

Oracle® Application Integration Architecture

Oracle Order to Activate Integration Pack for Siebel CRM and
Oracle Communications Order and Service Management
Implementation Guide

Release 2.5

E17427-03

February 2012

Oracle Application Integration Architecture Oracle Order to Activate Integration Pack for Siebel CRM and Oracle Communications Order and Service Management Implementation Guide, Release 2.5

E17427-03

Copyright © 2001, 2012, Oracle and/or its affiliates. All rights reserved.

Primary Author: Oracle Corporation

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle America, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

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

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

Contents.....	1
Oracle Order to Activate Integration Pack for Siebel CRM and Oracle Communications Order and Service Management Implementation Guide Preface	5
Oracle Application Integration Architecture – Foundation Pack 2.5: Getting Started with the Oracle AIA Foundation Pack and Demo	5
Oracle Application Integration Architecture - Foundation Pack 2.5: Core Infrastructure Components Guide	6
Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide	6
Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide	7
Oracle Application Integration Architecture Process Integration Packs.....	7
Additional Resources	8
What's New in This Guide for Release 2.5 IPS3	9
Chapter 1: Understanding the Oracle Order to Activate PIP	11
Oracle Order to Activate Business Process Overview	11
Oracle Order to Activate Business Process Flows.....	15
Data Requirements	28
Solution Assumptions and Constraints	28
Chapter 2: Understanding the Process Integration for Product Lifecycle Management Class Synchronization.....	31
Product Classes in Siebel CRM	31
Design-Time Setup in SCE Studio	31
Query Product Classes and Attributes Integration Flow	32
Support for Effectivity During Design-Time	40
Solution Assumptions and Constraints	42
Siebel CRM Interfaces	42
Industry AIA Components	42
Integration Services.....	43
Chapter 3: Understanding the Process Integration for Order Lifecycle Management.....	47
OLM Process Integration Overview	48
Supporting Order Priorities.....	55
Oracle OSM JMS Message Construction	57

Industry AIA Components	59
Chapter 4: Understanding the Process Integrations for Billing.....	61
Synchronizing Customer Accounts to Billing.....	61
Interfacing Customer Order Actions to Billing	64
Integration Services.....	65
Chapter 5: Understanding the Process Integrations for Siebel CRM	67
Process Integrations for Siebel CRM Overview	67
Submitting Orders from Siebel CRM to Oracle OSM CFS.....	68
Updating Statuses from Oracle OSM CFS to Siebel CRM	70
Siebel CRM Interfaces	72
Integration Services.....	72
Chapter 6: Understanding the Process Integration for Order Provisioning	75
Process Integration for Order Provisioning Overview	75
Creating Provisioning Orders	75
Updating Fulfillment Orders.....	76
Solution Assumptions and Constraints	78
Industry AIA Components	79
Integration Services.....	79
Chapter 7: Using Order Fallout Management in Oracle Order to Activate	83
Using Order Fallout Management in Oracle Order to Activate Overview	83
Creating Trouble Tickets in Oracle OSM	89
Solution Assumptions and Constraints	94
Siebel CRM Interfaces	95
Industry AIA Components	95
Integration Services.....	96
Configuring Order Fallout Using Oracle Order to Bill.....	99
Order to Activate Fallout Services.....	99
Chapter 8: Configuring the Oracle Order to Activate PIP	103
Setting Up Siebel CRM	103
Setting Up Oracle OSM.....	104
Setting Up Fusion Middleware	105
Guaranteeing Delivery.....	106
Using the ESB Resequencer Feature	107
Working with DVMs	109
Working with Cross-References.....	112

Handling Errors	113
Viewing EBO EIMs	113
Configuring Multiple BRM Instances	114
Configuring the Oracle Order to Activate PIP	114
Chapter 9: Configuring Order Fallout Management.....	115
Configuring Properties for Order Fallout Services	115
Installing Sequence and Configurations for Using Fallout	119
Using Error Type to Control Response to Order Fallout	124
Appendix A: Communications Orders Dictionary.....	129
Communications Orders - Order Header Component Attributes	130
Communications Orders - Order Line Component Attributes	139
Appendix B: Using Session Pool Manager	161
Configuring Session Pool Manager.....	161
Index	165

Oracle Order to Activate Integration Pack for Siebel CRM and Oracle Communications Order and Service Management 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.
- 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 Oracle AIA services to work with 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 performance 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 the 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 5: Understanding the Process Integrations for Siebel CRM	
Section: Submitting Orders from Siebel CRM to Oracle OSM CFS	A new section, Defining Transaction Boundaries and Recovery Details, has been added for the Submitting Orders from Siebel CRM to Oracle OSM CFS integration flow. This section documents transactions involved, the database operations, and what actions to take in case of errors.
Section: Updating Statuses from Oracle OSM CFS to Siebel CRM	A new section, Defining Transaction Boundaries and Recovery Details, has been added for Updating Statuses from Oracle OSM CFS to Siebel CRM integration flow. This section documents transactions involved, the database operations, and what actions to take in case of errors
Chapter 6: Understanding the Process Integration for Order Provisioning	
Section: Oracle OSM CFS to Oracle OSM Provisioning Flow	<p>Section revised to update the sequence diagram so that it matches the sequence of events.</p> <p>A new section, Defining Transaction Boundaries and Recovery Details, has been added for the Oracle OSM CFS to Oracle OSM Provisioning integration flow. This section documents transactions involved, the database operations, and what actions to take in case of errors.</p>
Chapter 7: Using Order Fallout Management in Oracle Order to Activate	
Section: Creating Trouble Tickets in Oracle OSM	A new section, Defining Transaction Boundaries and Recovery Details, has been added for the Creating a Trouble Ticket in Siebel CRM by Oracle OSM and Updating a Trouble Ticket in Siebel CRM by Oracle OSM integration flows. This section documents transactions involved, the database operations, and what actions to take in case of errors.

Chapter 1: Understanding the Oracle Order to Activate PIP

This chapter provides an overview of the Oracle Order to Activate business process and discusses:

- Oracle Order to Activate business process flows.
- Data requirements.
- Solution assumptions and constraints.

Oracle Order to Activate Business Process Overview

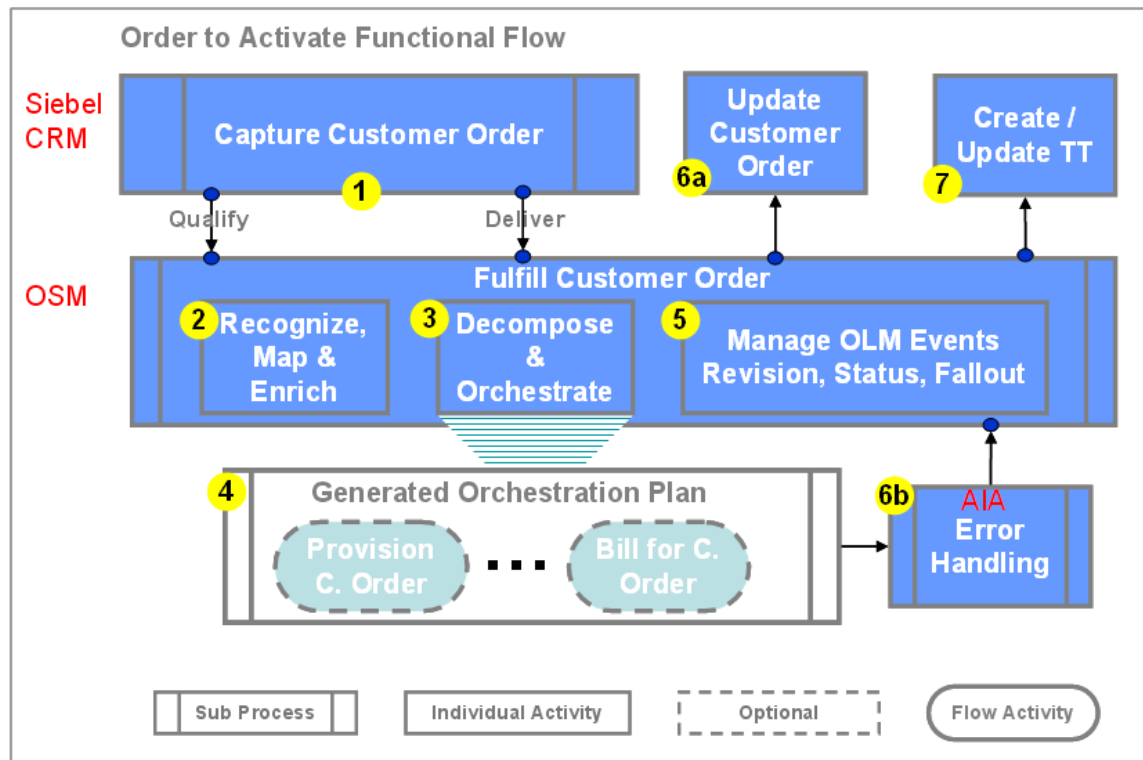
The Oracle Order to Activate business process is at the core of business and operational support systems for any Communications Service Provider (CSP). The process extends from the time a quote or order is created to the time when the goods and services are delivered and properly billed.

The Oracle Order to Activate Process Integration Pack (PIP) works with the Order to Bill PIP and participating applications for these business processes:

- Submitting orders from Siebel CRM to Oracle Application Integration Architecture (Oracle AIA) is available through both the Oracle Order to Activate PIP and the Order to Bill PIP. Taking the submitted orders from Oracle AIA to Oracle Order and Service Management (Oracle OSM) is only available through the Order to Activate PIP. When used without the Order to Activate PIP and Oracle OSM, the Order to Bill PIP assumes the existence of an order orchestration process that handles interfacing the order to billing and other fulfillment systems.
- Creating customer data in Oracle Communications Billing and Revenue Management (Oracle BRM). This integration uses the Order to Bill PIP to synchronize customers with Oracle BRM.
- Interfacing orders to Oracle BRM. This integration uses the Order to Bill PIP to interface orders with Oracle BRM.
- Provisioning orders in Oracle OSM Service Order Management. This integration is only available in Oracle Order to Activate.
- Updating orders and statuses in Oracle OSM Central Order Management. You do this through explicit order updates coming from Oracle OSM Service Order Management available as part of Oracle Order to Activate and Oracle BRM service responses available as part of Order to Bill.
- Sending order updates from Oracle OSM Central Order Management to Siebel Customer Relationship Management (Siebel CRM). This integration exists in both Oracle Order to Activate and Order to Bill.

- Managing order fallout in Oracle OSM and creating trouble tickets in Siebel CRM. This integration uses Oracle Order to Activate to manage order fallout.

This diagram illustrates these flows:



Oracle Order to Activate functional flow

Here are the steps:

- A customer order is captured in Siebel CRM. For some orders, the order may require technical qualification, such as validating that the network has enough capacity to offer the purchased products. After an order is complete and validated in Siebel CRM, the system submits it to Oracle OSM Central Order Management for delivery. The two arrows from Capture Customer Order to Fulfill Customer Order show the Qualify scenario and the Deliver scenario.
- Customer orders (both Qualify and Deliver request types) received in Oracle OSM are first recognized (as Oracle AIA Customer Orders), mapped to fulfillment patterns, and enriched with fulfillment metadata.
- Oracle OSM decomposes and orchestrates the customer order. Oracle OSM divides the order into suborders, called order components, which have cross-order components, cross-order lines, and cross-order dependencies that reflect the specific demands of the CSP.
- The outcome is an order orchestration plan. The fulfillment flow that is produced orchestrates fulfillment requests to different fulfillment providers using preconfigured fulfillment functions, such as sync customer into billing, initiate and fulfill billing, provision order, ship order, and install order.
- Oracle OSM manages order lifecycle management (OLM) events. For cancel and revision requests, Oracle OSM generates and executes compensation plans to efficiently match a change. OLM manages order data and status updates as well as fallout.

6. Throughout the fulfillment process, Oracle OSM maps fulfillment function responses to common statuses, which are then aggregated into order line statuses and order header status values. The status management capability updates Siebel CRM with relevant customer status and milestone values. Oracle OSM updates Siebel CRM when order lines reach their point-of-no-return (PONR) to prevent the submission of new revisions. It also updates Siebel CRM with any enrichment to order lines that may have occurred during fulfillment.

Errors may occur for many reasons. Oracle AIA reports such errors to Oracle OSM for fallout management. Additionally, validation logic in Oracle OSM may raise fallout incidents.

7. Oracle OSM detects, reports, and resolves order fulfillment fallout incidents such as system, validation, and fulfillment errors. The Oracle approach creates trouble tickets in Siebel CRM to take advantage of the rich notification, reporting, and management capabilities of Siebel CRM.

For more information about Oracle OSM, see the *Oracle Communications Order and Service Management Concepts Guide*.

Note that a PIP in Oracle OSM terms differs from a PIP in Oracle AIA terms. In Oracle OSM, a PIP is a pre-built cartridge whereas in Oracle AIA, it is a collection of processes.

Oracle OSM delivers these pre-built cartridges for use with the Oracle Order to Activate PIP:

- CommunicationsSalesOrderFulfillmentPIP
- CommunicationsProvisioningOrderFulfillmentPIP
- OracleCgbuCommunicationsORPFalloutPIP
- SIFalloutPIP

Additionally, Oracle OSM provides an Oracle AIA Emulator, which you can use to emulate an order.

For more information about how to install and deploy the delivered cartridges and the emulator, see the *Oracle Communications Order and Service Management Application Integration Architecture Order to Activate Cartridge Guide*.

Note: The focus of this guide is the automated integration points among Siebel CRM, Central Order Management, Service Order Management, and Billing. This guide does not cover process details within Service Order Management, for example, service design, assign, and activation.

Terminology

Note that we use these terms interchangeably in this document and its graphics:

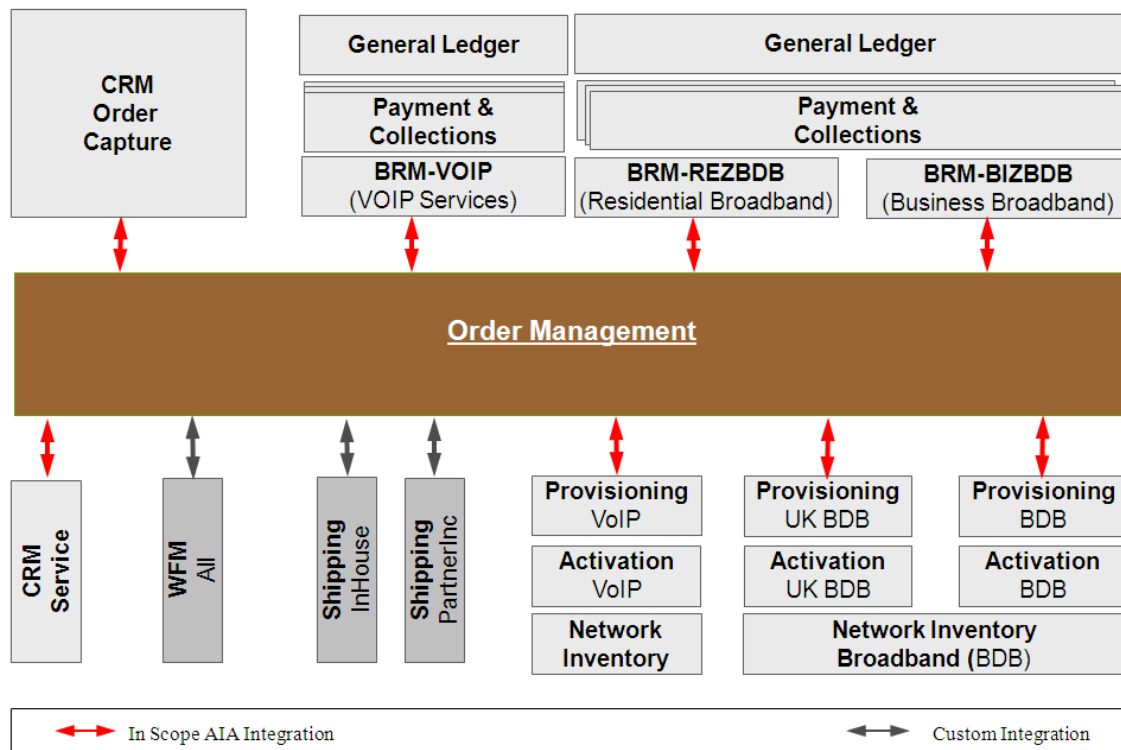
- **Fulfillment or Central Fulfillment System (CFS):** Order Management, Oracle OSM Central Order Management (Oracle OSM COM)
- **Provisioning:** Service Fulfillment, Oracle OSM Service Order Management (Oracle OSM SOM)

Typical Topology

A central piece of the Oracle Order to Activate solution is the Order Lifecycle Management system. Traditionally, CSPs deployed stovepipe BSS and OSS solutions with middleware-based custom order orchestration solutions. Deployment consolidation for cost savings, convergent bundling, and time-to-market demands are fostering increasingly complex requirements for the orchestration solution. These requirements include sophisticated order mapping, order decomposition, status composition, fallout management, changes to in-flight orders, future-dated orders, and cross order dependencies, among others. You cannot easily meet these requirements using middleware-based custom solutions.

However, Oracle and a large group of leading CSPs concluded that a prominent and distinct role exists for a commercial ready-to-use OLM solution. We recognize this concept as the Order Management solution responsible for central fulfillment functionality and, therefore, the CFS references throughout this document. Oracle OSM refers to this concept as Central Order Management.

This diagram illustrates a typical Oracle Order to Activate deployment topology. The Order Management system is at the center of this topology:



Typical Oracle Order to Activate deployment topology

The topology shown is typical of most CSPs, although many could include more fulfillment system types (for example, billing or workforce management) and fulfillment system stacks. Order management is at the center of the Oracle Order to Activate deployment, with order capture systems passing orders to the Order Management system. The Order Management system decomposes the order into suborders, each of which targets a particular fulfillment provider (that is, system instance) called order components. The topology shown uses three billing providers based on customer segment: wholesale, residential, and business. It uses three provisioning stacks based on service family and geography: VoIP, UK Broadband, and Broadband. It uses two shipping providers, one for in-house products and another for partner supplier products. Finally, it uses one workforce management provider and one separate Siebel CRM service provider (for trouble ticketing).

Oracle Order to Activate Business Process Flows

The following subsections describe the Oracle Order to Activate business process:

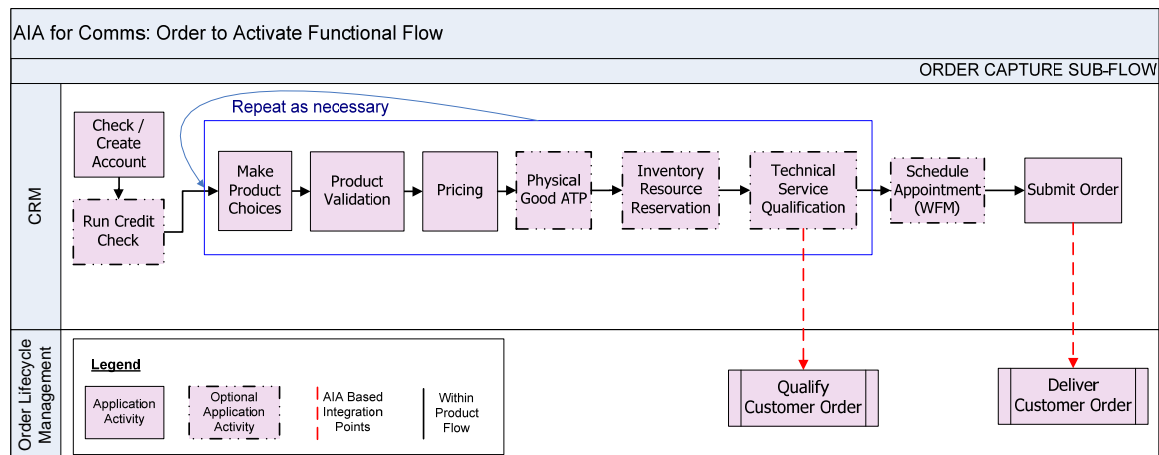
- Understanding the order capture subflow.
- Understanding the deliver customer order subflow.
- Understanding the qualify customer order subflow.

This section focuses on the elements of the Oracle Order to Activate business process flow that are relevant to its integration with Order Management. This section describes:

- Inbound and outbound integration points with Order Management for deliver and qualify customer order subprocesses.
- Functional expectations for product model and product mappings to facilitate order fulfillment.
- Functional methodology governing the distribution of roles across Siebel, Order Management, and key fulfillment systems, primarily Billing and Provisioning.

Understanding the Order Capture Subflow

The following diagram illustrates a typical order capture flow. This flow varies by CSP and may vary by service family, customer segment, line of business, and other considerations. Two important integration points between Siebel CRM and Order Management are illustrated for a Qualify customer order and a Deliver customer order. In Siebel CRM, a customer order is known as a sales order. In general, order-based system interactions between different BSS and OSS systems require that order decomposition and orchestration go through the Order Management layer. For the Oracle Order to Activate flow, at least two system interactions exist: Qualify customer order to validate the availability of a service design and the capacity to fulfill the customer order; and Deliver customer order to fulfill the products and services purchased by the customer or fulfill actions on existing customer assets.



Order capture subflow

Reading this Order Capture Subflow Diagram

The previous diagram shows two swim lanes, one for Siebel CRM and another for OLM. Each swim lane includes the typical application activities and user interactions that are part of that application. Arrows between such activities represent the typical sequence of events within the same application. Arrows across swim lanes represent system interactions across applications. See the legend in the diagram for other details.

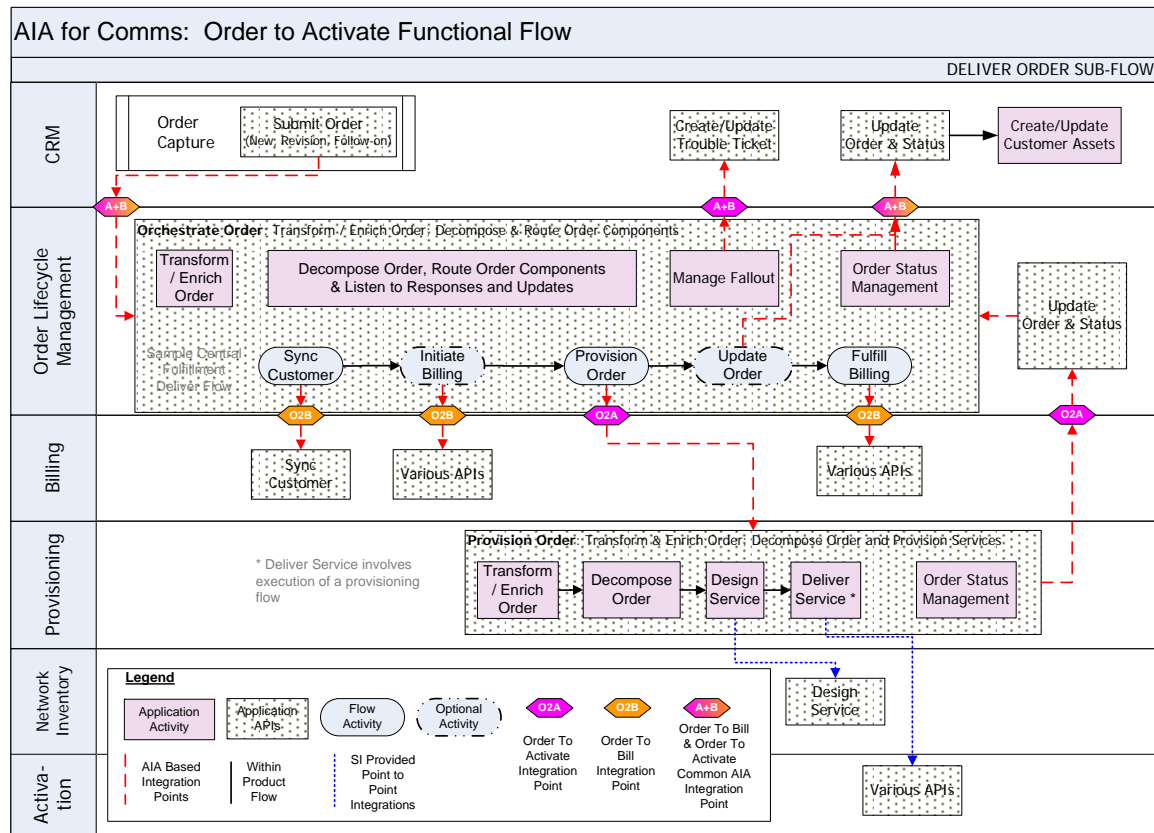
A typical flow starts with creating new customers and updating existing customer information. Depending on customer segment, line of business, or another consideration, you might capture customer information earlier, for example, when you create an opportunity or quote that you update during order capture. Depending on its business policy, some customers may pass through a credit check before starting the process of making product choices. While making product choices and at other points in the process, such as while capturing an order, the Siebel CRM system performs several validations. You price selected products and product options using relevant pricing logic. When physical goods are involved, the order capture process typically checks availability to purchase. For some services, resource reservation (for example, a phone number) also occurs during order capture. Before you submit an order and depending on the business practices of the CSP, the order may need to pass a technical service qualification. Some CSPs also require scheduling an engineer (when needed) at the time of order capture to synchronize both the availability of an engineer and a customer. After completing an order and having it validated, you submit it to start the delivery process.

The scope of this release of Oracle Order to Activate covers the integration points between Siebel CRM and Oracle OSM to qualify and deliver customer orders. System implementers are responsible for other integrations.

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.

Describing the Deliver Customer Order Subflow

The following diagram illustrates a typical Deliver customer order flow:



Deliver customer order flow

Reading this Deliver Customer Order subflow Diagram

This diagram shows six swim lanes, one for each of the following applications: Siebel CRM, Order Management, Billing, Provisioning, Network Inventory (Service and Resource Inventory), and Activation. Each swim lane includes the typical application activities and user interactions that are part of that application. Arrows between such activities represent the typical sequence of events within the same application. Arrows across swim lanes represent system interactions across applications. The Oracle AIA diamonds between swim lanes represent existing or planned Oracle AIA-based integration points. The diamonds are labeled O2A for integration points that are part of Oracle Order to Activate, O2B for integration points that are part of Oracle Order to Bill, and A+B for integration points that are common to both. See the legend in the diagram for other details.

An Order Management system must recognize three kinds of customer orders:

1. New orders

Orders for new purchases or changes to already delivered products. Already delivered products are known as customer assets.

2. Revision orders

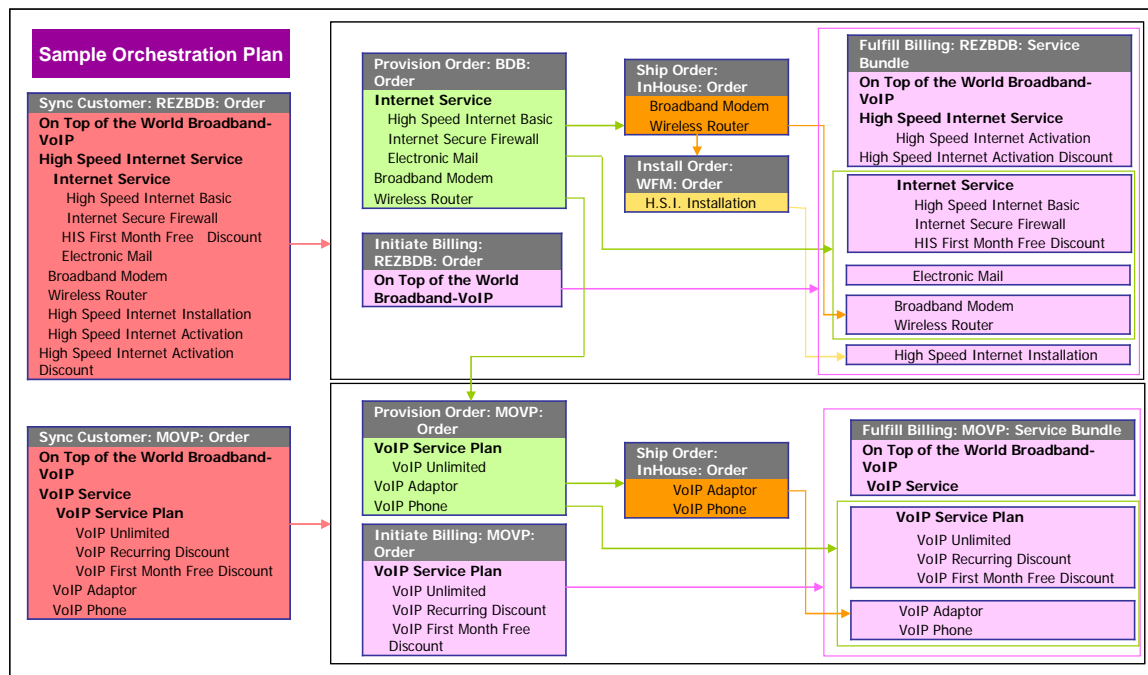
Amended versions of already submitted orders. You can submit revision orders to fulfillment while the revised order is in a fulfillment state that allows for order changes.

3. Follow-on orders

Orders that have a fulfillment completion dependency on other orders.

This flow starts with a new order, an order revision, or a follow-on order submitted from Siebel CRM to Oracle OSM. Order Management performs these key functions:

4. Transforms and enriches the order. It maps order lines to fulfillment flows and enriches it with fulfillment metadata and other relevant data.
5. Decomposes and routes the order. It divides the order into suborders, which are called order components. Order components have cross-order components, cross-order lines, and cross-order dependencies that reflect the specific needs of the CSP. The outcome is an order orchestration plan that is executed at the computed fulfillment start time to meet the requested delivery date. The previous diagram illustrates a simple flow; however, the flow is typically more complex, as shown in this next diagram. The produced fulfillment flow orchestrates fulfillment requests to different fulfillment providers using preconfigured fulfillment functions, such as sync customer into billing, initiate and fulfill billing, provision order, ship order, and install order. The Order Management decompose and route order function also generates compensation plans that are associated with revision orders.

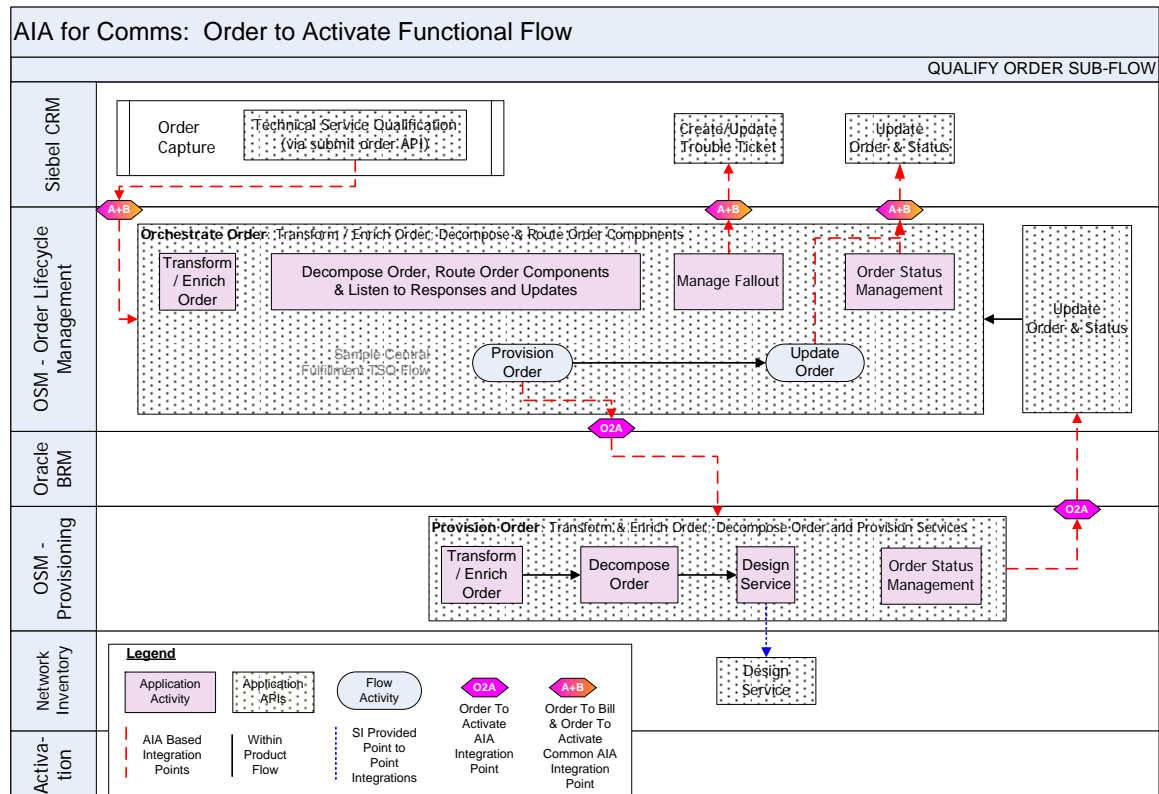


6. Manages fallout. The integration provides for detection, reporting, and resolution of order fulfillment fallout conditions such as system, validation, and fulfillment errors. The Oracle approach is to create trouble tickets in Siebel CRM to take advantage of its rich notification, reporting, and management capabilities.

7. Manages status. It maps fulfillment function responses to common statuses, which are then aggregated into order line statuses and order header status values. The status management capability updates Siebel CRM with relevant customer status and milestone values. It also updates Siebel CRM when order lines reach their PONR to prevent the submission of new revisions.

Describing the Qualify Customer Order Subflow

The following diagram illustrates a typical Qualify customer order flow:



Qualify customer order flow

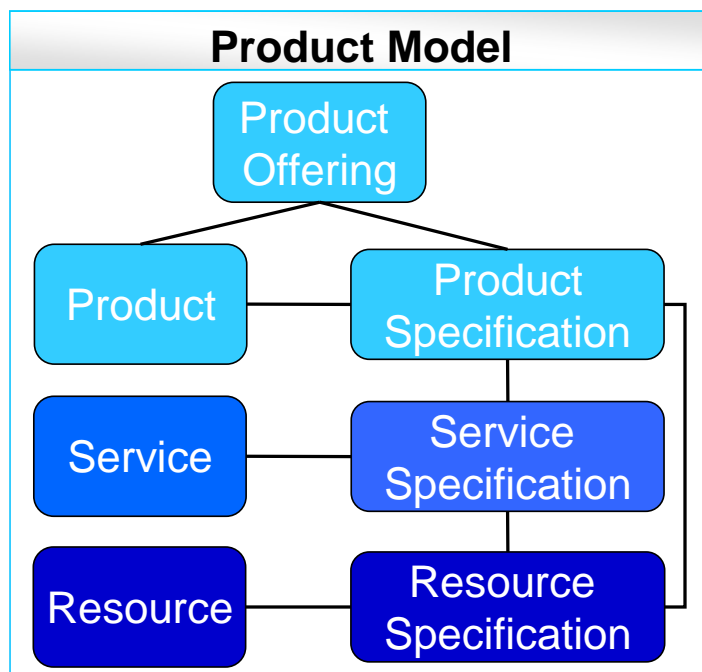
Reading this Qualify Customer Order Subflow Diagram

This diagram shows six swim lanes, one for each of the following applications: Siebel CRM, Order Management, Billing, Provisioning, Network Inventory (Service and Resource Inventory), and Activation. Each swim lane includes the typical application activities and user interactions that are part of that application. Arrows between such activities represent the typical sequence of events within the same application. Arrows across swim lanes represent system interactions across applications. The Oracle AIA diamonds between swim lanes represent existing or planned Oracle AIA-based integration points. The diamonds are labeled O2A for integration points that are part of Oracle Order to Activate and A+B for integration points that are common to Oracle Order to Activate and Order to Bill. See the legend in the diagram for other details.

This flow starts with a request to qualify the technical validity of a customer order submitted from Siebel CRM to Order Management. Order Management performs the same four functions detailed for the Deliver customer order with one key distinction: the metadata used and the fulfillment flow produced is for qualifying the customer order rather than delivering the customer order. Deliver order flows and Qualify order flows produce different order and order line status updates.

