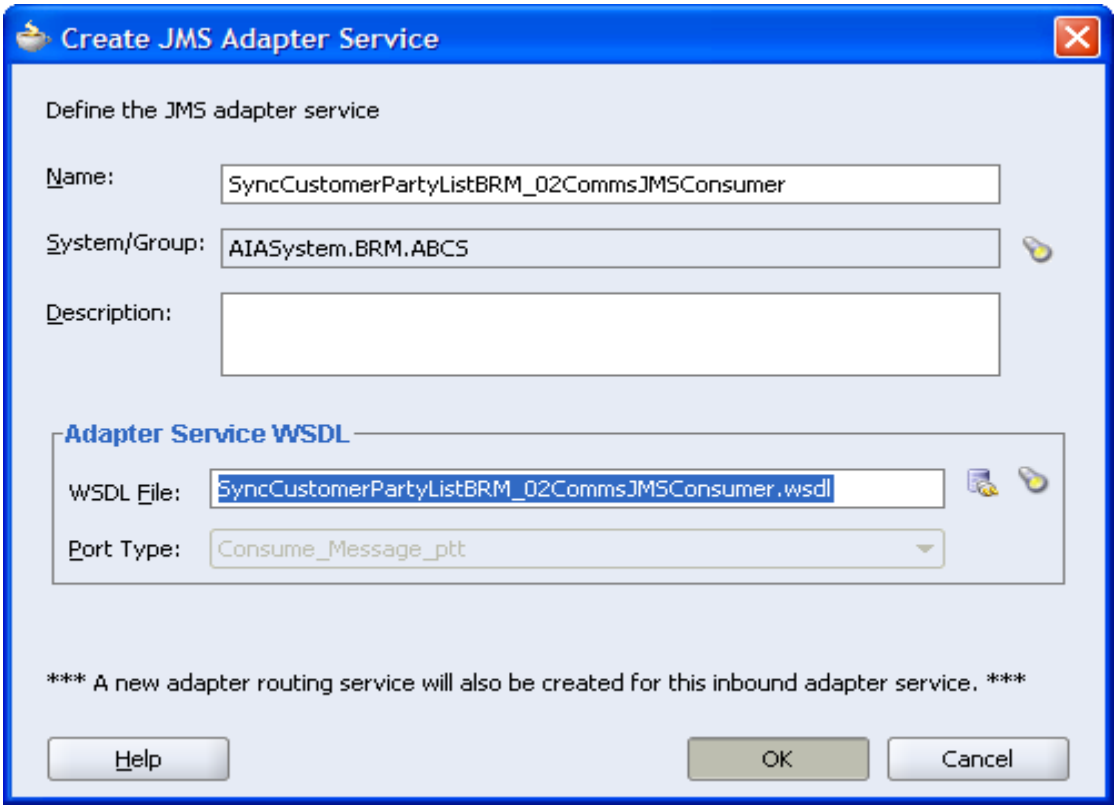

13. Click Next.

14. Provide the schema location, http://[HOST_NAME]:[PORT_NAME]/AIAComponents/EnterpriseObjectLibrary/Industry/Communications/EBO/CustomerParty/V2/CustomerPartyEBM.xsd, and then select SyncCustomerPartyListEBM as the schema element:



15. Click Next.
16. You have finished defining the JMS adapter:

17. Click Finish.
18. Now that the adapter has been defined. Click OK.




19. Double-click the newly created routing service.
20. Expand Routing rules and add a new routing rule.
21. In the Browse Target Service Operation pop-up window, expand BPELSystem/default and then select SyncCustomerPartyListBRMCommsProvABCSImpl from the list.
22. Add the filter expression.

    The filter expression should be similar to the following example:

```
{xref:lookupXRef('CUSTOMERPARTY_ACCOUNTID','COMMON',/imp1:SyncCustomerPartyListEBM/imp1:DataArea/imp1:SyncCustomerPartyList/imp1:CustomerPartyAccount/ns5:Identification/ns5:BusinessComponentID,'BRM_02',false()) != ''};{ namespace ns5=http://xmlns.oracle.com/EnterpriseObjects/Core/Common/V2 namespace imp1=http://xmlns.oracle.com/EnterpriseObjects/Core/EBO/CustomerParty/V2 namespace xref=http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXPathFunctions }
```

23. Add a new xsl file with SetActionCodeandTargetID_[BRM_02].xsl name.
24. Click the SOURCE tab.
25. Add the following contents into the xsl.

(This content is derived from the delivered file <AIA_HOME>\PIPS\Industry\Communications\BRM\JMSAdapterServices\SyncCustomerPartyListBRM_01CommsJMSConsumer\SetActionCodeandTargetID_BRM01.xsl. Please use this file for the latest content.)

```
<?xml version="1.0" encoding="UTF-8" ?>
<?oracle-xsl-mapper <!-- SPECIFICATION OF MAP SOURCES AND TARGETS, DO NOT
MODIFY. -->
  <mapSources>
    <source type="WSDL">
      <schema
location="http://<host>:<port>/AIAComponents/EnterpriseBusinessServiceLibrary/Industry/Communications/EBO/CustomerParty/V2/CommunicationsCustomerPartyEBSV2.wsdl"/>
      <rootElement name="SyncCustomerPartyListEBM"
namespace="http://xmlns.oracle.com/EnterpriseObjects/Core/EBO/CustomerParty/V2"/>
    </source>
  </mapSources>
  <mapTargets>
    <target type="WSDL">
      <schema
location="http://<host>:<port>/orabpel/default/SyncCustomerPartyListBRMCommsProvABCSImpl/1.0/SyncCustomerPartyListBRMCommsProvABCSImpl?wsdl"/>
      <rootElement name="SyncCustomerPartyListEBM"
namespace="http://xmlns.oracle.com/EnterpriseObjects/Core/EBO/CustomerParty/V2"/>
    </target>
  </mapTargets>
  <!-- GENERATED BY ORACLE XSL MAPPER 10.1.3.4.0(build 080718.0645) AT [FRI JUN
12 14:16:50 IST 2009]. -->
?>
<xsl:stylesheet version="1.0"
                xmlns:ns5="http://schemas.oracle.com/service/bpel/common"
                xmlns:ns4="http://xmlns.oracle.com/EnterpriseServices/CustomerParty/V2"
                xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
                xmlns:plnk="http://schemas.xmlsoap.org/ws/2003/05/partner-link/"

xmlns:customerpartyebo="http://xmlns.oracle.com/EnterpriseObjects/Core/EBO/CustomerParty/V2"

xmlns:tns="http://xmlns.oracle.com/ABCSImpl/BRM/Industry/Comms/SyncCustomerPartyListBRMCommsProvABCSImpl/V1"

xmlns:ehdr="http://www.oracle.com/XSL/Transform/java/oracle.tip.esb.server.headers.ESBHeaderFunctions"

xmlns:ns3="http://xmlns.oracle.com/EnterpriseObjects/Core/Custom/EBO/CustomerParty/V2"
                xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                xmlns:svcdoc="http://xmlns.oracle.com/Services/Documentation/V1"
                xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing"
                xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
                xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functions.Xpath20"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXPathFunctio
```

```
ns"
                xmlns:ns0="urn:oasis:names:tc:xacml:2.0:context:schema:cd:04"
                xmlns:ns1="http://xmlns.oracle.com/EnterpriseObjects/Core/Custom/Common/V2"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                xmlns:ora="http://schemas.oracle.com/xpath/extension"
                xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"

xmlns:orcl="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functions.ExtFunc"
                xmlns:ns2="urn:oasis:names:tc:xacml:2.0:policy:schema:cd:04"

xmlns:corecustomerpartyebs="http://xmlns.oracle.com/EnterpriseServices/Core/CustomerParty/V2"

xmlns:aia="http://www.oracle.com/XSL/Transform/java/oracle.apps.aia.core.xpath.AIAFunctions"
                xmlns:corecom="http://xmlns.oracle.com/EnterpriseObjects/Core/Common/V2"
                exclude-result-prefixes="xsl ns4 customerpartyebo ns3 xsd svcdoc wsa soap ns0 ns1 ns2 aia corecustomerpartyebs corecom ns5 plnk tns bpws ehdr hwf xp20 xref ora ids orcl">
   <xsl:template match="/">
      <customerpartyebo:SyncCustomerPartyListEBM>
         <xsl:apply-templates select="/customerpartyebo:SyncCustomerPartyListEBM/corecom:EBMHeader"/>
         <xsl:apply-templates select="/customerpartyebo:SyncCustomerPartyListEBM/customerpartyebo:DataArea"/>
      </customerpartyebo:SyncCustomerPartyListEBM>
   </xsl:template>
   <!-- Template for Stamping Target ID  -->
   <xsl:template match="corecom:EBMHeader">
      <xsl:copy>
         <xsl:apply-templates select="@*|node()"/>
      </xsl:copy>
   </xsl:template>
   <xsl:template match="corecom:EBMHeader/corecom:Sender">
      <xsl:copy-of select="."/>
      <xsl:if test="not(following-sibling::corecom:Target)">
         <corecom:Target>
            <corecom:ID>
               <xsl:text disable-output-escaping="no">BRM_02</xsl:text>
            </corecom:ID>
            <corecom:ApplicationTypeCode>
               <xsl:value-of select="aia:getSystemType('BRM_02')"/>
            </corecom:ApplicationTypeCode>
         </corecom:Target>
      </xsl:if>
   </xsl:template>
   <!-- Template for Customer Account  -->
   <xsl:template match="customerpartyebo:CustomerPartyAccount">
      <customerpartyebo:CustomerPartyAccount>
         <xsl:attribute name="actionCode">UPDATE</xsl:attribute>
         <xsl:apply-templates/>
      </customerpartyebo:CustomerPartyAccount>
   </xsl:template>
   <!-- Template for Identifying BillingProfile -->
   <xsl:template match="customerpartyebo:CustomerPartyBillingProfile">
```

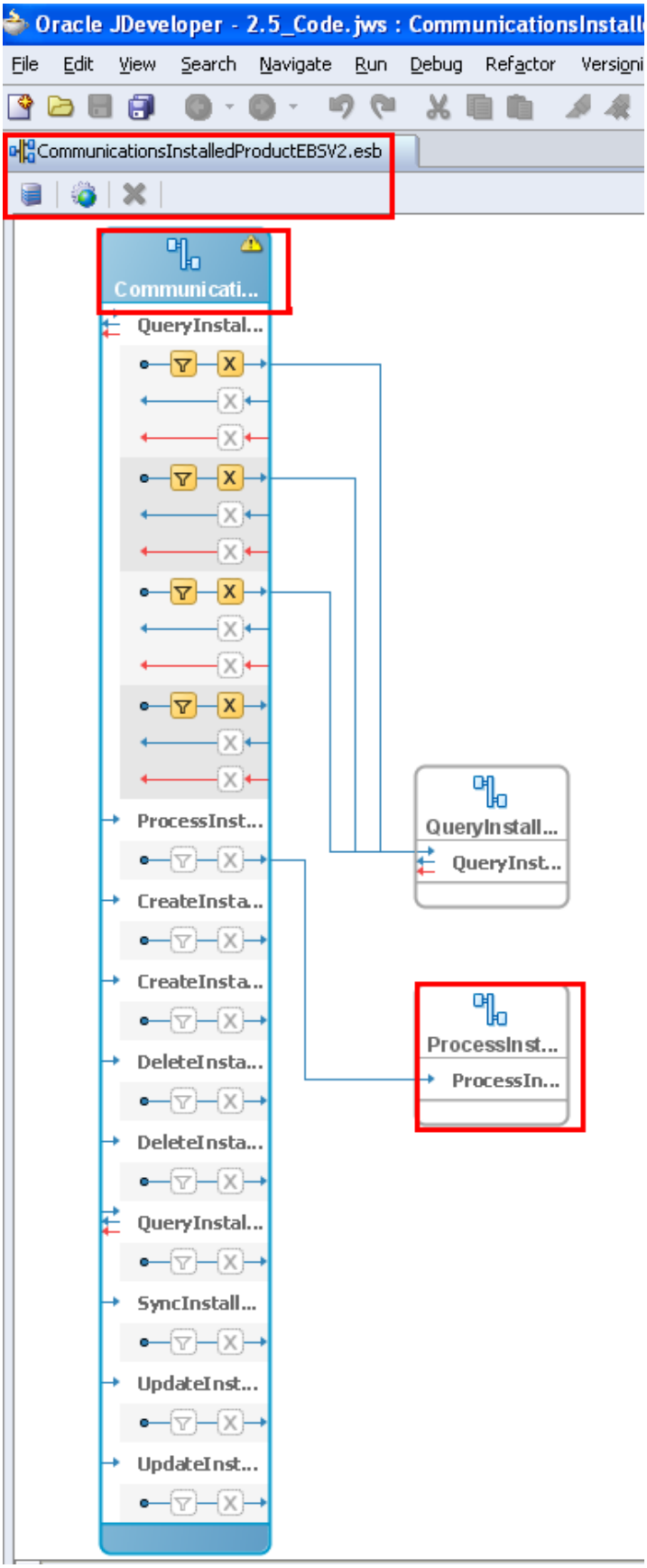
```
        <customerpartyebo:CustomerPartyBillingProfile>
            <xsl:attribute name="actionCode">UPDATE</xsl:attribute>
            <xsl:apply-templates/>
        </customerpartyebo:CustomerPartyBillingProfile>
    </xsl:template>
    <!-- Template for Identifying PaymentProfile -->
    <xsl:template match="customerpartyebo:CustomerPartyPaymentProfile">
        <customerpartyebo:CustomerPartyPaymentProfile>
            <xsl:attribute name="actionCode">UPDATE</xsl:attribute>
            <xsl:apply-templates/>
        </customerpartyebo:CustomerPartyPaymentProfile>
    </xsl:template>
    <!-- Template for Copy-Template -->
    <xsl:template match="@*|node()">
        <xsl:copy>
            <xsl:apply-templates select="@*|node()"/>
        </xsl:copy>
    </xsl:template>
</xsl:stylesheet>
```

26. Change the <host>:<port>.
27. Change the appropriate Oracle BRM system code (for example, BRM_02).
28. Register the ESB after the routing rule has been configured.

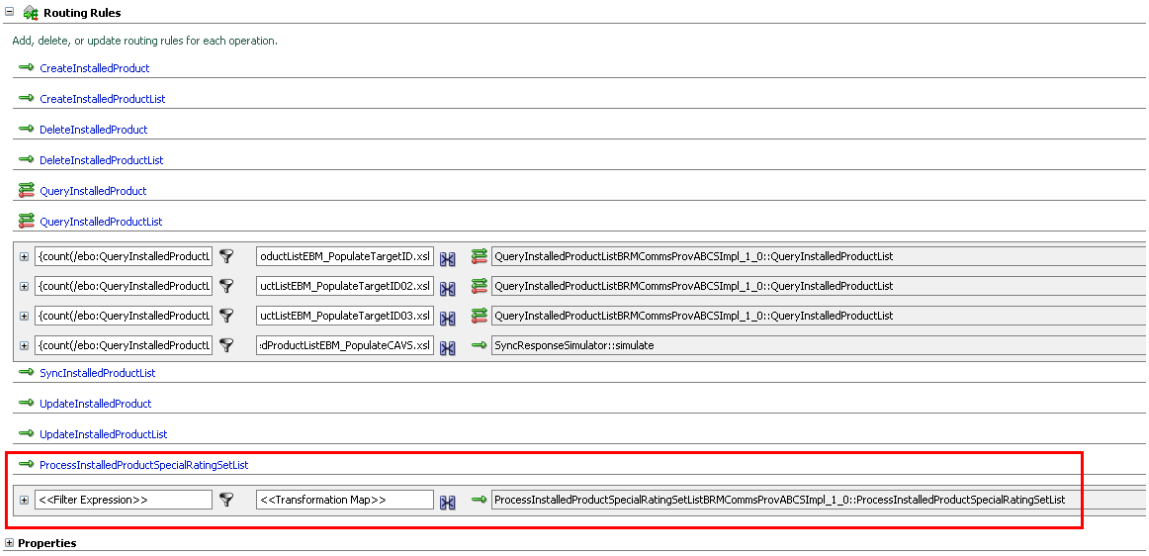
# Adding a New Oracle BRM Instance for the Friends and Family List Update Flow

This flow syncs Friends and Family List updates from Siebel CRM to Oracle BRM. Per the documented restriction, given a Friends and Family List, the flow can sync to one and only one Oracle BRM instance. But for each Oracle BRM instance, one routing rule needs to exist for the ProcessInstalledProductSpecialRatingSetList operation in CommunicationsInstalledProductEBSV2. As delivered, Oracle AIA ships routing rules to route to the BRM_01 instance. To add the additional Oracel BRM instances:

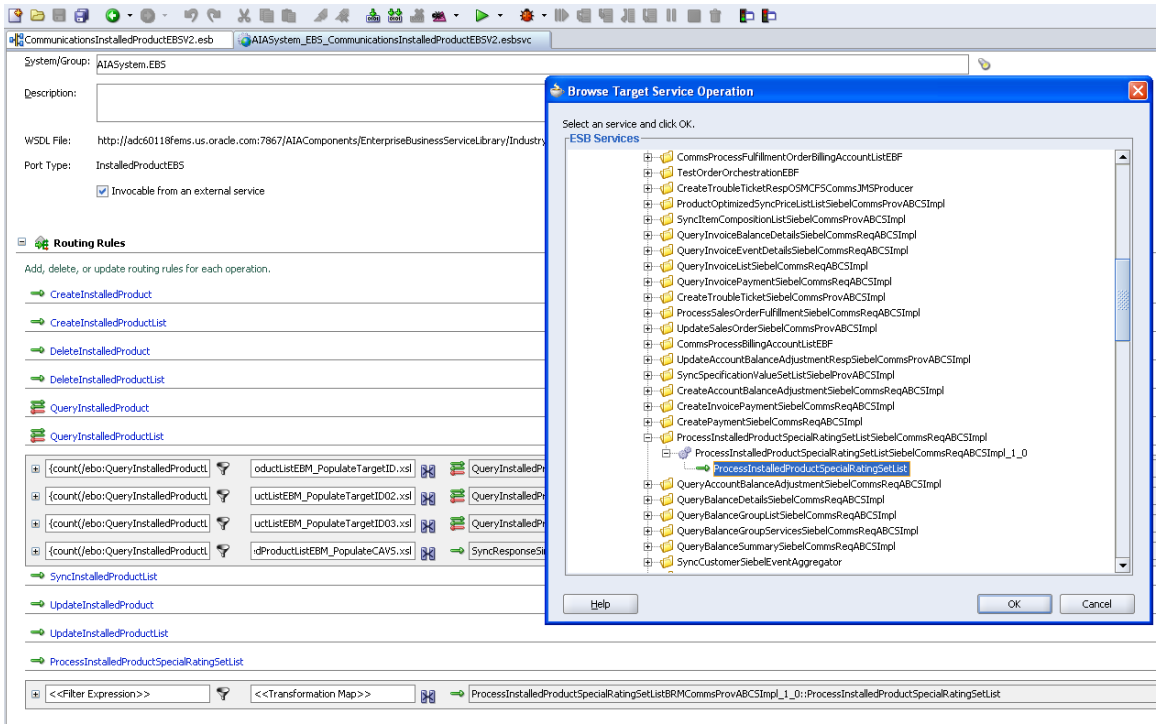
1. Copy the CommunicationsInstalledProductEBSV2 EBS with routing rules from this location: $AIA_HOME/PIPS/Industry/Communications/Setup/OrderToBill/EBS/Billing.
2. Open the ESB project in Oracle JDeveloper.
3. Create new transformation files (.xsl) similar to ProcessInstalledProductSpecialRatingSetList_TargetID.xsl, which populates the target section of the EBMHeader with the appropriate Oracle BRM internal system ID, (for example, BRM_02). Name it appropriately (such as ProcessInstalledProductSpecialRatingSetList_PopulateTargetID02.xsl.)
4. Click the portion highlighted in boxes to open the routing service.