Product Definition and Mapping Design Considerations

The product and service definition methodology has the greatest effect on time to market and on the cost of a Oracle Order to Activate deployment. Often, you define products and services in different network, IT, and business departments to serve the best interests of individual departments. This approach creates a challenge for bridging the gaps at runtime. We recommend a balanced approach that requires departments to make calculated compromises that result in simplified overall product life cycle and order life cycle business flows.



Product model

The product model diagram aligns with Tele Management Forum (TMF) terminology and guidelines.

A balanced model produces a catalog with product specifications represented by the least number of entities. Product specifications represent unique capabilities with commercial value but only sold through product offerings. A more technical definition is that product specifications are types of products.

The product model shown covers the three TMF SID key entities: product, service, and resource.

Product offerings represent tangible and intangible goods and services made available for a certain price to the market in the form of product catalogs. Product offerings take one of three possible forms—simple offerings, bundled offerings, and promotional offerings:

- Simple offerings are product offerings of a single good or service.
- Bundled offerings are a grouping of two or more simple offerings into a single offer.

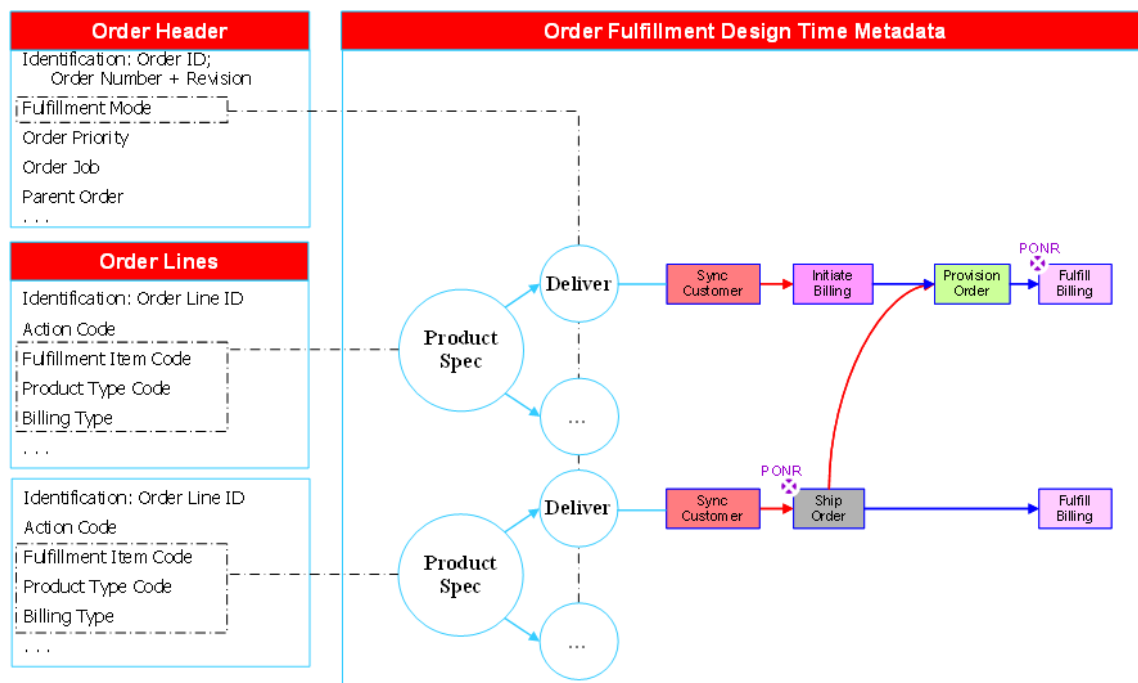
- Promotional offerings are time-bound, contract-bound, or discounted combinations of simple and bundled offerings.

A key element of the Oracle methodology is a one-to-one mapping of every order line to a product specification. This approach is key to achieving fast time-to-market and low-cost operations. The Oracle solution facilitates this mapping by associating product offerings with a product class in Siebel or Product Hub for Communications through the Fulfillment Item Code attribute.

Order Management acts on customer orders. Customer orders are composed of order lines. Each order line is represented by an action and a subject. Actions are verbs that represent the nature of the customer request, such as ADD to purchase an offering, UPDATE to modify a customer's subscription to an offering (for example, Customer Asset), and so on. A subject is the target of the action and can represent an offering, an asset, a discount, and so forth.

In the service fulfillment layer, a product specification can map to one or more technical services. A technical service is composed of one or more technical services and resources. The mapping from a customer order to a service order requires specific metadata modeled on products, product specifications, and service and resource configurations. Details about the service fulfillment layer are not in the scope of this guide.

This diagram illustrates how the Order Management system takes advantage of the product model to map customer order lines to fulfillment flows according to the Oracle methodology. Other approaches may be plausible, but you must maintain a balanced approach that facilitates achieving the business objectives of fast time-to-market, and low-cost operations.



Mapping customer order lines to fulfillment flows

At runtime, order capture copies key product offering attributes to each order line. These attributes include Fulfillment Item Code, Product Type Code, and Billing Type. Order management uses these attribute values to determine the corresponding product specification. The order header Fulfillment Mode attribute value determines the fulfillment requested type (for example, Deliver or Qualify). The intersection of a product specification and fulfillment request type determines the fulfillment actions and dependencies involved. When combined for all order lines in an order, an order fulfillment plan is generated dynamically.

Order Update Design Considerations

Superior visibility and experience are key business drivers for the Oracle Order to Activate process. Oracle OSM facilitates configurable and streamlined order fulfillment statuses and propagation across the fulfillment systems and Siebel CRM. Decomposition of an order into order components, as well as multiple fulfillment steps, places an extra burden on the Order Management system to manage the translation of fulfillment function responses to common status attribute values. Each response may contribute to different order line and order header status values, which are also the responsibility of the status management function of the Order Management system.

A single status attribute is not sufficient to provide comprehensive visibility into the fulfillment process. Oracle adopted the extended set of attributes listed in the following table as part of its methodology to implement the Oracle Order to Activate business process:

Functional Attribute Name	Usage
Order Header / Fulfillment Status	Updates Siebel CRM on the current status of order fulfillment at a high level. This is different from the Siebel Status attribute. The Fulfillment Status attribute tracks the order status while in fulfillment. The Status attribute tracks the order status across order capture and order fulfillment. Only Complete and Cancelled fulfillment status values are reflected (internally by Siebel) on Status.
Order Header / Status Context	Provides details about the current status. Implementers can configure this value. The Fulfillment Status attribute tracks the order status while in fulfillment. The Status attribute tracks the order status across order capture and order fulfillment. Only Complete and Cancelled fulfillment status values are reflected (internally by Siebel) on Status.
Order Line / Fulfillment Status	Provides a high-level update of the current status of order line fulfillment to Order Management and Siebel CRM.
Order Line / Milestone	The last reached fulfillment milestone.
Order Line / Status Context	Provides details about the current status. An implementer can configure this value. You can use Context Text to indicate: <ul style="list-style-type: none"> Required customer interaction. If delivery is expected to be delayed. Milestone/fulfillment function in which a failure occurred. Cause of a cancellation or who canceled an order.
Order Line / Point-of-no-return	Indicates if Siebel should allow revisions to an order line or submission of previously created revisions to an order line. Oracle OSM fulfillment flows allow configuration of setting a hard PONR when a condition is met for a particular order line. When a hard PONR is established for an order line in Oracle OSM, an update is issued to reflect the same in Siebel CRM. Siebel uses the PONR to block users from revising order lines.
Order Line / Actual Delivery Date-Time	Determines the date when the purchased product or service is considered available to the customer. This date may be the date physical goods are shipped, delivered, or their receipt acknowledged. For service-based products, this date is when the service is activated. This date is computed in the fulfillment flow.
Order Line / Expected	Provides the expected delivery date for an order line. When Siebel CRM creates the

Functional Attribute Name	Usage
Delivery Date-Time	order, the system provides this value by default. Oracle OSM uses this date to communicate changes for specific order line dates to Siebel CRM.

When referring to order or order line status in this guide, we are referring to values for all of the previous attributes. Some CSPs do not realize the processing complexity that is introduced when different fulfillment status values are used for different services. Users may need to configure additional status values, but we recommend they use a streamlined set of status values across product specifications. This practice has two advantages:

1. Enhances understanding for both the CSR and the customer.
2. Maximizes fulfillment flow reusability and enhances the time to market.

In addition to using streamlined statuses, the order management implementation should optimize the propagation of status changes as follows:

1. Not all status changes are relevant to the CSR or the customer in Siebel CRM. Therefore, not all changes should be propagated to Siebel CRM.
2. Not all status changes need to be reflected instantly, therefore, a throttling mechanism should be provided. Some statuses, however, should be reflected instantly, such as PONR being reached. Careful analysis is required to determine which status changes require instant propagation and which can wait. Too many status updates may cause performance and throughput problems.

Some status attribute values drive specific logic in Siebel CRM and should be preserved. For Siebel, these values are Complete and Canceled. Both affect the Asset Maintenance logic in Siebel. The Complete status value drives the logic to create and update Siebel Assets. The order management implementation should turn the status value to Complete for a parent order line only after the order line and all of its subordinate order lines (within the order hierarchy) have completed fulfillment successfully. A Canceled order status excludes the order from a Siebel calculation of the future state of the asset when creating follow-on or future-dated orders.

When making data updates to an order line in Siebel CRM, an order management implementation should avoid sending the data updates before the order line reaches the PONR. If a revision was created before the data update was sent to Siebel CRM and then the revision was submitted, then the data updates may be lost. Fulfillment flows in Order Management should delay sending data updates for as long as possible but no later than when the Complete status value is sent to Siebel. If any data update occurs after the Complete status value is propagated to Siebel, then the updated data would not be saved for the asset.

These practices are the default behavior of the Oracle OSM cartridges for Oracle AIA.

Revision Order Design Considerations

The fulfillment of some services may take days and weeks, and some business-to-business (B2B) and infrastructure projects may take months to complete. During this period, customers change their minds and request changes to their orders that become revision orders in Siebel. In many cases, continuing the base order when a revision is submitted is costly for the CSP, and sometimes the operation cannot be fully undone. For these reasons, support for revision orders provides the following benefits:

- Enhances customer satisfaction by allowing customers to change their orders within an agreed-upon limit.

- Reduces the costs associated with fulfilling unwanted goods and services requests and wasting system capacity, unrecoverable resources, acquired stock, and so forth.
- Reduces human intervention to manually retrofit data records when recovery cannot be automated.

A CSP can attain these benefits by adopting a balanced order revision policy. You must establish a PONR in the fulfillment flow of each product specification so that either an order revision is technically impossible or the cost of doing so outweighs the benefit of allowing a revision.

Siebel allows order line revisions as long as the order line did not complete. In this release, we enhanced Siebel to block the submission of an order line revision when it reached the PONR (Change Cost field = PONR). Despite this enhancement, a revision order might be submitted at the same time an order line reaches the PONR. The Oracle OSM implementation also rejects revisions that violate the PONR, placing them in fallout.

To avoid problems associated with stale revisions (that is, revisions that do not progress in Siebel and become out of sync with their underlying asset), we enhanced Siebel to allow only one pending revision for each order.

You should configure a PONR in the fulfillment flow of each product specification and propagate it to Siebel CRM at runtime.

After a revision is submitted, Oracle OSM recognizes a revision of an in-flight (being fulfilled) or pending (waiting turn to be fulfilled) order, and it ignores older revisions that may be out of date because of slow messaging queues or system outages. The Oracle OSM Order Change Management (Oracle OSM OCM) component is responsible for suspending the revised order, computing a compensation plan, and considering the revision order changes when fulfilling the order.

The Order Management system receives a revision order that represents the complete customer intent, not a delta order with changes from the original order. Revisions may include new order lines, modified order lines, and canceled order lines.

Oracle Order to Activate supports these cancelation patterns:

- Cancel the entire order. Siebel introduced a Cancel Order button that resubmits the order with the Fulfillment Mode order header attribute value set to CANCEL.
- Drop an order line. When an ADD for a new product is dropped, Siebel drops the order line from the revision order. Creating a revision order that drops all order lines produces the same effect as canceling the entire order.

For product attributes that are saved to the asset in Siebel, the order message includes Prior Value for changed attributes. Order Management and Fulfillment systems take advantage of these attributes to determine the changes and to undo unwanted actions. The order management implementation is responsible for maintaining correct Prior Values when attribute values change during fulfillment. The order management implementation should pass Prior Values to systems with the capability to process revisions.

For more information about assetable attributes, see [Appendix A: Communications Orders Dictionary](#).

Compensation for provisioning typically involves technical knowledge, and we recommend that you keep that knowledge out of the order management layer. The order management implementation should delegate to the provisioning stack for any provisioning-specific compensation by passing the revision order component as is to the provisioning system. The provisioning system, in turn, is responsible for computing delta changes, computing a compensation plan, and executing the revision. Oracle OSM is capable of handling revision orders at the order management and provisioning layers through its Oracle OSM OCM advanced capabilities.

For more information about OSM, see the *Oracle Communications Order and Service Management Concepts Guide*.

Billing Fulfillment Design Considerations

Billing fulfillment scenarios lead to one of two fulfillment patterns, each of which must be supported by the order management implementation.

Two-Phase Billing

In this pattern, a service is interfaced to billing twice:

1. **Initiate Billing.** The service and purchased products are interfaced early in the fulfillment flow and before actual delivery dates are known.
2. **Fulfill Billing.** Accurate billing dates are updated in billing after the order is delivered and the actual delivery date is known.

Two scenarios require this pattern.

Billing Scenario #1: Phased for Time Latency

In this scenario, the CSP has these concerns:

- Operational or deployment conditions produce a time lag between the time a service is made available for customer use and the time the service is interfaced into billing. As a result, usage records can go into error logs and the CSP may lose revenue.
- CSPs attempt to plan fulfillment of future-dated orders to meet the requested delivery date, often using a safe margin that produces a time lag between the time a service is made available for customer use and the requested delivery date.

In these cases, the usage cycle needs to start sooner than the billing cycle date. The fulfillment flow should be constructed such that the Usage Start Date is set to the current date during Initiate Billing, and the Cycle Start Date is set to a distant future date. At the time of Fulfill Billing, the Cycle Start Date is then reset to match the Actual Delivery Date or Requested Delivery Date, depending on business practices and legal requirements.

Billing Scenario #2: Phased for Validation

In this scenario, the CSP has these concerns:

- Inadequate controls are in place to guarantee that valid orders interface to billing. As a result, the CSP faces a high rate of invalid orders.
- The costs associated with delaying order line validation for interfacing to billing are prohibitive.

In these cases, orders need to be interfaced to billing early in the fulfillment flow to prove that the order can be interfaced successfully. The fulfillment flow should be constructed such that the Usage Start Date and the Cycle Start Date are set to a distant future date during Initiate Billing. At the time of Fulfill Billing, the Usage Start Date and Cycle Start Date can be reset to match the Actual Delivery Date or Requested Delivery Date, depending on business practices and legal requirements.

Single-Phase Billing

In this pattern, a service is interfaced to billing through Fulfill Billing towards the end of the fulfillment flow, after the order is delivered and the actual delivery date is known. The following scenario requires this pattern.

Billing Scenario #3: All at Once

This scenario is the most common. In it, the CSP does not have the concerns mentioned previously; interfacing to Billing takes place after the service or product is made available to the customer. The interpretation of *made available* may vary among CSPs, based on jurisdiction and based on whether the subject is a service or a physical good. For example, physical goods that require no network activation or onsite installation might be billed immediately after the goods are shipped. The exact timing is built into the fulfillment flows associated with the underlying product specification through the Actual Delivery Date and other billing date attributes.

In all of the previous cases, Oracle BRM requires a Service ID value and the Purchase Date be set to the earliest Usage Start Date or Cycle Start Date interfaced to billing.

For more information, see the *Siebel CRM Integration Pack for Oracle Communications BRM: Order to Bill Implementation Guide*, “Supporting Single Phase versus Two-Phase Billing.”

For all of the previous cases, the order management fulfillment flow should maintain accurate Prior Values for the date attributes.

Future-Dated Orders Design Considerations

Support for future-dated orders ranges from a necessity to a convenience for CSPs. It allows CSPs to comply with requested availability, update, move, or disconnect dates. It also allows CSPs to manage customer-initiated or CSP-initiated periods of Suspend and Resume of services.

Oracle OSM accepts future dated orders and computes a fulfillment start time that allow for services and goods to be delivered as close as possible to the requested delivery date. Determination of a fulfillment start time constitutes a time-based dependency for the start of fulfillment flow.

For more information about OSM, see the *Oracle Communications Order and Service Management Concepts Guide*.

Avoid creating multiple future-dated orders against the same asset. They create a complex future asset state that is difficult for both the CSR and the customer to comprehend. We recommend that only a trained CSR be allowed to enter multiple future-dated orders against the same asset and only when required. When introducing an order line against the same asset with a Requested Delivery Date sooner than another already created order, you should revise the latter to ensure that the order is based on an updated future state of the asset.

Follow-On Orders Design Considerations

The fulfillment of some services may take days and weeks, and some B2B and infrastructure projects may take months to complete. During this period, customers change their minds and request order changes that become revision orders in Siebel if the subject order lines did not reach the PONR or otherwise become follow-on orders. In many cases, not taking an order pending the completion of in-flight orders is not acceptable; therefore, Siebel simulates the future state of in-flight orders and allows for the creation and submittal of follow-on orders that are nothing more than change orders based on the projected future state of a customer's assets.

Follow-on orders are change orders that involve a dependency on the future fulfillment of at least one other order line in an order that is currently in flight. The follow-on order line may change another in-flight order line that is beyond the hard PONR or that depends on the future asset state of that line, as through an explicit dependency established in Siebel.

Follow-on orders are created and submitted to Order Management immediately, and Order Management provides for managing the fulfillment dependency between the follow-on order and other base orders. This responsibility is similar to the responsibility for determining the correct processing time for future-dated orders.

Order Priorities Design Considerations

Order fulfillment priorities reflect business factors such as customer experience and revenue recognition, as well as operational factors such as deployment throughput capacity and fluctuation.

Order fulfillment priority is specified in Siebel CRM and honored by the Order Management system along with the entire fulfillment deployment. Message queues are configured to honor priorities.

When two orders with different priorities are submitted by Siebel CRM and the two orders are within the same system and waiting for service, we expect that the higher priority order will be given precedence over the lower priority order.

The exception is for cases in which system interactions need to be sequenced based on time events. For example, status updates to orders need to be sequenced to guarantee that activities triggered by different status values occur in the correct sequence. For these cases, the system ignores the order priority and uses a sequencer service that sequences messages that meet a certain criteria, such as updates for orders belonging to the same customer account, based on message queuing time.

Fallout Management Design Considerations

Fallout conditions could arise as a result of internal conditions in Order Management and the fulfillment flow, such as a validation error or unknown product specification, respectively. In addition, Oracle AIA Error Handling is another channel for fallout conditions. It captures system interaction errors. Oracle Order to Activate integrates Oracle OSM and Oracle AIA Error Handling to process such errors as fallout conditions. Fallout management in Oracle OSM creates and updates trouble tickets in Siebel CRM using Oracle AIA to facilitate automatic assignment, notification, reporting, and management of fallout incidents across systems and departments.

For more information about order fallout, see [Chapter 7: Using Order Fallout Management in Oracle Order to Activate](#) and the Oracle OSM product documentation.

Data Requirements

Data requirements vary by fulfillment system. The Sales Order enterprise business object (EBO) includes a vast set of attributes that are sufficient for most fulfillment systems, and it is extensible.

- For Oracle OSM, this table lists mandatory attributes:

Order Header EBO Attributes	Order Line EBO Attributes
Order ID	Line ID
Order Number	Base Line ID
Revision	Action Code
Fulfillment Mode	Product Name
Order Type	Product Type

- Oracle BRM also has mandatory attributes.

For more information, see the *Siebel CRM Integration Pack for Oracle Communications BRM: Order to Bill Implementation Guide*, “Order Management Integration Data Requirements.”

Solution Assumptions and Constraints

The solution assumptions and constraints follow.

- Service points in Siebel are implemented as assets and typically are uploaded into Siebel from external sources. Ideally, service points are mastered in a common place and shared between Siebel CRM and Network Inventory (Service and Resource Inventory). The integration assumes that at least one of the following is true:
 - The determination of service point in Siebel CRM is irrelevant to Service and Resource Inventory.

- The determination of service point in Siebel CRM is replicated in Service and Resource Inventory (for example, the same result is achieved).
 - The service point attribute value is unique and common across Siebel and Service and Resource Inventory, such that Service and Resource Inventory can use the value directly.
 - The service point attribute value is a cross-reference that is understood by Service and Resource Inventory; no Oracle AIA cross-reference exists for this attribute.
2. In Siebel, order revisions are created as a copy of the previous revision and then changes are made to the revision. When created, the first order reflects the customer assets at the time. Revisions sometimes stay for a long period in Siebel CRM without submittal and may become stale if the customer assets change in the interim. The expectation for Siebel CRM is that it would ensure that the revision order data is up to date with the customer assets at the time the order is submitted. Any customization of Siebel or integration to a different CRM system should ensure that revision orders are brought up to date with the customer assets state before submitting the order to Order Management.
 3. Multiple future-dated orders require special care from CSR to ensure that orders are submitted in the correct sequence and that new orders do not invalidate formerly submitted orders. We recommend that providers limit future orders to one per customer.
 4. Follow-on orders, if submitted before base orders, are processed as base orders. CSRs must make sure they submit base orders first for follow-on orders dependency on base orders to take effect in Oracle OSM.
 5. Mixing future-dated, follow-on, and revision orders requires a well-trained CSR because some scenarios could produce unintended results. Ensure that:
 - Create follow-on events only when base orders are past the PONR.
 - Create and submit revisions as soon as they are firm; when revisions are pending, do not create follow-on orders before you discard revisions.
 - You can create future-dated orders against the same asset as long as you create them in chronological order.
 6. Siebel does not guarantee correct assets if follow-on orders are created before modified order lines reach the PONR. You should create follow-on orders only after modified order lines reach the PONR and any pending revisions are discarded.
 7. Siebel can capture revisions to order Due Date in Siebel (Requested Delivery Date in Oracle AIA) and submit them to Oracle OSM.
 8. Revising the requested delivery date for an order will only affect Oracle OSM if the base order did not start fulfillment by the time the revision was received in Oracle OSM.
 9. While in Siebel, you can create an Oracle AIA follow-on order even before an order reaches the PONR. Oracle OSM only accepts follow-on orders when the base order is already past the PONR.
 10. For this release, Oracle OSM does not support revisions to base orders with follow-on orders.

For more information, see the *Oracle Communications Order and Service Management Application Integration Architecture Order-to-Activate Cartridge Guide*.

11. For orders with multiple requested delivery dates on the same order, Oracle OSM honors the earliest requested delivery date for all order lines.
12. Oracle OSM only accepts incoming messages in UTC/GMT, and it sends messages in UTC/GMT. For the Oracle Order to Activate PIP to work, you must set up Siebel to send messages in UTC/GMT.

Note. All participating applications have assumptions and constraints. Always refer to the specific product documentation for information about any limitations for the Oracle Order to Bill PIP, Oracle OSM, and Siebel.

Chapter 2: Understanding the Process Integration for Product Lifecycle Management Class Synchronization

This chapter discusses:

- Product classes in Siebel Customer Relationship Management (Siebel CRM).
- Design-time setup in Service Creation Environment (SCE) Studio.
- Query product classes and attributes integration flow.
- Support for effectivity during design-time.
- Solution assumptions and constraints.
- Siebel CRM interfaces.
- Oracle Billing and Revenue Management (Oracle BRM) interfaces.
- Industry Application Integration Architecture (AIA) components.
- Integration services.

Product Classes in Siebel CRM

In Siebel, product classes provide a way to organize and maintain product attributes. Product attributes describe the characteristics of products. Examples include color, size, speed, and so on. You define product attributes and associate them with one or more product classes. Product attributes have a range of values associated with them called the attribute domain. For example, an attribute bandwidth has a range of values, such as 2Mbps, 3 Mbps, 5 Mbps, and so on.

Siebel supports product class hierarchies using inheritance; that is, the attributes that are associated with a class are automatically inherited by all of the subclasses in the hierarchy. The subclasses can be nested as deeply as needed. The standard recommendation is to use three levels in the hierarchy.

Design-Time Setup in SCE Studio

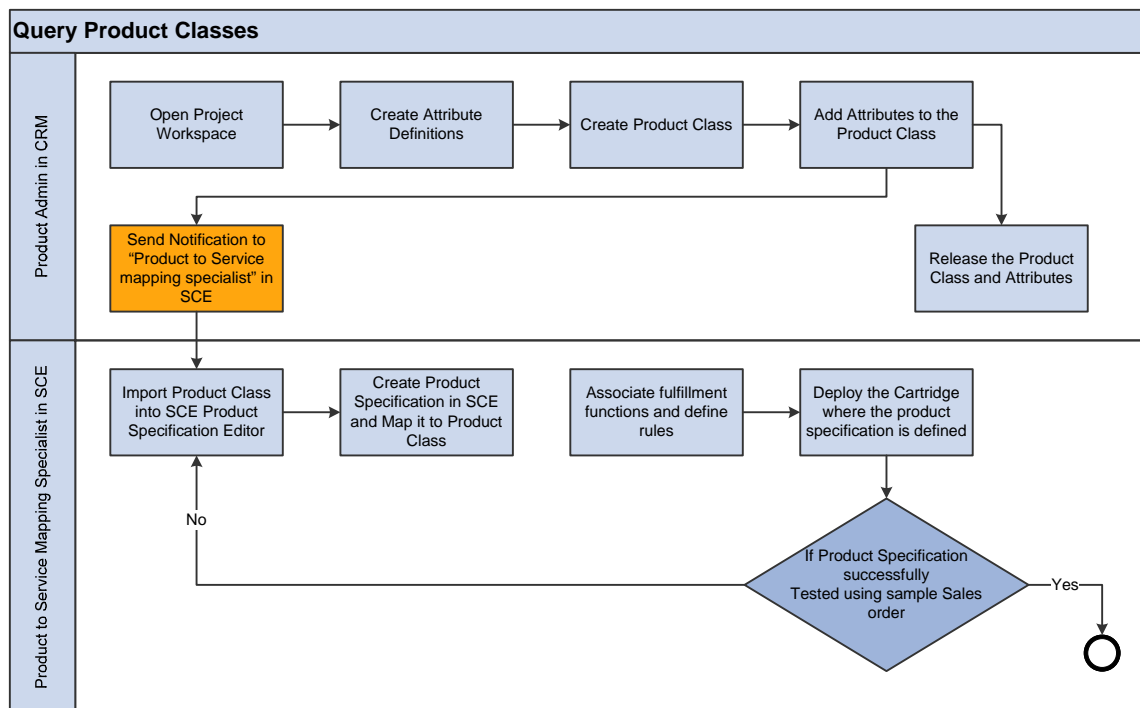
SCE Studio is an Eclipse plug-in-based application that maintains data in file format. It provides a data dictionary to store the metadata definitions, and data models to define various entities that are used by Oracle Order and Service Management (Oracle OSM). It provides a platform to define rules, functions, and dependencies that can be stored as files and deployed on a deployment server. These files are used during the order fulfillment process.

The current version of SCE Studio does not offer web services that you can invoke from an external service to create entities supported by it.

Query Product Classes and Attributes Integration Flow

Product administrators define product classes and transaction attributes in Siebel. The SCE product-to-service mapping specialist queries and imports the product class and the associated transaction attributes into a cartridge from SCE Studio. The product-to-service mapping specialist then maps the product class to a product specification in the cartridge. They use the product specification to associate decomposition rules, fulfillment functions, and their dependencies. After all the design time setup is completed, they deploy the cartridge to Oracle OSM. The cartridge defines various fulfillment topologies to process order lines during order processing.

This diagram illustrates the flow:



Business process model diagram

Query Product Class and Attributes

The product administrator performs the following operations in Workspace projects in the Siebel application:

1. Creates a new product class and transaction attributes and then associates attribute definitions to them.
2. Updates the product class by adding or deleting transaction attributes.
3. Updates the attribute definitions (valuesets) by adding or removing values from them.

For more information about creating classes, attributes, and attribute definitions in Siebel, see the *Siebel Product Administration Guide*.

Whenever these operations are performed, the Siebel product administrator notifies the product-to-service mapping specialist in SCE. The product administrator can send the notification in multiple ways:

- Use the notify menu function provided by Siebel. This function is provided in the Product Class UI in the Siebel application. The notification uses fixed templates to communicate the product class details and requires additional configuration in the Siebel application. One or more classes can be selected, and a single notification can be sent from Siebel.

For more information about setting up the notification in Siebel, see the *Siebel Quick Fix Installation Guide*, “Email Notification for the Product Class” and “Attribute Query Feature.”

- Use a company’s email application. This method provides additional flexibility to add more required information. This method is not dependent on a template and does not require additional configuration steps.

Note: Regardless of the mechanism used to send the notification, you must include the product class name. The product-to-service mapping specialist uses the product class name to query in SCE Studio. If you use the notify menu function, the template adds the product class name.

Note: For Step 3, select any class that has the transaction attribute that is associated with the valueset and send the notification.

The integration solution provides services that the product-to-service mapping specialist can use to query or import new or updated product classes from Siebel into SCE Studio.

The SCE product-to-service mapping specialist logs in to SCE Studio and queries or imports the product class and the transaction attribute details from Siebel into a cartridge. The import process uses the product class name. SCE Studio also offers the flexibility to query all the classes in the hierarchy associated with a product class. For more information about using SCE Studio, refer to the SCE documentation

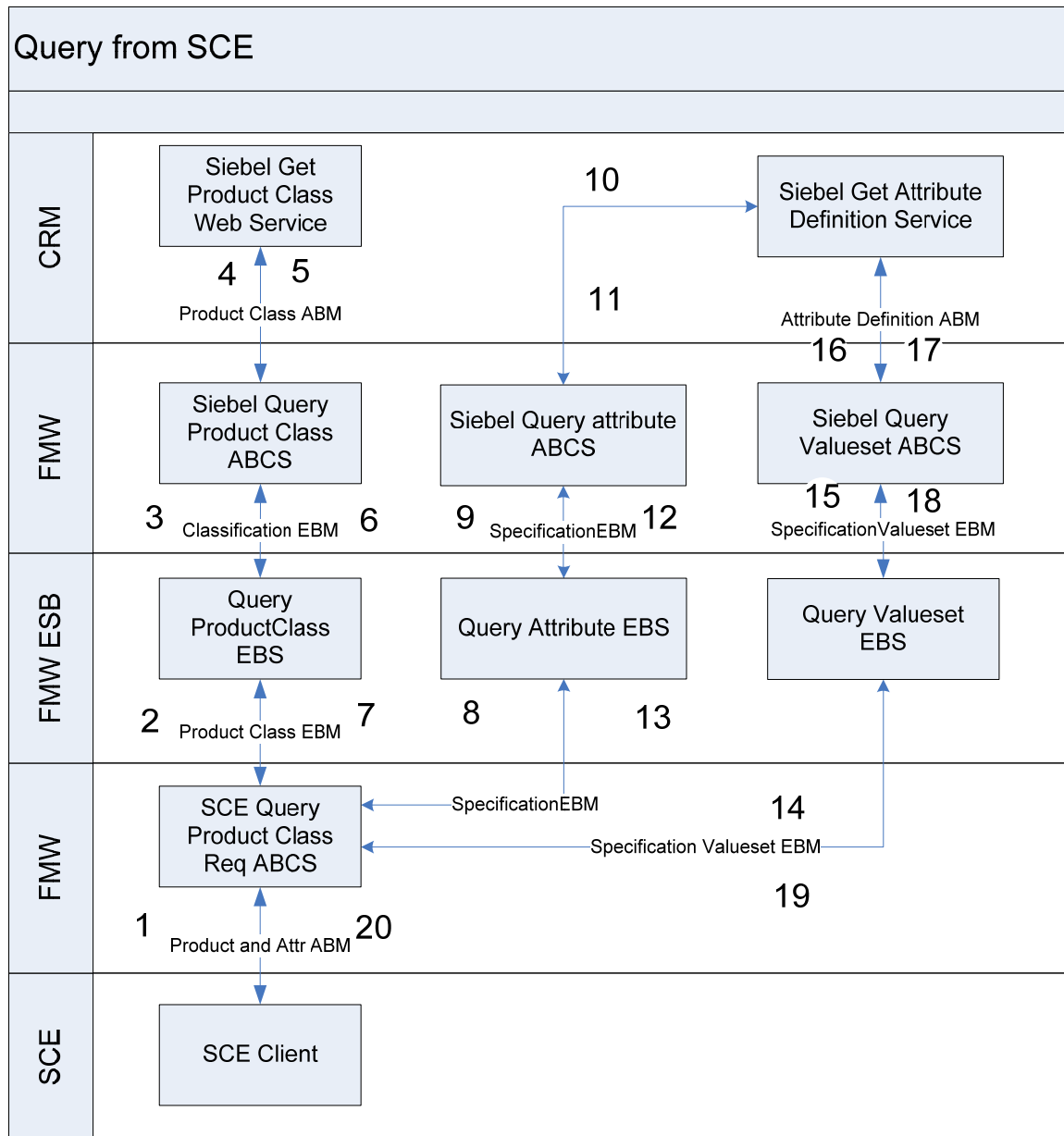
For more information about using the SCE Studio, see the design studio section in the “OSM System Components” chapter of the *Oracle Communications Order and Service Management Concepts Guide*. See also the Design Studio online help.

After you import the product classes into a cartridge, the system stores all the transaction attributes and the associated valuesets in the SCE Studio data dictionary. Map the product class to a product specification in the cartridge. You can create a new product specification or map an existing product specification to the product class.

After you map the product class and the product specification, the product-to-service mapping specialist defines and associates fulfillment functions and their dependencies to the product specification. After defining the validation and decomposition rules, they deploy the cartridge to the studio environments.

SCE Studio provides a function that tests the design-time configuration after you submit a sample test order. If issues occur with the product specification in the cartridge, then you create a new product specification or import the product class again. If you create a new product specification, you must perform the configuration again. You must redeploy the cartridge for your changes to take effect. Oracle OSM uses the deployed cartridge to process order lines that are submitted for fulfillment.

This diagram illustrates how you query the product class and attributes from SCE:



Querying product class and associated attributes from SCE

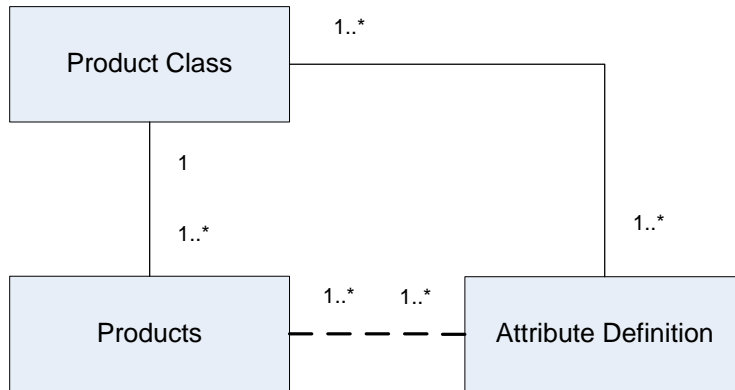
When this process initiates, the following events occur:

1. The product-to-service mapping specialist must include all the necessary product class names in the query dialogue in SCE and invoke the query operation. The SCE Client calls the SCE Query Product Class Application Business Connector Service (ABCS) and provides one or more Class details as input. SCE must be configured to provide the input in the Product Class EBO structure that has only the Product Class names.
2. The SCE Query Product Class ABCS invokes the Query Product Class enterprise business service (EBS) and provides the Product Class EBM.