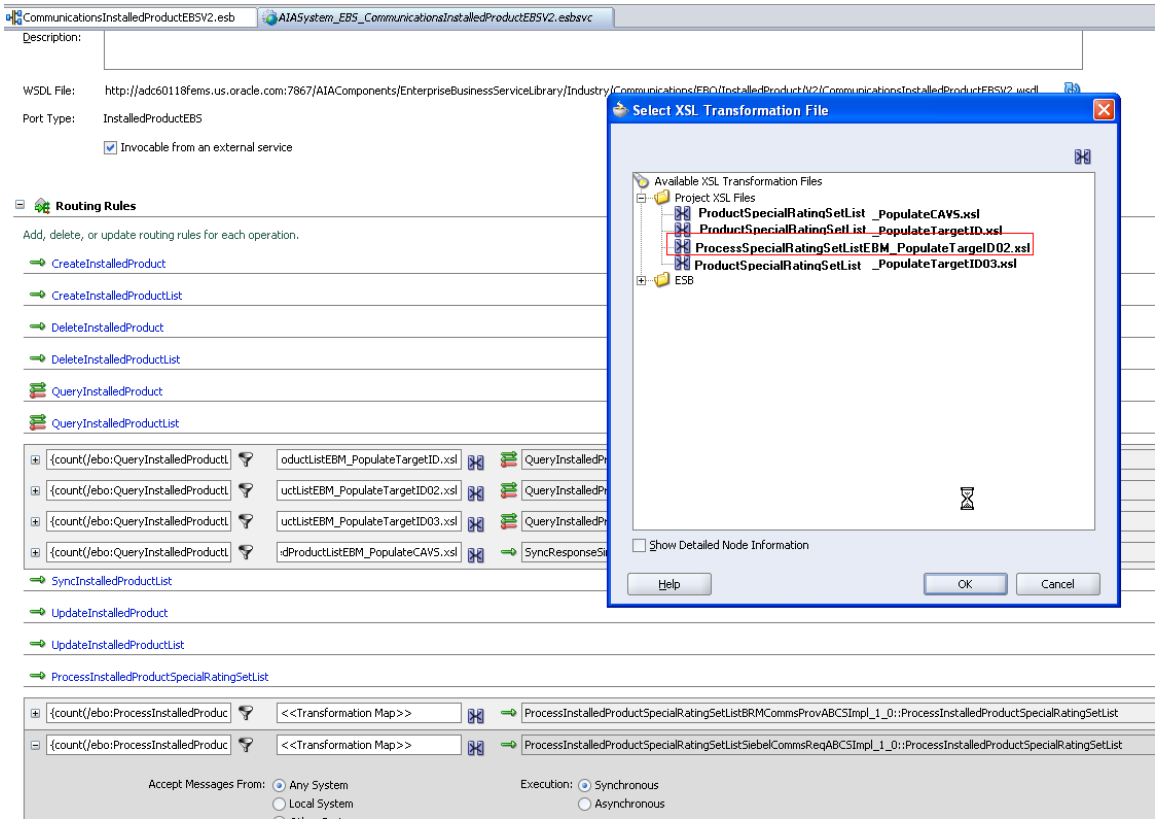
The routing service looks like this:




5. For ProcessInstalledProductSpecialRatingSetList operation, click + against the operation to create a routing rule.
6. The target service is the same as what the BRM_01 routing rule is using. This can be selected from the services available under the same project.



7. After the target is selected, the transformation file must be selected: Click the button indicating the Transformations. A pop-up window, Request Transformation Map, appears. Select Use Existing Mapper file, and then select the mapping file for BRM_02 (for example, ProcessInstalledProductSpecialRatingSetList_PopulateTargetID02.xsl).




8. In the Filter expression box, copy the filter expression used for the BRM_01 routing rule and replace BRM_01 with BRM_02 and add the filter expression.

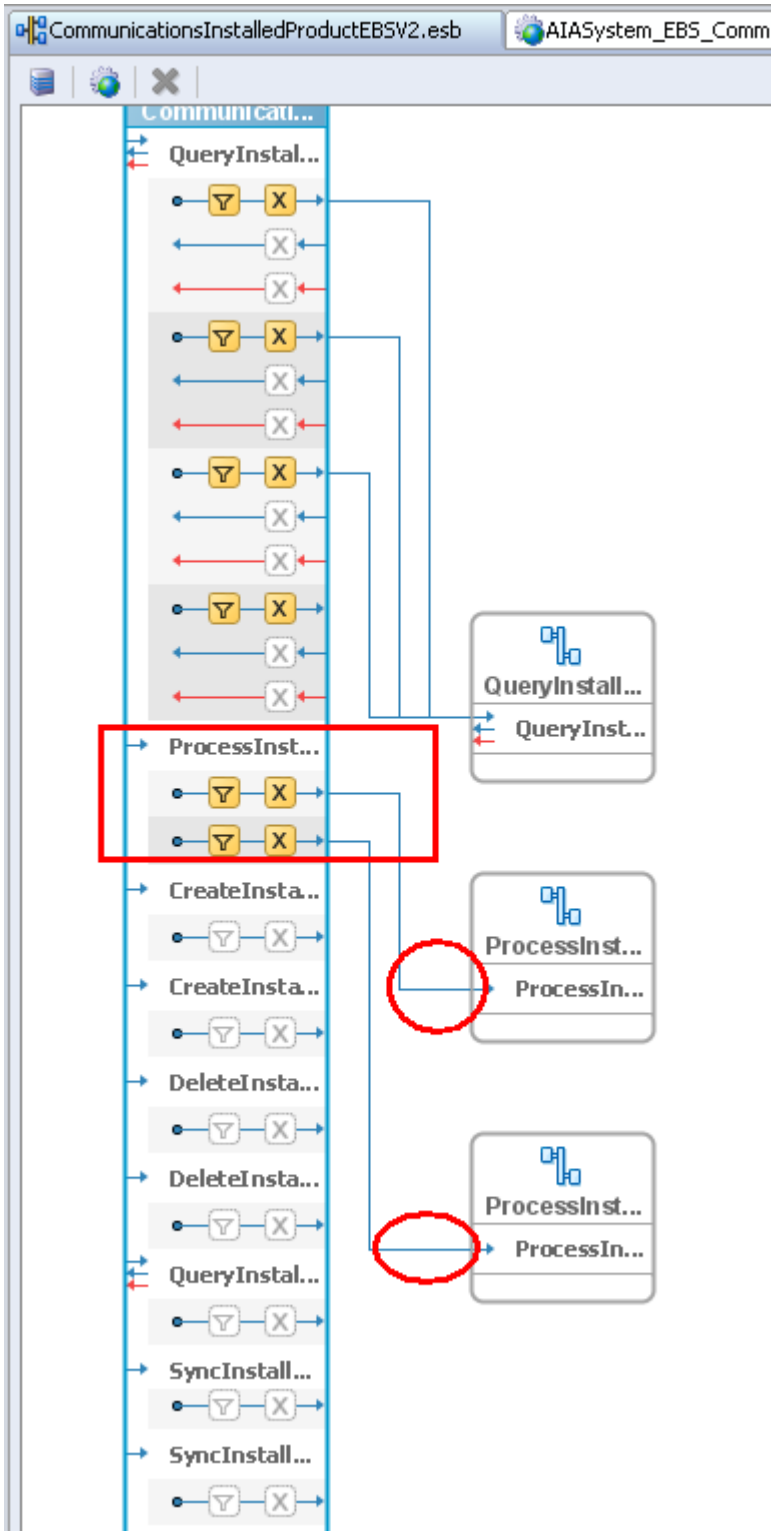
For example, this will be replaced with what follows it:

```
{count(/ebo:ProcessInstalledProductSpecialRatingSetListEBM/corecom:EBMHeader/corecom:MessageProcessingInstruction/corecom:EnvironmentCode[text()='CAVS'])=0) and xref:lookupXRef('INSTALLEDPRODUCT_ID','COMMON',/ebo:ProcessInstalledProductSpecialRatingSetListEBM/ebo:DataArea/ebo:ProcessInstalledProductSpecialRatingSetList/corecom:InstalledProductIdentification/corecom:BusinessComponentID,'BRM_01',false()) != ''};{ namespace ebo=http://xmlns.oracle.com/EnterpriseObjects/Core/EBO/InstalledProduct/V2 namespace corecom=http://xmlns.oracle.com/EnterpriseObjects/Core/Common/V2 namespace xref=http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXPathFunctions }
```

```
{count(/ebo:ProcessInstalledProductSpecialRatingSetListEBM/corecom:EBMHeader/corecom:MessageProcessingInstruction/corecom:EnvironmentCode[text()='CAVS'])=0) and xref:lookupXRef('INSTALLEDPRODUCT_ID','COMMON',/ebo:ProcessInstalledProductSpecialRatingSetListEBM/ebo:DataArea/ebo:ProcessInstalledProductSpecialRatingSetList/corecom:InstalledProductIdentification/corecom:BusinessComponentID,'BRM_01',false()) != ''};{ namespace ebo=http://xmlns.oracle.com/EnterpriseObjects/Core/EBO/InstalledProduct/V2 namespace corecom=http://xmlns.oracle.com/EnterpriseObjects/Core/Common/V2 namespace
```

xref=http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXPathFunctions }

After the routing rules are created, the ESB should look like this:



9. Save the files.
10. Make sure that the .esbsvc file of that ESB has the targetOperation qname attribute referring to the BPEL with the correct service qname.
11. Deploy the ESB using ANT.
12. Alternately, if you want to register the ESB using JDeveloper, you may want to select the routing services from the Integration server (services at ESB server connection) instead of picking them from the local project WSDL (Services in Project).

# Adding a New Oracle BRM instance: Configure Product Sync

If Oracle BRM is the product master and new products and discounts need to be synced from the new Oracle BRM instance into Siebel CRM, then follow these steps to add additional consumers for Product and Discount sync. Current consumers for BRM_01 instance are SyncProductInfoChangeBRMAQ and SyncDiscountInfoChangeBRMAQ.

## Creating a New AQ Adapter Connection Factory

In this procedure, you would create the Oracle Advanced Queuing (AQ) connection factory for connecting to the AQ Queue in Oracle BRM, where product or discount update events are published by the new Oracle BRM instance.

1. Create the connection pool and data sources in <SOA_HOME>/j2ee/<oc4j_soa>/config/data-sources.xml as follows. Provide the unique names, such as BRMConnectionPool2, BRMDataSource2, and jdbc/BRMEventSyncAQ2. (Be sure to change the user, password, and URL to reflect the correct database instances where Oracle BRM is installed.)

```
   <connection-pool name="BRMConnectionPool2">
      <connection-factory factory-
class="oracle.jdbc.xa.client.OracleXADataSource" user="pin7852"
password="pin7852" url="jdbc:oracle:thin:@maguro-
1.us.oracle.com:1521:pindb" commit-record-table-name=""/>
   </connection-pool>
```

```
   <managed-data-source connection-pool-name="BRMConnectionPool2"
jndi-name="jdbc/BRMEventSyncAQ2" name="BRMDataSource2"/>
```

2. Go to <ORACLE_HOME}/j2ee/oc4j_soa/application-deployments/default/AqAdapter/oc4j-ra.xml and create the entry shown here. eis/AQ/PortalEventSyncAQ2 is the unique name for the AQ queue connection factory for the new Oracle BRM instance. jdbc/BRMEventSyncAQ2 is the data source created in Step 1.

```
      <connector-factory location="eis/AQ/PortalEventSyncAQ2"
connector-name="AQ Adapter">
            <config-property name="xADataSourceName"
value="jdbc/BRMEventSyncAQ2"/>
            <config-property name="dataSourceName" value=""/>
            <config-property name="connectionString" value=""/>
            <config-property name="userName" value=""/>
            <config-property name="password" value=""/>
            <config-property name="defaultNChar" value="false"/>
```

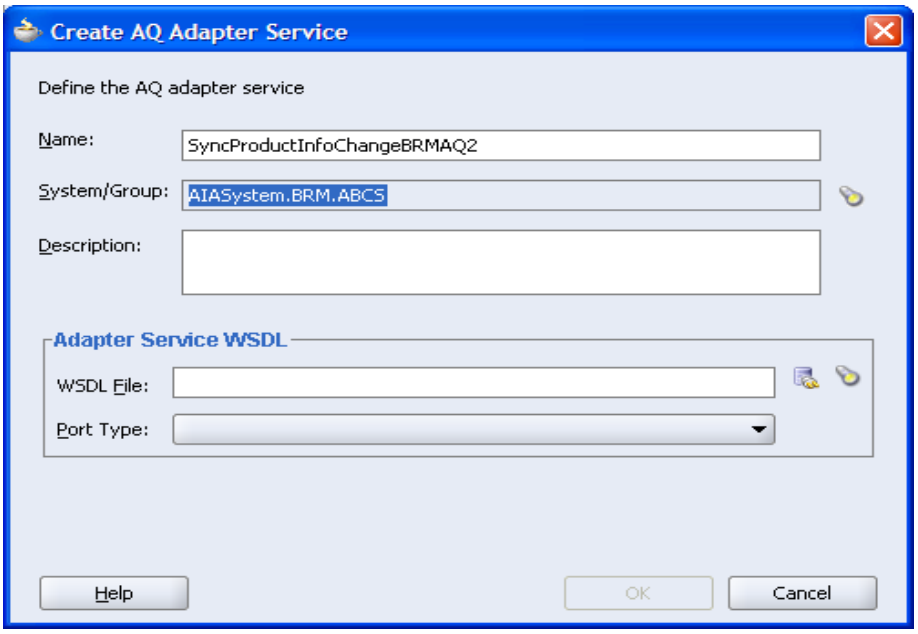
```
            <config-property name="useDefaultConnectionManager"
value="false"/>
            <connection-pooling use="none">
            </connection-pooling>
            <security-config use="none">
            </security-config>
         </connector-factory>
```

## Creating a New AQ Adapter for the Product Sync

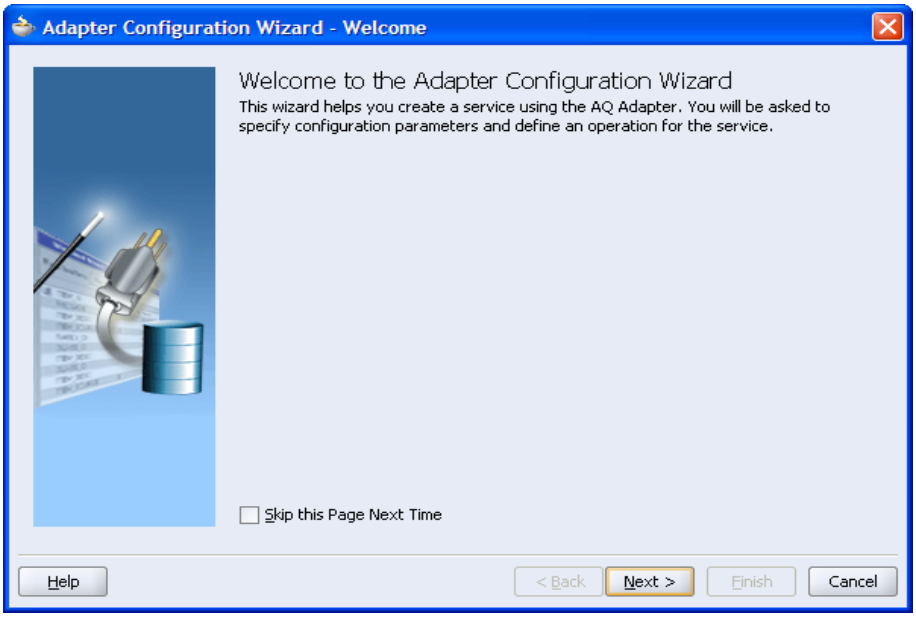
To create a new ESB project in JDeveloper:

1. Create a new ESB project in JDeveloper by the name SyncProductInfoChangeBRMAQ2.
2. In the ESB file that is generated, create a new AQ Adapter. The System/Group should be AIASystem.BRM.ABCS. The name should be SyncProductInfoChangeBRMAQ2.

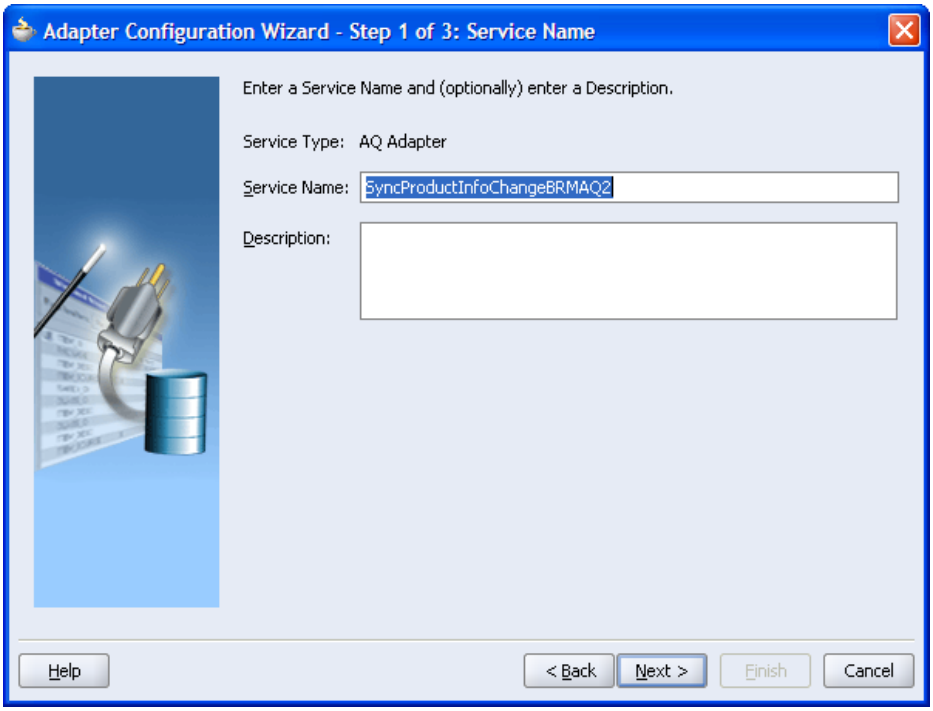
   The Create AQ Adapter Service screen appears:



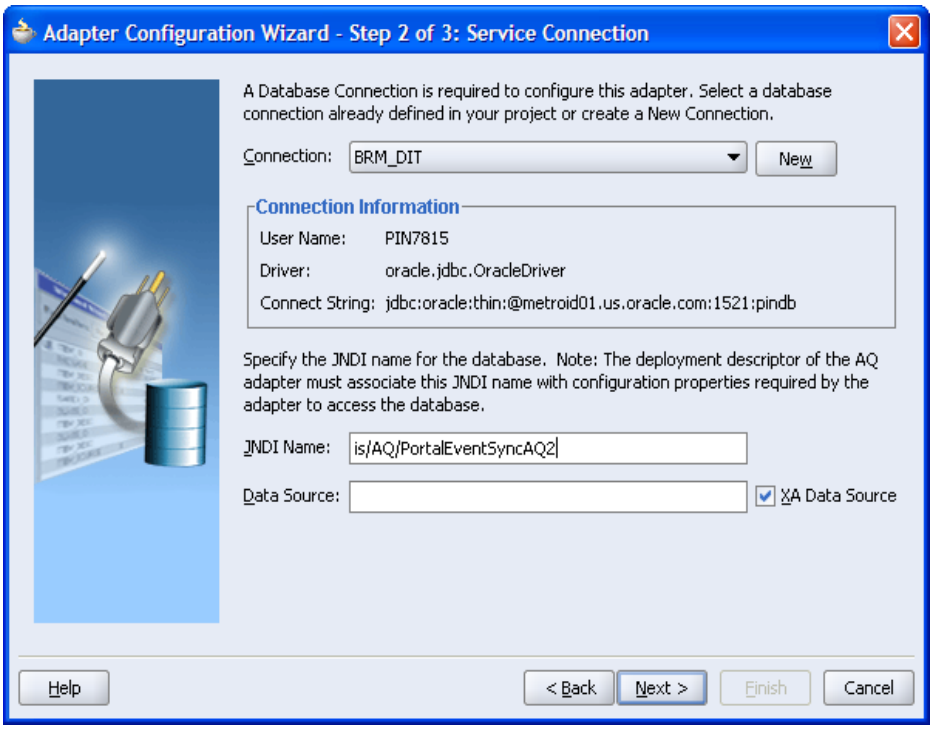

3. Click the Configure Adapter Service WSDL button next to the WSDL File field.
4. The following screen appears:



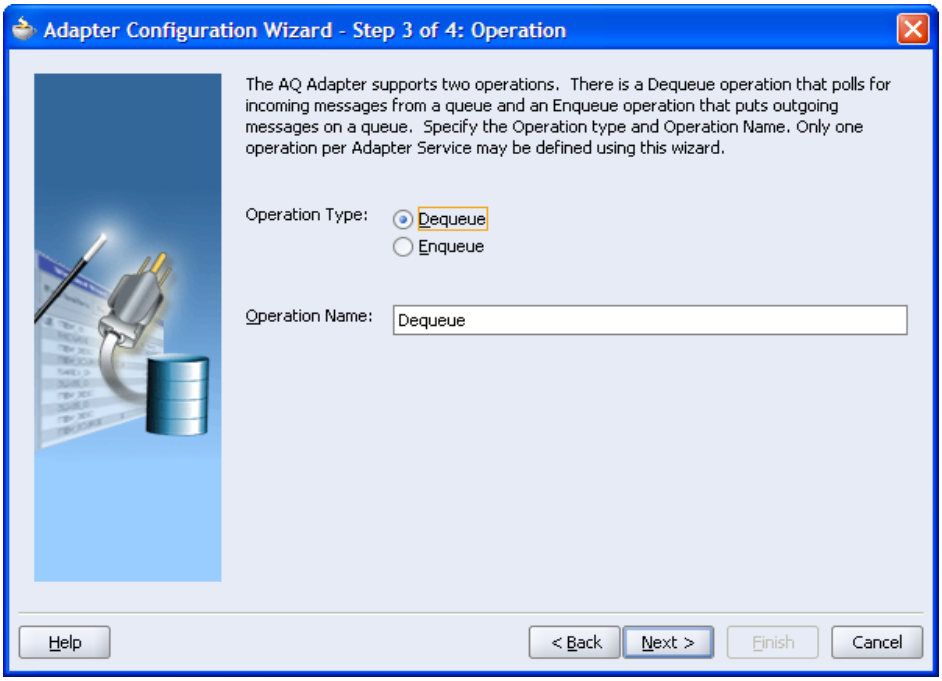

5. Click Next. The Wizard screen appears:



6. Click Next. The following screen appears:

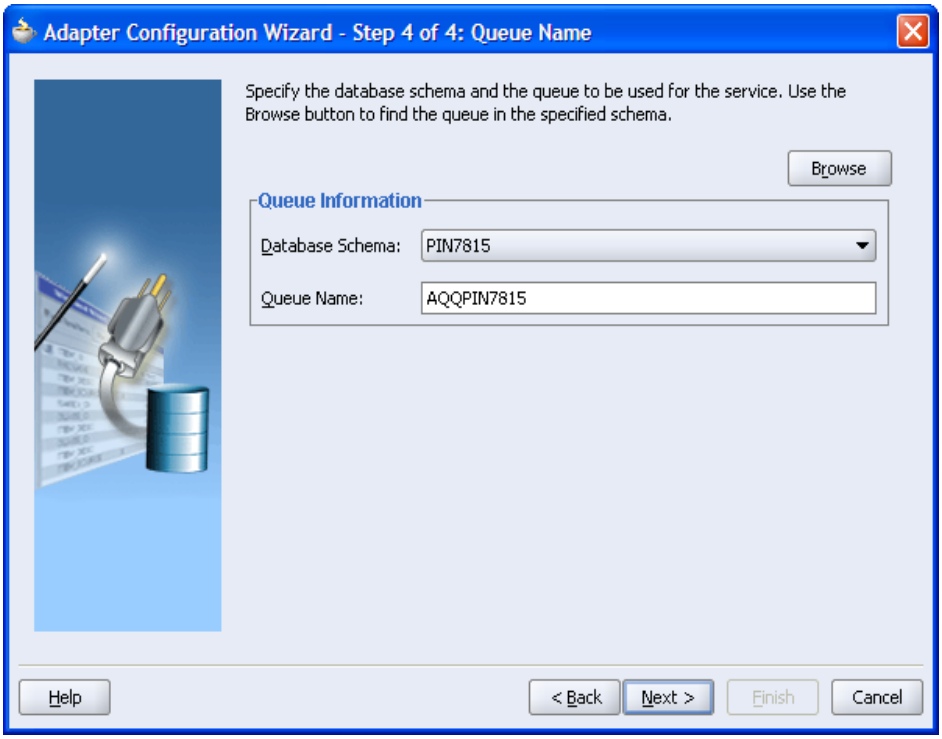



7. Select the BRM Database Connection and enter the JNDI Name from **Creating a New AQ Adapter Connection Factory section.**
8. Click Next.
9. Select the operation type Dequeue.



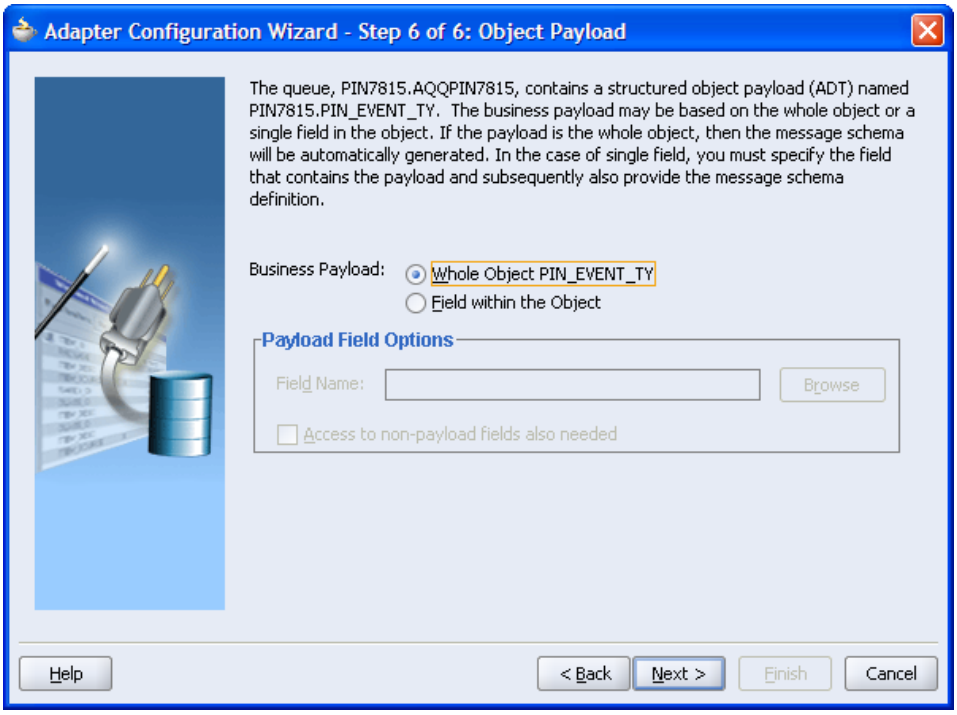

10. Click Next.

11. Select the additional Oracle BRM instance's database schema and enter the queue name configured for this Oracle BRM instance.

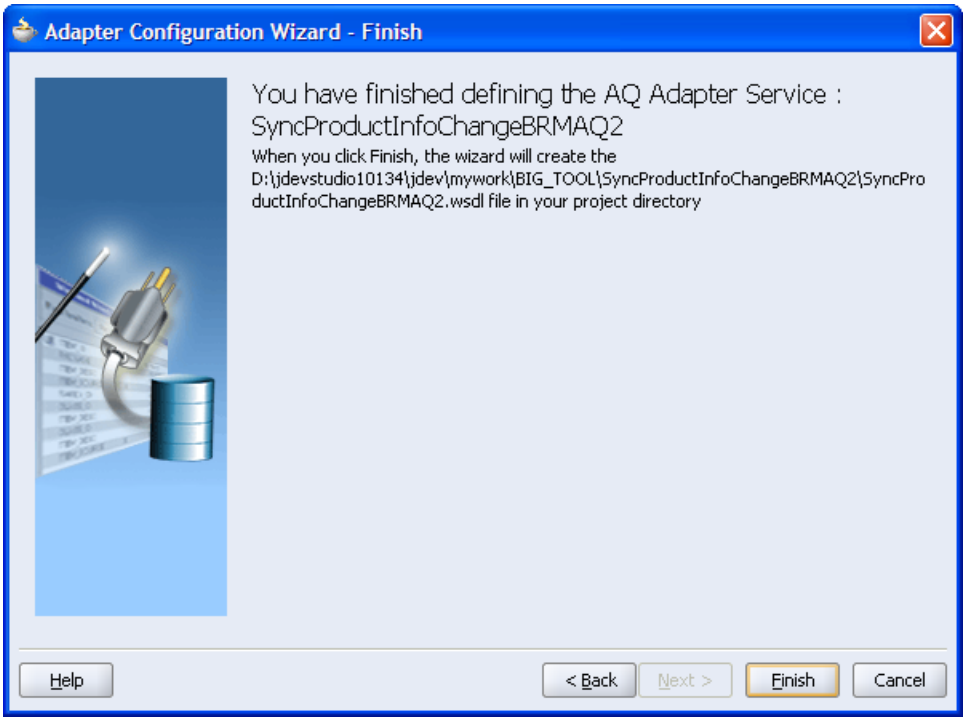



12. Click Next.
13. Enter ProductInfoChange as the correlation ID and click Next.

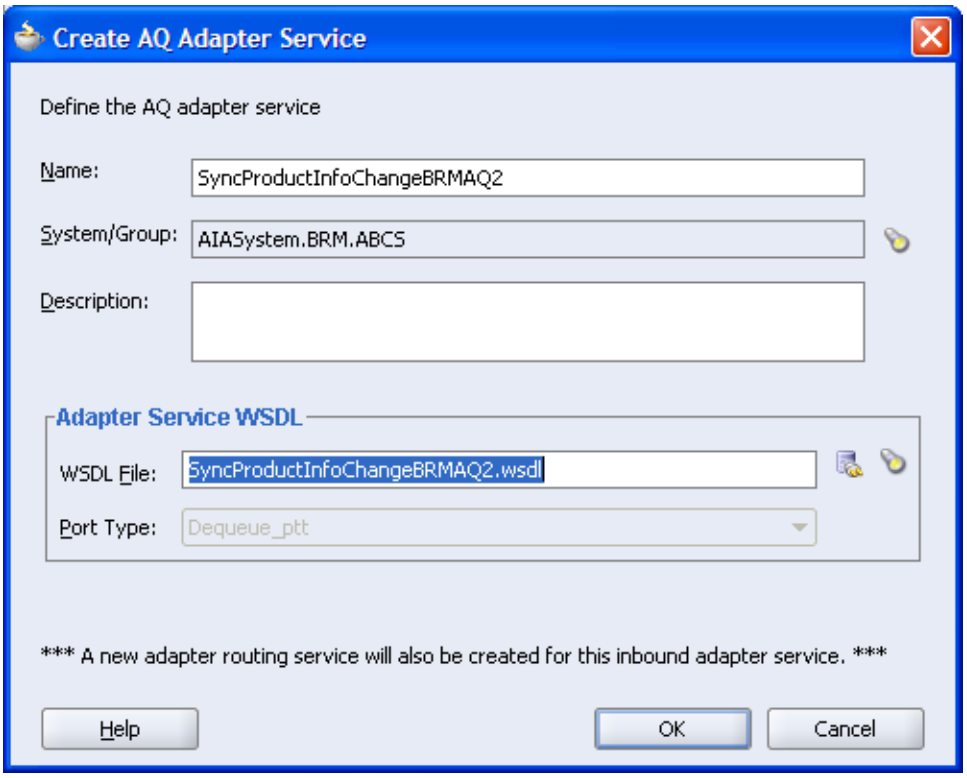

14. Select Whole Object PIN_EVENT_TY for the Business Payload option and click Next.




15. Click Finish.




16. The AQ Adapter service has been created; click OK.




17. Double-click the newly created routing service.
18. Expand Routing rules and add the new routing rule.
19. In the Browse Target Service Operation pop-up window, expand BPELSystem/Default/ SyncProductBRMCommsReqABCSImpl and select SyncProduct operation.
20. For transformation, add a new transformation file with a unique name such as PIN_EVENT_TY_To_ProductInfoChange2.xsl.
21. Click the Source tab.
22. Add the following content in the source. Make sure that you change the highlighted values appropriately.