3. The QueryProductClassEBS routes the Product ClassEBM to Siebel Query Product Class ABCS.
4. The Siebel Query Product Class ABCS transforms the Product ClassEBM, extracts the product class name, and invokes the Siebel getProductClass web service.
5. The Siebel getProductClass web service returns the complete product class information for all input product classes to the Siebel query product class ABCS.
6. The Siebel Query Product Class ABCS transforms the response message into the Product ClassEBM. It invokes the QueryProductClassEBS and provides the Product ClassEBM as the input.
7. The QueryProductClassEBS routes the Product ClassEBM to the SCE Query Product Class ABCS.
8. The SCE Query Product Class ABCS identifies that the ProductClass has references to one or more attribute definition entities. It has to query all the attribute definitions associated with all of the product classes. It aggregates all of the attributes that are associated with all of the product classes and makes one call to the QueryAttributeEBS. It provides all attribute IDs that are associated with all the product classes. For example, if three product classes were queried with each has five attributes, and then the QueryAttributeEBS is invoked using 15 attributes.
9. The QueryAttributeEBS routes the request to the Siebel Query Attribute ABCS.
10. The Siebel Query Attribute ABCS invokes the getAttributeDefinition web service provided by Siebel.
11. The getAttributeDefinition service returns the complete attribute information for all attribute IDs to the ABCS.
12. The Siebel Query Attribute ABCS performs the transformation, filters the necessary fields, and constructs the AttributeEBM. It invokes the QueryAttributeEBS and provides the AttributeEBM.
13. The QueryAttributeEBS routes the Attribute enterprise business message (EBM) back to the SCE Query Product Class ABCS.
14. The SCE Query Product Class ABCS identifies that the attributes have references to the valueset entity. It must query all of the valuesets associated with all the attributes. It aggregates all of the attributes that are associated with all the valuesets and makes one call to the QueryValuesetEBS. For example, if three attributes were queried, then the QueryValuesetEBS is invoked using a maximum of three valuesets.
15. The QueryValuesetEBS routes the request to the Siebel Query Valueset ABCS.
16. The Siebel Query Attribute ABCS invokes the getAttributeDefinition web service provided by Siebel.
17. The getAttributeDefinition service returns the complete attribute information to the ABCS.
18. The Siebel Query Valueset ABCS performs the transformation, filters the necessary fields, and constructs the ValuesetEBM. It invokes the QueryValuesetEBS and provides the ValuesetEBM.
19. The QueryValuesetEBS routes the ValuesetEBM back to the SCE Query Product Class ABCS.
20. The SCE Query Product Class ABCS composes the Product ClassEBM, the AttributeEBM, and the ValuesetEBM and returns it to the SCE Client application.

Logical Data Model in Siebel

This diagram illustrates the logical data model:



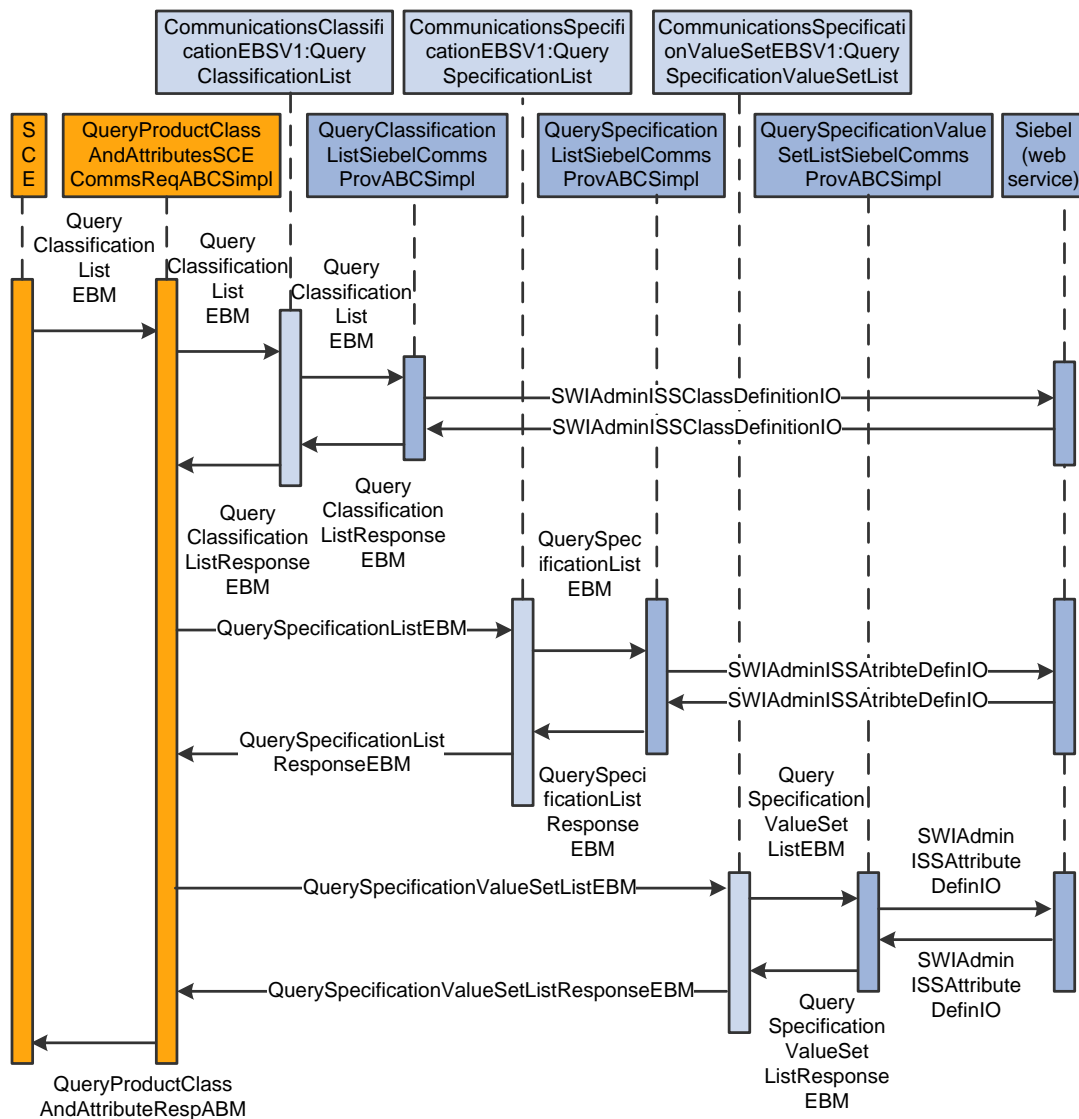
Logical data model of product class structure in Siebel

Updating the Attribute Valueset

The product administrator in Siebel can update the attribute valueset associated with the attribute definition in the project workspace. This action automatically updates all the classes and their subclasses. The product administrator selects one of the classes that is associated with the updated attribute definition and sends the notification. The product-to-service mapping specialist queries the product class from SCE Studio, which updates the corresponding valueset metadata in the data dictionary.

Querying Product Classes and Attributes Sequence

This sequence diagram illustrates the query product classes and attributes flow:



Query product classes and attributes sequence diagram

When this process initiates, the following events occur:

1. An SCE user (a product-to-service mapping specialist) triggers the Query Product Class and Attribute definitions integration flow using the SCE client, which provides the Class Code and the Oracle Fusion Middleware (FMW) URL. In the case of an update, SCE also provides the Class Codes for all of the subclasses.
2. The SCE invokes the QueryProdClassAndAttributesSCEReqCommsABCSImpl with the QueryClassificationListEBM, which contains the Product Class codes.

3. QueryProdClassAndAttributesSCEReqCommsABCImpl passes through the QueryClassificationListEBM to the CommunicationsClassificationEBSV1, which has an operation, QueryClassificationList.
4. CommunicationsClassificationEBSV1 invokes the QueryClassificationListSiebelCommsProvABCImpl with the QueryClassificationListEBM.
5. QueryClassificationListSiebelCommsProvABCImpl transforms the QueryClassificationListEBM into the Siebel Class Definition IO application business message (ABM), along with the Workspace Name from AIAConfigurationProperties.xml file and invokes the Siebel GetProductClass web service.
6. The Siebel GetProductClass web service returns the complete product class information and associated Attribute IDs for all input product Class Codes. The QueryClassificationListSiebelCommsProvABCImpl transforms the Siebel GetProductClassResponseABM into the QueryClassificationListRespEBM.
7. The QueryClassificationListRespEBM goes all the way back to the QueryProdClassAndAttributesSCEReqCommsABCImpl.
8. Next, the QueryProdClassAndAttributesSCEReqCommsABCImpl picks up the Attribute IDs from the QueryClassificationListRespEBM and maps these IDs to the QuerySpecificationListEBM and QuerySpecificationValueSetListEBM. One attribute may appear in more than one class definitions. While mapping Attribute IDs from the QueryClassificationListRespEBM to the QuerySpecificationListEBM and QuerySpecificationValueSetListEBM, we have to take the union of these attribute definitions.
9. The QueryProdClassAndAttributesSCEReqCommsABCImpl invokes the CommunicationsSpecificationEBSV1 and CommunicationsSpecificationValueSetEBSV1 with the QuerySpecificationListEBM and QuerySpecificationValueSetListEBM respectively. These two invocations happen simultaneously (in parallel).
10. The CommunicationsSpecificationEBSV1 invokes the QuerySpecificationListSiebelCommsProvABCImpl. The QuerySpecificationListSiebelCommsProvABCImpl transforms the QuerySpecificationListEBM into the Siebel Attribute definitions IO (ABM) along with the Workspace Name from the AIAConfigurationProperties.xml file and invokes the Siebel GetAttributeDefinition web service.
11. The GetAttributeDefinition service returns the complete attribute information for one or more attribute IDs. The QuerySpecificationListSiebelCommsProvABCImpl transforms the response into the QuerySpecificationListRespEBM.
12. The CommunicationsSpecificationValueSetEBSV1 invokes the QuerySpecificationValueSetListSiebelCommsProvABCImpl. The QuerySpecificationValueSetListSiebelCommsProvABCImpl transforms the QuerySpecificationValueSetListEBM into the Siebel Attribute definitions IO (ABM) along with the Workspace Name from the AIAConfigurationProperties.xml file and invokes the Siebel GetAttributeDefinition web service.
13. The GetAttributeDefinition service returns the complete attribute valueset information for one or more attribute IDs. The QuerySpecificationValueSetListSiebelCommsProvABCImpl transforms the response into the QuerySpecificationValueSetListRespEBM.
14. The QuerySpecificationListRespEBM and QuerySpecificationValueSetListRespEBM go all the way back to the QueryProdClassAndAttributesSCEReqCommsABCImpl. The QueryProdClassAndAttributesSCEReqCommsABCImpl merges the QuerySpecificationListRespEBM, QuerySpecificationValueSetListRespEBM and the QueryClassificationListRespEBM and maps them to the QueryProdClassAndAttributesRespABM.

15. The QueryProdClassAndAttributesSCEReqCommsABCImpl replies to SCE along with the QueryProdClassAndAttributesRespABM, which contains the class and attribute details for all the Class Codes provided by the product to the service mapping specialist.

Support for Effectivity During Design-Time

You can use product classes in Siebel (or the equivalent ICC in Product Hub) to represent unique product specifications. Product specifications represent a type of a product offering and can only be sold through a product offering. Product specifications represent the unique entities that need to be fulfilled. Not all product classes are created for this purpose. For the subset of product classes created to represent product specifications, at design time you must map the product class name and the corresponding fulfillment pattern name in Oracle OSM in its Central Order Management role. Consequently, changes to these product classes would affect the mapping for both design time and runtime Order Management handling. This section describes an important aspect of this mapping effectivity: how the time of a new mapping or a change to an existing mapping takes effect in coordination across Order Capture (Siebel) and Central Order Management (Oracle OSM).

References to product classes in this section are limited to those used to represent product specifications. To distinguish these, we recommend using a naming convention, such as ending the name with the term *ProdSpec*. You can create and update product classes directly in Siebel or from a Product Master, such as Oracle Product Hub for Communications. In this section, whenever you see Siebel design time product class changes, it is also applicable to product master when one is used.

Specify the effectivity for the product class in Siebel when:

- You create a new product class.
- You update an existing product class.
- You make an existing product class inactive.

Whenever these scenarios or any combination of these scenarios occurs, you must query the product classes in the Design Studio, which is the design time tool for OSM from the Siebel application. After the product classes are successfully queried, product classes and the product specification are updated manually, and one of the following conditions occurs:

- The product class is mapped to a new product specification. The new product specification definition involves defining the fulfillment metadata.
- The product class is mapped to an existing product specification.
- The product class mapping is changed to a different product specification.

Product class effectivity must be same as the product specification effectivity. You update product specification effectivity manually in the Design Studio after the product class is successfully queried and the product specification is mapped to it.

When the mapping between the product class and product specification is updated (when the effectivity of the product class and product specification changes), the cartridge version in SCE must be updated and the cartridge must be redeployed to the environment.

For more information about updating the cartridge version and the various deployment options, see the *Oracle Communications Order and Service Management Application Integration Architecture Order-to-Activate Cartridge Guide*.

Deciding on Effectivity and Cartridge Deployment

Whenever the previously mentioned scenarios or their combination occur, you must create a new version of the cartridge and redeploy it. All orders the Oracle OSM system has already submitted for processing will use the existing cartridge version. Any subsequent new orders (including revision orders, follow-on orders, and change orders) or existing orders that are not yet submitted for processing by the Oracle OSM system will use the new version of the deployed cartridge. You should group product class changes and set effectivity for a date-time that is suitable for deploying a new cartridge version, such as an off-peak hour.

Note: You deploy a new version of the cartridge only if you introduced a new mapping or changed an existing mapping.

The support for effectivity on the product class and the effectivity on the product specification is manifested in Oracle OSM by the effectivity of deployment of the cartridge. Effectivity is defined and controlled by the deployment dates for the cartridge. To manage effectivity across the applications, you must consider the following issues:

- The effective dates on the product class are the same as the deployment dates on the cartridge.

In this case, the product administrator in Siebel and the product-to-service mapping specialist in SCE must reach a consensus on the effectivity dates of the product class and the deployment dates of the cartridge, respectively.
- The effective dates on the product class differ from the deployment dates on the cartridge.

In this case, the deployment dates on the cartridge control effectivity. You can handle effectivity as follows:
 - Create a new product class or update an existing product class. Import the product class and define the mappings to the product specification in the cartridge. Deploy it based on when the changes need to be applied to the order lines in the Oracle OSM fulfillment system.
 - Create a new product class; the mappings to the product specification are not yet done in the cartridge. If the product class is subsequently updated, Siebel queries the updated version of the product class and defines the mappings in the cartridge before deploying it. In this case, multiple versions of the product class may be defined with different effective dates and queried before the mappings are defined and the cartridge is deployed. Again, the deployment is based on when the changes need to be applied to the order lines in the Oracle OSM fulfillment system.

Note: If you create a new product specification in the cartridge, you must configure the product specification before deploying the cartridge.

Solution Assumptions and Constraints

These are the assumptions and constraints:

1. The SCE queries the product classes from Siebel CRM. Any updates made to the product specification in SCE are not synchronized back to Siebel CRM.
2. SCE maintains the mapping between the Siebel product class and the production specification.
3. Only product class and the associated attributes are imported into the SCE. Structures, constraints, properties, user interface definitions, and so on, which can be associated with the product class in Siebel, are not sent to query from SCE Studio.

Siebel CRM Interfaces

The process integration for Product Lifecycle Management uses these Siebel CRM interfaces:

- SWI Product Class Hierarchy
- SWI Product Attributes

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

Industry AIA Components

The integration uses these industry components:

- ClassificationEBO
- QueryClassificationListEBM
- QueryClassificationListResponseEBM
- SpecificationEBO
- QuerySpecificationListEBM
- QuerySpecificationListResponseEBM
- SpecificationValueSetEBO
- QuerySpecificationValueSetListEBM
- QuerySpecificationValueSetListResponseEBM

The industry EBO and EBM XSD files are located here:

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

The industry 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.”

Integration Services

These services are delivered:

- QueryProdClassAndAttributesSCEReqCommsABCSImpl with operation QueryProdClassAndAttributes
- CommunicationsClassificationEBSV1 with operation QueryClassificationList
- QueryClassificationListSiebelCommsProvABCSImpl with operation QueryProductClass
- CommunicationsSpecificationEBSV1 with operation QuerySpecificationList
- QuerySpecificationListSiebelCommsProvABCSImpl with operation QuerySpecificationList
- CommunicationsSpecificationValueSetEBSV1 with operation QuerySpecificationValueSetList
- QuerySpecificationValueSetListSiebelCommsProvABCSImpl with operation QuerySpecificationValueSetList

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

For more information about using Session Pool Manager, see [Appendix B: Using Session Pool Manager](#).

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,” Viewing Integration Scenarios.

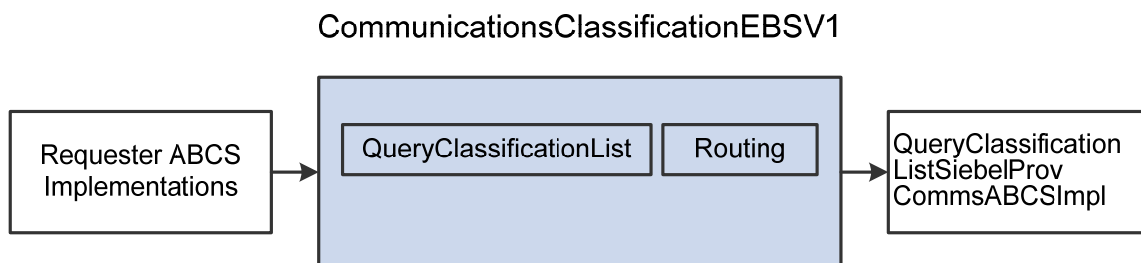
QueryProdClassAndAttributesSCEReqCommsABCImpl

The QueryProdClassAndAttributesSCEReqCommsABCImpl is a synchronous BPEL process and is the SCE requestor ABC implementation. This service follows all of the standards of a requester ABCS. This service has one operation: QueryProdClassAndAttributes. This accepts a QueryClassificationListEBM as a request and returns QueryProdClassAndAttributesRespABM as a response.

CommunicationsClassificationEBSV1

ClassificationEBS performs all of the Product Class related actions like Create Product Class, Update Product Class, Query Product Class, and so on. This service has one operation: QueryClassificationList. Based on the routing rules setup, it invokes a provider ABCS.

This diagram illustrates where the ClassificationEBS service lies in relation to the other services in the overall integration flow:

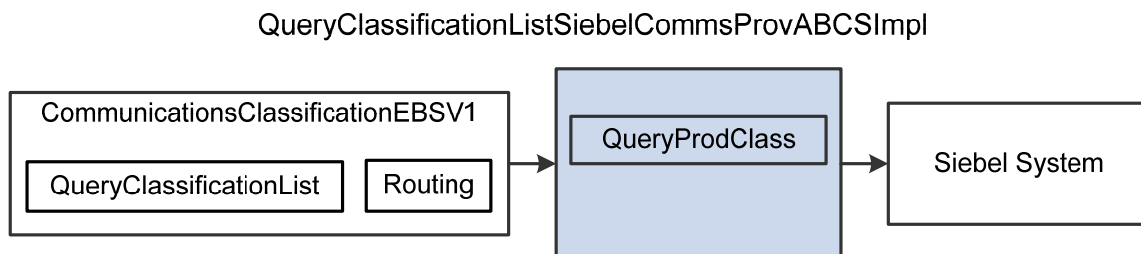


For more information about this EBS, see *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Developing EBSs" and *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding EBSs."

QueryClassificationListSiebelCommsProvABCImpl

This is the Siebel Classification List Provider ABC Implementation. This service follows all the standards of a Provider ABCS implementation. This service has one operation: QueryProductClass.

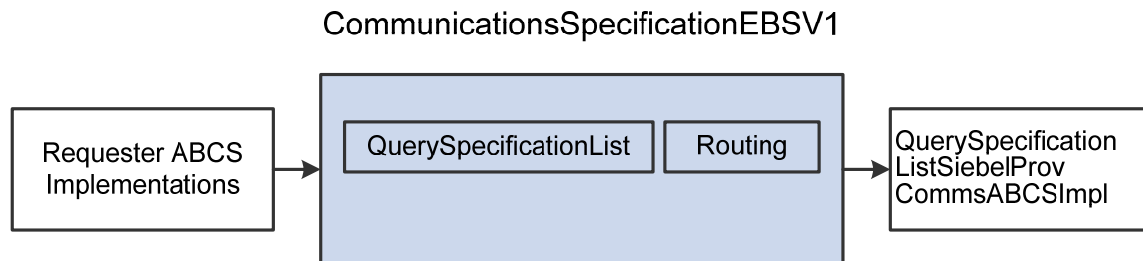
This diagram illustrates where the QueryClassificationListSiebelCommsProvABCImpl service lies in relation to the other services in the overall integration flow:



CommunicationsSpecificationEBSV1

The SpecificationEBS performs all of the Specification List related actions like Query Specification List, Create Specification List, Update Specification List, and so on. This service has one operation: QuerySpecificationList. Based on the routing rules setup, it invokes a provider ABCS.

This diagram illustrates where the SpecificationEBS service lies in relation to the other services in the overall integration flow:

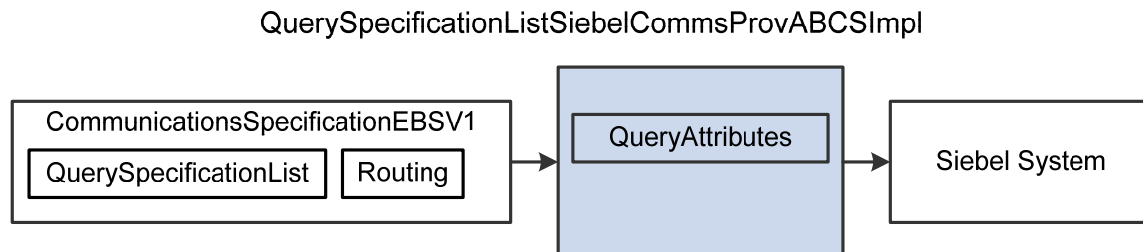


For more information about this EBS, see the *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Developing EBSs" and the *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding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

QuerySpecificationListSiebelCommsProvABCImpl

The QuerySpecificationListSiebelCommsProvABCImpl is the Siebel attribute provider ABC implementation. This service follows all the standards of a provider ABCS implementation. This service has one operation: QuerySpecificationList.

This diagram illustrates where the QuerySpecificationListSiebelCommsProvABCImpl service lies in relation to the other services in the overall integration flow:

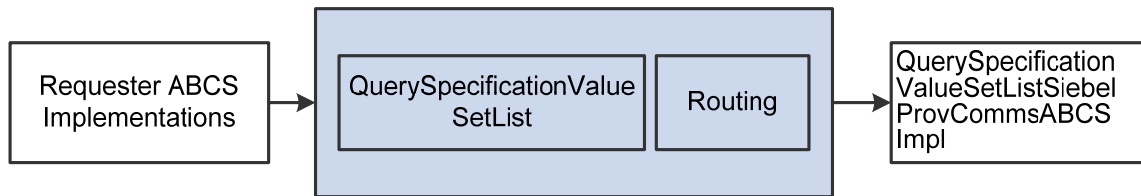


CommunicationsSpecificationValueSetEBSV1

The CommunicationsSpecificationValueSetEBSV1 performs all of the SpecificationValueSet List related actions like Query SpecificationValueSet List, Create SpecificationValueSet List, Update SpecificationValueSet List, and so on. This service has one operation: QuerySpecificationValueSetList. Based on the routing rules setup, it invokes a provider ABCS.

This diagram illustrates where the SpecificationValueSetEBS service lies in relation to the other services in the overall integration flow:

CommunicationsSpecificationValueSetEBSV1



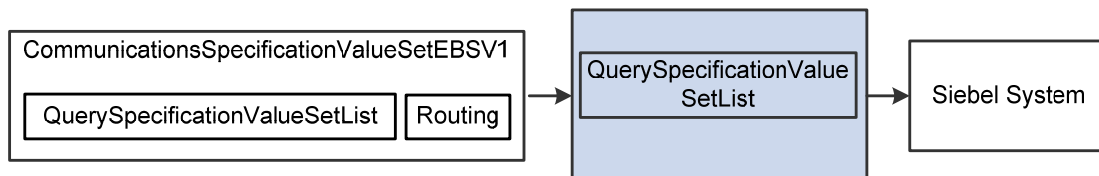
For more information about this EBS, see the *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Developing EBSs" and the *Oracle Application Integration Architecture - Foundation Pack 2.5: Concepts and Technologies Guide*, "Understanding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

QuerySpecificationValueSetListSiebelCommsProvABCSEBSV1

The QuerySpecificationValueSetListSiebelCommsProvABCSEBSV1 is the Siebel attribute value set provider ABC implementation. This service follows all the standards of a provider ABCS implementation. This service has one operation: QuerySpecificationValueSetList.

This diagram illustrates where the QuerySpecificationValueSetListSiebelCommsProvABCSEBSV1 service lies in relation to the other services in the overall integration flow:

QuerySpecificationValueSetListSiebelCommsProvABCSEBSV1



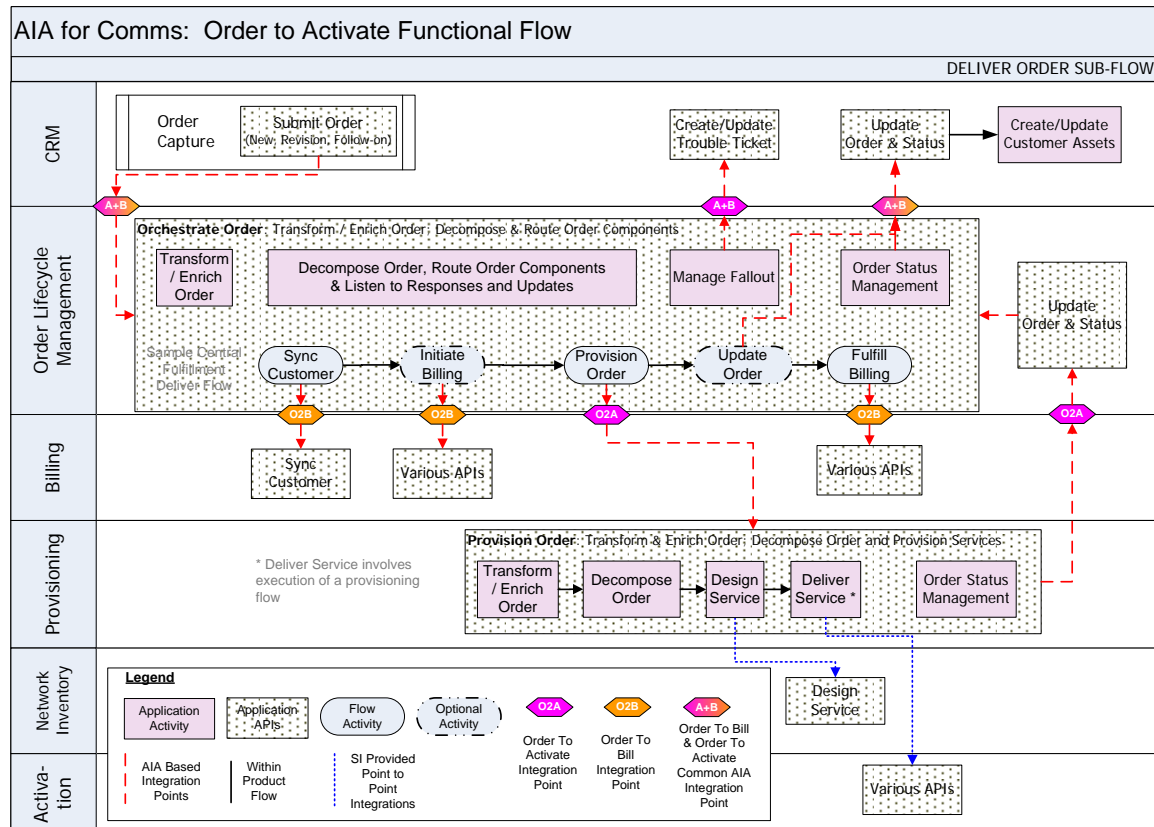
Chapter 3: Understanding the Process Integration for Order Lifecycle Management

This chapter provides an overview of the Order Lifecycle Management (OLM) process integration and discusses:

- Support of order priorities.
- Siebel Customer Relationship Management (Siebel CRM) interfaces.
- Oracle Order and Service Management (Oracle OSM) Java Message Service (JMS) message construction.
- Industry application integration architecture (AIA) components.

OLM Process Integration Overview

Order Management is at the center of an Oracle Order to Activate deployment, with one or more order capture systems passing orders to the Order Management system. The Order Management system decomposes the order into suborders, each of which targets a particular fulfillment provider (that is, a system instance). Each order, whether it is new, revised, a follow-on, or future-dated, must go through these flows for fulfillment or provisioning.



OLM flows

Order Flow

When a new order process is initiated, the following events occur:

1. In Siebel CRM, a user navigates to the Sales Order screen and clicks Submit to submit a new order. This action triggers Siebel to create a Siebel application business message (ABM) with all the captured details. Siebel then enqueues this ABM in the AIA_SALESORDERJMSQUEUE.
2. After the Siebel ABM is enqueued in the AIA_SALESORDERJMSQUEUE, the ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer dequeues the message and passes it to ProcessSalesOrderFulfillmentSiebelCommsReqABCImpl. The ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer has a JMS adapter service that polls for any messages in the AIA_SALESORDERJMSQUEUE.

Note: ProcessSalesOrderFulfillmentSiebelCommsJMConsumer also has an embedded sequence, which sequences the messages. This means, if the ProcessSalesOrderFulfillmentSiebelCommsReqABCImpl or the ProcessSalesOrderFulfillmentOSMCFSCCommsJMSPProducer fails, all subsequent new orders for that account are stuck in the Sequencer table. If the error is a system error the administrator must resubmit the order. If the error is a business error then the administrator must correct the error and then release old messages (orders) from the sequencer table for that account so that new orders don't continue waiting in the Sequencer table.

For more information about sequencing, see [Using the ESB Resequencer Feature](#).

3. ProcessSalesOrderFulfillmentSiebelCommsReqABCImpl invokes the transformation logic to transform Siebel ABM to the ProcessSalesOrderFulfillmentEBM, and it invokes the CommunicationsSalesOrderEBSV2 with the operation ProcessSalesOrderFulfillment.
4. CommunicationsSalesOrderEBSV2 routes this message to the ProcessSalesOrderFulfillmentOSMCFSCCommsJMSPProducer interface, which wraps ProcessSalesOrderFulfillmentEBM into the CreateOrder message format and enqueues the message into the AIA_CRTFO_IN_JMSQ.

The CreateOrder message is a web service Oracle OSM and needs a SOAP message to be passed.

For more information about the CreateOrder message structure, see the *Oracle Communications Order and Service Management, Application Integration Architecture Order to Activate Cartridge Guide*.

5. After the message is in the Oracle AIA queue, the Weblogic JMS Bridge picks up the message and moves it to the WebLogic queue
oracle/communications/ordermanagement/WebServiceQueue.

For more information about the services described here, see [Chapter 5: Understanding the Process Integrations for Siebel CRM](#).

6. This internal Oracle OSM activity occurs:

Oracle OSM picks up the CreateOrder message and then further decomposes the order for fulfillment and provisioning.

Siebel CRM must include all lines in the new order. When a new order is submitted to Oracle OSM, it checks the value of the point-of-no-return (PONR) on the base order for all order items (in the case of a revision). If even one corresponding order item on the base order has a PONR that equals Hard, then the whole order (in the case of a revision) is rejected with an order item status updated to Revision Rejected and an order item Status Context equal to "base order item already past the hard point-of-no-return".

Oracle OSM needs the following Sales Order attributes to perform the task:

- Order Header / Order Number

- Order Header / Order ID
- Order Header / Revision
- Order Line / Line Item Number
- Order Line / Line Item ID
- Order Line / Revised Line Item ID
- Order Line / Asset Integration ID

Oracle OSM checks the nature of the order (new or revision) and the Action Codes coming for each of the Order Line, and it passes the order to provisioning and billing.

7. Oracle OSM CFS syncs to the Customer (Billing) flow:

Oracle OSM dequeues the CreateOrder message and processes it.

For any customer-related information, for example to create a customer or account in Billing, Oracle OSM creates a message `ProcessFulfillmentOrderBillingAccountListEBM` and enqueues the same in the Oracle OSM CFS Outbound queue `oracle/communications/ordermanagement/WebServiceCreateCustomerQueue`. The WebLogic JMS Bridge then moves the message to the `AIA_CRTCUST_OUT_JMSQ`.

8. The `ProcessFulfillmentOrderBillingAccountListOSMCFSCommsJMSConsumer` monitors this queue and will consume any message in this queue and route it to `CommunicationsBillingEBSV1` by invoking `ProcessFulfillmentOrderBillingAccountList` operation.

The `CommunicationsBillingEBSV1` then routes this message to the custom process billing account list enterprise business flow to create the customer.

9. After the customer is created, “Custom Process Billing Account List Enterprise Business Flow” needs to send a response to the `CommunicationsBillingResponseEBSV1` as `ProcessFulfillmentOrderBillingAccountListResponseEBM`.

`CommunicationsBillingResponseEBSV1` then routes this message to the `ProcessFOBillingAccountListRespOSMCFSCommsJMSProducer` process, which enqueues the message into `AIA_UPDCUST_IN_JMSQ`.

The WebLogic JMS Bridge then picks up the message and puts it into the WebLogic queue `oracle/communications/ordermanagement/WebServiceCreateCustomerResponseQueue` (Oracle OSM Inbound queue for Oracle OSM CFS) for updating Oracle OSM.

For more information about the services described here, see [Chapter 4: Understanding the Process Integrations for Billing](#).

Steps 10 through 14 describe the Oracle OSM-Billing Flow:

10. Oracle OSM CFS enqueues the `ProcessFulfillmentOrderBillingEBM` message in the OSM CFS Outbound Queue `oracle/communications/ordermanagement/WebServiceCreateBillingOrderQueue`. The Weblogic JMS Bridge moves this message into the AIA Queue `AIA_CRTBO_OUT_JMSQ`.

11. The ProcessFulfillmentOrderBillingOSMCFSCommsJMSConsumer dequeues ProcessFulfillmentOrderBillingEBM and passes it to the CommunicationsBillingEBSV1 with the operation: ProcessFulfillmentOrderBilling.
12. The CommunicationsBillingEBSV1 routes the ProcessFulfillmentOrderBillingEBM to “Custom Billing ABCS”.
13. Custom Billing ABCS passes the ProcessFulfillmentOrderBillingResponseEBM to the CommunicationsBillingResponseEBSV1 with the operation: ProcessFulfillmentOrderBillingResponse.

The CommunicationsBillingResponseEBSV1 then routes the ProcessFulfillmentOrderBillingResponseEBM to the ProcessFulfillmentOrderBillingResponseOSMCFSCommsJMSProducer.
14. The ProcessFulfillmentOrderBillingResponseOSMCFSCommsJMSProducer enqueues this to the AIA_UPDBO_IN_JMSQ. The WebLogic JMS Bridge then moves the message to the WebLogic queue oracle/communications/ordermanagement/WebServiceCreateBillingOrderResponseQueue (Oracle OSM CFS Inbound queue) for updating Oracle OSM CFS about the response received.

For more information about the services described here, see [Chapter 4: Understanding the Process Integrations for Billing](#).

Steps 15 through 21 describe the Oracle OSM Provisioning flow.

15. For order components that need to go to provisioning, the Oracle OSM CFS creates a ProcessProvisioningOrderEBM message.

Oracle OSM then enqueues the message ProcessProvisioningOrderEBM into the Oracle OSM CFS outbound queue: oracle/communications/ordermanagement/WebServiceCreateProvisioningOrderQueue. The WebLogic JMS Bridge then moves this message into the AIA_CRTFO_OUT_JMSQ.

ProcessProvisioningOrderOSMCFSCommsJMSConsumer monitors the AIA_CRTFO_OUT_JMSQ queue and dequeues the message from the Oracle AIA queue using a JMS Adapter service whenever a message exists.