    (This content is derived from the delivered file <AIA_HOME>\PIPS\Industry\Communications\BRM\AQAdapterServices\SyncProductInfoChangeBRMAQ\PIN_EVENT_TY_To_ProductInfoChange.xsl. Please use this file for the latest content.)

```
<?xml version="1.0" encoding="UTF-8" ?>
<?oracle-xsl-mapper <!-- SPECIFICATION OF MAP SOURCES AND TARGETS, DO NOT MODIFY. --
>
 <mapSources>
  <source type="WSDL">
   <schema location="SyncProductInfoChangeBRMAQ2.wsdl"/>
   <rootElement name="PIN_EVENT_TY" namespace="http://xmlns.oracle.com/xdb/PIN7851"/>
  </source>
```

```
  </mapSources>
  <mapTargets>
    <target type="WSDL">
      <schema
location="http://<host>:<port>/orabpel/default/SyncProductBRMCommsReqABCSImpl/1.0/SyncProductBRMCommsReqABCSImpl?wsdl"/>
      <rootElement name="ProductInfoChange"
namespace="http://www.portal.com/schemas/CRMSync"/>
    </target>
  </mapTargets>
  <!-- GENERATED BY ORACLE XSL MAPPER 10.1.3.4.0(build 080718.0645) AT [WED JUL 22
12:51:15 IST 2009]. -->
?>
<xsl:stylesheet version="1.0"
                xmlns:ns5="http://schemas.oracle.com/service/bpel/common"
                xmlns:svcdoc="http://xmlns.oracle.com/Services/Documentation/V2"
                xmlns:plt="http://schemas.xmlsoap.org/ws/2003/05/partner-link/"

xmlns:ehdr="http://www.oracle.com/XSL/Transform/java/oracle.tip.esb.server.headers.ESBHeaderFunctions"
                xmlns:jca="http://xmlns.oracle.com/pcbpel/wsdl/jca/"
                xmlns:ns0="http://www.w3.org/2001/XMLSchema"
                xmlns:ns8="http://xmlns.oracle.com/EnterpriseObjects/Core/CommonEBO/V1"
                xmlns:ns3="http://xmlns.oracle.com/EnterpriseObjects/Core/Custom/CommonEBO/V1"

xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functions.Xpath20"
                xmlns:tns="http://xmlns.oracle.com/pcbpel/adapter/aq/SyncProductInfoChangeBRMAQ2/"
                xmlns:ns4="http://xmlns.oracle.com/EnterpriseObjects/Core/Custom/Common/V2"
                xmlns:ora="http://schemas.oracle.com/xpath/extension"
                xmlns:pns1="http://xmlns.oracle.com/SyncProductBRMCommsReqABCSImpl/correlationset"

xmlns:orcl="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functions.ExtFunc"
                xmlns:ns6="urn:oasis:names:tc:xacml:2.0:policy:schema:cd:04"

xmlns:itemcompositionebo="http://xmlns.oracle.com/EnterpriseObjects/Core/EBO/ItemComposition/V1"
                xmlns:ns9="http://xmlns.oracle.com/EnterpriseObjects/Core/Common/V2"
                xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
                xmlns:pc="http://xmlns.oracle.com/pcbpel/"
                xmlns:str="http://www.oracle.com/XSL/Transform/java/java.lang.String"
                xmlns:ns7="http://schemas.xmlsoap.org/ws/2003/03/addressing"
                xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
                xmlns:obj1="http://xmlns.oracle.com/xdb/PIN7851"
```

```
xmlns:ns1="http://xmlns.oracle.com/ABCSImpl/BRM/Industry/Comms/SyncProductBRMCommsReqABCSImpl/V1"
                xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXPathFunctions"
                xmlns:brmproductabo="http://www.portal.com/schemas/CRMSync"
                xmlns:ns2="urn:oasis:names:tc:xacml:2.0:context:schema:cd:04"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"
                xmlns:hdr="http://xmlns.oracle.com/pcbpel/adapter/aq/inbound/"

xmlns:ns10="http://xmlns.oracle.com/EnterpriseObjects/Core/Custom/EBO/ItemComposition/V1"
                exclude-result-prefixes="xsl plt str jca ns0 tns pc obj1 hdr ns5 svcdoc ns8 ns3 ns4 pns1 ns6 itemcompositionebo ns9 bpws ns7 ns1 soap brmproductabo ns2 ns10 ehdr xp20 ora orcl bpws hwf xref ids">
  <xsl:template match="/">
    <xsl:choose>
      <xsl:when test='/obj1:PIN_EVENT_TY/FLIST_BUF != ""'>
        <!-- Added to support "&" character. orcl:parseEscapedXML() fails If your input contains "&". -->
        <xsl:variable name="AmpersandReplacement">&amp;amp;amp;</xsl:variable>
        <xsl:variable name="EscapedRequestEBM_fixed"

select="str:replaceAll(string(/obj1:PIN_EVENT_TY/FLIST_BUF),'\u0026(amp;)*+',$AmpersandReplacement)"/>
        <xsl:variable name="EVENT_msg"
                      select="orcl:parseEscapedXML($EscapedRequestEBM_fixed)"/>
        <brmproductabo:ProductInfoChange>
          <xsl:attribute name="brmproductabo:InstanceId">
            <xsl:value-of select="$EVENT_msg/@InstanceId"/>
          </xsl:attribute>
          <brmproductabo:AccountObj>
            <xsl:value-of select="$EVENT_msg/brmproductabo:AccountObj"/>
          </brmproductabo:AccountObj>
          <xsl:for-each select="$EVENT_msg/brmproductabo:ProductInfo">
            <brmproductabo:ProductInfo>
              <brmproductabo:ProductPoid>
                <xsl:value-of select="current()/brmproductabo:ProductPoid"/>
              </brmproductabo:ProductPoid>
              <brmproductabo:Name>
                <xsl:value-of select="current()/brmproductabo:Name"/>
              </brmproductabo:Name>
              <brmproductabo:ProductStartTime>
```

```
  <xsl:value-of select="current()/brmproductabo:ProductStartTime"/>
</brmproductabo:ProductStartTime>
<brmproductabo:ProductEndTime>
  <xsl:value-of select="current()/brmproductabo:ProductEndTime"/>
</brmproductabo:ProductEndTime>
<brmproductabo:PermittedTypes>
  <xsl:value-of select="current()/brmproductabo:PermittedTypes"/>
</brmproductabo:PermittedTypes>
<brmproductabo:Priority>
  <xsl:value-of select="current()/brmproductabo:Priority"/>
</brmproductabo:Priority>
<brmproductabo:Type>
  <xsl:value-of select="current()/brmproductabo:Type"/>
</brmproductabo:Type>
<brmproductabo:Description>
  <xsl:value-of select="current()/brmproductabo:Description"/>
</brmproductabo:Description>
<xsl:for-each select="current()/brmproductabo:UsageMap">
  <brmproductabo:UsageMap>
    <brmproductabo:UsageEventType>
      <xsl:value-of select="current()/brmproductabo:UsageEventType"/>
    </brmproductabo:UsageEventType>
    <brmproductabo:RatePlanName>
      <xsl:value-of select="current()/brmproductabo:RatePlanName"/>
    </brmproductabo:RatePlanName>
    <brmproductabo:RumName>
      <xsl:value-of select="current()/brmproductabo:RumName"/>
    </brmproductabo:RumName>
    <brmproductabo:RatePlan>
      <brmproductabo:Name>
        <xsl:value-of select="current()/brmproductabo:RatePlan/brmproductabo:Name"/>
      </brmproductabo:Name>
      <brmproductabo:Currency>
        <xsl:value-of select="current()/brmproductabo:RatePlan/brmproductabo:Currency"/>
      </brmproductabo:Currency>
      <xsl:for-each select="current()/brmproductabo:RatePlan/brmproductabo:RateDetails">
        <brmproductabo:RateDetails>
          <xsl:for-each select="current()/brmproductabo:QuantityTiers">
            <brmproductabo:QuantityTiers>
```

```
                    <brmproductabo:StepMin>
                      <xsl:value-of select="current()/brmproductabo:StepMin"/>
                    </brmproductabo:StepMin>
                    <brmproductabo:StepMax>
                      <xsl:value-of select="current()/brmproductabo:StepMax"/>
                    </brmproductabo:StepMax>
                    <xsl:for-each select="current()/brmproductabo:BalImpacts">
                      <brmproductabo:BalImpacts>
                        <brmproductabo:ResourceId>
                          <xsl:value-of select="current()/brmproductabo:ResourceId"/>
                        </brmproductabo:ResourceId>
                        <brmproductabo:FixedAmount>
                          <xsl:value-of select="current()/brmproductabo:FixedAmount"/>
                        </brmproductabo:FixedAmount>
                        <brmproductabo:ScaledAmount>
                          <xsl:value-of select="current()/brmproductabo:ScaledAmount"/>
                        </brmproductabo:ScaledAmount>
                        <brmproductabo:ScaledUnit>
                          <xsl:value-of select="current()/brmproductabo:ScaledUnit"/>
                        </brmproductabo:ScaledUnit>
                      </brmproductabo:BalImpacts>
                    </xsl:for-each>
                  </brmproductabo:QuantityTiers>
                </xsl:for-each>
              </brmproductabo:RateDetails>
            </xsl:for-each>
          </brmproductabo:RatePlan>
        </brmproductabo:UsageMap>
      </xsl:for-each>
    </brmproductabo:ProductInfo>
  </xsl:for-each>
 </brmproductabo:ProductInfoChange>
</xsl:when>
<xsl:otherwise>
 <xsl:variable name="EVENT_msg"
            select="orcl:parseEscapedXML(/obj1:PIN_EVENT_TY/LARGE_FLIST_BUF)"/>
 <brmproductabo:ProductInfoChange>
  <xsl:attribute name="brmproductabo:InstanceId">
   <xsl:value-of select="$EVENT_msg/@InstanceId"/>
```

```
        </xsl:attribute>
        <brmproductabo:AccountObj>
          <xsl:value-of select="$EVENT_msg/brmproductabo:AccountObj"/>
        </brmproductabo:AccountObj>
        <xsl:for-each select="$EVENT_msg/brmproductabo:ProductInfo">
          <brmproductabo:ProductInfo>
            <brmproductabo:ProductPoid>
              <xsl:value-of select="current()/brmproductabo:ProductPoid"/>
            </brmproductabo:ProductPoid>
            <brmproductabo:Name>
              <xsl:value-of select="current()/brmproductabo:Name"/>
            </brmproductabo:Name>
            <brmproductabo:ProductStartTime>
              <xsl:value-of select="current()/brmproductabo:ProductStartTime"/>
            </brmproductabo:ProductStartTime>
            <brmproductabo:ProductEndTime>
              <xsl:value-of select="current()/brmproductabo:ProductEndTime"/>
            </brmproductabo:ProductEndTime>
            <brmproductabo:PermittedTypes>
              <xsl:value-of select="current()/brmproductabo:PermittedTypes"/>
            </brmproductabo:PermittedTypes>
            <brmproductabo:Priority>
              <xsl:value-of select="current()/brmproductabo:Priority"/>
            </brmproductabo:Priority>
            <brmproductabo:Type>
              <xsl:value-of select="current()/brmproductabo:Type"/>
            </brmproductabo:Type>
            <brmproductabo:Description>
              <xsl:value-of select="current()/brmproductabo:Description"/>
            </brmproductabo:Description>
            <xsl:for-each select="current()/brmproductabo:UsageMap">
              <brmproductabo:UsageMap>
                <brmproductabo:UsageEventType>
                  <xsl:value-of select="current()/brmproductabo:UsageEventType"/>
                </brmproductabo:UsageEventType>
                <brmproductabo:RatePlanName>
                  <xsl:value-of select="current()/brmproductabo:RatePlanName"/>
                </brmproductabo:RatePlanName>
                <brmproductabo:RumName>
```

```
    <xsl:value-of select="current()/brmproductabo:RumName"/>
  </brmproductabo:RumName>
  <brmproductabo:RatePlan>
    <brmproductabo:Name>
      <xsl:value-of select="current()/brmproductabo:RatePlan/brmproductabo:Name"/>
    </brmproductabo:Name>
    <brmproductabo:Currency>
      <xsl:value-of select="current()/brmproductabo:RatePlan/brmproductabo:Currency"/>
    </brmproductabo:Currency>
    <xsl:for-each select="current()/brmproductabo:RatePlan/brmproductabo:RateDetails">
      <brmproductabo:RateDetails>
        <xsl:for-each select="current()/brmproductabo:QuantityTiers">
          <brmproductabo:QuantityTiers>
            <brmproductabo:StepMin>
              <xsl:value-of select="current()/brmproductabo:StepMin"/>
            </brmproductabo:StepMin>
            <brmproductabo:StepMax>
              <xsl:value-of select="current()/brmproductabo:StepMax"/>
            </brmproductabo:StepMax>
            <xsl:for-each select="current()/brmproductabo:BalImpacts">
              <brmproductabo:BalImpacts>
                <brmproductabo:ResourceId>
                  <xsl:value-of select="current()/brmproductabo:ResourceId"/>
                </brmproductabo:ResourceId>
                <brmproductabo:FixedAmount>
                  <xsl:value-of select="current()/brmproductabo:FixedAmount"/>
                </brmproductabo:FixedAmount>
                <brmproductabo:ScaledAmount>
                  <xsl:value-of select="current()/brmproductabo:ScaledAmount"/>
                </brmproductabo:ScaledAmount>
                <brmproductabo:ScaledUnit>
                  <xsl:value-of select="current()/brmproductabo:ScaledUnit"/>
                </brmproductabo:ScaledUnit>
              </brmproductabo:BalImpacts>
            </xsl:for-each>
          </brmproductabo:QuantityTiers>
        </xsl:for-each>
      </brmproductabo:RateDetails>
    </xsl:for-each>
```

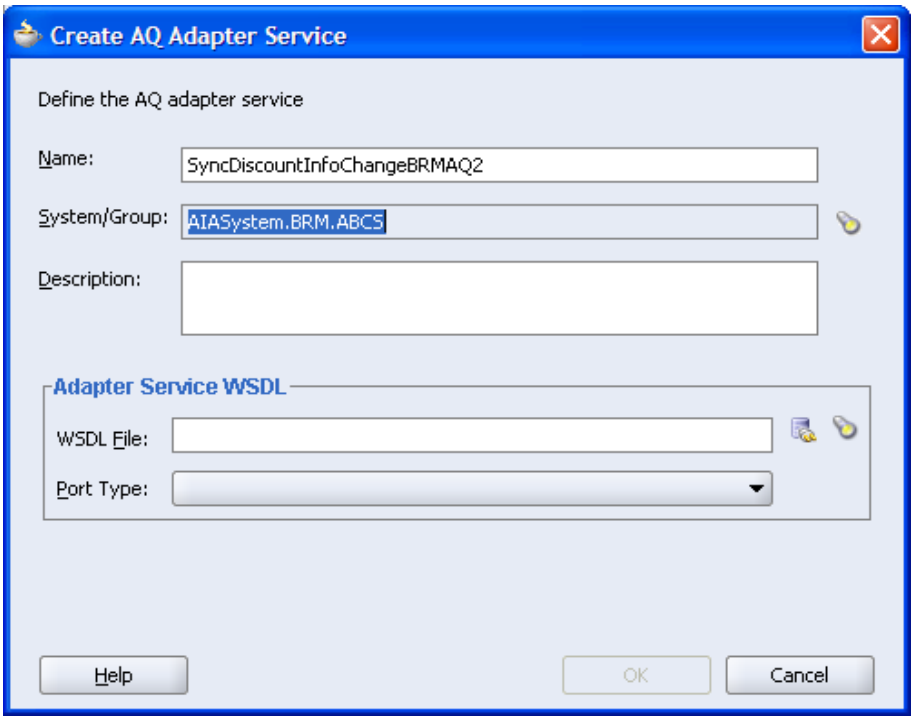
```
            </brmproductabo:RatePlan>
          </brmproductabo:UsageMap>
        </xsl:for-each>
      </brmproductabo:ProductInfo>
    </xsl:for-each>
  </brmproductabo:ProductInfoChange>
 </xsl:otherwise>
</xsl:choose>
</xsl:template>
</xsl:stylesheet>
```

23. Change the <host>, <port>, SyncProductInfoChangeBRMAQ2 (name of the new service), and PIN7851 (BRM schema name) appropriately.
24. Register the ESB after the routing rule has been configured.

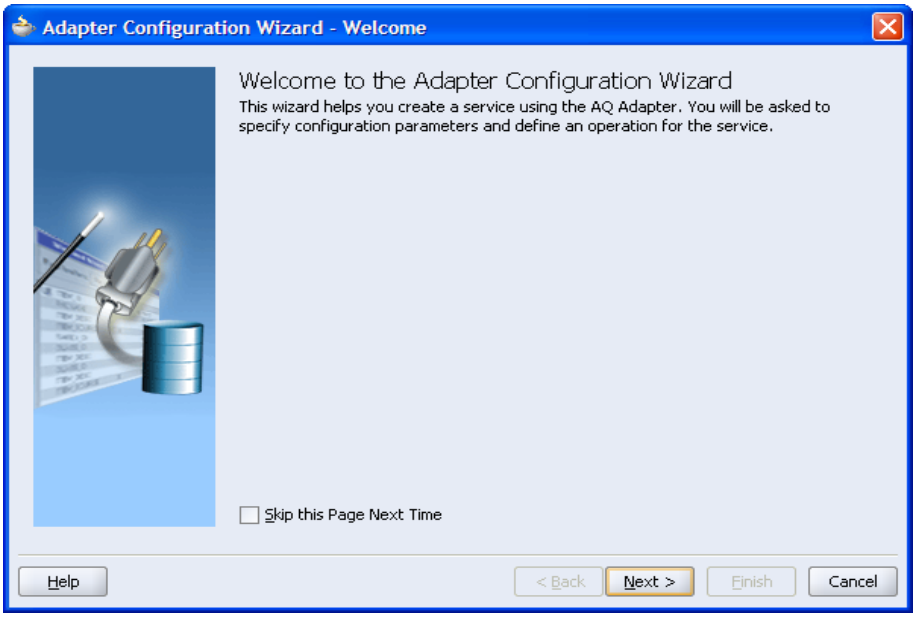
## Create an AQ Adapter for the Discount Synchronization

Create a new consumer for every new Oracle BRM instance. This consumer is used for the Discount sync.