The message is then routed to its respective enterprise business service (EBS) as follows: ProcessProvisioningOrderOSMCFSCommsJMSConsumer routes ProcessProvisioningOrderEBM to CommunicationsProvisioningOrderEBSV1 by calling the ProcessProvisioningOrder operation.
16. The CommunicationsProvisioningOrderEBSV1 then routes the message to the appropriate producers: for the ProcessProvisioningOrder operation, the EBS routes the message to the ProcessProvisioningOrderOSMPROVJMSProducer.
17. The previous step produces the message and sends it into the AIA_FOPROV_IN_JMSQ. The WebLogic JMS Bridge then moves the message into the WebLogic JMS queue oracle/communications/ordermanagement/WebServiceQueue of Oracle OSM Provisioning instance (the Oracle OSM Provisioning Inbound queue). Oracle OSM Provisioning then picks up the message from this queue and processes it accordingly.

18. After provisioning is done, an update message is enqueued by Oracle OSM Provisioning in the WebLogic queue
oracle/communications/ordermanagement/WebServiceUpdateFulfillmentOrderQueue (the Oracle OSM Provisioning Outbound queue).

The WebLogic JMS Bridge then moves the message back to the
AIA_FOPROV_OUT_JMSQ.

The ProcessFulfillmentOrderUpdateOSMPROVCommsJMSConsumer dequeues the
ProcessFulfillmentOrderUpdateEBM message and passes it on to the
CommunicationsFulfillmentOrderEBSV1 with the operation ProcessFulfillmentOrderUpdate.

19. The CommunicationsFulfillmentOrderEBSV1 routes the message to the
ProcessFulfillmentOrderUpdateOSMCFSCommsJMSProducer. The producer then enqueues
the ProcessFulfillmentOrderUpdateEBM to AIA_FOCFS_IN_JMSQ. The WebLogic JMS
Bridge moves this message into the WebLogic queue
oracle/communications/ordermanagement/WebServiceCreateProvisioningOrderResponseQueue (the Oracle OSM CFS Inbound queue). Oracle OSM picks up this message to update the
status of the order.

20. For updating Siebel on the order status, Oracle OSM CFS creates a message
UpdateSalesOrderEBM and enqueues the message to
oracle/communications/ordermanagement/WebServiceUpdateSalesOrderQueue. The
WebLogic JMS Bridge moves the message to AIA_UPDSO_OUT_JMSQ.
UpdateSalesOrderOSMCFSCommsJMSConsumer consumes this message and calls the
UpdateSalesOrder operation in the CommunicationsSalesOrderEBSV2.

21. The CommunicationsSalesOrderEBSV2 then routes the message to the
UpdateSalesOrderSiebelCommsProvABCImpl. This process converts the EBM into the
Siebel ABM and invokes the Siebel web service to update the order.

For more information about the services described here, see [Chapter 6: Understanding the Process Integration for Order Provisioning](#).

New Orders Technical Considerations

New orders include first time purchases as well as changes to existing (already asseted) service subscriptions and products. Siebel Order Capture captures new orders and, after a customer revises an order (not necessarily the first revision), submits it to Oracle OSM Fulfillment to deliver on the promises made to the customer.

Sales orders are primarily composed of two key parts: the order header and the order line. The order header includes attributes applicable to the customer and to all order lines. Order lines are composed of an action and a subject.

Order lines can include any combination of order line actions supported in Siebel CRM. Possible order line actions are:

- Add
- Delete
- Update
- Suspend

- Resume
- Move-Delete
- Move-Add
- Existing (no change is required)

Order lines can include a variety of subjects, including but not limited to products, discounts (modeled as products), bundles, offers, and pricing event products (used with multi-event billing products).

The key function of the Oracle AIA integration is to pass enough order header and order line attributes to facilitate order fulfillment and to establish the necessary cross-references.

Revision Orders Technical Considerations

Revision orders are changes made to a previously revised order. Siebel allows users to revise an order line as long as the order line did not reach the PONR or complete. A PONR is configured on the fulfillment flow of each product specification in Oracle OSM and is propagated to Siebel CRM to indicate that an order line cannot be revised beyond that point in time. Not all revisions are submitted to fulfillment. Note that only submitted revisions factor into fulfillment. After a revision is submitted, Oracle OSM Fulfillment Order Change Management (Oracle OSM OCM) takes three actions:

1. Suspends the fulfillment flows associated with the revised order.
2. Computes the delta changes for each order line.
3. Leverages the metadata configured for the flow to devise a compensation plan for fulfillment activities that already occurred and that are affected by the revision. The compensation plan is woven into the fulfillment plan for the revision order, and the revision fulfillment does not begin until completion or another revision is submitted.

In Siebel CRM, for the sales order that is to be revised, a CSR navigates to the Sales Order screen, revises a base order, makes the required changes, and then submits the revision. This triggers Siebel to create a Siebel ABM that contains all of the details captured for that order and enqueues the ABM in the AIA_SALESORDERJMSQUEUE. Thereafter, the process within Oracle AIA is no different from that for new orders. The remaining steps are the same as Steps 7 through 21 in the [new order scenario](#).

Follow-On Orders Technical Considerations

The fulfillment of certain services may take days or weeks, and some B2B and infrastructure projects may take months to complete. During this period, customers change their minds and request changes to their orders that translate into revision orders in Siebel if the subject order lines did not reach the hard PONR or become follow-on orders.

In many cases, not taking the order pending the completion of in-flight orders is not acceptable, therefore, Siebel simulates the future state of in-flight orders and allows users to create and submit follow-on orders that are nothing more than change orders based on the projected future state of customer assets.

Follow-on orders are change orders that involve a dependency on the future fulfillment of at least one other order line in an order that is currently in-flight. The follow-on order line may change another in-flight order line that is beyond the hard PONR or simply depend on the future asset state of that item as a kind of a dependency established in Siebel.

Follow-on orders are created and submitted to Oracle OSM immediately, and Oracle OSM is responsible for managing the fulfillment dependency between the follow-on order and other base orders, much like its responsibility for determining the proper processing time for future-dated orders.

In Siebel CRM, a CSR navigates to the Sales Order screen (for the sales order that is supposed to undergo follow-on), and creates and submits the follow-on order. This action triggers the Siebel ABM in the AIA_SALESORDERJMSQUEUE to be enqueued.

After the follow-on order start-fulfillment dependencies are resolved, the follow-on order becomes like any other change order. It is also subject to revisions and other follow-on orders.

Different scenarios exist for when the order status needs to be sent back to Siebel CRM to update the CSR and the customer:

- Whenever the order fulfillment status changes, an update message is created by Oracle OSM CFS and sent to Siebel CRM.
- Whenever Oracle OSM provisioning updates the status of an order, Oracle OSM provisioning creates an update message to update Siebel CRM using Oracle OSM CFS.
- Whenever an account is synced to billing.
- Whenever billing is initiated or fulfilled.

Future-Dated Orders Technical Considerations

Future-dated orders readies order capture to accommodate customer requirements for activation or deactivation of services while on vacation and other similar conditions. Support for future-dated orders ranges from a necessity to a convenience for a CSP.

A variety of reasons require a CSP to take or place an order with a future-requested delivery date. Future-dated orders are submitted immediately to Oracle OSM when they are ready. Oracle OSM is responsible for computing the fulfillment start date-time.

When a CSR receives a request from the customer to submit an order on a future date, he sets the Due Date attribute to the specified date and submits the order. This action triggers Siebel to create a Siebel ABM with all the captured details and enqueues the ABM in the AIA_SALESORDERJMSQUEUE. The remaining steps are the same as in the [new order scenario](#).

For more information about handling current, past, future, and requested but not provided delivery date-time values, see the *Oracle Communications Order and Service Management, Application Integration Architecture Order to Activate Cartridge Guide*.

Updating Order Status Technical Considerations

The Oracle OSM orchestration engine centrally tracks fulfillment order status using a number of built-in and configured status maps and updates. Customers and CSRs are concerned about where submitted orders are in relation to their fulfillment journey and their current status. Several attributes are used to track different status characteristics in Siebel CRM.

The Oracle OSM Cartridge for Oracle AIA seeds order life cycle-related status mappings and trigger points that compute the fulfillment status fields in central fulfillment and, for a subset of these, trigger updates into Siebel CRM.

For more information about status attributes and the Oracle OSM Cartridge for Oracle AIA, see the *Oracle Communications Order and Service Management, Application Integration Architecture Order to Activate Cartridge Guide*.

Oracle OSM receives status updates and fulfillment request responses from participating fulfillment systems, maps the responses to a common status, and aggregates the status at the order line as well as at the order header level. Status changes deemed significant are propagated to Siebel using Oracle AIA.

For more information about specific status management, see the Oracle Order and Service Management product documentation.

Supporting Order Priorities

Order fulfillment priority is specified in Siebel CRM and honored by message queues, Oracle AIA, Oracle OSM, and fulfillment systems unless data integrity dictates a deferent processing sequence, such as in the case of update sales orders from Oracle OSM to Siebel CRM.

Order priority affects the sequence in which orders are picked up from queues and processed in Oracle AIA, Oracle OSM, and fulfillment systems. Orders with a higher priority take precedence over orders with a lower priority that have not yet started fulfillment.

Order priorities work as follows:

1. The submission process for orders is the same for new orders, revision orders, and follow-on orders. The CSR selects a priority for the order when they submit it.
2. As delivered, Siebel provides and maps these priority values:

Order Priority	JMS Priority
Low	3
Medium	5
High	7
Urgent	9

Customers can add other values in Siebel. Additionally, customers can use the SWI_ORDER_JMS_PRIORITY mapping, which maps these string values to integers.

You are required to set up some JMS compatibility properties on the Siebel queue and to make manual changes to seeded priority values.

For more information about priority values, see the *Siebel Order Management Guide Addendum for Communications, Employee Asset-Based Ordering*, “Modifying the Order Priority Mapping.”

The integration supports 10 priority values, 0-9, as dictated by JMS queueing technology. Implementers can extend Siebel to support priority values other than the four that are supported when delivered.

For more information, see the Siebel product documentation for Lists of Values.

These steps describe how the integration handles order priorities:

1. When the order is submitted, Siebel sets the JMS Priority message in the JMS headers and also populates the following field: ListOfSWIOrderIO/SWIOrder/OrderPriority.
2. Based on the priority, the ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer consumes the message and routes it to the ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl. This process does a lookup on the SALESORDER_PRIORITY domain value map (DVM) and populates the ProcessSalesOrderFulfillmentEBM/DataArea/ProcessSalesOrderFulfillment/FulfillmentPriority Code.
3. The ProcessSalesOrderFulfillmentOSMCFSCommsJMSProducer looks for the FulfillmentPriorityCode and does a lookup on SALESORDER_PRIORITY DVM for the JMS column and populates the priority in the JMS Headers in JMSPriority field.
4. The WebLogic JMS Bridge honors the JMS priority and picks up the message with high priority first and passes it to Oracle OSM CFS.
5. Oracle OSM CFS and Oracle OSM Provisioning honor the priority through internal mechanisms. Higher priority orders are fulfilled and provisioned first, followed by lower priority orders.
6. Oracle OSM is expected to maintain the priority of the orders and needs to populate the FulfillmentPriorityCode element in all outbound messages.
7. The population of JMS priority from the FulfillmentPriorityCode is done using the BPEL assign activity as follows:

Look up the DVM column for priority value and copy it to the JMSPriority field in JMS headers:

```
<assign>
  <copy>
    <from expression="orcl:lookup-
dvm( 'SALESORDER_PRIORITY', 'COMMON', bpws:getVariableData('priority_va
lue'), 'JMS', null)"/>
    <to variable="msg_priority"/>
  </copy>
  <copy>
    <from variable="msg_priority"/>
    <to variable="jmsHeaders" part="outboundHeader"
```

```

query="/ns5:JMSOutboundHeadersAndProperties/ns5:JMSOutboundHeaders/n
s5:JMSPriority"/>
    </copy>
</assign>

```

- jmsHeaders is a variable which is of type JMSOutboundHeadersAndProperties.
- msg_priority is a string variable.

The ProcessSalesOrderFulfillmentOSMCFSCCommsJMSProducer follows this logic to populate the JMSPriority.

8. Other producers need not do a DVM lookup since the priority value is an integer and is directly populated in the EBMs in the FulfillmentPriorityCode. The integration can use this value to populate the JMS priority. Oracle OSM and Oracle AIA, unlike Siebel, follow the same values for JMS priorities.

Oracle OSM JMS Message Construction

Oracle OSM exposes its application programming interfaces (APIs) as web services, and the invocation of these web services is done using JMS protocol and HTTP protocol. Since HTTP does not offer guaranteed delivery, we use JMS in these flows. For this, a JMS payload is created as in the following diagram and then the messages are enqueued in Oracle AIA queues. The WebLogic JMS Bridge picks up the message from Oracle AIA Queue and moves it to the respective WebLogic queues. Oracle OSM picks up the message from the WebLogic JMS queues and processes it further. Similarly, the Oracle OSM enqueues the message in a WebLogic queue, which is moved by the bridge to Oracle AIA queues. Oracle AIA consumers pick up this message and pass it to the next artifact for further processing.

This diagram illustrates the payload that is enqueued in the Oracle AIA (and then to WebLogic) queues:



Payload that is sent to Oracle OSM

The CreateOrder element is picked up from
AIAComponents/ApplicationObjectLibrary/OSM/V1/OrderManagementWS.xsd.

Oracle OSM also expects the JMS messages with the following additional properties:

- URI -- /osm/wsapi
- _wls_mimehdrContent_Type -- text/xml; charset=utf-8

These two properties are set at the JMS Header level using the BPEL Assign activity. You use the following example to assign these values:

```
<assign name="Assign_JMSHeaders">
  <copy>
    <from>
      <JMSOutboundHeadersAndProperties
xmlns="http://xmlns.oracle.com/pcbpel/adapter/jms/">
        <JMSOutboundHeaders>
          <JMSDestinationName/>
          <JMSDestinationProperties/>
          <JMSCorrelationID/>
          <JMSType/>
          <JMSReplyTo/>
        </JMSOutboundHeaders>
      </JMSOutboundHeadersAndProperties>
    </from>
  </copy>
</assign>
```

```

        <JMSPriority/>
        <JMSEExpiration/>
    </JMSOutboundHeaders>
    <JMSOutboundProperties
xmlns:ns4="http://xmlns.oracle.com/pcbpel/adapter/jms/">
        <ns4:Property
name="_wls_mimehdrContent_Type"
                                type="String"
                                value="text/xml;"
charset=utf-8"/>
        <ns4:Property name="URI" type="String"
                                value="/osm/wsapi"/>
    </JMSOutboundProperties>
</JMSOutboundHeadersAndProperties>
</from>
<to variable="jmsHeaders" part="outboundHeader"
    query="/ns5:JMSOutboundHeadersAndProperties"/>
</copy>
</assign>

```

Oracle OSM web service calls expect the web service authentication (username and password) details in the SOAP headers. For this, use an additional transformation. The username and password are configured in the `AIAConfigurationProperties.xml` as follows:

```

<ModuleConfiguration moduleName="OSM_Module">
    <Property name="OSMCFS.USERNAME">osm</Property>
    <Property name="OSMCFS.PASSWORD">osmadmin</Property>
    <Property name="OSMLF.USERNAME">osmlf</Property>
    <Property name="OSMLF.PASSWORD">osmlfadmin</Property>
</ModuleConfiguration>

```

Customers can change the username and password in the `AIAConfigurationProperties.xml` and reload that file to reflect the changes.

For more information about bridges in Oracle OSM, see the *Oracle Communications Order and Service Management, System Administration Guide*.

For more information about JMS bridges concepts and how to configure JMS bridges, see the WebLogic documentation.

Industry AIA Components

The process integration uses these industry components:

- SalesOrderEBO
 - ProcessSalesOrderFulfillmentEBM
 - UpdateSalesOrderEBM
- ProvisioningOrderEBO

- ProcessProvisioningOrderEBM
- FulfillmentOrderEBO
 - ProcessFulfillmentOrderBillingEBM
 - ProcessFulfillmentOrderBillingReponseEBM
 - ProcessFulfillmentOrderBillingAccountListEBM
 - ProcessFulfillmentOrderBillingAccountListResponseEBM
 - ProcessFulfillmentOrderUpdateEBM

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

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

The industry 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.”

Chapter 4: Understanding the Process Integrations for Billing

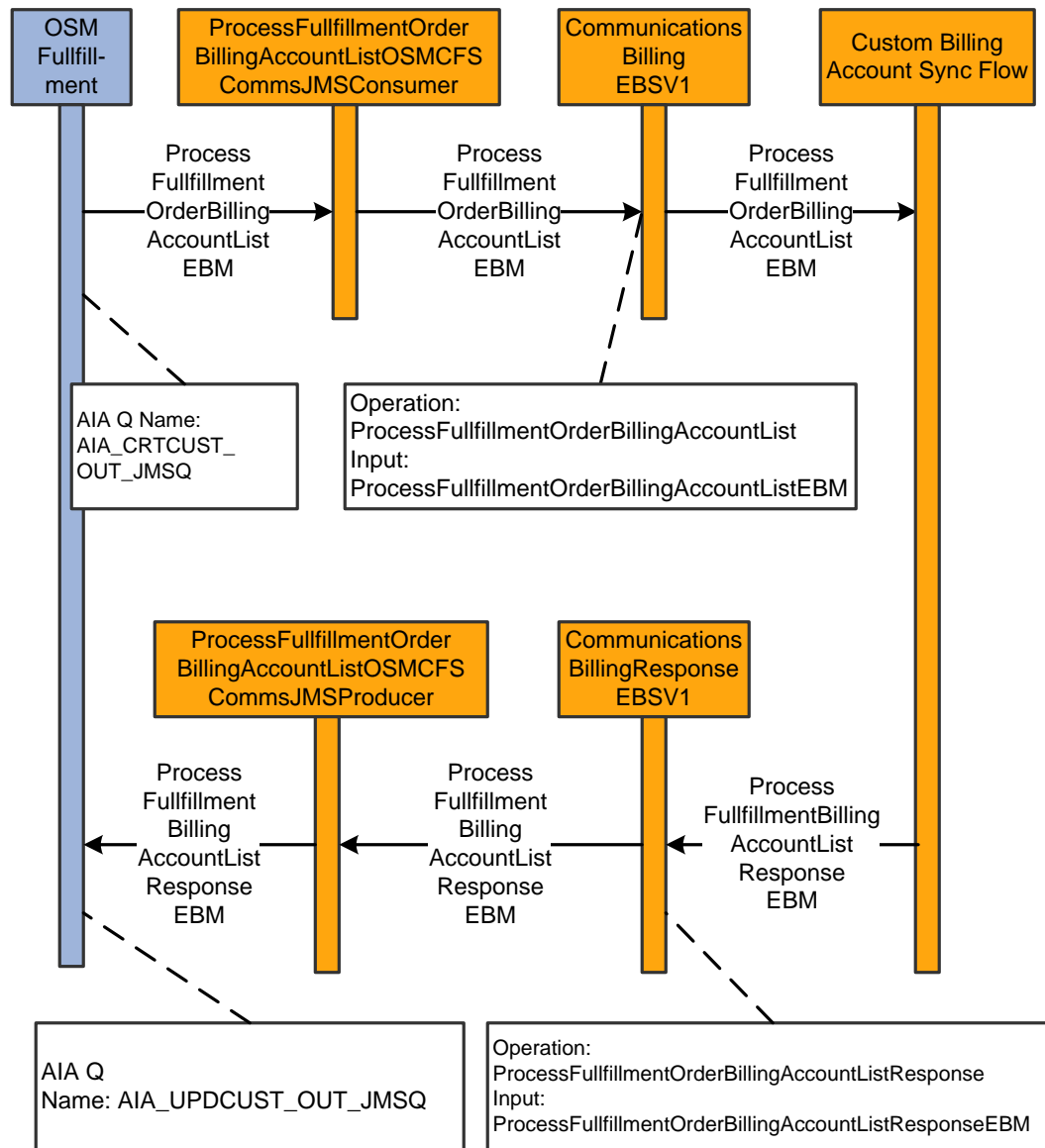
This chapter discusses:

- Billing integrations used by Oracle Order and Service Management (Oracle OSM) to:
 - Synchronize customer accounts on a customer order to one or more billing systems.
 - Interface customer order actions to one or more billing systems.
- Integration services.

Synchronizing Customer Accounts to Billing

Customer order fulfillment involves interfacing order customer accounts and actions to one or more billing system instances (also known as billing providers).

This diagram illustrates the Oracle AIA integration components used by Oracle OSM to synchronize customer accounts on an order to a billing provider:



Oracle OSM synchronizing customer accounts

When this process is initiated, the following events occur:

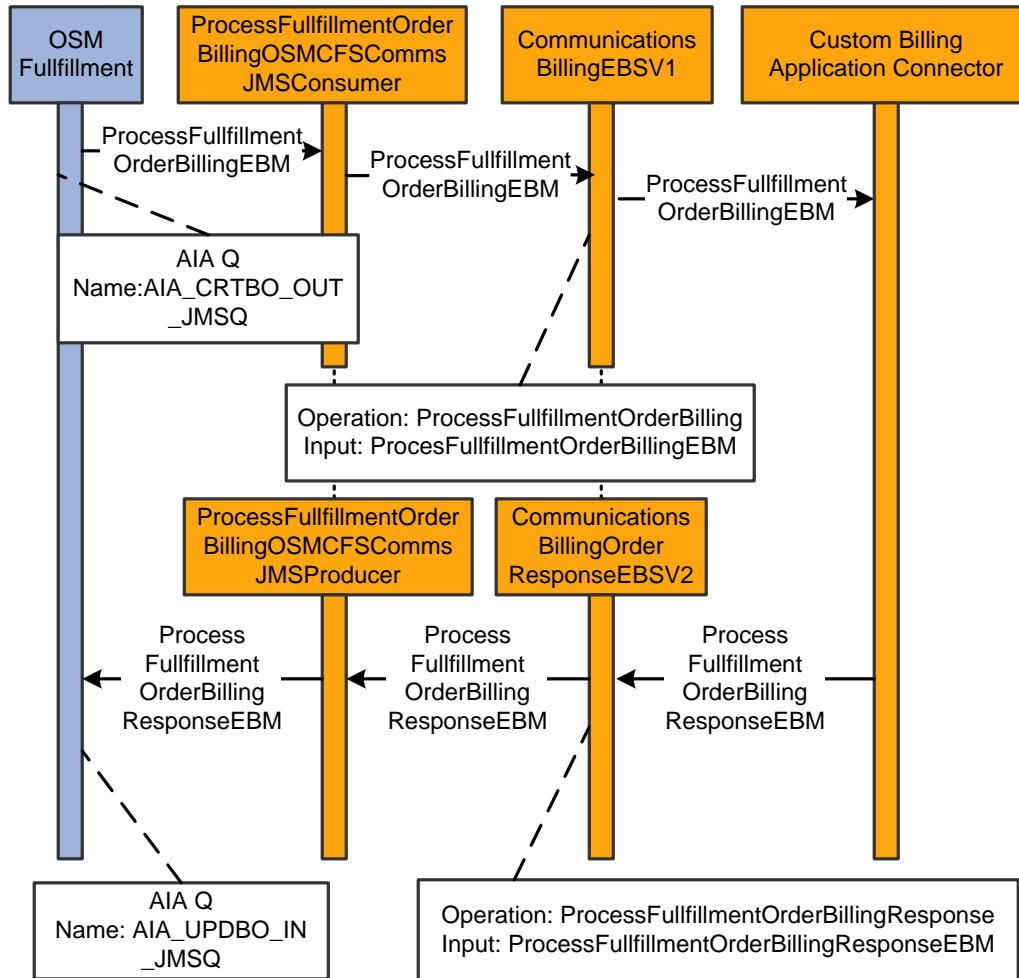
1. After an order is decomposed to order components (suborders), each is targeted to a billing provider, Oracle OSM composes a ProcessFulfillmentOrderBillingAccountListEBM and places it in a JMS Queue, and a messaging bridge pushes the message to the AIA_CRTCUST_OUT_JMSQ messaging queue.
2. The ProcessFulfillmentOrderBillingAccountListOSMCFSCommsJMSConsumer consumes the message from the queue and passes it to the CommunicationsBillingEBSV1 ProcessFulfillmentOrderBillingAccountList operation.

3. A deployment-specific routing rule routes the message to the target billing provider's application business connector service (ABCS). In the case of Oracle Communications Billing and Revenue Management (Oracle BRM) the routing rule routes the message to the appropriate billing account sync flow (CommsProcessFulfillmentOrderBillingAccountListEBF).
4. After a response is received from the billing provider, the CommunicationsBillingResponseEBSV1 ProcessFulfillmentOrderBillingAccountListResponse routes the response back, ProcessFulfillmentOrderBillingAccountListResponseOSMCFSCommsJMSProducer puts the message in the AIA_UPDCUST_IN_JMSQ, which is bridged to an Oracle OSM messaging queue on the WebLogic server.

For more information for when Oracle Communications BRM is the billing system, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill 2.5 Implementation Guide*, "Interfacing Orders to Create Customer Data in BRM."

Interfacing Customer Order Actions to Billing

This diagram illustrates the Oracle Application Integration Architecture (Oracle AIA) integration components used by Oracle OSM to interface order actions to a billing provider:



Oracle OSM interfacing order actions to billing

When this process is initiated, the following events occur:

1. After an order is decomposed to order components (suborders) where each is targeted at a billing provider, Oracle OSM composes a **ProcessFulfillmentOrderBillingEBM** and places it in a JMS Queue and a messaging bridge pushes the message to the **AIA_CRTBO_OUT_JMSQ** messaging queue.
2. The **ProcessFulfillmentOrderBillingOSMCFSCCommsJMSConsumer** consumes the message from the queue and passes it to the **CommunicationsBillingEBSV2** for **ProcessFulfillmentOrderBilling** operation.

A deployment-specific routing rule routes the message to the target billing provider's ABCS. In the case of Oracle BRM, the routing rule routes the message to the appropriate billing connector service (**ProcessFulfillmentOrderBillingBRMCommsProvABCImpl**).

3. After a response is received from the billing provider, the response is routed back by the ProcessFulfillmentOrderBillingResponse operation to the ProcessFulfillmentOrderBillingResponseOSMCFSCommsJMSProducer. This action puts the message in the AIA_UPDBO_IN_JMSQ, which is bridged to an Oracle OSM messaging queue on the WebLogic server.

For more information for when Oracle BRM is the billing system, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide*, “Interfacing Orders to Billing.”

Integration Services

These services are delivered with the integrations:

- ProcessFulfillmentOrderBillingAccountList OSMCFSCommsJMSConsumer
- CommunicationsBillingEBSV1
- ProcessFulfillmentOrderBillingAccountListResponseOSMCFSJMSProducer
- ProcessFulfillmentOrderBillingOSMCFSCommsJMSConsumer
- ProcessFulfillmentOrderBillingResponseOSMCFSCommsJMSProducer

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,” Viewing Integration Scenarios.

ProcessFulfillmentOrderBillingAccountListOSMCFSCommsJMSConsumer

The ProcessFulfillmentOrderBillingAccountListOSMCFSCommsJMSConsumer is an Enterprise Service Bus (ESB) process that has a JMS Adapter Service, which continuously polls the Oracle AIA queue AIA_CRTCUST_OUT_JMSQ.

The ProcessFulfillmentOrderBillingAccountListOSMCFSCommsJMSConsumer dequeues the ProcessFulfillmentOrderBillingAccountListEBM message and routes it to the CommunicationsBillingEBSV1 by calling the ProcessFulfillmentOrderBillingAccountList operation.

This service has one operation: Consume_Message.

CommunicationsBillingEBSV1

For more information about this service, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide*.

ProcessFulfillmentOrderBillingAccountListResponseOSMCFSCommsJMSProducer

The ProcessFulfillmentOrderBillingAccountListResponseOSMCFSCommsJMSProducer is a BPEL process that has an adapter service, which produces the customer response messages to AIA_UPDCUST_IN_JMSQ.

This process has one operation: Produce_PFOBALResponse to produce the message into the AIA_UPDCUST_IN_JMSQ queue. This operation is called by the CommunicationsBillingResponseEBSV1 after the account or customer is interfaced in the billing system.

ProcessFulfillmentOrderBillingOSMCFSCommsJMSPConsumer

The ProcessFulfillmentOrderBillingOSMCFSCommsJMSPConsumer is an ESB process that has a JMS Adapter Service, which continuously polls the AIA_CRTBO_OUT_JMSQ. The ProcessFulfillmentOrderBillingOSMCFSCommsJMSPConsumer dequeues the ProcessFulfillmentOrderBillingEBM message and routes it to the CommunicationsBillingEBSV1 by calling the ProcessFulfillmentOrderBilling operation.

This service has one operation: Consume_Message.

ProcessFulfillmentOrderBillingResponseOSMCFSCommsJMSProducer

The ProcessFulfillmentOrderBillingResponseOSMCFSCommsJMSProducer is a BPEL process that has an adapter service, which produces the order response messages to the AIA_UPDBO_IN_JMSQ.

This process has only one operation: Produce_ProcessFOBResponse to produce the message into the AIA_UPDBO_IN_JMSQ AIA queue. This operation is called by CommunicationsBillingResponseEBSV1 after the order is interfaced into the billing system.

Chapter 5: Understanding the Process Integrations for Siebel CRM

This chapter provides an overview of process integration for Siebel Customer Relationship Management (Siebel CRM) and discusses:

- Submitting orders from Siebel CRM to Oracle Order and Service Management (Oracle OSM).
- Updating statuses from Oracle OSM to Siebel CRM.
- Siebel CRM interfaces.
- Integration services.

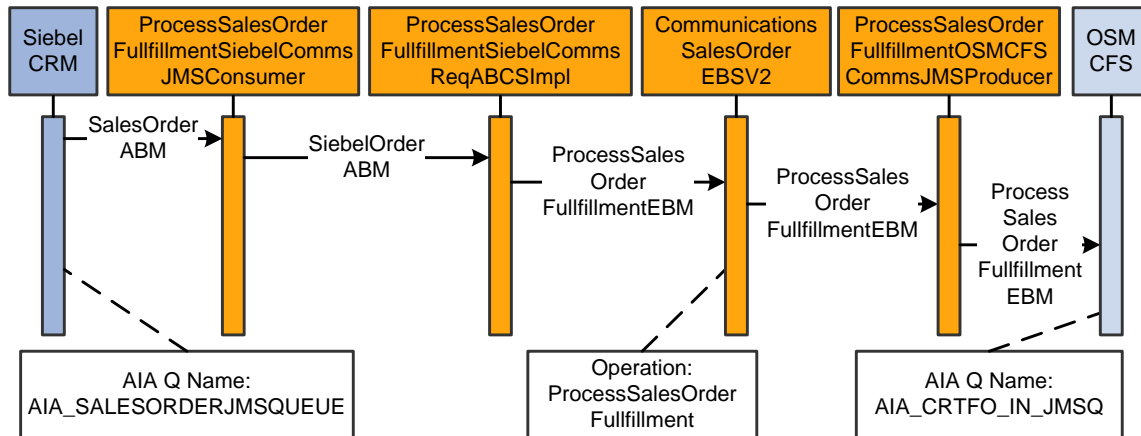
Process Integrations for Siebel CRM Overview

The Oracle Order to Activate Process Integration Pack (PIP) works with the Oracle Order to Bill PIP and participating applications to the following business processes:

- Submitting orders from Siebel CRM to an Oracle OSM Central Fulfillment System (Oracle OSM CFS) is available using both the Oracle Order to Activate PIP and the Oracle Order to Bill PIP. Taking the submitted orders from Oracle Application Integration Architecture (Oracle AIA) to Oracle OSM is only available through the Oracle Order to Activate PIP. When used without the Oracle Order to Activate PIP and Oracle OSM, the Oracle Order to Bill PIP assumes the existence of an order orchestration process that handles interfacing the order to billing and other fulfillment systems.
- Sending order updates from Oracle OSM Central Order Management (Oracle OSM COM) to Siebel CRM. The integration exists in both the Oracle Order to Activate PIP and the Oracle Order to Bill PIP.

Submitting Orders from Siebel CRM to Oracle OSM CFS

This diagram illustrates an order submitted from Siebel CRM to Oracle OSM CFS using Oracle AIA:



Siebel CRM to Oracle OSM CFS sequence diagram

When a new order process is initiated, the following events occur:

1. In Siebel CRM, a user navigates to the Sales Order screen and clicks Submit to submit a new order. This action triggers Siebel to create a Siebel application business message (ABM) with all the captured details. Siebel then enqueues this ABM in the AIA_SALESORDERJMSQUEUE JMS Queue.

2. After the Siebel ABM is enqueued in the AIA_SALESORDERJMSQUEUE, the ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer dequeues the message and passes it on to the ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl. The ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer has a JMS adapter service that polls for any messages in the AIA_SALESORDERJMSQUEUE.

ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer has an embedded sequencer to sequence the messages. If the ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl fails or if the ProcessSalesOrderFulfillmentOSMCFSCommsJMSProducer fails, then the new orders for that account remain waiting in the sequencer table. If the error is a system type error you must resubmit the message from the sequencer table. If the error is a business type error then you must remove the message from sequencer table so that order processing for the customer can resume.

3. The ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl invokes the transformation logic to transform Siebel ABM to the ProcessSalesOrderFulfillmentEBM, and invokes the CommunicationsSalesOrderEBSV2 with the operation ProcessSalesOrderFulfillment.
4. The CommunicationsSalesOrderEBSV2 routes this message to the ProcessSalesOrderFulfillmentOSMCFSCommsJMSProducer interface, which wraps ProcessSalesOrderFulfillmentEBM into CreateOrder message format and enqueues the message into the AIA_CRTFO_IN_JMSQ.

The CreateOrder message is then wrapped in a SOAP envelope. Oracle OSM expects the following additional JMS properties to be set when the JMS payload:

- URI -- /osm/wsapi
 - _wls_mimehdrContent_Type -- text/xml; charset=utf-8
5. After the message is in the Oracle AIA queue, the WebLogic JMS Bridge picks up the message and moves it over to oracle/communications/ordermanagement/WebServiceQueue.
- Oracle OSM picks up the CreateOrder message and then further decomposes the order for fulfillment and provisioning.

For more information about how Oracle OSM processes the order, see [Chapter 3: Understanding the Process Integration for Order Lifecycle Management](#).

For every interaction between Oracle AIA and Oracle OSM, queues and JMS bridges are used to pass information.

For more information about bridges in Oracle OSM, see the *Oracle Communications Order and Service Management, System Administration Guide*.

For more information about JMS bridges concepts and how to configure JMS bridges, see the WebLogic documentation.

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:

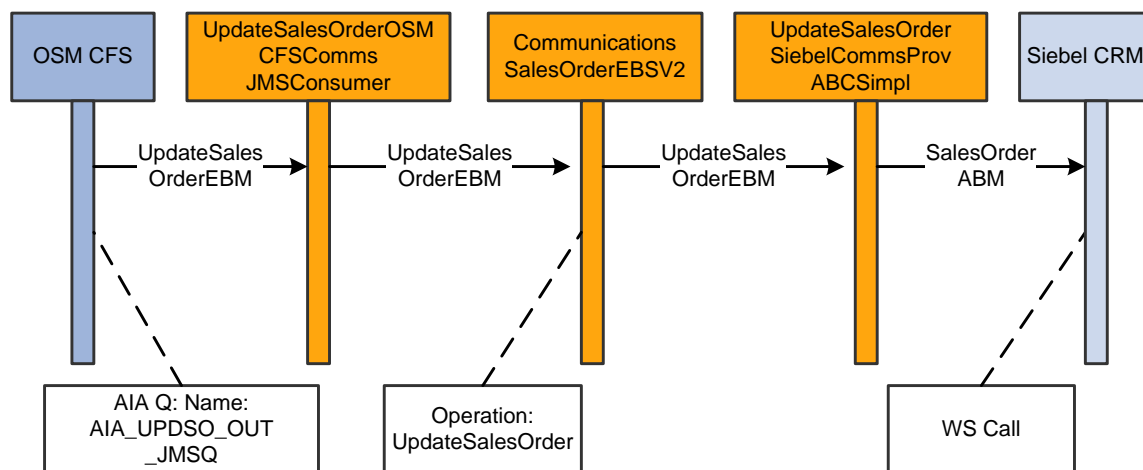
- ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer
- ProcessSalesOrderFulfillmentSiebelCommsSequencer
- ProcessSalesOrderFulfillmentSiebelCommsReqABCImpl
- CommunicationsSalesOrderEBSV2
- ProcessSalesOrderFulfillmentOSMCFSCommsJMSProducer

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 ProcessSalesOrderFulfillmentSiebelCommsReqABCImpl, which invokes transformation logic and then invokes CommunicationsSalesOrderEBSV2. The message is then routed to ProcessSalesOrderFulfillmentOSMCFSCommsJMSProducer	AIA cross-reference entries.	Rollback cross-reference transactions. 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.

Updating Statuses from Oracle OSM CFS to Siebel CRM

This diagram illustrates how to use Oracle OSM CFS to send a sales order data and status update to Siebel:



Oracle OSM CFS to Siebel CRM sequence diagram

When this process is initiated, the following events occur:

1. Oracle OSM creates a message UpdateSalesOrderEBM and enqueues the message to the oracle/communications/ordermanagement/WebServiceUpdateSalesOrderQueue. The WebLogic JMS Bridge moves the message to the Oracle AIA queue called AIA_UPDSO_OUT_JMSQ. The UpdateSalesOrderOSMCFSCCommsJMSConsumer consumes this message and calls the UpdateSalesOrder operation in CommunicationsSalesOrderEBSV2.

Note: The UpdateSalesOrderOSMCFSCCommsJMSConsumer also has a sequencer. This means that 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.

2. The CommunicationsSalesOrderEBSV2 then routes the message to UpdateSalesOrderSiebelCommsProvABCSImpl. This process converts the enterprise business message (EBM) into a Siebel ABM and invokes the Siebel web service to update the order.

For more information about using sequencing logic to make updates to Siebel, see these sections: [“Supporting Order Priorities”](#) and [“Using the ESB Resequencer Feature”](#).

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:

- UpdateSalesOrderOSMCFSCCommsJMSConsumer
- UpdateSalesOrderOSMCFSCCommsJMSConsumer_RS
- CommunicationsSalesOrderEBSV1
- UpdateSalesOrderSiebelCommsProvABCSImpl

Transaction	DB Operations	In Case of Error	Recovery
UpdateSalesOrderOSMCFSCCommsJMSConsumer consumes the message and puts it in the sequencer table defined at the Routing	Message goes into the sequencer table.	Rollback JMS message to AIA_UPDSO_OUT_JMSQ.	Resubmit the order from AIA_UPDSO_OUT_JMSQ.

Transaction	DB Operations	In Case of Error	Recovery
Service UpdateSalesOrderOSMCFSCommsJMConsumer_RS.			
UpdateSalesOrderOSMCFSCommsJMConsumer_RS invokes CommunicationsSalesOrderEBSV1, which routes the message to UpdateSalesOrderSiebelCommsProvABCImpl. 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.*

Siebel CRM Interfaces

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

- **SISOMBillingSubmitOrderWebService:** is the outbound Siebel web service used to submit orders.
- **SWIOrderUpsert** and **SWIOrderUpsertSubProcess:** These are the inbound Siebel web services used to update order information back to Siebel.

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

Integration Services

These services are delivered with the integration:

- **ProcessSalesOrderFulfillmentSiebelCommsJMConsumer**
- **ProcessSalesOrderFulfillmentSiebelCommsReqABCImpl**
- **CommunicationsSalesOrderEBSV2**
- **ProcessSalesOrderFulfillmentOSMCFSCommsJMSProducer**
- **UpdateSalesOrderSiebelCommsProvABCImpl**
- **CommunicationsSalesOrderEBSV2**
- **UpdateSalesOrderOSMCFSCommsJMConsumer**

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

For more information about using Session Pool Manager, see [Appendix B: Using Session Pool Manager](#).

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,” Viewing Integration Scenarios.

Siebel CRM to Oracle OSM CFS

This section describes the services delivered for the Siebel CRM to Oracle OSM CFS flow.

ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer

For information about this service, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide*.

ProcessSalesOrderFulfillmentSiebelCommsReqABCImpl

For information about this service, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide*.

CommunicationsSalesOrderESV2

For information about this service, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide*.

ProcessSalesOrderFulfillmentOSMCFSCommsJMSProducer

The ProcessSalesOrderFulfillmentOSMCFSCommsJMSProducer is a BPEL process that has a JMS Adapter Service, which enqueues the message ProcessSalesOrderFulfillmentEBM into AIA_CRTFO_IN_JMSQ after wrapping it into the CreateOrder message. After the message is enqueued into the Oracle AIA queue, the WebLogic JMS Bridge moves this message into the Oracle OSM CFS Inbound queue called oracle/communications/ordermanagement/WebServiceQueue.

This service has one operation: Initiate. It takes as input the ProcessSalesOrderFulfillmentOSMCFSCommsJMSProducerRequestMessage.

Oracle OSM CFS to Siebel CRM

This section describes the services delivered for the Oracle OSM CFS to Siebel CRM update status flow.

UpdateSalesOrderSiebelCommsProvABCSImpl

For information about this service, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide*.

CommunicationsSalesOrderEBSV2

For information about this service, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide*.

UpdateSalesOrderOSMCFSCommsJMSConsumer

The UpdateSalesOrderOSMCFSCommsJMSConsumer is an ESB process with a JMS Adapter Service. This process dequeues the UpdateSalesOrderEBM message from the AIA_UPDSO_OUT_JMSQ. The Oracle OSM CFS, using the WebLogic JMS Bridge, then enqueues the messages to update the order status in Siebel.

This service has one operation: Consume_Message.

Chapter 6: Understanding the Process Integration for Order Provisioning

This chapter provides an overview of the process integration for order provisioning and discusses:

- Creating provisioning orders.
- Updating fulfillment orders.
- Solution assumptions and constraints.
- Industry application integration architecture (AIA) components.
- Integration services.

Process Integration for Order Provisioning Overview

Customer Order fulfillment requests, both Qualify and Deliver types, received in Oracle Order and Service Management Central Order Management (Oracle OSM COM) from Siebel Customer Relationship Management (Siebel CRM) are decomposed into suborders (called order components) depending on the fulfillment topology and the fulfillment dependencies. Some of the order components would be targeted for a provisioning (Service Order Management) stack. Oracle OSM COM uses Oracle AIA services to pass provision order requests to Oracle OSM Service Order Management (Oracle OSM Provisioning) instance or any third-party OSM system.

When Oracle OSM is also used as the Service Order Management application, Oracle OSM manages the order lifecycle management (OLM) events of the service order. For cancel and revision requests, Oracle OSM generates and executes compensation plans to efficiently match the change. OLM also manages order data and status updates, as well as fallout incidents.

Throughout the fulfillment process, Oracle OSM Provisioning sends status and data updates to Oracle OSM COM. Within this guide, we use *Provisioning* and *Service Order Management* interchangeably.

Creating Provisioning Orders

ProcessProvisioningOrder is the ProvisioningOrderEBS operation used by orchestration to request provisioning for a customer Order Component (suborder). It is an asynchronous service that takes ProcessProvisioningOrderEBM, which includes most of the SalesOrderEBO attributes, as its input. When it errors, the response comes through the Oracle AIA common error schema, otherwise, there is no response for this message. Provisioning Service operation responses are made through ProcessFulfillmentOrderUpdate service operations in all cases except an interface error or request failure. In these cases, the responses are passed to an Oracle AIA Error Handling service, which passes order failure information from provisioning to customer Order Management for order fallout handling.

For more information about order fallout, see [Chapter 7: Using Order Fallout Management in Oracle Order to Activate](#).

Updating Fulfillment Orders

This feature provides the ability to update Oracle OSM COM with Oracle OSM Provisioning milestones, status, and data.

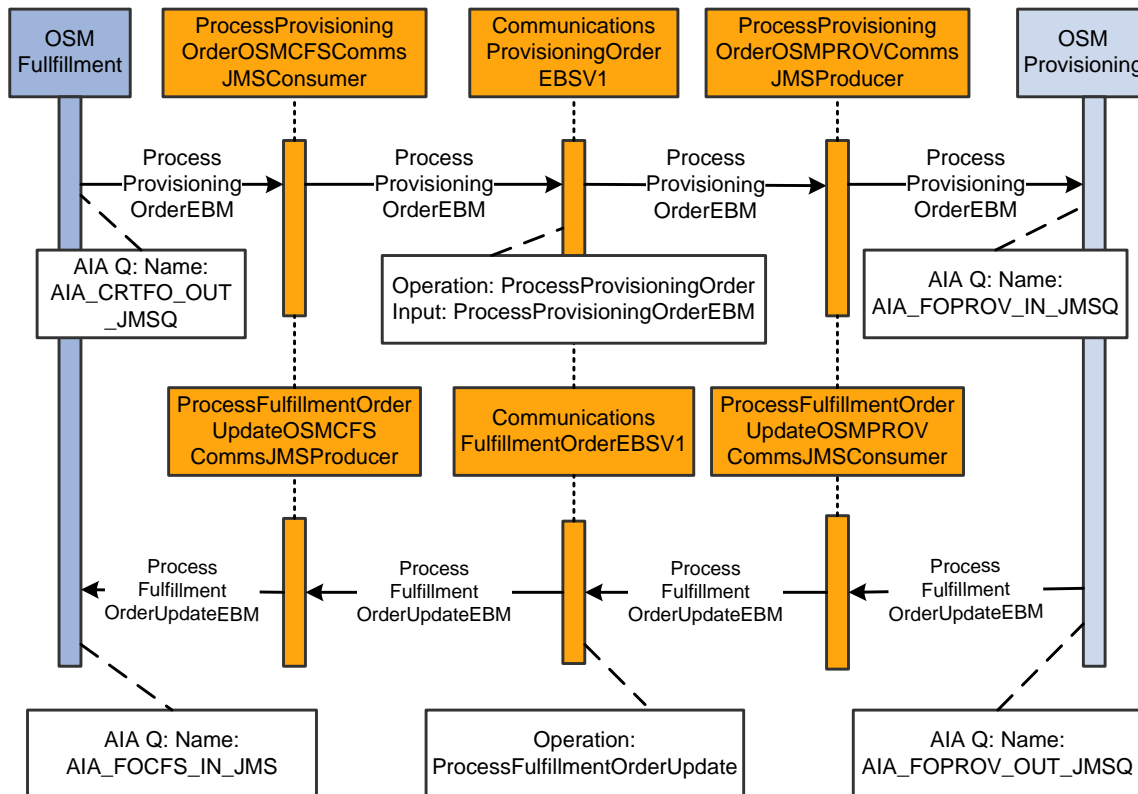
Order Status Management is an integral capability of Oracle OSM COM. Oracle OSM COM provides for a configurable order status management across different fulfillment systems, including Oracle OSM Provisioning.

In addition, order milestones are configured to track order fulfillment progress. Fulfillment system responses and status updates are used to trigger evaluation of rules that progress the order item status and realize new milestones and in turn trigger aggregation rules that update the order level status.

Finally, several attributes are populated during design and assign that are critical to pass to other fulfillment systems and Siebel CRM.

Oracle OSM CFS to Oracle OSM Provisioning Flow

This is the sequence diagram requesting provisioning of an order to Oracle OSM Provisioning and receiving updates back:



Oracle OSM fulfillment (COM) to Oracle OSM Provisioning sequence diagram

When this process is initiated, the following events occur:

1. Whenever a new order needs to be created in Provisioning, a ProcessProvisioningOrderEBM message is created by the Oracle OSM Central Fulfillment System (Oracle OSM CFS) (for OSM COM).

The message is enqueued in:

oracle/communications/ordermanagement/WebServiceCreateProvisioningOrderQueue.

The WebLogic JMS Bridge then moves this message into the AIA_CRTFO_OUT_JMSQ.

2. The ProcessProvisioningOrderOSMCFSCommsJMSConsumer monitors the AIA_CRTFO_OUT_JMSQ queue and dequeues the message from the oracle AIA queues using a JMS adapter service whenever a message exists. The message is then routed to the CommunicationsProvisioningOrderEBSV1 by calling the ProcessProvisioningOrder operation.
3. The CommunicationsProvisioningOrderEBSV1 then routes the message to the ProcessProvisioningOrderOSMPROVJMSProducer.
4. The routing mentioned in the previous step produces the message into the AIA_FOPROV_IN_JMSQ. The WebLogic JMS Bridge then moves the message into the WebLogic JMS queue: oracle/communications/ordermanagement/WebServiceQueue of the Oracle OSM Provisioning instance (Oracle OSM Provisioning Inbound queue). Oracle OSM Provisioning then picks up the message from this queue and processes it accordingly.
5. During provisioning, one or more update messages are enqueued by Oracle OSM Provisioning in the WebLogic queue called oracle/communications/ordermanagement/WebServiceUpdateFulfillmentOrderQueue (Oracle OSM Provisioning Outbound queue). The WebLogic JMS Bridge then moves the message back to the AIA_FOPROV_OUT_JMSQ. The ProcessFulfillmentOrderUpdateOSMPROVCommsJMSConsumer dequeues the ProcessFulfillmentOrderUpdateEBM message and passes it on to the CommunicationsFulfillmentOrderEBSV1 with the operation ProcessFulfillmentOrderUpdate.
6. CommunicationsFulfillmentOrderEBSV1 routes the message to the ProcessSalesOrderFulfillmentUpdateOSMCFSCommsJMSProducer. The producer then enqueues the ProcessFulfillmentOrderUpdateEBM to the AIA_FOCFS_IN_JMSQ. The WebLogic JMS Bridge moves this message into the WebLogic queue: oracle/communications/ordermanagement/WebServiceCreateProvisioningOrderResponseQueue (OSM CFS Inbound queue). Oracle OSM picks up this message to update the status of the order.

For more information about the events that occur when this process initiates, see [Chapter 3: Understanding the Process Integration for Order Lifecycle Management](#).

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:

- ProcessProvisioningOrderOSMCFSCCommsJMSConsumer
- CommunicationsProvisioningOrderEBSV1
- ProcessProvisioningOrderOSMPROVCommsJMSProducer
- ProcessFulfillmentOrderUpdateOSMPROVCommsJMSConsumer
- CommunicationsFulfillmentOrderEBSV1
- ProcessFulfillmentOrderUpdateOSMCFSCCommsJMSProducer

Transaction	DB Operations	In Case of Error	Recovery
ProcessProvisioningOrderOSMCFSCCommsJMSConsumer calls CommunicationsProvisioningOrderEBSV1, which routes the message to ProcessProvisioningOrderOSMPROVCommsJMSProducer. The producer produces the message into AIA_FOPROV_IN_JMSQ. The WebLogic JMS Bridge moves the message from AIA_FOPROV_IN_JMSQ into the WebLogic JMS queue.	None.	Rollback JMS message to originating queue AIA_CRTFO_OUT_JMSQ.	Resubmit the message from AIA_CRTFO_OUT_JMSQ.
During provisioning, update messages are dequeued by Oracle OSM and eventually moves to AIA_FOPROV_OUT_JMSQ. ProcessFulfillmentOrderUpdateOSMPROVCommsJMSConsumer passes the message to CommunicationsFulfillmentOrderEBS1, which routes the message to ProcessFulfillmentOrderUpdateOSMCFSCCommsJMSProducer. The producer produces the message to AIA_FOCFS_IN_JMSQ.	None.	Rollback JMS message to the originating queue AIA_FOPROV_OUT_JMSQ.	Resubmit the order from AIA_FOPROV_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.

Solution Assumptions and Constraints

One or more Oracle OSM Provisioning Cartridges must be deployed. They preserve the Oracle AIA interfaces listed in this chapter.

For more information about product-specific assumptions and constraints, see the Oracle Order and Service Management product documentation.

Industry AIA Components

The process integration uses these industry components:

- ProvisioningOrderEBO
 - ProcessProvisioningOrderEBM
- FulfillmentOrderEBO
 - ProcessFulfillmentOrderUpdateEBM

The industry EBO and EBM XSD files are located here: `http://<server name>:<port number>/AIAComponents/EnterpriseObjectLibrary/Industry/Communications/EBO/`

The industry EBS WSDL files are located here: `http://<server name>:<port number>/AIAComponents/EnterpriseBusinessServiceLibrary/Industry/Communications/`

For detailed documentation about individual enterprise business objects (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 enterprise business service (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.”

Integration Services

These services are delivered with this integration:

- ProcessProvisioningOrderOSMCFSCommsJMConsumer
- CommunicationsProvisioningOrderEBSV1.
- ProcessFulfillmentOrderUpdateOSMPROVCommsJMConsumer
- ProcessProvisioningOrderOSMPROVCommsJMSPProducer
- CommunicationsFulfillmentOrderEBSV1.
- ProcessFulfillmentOrderUpdateOSMCFSCommsJMSPProducer

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.”

ProcessProvisioningOrderOSMCFSCommsJMSConsumer

For interacting with Oracle OSM Provisioning, OSM COM pushes ProcessProvisioningOrderEBM message into
oracle/communications/ordermanagement/WebServiceCreateProvisioningOrderQueue queue.
WebLogic JMS Bridge transfers the message into AIA_CRTFO_OUT_JMSQ.

The ProcessProvisioningOrderOSMCFSCommsJMSConsumer is an ESB process that has a JMS Adapter Service. This Enterprise Service Bus (ESB) service continuously polls the AIA_CRTFO_OUT_JMSQ. The ProcessProvisioningOrderOSMCFSCommsJMSConsumer dequeues the ProcessProvisioningOrderEBM and routes it to the CommunicationsProvisioningOrderEBSV1 by calling the ProcessProvisioningOrder operation. This service has one operation: Consume_Message.

CommunicationsProvisioningOrderEBSV1

The CommunicationsProvisioningOrderEBSV1 process performs this routing operation:

ProcessProvisioningOrder: Routes the ProcessProvisioningOrderEBM to the ProcessProvisioningOrderOSMPROVJMSProducer.

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.”

ProcessFulfillmentOrderUpdateOSMPROVCommsJMSConsumer

For interacting with Oracle OSM COM, Oracle OSM Provisioning pushes ProcessFulfillmentOrderUpdateEBM message into
oracle/communications/ordermanagement/FulfillmentUpdateWSQueue. WebLogic JMS Bridge transfers this message into AIA_FOPROV_OUT_JMSQ.

The ProcessFulfillmentOrderUpdateOSMPROVCommsJMSConsumer is an ESB process with a JMS Adapter Service. The ProcessFulfillmentOrderUpdateOSMPROVCommsJMSConsumer dequeues the ProcessFulfillmentOrderUpdateEBM from AIA_FOPROV_OUT_JMSQ and routes it to the CommunicationsFulfillmentOrderEBSV1 by calling the ProcessFulfillmentOrderUpdate operation.

This service has one operation: Consume_Message.

ProcessProvisioningOrderOSMPROVCommsJMSProducer

The ProcessProvisioningOrderOSMPROVCommsJMSProducer is a BPEL process that has a JMS Adapter Service. This BPEL process is responsible for pushing the ProcessProvisioningOrderEBM message into the AIA_FOPROV_IN_JMSQ. The WebLogic JMS Bridge then enqueues the message into the Oracle OSM Provisioning queue oracle/communications/ordermanagement/WebServiceQueue. Oracle OSM Provisioning then consumes this message and processes it further.

This service has one operation: Initiate.

CommunicationsFulfillmentOrderEBSV1

The CommunicationsFulfillmentOrderEBSV1 is an ESB process that performs this routing operation:

ProcessFulfillmentOrderUpdate : Routes the ProcessFulfillmentOrderUpdateEBM message to the ProcessFulfillmentOrderUpdateOSMCFSCCommsJMSProducer

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."

ProcessFulfillmentOrderUpdateOSMCFSCCommsJMSProducer

The ProcessFulfillmentOrderUpdateOSMCFSCCommsJMSProducer is a BPEL process that has a JMS Adapter Service. This BPEL process is responsible for pushing the ProcessFulfillmentOrderUpdateEBM Message into the AIA_FOCFS_IN_JMSQ. The WebLogic JMS Bridge then enqueues the message into the Oracle OSM CFS queue oracle/communications/ordermanagement/WebServiceCreateProvisioningOrderResponseQueue.

This service has one operation: Initiate.

Chapter 7: Using Order Fallout Management in Oracle Order to Activate

This chapter provides an overview of using Order Fallout Management in Oracle Order to Activate and discusses:

- Creating trouble tickets in Oracle Application Integration Architecture (Oracle AIA).
- Creating trouble tickets in Oracle Order and Service Management (Oracle OSM).
- Solution assumptions and constraints.
- Siebel Customer Relationship Management (Siebel CRM) interfaces.
- Industry AIA components.
- Integration services.
- Configuring order fallout using the Oracle Order to Bill Process Integration Pack (PIP).
- Fallout enabled services.

Using Order Fallout Management in Oracle Order to Activate Overview

The order fallout management integration in the Oracle Order to Activate PIP consists of the following high-level flows:

- Listen to the common error topic used in the Oracle AIA Error Handling Framework (AIA_ERROR_TOPIC) for errors or faults specific to orders, enrich these fault messages, and then publish them to the central OFM in Oracle Order and Service Management (Oracle OSM).
- Create trouble tickets in Siebel for individual and batch or bulk orders from Oracle OSM.
- Update trouble tickets in Siebel from Oracle OSM.

Oracle AIA or Oracle OSM can initiate the creation of trouble tickets. This option is configurable. Installing the Oracle Order to Activate PIP automatically configures order fallout to occur in the Oracle OSM Central Fulfillment System (Oracle OSM CFS).

For more information about how to configure fallout, see [Chapter 8: Configuring the Oracle Order to Activate PIP](#), “Install Sequence and Configurations for Using Fallout”.

Consider the following points with regard to Oracle Order to Activate fallout flows:

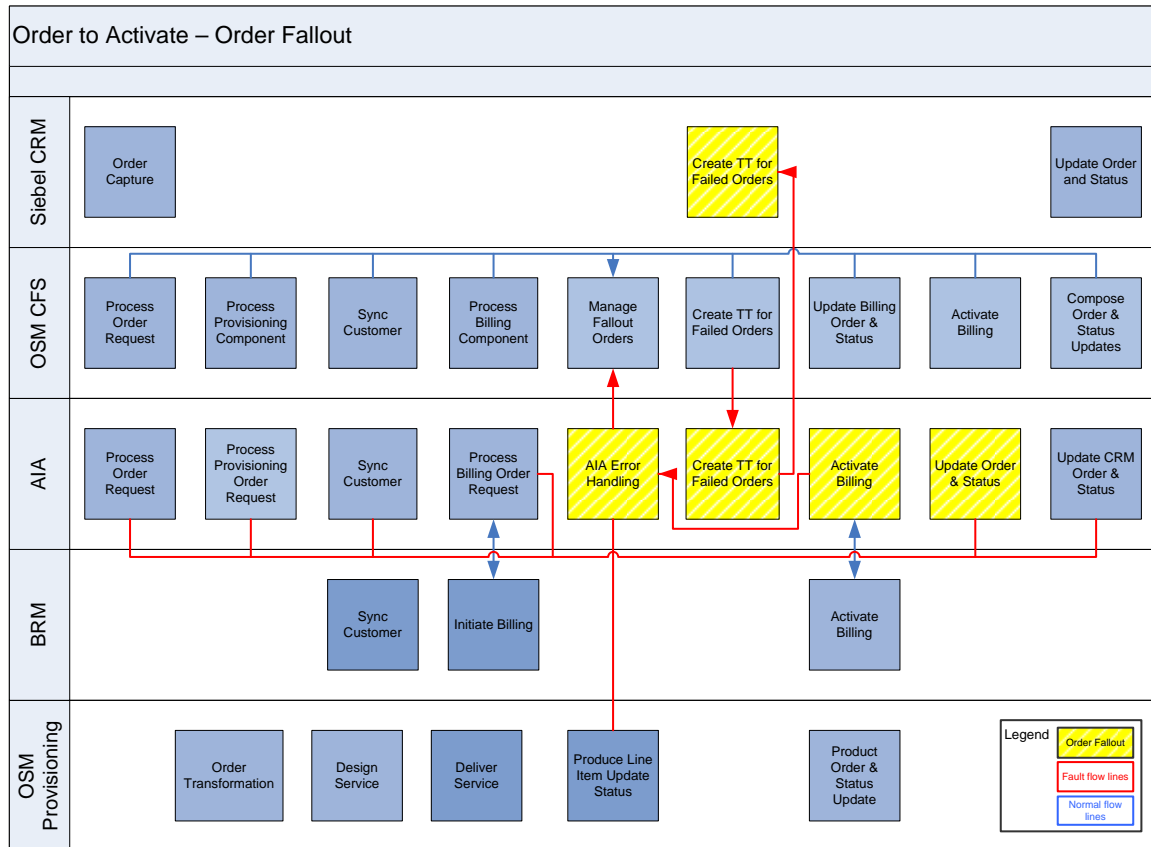
- Trouble tickets are created in Siebel by Oracle OSM on a per-order or per-system basis. The failure of different orders in the same system will generate different trouble tickets, and the

failure of the same order in a different system will generate a different trouble ticket, but multiple order line item failures for the same order in the same system will generate only one trouble ticket. The additional order line item failure information will be appended.

- If the cancellation of a failed order is needed as part of the recovery flow, the Fallout Specialist should cancel the order from Oracle OSM.
- Any custom process flow that invokes the creation of an order failure notification must ensure that no multiple notifications are generated for the same order failure.

Order to Activate Fallout Integration Flows

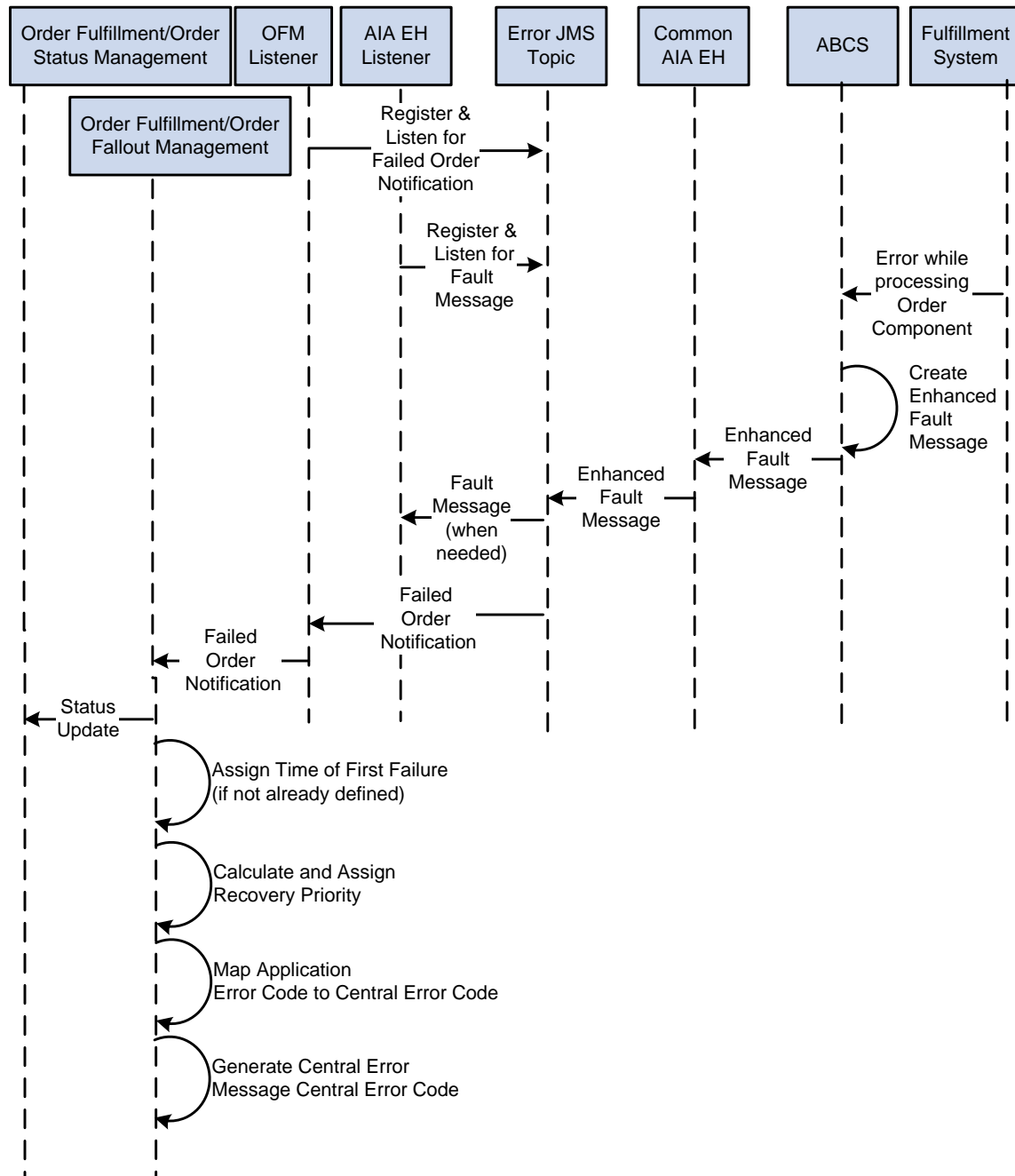
This diagram shows the high-level flows involved in Oracle Order to Activate fallout. It identifies the possible sources of failed orders, capturing these faults using the Oracle AIA Error Handling Framework and creation of the trouble ticket from OSM in Siebel for the failed order:



Order fallout flows using Oracle OSM

Capturing the Fault

This sequence diagram shows the interactions taking place when an order failure is detected by a fulfillment system:



Capturing the fault sequence diagram

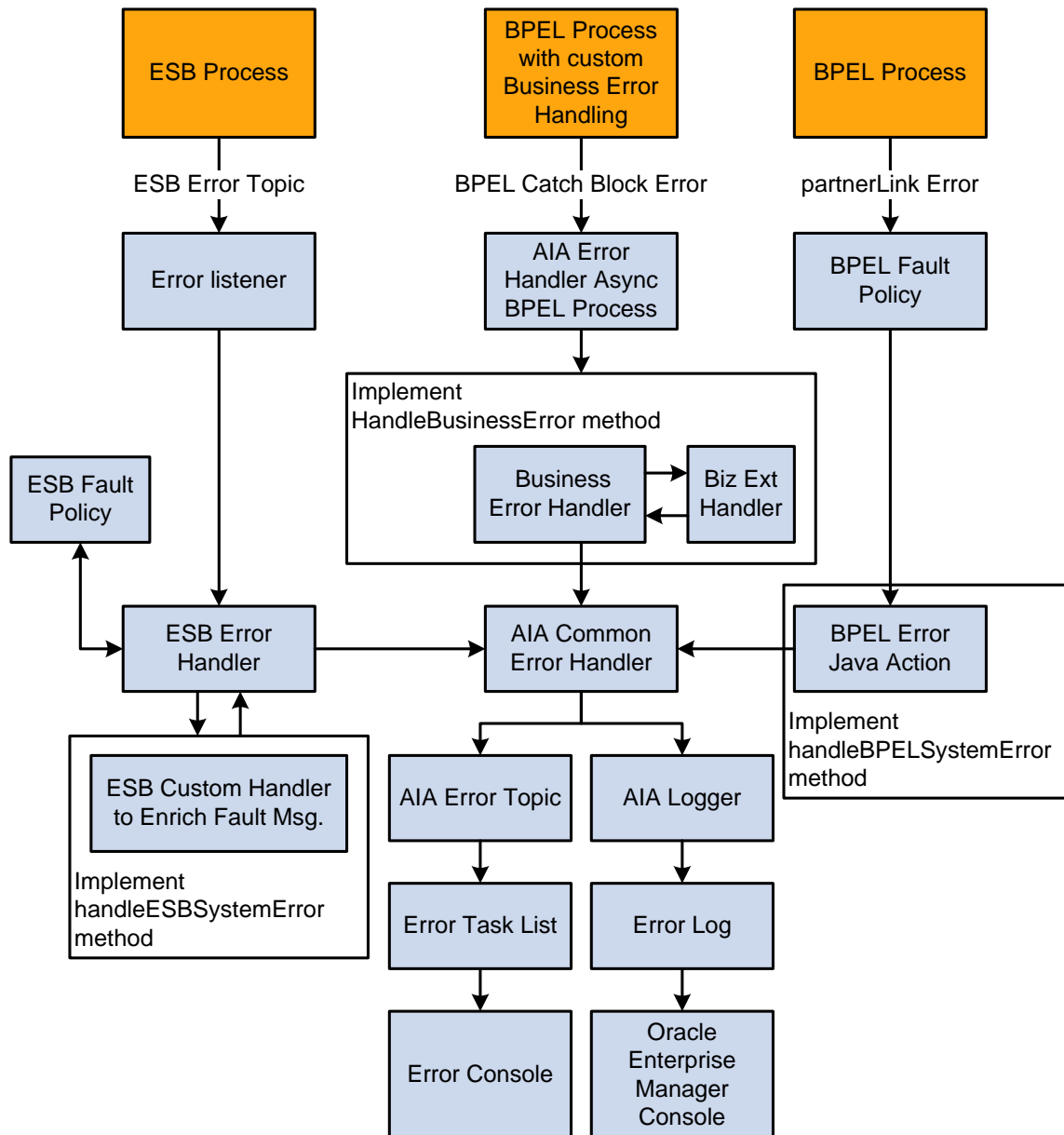
Oracle Order to Activate Order Fallout Management uses the Oracle AIA Error Handling Framework to capture faults across order processing.

The Oracle AIA Error Handling Framework:

- Allows custom enrichments to the fault message.

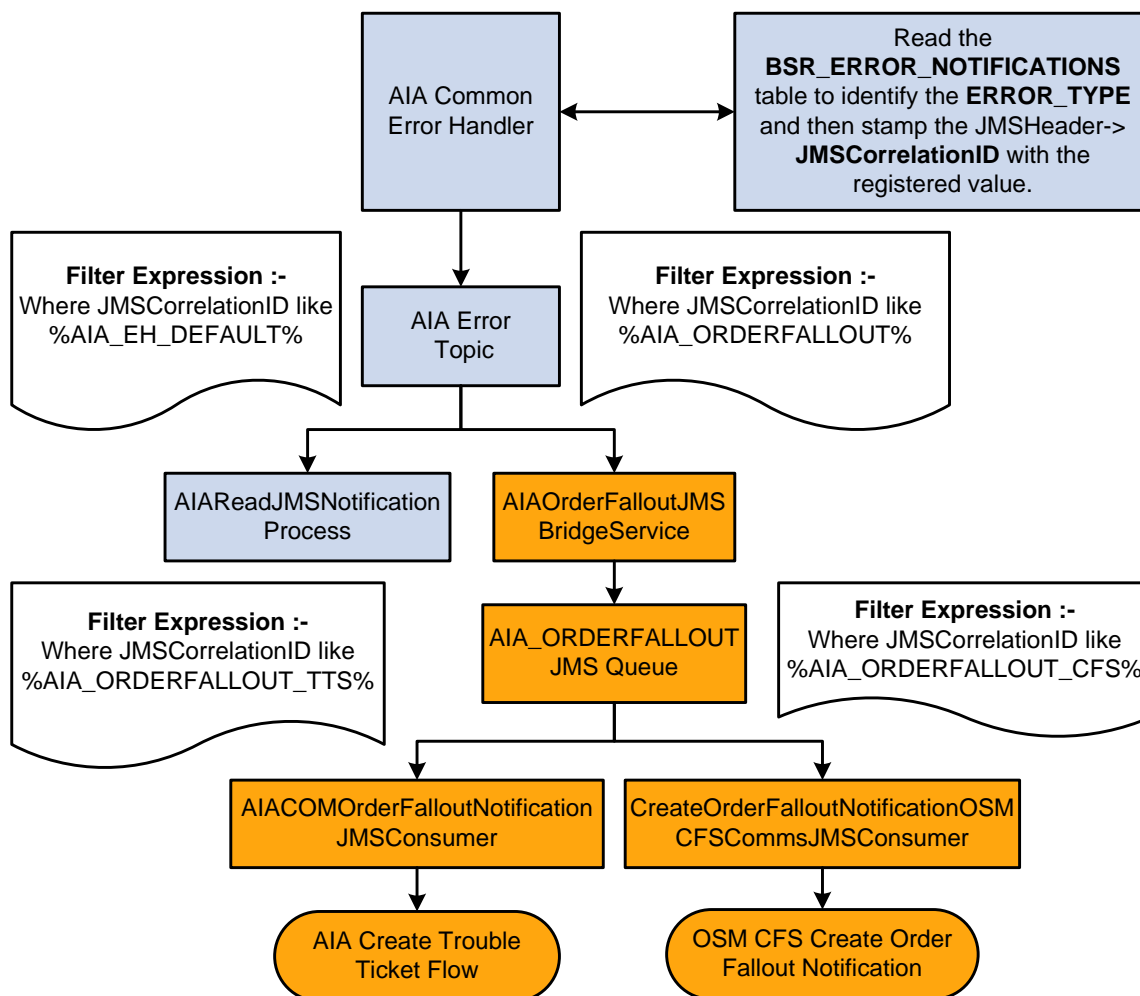
- Publishes the enriched fault message to the AIAError topic.
- Provides a mechanism by which the Order Fallout Listener process picks only the messages that are relevant to the order failure.