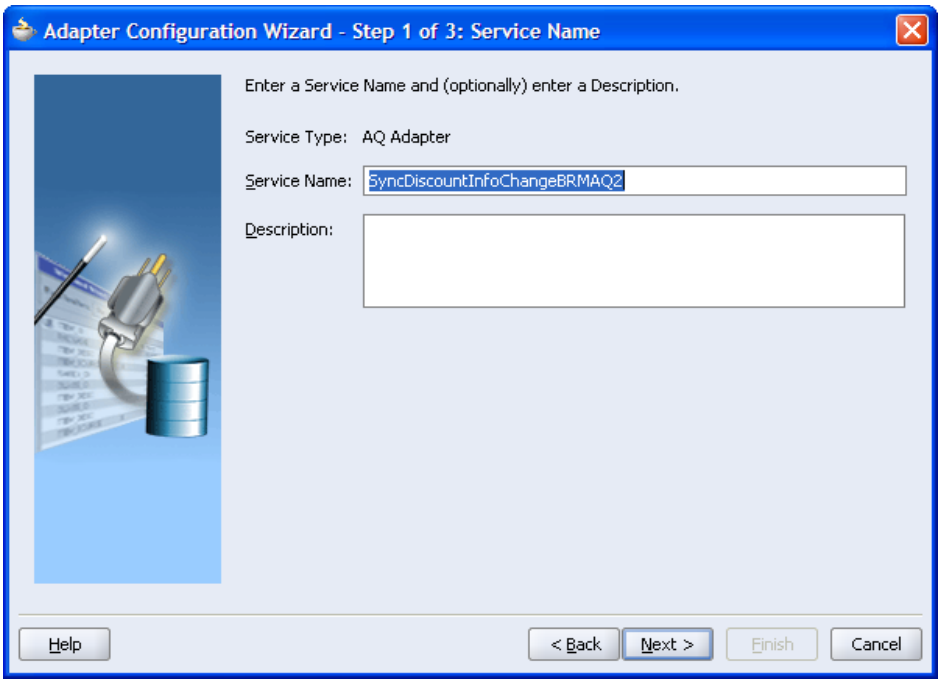
1. Create a new ESB project in JDeveloper by the name SyncDiscountInfoChangeBRMAQ2.
2. In the ESB file that is generated, create a new AQ Adapter. The System/Group should be AIASystem.BRM.ABCS. The name should be SyncDiscountInfoChangeBRMAQ2.
3. The Create AQ Adapter Service screen appears:



4. Click the Configure Adapter Service WSDL button, next to the WSDL File location.
5. The following screen appears:

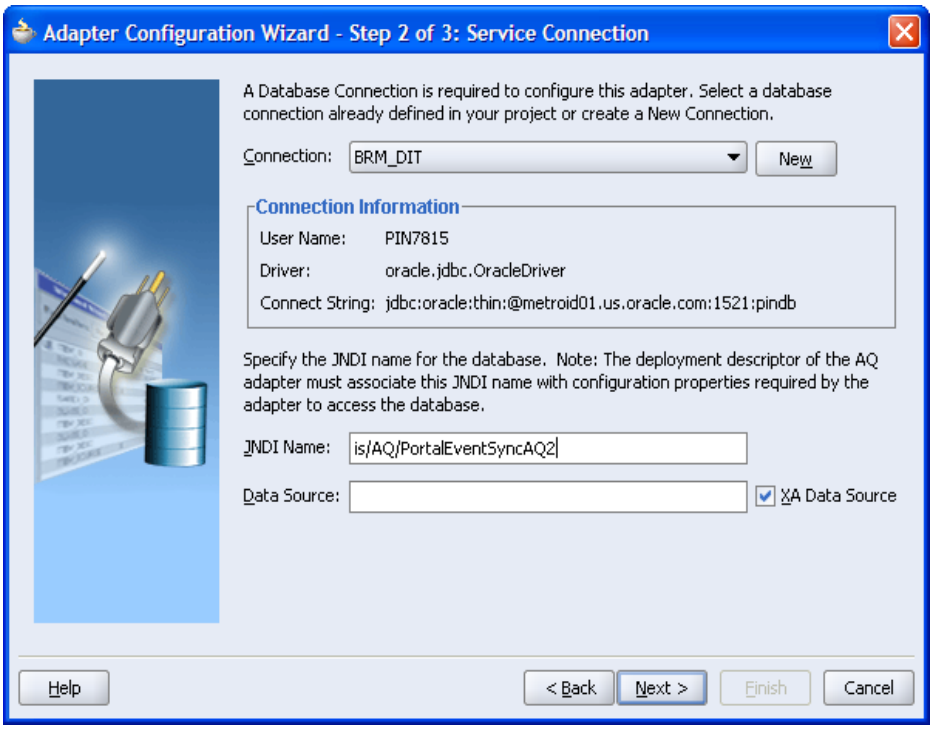



6. Click Next.
7. The following screen appears:



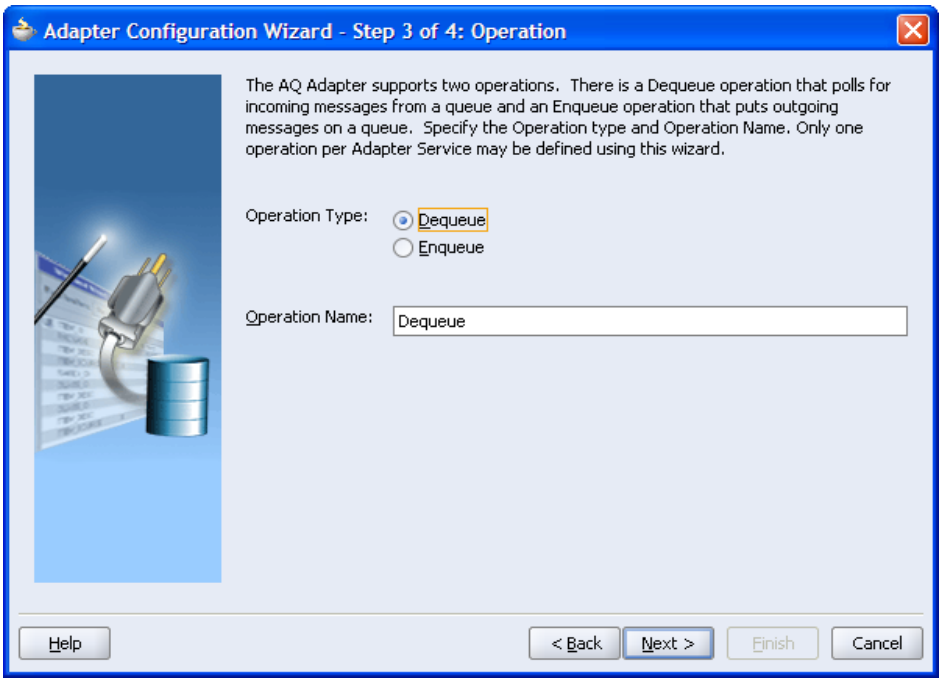

8. Select the Oracle BRM database connection and enter the JNDI name from **Creating a New AQ Adapter Connection Factory** section.

9. Enter the JNDI name of the additional Oracle BRM instance.



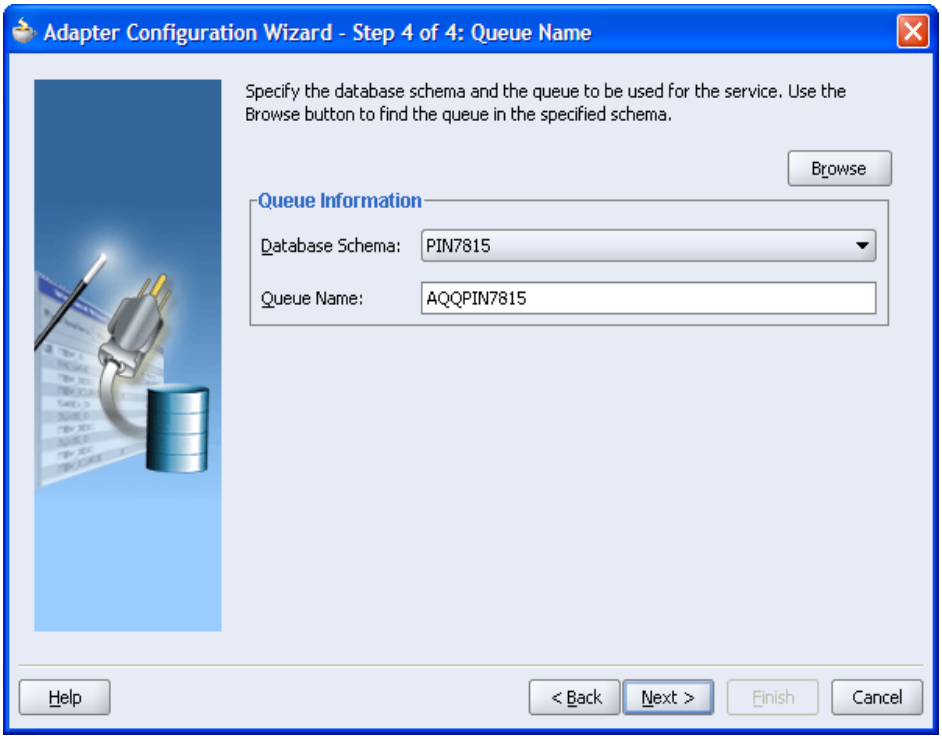

10. Click Next.

11. Select Dequeue as the operation type.




12. Click Next.

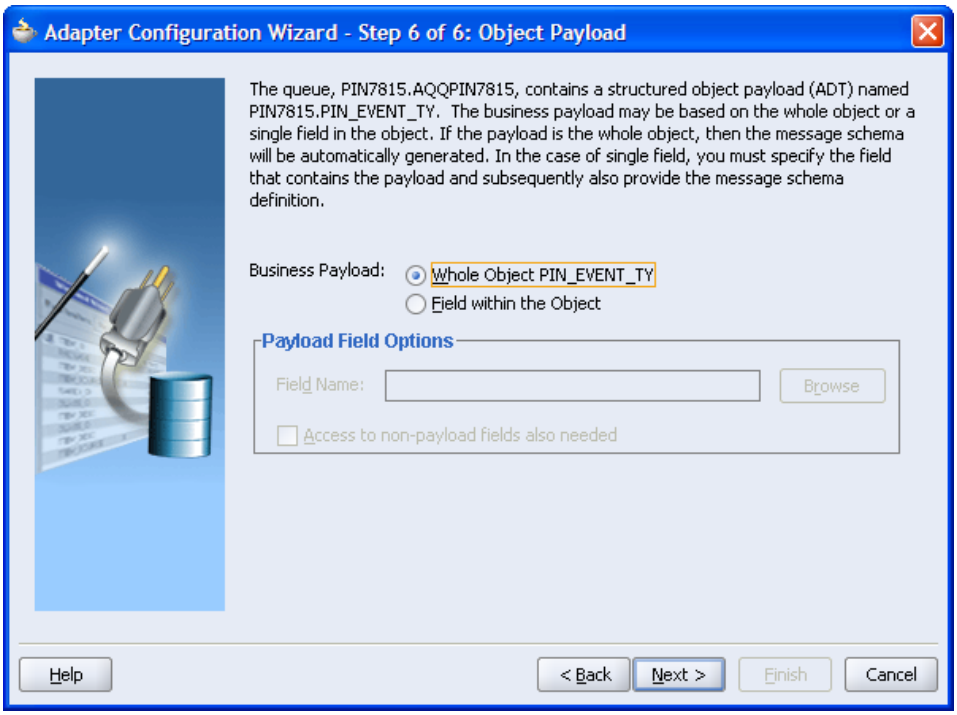
13. Select the database schema for the additional Oracle BRM instance and enter the queue name configured for this Oracle BRM instance.



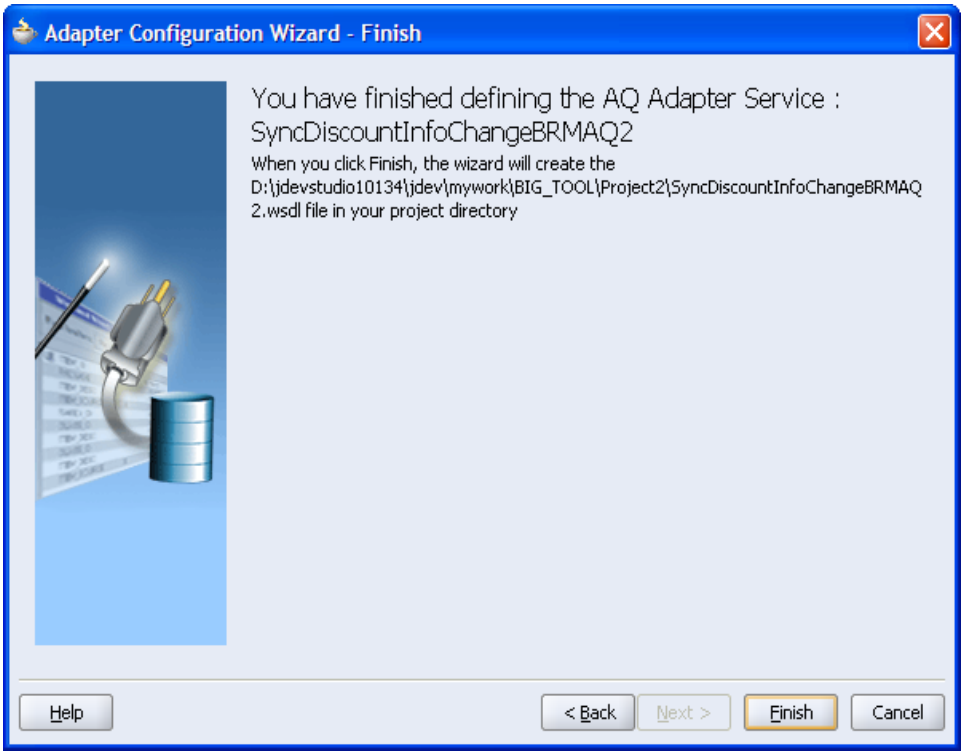

14. Click Next

15. Enter the correlation ID as DiscountInfoChange and click Next.

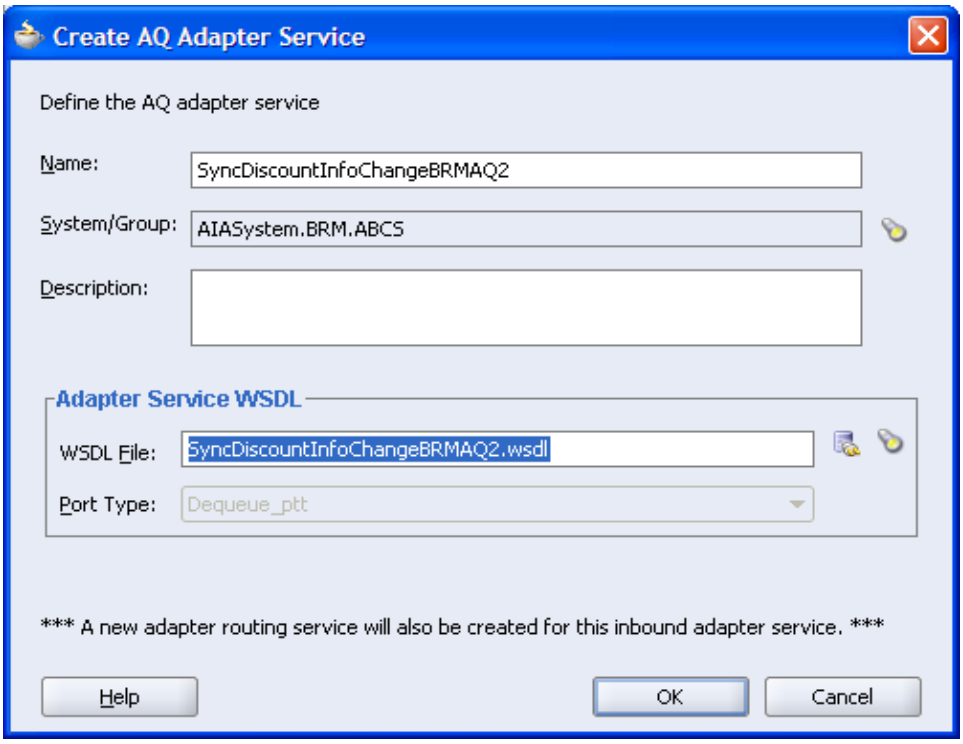
16. Select Whole Object PIN_EVENT_TY for the Business Payload option and click Next.




17. Click Finish.



18. The AQ Adapter service has been created; click OK.




19. Double-click the newly created routing service.
20. Expand routing rules and add a new routing rule.
21. In the Browse Target Service Operation pop-up window, expand BPELSystem/Default/ SyncDiscountBRMCommsReqABCSImpl and select the SyncDiscount operation.
22. For transformation, add a new transformation file with a unique name, such as PIN_EVENT_TY_To_DiscountInfoChange2.xsl.
23. Click the Source tab.
24. Add the following content in the source. Make sure that you change the highlighted values appropriately.

    (This content is derived from the delivered file <AIA_HOME>\PIPS\Industry\Communications\BRM\AQAdapterServices\SyncDiscountInfoChangeBRMAQ\PIN_EVENT_TY_To_DiscountInfoChange.xsl. Please use this file for the latest content.)

```
<?xml version="1.0" encoding="UTF-8" ?>
<?oracle-xsl-mapper <!-- SPECIFICATION OF MAP SOURCES AND TARGETS, DO NOT MODIFY. --
>
 <mapSources>
  <source type="WSDL">
   <schema location="SyncDiscountInfoChangeBRMAQ2.wsdl"/>
   <rootElement name="PIN_EVENT_TY" namespace="http://xmlns.oracle.com/xdb/PIN7851"/>
  </source>
```

```
  </mapSources>
  <mapTargets>
    <target type="WSDL">
      <schema
location="http://<host>:<port>/orabpel/default/SyncDiscountBRMCommsReqABCSImpl/1.0/SyncDiscountBRMCommsReqABCSImpl?wsdl"/>
      <rootElement name="DiscountInfoChange"
namespace="http://www.portal.com/schemas/CRMSync"/>
    </target>
  </mapTargets>
  <!-- GENERATED BY ORACLE XSL MAPPER 10.1.3.4.0(build 080718.0645) AT [WED JUL 22
12:43:45 IST 2009]. -->
?>
<xsl:stylesheet version="1.0"
                xmlns:svcdoc="http://xmlns.oracle.com/Services/Documentation/V1.0"
                xmlns:ns2="http://schemas.oracle.com/service/bpel/common"
                xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
                xmlns:plt="http://schemas.xmlsoap.org/ws/2003/05/partner-link/"
                xmlns:pc="http://xmlns.oracle.com/pcbpel/"

xmlns:ehdr="http://www.oracle.com/XSL/Transform/java/oracle.tip.esb.server.headers.ESBHeaderFunctions"
                xmlns:ns0="http://www.w3.org/2001/XMLSchema"
                xmlns:jca="http://xmlns.oracle.com/pcbpel/wsdl/jca/"

xmlns:ns1="http://xmlns.oracle.com/ABCSImpl/BRM/Industry/Comms/SyncDiscountBRMCommsReqABCSImpl/V1"
                xmlns:obj1="http://xmlns.oracle.com/xdb/PIN7851"
                xmlns:hwf="http://xmlns.oracle.com/bpel/workflow/xpath"
                xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functions.Xpath20"

xmlns:xref="http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXPathFunctions"
                xmlns:brmdiscountabo="http://www.portal.com/schemas/CRMSync"
                xmlns:tns="http://xmlns.oracle.com/pcbpel/adapter/aq/SyncDiscountInfoChangeBRMAQ2/"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                xmlns:ora="http://schemas.oracle.com/xpath/extension"
                xmlns:str="http://www.oracle.com/XSL/Transform/java/java.lang.String"
                xmlns:ids="http://xmlns.oracle.com/bpel/services/IdentityService/xpath"

xmlns:orcl="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functions.ExtFunc"
                xmlns:hdr="http://xmlns.oracle.com/pcbpel/adapter/aq/inbound/"
```

```
            exclude-result-prefixes="xsl plt pc ns0 jca obj1 str tns hdr svcdoc ns2 ns1 soap
brmdiscountabo bpws ehdr hwf xp20 xref ora ids orcl">
  <xsl:template match="/">
    <xsl:choose>
      <xsl:when test='/obj1:PIN_EVENT_TY/FLIST_BUF != ""'>
        <!-- Added to support "&" character. orcl:parseEscapedXML() fails If your input contains "&". -->
        <xsl:variable name="AmpersandReplacement">&amp;amp;amp;</xsl:variable>
        <xsl:variable name="EscapedRequestEBM_fixed"

select="str:replaceAll(string(/obj1:PIN_EVENT_TY/FLIST_BUF),'\u0026(amp;)*+',$AmpersandReplace
ment)"/>
        <xsl:variable name="EVENT_msg"
                      select="orcl:parseEscapedXML($EscapedRequestEBM_fixed)"/>
        <brmdiscountabo:DiscountInfoChange>
          <xsl:attribute name="InstanceId">
            <xsl:value-of select="$EVENT_msg/@InstanceId"/>
          </xsl:attribute>
          <brmdiscountabo:AccountObj>
            <xsl:value-of select="$EVENT_msg/brmdiscountabo:AccountObj"/>
          </brmdiscountabo:AccountObj>
          <xsl:for-each select="$EVENT_msg/brmdiscountabo:DiscountInfo">
            <brmdiscountabo:DiscountInfo>
              <brmdiscountabo:DiscountPoid>
                <xsl:value-of select="current()/brmdiscountabo:DiscountPoid"/>
              </brmdiscountabo:DiscountPoid>
              <brmdiscountabo:Name>
                <xsl:value-of select="current()/brmdiscountabo:Name"/>
              </brmdiscountabo:Name>
              <brmdiscountabo:DiscountStartTime>
                <xsl:value-of select="current()/brmdiscountabo:DiscountStartTime"/>
              </brmdiscountabo:DiscountStartTime>
              <brmdiscountabo:DiscountEndTime>
                <xsl:value-of select="current()/brmdiscountabo:DiscountEndTime"/>
              </brmdiscountabo:DiscountEndTime>
              <brmdiscountabo:PermittedTypes>
                <xsl:value-of select="current()/brmdiscountabo:PermittedTypes"/>
              </brmdiscountabo:PermittedTypes>
              <brmdiscountabo:Priority>
                <xsl:value-of select="current()/brmdiscountabo:Priority"/>
              </brmdiscountabo:Priority>
```

```
          <brmdiscountabo:Type>
            <xsl:value-of select="current()/brmdiscountabo:Type"/>
          </brmdiscountabo:Type>
          <brmdiscountabo:Description>
            <xsl:value-of select="current()/brmdiscountabo:Description"/>
          </brmdiscountabo:Description>
        </brmdiscountabo:DiscountInfo>
      </xsl:for-each>
    </brmdiscountabo:DiscountInfoChange>
  </xsl:when>
  <xsl:otherwise>
    <xsl:variable name="EVENT_msg"
                  select="orcl:parseEscapedXML(/obj1:PIN_EVENT_TY/LARGE_FLIST_BUF)"/>
    <brmdiscountabo:DiscountInfoChange>
      <xsl:attribute name="InstanceId">
        <xsl:value-of select="$EVENT_msg/@InstanceId"/>
      </xsl:attribute>
      <brmdiscountabo:AccountObj>
        <xsl:value-of select="$EVENT_msg/brmdiscountabo:AccountObj"/>
      </brmdiscountabo:AccountObj>
      <xsl:for-each select="$EVENT_msg/brmdiscountabo:DiscountInfo">
        <brmdiscountabo:DiscountInfo>
          <brmdiscountabo:DiscountPoid>
            <xsl:value-of select="current()/brmdiscountabo:DiscountPoid"/>
          </brmdiscountabo:DiscountPoid>
          <brmdiscountabo:Name>
            <xsl:value-of select="current()/brmdiscountabo:Name"/>
          </brmdiscountabo:Name>
          <brmdiscountabo:DiscountStartTime>
            <xsl:value-of select="current()/brmdiscountabo:DiscountStartTime"/>
          </brmdiscountabo:DiscountStartTime>
          <brmdiscountabo:DiscountEndTime>
            <xsl:value-of select="current()/brmdiscountabo:DiscountEndTime"/>
          </brmdiscountabo:DiscountEndTime>
          <brmdiscountabo:PermittedTypes>
            <xsl:value-of select="current()/brmdiscountabo:PermittedTypes"/>
          </brmdiscountabo:PermittedTypes>
          <brmdiscountabo:Priority>
            <xsl:value-of select="current()/brmdiscountabo:Priority"/>
```

```
            </brmdiscountabo:Priority>
            <brmdiscountabo:Type>
              <xsl:value-of select="current()/brmdiscountabo:Type"/>
            </brmdiscountabo:Type>
            <brmdiscountabo:Description>
              <xsl:value-of select="current()/brmdiscountabo:Description"/>
            </brmdiscountabo:Description>
          </brmdiscountabo:DiscountInfo>
        </xsl:for-each>
      </brmdiscountabo:DiscountInfoChange>
    </xsl:otherwise>
  </xsl:choose>
 </xsl:template>
</xsl:stylesheet>
```

25. Change the <host>, <port>, SyncDiscountInfoChangeBRMAQ2 (name of the new service), and PIN7851 (BRM schema name) appropriately.
26. Register the ESB after the routing rule has been configured.

# Adding a New Oracle BRM Instance: Add Routing Rules for Agent Assisted Billing Care PIP

The purpose of doing this change is to enable the EBS services in the Agent Assisted Billing Care process integration pack (AABC PIP) to route the billing profile-related query, create, and update sync requests to the new Oracle BRM instance. The requests are sent only to the Oracle BRM instance, where the billing profile exists. Per AABC restrictions, one billing profile cannot exist in more than one Oracle BRM system.

To ensure that this is done, for each relevant ESB operation, add additional routing rules, one for each additional Oracle BRM instance. These are the general steps:

1. Copy the EBS services with routing rules from the locations mentioned in step 3. The following general steps must be repeated for each EBS service.
2. Open the EBS in JDeveloper, using the project control (.jpr) file.
3. Copy the transformation file that populates the EBMHeader/Target/ID. The following names are the names of the transformation files for each EBS process. Name the new files appropriately by post fixing the system code (for example, PopulateTargetId_BRM_02.xsl). Change the new files to set EBMHeader/Target/ID appropriately (such as BRM_02). This has to be done for all the EBSs that are used across the AABC PIP. These are:

    - EBS: CommunicationsAccountBalanceAdjustmentEBSV2

      **File Location:**

      $AIA_HOME/PIPS/Industry/Communications/Setup/AgentAssistedBillingCare/EBS/Billing

**File Name:** PopulateTargetId_BRM_01.xsl

- EBS: CommunicationsInstalledProductEBSV2

  **File Location:**

  **If Order To Bill is installed, then**
  $AIA_HOME/PIPS/Industry/Communications/Setup/OrderToBill/EBS/Billing

  **If Order To Bill is *not* installed, then**
  $AIA_HOME/PIPS/Industry/Communications/Setup/AgentAssistedBillingCare/EBS/Billing

  **File Name:** QueryInstalledProductListEBM_PopulateTargetID.xsl

- **EBS:** CommunicationsInvoiceEBSV2

  **File Location:**
  $AIA_HOME/PIPS/Industry/Communications/Setup/AgentAssistedBillingCare/EBS/Billing

  **File Name:** Populate_Target_Info_BRM_01.xsl

- **EBS:** CommunicationsReceivedPaymentEBSV1

  **File Location:**
  $AIA_HOME/PIPS/Industry/Communications/Setup/AgentAssistedBillingCare/EBS/Billing

  **File Name:** Populate_Target_Info_BRM_01.xsl

- **EBS:** CommunicationsServiceUsageEBSV2

  **File Location:**
  $AIA_HOME/PIPS/Industry/Communications/Setup/AgentAssistedBillingCare/EBS/Billing

  **File Name:** Populate_Target_Info_BRM_01.xsl

- **EBS:** CommunicationsCreditAlertEBSV1

  **File Location:**
  $AIA_HOME/PIPS/Industry/Communications/Setup/AgentAssistedBillingCare/EBS/Billing

  **File Name:** SetTargetID.xsl

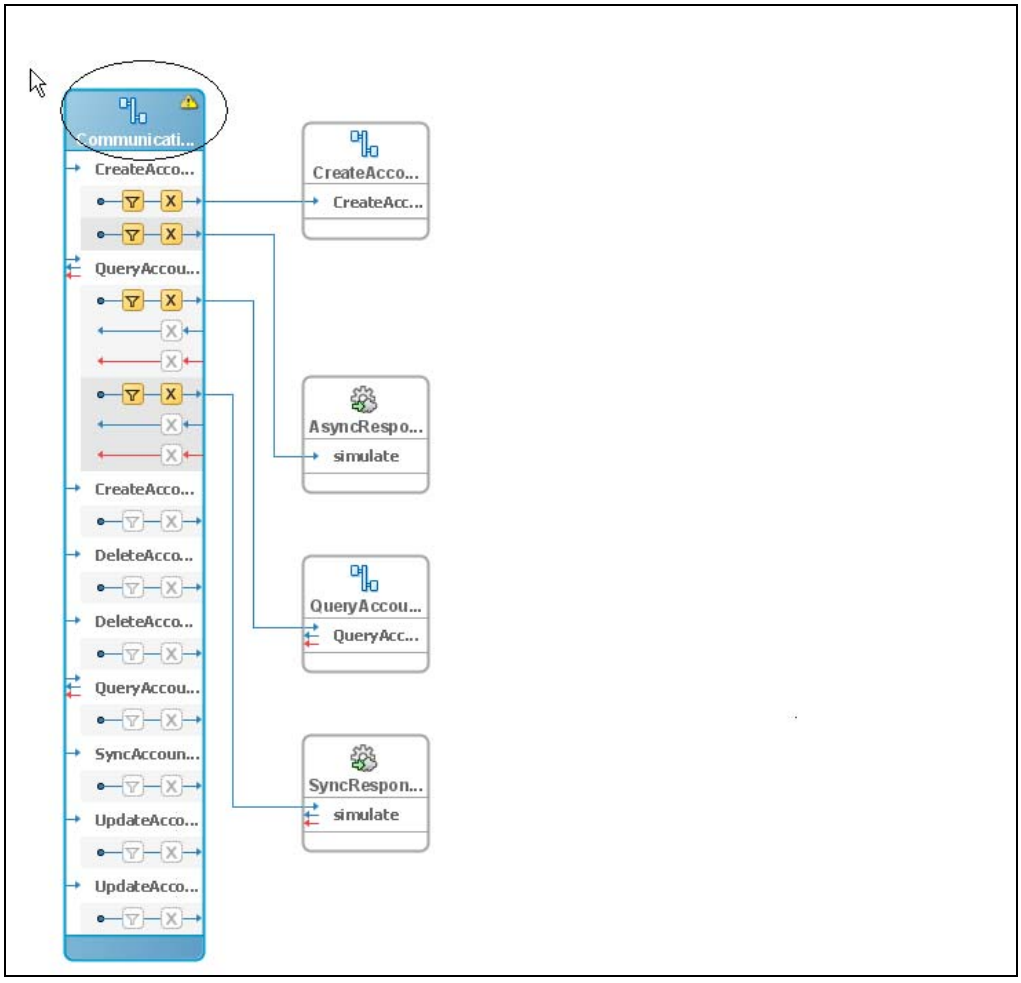
- **EBS:** CommunicationsCustomerPartyEBSV2 (only for QueryCustomerPartyList operation going to BRM)

**File Location:**

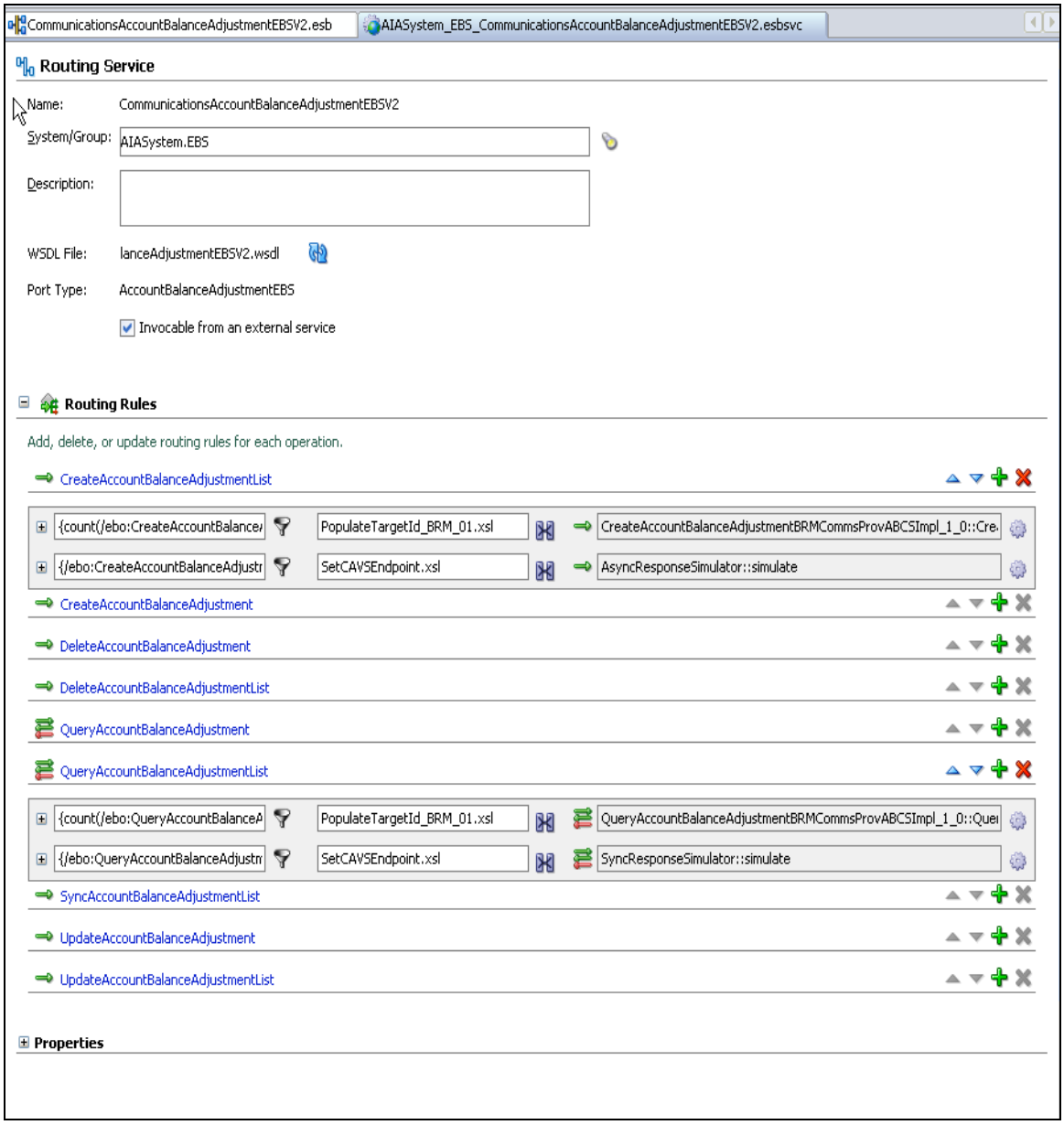
$AIA_HOME/PIPS/Industry/Communications/Setup/AgentAssistedBillingCare/EBS/Customer

**File Name:** AddTargetID_BRM01.xsl (for QueryCustomerPartyList going to BRM).

4. Click the portion circled here to open the routing service:

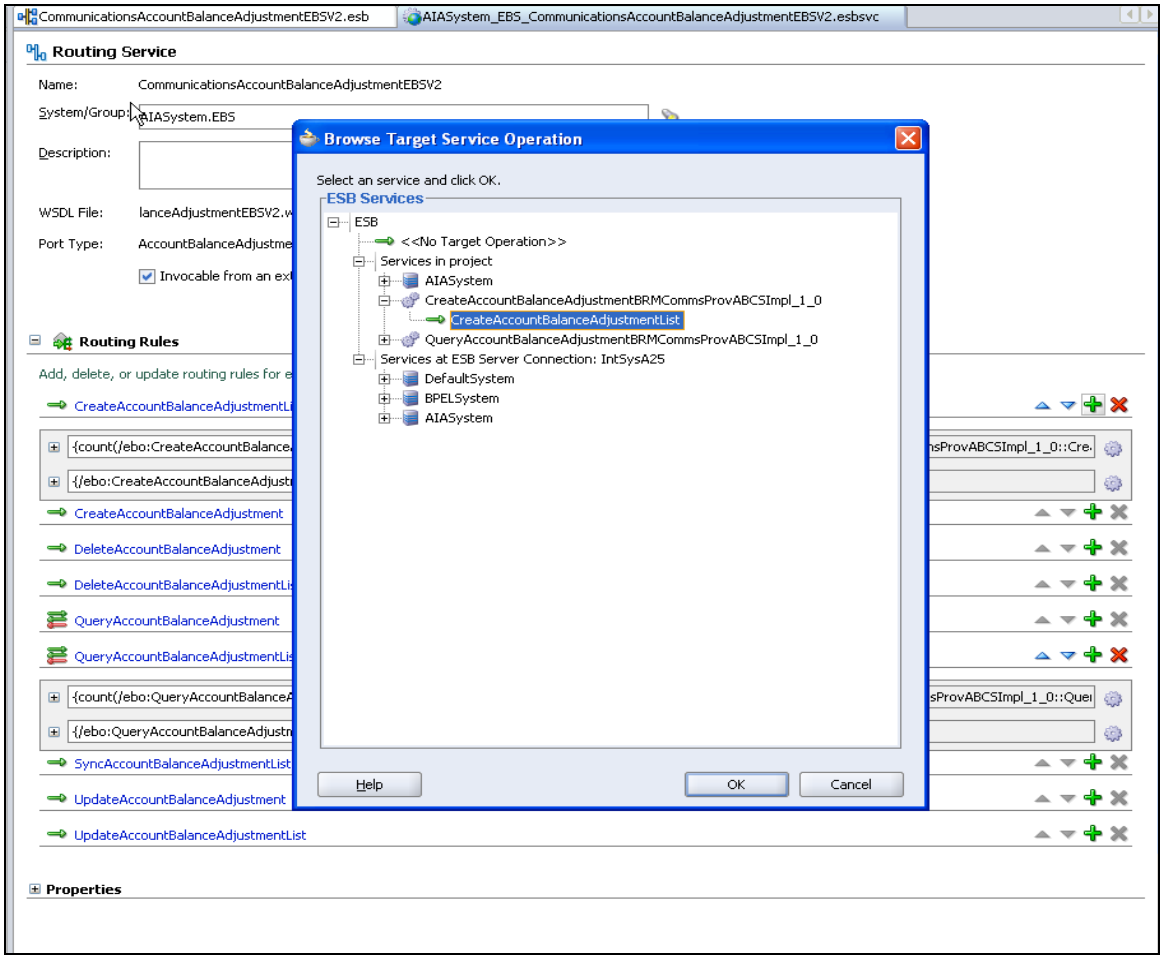


The routing service looks like this:

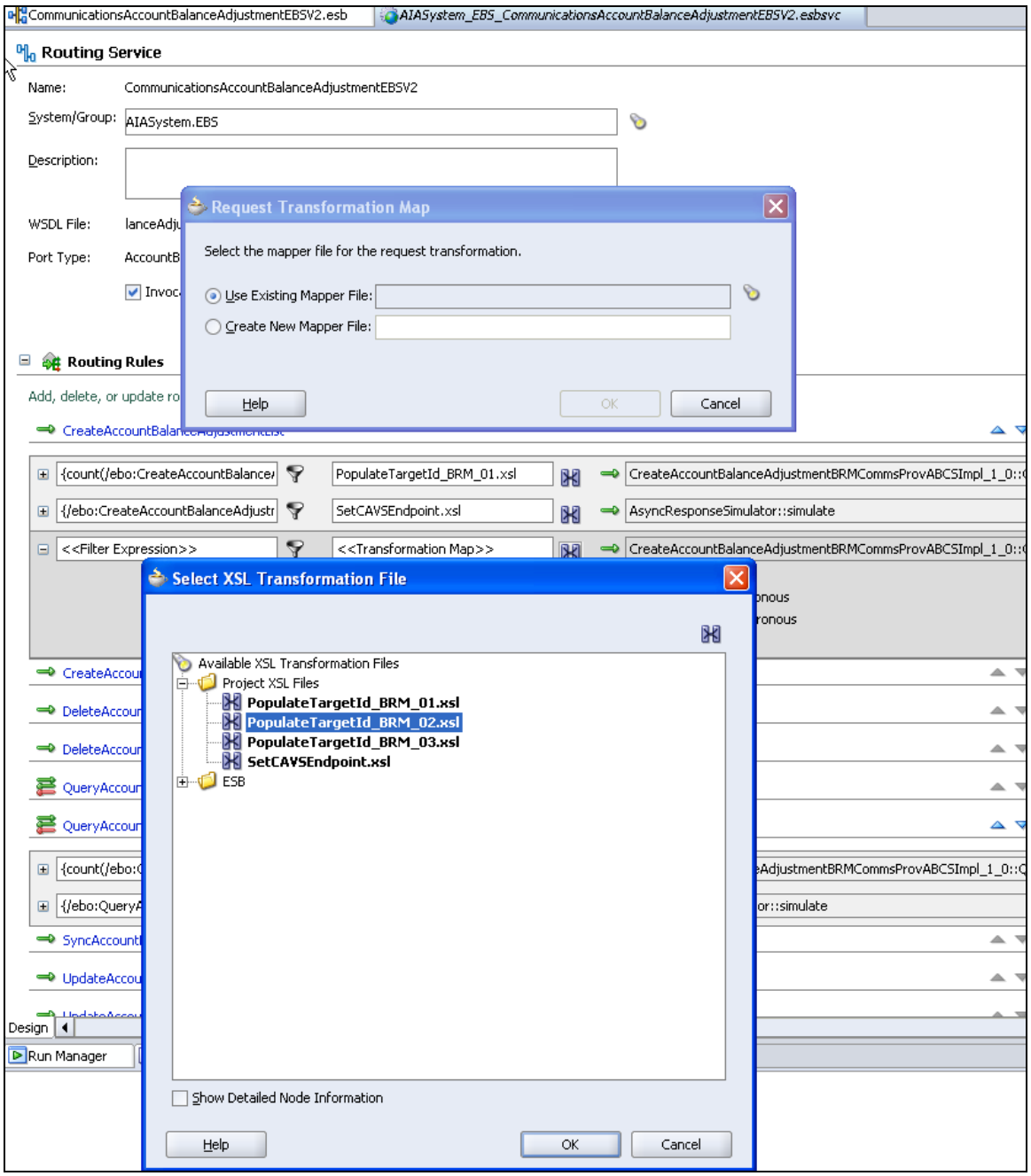



5. For each operation, click + against the operation to create a routing rule.

6. The Target service is the same as what the BRM_01 routing rule is using. This can be selected from the services available under the same project.



7. After the target is selected, the transformation file must be selected: Click the button indicating the transformations. A pop-up window Request Transformation Map appears. Select Use Existing Mapper file, and select the mapping file for BRM_02.




8. In the Filter expression box, copy the filter expression used for the BRM_01 routing rule and replace BRM_01 with BRM_02 and add the filter expression.

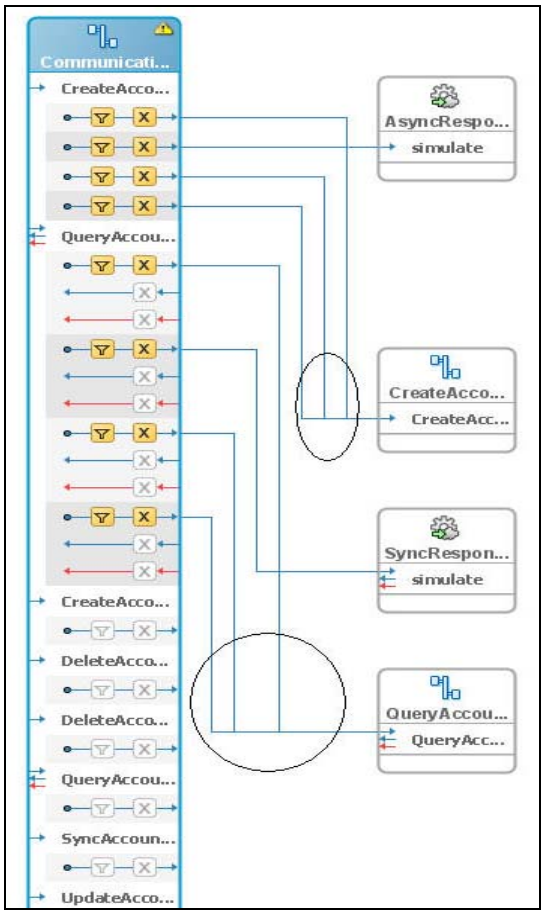
For example, this will be replaced with what follows it:

```
{count(/ebo:CreateAccountBalanceAdjustmentListEBM/corecom:EBMHeader/corecom:MessageProcessingInstruction/corecom:EnvironmentCode[text() = 'CAVS']) = 0 and xref:lookupXRef('CUSTOMERPARTY_BILLPROFILEID','COMMON',/ebo:CreateAccountBalanceAdjust
```

```
mentListEBM/ebo:DataArea/ebo:CreateAccountBalanceAdjustmentList/corecom:BillToPartyReference/
corecom:BillingProfileReference/corecom:BillingProfileIdentification/corecom:ID,'BRM_01',false())!='};{
namespace ebo=http://xmlns.oracle.com/EnterpriseObjects/Core/EBO/AccountBalanceAdjustment/V2
namespace corecom=http://xmlns.oracle.com/EnterpriseObjects/Core/Common/V2 namespace
xref=http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXPathFunctions }
```

```
{count(/ebo:CreateAccountBalanceAdjustmentListEBM/corecom:EBMHeader/corecom:MessageProces
singInstruction/corecom:EnvironmentCode[text() = 'CAVS']) = 0 and
xref:lookupXRef('CUSTOMERPARTY_BILLPROFILEID','COMMON',/ebo:CreateAccountBalanceAdjust
mentListEBM/ebo:DataArea/ebo:CreateAccountBalanceAdjustmentList/corecom:BillToPartyReference/
corecom:BillingProfileReference/corecom:BillingProfileIdentification/corecom:ID,'BRM_02',false())!='};{
namespace ebo=http://xmlns.oracle.com/EnterpriseObjects/Core/EBO/AccountBalanceAdjustment/V2
namespace corecom=http://xmlns.oracle.com/EnterpriseObjects/Core/Common/V2 namespace
xref=http://www.oracle.com/XSL/Transform/java/oracle.tip.xref.xpath.XRefXPathFunctions }
```

After the routing rules are created, the ESB should look like this:




9. Save the files.
10. Make sure that the .esbsvc file of that ESB has the targetOperation qname attribute referring to the BPEL with the correct service qname.
11. Deploy the ESB using ANT.

12. Alternatively, if you want to register the ESB using Jdev, you may want to select the routing services from the Integration server (services at ESB Server Connection) instead of picking them from the local project WSDL (Services in Project).

# Appendix F: Reconfiguring AIA for Comms

This appendix provides information about how to change the Oracle Communications Billing and Revenue Management (Oracle BRM) instance post-installation.

Many functional occasions occur when the Oracle BRM instance that Oracle Application Integration Architecture (Oracle AIA) points to needs to be changed post-installation. These include:

- Moving to a new Oracle BRM server due to replacement of hardware
- Switching from Test to Production instance

**Caution:** Before switching from one Oracle BRM instance to another, you must make sure that the new instance is a replica of the old instance. That is, all the data (such as accounts, services, products, discounts, and so on) in the old instance must also exist in the new instance, and they must also have matching IDs (POIDs). If this is not the case, failures will occur in Oracle AIA. If any difference exists, then cross-reference (XREF) tables must be updated with the correct IDs before any of the flows are run.

## Changing the Oracle BRM Instance

Oracle AIA and Oracle BRM communication happens through two adapters: inbound to Oracle AIA through Oracle Advanced Queuing (AQ) Adapter and inbound to Oracle BRM through Oracle BRM JCA Adapter. If a change occurs in the Oracle BRM instance, then the connection factories for both of these adapters must be changed.

To change the Oracle BRM instance:

1. Update connection parameters for the eis/BRM as well as any custom-created Oracle BRM connection factories for BRMJCAAdapter.
2. Update connection parameters for eis/AQ/PortalEventSyncAQ as well as any custom-created Oracle BRM connection factories for AQAdapter.
3. If the BRM Event AQ queue name or the BRM schema name for the AQ Queue (or both) are changed, then replace occurrences of the old Event AQ queue name or the Oracle BRM schema name (or both) with the new names from $AIA_HOME/PIPS/Industry/Communications/BRM/AQAdapterServices/SyncProductInfoChangeBRMAQ and SyncDiscountInfoChangeBRMAQ:
4. Replace occurrences of the old Event AQ queue name or the BRM schema name (or both) with the new names from $AIA_HOME/PIPS/Industry/Communications/BRM/AQAdapterServices/SyncProductInfoChangeBRMAQ and SyncDiscountInfoChangeBRMAQ.

Redeploy the services.

# Appendix G: Using the Resequencer Feature of ESB

The Oracle Enterprise Service Bus Resequencer feature is used by various integration flows to ensure that messages are processed in a particular sequence.

**For more information** about the resequencer, see the *Oracle AIA Foundation Pack - Integration Developer's Guide*, “Designing and Constructing ABC Services,” Interacting with Participating Applications, Oracle Enterprise Service Bus Resequencer.

This table lists the queues and flows that are enabled for sequencing.

**Note:** Revision Order support - Oracle OSM manages scenarios where multiple revisions for the same order are sent out of sequence. If you are using a different Order Management system it must have similar support.

| Oracle AIA Queue | Flow | JMS Priority Set By | Sequencing Criteria | Comments |
|---|---|---|---|---|
| AIA_SALESORDERJMSQ | Order submission flow from Siebel CRM to Oracle AIA. | Siebel | Group By: Billing Account on Order Header (/ListOfSWIOrderIO/SWIOrder/BillingAccountId)<br>Order of Processing: FIFO (First in First Out).<br>ESB Consumer: ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer. | Available in both the Order to Bill and the Order to Activate PIPs.<br>The resequencer in this flow ensures that in scenarios where concurrent orders for the same customer are submitted, the AIA Siebel provider does not fail while creating cross-reference entries. |
| AIA_CRTCUST_OUT_JMSQ | Order flow from Order Management/OSM to Oracle AIA for customer data creation in billing. | Order Management/OSM | Group By: Account ID on the message. (This is either the Billing account or the Service account on the order line that must be created in billing).(/SyncCustomerPartyListEBM/DataArea/SyncCustomerPartyList/CustomerPartyAccount/Identification/ApplicationObjectKey/ID[@schemeID='AccountId'])<br>Order of Processing: FIFO (First in First Out).<br>ESB Consumer: CommunicationsCustomerPartyEBSV | Available only in the Order to Bill PIP.<br>The resequencer in this flow ensures that the solution can successfully handle processing of concurrent orders for the same customer. |

| Oracle AIA Queue | Flow | JMS Priority Set By | Sequencing Criteria | Comments |
|---|---|---|---|---|
| | | | 2Resequencer. | |
| AIA_UPDSO_OUT_JMSQ | Update order flow from Order Management/OSM to Oracle AIA for Siebel CRM system. | Not Set | Group By: Account ID mentioned in the ObjectCrossReference section of the update message( /UpdateSalesOrderEBM/EBMHeader/Sender/ObjectCrossReference/SenderObjectIdentification/AlternateObjectKey/ID[@schemeID = 'CUSTOMERPARTY_ACCOUNTID' and @schemeAgencyID = 'COMMON']) Order of Processing: FIFO (First in First Out). ESB Consumer: UpdateSalesOrderOSMCFSCommsJMSConsumer. | Available in both the Order to Bill and Order to Activate PIPs. Note: The consumer in the Order to Bill PIP is only a sample. The resequencer in this flow ensures that multiple updates for the same order are processed in the right sequence. |
| -- | Sync customer flow from Siebel CRM system to Oracle Customer Hub. | Not Set | Group By: AccountID. Order of Processing: FIFO (First in First Out). ESB Consumer: SyncAcctSiebelAggrEventConsumer SyncContSiebelAggrEventConsumer. | Available in the Order to Bill and Agent Assisted Billing Care PIPs. The resequencer in this flow ensures that multiple updates for the same customer are processed in the right sequence. |

# Resolving Errors in Flows with Resequencer

An error may occur in the order process after the order was consumed by ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer but failed in any of the subsequent processes. As a result, the messages will be rolled back to the resequencer. If this occurs, the fallout specialist must take corrective action on this resequencer to move the flow. If the message fails due to a system error (for example, if the target system is unavailable), then fallout specialists must retry the message from resequencer. If the message fails because of a business error, then the fallout specialist must unblock the resequencer.

If an error occurs in the Oracle BRM Customer provider, the message may be blocked in the CommunicationsCustomerPartyEBSV2Resequencer service and the error message may not propagate back to CommsProcessFulfillmentOrderBillingAccountListEBF. In these situations, fallout specialists must take corrective action on the resequencer to move the flow. If the message fails due to a system error (for example, if the target system is unavailable), then fallout specialists must retry the message from resequencer. If the message fails because of a business error, then the fallout specialist must unblock the resequencer.

An error may occur in the Siebel provider after it is consumed by UpdateSalesOrderOSMCFSCommsJMSConsumer and sent for processing. In this situation the messages will be rolled back to the resequencer for this consumer and any subsequent order updates for that particular order will not be processed. If this occurs, the fallout specialist must take corrective action on this resequencer to move the flow like the ones described above. If the message fails due to a system error (for example, if the target system is unavailable), then fallout specialists must retry the message from resequencer. If the message fails because of a business error, then the fallout specialist must unblock the resequencer.

**For more information** about retrying messages from resequencer and unblocking the resequencer, see *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite and Oracle Business Process Management Suite*, "Resequencing Messages", Monitoring Resequenced Messages.

# Appendix H: Using Session Pool Manager

Session Pool Manager (SPM) is a service in the Oracle SOA Suite web server whose primary function is to manage a pool of web server session tokens that can be reused by BPEL flows.

Note: At this time, SPM is only available for the Siebel web server and the CRM On Demand webserver.

Reusing session tokens significantly enhances the overall performance of BPEL flows that call web services. This is because the session token creation process is a time-consuming operation in the application web server.

Some features of SPM include:

- SPM is automatically initialized upon the request of a session token.
- The session token pool is stored in memory.
- SPM supports concurrent (multithreaded) BPEL flows
- SPM supports multiple application web server instances (hosts), with one SPM instantiated per host.
- Includes the Sentinel, which periodically checks session tokens, removes expired sessions, and replaces them with new ones.

**For more information** about Session Pool Manager, see the *Oracle Application Integration Architecture 2.5 - Session Pool Manager: User Guide.*

## Configuring Session Pool Manager

SPM configuration properties are set in the AIAConfigurationProperties.xml file as a Module Configuration. The module name is SessionPoolManager.

SPM can work with multiple hosts (application web servers). Therefore, each property can be set as a default for all hosts, and overridden for a specific host. The only exception is the TRACE.LOG.ENABLED property, which cannot be set to be server-specific.