This diagram shows how the Oracle AIA Error Handling Framework is leveraged to submit an order failure notification to the central Order Fallout Management functionality within Oracle OSM Fulfillment:



Creation and submission of a fault message for order failure to the AIA Error Topic

The custom listener selectively picks up the messages from the AIA Error Topic and publishes them to Oracle OSM, as shown in this diagram:



Custom listener picks up fault message for order failure and publishes to Oracle OSM

Here are the steps:

1. All of the enriched fault messages with the Order Failure details are posted to the AIA Error Topic (AIA_ERROR_TOPIC).
2. Messages that are specific to order failure are stamped with a JMS Correlation ID like AIA_ORDERFALLOUT.
3. The AIAOrderFalloutJMSBridgeService consumes the messages from the AIA_ERROR_TOPIC with JMSCorrelationID like AIA_ORDERFALLOUT and publishes them to the AIA_ORDERFALLOUT_JMSQ queue. (This queue is introduced to persist the order failure messages and ensure the messages are not lost if there are errors.)
4. Messages that are specific to order failure will have AIA_ORDERFALLOUT_TTS or AIA_ORDERFALLOUT_CFS, depending on whether the trouble ticket is created directly by Oracle AIA or the order failure notification is sent to Oracle OSM CFS in the JMS Correlation ID.
5. The AIACOMOrderFalloutNotificationJMSConsumer picks up the fault messages and initiates the appropriate Create Trouble Ticket flow. For the Create Trouble Ticket flow:

- a. If the JMSCorrelationID = AIA_ORDERFALLOUT_TTS, the trouble ticket is directly created by Oracle AIA. (This is the default configuration.)
- b. If the JMSCorrelationID = AIA_ORDERFALLOUT_CFS, the order failure notification is sent to Oracle OSM CFS and Oracle OSM CFS initiates the Create Trouble Ticket Request.

Describing Business Errors and System Errors

Two types of errors might occur: business errors or system errors. If a system error occurs, the fallout specialist may not have to take any action in Siebel CRM or Oracle OSM CFS. Instead, the specialist must use the resubmit utility to resubmit the stalled message after correcting the system problem.

For more information about the Message Resubmission Utility application programming interface (API), see the *Oracle Enterprise Service Bus Developer's Guide*, "Using the Message Resubmission Utility API."

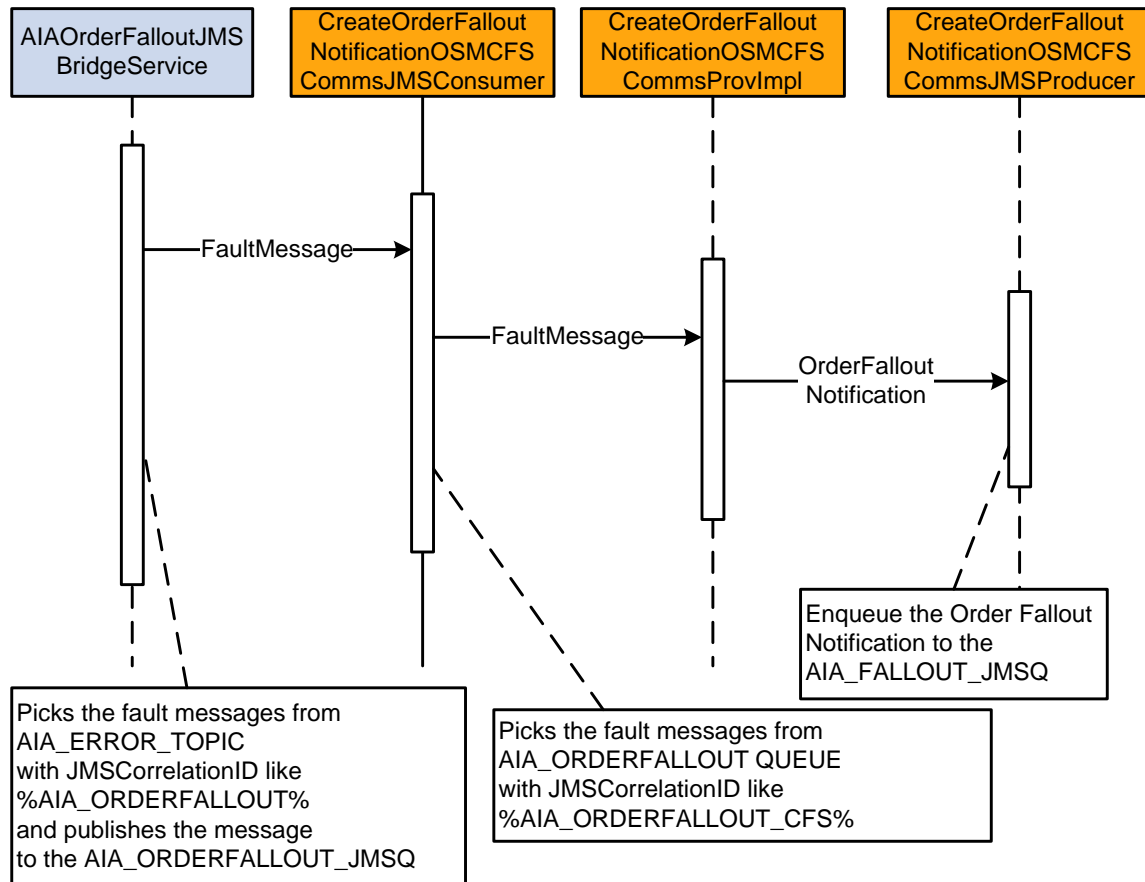
If a business error occurs, the fallout specialist takes the corrective actions mentioned previously. In addition, the fallout specialist may have to unblock the resequencer queue.

For more information about how to resolve errors in flows where sequencing is enabled, see [Using the ESB Resequencer Feature](#).

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.

Creating Trouble Tickets in Oracle OSM

This diagram illustrates how Oracle OSM creates a trouble ticket:



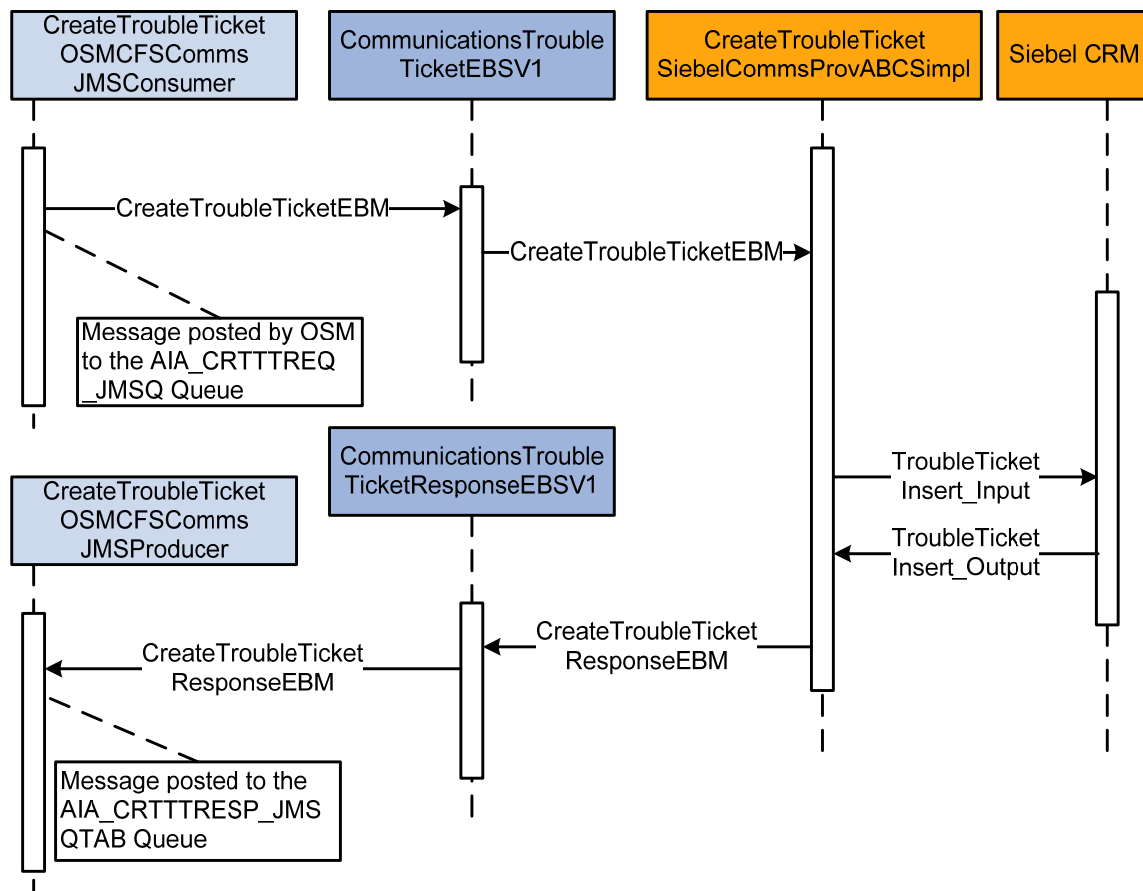
Order failure notification to Oracle OSM

When this process initiates, the following events occur:

1. The Enriched Fault Message containing the details pertaining to orders are pushed to the AIA_ERROR_TOPIC using the Oracle AIA Error Handling Framework. These messages are stamped with a JMS Correlation ID, for example, AIA_ORDERFALLOUT.
2. AIAOrderFalloutJMSBridgeService picks up the messages with JMSCorrelationID, for example, AIA_ORDERFALLOUT and publishes them to the AIA_ORDERFALLOUT_JMSQ queue.
3. AIACOMOrderFalloutNotificationJMSConsumer picks up the messages stamped with the JMS Correlation ID AIACOM_ORDERFALLOUT_CFS from the AIA_ORDERFALLOUT_JMSQ and invokes CreateOrderFalloutNotificationOSMCFSCommsProvImpl.
4. CreateOrderFalloutNotificationOSMCFSCommsProvImpl parses the fault message, transforms it to OrderFalloutNotification message, and invokes the CreateOrderFalloutNotificationOSMCFSCommsJMSProducer service.
5. CreateOrderFalloutNotificationOSMCFSCommsJMSProducer pushes the message to the AIA_FALLOUT_JMSQ queue. Oracle OSM CFS picks up this message, marks the order as failed, and initiates the request to create a trouble ticket.

Creating a Trouble Ticket in Siebel CRM by Oracle OSM

This sequence diagrams describe the creation of trouble tickets in Siebel CRM by Oracle OSM:



Creating a trouble ticket in Siebel CRM by Oracle OSM

When this process initiates, the following events occur:

1. The Oracle OSM Fulfillment system produces the CreateTroubleTicketEBM in a queue. The message bridge pushes this message to the AIA_CRTTTREQ_JMSQ queue.
2. CreateTroubleTicketOSMCFSCCommsJMSConsumer picks up the message from the queue and passes it to the CommunicationsTroubleTicketEBSV1 service using the CreateTroubleTicket operation.
3. The CommunicationsTroubleTicketEBSV1 routes the message to the CreateTroubleTicketSiebelCommsProvABCSImpl service.
4. The CreateTroubleTicketSiebelCommsProvABCSImpl service invokes the Siebel web service (SWITroubleTicketIO.wsdl: SWITroubleTicketInsert) synchronously, and the response trouble ticket ID is received in the form of a SWITroubleTicketInsert_Output message.
5. CommsTroubleTicketEBS invokes the CreateTroubleTicketSiebelCommsProvABCSImpl with the CreateTroubleTicketEBM. This process transforms the CreateTroubleTicketEBM to TroubleTicketInsert_Input ABM.

6. The CreateTroubleTicketSiebelCommsProvABCImpl service invokes the Siebel web service (ServiceRequest.wsdl) synchronously, and the response trouble ticket ID is received in the form of a TroubleTicketInsert_Output message. This ABM is transformed to CreateTroubleTicketResponseEBM, and the CommunicationsTroubleTicketResponseEBSV1 is invoked with the CreateTroubleTicketResponse operation.
7. The CommunicationsTroubleTicketResponseEBSV1 routes the message to the CreateTroubleTicketRespOSMCFSCommsJMSProducer service, which pushes the message to the AIA_CRTTRESP_JMSQTAB queue.
8. Oracle OSM CFS picks up the message and stores the TroubleTicketID for reference.

Defining Transaction Boundaries and Recovery Details

For this flow there are three 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:

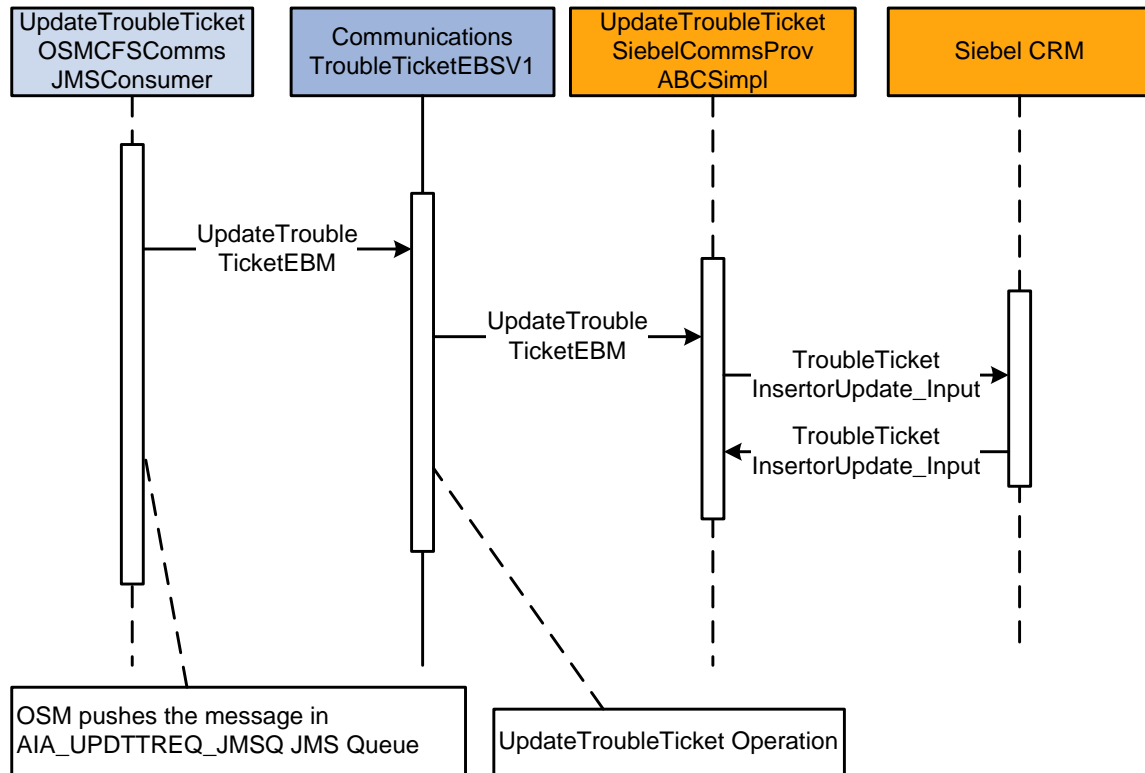
- AIAOrderFalloutJMSBridgeService
- CreateOrderFalloutNotificationOSMCFSCommsJMSConsumer
- CreateOrderFalloutNotificationOSMCFSCommsProvImpl
- CreateOrderFalloutNotificationOSMCFSCommsJMSProducer
- CreateTroubleTicketOSMCFSCommsJMSConsumer
- CommunicationsTroubleTicketEBSV1
- CreateTroubleTicketSiebelCommsProvABCImpl
- CommunicationsTroubleTicketResponseEBSV1
- CreateTroubleTicketRespOSMCFSCommsJMSProducer

Transaction	DB Operations	In Case of Error	Recovery
The AIAOrderFalloutJMSBridgeService picks up the messages from AIA_ERROR_TOPIC 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.
CreateOrderFalloutNotificationOSMCFSJMSC onsumer picks up messages with the JMS Correlation ID like AIA_ORDERFALLOUT_CFS and invokes CreateOrderFalloutNotificationOSMCFSCommsProvImpl, which parses fault message and invokes CreateOrderFalloutNotificationOSMCFSJMSProducer.	Message is submitted to Oracle OSM through AIA_FALLOUT_JMSQ.	Rollback the message to AIA_ORDERFALLOUT_JMSQ.	Resubmit from AIA_ORDERFALLOUT_JMSQ.
CreateTroubleTicketOSMCFSCommsJMSC onsumer picks up message and invokes CommunicationsTroubleTicketEBSV1. The EBS routes the message to CreateTroubleTicketSiebelCommsProvABCSImpl, which invokes Siebel web service. Response trouble ticket ID is received. Response message is created, which invokes CommunicationsTroubleTicketResponseEBSV1. The EBS routes message to CreateTroubleTicketRespOSMCFSCommsJMSProducer.	AIA cross-references created. Message goes to queue AIA_CRTTTRESP_JMSQ.	Rollback the message to AIA_FALLOUT_JMSQ.	Resubmit from AIA_FALLOUT_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 a Trouble Ticket in Siebel CRM by Oracle OSM

This sequence diagram describes the Trouble Ticket Update flow from Oracle OSM to Siebel CRM:



Updating a Trouble Ticket in Siebel CRM by Oracle OSM

When this process initiates, the following events occur:

1. The Oracle OSM Fulfillment system produces the **UpdateTroubleTicketEBM** in the **AIA_UPDTTREQ_JMSQ** queue.
2. **UpdateTroubleTicketOSMCFSCCommsJMSConsumer** picks up the message from the queue and invokes **CommunicationsTroubleTicketEBSV1** using the **UpdateTroubleTicket** operation.
3. The **CommunicationsTroubleTicketEBSV1** invokes the **UpdateTroubleTicketSiebelCommsProvABCSImpl** service. In this provider process, the EBM is transformed to **TroubleTicketInsertorUpdate_Input** ABM, and the Siebel web service is invoked to update the trouble ticket.

For more information about using order fallout, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide*.

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:

- UpdateTroubleTicketOSMCFSCommsJMSConsumer
- CommunicationsTroubleTicketEBSV1
- UpdateTroubleTicketSiebelCommsProvABCSImpl

Transaction	DB Operations	In Case of Error	Recovery
UpdateTroubleTicketOSMCFSCommsJMSConsumer picks up the message from the queue and invokes CommunicationsTroubleTicketEBSV1. The EBS routes the message to UpdateTroubleTicketSiebelCommsProvABCSImpl, which invokes the Siebel web service to update the trouble ticket.	AIA cross-references updated. Trouble Ticket updated in Siebel.	Message goes back to the originating queue AIA_UPDTTREQ_JMSQ.	Resubmit from AIA_UPDTTREQ_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.*

Solution Assumptions and Constraints

These are the assumptions and constraints:

- Order fallout management functionality handles orders that failed after being submitted by Siebel CRM.
- When an order revision fails upon arrival in Oracle OSM, a new trouble ticket for the revision is created, and any existing trouble ticket for the base order is preserved. In this case, the trouble ticket acts as an important notification of the failed on arrival condition. The side effect is that the fallout specialist must manually close the trouble ticket for the revision that failed upon arrival.
- The Oracle Order to Activate PIP is a superset of the Oracle Order to Bill PIP.

For more information about the assumptions and constraints, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide*, "Understanding the Process Integration for Order Fallout Management."

Siebel CRM Interfaces

The process integration for order fallout management uses this Siebel CRM interface:

- SWI Trouble Ticket Service

This inbound web 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, then a new trouble ticket is created and the trouble ticket number is returned. If the request is to update a specific trouble ticket, typically to close the trouble ticket, then the trouble ticket is updated.

For more information about web services, 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
 - UpdateTroubleTicketEBM

The industry enterprise business object (EBO) and EBM XSD files are located here: `http://<server name>:<port number>/AIAComponents/EnterpriseObjectLibrary/Industry/Communications/EBO/TroubleTicket/V1/`

The industry EBS 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 Oracle AIA Artifacts.”

Integration Services

These services are delivered with this integration:

- CommunicationsTroubleTicketEBSV1
- CommunicationsTroubleTicketResponseEBSV1
- CreateTroubleTicketSiebelCommsProvABCImpl
- UpdateTroubleTicketSiebelCommsProvABCImpl
- AIAOrderFalloutJMSBridgeService
- AIACOMOrderFalloutNotificationJMSConsumer
- CreateTroubleTicketAIACommsReqImpl
- CreateOrderFalloutNotificationOSMCFSCommsProvImpl
- CreateOrderFalloutNotificationOSMCFSCommsJMSConsumer
- CreateOrderFalloutNotificationOSMCFSCommsJMSProducer
- CreateTroubleTicketOSMCFSCommsJMSConsumer
- CreateTroubleTicketRespOSMCFSCommsJMSProducer
- UpdateTroubleTicketOSMCFSCommsJMSConsumer
- CreateFaultNotificationLFCommsJMSConsumer

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

For more information about using Session Pool Manager, see [Appendix B: Using Session Pool Manager](#).

Use the Integration Scenario Summary page of 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.”

The Oracle Order to Bill PIP also uses these services for fallout:

- CommunicationsTroubleTicketEBSV1
- CommunicationsTroubleTicketResponseEBSV1
- CreateTroubleTicketSiebelCommsProvABCImpl
- AIAOrderFalloutJMSBridgeService
- AIACOMOrderFalloutNotificationJMSConsumer
- CreateTroubleTicketAIACommsReqImpl

For more information about these integration services, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide*.

Note. Install sequence considerations exist for both the Oracle Order to Activate PIP and the Oracle Order to Bill PIP.

For more information, see [Chapter 8: Configuring the Oracle Order to Activate PIP](#), Configuring Order Fallout Management.

UpdateTroubleTicketSiebelCommsProvABCImpl

The UpdateTroubleTicketSiebelCommsProvABCImpl is a service that acts as the provider for the UpdateTroubleTicket operation of the CommunicationsTroubleTicketEBSV1. This service does not return any response.

This BPEL process takes the UpdateTroubleTicketEBM as input and invokes the Siebel web service SWITroubleTicket to update the trouble ticket.

Updating open trouble tickets to Closed or adding additional failed order lines to an existing open trouble ticket are the scenarios in which an update to trouble ticket request originates from Oracle OSM CFS.

CreateOrderFalloutNotificationOSMCFSCommsProvImpl

The CreateOrderFalloutNotificationOSMCFSCommsProvImpl service is a BPEL process that picks up the Fault message from CreateOrderFalloutNotificationOSMCFSCommsJMSConsumer. It parses the Fault message and then constructs the OrderFalloutNotification message.

Next, the CreateOrderFalloutNotificationOSMCFSCommsJMSProducer service is invoked to enqueue the order fallout notification message in the AIA_FALLOUT_JMSQ for Oracle OSM CFS. This action will consume and trigger a fallout event for the particular order.

CreateOrderFalloutNotificationOSMCFSCommsJMSConsumer

The CreateOrderFalloutNotificationOSMCFSCommsJMSConsumer service is an Enterprise Service Bus (ESB) service that picks up the Fault message from the AIA_ORDERFALLOUT_JMSQ. It passes the Fault message to the CreateOrderFalloutNotificationOSMCFSCommsProvImpl process based on the JMS Correlation ID.

This service acts as a consumer, listening to the messages produced in AIA_ORDERFALLOUT_JMSQ.

CreateOrderFalloutNotificationOSMCFSCommsJMSProducer

The CreateOrderFalloutNotificationOSMCFSCommsJMSProducer is a BPEL process that enqueues the OrderFalloutNotification message to the AIA_FALLOUT_JMSQ. The Oracle OSM Bridge then picks the message from this queue and triggers a fallout event in Oracle OSM. The CreateOrderFalloutNotificationOSMCFSCommsProvImpl service invokes this service.

CreateTroubleTicketOSMCFSCommsJMSConsumer

The CreateTroubleTicketOSMCFSCommsJMSConsumer is an ESB service that picks up the CreateTroubleTicketEBM message from the AIA_CRTTTREQ_JMSQ. It routes the message to the Create Trouble Ticket operation of the CommunicationsTroubleTicketEBSV1. This service acts as a consumer, listening to the messages produced in the AIA_CRTTTREQ_JMSQ.

CreateTroubleTicketRespOSMCFSCommsJMSProducer

The CreateTroubleTicketRespOSMCFSCommsJMSProducer is a BPEL process that enqueues the CreateTroubleTicketResponseEBM message to the AIA_CRTTTRESP_JMSQ. The Oracle OSM Bridge then picks up the message from this queue and then updates the order task with the created trouble ticket ID. The CommunicationsTroubleTicketResponseEBSV1 service invokes this service.

UpdateTroubleTicketOSMCFSCommsJMSConsumer

The UpdateTroubleTicketOSMCFSCommsJMSConsumer is an ESB service that picks up the UpdateTroubleTicketEBM message from the AIA_UPDTTREQ_JMSQ. It routes the message to the Update Trouble Ticket operation of the CommunicationsTroubleTicketEBSV1. This service acts as a consumer, listening to the messages produced in the AIA_UPDTTREQ_JMSQ.

CreateFaultNotificationLFCommsJMSConsumer

The CreateFaultNotificationLFCommsJMSConsumer is an ESB service that picks up the Fault message from the AIA_LFERROR_JMSQ queue. It routes this message to the BPEL service AIAAsyncErrorHandlingBPELProcess, which queues the message in the AIA_ERROR_TOPIC.

Line Fulfillment (provisioning) systems that want to notify the central fulfillment system about an error in processing the order will create an enriched fault message (fault message with order details). This enriched fault message is queued to the AIA_LFERROR_JMSQ in order to get the fault processed by the order fallout framework. This fault message is processed by the Order Fallout Management framework, and Oracle OSM CFS is notified about the errors in the Line Fulfillment system for a particular order.

This service acts as a consumer, listening to the messages produced in the AIA_LFERROR_JMSQ.

Configuring Order Fallout Using Oracle Order to Bill

For information about using the Order to Bill PIP for order fallout, see [Chapter 8: Configuring the Oracle Order to Activate PIP](#), Using Error Type to Control Response to Order Fallout.

Order to Activate Fallout Services

These Order to Activate services are fallout-enabled:

1. UpdateSalesOrderSiebelCommsProvABCImpl
2. ProcessSalesOrderFulfillmentSiebelCommsReqABCImpl
3. ProcessSalesOrderFulfillmentOSMCFSCCommsJMSProducer
4. ProcessFOBillingAccountListRespOSMCFSCCommsJMSProducer
5. ProcessFulfillmentOrderBillingResponseOSMCFSCCommsJMSProducer
6. TestOrderOrchestrationEBF
7. EBS.CommunicationsBillingEBSV1
 default.ProcessFulfillmentOrderBillingBRMCommsProvABCImpl.ProcessFulfillmentOrderBillingBRMCommsProvABCImpl_1_0

 default.CommsProcessBillingAccountListEBF.CommsProcessBillingAccountListEBF_1_0
8. EBS.CommunicationsBillingResponseEBSV1
 default.ProcessFulfillmentOrderBillingResponseOSMCFSCCommsJMSProducer.ProcessFulfillmentOrderBillingResponseOSMCFSCCommsJMSProducer_1_0

 default.CommsProcessFulfillmentOrderBillingAccountListEBF.CommsProcessFulfillmentOrderBillingAccountListEBF_1_0

 default.ProcessFOBillingAccountListRespOSMCFSCCommsJMSProducer.ProcessFOBillingAccountListRespOSMCFSCCommsJMSProducer_1_0
9. EBS.CommunicationsSalesOrderEBSV2
 default.UpdateSalesOrderSiebelCommsProvABCImpl.UpdateSalesOrderSiebelCommsProvABCImpl_1_0

 default.ProcessSalesOrderFulfillmentOSMCFSCCommsJMSProducer.ProcessSalesOrderFulfillmentOSMCFSCCommsJMSProducer_1_0
10. Siebel.ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer
11. Siebel.ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer_RS

default.ProcessSalesOrderFulfillmentSiebelCommsReqABCImpl.ProcessSalesOrderFulfillmentSiebelCommsReqABCImpl_1_0

12. OSM.ABCS.ConsumeCustomerCFO_JMSAdapter
 13. OSM.ABCS.ConsumeCustomerCFO_JMSAdapter_RS
 14. OSM.ABCS.ConsumeBillingCFO_JMSAdapter
 15. OSM.ABCS.ConsumeBillingCFO_JMSAdapter_RS
 16. OSM.ABCS.ConsumeUpdateFulfillmentOrder_JMSAdapter_RS
 17. OSM.ABCS.ConsumeUpdateFulfillmentOrder_JMSAdapter
 18. OSM.ABCS.BillingResponseConsumer
 19. OSM.ABCS.BillingResponseConsumer_RS
 20. OSM.ABCS.CustomerResponseConsumer
 21. OSM.ABCS.CustomerResponseConsumer_RS
 22. OSM.ABCS.OrderOrchestrationConsumer
 23. OSM.ABCS.OrderOrchestrationConsumer_RS
- default.TestOrderOrchestrationEBF.TestOrderOrchestrationEBF_1_0
24. Siebel.ProcessSalesOrderFulfillmentSiebelCommsSequencer
 25. ProcessProvisioningOrderOSMPROVCommsJMSProducer
 26. ProcessFulfillmentOrderUpdateOSMCFSCCommsJMSProducer
 27. OSMPROV
 28. OSM.ABCS.ConsumeProvUpdate_RS
 29. OSM.ABCS.ConsumeProvUpdate
 30. OSM.ABCS.Consume_ProcessProvOrder
 31. OSM.ABCS.Consume_ProcessProvOrder_RS
 32. EBS.CommunicationsProvisioningOrderEBSV1
 33. EBS.CommunicationsFulfillmentOrderEBSV1

For more information about Order to Bill services enabled for fallout, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide*, “Order to Bill Fallout Services”.

Chapter 8: Configuring the Oracle Order to Activate PIP

Chapter 8: Configuring the Oracle Order to Activate PIP

This chapter discusses how to:

- Set up Siebel Customer Relationship Management (Siebel CRM).
- Set up Oracle Order and Service Management (Oracle OSM).
- Set up Fusion Middleware (FMW).
- Guarantee delivery.
- Use the Enterprise Service Bus (ESB) resequencer feature.
- Work with domain value maps (DVMs).
- Work with cross-references.
- Handle errors.
- View enterprise business object (EBO) implementation maps.
- Configure multiple Oracle Billing and Revenue Management (Oracle BRM) instances.
- Configure the Oracle Order to Activate Process Integration Pack (PIP).

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 the SISOMBillingSubmitOrderWebService Siebel outbound workflow to enqueue the Siebel messages in AIA_SALESORDERJMSQUEUE.

For more information about the 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 Application Integration Architecture (Oracle AIA)-side configuration, see the *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Constructing JMS Adapter Services."

- For updating the order information from your central fulfillment system (CFS) to Siebel CRM, set up the SWIOrderUpsert Siebel inbound web service. It should be enabled.

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

As delivered, Siebel provides and maps these priority values:

Order Priority	JMS Priority
Low	3
Medium	5
High	7
Urgent	9

Customers need to manually change the order priority values that are seeded when the application is delivered.

For more information about changing the order priority values, see the *Siebel Order Management Guide Addendum for Communications*, "Modifying the Order Priority Mapping."

Customers can add other values in Siebel. Additionally, customers can use the SWI_ORDER_JMS_PRIORITY mapping, which maps these string values to integers.

For more information, see the Siebel product documentation.

Setting Up Oracle OSM

Perform these setups:

- The AIAConfigurationProperties.xml is populated with all Oracle BRM, Siebel, and Oracle OSM instance names and the respective IDs. These IDs are also populated in the Oracle OSM cartridges. Oracle OSM needs to synchronize these IDs as part of its setup to use the Oracle BRM and Siebel instances. Oracle OSM defines the instances in the cartridge with seeded data and enables a user with administrator privileges the ability to alter the logical names using the Service Creation Environment (SCE) design-time tool.

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

- For all Oracle OSM bridges, ensure that the Preserve Msg property is set to True.

For more information about bridges in Oracle OSM, see the *Oracle Communications Order and Service Management, System Administration Guide*.

For more information about JMS bridges concepts and how to configure JMS bridges, see the WebLogic documentation.

Setting Up Fusion Middleware

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 TCP connection more than once to call a Siebel web service. Simultaneous calls to the same Siebel web service will cause errors if the same TCP connection is used. You should add this property in the startup parameters of `oc4j_soa`.

After you add the property, the `opmn.xml` file will look like this:

```
<process-type id="oc4j_soa" module-id="OC4J" status="d">
  <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=$ORACLE_HOME/j2ee/oc4j_soa/config/java2.policy
-Djava.awt.headless=true -Dhttp.webdir.=false -
Doraesb.home=/slot/ems1936/oracle/product/10.1.3.1/OracleAS_1/integr
ation/esb -Dhttp.proxySet=false -Doc4j.userThreads=true -
Doracle.mdb.fastUndeploy=60 -Doc4j.formauth.redirect=true -
Djava.net.preferIPv4Stack=true -
Dorabpel.home=/slot/ems1936/oracle/product/10.1.3.1/OracleAS_1/bpel
-
Xbootclasspath^/p:/slot/ems1936/oracle/product/10.1.3.1/OracleAS_1/b
pel/lib/orabpel-boot.jar -Dhttp.proxySet=false -
Daia.home=/slot/ems1936/oracle/product/AIA_HOME" -
DHTTPClient.disableKeepAlives=true/>
    </category>
    <category id="stop-parameters">
      <data id="java-options" value="-
Djava.security.policy=$ORACLE_HOME/j2ee/oc4j_soa/config/java2.policy
-Djava.awt.headless=true -Dhttp.webdir.=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" />
</process-type>
```

```
<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 crash the server.

Guaranteeing Delivery

When message submission is decoupled from message processing, we recommend the use of queues. When guaranteed delivery of a message is a requirement, we recommend the use of Oracle Advanced Queuing (AQ) for persistence. For all Order Lifecycle Management process integration flows, two sets of queues are used: one set on the Oracle AIA side and one set on the WebLogic side.

The participating application (Oracle OSM) does not have the ability to use JMS adapters, so a WebLogic JMS Bridge is used to guarantee delivery.

For more information about guaranteed delivery, see My Oracle Support document 1077055.1 and the *Oracle Application Integration Architecture - Foundation Pack 2.5: Integration Developer's Guide*, "Designing and Constructing ABC Services," Interacting with Participating Applications.

For more information about bridges in Oracle OSM, see the *Oracle Communications Order and Service Management, System Administration Guide*.

For more information about JMS bridges concepts and how to configure JMS bridges, see the WebLogic documentation.

Using the ESB Resequencer Feature

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:

Oracle AIA Queue	Flow	JMS Priority Set By	Sequencing Criteria
AIA_SALESORDERJMSQ	Order flow from Siebel CRM to Oracle AIA.	Siebel	Group By: Billing Account on Order Header (/ListOfSWIOrderIO/SWIOrder/BillingAccountId) Order of Processing: FIFO (First in First Out). ESB Consumer: ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer.
AIA_CRTFO_IN_JMSQ	Order flow from Oracle AIA to OSM	ProcessSalesOrderFulfillmentOSMCFSCCommsJMSProducer	None. (Onus is on OSM.)
AIA_CRTBO_OUT_JMSQ	Order flow from OSM to AIA for billing.	OSM)	None as delivered. Customers can use ProcessFulfillmentOrderBillingOSMCFSCCommsJMSConsumer to implement custom sequencing.
AIA_UPDBO_IN_JMSQ	Order flow from AIA (from billing) to OSM	ProcessFulfillmentOrderBillingResponseOSMCFSCCommsJMSProducer	None. (Onus is on OSM.)
AIA_UPDCUST_IN_JMSQ	Response of the customer creation in billing from AIA to OSM	ProcessFOBillingAccountListRespOSMCFSCCommsJMSProducer	None. (Onus is on OSM.)
AIA_CRTFO_OUT_JMSQ	Create Fulfillment Order flow from OSM to Oracle AIA for the provisioning system	OSM	None as delivered. Customer can use ProcessProvisioningOrderOSMCFSCCommsJMSConsumer to implement custom sequencing.

Oracle AIA Queue	Flow	JMS Priority Set By	Sequencing Criteria
AIA_FOCFS_IN_JMSQ	Update Fulfillment Order flow from Oracle AIA (from the provisioning system) to OSM)	ProcessFulfillmentOrderUpdateOSMCFSCommsJMSProducer	None. (Onus is on OSM.)
AIA_FOPROV_OUT_JMSQ	Update Fulfillment Order flow from the provisioning system to Oracle AIA (for OSM)	Provisioning System	None as delivered. Customer can use ProcessFulfillmentOrderUpdateOSMPCOMMCommsJMConsumer to implement custom sequencing.
AIA_FOPROV_IN_JMSQ	Create Fulfillment Order from Oracle AIA (from OSM) to the provisioning system.	ProcessProvisioningOrderOSMPCOMMCommsJMSProducer	None. (Onus is on OSM.)
AIA_UPDSO_OUT_JMSQ	Update order flow from OSM to Oracle AIA for Siebel CRM system.	Not Set	<p>Group By: Account ID mentioned in the ObjectCrossReference section of the update message(/UpdateSalesOrderEBM/EBM Header/Sender/ObjectCrossReference/SenderObjectIdentification/AlternateObjectKey/ID[@schemeID = 'CUSTOMERPARTY_ACCOUNTID' and @schemeAgencyID = 'COMMON'])</p> <p>Order of Processing: FIFO (First in First Out).</p> <p>ESB Consumer: UpdateSalesOrderOSMCFSCommsJMConsumer.</p>

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.

An error may occur in the update order Siebel provider after it is consumed by UpdateSalesOrderOSMCFSCCommsJMSConsumer 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 how to retry messages from resequencer and how to unblock the resequencer, see Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite and Oracle Business Process Management Suite, "Resequencing Messages", Monitoring Resequenced Messages.

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, 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 runtime.

DVM types are seeded for the Oracle Order to Activate 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.

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.

This table describes the DVMs for Oracle Order to Activate:

DVM	Description
SALESORDER_DYNAMICPRICEIND	Dynamic Pricing Indicator
SALESORDER_FULFILLCOMPOSITIONTYPE	Fulfillment Composition Type Code
SALESORDER_FULFILLMENTMODECODE	Fulfillment Mode Code

DVM	Description
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_REVISIONPERMISSIBLECODE	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	Billing Type Code. Examples of values are 'Subscription', 'Discount', 'Item', 'Special Rating' and so on.
SALESORDER_CHANGEDIND	Order Changed Indicator. Values are 'True' or 'False'. Used to validate the OrderChangedIndicator attribute. For example, OSM sets 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_REVISIONPERMISSIBLECODE	Revision Permissible Code
SALESORDER_LINESTATUS	Order line Status
DISCOUNT_METHODCODE	Discount Method Code
CURRENCY_CODE	Currency Code
PRICE_TYPE	Price Type

DVM	Description
TROUBLETICKET_AREA	Maps the Area of the trouble ticket SEBL_01 column to the Area element in Siebel. The COMMON column points to the SystemCode column of the corresponding system on the BSR Systems page.
TROUBLETICKET_SUBAREA	Maps the SubArea of the trouble ticket. SEBL_01 column to the Sub-Area element in Siebel. The COMMON column points to the appropriate FailureSubSystemCode or the faulting service.
TROUBLETICKET_STATUS	Maps the status of the trouble ticket SEBL_01 column to the Status element in Siebel. The COMMON column maps to the appropriate status in Oracle AIA.
TROUBLETICKET_SEVERITY	Maps the severity of the trouble ticket. SEBL_01 column to the Severity element in Siebel. The COMMON column maps to the appropriate severity (1-5) in Oracle AIA.
TROUBLETICKET_PRIORITY	Maps the recovery priority of the trouble ticket. SEBL_01 column to the Priority element in Siebel. The COMMON column maps to the appropriate priority (1-4) in Oracle AIA.

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 an application network, and they use 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 and describes the cross-references:

Cross-Reference Table Name	Column Names: Column Values		Description
CUSTOMERPARTY_PAYPROFIL EID	COMMON	SEBL_01	Maps Siebel Bill Profile ID one-to-one to the Oracle BRM PayInfoID.
	Pay info ID	Pay info ID	
INSTALLEDPRODUCT_ID	COMMON	SEBL_01	Maps AssetID information from Siebel to Oracle BRM. (Service Instance ID, Purchased Product ID, Purchased Discount ID)
	Asset ID	Asset ID	
ITEM_ITEMID	COMMON	SEBL_01	Maps Item ID (Product ID) information between Siebel and Oracle BRM.
	Item ID	Item ID	
SALESORDER_LINEID	COMMON	SEBL_01	Maps the Siebel Line ID to Common.
	Order Line ID	Order Line ID	
SALESORDER_ID	COMMON	SEBL_01	Maps the Siebel ID to Common.
	Siebel Order ID	Siebel Order ID	
CUSTOMERPARTY_CONTACTID	COMMON	SEBL_01	Maps Siebel Contact ID to Oracle BRM Contact POID.
	Contact ID	Contact ID	
CUSTOMERPARTY_ACCOUNTID	COMMON	SEBL_01	Maps Siebel AccountID one-to-one to the Common AccountID.
	Account ID	Account ID	
CUSTOMERPARTY_BILLPROFIL EID	COMMON	SEBL_01	Maps Siebel Bill Profile ID one-to-one to Portal BillInfo ID.
	Bill Profile ID	Bill Profile ID	

Cross-Reference Table Name	Column Names: Column Values		Description
TROUBLETICKET_TROUBLETICKETID	COMMON	SEBL_01	Cross-references Siebel trouble ticket ID to the BusinessComponentID of the TroubleTicket Response EBM.
	Trouble TicketID	Siebel Trouble Ticket ID	

Handling Errors

Based on the roles defined for the services, the system sends email notifications if a service errors out.

This table describes localized custom errors thrown by the order fallout management services for insufficient data when creating a trouble ticket:

Error Code	Message Text
AIA_ERR_AIACOMOFMPI_0001	Data Insufficient for Trouble Ticket Creation. Order Originating System Code not available.
AIA_ERR_AIACOMOFMPI_0002	Data Insufficient for Trouble Ticket Creation. Order ID not available.

For more information about the errors thrown by participating applications, 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*.

Viewing EBO EIMs

For more information about how to map services, see the My Oracle Support document: EBO Implementation Maps (EIMs) 795541.1.

Configuring Multiple BRM Instances

For more information about configuring multiple instances of Oracle BRM, see the appendix in the *Siebel CRM Integration Pack for Oracle Communications BRM: Order to Bill Implementation Guide*, “Configuring Multiple Instances of Oracle BRM.”

Configuring the Oracle Order to Activate PIP

Configure the properties listed in the following table in the `AIAConfigurationProperties.xml` file. The file is located in `<aia.home>/config/`. Entries in the `AIAConfigurationProperties.xml` file are case-sensitive.

Note: Whenever you update the `AIAConfigurationProperties.xml` file, you must reload the file 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 reload the file 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.”

This table provides the settings for the Oracle Order to Activate PIP. Oracle Order to Activate has properties at the module level. All of the Oracle Order to Activate services use the same properties:

Property Name	Value/Default Values	Description
OSMCFS.USERNAME	Default = osm	Username of the CFS user who has the authorization to create and modify an order in Oracle OSM CFS.
OSMCFS.PASSWORD	Default = osmadmin	Password of the CFS user who has the authorization to create and modify an order in Oracle OSM CFS.
OSMLF.USERNAME	Default = osm	Username of the Local Fulfillment (LF) user who has the authorization to create and modify an order in LF.
OSMLF.PASSWORD	Default = osmadmin	Password of the LF user who has the authorization to create and modify an order in LF.

Chapter 9: Configuring Order Fallout Management

This chapter discusses how to:

- Configure properties for Order Fallout services
- Install sequence and configurations for using Fallout
- Use error type to control response to Order Fallout

Configuring Properties for Order Fallout Services

Configure the properties for these services in the AIAConfigurationProperties.xml file.

- CreateTroubleTicketAIACommsReqImpl
- CreateTroubleTicketSiebelCommsProvABCImpl
- UpdateTroubleTicketSiebelCommsProvABCImpl
- CreateOrderFalloutNotificationOSMCFSCommsProvImpl

It is located here: <aia.home>/config/. All the property values are case-sensitive. All Boolean values are in lowercase.

Note: Whenever you update the AIAConfigurationProperties.xml file, you must reload the file 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 reload the file 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/ABCImpl/AIA/Industry/Comms/CreateTroubleTicketAIACommsReqImpl/V1}CreateTroubleTicketAIACommsReqImpl

Property Name	Value/Default Values	Description
Sender.Default. SystemID	COMMON	Use this only if the request message does not contain the system instance ID. This value is always COMMON because this service is triggered by oracle AIA.
Routing.TroubleTicketEBSV1.Create	true/false. Default = false.	Controls whether TroubleTicketEBS routes messages

Property Name	Value/Default Values	Description
TroubleTicket.RouteToCAVS		to the validation system or to the Provider ABCS implementation.
Routing.TroubleTicketEBSV1.CreateTroubleTicket.CAVS.EndpointURI	http://{http.hostname}:{http.port}/AIAValidationSystemServlet/asyncrequestrecipient	The endpoint URI of the CAVS simulator.
Routing.TroubleTicketEBSV1.CreateTroubleTicket.MessageProcessingInstruction.EnvironmentCode	CAVS/PRODUCTION / Any Value Default: PRODUCTION	If CAVS, then the message is routed to CAVS. For other values, the message is routed to the Provider ABCS Implementation.
TroubleTicket.DefaultSeverity	Any number from 1 to 5. Default = 2.	If the fault message does not have any severity associated with it, the default severity is assigned to the fault message and reflects the same in the trouble ticket.
TroubleTicket.DefaultPriority	Any number from 1 to 4. Default = 2.	This service assigns the recovery priority for the trouble ticket by default to the value specified in this configuration property.

Here are the settings for the following CreateTroubleTicketSiebelCommsProvABCServiceImpl service name:

{http://xmlns.oracle.com/ABCServiceImpl/Siebel/Industry/Comms/CreateTroubleTicketSiebelCommsProvABCServiceImpl/V1}CreateTroubleTicketSiebelCommsProvABCServiceImpl

Property Name	Value/Default Values	Description
Default.SystemID	SEBL_01	Siebel system instance code (defined in BSR). 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 Default: false	Value determines whether the ABCS should invoke the Extension service (before the EBM to ABM transformation).
ABCSExtension.PostXformABMtoEBMTroubleTicketEBM	true/false Default: false	Value determines whether the ABCS should invoke the Extension service (after the ABM to EBM transformation).
ABCSExtension.PreInvokeABSSWITroubleTicketIOABM	true/false Default: false	Value determines whether the ABCS should invoke the Extension service (PreInvoke Application).
ABCSExtension.PostInvokeABSSWITroubleTicketIOABM	true/false Default: false	Value determines whether the ABCS should invoke the Extension service (PostInvoke Application).
Routing.SWI_spcTrouble_spcTicket_spcService.RouteToCAVS	true/false 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/asyncrequestrecipient	Endpoint URI of the CAVS simulator for this partner link - SWI_spcTrouble_spcTicket_spcService.

Property Name	Value/Default Values	Description
	mServlet/asyncreponsesimulator	
Routing.SWI_spcTrouble_spcTicket_spcService.SEBL_01.EndpointURI	Endpoint URI of the SEBL_01 Siebel instance	Endpoint 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 Default: false	CreateTroubleTicketSiebelCommsProvABCSEBSImpl 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 response.
TroubleTicket.UseDefaultInstance	true/false Default: false	If set to true, overwrites the target Siebel instance to the default instance indicated by the property Default.SystemID. Gives the user an option to create a trouble ticket in a Siebel instance different from the one where the order was placed.
TroubleTicket.SR_TYPE	Order Failure	SR_TYPE identifies 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 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/asyncreponseserecipient	Endpoint URI for the CAVS simulator.
Routing.TroubleTicketEBSResponseV1.CreateTroubleTicketEBSResponse.MessageProcessingInstruction.EnvironmentCode	CAVS/PRODUCTION / Any Value Default : PRODUCTION	If CAVS, routes the message to CAVS. For other values, routes the message to the target service.

Here are the settings for the following UpdateTroubleTicketSiebelCommsProvABCSEBSImpl service name:

{http://xmlns.oracle.com/ABCSEBSImpl/Siebel/Industry/Comms/UpdateTroubleTicketSiebelCommsProvABCSEBSImpl/V1}UpdateTroubleTicketSiebelCommsProvABCSEBSImpl

Property Name	Value/Default Values	Description
Default.SystemID	SEBL_01	Siebel system instance code (defined in BSR). Used when the target system cannot be identified from the

Property Name	Value/Default Values	Description
		request message or if the configuration property TroubleTicket.UseDefaultInstance is set to true.
ABCSExtension.PreXformEBMtoABMTroubleTicketEBM	true/false Default: false	Value determines whether the ABCS should invoke the Extension service (before the EBM to ABM transformation).
ABCSExtension.PreInvokeABSSWITroubleTicketIOABM	true/false Default: false	Value determines whether the ABCS should invoke the Extension service (PreInvoke Application).
Routing.SWI_spcTrouble_spcTicket_spcService.RouteToCAVS	true/false 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/asyncrequestrecipient	Endpoint URI of the CAVS simulator for this partner link - SWI_spcTrouble_spcTicket_spcService.
Routing.SWI_spcTrouble_spcTicket_spcService.SEBL_01.EndpointURI	Endpoint URI of the SEBL_01 Siebel instance	Endpoint 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.

Here are the settings for the following CreateOrderFalloutNotificationOSMCFSCCommsProvImpl service name:

{http://xmlns.oracle.com/ABCSExtension/OSMCFSC/Industry/Comms/CreateOrderFalloutNotificationOSMCFSCCommsProvImpl/V1}CreateOrderFalloutNotificationOSMCFSCCommsProvImpl

Property Name	Value/Default Values	Description
Default.SystemID	OSMCFSC_01	Siebel system instance code (defined in BSR). Used when the target system cannot be identified from the request message or if the configuration property TroubleTicket.UseDefaultInstance is set to true.
Routing.CreateOrderFalloutNotification.RouteToCAVS	true/false Default: false	Indicates whether the Partner Link CreateOrderFalloutNotification should be routed to CAVS or the JMS Producer service.
Routing.CreateOrderFalloutNotification.CAVS.EndpointURI	http://\${http.hostname}:\${http.port}/AIAValidationSystemServlet/asyncrequestrecipient	Endpoint URI of the CAVS instance.
Routing.CreateOrderFalloutNotification.OSMCFSC_01.EndpointURI	http://\${http.hostname}:\${http.port}/orabpel/default/CreateOrderFalloutNotificationOSMCFSCCommsJMSProducer/1.0	Endpoint URI of the actual target service.
Routing.CreateOrderFalloutNotification	CAVS/PRODUCTION/Any	Acts as a reference and is not used in the service.

Property Name	Value/Default Values	Description
n.MessageProcessingInstruction.EnvironmentCode	Value	
Fault.DefaultSeverity	Any number from 1 to 5 Default – 2	If the fault message has no severity associated with it, assigns the default severity to the fault message and reflects the same in the trouble ticket.

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}**

Installing Sequence and Configurations for Using Fallout

Two scenarios for the install sequence affect whether the integration will use order fallout management functionality from Oracle Order to Activate or from Oracle Order to Bill:

1. If the Oracle Order to Activate PIP is installed after the Oracle Order to Bill PIP, then CommunicationsTroubleTicketEBSV1 is not redeployed to preserve any customizations that were made.
2. If the Oracle Order to Activate PIP is installed before the Oracle Order to Bill PIP, then both of the required routing rules are present.

The CommunicationsTroubleTicketEBSV1 has the following routing rules:

- As a part of the Oracle Order to Bill PIP, only one routing rule is delivered for the CreateTroubleTicket operation.
- As a part of the Oracle Order to Activate PIP, two routing rules are delivered: one for the CreateTroubleTicket operation and one for the UpdateTroubleTicket operation.

For more information about using order fallout in Oracle OSM, see the *Oracle Communications Order and Service Management, Application Integration Architecture Order to Activate Cartridge Guide*, “Overview of the Order to Activate PIP,” Fallout Management.

For more information about how the Oracle Order to Bill PIP uses order fallout, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide*.

Adding a Routing Rule for Oracle Order to Activate

This section describes the procedure to add the necessary routing rule for the UpdateTroubleTicket operation required in the Oracle Order to Activate PIP.

Case 1: No Customizations Made to CommunicationsTroubleTicketESV1

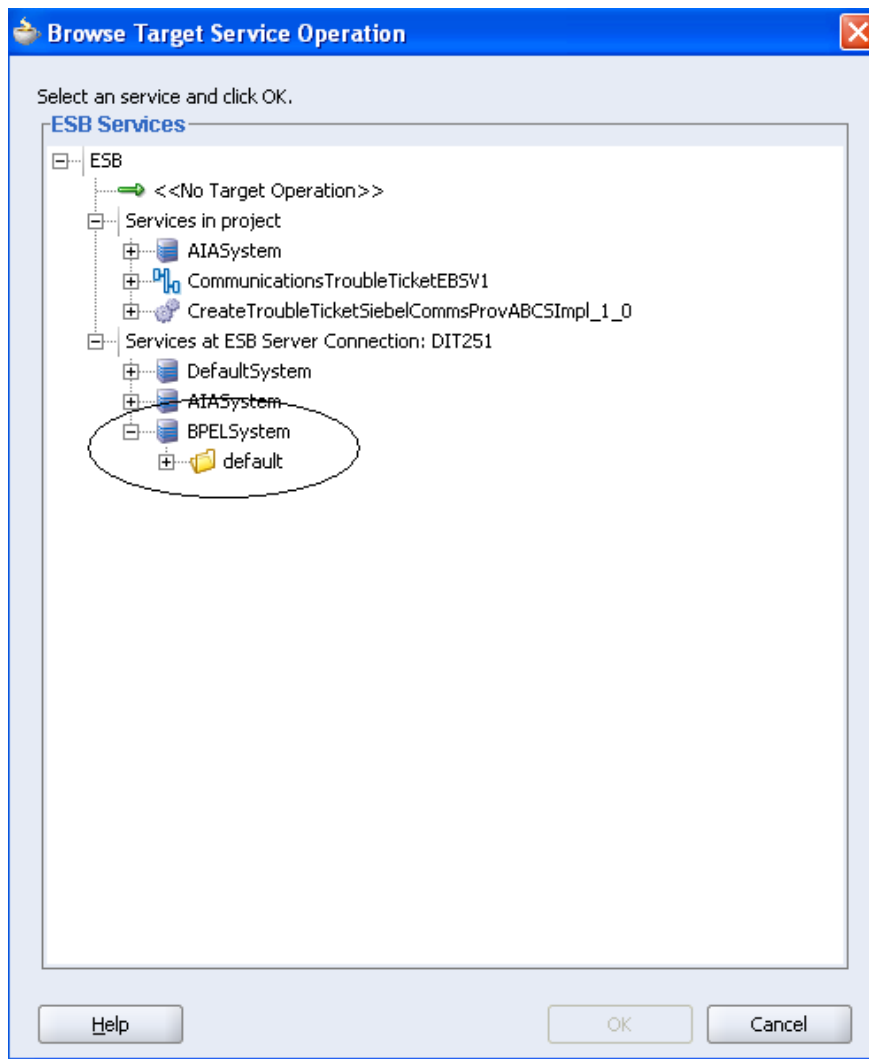
After installing the Oracle Order to Activate PIP, just deploying the EBS will ensure that the required routing rules are present.

Steps to redeploy the CommunicationsTroubleTicketESV1 for Oracle Order to Activate are:

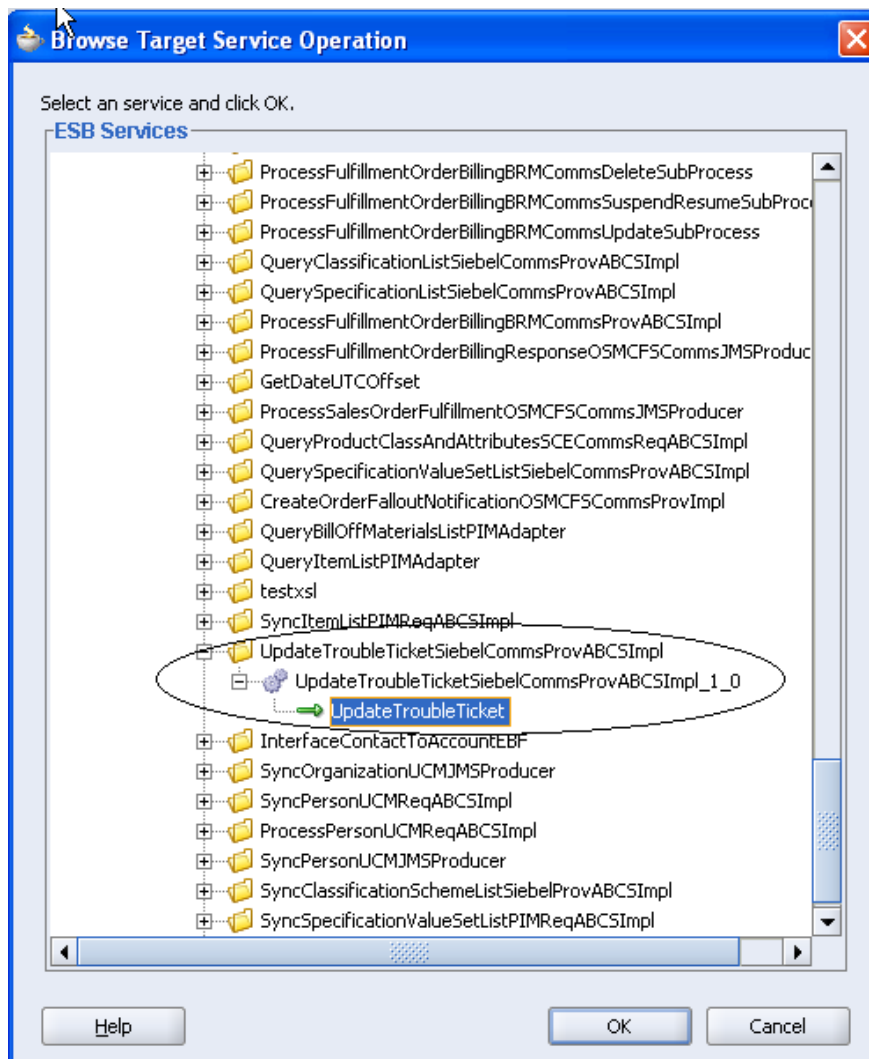
1. Source the AIA environment variables by executing the aiaenv.bat or aiaenv.sh files located in \$AIA_HOME/bin.
2. Navigate to \$AIA_HOME/PIPS/Industry/Communications/Setup/OrderToActivate/EBS/OFM/CommunicationsTroubleTicketESV1.
3. Run ant.

Case 2: Customizations Made to CommunicationsTroubleTicketESV1

1. Open the customized EBS project in Oracle JDeveloper.
2. Open the esbsvc file of the EBS (AIASystem_EBS_CommunicationsTroubleTicketESV1.esbsvc) and expand the routing rules.
3. Add a routing rule to route the messages to the UpdateTroubleTicketSiebelCommsProvABCSImpl service.
 - a. Click the expand icon (+) to create a routing rule for the UpdateTroubleTicket operation.
 - b. In “Browse Target Service Operation,” expand the BPELSystem -> default Service Group under “Services at ESB Server Connection : <Default Integration Server>”. The Oracle Order to Activate PIP should be installed on the Default Integration Server.



- c. In the default folder, select the UpdateTroubleTicket operation under the UpdateTroubleTicketSiebelCommsProvABCImpl BPEL Service.



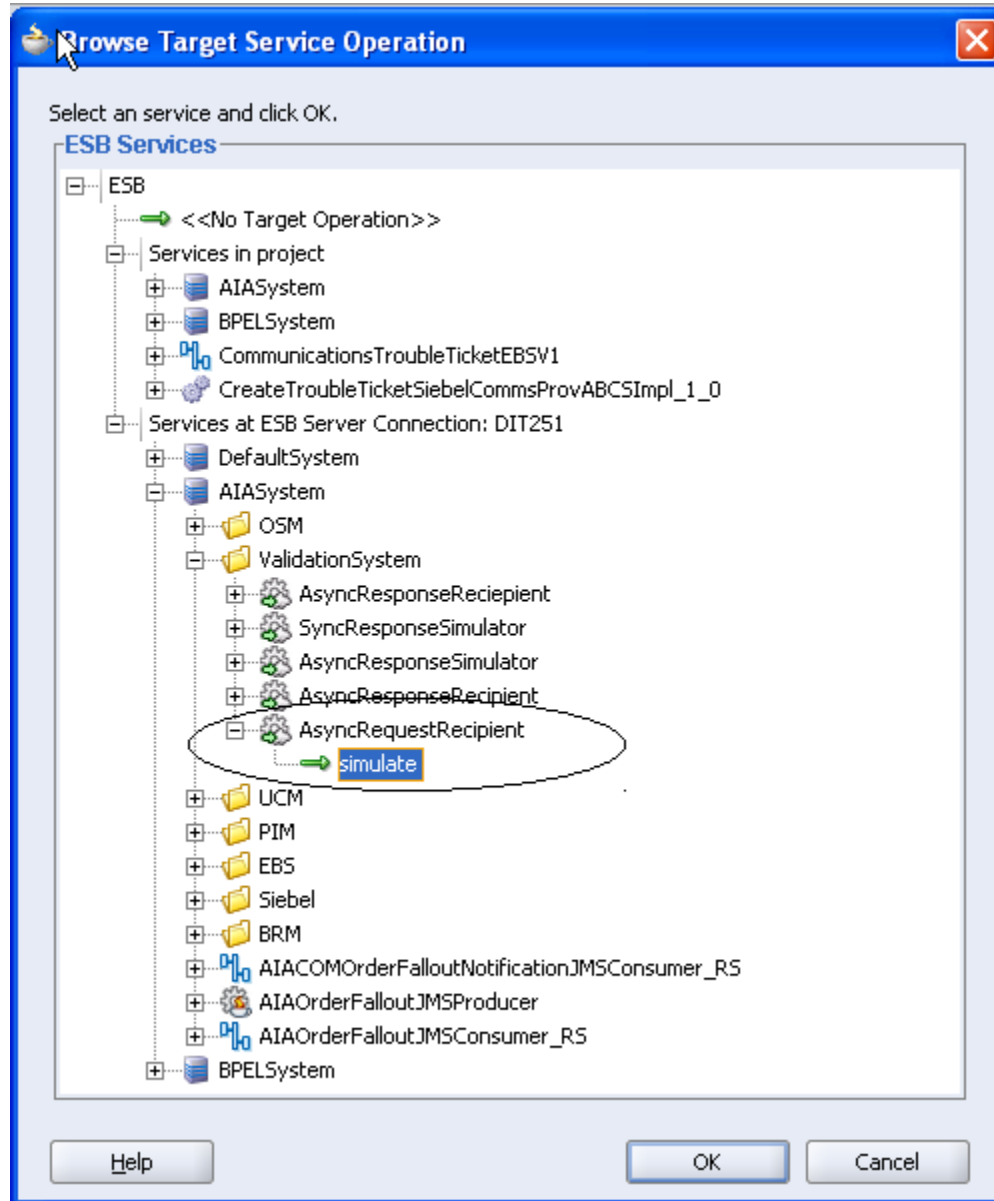
You have added the routing rule to the ESB service.

- d. In the <<Filter Expression>> box, add the following filter expression, which checks if the message is not to be routed to CAVS:

```
{count (/ebo:UpdateTroubleTicketEBM/corecom:EBMHeader/corecom:Message
ProcessingInstruction/corecom:EnvironmentCode[text()='CAVS'])=0};{
namespace
corecom=http://xmlns.oracle.com/EnterpriseObjects/Core/Common/V2
namespace
ebo=http://xmlns.oracle.com/EnterpriseObjects/Core/EBO/TroubleTicket
/V1 }
```

- e. Save the file.
4. Add a routing rule for CAVS for the UpdateTroubleTicket operation.
 - a. Click the expand icon (+) to create a routing rule for the UpdateTroubleTicket operation.

- b. In “Browse Target Service Operation,” expand the AIASystem -> ValidationSystem Service Group under the “Services at ESB Server Connection : <Default Integration Server>”. The Oracle Order to Activate PIP should be installed on the Default Integration Server.
- c. Select the ‘simulate’ operation of the AsyncRequestRecipient service:

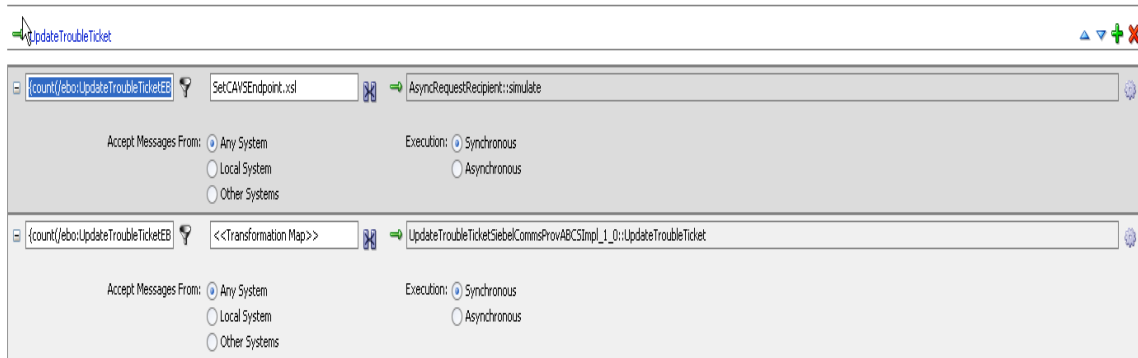


- d. In the Transformation Map box, select the SAVSEndpoint.xsl available in the project.
- e. Add the following filter expression:

```
{count(/ebo:UpdateTroubleTicketEBM/corecom:EBMHeader/corecom:Message
ProcessingInstruction/corecom:EnvironmentCode[text()='CAVS'])!=0};{
namespace
corecom=http://xmlns.oracle.com/EnterpriseObjects/Core/Common/V2
```