Each property has a prefix that indicates the application web server. For example:

```
<Property name="all_hosts.PoolSize_Max">40</Property>
<Property name="SEBL_01.PoolSize_Max">20</Property>
```

The first line defines ***40*** as the default maximum pool size for all hosts.

The second line overrides the default pool size to ***20*** for the application web server SEBL_01.

The concept of system ID and HostId are synonymous.

For example, a customer installing a Process Integration Pack (PIP) for Siebel may use SEBL_01 as the system ID for the Siebel application web server. They will see SEBL_01 in the AIAConfigurationProperties.xml file as the “Default.SystemID" property for the services connecting to the Siebel application web server. This SEBL_01 value should also be used as the HostId value in SPM to refer to the Siebel application web server.

All properties must be defined by application web server or default. If a property is not defined for a specific application web server, then the default property (all_hosts) will be used. If no all_hosts default property is defined, the caller will receive a fault indicating the missing property.

Based on the patch set level that you are on, Siebel connectors may have been enabled to use Session Pool Manager:

To see if you are at the correct patch set level, go to $AIA_HOME/config/AIAConfigurationProperties.xml and check the respective process sections. The values of "SEBL_01.EndpointURI" properties for the process, should match with the value of "SEBL_01.EndpointURI" property under the SessionPoolManager module configuration section. For example:

```
   <Property
name="Routing.<*>.SEBL_01.EndpointURI">http://[siebel.http.host]:[si
ebel.http.port]/eai_enu/start.swe?SWEExtSource=SecureWebService&amp;
SWEExtCmd=Execute&amp;WSSOAP=1</Property>
```

**For more information** about the SPM configuration properties, see the *Oracle Application Integration Architecture 2.5 - Session Pool Manager: User Guide*. “Setting Up Session Pool manager Configuration Properties”.

**Caution:** In the case where a Siebel instance (such as for System Code "SEBL_01") is restarted, you must terminate the Session Pool Manager for that Siebel instance because its Siebel sessions are no longer valid.

# Appendix I: Mapping Billing Dates

The following table provides information about how dates are set in Oracle BRM. These terms and abbreviations are used in the table:

- ODT: Order Datetime.

  This is the date that the order was placed by the customer and is captured on the order in the order capture system (Siebel). Siebel defaults this, but it can be changed by the user.

- RDDT: Requested Delivery Datetime.

  This is the delivery date requested by the customer; it is captured on the order in the order capture system (Siebel). It is also known as *Due Date*.

- ADDT: Actual Delivery Datetime.

  This is the actual delivery date time; it is supplied by the order management system that fulfills the order, and is updated in the order capture system (Siebel).

- Purchase Start Date: The date as of which Oracle BRM applies purchase fees.
- Cycle Start Date: The date as of which Oracle BRM applies cycle fees.
- Usage Start Date: The date as of which Oracle BRM rates usage and applies usage fees.

| Operation Being Performed in BRM | Dates Set by AIA When the Service is Called | BRM Opcodes Invoked | Expectations of the Order Management System |
|---|---|---|---|
| For the CommunicationsBillingEBSV1.ProcessFulfillmentOrderBillingAccountList service | | | |
| Customer data creation | AIA uses order date as the effective date for customer data creation | PCM_OP_CUST_COMMIT_CUSTOMER | Pass Order Date coming from Siebel CRM. |
| For the CommunicationsBillingEBSV1.ProcessFulfillmentOrderBilling service | | | |
| Single Phase Billing - Billing Fulfillment<br>Promotion Purchase | AIA passes the Purchase Date as the Valid From date for bundle purchase (that represents purchased promotion).<br>If Purchase Date is null, then it passes Requested Delivery Date and if that is null, it passes no date and Oracle BRM defaults current date. | PCM_OP_SUBSCRIPTION_SET_BUNDLE | Pass Order Date and Requested Delivery Date coming from Siebel CRM.<br>Set Purchase Date to Actual Delivery Datetime. |
| Single Phase Billing - Billing Fulfillment | If all three of the billing dates are set, then Oracle AIA uses | PCM_OP_CUST_MODIFY_CUSTOMER | Pass Order Date and Requested Delivery Date |

| Operation Being Performed in BRM | Dates Set by AIA When the Service is Called | BRM Opcodes Invoked | Expectations of the Order Management System |
|---|---|---|---|
| Account level product (Item or Subscription)/Discount Purchase.<br>Service Purchase, this includes service-level product (Item or Subscription)/Discount Purchase. | Order Date as Effective Date, and sets respective offset (Order Date - respective billing date).<br>Billing dates are: Purchase Date, Cycle Start Date and Usage Start Date.<br><br>If any of the three billing dates are not set, then Oracle AIA passes no dates to Oracle BRM and lets Oracle BRM default the Purchase, Cycle Start and Usage Start dates.<br><br>Note that for purchase of a service bundle, this check for existence of billing dates applies to ALL products and discounts included in the service bundle. | PCM_OP_SUBSCRIPTION_PURCHASE_DEAL | coming from Siebel CRM.<br>Set Purchase Date, Start Cycle, and Start Usage to Actual Delivery Datetime to explicitly control setting of billing dates. |
| Single Phase Billing - Billing Fulfillment.<br>Time Based Account or Service level Subscription Product/Discount Purchase | In addition to setting of billing dates as described previously, if Service End Date is passed, then Oracle AIA additionally sets the Purchase, Cycle and Usage end date offsets (difference between the respective billing date and service end date).<br><br>If any of the billing dates (Purchase, Cycle, or Usage start) are not set then Oracle AIA uses the Order Date to calculate the Purchase, Cycle and Usage end date offsets (difference between the Order Date and Service End Date). | PCM_OP_MODIFY_CUSTOMER<br>PCM_OP_SUBSCRIPTION_PURCHASE_DEAL | Populate Purchase, Cycle and Usage Start dates (this is required for enabling time-based offerings (TBO).<br>Calculate the Service End Date based on TBO attributes as documented in TBO section. |
| Single Phase Billing - Billing Fulfillment.<br>Time Based Account or service-level Subscription | If Service End Date is passed (and prior value is set), then Oracle AIA uses that to reset the Purchase, | PCM_OP_SUBSCRIPTION_SET_PRODINFO<br>PCM_OP_SUBSCRIPTIO | Calculate the Service End Date based on TBO attributes as documented in TBO section. Populate prior |

| Operation Being Performed in BRM | Dates Set by AIA When the Service is Called | BRM Opcodes Invoked | Expectations of the Order Management System |
|---|---|---|---|
| Product/Discount Update (of end date due to promotion upgrade or downgrade, or other pricing changes). | Cycle and Usage end dates. | N_SET_DISCOUNTINFO | value to trigger update. |
| Single Phase Billing - Billing Fulfillment.<br>Promotion Cancellation | If ADDT is passed, Oracle AIA uses that to set the VALID_TO date in Oracle BRM for the bundle. If ADDT is not passed then Oracle AIA uses the Requested Delivery Datetime. If Requested Delivery Datetime is not passed then Oracle AIA does not set the VALID_TO date. | PCM_OP_SUBSCRIPTION_SET_BUNDLE | Pass Order Date and Requested Delivery Date coming from Siebel CRM.<br>Set Actual Delivery Datetime |
| Single Phase Billing - Billing Fulfillment<br>Application of Promotion Penalties or MACD One Time Charge (Suspend, Resume, Disconnect, or Move charge)<br>Note - These are processed *only* in Billing Fulfillment. | If ADDT is passed, Oracle AIA sets the effective date to ADDT. If ADDT is not passed then Oracle AIA lets Oracle BRM default the purchase date (to current date). | PCM_OP_SUBSCRIPTION_PURCHASE_DEAL | Set Actual Delivery Datetime |
| Single Phase Billing - Billing Fulfillment.<br>Suspend, Resume, or Cancellation of Service or account-level or service-level Subscription Product/Discount. | If ADDT is passed, then AIA uses that as the effective date for the operation, else it lets BRM default the date (to current date) | PCM_OP_SUBSCRIPTION_SET_PRODUCT_STATUS<br>PCM_OP_SUBSCRIPTION_SET_DISCOUNT_STATUS<br>PCM_OP_CUST_SET_STATUS | Set Actual Delivery DateTime. |
| Two-Phase Billing - Billing Initiation.<br>Promotion Purchase. | Oracle AIA passes Purchase Date as the Valid From date.<br>If Purchase Date is null, then Oracle AIA passes Requested Delivery Date and if that is null, iOracle AIA passes no date and Oracle BRM defaults current date | PCM_OP_SUBSCRIPTION_SET_BUNDLE | Pass Order Date and Requested Delivery Date coming in from Siebel CRM.<br>Set Purchase Date to Expected Delivery Date. |
| Two Phase Billing - Billing Initiation.<br>Account-level or service-level Item Type Product Purchase. | Oracle AIA validates that Purchase Date is set to future (based on value of configuration property - | PCM_OP_CUST_MODIFY_CUSTOMER<br>PCM_OP_SUBSCRIPTION_PURCHASE_DEAL | Pass Order Date coming in from Siebel CRM.<br>Set Purchase, Cycle, and Usage Date to Future (one |

| Operation Being Performed in BRM | Dates Set by AIA When the Service is Called | BRM Opcodes Invoked | Expectations of the Order Management System |
|---|---|---|---|
| | FutureTimeThreshold).<br>Uses Order Date as Effective Date, and sets respective offset for each billing date (calculated as Order Date - respective billing date).<br>Billing Dates are - Purchase Date, Cycle Start Date and Usage Start Date. | | year out to match default threshold). |
| Two Phase Billing - Billing Initiation.<br>Account-level or service-level Subscription Type Product/Discount Purchase. | Oracle AIA validates that Start Cycle Date is set to future (based on value of configuration property - FutureTimeThreshold).<br>Uses Order Date as Effective Date, and sets respective offset for each billing date (calculated as Order Date - respective billing date).<br>Billing Dates are - Purchase Date, Cycle Start Date, and Usage Start Date | PCM_OP_CUST_MODIFY_CUSTOMER<br>PCM_OP_SUBSCRIPTION_PURCHASE_DEAL | Pass Order Date coming in from Siebel CRM.<br>To support validation mode, set all three billing dates to the Future (one year out to match default threshold).<br>To support latency mode, set Purchase and Start Usage Date to Current, but set Cycle Start Date to Future (one year out to match threshold). |
| Two-Phase Billing - Billing Fulfillment.<br>Promotion Purchase. | Oracle AIA uses Purchase date to reset Valid From date. | PCM_OP_SUBSCRIPTION_SET_BUNDLE | If purchase date had been set to Expected Delivery Date in Billing Initiation, reset purchase date to Actual Delivery Date |
| Two Phase Billing - Billing Fulfillment.<br>Account-level or service-level Item Type Product Purchase. | If prior values are set, Oracle AIA resets respective billing date by passing in absolute values for each billing date that needs to be reset.<br>Billing Dates are - Purchase Date, Cycle Start Date, and Usage Start Date. | PCM_OP_SUBSCRIPTION_SET_PRODINFO | Reset all three billing dates to Actual Delivery Datetime (set prior values to trigger update). |
| Two Phase Billing - Billing Fulfillment.<br>Account-level or service-level Subscription Type Product/Discount Purchase. | If prior values are set, Oracle AIA resets respective billing date by passing in absolute values for each billing date that needs to be reset.<br>Billing Dates are - Purchase Date, Cycle Start Date, and Usage Start Date. | PCM_OP_SUBSCRIPTION_SET_PRODINFO<br>PCM_OP_SUBSCRIPTION_SET_DISCOUNTINFO | To support validation mode, reset all three billing dates to Actual Delivery Datetime (set prior values to trigger update).<br>To support latency mode, reset Cycle date to Actual Delivery Datetime (set prior value to trigger update). |

| Operation Being Performed in BRM | Dates Set by AIA When the Service is Called | BRM Opcodes Invoked | Expectations of the Order Management System |
|---|---|---|---|
| Two Phase Billing - Billing Fulfillment.<br>Time-Based account-level or service-level Subscription Product/Discount Purchase. | If Service End Date is passed, then Oracle AIA uses that to set the Purchase, Cycle, and Usage end dates for products/discounts purchased. | PCM_OP_SUBSCRIPTION_SET_PRODINFO<br>PCM_OP_SUBSCRIPTION_SET_DISCOUNTINFO | Calculate the Service End Date based on TBO attributes as documented in TBO section. Populate Purchase, Cycle, and Usage start dates. |

# Index


AIACOMOrderFalloutNotificationJMSConsumer, 231

AIAOrderFalloutErrorHandlerExtension, 231

AIAOrderFalloutJMSBridgeService, 231

CommsProcessBillingAccountListEBF, 187

CommsProcessFulfillmentOrderBillingAccountListEBF, 137, 187

CommunicationsBillingEBSV1, 136

CommunicationsBillingResponseEBSV1, 137

CommunicationsCustomerPartyEBSV2, 186

CommunicationsCustomerPartyEBSV2Resequencer, 186

CommunicationsCustomerPartyResponseEBSV2, 192

CommunicationsInstalledProductEBSV2, 145

CommunicationsItemCompositionEBSV1, 44

CommunicationsItemCompositionResponseEBSV1, 44

CommunicationsPriceListEBSV2, 45

CommunicationsSalesOrderEBSV2, 136

CommunicationsTroubleTicketEBSV1, 230

CommunicationsTroubleTicketResponseEBSV1, 230

Configuring Multiple Instances of Oracle BRM, 281

CreateTroubleTicketAIACommsReqImpl, 231

CreateTroubleTicketSiebelCommsProvABCSImpl, 230

Customer Management

- account status sync methodology, 181
- assumptions and constraints, 169
- configuration properties, 202
- cross-references, 198
- data requirements, 170
- DVMs, 197
- EBO implementation maps, 202
- handling errors, 201
- industry AIA components, 184
- integration services, 185
- Oracle BRM interfaces, 182
- overview, 169
- setting up FMW, 195
- setting up Oracle BRM, 197
- setting up Siebel CRM, 197
- Siebel CRM interfaces, 183

CustomerPartyEBSV2, 191

Order Fallout Management

- assumptions and constraints, 220
- business process task flow, 215
- compliance, 259
- configuring, 240
- correction, 215
- creating listeners, 220
- creating trouble tickets, 226
- detection, 215
- EBO implementation maps, 239
- error notification roles and users, 237
- handling errors, 236
- Industry AIA Components, 228
- Integration services, 229
- notification, 215
- overview, 213
- setting up FMW, 233

setting up Oracle AIA, 233
Siebel CRM interfaces, 228
understanding, 213
working with cross-references, 236
working with Oracle AIA, 235
Order Management
business process flow, 86
configuring, 149
configuring properties, 157
creating assets, 113
creating customer data in billing, 89
cross-references, 154
data requirements, 87
delivered integration flows, 114
domain value maps, 152
EBO implementation maps, 157
error handling, 156
friends and family lists, 109
fulfill billing, 101
industry AIA components, 133
initiate billing, 101
integration services, 134
interfacing orders to billing, 94
Oracle BRM interfaces, 132
orchestrating the services, 127
overview, 85
revisions, 105
sending order updates to CRM, 110
setting up FMW, 149
setting up Oracle BRM, 151
setting up Siebel CRM, 151
Siebel CRM interfaces, 133
simple service bundles, 99
solution assumptions and constraints, 129
special rating, 109
submitting orders to order orchestration, 88
two phase billing, 101
Process Integration for Billing Management
setting up FMW, 73
ProcessFulfillmentOrderBillingBRMCommsAddSubProcess, 141
ProcessFulfillmentOrderBillingBRMCommsDeleteSubProcess, 145
ProcessFulfillmentOrderBillingBRMCommsMoveAddSubProcess, 142
ProcessFulfillmentOrderBillingBRMCommsProvABCSImpl, 138
ProcessFulfillmentOrderBillingBRMCommsSuspendResumeSubProcess, 143
ProcessFulfillmentOrderBillingBRMCommsUpdateSubProcess, 144
ProcessInstalledProductSpecialRatingSetListBRMCommsProvABCSImpl, 146
ProcessInstalledProductSpecialRatingSetListSiebelCommsJMSConsumer, 146
ProcessInstalledProductSpecialRatingSetListSiebelCommsReqABCSImpl, 146
ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer, 135
ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl, 135
Product Bundling Methodology
balance groups, 63
basic entity mappings, 49
credit limits, 63
defining products, 52
defining products and discounts, 50
friends and family lists, 64
marketing bundles, 62
penalty products, 63
physical goods, 51
sales catalogs, 52
service bundles, 54

time-based offerings, 70
understanding, 49
Product Lifecycle Management
batch product and discount sync, 25
configuring, 73, 79
DVMs, 77
EBO implementation maps, 79
handling errors, 79
industry components, 42
integration services, 43
Oracle BRM interfaces, 42
overview, 21
prerequisites, 28
product and discount sync, 30
realtime product and discount sync, 22
setting up Oracle BRM, 74
setting up Siebel CRM, 76
Siebel CRM interfaces, 42
simple and customizable products, 28
understanding product bundling methodology, 49
update batch product and discount sync, 27
update realtime product and discount sync, 24
working with cross-references, 78
ProductOptimizedSyncPriceListListSiebelCommsProvABCSImpl, 47
QueryCustomerPartyListSiebelProvABCSImplV2, 191
SyncAccountSiebelAggregatorAdapter, 190
SyncAccountSiebelReqABCSImpl, 190
SyncAcctSiebelAggrEventConsumer, 190
SyncAddressSiebelAggregatorAdapter, 190
SyncBPSiebelAggregatorAdapter, 190
SyncContactSiebelAggregatorAdapter, 190
SyncCustomerPartyListBRM_01CommsJMSConsumer, 194
SyncCustomerPartyListBRMCommsProvABCSImpl, 192
SyncCustomerSiebelEventAggregator, 189
SyncDiscountBRMCommsReqABCSImpl, 46
SyncItemCompositionListSiebelCommsProvABCSImpl, 46
SyncProductBRMCommsReqABCSImpl, 45
TestOrderOrchestrationEBF, 147
Time-Based Offerings
order methodology, 107
product methodology, 70
UpdateSalesOrderSiebelCommsProvABCSImpl, 136