```
namespace
ebo=http://xmlns.oracle.com/EnterpriseObjects/Core/EO/TroubleTicket
/V1 }
```

The UpdateTroubleTicket operation should look like this example after you add the two routing rules:



- f. Save the file.
- g. Register this ESB project.

For more information about using order fallout in Oracle OSM, see the *Oracle Communications Order and Service Management, Application Integration Architecture Order to Activate Cartridge Guide*, “Overview of the Order to Activate PIP,” Fallout Management.

For more information about how the Oracle Order to Bill PIP uses order fallout, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide*.

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
	ProcessFulfillmentOrderBillingBRMCommsAddSubProcess	AIA_EH_DEFAULT	AIACOM_OFM_EXT
{http://schemas.oracle.com/bpel/extension}bindingFault	ProcessFulfillmentOrderBillingBRMCommsAddSubProcess	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 directs 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 using order fallout with the Oracle Order to Activate PIP, see [Chapter 7: Using Order Fallout Management in Oracle Order to Activate](#).

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*, “Understanding Error Handling and Logging.”

For more information about using order fallout with the Oracle Order to Bill PIP, see the *Siebel CRM Integration Pack for Oracle Communications Billing and Revenue Management: Order to Bill Implementation Guide*, "Understanding the Process Integration for Order Fallout Management."

Appendix A: Communications Orders Dictionary

Appendix A: Communications Orders Dictionary

The following tables provide a snapshot of the Communications Orders Dictionary at the time this appendix was created. Communications Orders include enterprise business objects (EBOs) for Sales Order, Fulfillment Order, and Provisioning Order. We refer to any of the three orders using the token <CommsOrder>.

To understand the tables that follow, you must be familiar with the terms defined in this table:

Term	Definition
Assetable	Indicates if an attribute value is saved to the corresponding asset in Siebel. An asset here refers to purchased product offering instance by a customer.
Prior Value	Indicates if, when the attribute changes, a prior value is also sent on the order message. Prior values sometimes are used to determine if a change occurred and sometimes used to roll back changes.
OM	Order Management
CRM	Siebel Customer Relationship Management

Note: Oracle Application Integration Architecture (Oracle AIA) EBOs may present more attributes than used by one business process or application because they cross industries and are built as part of the Foundation Pack. Therefore, the listing of an attribute in one of the following tables does not mean that the attribute is used and the corresponding feature is available. The supported features are listed in the Oracle AIA Process Integration Packs (PIPs) and in documentation and collateral for the participating applications. Although the remarks against some attributes indicate lack of support for some attributes, they are not a complete account of uptake of these attribute across different applications.

Communications Orders - Order Header

Component Attributes

Functional Attribute Name	Attribute Usage Semantics	Seeded Values	Asset-able	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows: <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Order ID	Uniquely identifies each order.	NA	No	None	Produces a unique identifier for all orders, including revision orders. Unlike Order Number, Order ID is different for revisions of the same base order. Used by AIA for cross-reference.	SaleOrderEBO/Identification/BusinessComponentID
Order Number	Identifies an order across revisions.	NA	No	None	A revision number >1 does not necessarily mean that this is a revision order from OM Fulfillment. You can create an order in Siebel and revise it several times before submitting it. If an Order Number matches an already in-flight order, then the order is treated as a revision order. When an order is revised, this number stays the same. OM uses this number to identify the base order. If the same order number with the same revision is submitted, then OM would reject the revision order and place it in fallout.	<CommsOrder>EBO/Identification/ID

Functional Attribute Name	Attribute Usage Semantics	Seeded Values	Asset-able	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows: <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Revision	A revision sequence number that, together with the order number, represents the user key to an order.	NA	No	None	If an order is received with an Order Number equal to that of an in-flight order and the newly received order has a higher revision number, then OM assumes the order is a revision order and proceeds to analyze the Order Lines. If the revision number is equal or lower than that of the base order, the revision is rejected.	<CommsOrder>EBO/Identification/Revision/Number
Success Dependency	Declares if all order lines must fulfill successfully or else the whole order fails (all or none). When the order level Success Dependency is set to All or None, it takes precedence over Order Line Success Dependency designations because it is more restrictive.	DEFAULT ALL OR NONE	No	None		<CommsOrder>EBO/PartialFulfillmentAllowedIndicator

Functional Attribute Name	Attribute Usage Semantics	Seeded Values	Asset-able	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows: <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Fulfillment Mode	Qualifies the nature of fulfillment request.	Deliver, Qualify, Cancel, Initiate billing, Fulfill billing	No	None	<p>CSPs may extend support to other modes, such as Design, Schedule and Cost.</p> <p>CRM can cancel an order through a revision order with no order lines or by resending the order with Fulfillment Mode = Cancel. OM is expected to honor the two alternatives for canceling an order, provided no order lines reached the point of no return.</p> <p>When used on billing EBS, Fulfillment Mode has a different meaning. It determines the type of Billing request: Initiate or Fulfill.</p>	<CommsOrder>EBO/FulfillmentModeCode
Customer Class	Identifies type of customer: Residential, Business, and so on	Residential Business	No	None		<CommsOrder>EBO/CustomerPartyReference/CustomerPartyAccountTypeCode
Organization ID	Identifies the organization/LOB generating the order. No cross-reference exists.	NA	No	None	No cross-reference. OM should use the application-specific ID if needed in any of the rules; provided that Organization IDs are unique and synchronized across all order capture systems.	<CommsOrder>EBO/BusinessUnitReference/BusinessUnitIdentification/ID
Sales Channel	Identifies the sales channel.	NA	No	None		<CommsOrder>EBO/SalesChannelCode
Job ID	A string or number that uniquely identifies the job to orchestration.	NA	No	None	Track orders that belong to a bulk or batch job.	<CommsOrder>EBO/ProcessingNumber

Functional Attribute Name	Attribute Usage Semantics	Seeded Values	Asset-able	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows: <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Sequence in Job	A number that identifies the order sequence within the job.	NA	No	None		<CommsOrder>EBO/ProcessingSequenceNumber
Job Type	Identifies the type of job. This information identifies the threshold for creating a consolidated SR for Bulk or Batch Orders. This value is optional for orders whose Job Cardinality is 1. By default, this value is HETROGENEOUS.	Heterogeneous, homogeneous, third-party homogeneous, third-party heterogeneous, correlated	No	None		<CommsOrder>EBO/ProcessingTypeCode
Job Cardinality	Indicates the total number of orders within the job.	NA	No	None		<CommsOrder>EBO/ProcessingQuantity
Parent Order ID	Order ID of another order that indicates the fulfillment for this order will not start before the parent order fulfillment completes.	NA	No	None	This attribute applies to explicit order-to-order dependencies and is not limited to follow-on orders. For example, in a B2B scenario, a large order can be divided into a number of smaller orders, with one order acting as the root order for all other orders and the remainder of the orders chained using the parent order ID attribute.	<CommsOrder>EBO/Parent<CommsOrder>Reference/<CommsOrder>Identification/BusinessComponentID

Functional Attribute Name	Attribute Usage Semantics	Seeded Values	Asset-able	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows: <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Fulfillment Priority	Indicates relevant priority of order fulfillment across orders. A lower value indicates a higher priority. Accepts values 0 to 9 in accordance with JMS Queue support.	9,7,5,3	No	None	EBM value: Siebel value 9: Urgent. Used for expedited orders. 7: High. CSP determines its use. 5: Medium. CSP determines its use. 3: Low. Recommended for job orders. Notice that Oracle Advanced Queuing (AQ) and JMS priority values have the inverse order of precedence	<CommsOrder>EBO/FulfillmentPriorityCode
Order Type	Sometimes indirectly determines sales channel to drive compensation process.	Sales Order	No	None		<CommsOrder>EBO/TypeCode
Requested Delivery Date Time	Overall order level due date that provides the default due date at each line level. Can be overridden at each line.	NA	Yes	None		<CommsOrder>EBO/RequestedDeliveryDateTime
Fulfillment Status	Reports aggregate order fulfillment status.	In Progress, Failed, Canceled Complete	Yes	None	This is different from the Status attributed tracked within Siebel	<CommsOrder>EBO/Status/Code

Functional Attribute Name	Attribute Usage Semantics	Seeded Values	Asset-able	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows: <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Status Context	Provides details about the current status. The implementer configures this value.	NA	Yes	None	OM can use this to track the milestone causing the status change along with context details such as error message, cause for cancel, . One primary scenario that the Order Header / Status Context is populated: with revision orders that cancel Order Lines by dropping them from the revision and if the revision is rejected. In that case the orchestration system does not have a line on the revision order to provide fallout status and context. In such a case the header level status context is used to identify the base line the cause for the fallout.	<CommsOrder>EBO/Status/Description
Owner Account ID	Identifies the owner account.	NA	Yes	None	Cross-referenced.	<CommsOrder>EBO/Custom erPartyReference/CustomerP artyAccountIdentification/Busi nessComponentID
Owner Account Name	Identifies the Account Name. You can enter or derive this value from contact first name + last name of primary contact associated with the account.	NA	Yes	None	Required for network inventory tracking of service owner.	<CommsOrder>EBO/Custom erPartyReference/CustomerP artyAccountName
Owner Account Number	Identifies account number to customer.	NA	Yes	None		<CommsOrder>EBO/Custom erPartyReference/CustomerP artyAccountIdentification/ID

Functional Attribute Name	Attribute Usage Semantics	Seeded Values	Asset-able	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows: <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Account Contact ID	Foreign key to contact record that holds personal and contact details of the customer/company representative who is placing the order and is the contact person for anything related to the order process.	NA	Yes	None		<CommsOrder>EBO/Custom erPartyReference/CustomerP artyAccountContactIdentificat ion/BusinessComponentID
Account Contact Address (component)	Identifies the address used to communicate with the Contact ID.	NA	Yes	None		<CommsOrder>EBO/Custom erPartyReference/CustomerP artyAccountContactAddressC ommunication/AddressComm unication/Address
Project ID	Identifies project record if the order to be delivered is part of a project that contains related orders. Foreign key reference. No cross-reference.	NA	Yes	None	No cross-reference for 2.4.	<CommsOrder>EBO/Project Reference/ProjectIdentificatio n/ID

Functional Attribute Name	Attribute Usage Semantics	Seeded Values	Asset-able	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows: <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Fulfillment System Type	For the Get Target Fulfillment Provider utility service, determines the logical identifier for appropriate target system instance among those serving this Fulfillment System Type.	NA	No	None		FulfillmentOrderEBO/FulfillmentSystemTypeCode
Target Instance	For the Get Target Fulfillment Provider utility service returns the logical identifier for appropriate target system instance among those serving this Fulfillment System Type.	NA	No	None		FulfillmentOrderEBO/FulfillmentTargetSystemID

Functional Attribute Name	Attribute Usage Semantics	Seeded Values	Asset-able	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows: <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Order Changed Indicator	OM sets this attribute to Yes if the order changed significantly such that CRM should make a copy of the customer order to preserve the customer intent before updating the working version of the order.	True, False	No	None	Allows Siebel to make a copy of the order if the order changed to the extent that the customer's intent is compromised.	<CommsOrder>EBO/OrderChangedIndicator
Sales Representative ID	CRM User ID that identifies the sales representative who entered the order.	NA	No	None	No cross-reference. Use the application ID.	<CommsOrder>EBO/SalespersonPartyReference/PartyIdentification/ID
Owner Account Contact (multiple fields)	Identifies if the address is used to communicate with the contact ID. Includes these fields: First Name, Last Name, Phone Number, and Email.	NA	No	None		<CommsOrder>EBO/Custom erPartyReference/CustomerP artyAccountContact/FirstNam e <CommsOrder>EBO/Custom erPartyReference/CustomerP artyAccountContact/LastNam e <CommsOrder>EBO/Custom erPartyReference/CustomerP artyAccountContactPhoneCo mmunication/PhoneCommuni cation/CompleteNumber <CommsOrder>EBO/Custom erPartyReference/CustomerP artyAccountContactEmailCo mmunication/EmailCommuni cation/

Communications Orders - Order Line Component Attributes

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Line ID	Uniquely identifies the order line item across orders and order revisions. Automatically generated.	NA	No	None	Cross-referenced. Produces a unique identifier for all Order Lines, including revision Order Lines.	<CommsOrder>EBO/<CommsOrder>Line/Identification/BusinessComponentID
Base Line ID	References base order line revised by this order line.	NA	No	None	Uses a cross-reference.	<CommsOrder>EBO/<CommsOrder>Line/Original<CommsOrder>LineReference/<CommsOrder>LineIdentification/BusinessComponentID
Asset Integration ID	Uniquely identifies an instance of a product that was or is being purchased.	NA	Yes	AIA2.0	Cross-referenced CRM populates the Asset Integration ID on all Order Lines, regardless of the Assetable state on the subject of the Order Line or whether the Order Line is for a new or existing service. A revision should never change the Asset Integration ID. When a product is dropped as part of one product hierarchy (CP or Promotion) and then added through another product hierarchy (CP or promotion), the Asset Integration ID for the two line items are different, although for the same product.	<CommsOrder>Line/InstalledProductReference/InstalledProductIdentification/BusinessComponentID

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Line Number	Identifies the line with respect to its position in the line item tree.	NA	No	None	Line number establishes the parent child relationship between Order Lines of the same order, but it may vary across revisions. Therefore, do not rely on it for matching Order Lines across revisions.	<CommsOrder>EBO/<CommsOrder>Line/Identification/ID
Parent Line	References parent order line in the line items tree instantiated as per the product model definition. Points to itself if the item does not have an associated parent item.	NA	No	None		<CommsOrder>EBO/<CommsOrder>Line/Parent<CommsOrder>LineIdentification/BusinessComponentID
Root Line	References the root order line in the line item tree instantiated as per the product model definition. Points to itself if the item is a root item itself.	NA	No	None		<CommsOrder>EBO/<CommsOrder>Line/RootParent<CommsOrder>LineIdentification/BusinessComponentID
Related Line ID	Links one-time charges to Suspend/Resume order lines.	NA	No	None		<CommsOrder>EBO/<CommsOrder>Line/Related<CommsOrder>LineIdentification/BusinessComponentID
Charge Parent Line ID	BRM adaptors use to relate one-time charges to base line ID.	NA	No	None		<CommsOrder>EBO/<CommsOrder>Line/ChargeParentLineIdentification/BusinessComponentID

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Related Asset Integration ID	Links Move-Add to Move-Delete line items.	NA	No	None		<CommsOrder>EBO/<CommsOrder>Line/InstalledProductReference/PriorInstalledProductIdentification/BusinessComponentID
Depends On Line ID	Indicates order line item ID of a previous order line item that is changed by this order. Follow-on orders use this value to capture dependencies of the order line items in the follow-on order-to-order line items of original orders.	NA	No	None	Cross-referenced.	<CommsOrder>EBO/<CommsOrder>Line/Depending<CommsOrder>LineReference/<CommsOrder>LineIdentification/BusinessComponentID
Depends On Order ID	Identifies order ID of an in-flight order, which is the basis for this follow-on order line item.	NA	No	None	Cross-referenced.	<CommsOrder>EBO/<CommsOrder>Line/Depending<CommsOrder>Reference/<CommsOrder>Identification/BusinessComponentID
Promotion Line ID	References an order line that represents the promotion/marketing offer under which the order line is being purchased.	NA	No	AIA2.0		<CommsOrder>EBO/<CommsOrder>Line/Promotion<CommsOrder>LineReference/Promotion<CommsOrder>LineIdentification/Identification/BusinessComponentID

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Asset-able	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Promotion Asset Integration ID	References an asset that represents the promotion/marketing offer under which the order line is being purchased.	NA	Yes	AIA2.0		<CommsOrder>EBO/<CommsOrder>Line/Promotion<CommsOrder>LineReference/InstalledProductReference/InstalledProductIdentification/BusinessComponentID
Product ID	References product record based on which order line is instantiated. Foreign key reference.	NA	Yes	None		<CommsOrder>EBO/<CommsOrder>Line/ItemReference/Identification/BusinessComponentID
Quantity	Identifies the quantity of the item requested by a customer. Default is 1.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/OrderQuantity
Action Code	Specify action required to meet customer request	None, Add, Update, Suspend Resume, Delete, Move-Add, Move-Delete	No	None		<CommsOrder>EBO/<CommsOrder>Line/ServiceActionCode

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Asset-able	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Requested Delivery Date Time	When Null, the requested date for delivery of the goods or service is ASAP; otherwise, it is the specified date. This date is not guaranteed. Typically, it is a future date; if it is a past date, then the default behavior is the same as a Null value.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/RequestedDeliveryDateTime
Deliver To Address	Address record that represents the delivery/service installation address.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/ServiceAddress/Address
Usage Start Date	Determines the date when usage events should start being rated. The value for this attribute is populated by CRM, OM Fulfillment flows, or kept to Null for BRM default to the current date.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/ServiceUsageStartDate

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Cycle State Date	Determines the date when cycle charges should start being billed. The value for this attribute is populated by CRM, OM Fulfillment flows, or kept to Null for BRM default to the current date as per previous patterns.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/CycleStartDate
Purchase Date	Determines the date when one-time purchase charges should be billed. The value for this attribute is populated by CRM, OM Fulfillment flows, or kept to Null for BRM default to current date as per previously mentioned patterns.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/PurchaseDate
Service Start Date	Indicates effective start date of service.	NA	Yes	None	Initially computed by Siebel based on Due Date and then updated by Order Management based on Actual Delivery Date.	<CommsOrder>EBO/<CommsOrder>Line/EffectiveTimePeriod/StartDateTime

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Earliest Delivery Date	Identifies the date when the work associated to the order can start. Typically used for fulfillment actions that require customer presence such as in cases customer need to be available to install service or deliver shipment	NA	No	None		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/EarliestDeliveryDate/DateTime
Service End Date	Indicates the effective end date of service. Applies to services with a specified duration.	NA	Yes	None	Initially computed in Siebel and then updated by Order Management. Update is sent to Siebel.	<CommsOrder>EBO/<CommsOrder>Line/EffectiveTimePeriod/EndTime/DateTime

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Actual Delivery Date Time	Determines the date when the purchased product or service is considered available to the customer by the CSP. This date may be when physical goods are shipped, delivered, or their receipt is acknowledged. For service-based products, the service is activated on this date. This date is computed in the OM Fulfillment flow as per previous patterns.	NA	Yes	None	BRM does not allow for starting any charges before the Purchase Date; therefore, the ABCS for BRM will always override the Purchase Date if it is later than any of the Cycle or Usage start dates. OM should facilitate calculation of Order Line level Actual Delivery Date as well as Order Line attributes for billing Usage Start Date, Cycle Start Date, and Purchase Date.	<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/ActualDeliveryDateTime
Expected Delivery Date Time	Indicates the due date expected by the system as a result of Design and Assign. The default is the Order Due Date when the order is created by CRM.	NA	No	None	Computed by OM based on preconfigured time estimates on fulfillment actions. Used by OM to communicate to CRM changes to expected delivery date of specific Order Lines.	<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/ExpectedDeliveryDateTime
Fulfillment Status	Updates orchestration and CRM as to the current status of order line fulfillment at a high level.	In Progress, Failed, Canceled Complete	Yes	None	Additional values can be added as an extension	<CommsOrder>EBO/<CommsOrder>Line/Status/Code

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Milestone	Fulfillment passes the last reached milestone into this field.	NA	No	None		<CommsOrder>/<CommsOrder>Line/MilestoneCode
Status Context	Provides details about the current status of the order line. The implementer configures this value.	NA	Yes	None	<p>OM could include the reached milestone (from the fulfillment system, the cause for the status update that is necessary because of dynamic nature of fulfillment plan) and a textual string for context per current status as follows (canonical Status / status context):</p> <p>Submitted / NA In Progress / <milestone>: context text Failed / <milestone>: reason text Canceled / <milestone>: reason text Complete / NA</p> <p>In Progress: Context Text could be used to indicate any of the following among others:</p> <ul style="list-style-type: none"> • Requires customer interaction • Delivery is expected to be delayed 	<CommsOrder>EBO/<CommsOrder>Line/Status/Description
Point-of-no-return	Determines if Siebel should allow order line revisions to be submitted.	Not yet, Hard	No	None	<p>OM Fulfillment flows allow configuration of setting a hard PONR when a condition is met for a particular service. When a hard PONR is reached for an Order Line in OM, a status update is issued to reflect the same in CRM.</p> <p>Additional values such as SOFT can be added as an extension.</p>	<CommsOrder>EBO/<CommsOrder>Line/RevisionPermissibleCode

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Billing Account	References an account record that represents the bill payer or the branch of a company responsible for bill payment. This value may be a customer account or an account from the account hierarchy.	NA	Yes	AIA2.0		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/BillToPartyReference/CustomerPartyAccountIdentification/BusinessComponentID
Billing Profile	References the billing profile record that holds the customer's billing/payment preferences. This value may be associated to the customer account or to a separate billing account.	NA	Yes	AIA2.0		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/BillToPartyReference/BillingProfileReference/BillingProfileIdentification/BusinessComponentID
Payment Profile	Identifies the Payment Profile.	NA	No	None		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/BillToPartyReference/BillingProfileReference/PaymentProfileReference/PaymentProfileIdentification/BusinessComponentID

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Service Account	References an account record that represents a service user or the branch of the company where service is installed. This value may be customer account or an account from the account hierarchy.	NA	Yes	AIA2.0		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/OwnerPartyReference/CustomerPartyAccountIdentification/BusinessComponentID
Owner Contact	Represents a contact of the customer account or service account who should be contacted during fulfillment of the line if required.	NA	Yes	None		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/OwnerPartyReference/CustomerPartyAccountContactIdentification/BusinessComponentID
Shipping Contact	Represents a contact of the customer account or service account who should be contacted for shipping purposes.	NA	Yes	None		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/ShipToPartyReference/CustomerPartyAccountContactIdentification/BusinessComponentID

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Node	Alphanumerically references the root order line that corresponds to access at site A of a connection. This value is relevant for network ordering only.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>LineSpecificationGroup/SpecificationGroup[./name="ExtensibleAttributes"]/Specification[./name="Node"]/ValueText
To Node	Alphanumerically references the root order line that corresponds to access at site B of a connection. This value is relevant for network ordering only.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>LineSpecificationGroup/SpecificationGroup[./name="ExtensibleAttributes"]/Specification[./name="ToNode"]/ValueText
Network ID	Unique compound product number that represents the virtual network ID. Relevant for network orders. Provided by default from the order number and cascaded to network connection items.	NA	Yes	AIA2.4	Identifies which Access and Nodes belong to the same network. This information may be of value to decomposition.	<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>LineSpecificationGroup/SpecificationGroup[./name="ExtensibleAttributes"]/Specification[./name="NetworkID"]/ValueText

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Asset-able	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Port Number	Identifies the port number allocated to the access circuit connected to provide (starting) edge router during the fulfillment process.	NA	Yes	AIA2.4	For new services, port number comes back from Network Inventory through provisioning.	<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>LineSpecificationGroup/SpecificationGroup[./name="ExtensibleAttributes"]/Specification[./name="PortNumber"]/ValueText
To Port Number	Identifies the port number allocated to the access circuit connected to provide (ending) edge router during the fulfillment process.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>LineSpecificationGroup/SpecificationGroup[./name="ExtensibleAttributes"]/Specification[./name="ToPortNumber"]/ValueText
Service Address Prefix	Identifies the area code/NPA for the access circuits on starting or two ends of the connection.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>LineSpecificationGroup/SpecificationGroup[./name="ExtensibleAttributes"]/Specification[./name="ServiceAddressPrefix"]/ValueText
To Service Address Prefix	Identifies the area code/NPA for the access circuits on the end of the connection.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>LineSpecificationGroup/SpecificationGroup[./name="ExtensibleAttributes"]/Specification[./name="ToServiceAddressPrefix"]/ValueText
Access Circuit	Provides the Common Language Location Identification (CLLI) for the access circuit on two sides or starting side of the connection.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>LineSpecificationGroup/SpecificationGroup[./name="ExtensibleAttributes"]/Specification[./name="AccessCircuit"]/ValueText

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Asset-able	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
To Access Circuit	Provides the CLLI for the access circuit on ending side of the connection.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>LineSpecificationGroup/SpecificationGroup[./name="ExtensibleAttributes"]/Specification[./name="ToAccessCircuit"]/ValueText
To Service Account ID	Identifies the Service Account ID associated with the end side of a network.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>LineSpecificationGroup/SpecificationGroup[./name="ExtensibleAttributes"]/Specification[./name="ToServiceAccountID"]/ValueText
From Service Address ID	Identifies the Service Address ID for the starting point of a network.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>LineSpecificationGroup/SpecificationGroup[./name="ExtensibleAttributes"]/Specification[./name="FromServiceAddressID"]/ValueText
To Service Address ID	Identifies the Service Address ID for the ending point of a network.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>LineSpecificationGroup/SpecificationGroup[./name="ExtensibleAttributes"]/Specification[./name="ToServiceAddressID"]/ValueText
To Service Point ID	References a dummy asset record that represents the access point to which the starting side of a network service will be connected on the customer's premises.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>LineSpecificationGroup/SpecificationGroup[./name="ExtensibleAttributes"]/Specification[./name="ToServicePointID"]/ValueText

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Asset-able	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Service Point	References a dummy asset record that represents the access point to which this service will be connected on the customer's premises. For example, NTE for PSTN, Set top box for Broadband/Cable service.	NA	Yes	AIA2.4	Expected to be mastered in network inventory and loaded in Siebel in batch.	<CommsOrder>EBO/<CommsOrder>Line/ServicePointCode
Promotion Description	Provides short description that will appear on the invoice.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/ItemReference/Description This is Promotion Description used for display purposes on customer invoice
Service ID	Identifies the product/service instance as recognized across BSS and OSS applications. Most significantly this is the ID used to correlate rating records to customer accounts.	NA	Yes	AIA2.0	Can be populated part of order capture process or during fulfillment, but before interface an order to billing.	<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/<CommsOrder>ItemInstance/Identification/ID
Balance Bundle Identification	Identifies the Balance Bundle to which a service instance belongs.	NA			Not Used by AIA for Communications	<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/BalanceBundleIdentification/BusinessComponentID

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Line Description	Provides additional description for an order line. For example, to indicate that a charge is being applied for a penalty.	NA	No	None	Not used by AIA for Communications	<CommsOrder>EBO/<CommsOrder>Line/Description
Service Length	Indicates requested service length in Service Length Unit of Measure.	NA	Yes	Yes		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/ServiceTimePeriod/Duration
Service Length Unit of Measure	Indicates the service length unit of measure.	NA	Yes	Yes		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/ServiceTimePeriod/Duration
Fulfillment Mode	Designates compensation operations for Initiate Billing. May be used in the future to provide explicit revision operations at the line level.	DO, NOOP, REDO, UNDO	No	None		<CommsOrder>EBO/<CommsOrder>Line/FulfillmentMode Code
Product Name	Provides the name of the product.	NA				<CommsOrder>EBO/<CommsOrder>Line/ItemReference/Name

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Composition Type	<p>Determines product composition granularity:</p> <p>PartialItem is an order line that constitutes an indivisible element of another order line. This type typically denotes a piece of a product.</p> <p>WholeItem is an order line that represents a self-contained subject. A WholeItem may be represented by a single line item or a number of PartialItem order lines.</p> <p>May also assume no value signified by a Null value or absence of value.</p>	<no value> for NULL, PARTIAL ITEM, WHOLE ITEM	No	None	Consult Oracle on usage.	<CommsOrder>EBO/<CommsOrder>Line/ItemReference/FulfillmentCompositionTypeCode
Product Type	Classifies products into Products, Discounts, Bundles, Promotion (Offer), and so on.	Product, Offer, Bundle	No	None	Used part of fulfillment to determine the order lines Subject Type, which drives the mapping to Product Specifications.	<CommsOrder>EBO/<CommsOrder>Line/ItemReference/TypeCode

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Billing Type	Classifies products for Billing into Service Bundles, Subscriptions, Items, Discounts, and Special Ratings.	Service Bundle, Subscription, Item, Discount, Special Rating	No	None	Used in conjunction with Product Type.	<CommsOrder>EBO/<CommsOrder>Line/ItemReference/ClassificationCode [listID="BillingProductTypeCode"]
Billing Service Type	Specifies the service type so that when a corresponding product is created in Billing, it is associated to the specified service.	NA	No	None		<CommsOrder>EBO/<CommsOrder>Line/ItemReference/ClassificationCode [listID="PermittedTypeCode"]
Service Flag	Indicates the product of a service or non-service, for example, physical goods.	TRUE, FALSE	No	None	Used in conjunction with Product Type and may be used to parameterize fulfillment flows.	<CommsOrder>EBO/<CommsOrder>Line/ItemReference/ServiceIndicator
Vendor	Identifies the vendor supplying the product when the product is supplied by a third-party.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/ItemReference/SupplierPartyReference/PartyIdentification/ID
Vendor Part Number	Identifies the product part number to the vendor.	NA	Yes	AIA2.4		<CommsOrder>EBO/<CommsOrder>Line/ItemReference/ItemIdentification/SupplierItemID
Fulfillment Item Code	Uniquely identifies the mapping of an Order Line Subject to a Product Specification.	1) Null 2) A unique code that identifies the Product Spec to OM	No	None		<CommsOrder>EBO/<CommsOrder>Line/ItemReference/ClassificationCode [listID = "FulfillmentItemCode"]

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Item Class Name	Determines business classification of a product.	NA	No	None		<CommsOrder>EBO/<CommsOrder>Line/ItemReference/PrimaryClassificationCode
Success Dependency	Declares if all order lines of a bundle or offer must fulfill successfully or else the whole bundle or offer fails (all or none).	Default, All Or None	No	None		<CommsOrder>EBO/<CommsOrder>Line/ItemReference/FulfillmentSuccessCode
Start Billing on First Usage	When set to Yes by CRM or OSM, passes the request along to BRM. In this case, Usage Start Date, Cycle Start Date, and Purchase Date should have no effect.	True, False	No	None	Not yet supported by integration.	<CommsOrder>EBO/<CommsOrder>Line/StartBillingOnFirstServiceUsageIndicator We have added BillingStartCode to ItemReference, if this requirement is at the item/itemReference level and not line level then BillingStartCode from ItemReference should be used.
Smart Part Number	Automatically generated based on a predefined scheme. Mainly, drives dynamic product configuration/ pricing rules in CRM. The billing system may use it to dynamically derive a price/discount value.	NA	Yes	None		<CommsOrder>EBO/<CommsOrder>Line/ItemReference/AlternateObjectKey [ContextID=SmartPartNumber]

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Asset-able	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Network Product Flag	Indicates if this is a network product, which helps determine which user-defined attributes to expect.	True, False	No	None		<CommsOrder>EBO/<CommsOrder>Line/ItemReference/NetworkIndicator
Network Element Type	Indicates if this network product represents a node, a connection, or a network.	NA	No	None		<CommsOrder>EBO/<CommsOrder>Line/ItemReference/NetworkItemTypeCode
Charge Frequency Code	Indicates charge frequency unit of measure, for example, monthly, quarterly, yearly.	NA				<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/<CommsOrder>ScheduleCharge/Charge/ChargeFrequencyCode
List Price Type	Identifies price type.	One-Time, Recurring, Usage	No	None		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/<CommsOrder>ScheduleCharge/Charge/TypeCode
List Price	Identifies base price of the item.	NA	Yes	None		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/<CommsOrder>ScheduleCharge/Charge/UnitListPrice/Amount
Sale Price Type	Identifies price type.	One-Time, Recurring, Usage	No	None		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/<CommsOrder>ScheduleCharge/Charge/TypeCode
Sale Price	Identifies net price of the item.	NA	Yes	AIA2.0		<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>Schedule/<CommsOrder>ScheduleCharge/Charge/UnitSalePrice/Amount

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
Pricing Commit Type	Indicates whether the pricing is Committed or Dynamic.	Common/Siebel values are true/Dynamic, false/Committed.	Yes	AIA2.4		<CommsOrder>/<CommsOrder>Line/<CommsOrder>Schedule/<CommsOrder>ScheduleCharge/Charge/DynamicPricingIndicator
Dynamic Discount Method	Indicates whether the discount is of type amount or percent.	Amount, Percent	Yes	AIA2.4		<CommsOrder>/<CommsOrder>Line/<CommsOrder>Schedule/<CommsOrder>ScheduleCharge/Charge/DiscountMethodCode
Discount Percent	Indicates the percent by which the list price is discounted.	NA	Yes	AIA2.4		<CommsOrder>/<CommsOrder>Line/<CommsOrder>Schedule/<CommsOrder>ScheduleCharge/Charge/DiscountPercent
Discount Amount	Indicates the amount by which the list price is discounted	NA	Yes	AIA2.4		<CommsOrder>/<CommsOrder>Line/<CommsOrder>Schedule/<CommsOrder>ScheduleCharge/Charge/DiscountAmount
Member [0..N]	Represents a member of a list by their phone number.	NA	No	None	Used for capturing membership to friends and family plans.	<CommsOrder>EBO/<CommsOrder>Line/<CommsOrder>LineSpecificationGroup/SpecificationGroup[.name="ExtensibleAttributes"]/Specification[.name="SpecialRating"]/ValueText [0..N]
User Defined Attributes	Indicates attribute is common across all Specification components.	NA	Yes	None	UDA Name	<CommsOrder>/<CommsOrder>Line/ItemReference/SpecificationGroup[name="ExtensibleAttributes"]/Specification/Name
User Defined Attributes	Indicates attribute is common across all Specification components.	Add, Update, Delete	Yes	None	UDA Action Code (Expected to change to a Service Action Code element to allow additional value NONE.)	<CommsOrder>/<CommsOrder>Line/ItemReference/SpecificationGroup[name="ExtensibleAttributes"]/Specification[name="<OrderLine.XA.Attribute>"]/@actionCode
User Defined Attributes	Indicates attribute is common across all Specification components.	NA	Yes	has Previous LIC Value	UDA language-independent code Value	<CommsOrder>/<CommsOrder>Line/ItemReference/SpecificationGroup[name="ExtensibleAttributes"]/Specification[name="<OrderLine.XA.Attribute>"]/Value

Functional Attribute Name	Attribute Usage Semantics	Seeded Values and Value Type	Assetable	Prior Value Available	Remarks	EBO Structure XPath. Depends on context as follows <CommsOrder> variable for SalesOrder, FulfillmentOrder, or ProvisioningOrder
User Defined Attributes	Indicates attribute is common across all Specification components.	String, Date, Number	Yes	None	UDA Data Type	<CommsOrder>/Prior<CommsOrder>/<CommsOrder>Line/ItemReference/SpecificationGroup[name="ExtensibleAttributes"]/Specification[name="<OrderLine.XA.Attribute>"]/Data TypeCode
User Defined Attributes	Indicates attribute is common across all Specification components.	NA	Yes	None	UDA language-independent code Prior Value	<CommsOrder>/Prior<CommsOrder>/<CommsOrder>Line/ItemReference/SpecificationGroup[name="ExtensibleAttributes"]/Specification[name="<OrderLine.XA.Attribute>"]/Value

Appendix B: 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]:[siebel.http.port]/eai_enu/start.swe?SWEExtSource=SecureWebService&
SWEExtCmd=Execute&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.

Index

- Billing Fulfillment Design Considerations, 27
- Business Process Flows, 17
- CommunicationsBillingEBSV1, 68
- CommunicationsClassificationEBSV1, 46
- CommunicationsFulfillmentOrderEBSV1, 83
- CommunicationsProvisioningOrderEBSV1, 82
- CommunicationsSalesOrderEBSV2, 75
- CommunicationsSpecificationEBSV1, 47
- CommunicationsSpecificationValueSetEBSV1, 47
- Configuring Multiple BRM Instances, 114
- Configuring Order Fallout
 - adding routing rules, 120
- Configuring the Order to Activate PIP, 114
 - configuring order fallout management, 115
 - guaranteed delivery, 106
 - handling errors, 113
 - install sequence and configurations for using fallout, 119
 - overview, 103
 - resequencer, 107
 - setting up Fusion middleware, 105
 - setting up OSM, 104
 - setting up Siebel CRM, 103
 - viewing EBO Implementation Maps, 113
 - working with cross-references, 112
 - working with DVMs, 109
- CreateFaultNotificationLFCommsJMSSConsumer, 100
- CreateOrderFalloutNotificationOSMCFSCommsJMSSConsumer, 99
- CreateOrderFalloutNotificationOSMCFSCommsJMSSProducer, 100
- CreateOrderFalloutNotificationOSMCFSCommsProvImpl, 99
- CreateTroubleTicketOSMCFSCommsJMSSConsumer, 100
- CreateTroubleTicketRespOSMCFSCommsJMSSProducer, 100
- Cross-References, 112
- Data Requirements, 30
- Deliver Customer Order Sub-Flow, 19
- Dictionary, 129
- DVMs, 109
- EBO Implementation Maps, 113
- Fallout Management Design Considerations, 30
- Follow-On Orders Design Considerations, 29
- Future-Dated Orders Design Considerations, 28
- Mapping Design Considerations, 22
- Order Capture Sub-Flow, 18
- Order Fallout Management
 - assumptions and constraints, 96
 - configuring fallout using Order to Bill, 101
 - creating trouble tickets in OSM, 91
 - errors, 90
 - fallout d services, 101
 - Industry AIA components, 97
 - integration services, 98
 - overview, 85
 - Siebel CRM interfaces, 97
- Order Lifecycle Management

- follow-on order considerations, 55
- future-dated order considerations, 56
- Industry AIA components, 61
- JMS message construction, 59
- new order considerations, 54
- order priorities, 57
- overview, 49
- revision order considerations, 55
- updating order status considerations, 57
- Order Priorities Design Considerations, 29
- Order to Activate
 - overview, 13
- Order Update Design Considerations, 24
- Orders Dictionary, 129
- Overview, 13
- PLM Class Synchronization
 - assumptions, 44
 - design-time setups in SCE Studio, 33
 - effectivity during design-time, 42
 - Industry AIA components, 44
 - Integration services, 45
 - overview, 33
 - product classes in Siebel, 33
 - query product classes and attributes, 34
 - Siebel CRM interfaces, 44
- Process Integration for Order Provisioning
 - assumptions and constraints, 80
 - creating provisioning orders, 77
 - Industry AIA components, 81
 - integration services, 81
 - overview, 77
 - updating fulfillment orders, 78
- Process Integrations for Billing
 - integration services, 67
 - interfacing customer order actions, 66
 - overview, 63
 - synchronizing customer accounts, 63
- Process Integrations for Siebel CRM
 - integration services, 74
 - overview, 69
 - Siebel CRM interfaces, 74
 - submitting orders, 70
 - updating statuses, 72
- ProcessFulfillmentOrderBillingAccountListOSMCFSCommsJMSConsumer, 67
- ProcessFulfillmentOrderBillingAccountListResponseOSMCFSCommsJMSProducer, 68
- ProcessFulfillmentOrderBillingOSMCFSCommsJMSConsumer, 68
- ProcessFulfillmentOrderBillingResponseOSMCFSCommsJMSProducer, 68
- ProcessFulfillmentOrderUpdateOSMCFSCommsJMSProducer, 83
- ProcessFulfillmentOrderUpdateOSMPROVCommsJMSConsumer, 82
- ProcessProvisioningOrderOSMCFSCommsJMSConsumer, 82
- ProcessProvisioningOrderOSMPROVCommsJMSProducer, 83
- ProcessSalesOrderFulfillmentOSMCFSCommsJMSProducer, 75
- ProcessSalesOrderFulfillmentSiebelCommsJMSConsumer, 75
- ProcessSalesOrderFulfillmentSiebelCommsReqABCSImpl, 75
- Product Definition and Mapping Design Considerations, 22
- Qualify Customer Order Sub-Flow, 21
- QueryClassificationListSiebelCommsProvABCSImpl, 46
- QueryProdClassAndAttributesSCEReqCommsABCSImpl, 46
- QuerySpecificationListSiebelCommsProvABCSImpl, 47

QuerySpecificationValueSetListSiebelCommsProvABCSImpl, 48	UpdateSalesOrderOSMCFSCommsJMSCOnsumer, 76
Revision Orders Design Considerations, 25	UpdateSalesOrderSiebelCommsProvABCSImpl, 76
Single-Phase Billing, 28	UpdateTroubleTicketOSMCFSCommsJMSCOnsumer, 100
Solution Assumptions and Constraints, 30	UpdateTroubleTicketSiebelCommsProvABCSImpl, 99
Two-Phase Billing, 27	
Typical Topology, 16	